Fast Algorithms for Spanwise Periodic Incompressible External Flows: From Simulation to Analysis

Thesis by
Wei Hou

In Partial Fulfillment of the Requirements for the
Degree of
Doctor of Philosophy

CALIFORNIA INSTITUTE OF TECHNOLOGY
Pasadena, California

2025
Defended December 12, 2024

Wei Hou
ORCID: 0000-0001-8023-6395

ACKNOWLEDGEMENTS

First, I would like to thank my advisor Tim Colonius for being a wonderful mentor. I came to Caltech as a naive college graduate. Tim showed me how to, in his words, "do scholarly research" and taught me the importance of communicating my ideas. Working with Tim is like a challenging but rewarding journey: Only at the end of the voyage can you finally begin to realize and appreciate how far you have come, and I could never have traveled this far without Tim's guidance.

I would also like to thank my committee members, Guillaume Blanquart, Jane Bae, and John Sader, for the valuable feedback and comments.

During my years at Caltech, I really appreciate all the people in the Colonius group: Omar, Michael, Vishal, Rahul, Caroline, Soto, Jose, Ethan, Hyeoksu, Franz, Michael, Chris, Ke, Liam, Jean, Marcus, Ben, Ethan, Akhil, Isabel, Shunxiang, Lennart, Spencer, and Mauro. My life at Caltech is so much better with you guys. Especially, I would like to thank Liam for helping me with Richardson through all the outages and upgrades. Now that the responsibility falls on Hyeoksu, I am sure that Richardson is in good hands. Also, I would like to thank Ke for helping me understand IBLGF-AMR and appreciate its beauty and complexity. And Caroline, thank you for keeping IBLGF-AMR alive. I cannot wait to see what the future holds.

In addition, I would like to thank my undergraduate advisor and lifetime mentor, Prof. Jeff Eldredge. You set me on the path of doing research and helped me countless times throughout my research career. For this, I am forever grateful.

Most importantly, I would like to thank my family, especially my parents, for always supporting and encouraging me.

Lastly, I want to thank anyone who is reading this thesis. I hope you find it useful and, more importantly, enjoy the time you spent reading it.

ABSTRACT

External flows over spanwise-homogeneous geometries are ubiquitous in science and engineering applications (Mittal and Balachandar, 1995; Mittal and Balachandar, 1997; Dong and Karniadakis, 2005; Lehmkuhl et al., 2013). In this thesis, we propose algorithms to simulate and analyze these flows using the lattice Green's function (LGF) approach. The LGF is the analytical inverse of a discrete elliptic operator that automatically incorporates exact far-field boundary conditions and minimizes computational expense by allowing snug computational regions encompassing only vortical flow regions. By combining LGFs with adaptive mesh refinement (AMR) and immersed boundary (IB) methods, we present two numerical algorithms specially designed for spanwise periodic incompressible external flows: one to directly solve the nonlinear equations of motion and one to compute stability and resolvent analyses.

For these algorithms, the LGFs of the screened Poisson equation must be computed at runtime. To enable efficient flow simulation and analysis algorithms, we propose a fast numerical algorithm to tabulate these LGFs. We derive convergence results for the algorithms and show that they are orders of magnitude faster than existing algorithms. Armed with the LGF for the screened Poisson equation, we further develop algorithms to solve the Navier-Stokes equations and associated linearized eigenvalue problems.

We present two applications of these algorithms. We perform simulations to validate the starting vortex theory proposed by Pullin and Sader (2021), and we perform stability analyses of flow past a rotating cylinder with a control cylinder in its wake.

PUBLISHED CONTENT AND CONTRIBUTIONS

Hou, Wei and Tim Colonius (Dec. 2024a). "An adaptive lattice Green's function method for external flows with two unbounded and one homogeneous directions". In: *Journal of Computational Physics* 519, p. 113370. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2024.113370. URL: http://dx.doi.org/10.1016/j.jcp.2024.113370.
W. H. participated in the conception of the project, derived and implemented the algorithm, performed numerical experiments, and wrote the manuscript.

– (2024b). "Fast and robust method for screened Poisson lattice Green's function using asymptotic expansion and Fast Fourier Transform". In: *arXiv preprint arXiv:2403.03076*. URL: https://arxiv.org/abs/2403.03076.
W. H. participated in the conception of the project, derived and implemented the algorithm, performed numerical experiments, and wrote the manuscript.

Sader, John E. et al. (2024). "The starting vortices generated by bodies with sharp and straight edges in a viscous fluid". In: *Journal of Fluid Mechanics* 992, A15. URL: https://doi.org/10.1017/jfm.2024.515.
W. H. implemented the numerical model and performed numerical experiments.

Hou, Wei and Tim Colonius (2023). "Three-dimensional stability and resolvent analysis of external flows over spanwise-homogeneous immersed bodies". In: *AIAA AVIATION 2023 Forum*, p. 3414. URL: https://arc.aiaa.org/doi/10.2514/6.2023-3414.
W. H. participated in the conception of the project, derived and implemented the algorithm, performed numerical experiments, and wrote the manuscript.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Number *Page*

LIST OF TABLES

Chapter 1

INTRODUCTION

In this thesis, we discuss algorithms designed for a specific type of fluid flow: spanwise periodic incompressible external flow. In its name, there are three main adjectives: spanwise periodic, incompressible, and external. Incompressibility is a straightforward word that does not need further introduction and is embedded within the governing equations. As such, I will discuss the remaining two words and their mathematical and physical implications in the subsequent paragraphs.

"External" means that the flow we are interested in is situated on an unbounded domain. Specifically, we have immersed bodies in some unbounded fluid domain. The flow is generated by the immersed body moving in the said fluid. This unbounded domain has two implications: first, the boundary condition is only posed at infinity; second, the flow is restricted only by the immersed body without the presence of any other boundaries, so flow structures can exist in a wide range of scales. These two characteristics pose their corresponding challenges. Traditionally, one can solve these problems using different numerical techniques such as unstructured meshes (Mittal and Balachandar, 1995; Borrell et al., 2011; Fischer, Lottes, and Tufo, 2007) and overset methods (Steger, Dougherty, and Benek, 1983; Tang, Jones, and Sotiropoulos, 2003). Although these methods are compatible with flexible nodal distribution and grid resolution, they suffer from the requirements of unbounded computational domains or posing non-trivial boundary conditions (Mittal and Balachandar, 1996; Lai and Peskin, 2000; Yu, 2021). Moreover, they are subject to remeshing when the immersed body moves or deforms. By combining lattice Green's function method (LGF), adaptive mesh refinement (AMR), and immersed boundary (IB) method, one can devise efficient numerical frameworks to simulate external flows (Liska and Colonius, 2017; Yu, 2021; Yu, Dorschner, and Colonius, 2022).

"Spanwise periodic" means that the fluid flow we are simulating and studying repeats itself in the spanwise direction. Spanwise periodicity is used to model the flow past an immersed body with infinite span and constant cross-section. Namely, an infinitely long cylinder with arbitrary cross-section. This spanwise periodic model stems from experimental observation (Williamson and Roshko, 1988; Williamson,

1996; Mittal and Balachandar, 1995; Mittal and Balachandar, 1997). As a result, one can leverage this property to devise efficient numerical algorithms for flow analysis and simulations. Indeed, based on different discretizations, the treatment for spanwise periodicity can vary. Posing spanwise periodic boundary conditions after discretizing the PDE can lead to a circulant matrix that is diagonalizable by means of discrete Fourier transform (Borrell et al., 2011). However, in this thesis, we take a different approach by discretizing the governing equations after the Fourier transform (Mittal and Balachandar, 1996). By doing so, we can variably truncate the Fourier series for further computational savings. In addition, we propose efficient numerical algorithms that can leverage LGF, AMR, and IB methods and are compatible with the spanwise periodicity nature.

By understanding the challenges and opportunities associated with the properties of spanwise periodic incompressible external flows, we created corresponding simulation and analysis algorithms that enable us to efficiently study this particular type of flow. Enabled by these algorithms, we studied various flows that have both theoretical and practical importance. Overall, this thesis can be divided into three components: novel algorithms for computing the LGF of the screened Poisson operator and their analysis, simulation and analysis algorithms based on the LGF techniques, and the study of various flow phenomena using the simulation and analysis algorithms.

1.1 Fast algorithm for computing LGF of the screened Poisson operator

Given an invertible discrete elliptical operator, its LGF is its inverse in free space. The LGF of the finite difference Poisson operator is well studied and has readily available asymptotic expansion formulae for efficient numerical evaluation (Duffin, 1953; Duffin and Shelly, 1958; Martinsson and Rodin, 2002). As a result, this LGF has been applied to efficiently solve many problems arising from discretizing partial differential equations (Liska and Colonius, 2014; Liska and Colonius, 2016; Liska and Colonius, 2017; Dorschner et al., 2020; Yu, 2021; Yu, Dorschner, and Colonius, 2022; Caprace, Gillis, and Chatelain, 2021; Balty, Chatelain, and Gillis, 2023). However, to simulate spanwise periodic external incompressible flows via Fourier expansion, instead of the Poisson equation, the screened Poisson equation arises. As such, we need the corresponding LGF to solve the pressure-Poisson equation in the Fourier expansion formulation. Yet, efficient algorithms to evaluate this LGF were not readily available. In Chapter 2, we introduce a set of new algorithms to efficiently evaluate the LGF of the screened Poisson operator. Correspondingly,

we present rigorous convergence bounds for these algorithms and computational complexity analysis in different scenarios. By comparing to existing methods to evaluate the LGF of the screened Poisson operator, we demonstrate that our methods show significant improvement in terms of both speed of robustness.

Although our motivation lies in solving numerical partial differential equations arising from fluid mechanics, the applications of this particular LGF extend to many other fields of science and engineering. We demonstrate this by showing two applications of our algorithms: solving a three-dimensional Poisson equation with one periodic dimension and computing the return probability of a two-dimensional random walk with killing (Lawler and Limic, 2010).

1.2 Fast flow simulation and analysis algorithms

Equipped with the fast algorithms for tabulating the LGFs of screened Poisson equations, we are ready to leverage the LGFs to create numerical algorithms to conduct simulations and flow analyses. In Chapter 3 and Chapter 4, we proposed two algorithms related to spanwise periodic external incompressible flows.

In Chapter 3, we present an algorithm that leverages the LGFs, AMR, and the immersed boundary (IB) method to simulate spanwise periodic external incompressible flows efficiently. In this algorithm, we Fourier-expand the velocity and pressure and time evolve the resulting Fourier coefficients of the velocity and pressure instead of time evolving velocity and pressure directly. Then we adapted an existing half explicit Runge-Kutta method by Yu (2021). Combining the Fourier coefficient formulation of the Navier-Stokes equations, AMR, LGF, and the IB method, we created a fast flow simulation algorithm tailored for spanwise periodic incompressible external flows. The algorithm is then verified and validated through a range of numerical simulations and tests.

With the fast numerical simulation algorithm, we hope to leverage its advantages to create a stability analysis and resolvent analysis algorithm. In Chapter 4, we combine the same components that enhance the efficiency of the flow solver, i.e., LGF, AMR, and the IB method, and devise a computational advantageous form of the discretized linearized Navier-Stokes equations (LNSE), which is the key to create efficient linear stability analysis and resolvent analysis. In addition, we leveraged the fast multipole method (FMM) to further enhance the computational efficiency. The resulting algorithm is validated by various numerical experiments involving both canonical and non-canonical examples of resolvent and stability analysis.

1.3 Applications of the simulation and analysis algorithms

With the aforementioned numerical algorithms, we can solve a wide range of interesting problems. In this part, we present three numerical studies we conducted using the above algorithms.

In Chapter 5, we first apply the two-dimensional version of the flow solver to study the early time behavior of the flow field generated by bodies with straight and sharp trailing edges that are translating and rotating in the viscous fluid. Examples of such geometries are flat plates and Joukowski airfoils. Using the inviscid flow model, Pullin and Sader (2021) showed that there are three types of early-time trailing edge vortex behaviors: a vortex sheet, a vortex roll-up, and a combination of these two. In addition, the behavior is purely determined by the rate of pitching and translating of the immersed body. In this chapter, we conducted a series of direct numerical simulations (DNS) at high Reynolds numbers to show that the theoretical results derived from the inviscid flow model manifest themselves in the Navier-Stokes equations where viscosity is present.

Also, in Chapter 6, we conducted a sequence of stability analysis of the flow past a rotating cylinder with a control cylinder in its wake. As demonstrated by Strykowski and Sreenivasan (1990), for the flow past a stationary cylinder, putting a control cylinder in the wake of the main cylinder suppresses the onset of vortex shedding. Meanwhile, as shown by Pralits, Giannetti, and Brandt (2013), if we consider the flow past a rotating cylinder, with no control cylinder in the wake, the rotational motion also has a stabilizing effect when the rotational rate is below a certain threshold. One question we ask is: If these two stabilizing mechanisms are combined, can we stabilize the flow even further? To answer this question, we conducted a set of three-dimensional stability analyses of the flow past a rotating cylinder with a control cylinder in the wake across a vast parameter space. What we found is that combining those two stabilizing mechanisms will suppress the two-dimensional instabilities while slightly exciting three-dimensional instabilities.

1.4 Summary

In this thesis, we will present various numerical techniques and algorithms to investigate the spanwise periodic external incompressible flows. This thesis is organized as follows: in Chapter 2, we present novel techniques to compute the LGF of the screened Poisson equation; in Chapter 3, we present a flow simulation algorithm combining LGF, AMR, and the IB method; in Chapter 4, we present a stability

analysis and resolvent analysis framework; in Chapter 5, we present a numerical study enabled by the algorithm we proposed in Chapter 3; finally, in Chapter 6, we present an application enabled by the linear stability analysis algorithm proposed in Chapter 4.

Chapter 2

FAST AND ROBUST METHOD FOR SCREENED POISSON LATTICE GREEN'S FUNCTION USING ASYMPTOTIC EXPANSION AND FAST FOURIER TRANSFORM

This chapter is adapted from Hou and Colonius (2024b). As we mentioned in the previous chapter, the prerequisite from all subsequent flow simulation and analysis algorithms is an efficient way to tabulate the lattice Green's function (LGF) of the screened Poisson equation. In this chapter, we present three algorithms to compute the entries of this LGF. Specifically, we derive an asymptotic expansion and a trapezoidal rule approximation of this LGF. In addition, we derive the convergence properties and computational complexities of these approximations and their accompanying numerical algorithms. We present several numerical experiments to demonstrate the speedup offered by the proposed algorithms. Even though our motivation lies in solving the Navier-Stokes equations, we recognize that our algorithms for tabulating the LGF have more extensive applications. We demonstrate this point by showing two more applications: solving the three-dimensional Poisson equation with one periodic dimension and computing the return probability of a two-dimensional random walk with killing.

2.1 Introduction

The discrete screened Poisson equation for a k-dimensional space with parameter $c^2 > 0$ is defined as

$$L_c u(\boldsymbol{n}) = c^2 u(\boldsymbol{n}) + \sum_{j=1}^{k} \left[-\alpha_j u(\boldsymbol{n} - \boldsymbol{e}_j) + 2\alpha_j u(\boldsymbol{n}) - \alpha_j u(\boldsymbol{n} + \boldsymbol{e}_j) \right] = f(\boldsymbol{n}) \quad \forall \boldsymbol{n} \in \mathbb{Z}^k, \tag{2.1}$$

where $\boldsymbol{e}_1$, ..., $\boldsymbol{e}_k$ are the coordinate vectors of $\mathbb{R}^k$, and α_1,...,$\alpha_k > 0$ are the anisotropy coefficients. The c^2 term is sometimes called the screening term (Kazhdan and Hoppe, 2013). The LGF is the fundamental solution of the equation above. It plays a role in physics (Kotera, 1962; Katsura and Inawashiro, 1971; Katsura and Inawashiro, 1973), mathematics (Madras, 1989; Lawler and Limic, 2010), and engineering (Cserti, 2000; Kazhdan and Hoppe, 2013). Theoretical aspects of the LGF of the screened Poisson equation have been studied extensively (Morita and Horiguchi, 1971; Katsura and Inawashiro, 1971; Katsura and Inawashiro, 1973;

Michta and Slade, 2021; Maassarani, 2000; Duffin, 1953; Duffin and Shelly, 1958). While the LGF of the Poisson equation has an asymptotic expansion at arbitrary orders (Duffin, 1953; Duffin and Shelly, 1958; Martinsson and Rodin, 2002), the LGF of the screened Poisson equation does not (Gabbard and Rees, 2024b).

Existing theoretical analysis of the LGF of the screened Poisson equation suffices the need for qualitative applications (Kotera, 1962; Katsura and Inawashiro, 1973). However, accurate computation of this LGF is needed for quantitative applications (Cserti, 2000; Gillman and Martinsson, 2010; Kazhdan and Hoppe, 2013; Liska and Colonius, 2017; Caprace, Gillis, and Chatelain, 2021; Gabbard and Rees, 2024b). Thus, in this chapter, we focus on its computational aspects. After reviewing previous results (Section 2.2), we find an asymptotic expansion in terms of the associated value of c^2 and establish the decay rate when c^2 is relatively large (Section 2.3). Next, for small c^2, we derive a one-dimensional integral representation of the LGF (Section 2.4). The same one-dimensional integral representation is also applicable to the LGF of the Poisson equation. We then show that, for screened Poisson equation LGF, the error of a trapezoidal rule approximation can be strictly bounded and converges exponentially fast (Section 2.5). By exploiting the structure of the integrand, we propose a Fast Fourier Transform (Cooley and Tukey, 1965) method for batch evaluation of the LGF (Section 2.7). We show that our algorithm is robust and highly efficient.

We provide two examples to demonstrate how our algorithm can be used in practice. The first example (Section 2.8) is to use the LGF of the screened Poisson equation to solve for the LGF of the three-dimensional Poisson equation with one periodic dimension. The second example (Section 2.9) is to use the LGF of the screened Poisson equation to compute the return probability of a two-dimensional random walk with killing.

2.2 Lattice Green's function of the two-dimensional screened Poisson equation

2.2.1 Definition and two-dimensional integral representation

We consider Eq. 2.1 with $k = 2$, denote $\boldsymbol{n} = (n, m)$, and use $u(\boldsymbol{n})$ and $u(n, m)$ interchangeably. Since $\alpha_1, \alpha_2 > 0$, by re-scaling the coefficients, we can make $\alpha_1 = 1$ or $\alpha_2 = 1$. Thus, without loss of generality, we assume $0 < \alpha_1 \leq 1$ and $\alpha_2 = 1$.

The LGF, denoted as $B_c(\boldsymbol{n}) = B_c(n, m)$, is the solution of

$$[L_c B_c](n, m) = \delta_{0n}\delta_{0m}, \qquad \lim_{|n|+|m|\to\infty} B_c(n, m) = 0, \tag{2.2}$$

where δ_{ij} denotes the Kronecker delta. The Fourier transform method allows the solution to be written as (Katsura and Inawashiro, 1971)

$$B_c(\boldsymbol{n}) = \frac{1}{(2\pi)^2} \int_{I^2} e^{-i\boldsymbol{n}\cdot\boldsymbol{\xi}} \frac{1}{\sigma(\boldsymbol{\xi}) + c^2} d\boldsymbol{\xi}, \tag{2.3}$$

where $I^2 = [-\pi, \pi]^2$ is the integration domain, and with $\boldsymbol{\xi} = (\xi_1, \xi_2)$, the function $\sigma(\boldsymbol{\xi})$ is:

$$\sigma(\boldsymbol{\xi}) = 2\alpha_1 - 2\alpha_1 \cos(\xi_1) + 2 - 2\cos(\xi_2). \tag{2.4}$$

From Eq. 2.3, it is clear that $B_c(n, m) = B_c(|n|, |m|)$ so it suffices to consider $n, m \geq 0$.

For $c = 0$, we recover the standard 2D Poisson equation, and the corresponding LGF can be represented using an asymptotic expansion valid to arbitrarily high order-of-accuracy (Martinsson and Rodin, 2002). In practice, one typically precomputes LGF near-field values and uses these asymptotic expansions for far-field values (Liska and Colonius, 2014). However, when $c \neq 0$, a high-order asymptotic expansion does not exist, and we must rely on numerical integration for all values. Thus, efficient computation is essential.

2.2.2 Representations using special functions

The values of B_c can be expressed using Appell's double hypergeometric functions (F_4):

Theorem 2.2.1. *(Katsura and Inawashiro, 1971).The solution of $B_c(n, m)$ can be written as*

$$B_c(n, m) = \frac{1}{2^{m+n+1}} \frac{1}{a^{m+n+1}} \frac{(m+n)!}{n!m!} \frac{1}{a} \left(\frac{\alpha_1}{a}\right)^n \left(\frac{1}{a}\right)^m F_4\left[(m+n+1)/2, (m+n)/2+1, n+1, m+1; \left(\frac{\alpha_1}{a}\right)^2, \left(\frac{1}{a}\right)^2\right], \tag{2.5}$$

where

$$a = 1 + \alpha_1 + c^2/2. \tag{2.6}$$

Evaluating B_c using Thm. 2.2.1 requires evaluating Appell's double hypergeometric function which is not available in common libraries and contains a doubly infinite sum. Thus in most numerical applications of the LGF, the Bessel function representation (Koster and Slater, 1954; Maradudin, 1960; Katsura and Inawashiro, 1971; Maassarani, 2000; Delves and Joyce, 2001) of $B_c(n, m)$ is used to compute the LGF (Liska and Colonius, 2017; Dorschner et al., 2020; Yu, Dorschner, and Colonius, 2022; Gabbard and Rees, 2024b). The Bessel function representation is a way to write $B_c(n, m)$ as an improper integral of the Bessel function. The specific formulation reads

$$B_c(n,m) = i^{n+m+1}\frac{1}{2}\int_0^\infty e^{-i(2+2\alpha_1+c^2)t/2} J_n(\alpha_1 t) J_m(t)\, dt, \tag{2.7}$$

where $J_k(t)$ is the Bessel function of the first kind (Abramowitz and Stegun, 1948). This formulation is hard to integrate numerically because of the highly oscillatory and slowly decaying integrand. One can further simplify it to a more computationally advantageous form:

$$B_c(n,m) = \int_0^\infty e^{-(2+2\alpha_1+c^2)t} I_n(2\alpha_1 t) I_m(2t)\, dt, \tag{2.8}$$

where $I_k(t)$ is the modified Bessel function of the first kind (Abramowitz and Stegun, 1948).

When evaluating the above integral, there are two challenges: effectively evaluating the function I_n and accurately approximating the improper integral. In practice, for the first challenge, one can compute the modified Bessel function using existing numerical libraries (Schäling, 2011; Virtanen et al., 2020; The mpmath development team, 2023). However, this function is still defined as an infinite series or integral and can be computationally expensive. For the second challenge, the improper integral can either be approximated by integrating up to a large value or be evaluated using a variable transformation (Doncker, 1978). The former can be computationally expensive and unstable, and the latter can create a singularity at the origin. Indeed, it has been reported that numerical evaluation of the LGF of the screened Poisson equation can fail catastrophically (Gabbard and Rees, 2024b). In the rest of the chapter, we introduce two methods to efficiently compute the LGF.

2.3 Fast evaluation and compact support at large c^2

In this section, we derive a series expansion for B_c. From the expansion, we can obtain two unique properties of the LGF of the screened Poisson equation: (a)

exponential convergence in series expansion, and (b) exponential decay in $|m| + |n|$. We will later show the duality between them. These two properties give a fast method to evaluate the LGF at relatively large c^2 and a fast way to solve the screened Poisson equation by applying the LGF.

2.3.1 Series expansion

Recall the LGF, B_c, at an arbitrary point $\boldsymbol{n}$ can be written as:

$$B_c(\boldsymbol{n}) = \frac{1}{(2\pi)^2} \int_{I^2} e^{-i\boldsymbol{n}\cdot\boldsymbol{\xi}} \frac{1}{\sigma(\boldsymbol{\xi}) + c^2} 2\boldsymbol{\xi}, \tag{2.9}$$

where the function $\sigma(\boldsymbol{\xi})$ reads

$$\sigma(\boldsymbol{\xi}) = 2\alpha_1 - 2\alpha_1 \cos(\xi_1) + 2 - 2\cos(\xi_2). \tag{2.10}$$

We define

$$\rho(\boldsymbol{\xi}) = 2\alpha_1 \cos(\xi_1) + 2\cos(\xi_2), \quad \lambda = 2 + 2\alpha_1. \tag{2.11}$$

Thus, we can write:

$$\sigma(\boldsymbol{\xi}) = \lambda - \rho(\boldsymbol{\xi}), \tag{2.12}$$

and thus we can write Eq. 2.3 as

$$B_c(\boldsymbol{n}) = \frac{1}{(2\pi)^2} \int_{I^2} e^{-i\boldsymbol{n}\cdot\boldsymbol{\xi}} \frac{1}{\lambda - \rho(\boldsymbol{\xi}) + c^2} d\boldsymbol{\xi}. \tag{2.13}$$

Now since

$$\rho(\boldsymbol{\xi}) \in [-\lambda, \lambda] \Rightarrow |\rho(\boldsymbol{\xi})| < \lambda + c^2, \tag{2.14}$$

and $c^2 > 0$, we can expand the integral formally as

$$\begin{aligned} B_c(\boldsymbol{n}) &= \frac{1}{(2\pi)^2} \int_{I^2} e^{-i\boldsymbol{n}\cdot\boldsymbol{\xi}} \frac{1}{\lambda + c^2} \frac{1}{1 - \rho(\boldsymbol{\xi})/(\lambda + c^2)} d\boldsymbol{\xi} \\ &= \frac{1}{(2\pi)^2} \int_{I^2} e^{-i\boldsymbol{n}\cdot\boldsymbol{\xi}} \frac{1}{\lambda + c^2} \sum_{k=0}^{\infty} \left(\frac{\rho(\boldsymbol{\xi})}{\lambda + c^2} \right)^k d\boldsymbol{\xi} \\ &= \frac{1}{(2\pi)^2} \frac{1}{\lambda + c^2} \sum_{k=0}^{\infty} \left(\frac{\lambda}{\lambda + c^2} \right)^k \int_{I^2} e^{-i\boldsymbol{n}\cdot\boldsymbol{\xi}} \left(\frac{\rho(\boldsymbol{\xi})}{\lambda} \right)^k d\boldsymbol{\xi}. \end{aligned} \tag{2.15}$$

To show that this series converges, it suffices to show that the dominated convergence theorem applies. That is, as long as we can find an integrable function that dominates the sequence of the integrand, the equation above holds, which leads to the following lemma:

Lemma 2.3.1. *Fix $\boldsymbol{n} \in \mathbb{Z}^2$. Define*

$$f_k(\boldsymbol{\xi}) = e^{-i\boldsymbol{n}\cdot\boldsymbol{\xi}} \frac{1}{\lambda + c^2} \sum_{l=0}^{k} \left(\frac{\rho(\boldsymbol{\xi})}{\lambda + c^2} \right)^l. \tag{2.16}$$

Then f_k is dominated by a constant:

$$|f_k| \leq \frac{1}{c^2}. \tag{2.17}$$

Proof. Consider

$$\begin{aligned} |f_k| &= \left| e^{-i\boldsymbol{n}\cdot\boldsymbol{\xi}} \frac{1}{\lambda + c^2} \sum_{l=0}^{k} \left(\frac{\rho(\boldsymbol{\xi})}{\lambda + c^2} \right)^l \right| \\ &\leq |e^{-i\boldsymbol{n}\cdot\boldsymbol{\xi}}| \left| \frac{1}{\lambda + c^2} \right| \left| \sum_{l=0}^{k} \left(\frac{\lambda}{\lambda + c^2} \right)^l \left(\frac{\rho(\boldsymbol{\xi})}{\lambda} \right)^l \right| \\ &\leq \left| \frac{1}{\lambda + c^2} \right| \left| \sum_{l=0}^{k} \left(\frac{\lambda}{\lambda + c^2} \right)^l \right| \\ &\leq \left| \frac{1}{\lambda + c^2} \right| \frac{1}{1 - \lambda/(\lambda + c^2)} = \frac{1}{c^2}. \end{aligned} \tag{2.18}$$

Thus, f_k is dominated by $1/c^2$. □

Since the integration domain I^2 is finite, f_k is integrable. As a result, the dominated convergence theorem applies, so the series expansion of $B_c(\boldsymbol{n})$ given in Eq. 2.15 converges to $B_c(\boldsymbol{n})$. With Eq. 2.15, we can define an approximation of B_c.

Definition 2.3.1.1. *The N-term approximation of $B_c(\boldsymbol{n})$, denoted as $G_N(c, \boldsymbol{n})$, is defined as*

$$G_N(c, \boldsymbol{n}) = \frac{1}{(2\pi)^2} \frac{1}{\lambda + c^2} \sum_{k=0}^{N-1} \left(\frac{\lambda}{\lambda + c^2} \right)^k \int_{I^2} e^{-i\boldsymbol{n}\cdot\boldsymbol{\xi}} \left(\frac{\rho(\boldsymbol{\xi})}{\lambda} \right)^k d\boldsymbol{\xi}. \tag{2.19}$$

We can bound the error of this N-term approximation with the following theorem.

Theorem 2.3.2. *The truncation error from the N-term approximation of $B_c(\boldsymbol{n})$ is bounded by*

$$|\epsilon_N(\boldsymbol{n})| = |B_c(\boldsymbol{n}) - G_N(c, \boldsymbol{n})| \leq \frac{1}{c^2} \left(\frac{\lambda}{\lambda + c^2} \right)^N \quad \forall \boldsymbol{n} \in \mathbb{Z}^2. \tag{2.20}$$

Proof. Fix $\boldsymbol{n} \in \mathbb{Z}^n$, and the truncation error is:

$$\begin{aligned}\epsilon_N &= B_c(\boldsymbol{n}) - G_N(c, \boldsymbol{n}) \\ &= \frac{1}{(2\pi)^2} \frac{1}{\lambda + c^2} \sum_{k=N}^{\infty} \left(\frac{\lambda}{\lambda + c^2} \right)^k \int_{I^2} e^{-i\boldsymbol{n}\cdot\boldsymbol{\xi}} \left(\frac{\rho(\boldsymbol{\xi})}{\lambda} \right)^k d\boldsymbol{\xi}.\end{aligned} \tag{2.21}$$

The error, ϵ_N, can be bounded by the following:

$$\begin{aligned}|\epsilon_N| &= \left| \frac{1}{(2\pi)^2} \frac{1}{\lambda + c^2} \sum_{k=N}^{\infty} \left(\frac{\lambda}{\lambda + c^2} \right)^k \int_{I^2} e^{-i\boldsymbol{n}\cdot\boldsymbol{\xi}} \left(\frac{\rho(\boldsymbol{\xi})}{\lambda} \right)^{k+N} d\boldsymbol{\xi} \right| \\ &\leq \left| \frac{1}{(2\pi)^2} \frac{1}{\lambda + c^2} \left(\frac{\lambda}{\lambda + c^2} \right)^N \sum_{k=0}^{\infty} 4\pi^2 \left(\frac{\lambda}{\lambda + c^2} \right)^k \right| \\ &= \frac{1}{\lambda + c^2} \left(\frac{\lambda}{\lambda + c^2} \right)^N \frac{1}{1 - \lambda/(\lambda + c^2)} \\ &= \frac{1}{c^2} \left(\frac{\lambda}{\lambda + c^2} \right)^N.\end{aligned} \tag{2.22}$$

Since this error bound is independent of $\boldsymbol{n}$, it is true for all $\boldsymbol{n}$. □

2.3.2 Analytical expressions of the N-term approximation

It turns out that each term in the series representation of B_c can be analytically computed as functions of multinomial coefficients. To do so, we define

$$g_k(\boldsymbol{n}) = \frac{1}{4\pi^2} \int_{I^2} e^{-i\boldsymbol{n}\cdot\boldsymbol{\xi}} \left(\frac{\rho(\boldsymbol{\xi})}{\lambda} \right)^k d\boldsymbol{\xi}, \tag{2.23}$$

so that we can write

$$G_N(c, \boldsymbol{n}) = \frac{1}{\lambda + c^2} \sum_{k=0}^{N-1} \left(\frac{\lambda}{\lambda + c^2} \right)^k g_k(\boldsymbol{n}). \tag{2.24}$$

Now we can focus on those $g_k(\boldsymbol{n})$ terms alone. The following theorem gives an analytical expression of $g_k(\boldsymbol{n})$.

Theorem 2.3.3. *The function $g_k(n, m)$ is nonzero if and only if $k \geq |n| + |m|$ and $k - |n| - |m|$ is even. In that case,*

$$g_k(n, m) = \frac{1}{\lambda^k} \sum_{l=0}^{(k-n-m)/2} \alpha_1^{n+2l} \binom{k}{l, n+l, (k-n-2l-m)/2, (k-n-2l+m)/2} \tag{2.25}$$

where

$$\binom{k}{a, b, c, d} = \frac{k!}{a!b!c!d!} \tag{2.26}$$

is the multinomial coefficient.

Remark 2.3.3.1. *A way to effectively evaluate Eq. 2.25 without numerical overflow is using the log Gamma function, which is relatively well-behaved. With this result, for a finite k, one can efficiently evaluate g_k directly using built-in log game functions, e.g., the* `lgamma` *function in C++ or using existing numerical packages such as* `Boost` *(Schäling, 2011) and* `SciPy` *(Virtanen et al., 2020).*

Remark 2.3.3.2. *This theorem completes the entire asymptotic expansion of B_c. This expression can also be derived from the perspective of a random walk with killing. Similar results for the LGF on square lattices ($\alpha_1 = 1$) have been derived using a random walk argument (Michta and Slade, 2021).*

Proof of Thm. 2.3.3. We directly expand the integral form of g_k

$$\begin{aligned} g_k(\boldsymbol{n}) &= \frac{1}{4\pi^2}\int_{I^2} e^{-i\boldsymbol{n}\cdot\boldsymbol{\xi}}\left(\frac{\rho(\boldsymbol{\xi})}{\lambda}\right)^k d\boldsymbol{\xi} \\ &= \frac{1}{4\pi^2}\frac{1}{\lambda^k}\int_{I^2} e^{-i\boldsymbol{n}\cdot\boldsymbol{\xi}}\left(2\alpha_1\cos(\xi_1)+2\cos(\xi_2)\right)^k d\boldsymbol{\xi} \\ &= \frac{1}{4\pi^2}\frac{1}{\lambda^k}\int_{I^2} e^{-i\boldsymbol{n}\cdot\boldsymbol{\xi}}\sum_{l=0}^{k}\binom{k}{l}\left(2\alpha_1\cos(\xi_1)\right)^l\left(2\cos(\xi_2)\right)^{k-l} d\boldsymbol{\xi} \\ &= \frac{1}{4\pi^2}\frac{1}{\lambda^k}\sum_{l=0}^{k}\binom{k}{l}\alpha_1^l\int_I e^{-in_1\xi_1}\left(2\cos(\xi_1)\right)^l d\xi_1\int_I e^{-in_2\xi_2}\left(2\cos(\xi_2)\right)^{k-l} d\xi_2. \end{aligned} \tag{2.27}$$

A direct calculation shows that

$$\int_I e^{-in\xi}\left(2\cos(\xi)\right)^p d\xi = \begin{cases} 2\pi\dbinom{p}{(p-n)/2} & \text{if } (p-n)\geq 0 \text{ and } (p-n) \text{ is even} \\ 0 & \text{otherwise.} \end{cases} \tag{2.28}$$

Plugging this expression into Eq. 2.27, we obtain the desired result. □

In Figure 2.1, we demonstrate the error convergence rate of the LGF approximated by G_N at selected values of (n, m) at $\alpha_1 = 0.75$ compared with corresponding error bounds given by Thm. 2.3.2. The error is computed by comparing against the analytical values computed using the Appell function representation given in Eq. 2.5. The analytical values are computed to 50 decimal places using the Python package `mpmath` (The mpmath development team, 2023). This figure shows that for a fixed point, the error from the G_N approximations monotonically decreases as N increases. The monotonicity arises from the fact that g_k is nonnegative.

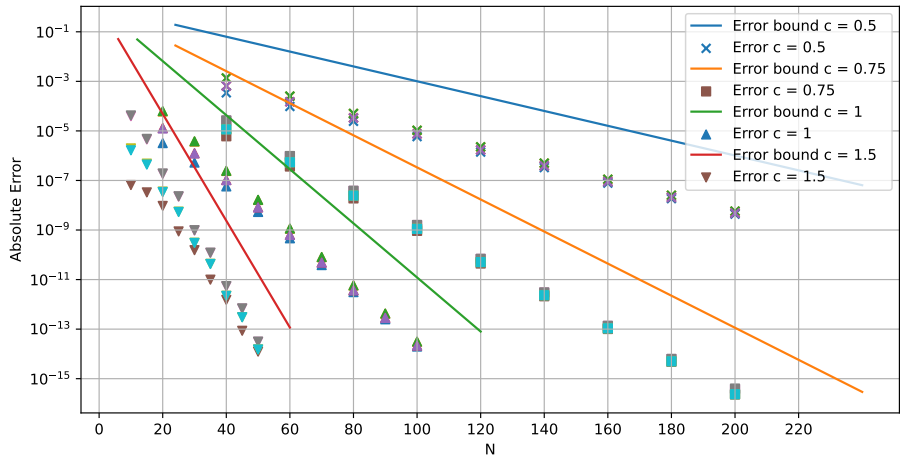


Figure 2.1: Error of G_N for various c at $\alpha_1 = 0.75$. We randomly choose 5 points within the square $[0, 10)^2$ and evaluate their G_N approximation using various N at different c. We compare the resulting G_N with the solution obtained by evaluating B_c at those points using Eq. 2.5. We also show the error bounds given by Thm. 2.3.2.

2.3.3 Spatial decay of the lattice Green's function

Fix $n, m \geq 0$. With $\boldsymbol{n} = (n, m)$,

$$G_{n+m}(c, \boldsymbol{n}) = \frac{1}{\lambda + c^2} \sum_{k=0}^{n+m-1} \left(\frac{\lambda}{\lambda + c^2} \right)^k g_k(\boldsymbol{n}). \tag{2.29}$$

By Thm. 2.3.3, $g_k(n, m) = 0$ for $k < n + m$. As a result, $G_{n+m}(c, \boldsymbol{n}) = 0$. Thus, by Thm. 2.3.2, we can write

$$|B_c(\boldsymbol{n})| = |B_c(\boldsymbol{n}) - G_{n+m}(c, \boldsymbol{n})| = |\epsilon_{n+m}| \leq \frac{1}{c^2} \left(\frac{\lambda}{\lambda + c^2} \right)^{(n+m)}. \tag{2.30}$$

Thus, $B_c(n, m)$ decays exponentially fast as $n + m$ increases.

Although this result is immediate from Thm. 2.3.2 and Thm. 2.3.3, it has at least two important implications.

First, when approximating B_c on a large domain using G_N, the number of terms in G_N we need to evaluate decreases as $m + n$ increases. By only evaluating those nonzero terms, we can achieve significant computational savings when evaluating each term of B_c.

Second, when solving the screened Poisson equation using B_c to a certain precision, we need only convolve B_c within regions where B_c is larger than the prescribed

precision. In this way, applying B_c can be made much more computationally efficient.

As it is evident from the error estimates in Thm. 2.3.2, when c^2 is large, one can approximate B_c to the machine precision using only a few terms, and we only need to evaluate a small number of B_c since it decays exponentially fast. As a result, when c^2 is large, it is favorable to evaluate the LGF using Eq. 2.19.

2.4 Calculation of the lattice Green's function at arbitrary nonzero c^2

For smaller values of c^2, evaluating B_c using Eq. 2.5 and Eq. 2.19 becomes more expensive. To resolve this problem, we introduce a one-dimensional integral formulation of B_c through the following theorem:

Theorem 2.4.1. *The value of $B_c(n,m)$ for any $n,m \in \mathbb{Z}$ and $c > 0$ can be written as:*

$$B_c(n,m) = \frac{1}{2\pi}\int_{-\pi}^{\pi} \frac{e^{i\theta n}}{K^{|m|}} \frac{d\theta}{K - 1/K} \tag{2.31}$$

where

$$K = \frac{\phi + \sqrt{\phi^2 - 4}}{2}, \quad \phi = \lambda + c^2 - 2\alpha_1 \cos(\theta), \quad \lambda = 2 + 2\alpha_1. \tag{2.32}$$

Proof. We first rewrite the governing equation (Eq. 2.2 without the far field boundary condition) as

$$\begin{aligned}\lambda B_c(n,m) - (B_c(n,m-1) + B_c(n,m+1)) \\ = \alpha_1(B_c(n-1,m) + B_c(n+1,m)) - c^2 B_c(n,m) + \delta_{0n}\delta_{0m}.\end{aligned} \tag{2.33}$$

The N-term discrete Fourier transform in the first coordinate is

$$\tilde{B}_c^N(k,m) = \sum_{n=1-N/2}^{N/2} B_c(n,m) e^{-2\pi i k n/N}. \tag{2.34}$$

We first impose that $B_c(n,m)$ is periodic in n with periodicity N (assuming that N

is even); we later relax periodicity in the limit $N \to \infty$. Consider the following:

$$\begin{aligned}
&\tilde{B}_c^N(k, m-1) + \tilde{B}_c^N(k, m+1) - \lambda \tilde{B}_c^N(k, m) \\
&= \sum_{n=1-N/2}^{N/2} \left[B_c(n, m-1) e^{-2\pi i k n/N} + B_c(n, m+1) e^{-2\pi i k n/N} - \lambda B_c(n, m) e^{-2\pi i k n/N} \right] \\
&= \sum_{n=1-N/2}^{N/2} \left[B_c(n, m-1) + B_c(n, m+1) - \lambda B_c(n, m) \right] e^{-2\pi i k n/N} \\
&= \sum_{n=1-N/2}^{N/2} \left[c^2 B_c(n, m) - \alpha_1 B_c(n-1, m) - \alpha_1 B_c(n+1, m) - \delta_{0m}\delta_{0n} \right] e^{-2\pi i k n/N}.
\end{aligned}$$

With the periodicity assumption, we can write

$$\begin{aligned}
&\sum_{n=1-N/2}^{N/2} \left[c^2 B_c(n, m) - \alpha_1 B_c(n-1, m) - \alpha_1 B_c(n+1, m) \right] e^{-2\pi i k n/N} \\
&= \sum_{n=1-N/2}^{N/2} B_c(n, m) \left[c^2 e^{-2\pi i k n/N} - \alpha_1 e^{-2\pi i k (n+1)/N} - \alpha_1 e^{-2\pi i k (n-1)/N} \right] \\
&= \sum_{n=1-N/2}^{N/2} B_c(n, m) e^{-2\pi i k n/N} (c^2 - 2\alpha_1 \cos(2\pi k/N)) \\
&= [c^2 - 2\alpha_1 \cos(2\pi k/N)] \tilde{B}_c^N(k, m)
\end{aligned}$$

and

$$\sum_{n=1-N/2}^{N/2} \delta_{0m}\delta_{0n} e^{-2\pi i k n/N} = \delta_{0m}. \tag{2.35}$$

As a result, we have

$$\tilde{B}_c^N(k, m-1) + \tilde{B}_c^N(k, m+1) = [\lambda + c^2 - 2\alpha_1 \cos(2\pi k/N)] \tilde{B}_c^N(k, m) - \delta_{0m}. \tag{2.36}$$

On the one hand, if $m \neq 0$, we have

$$\tilde{B}_c^N(k, m-1) + \tilde{B}_c^N(k, m+1) + [2\alpha_1 \cos(2\pi k/N) - \lambda - c^2] \tilde{B}_c^N(k, m) = 0. \tag{2.37}$$

This type of recurrence relation can be solved by assuming the following ansatz (Buneman, 1971):

$$\tilde{B}_c^N(k, m) = \tilde{B}_c^N(k, 0)/K^{|m|}. \tag{2.38}$$

By directly plugging in our recurrence relation, K can be solved using

$$K + 1/K = \lambda + c^2 - 2\alpha_1 \cos(2\pi k/N) := \phi_N. \tag{2.39}$$

To enforce the condition that $B_c(n,m) \to 0$ as $|m| \to \infty$, we need $|K| > 1$. Thus, the appropriate quadratic root is

$$K = \frac{\phi_N + \sqrt{\phi_N^2 - 4}}{2}. \tag{2.40}$$

On the other hand, if $m = 0$, we have

$$\tilde{B}_c^N(k,-1) + \tilde{B}_c^N(k,1) + [2\alpha_1 \cos(2\pi k/N) - \lambda - c^2]\tilde{B}_c^N(k,0) = -1. \tag{2.41}$$

Substituting the equation of K, we obtain the solution of $\tilde{B}_c^N(k,0)$ as

$$2\tilde{B}_c^N(k,0)/K - \phi_N \tilde{B}_c^N(k,0) = -1 \Rightarrow \tilde{B}_c^N(k,0) = \frac{1}{K - 1/K}. \tag{2.42}$$

Thus, the expression of $\tilde{B}_c^N(k,m)$ comes out to be

$$\tilde{B}_c^N(k,m) = \frac{1}{K - 1/K}\frac{1}{K^{|m|}}. \tag{2.43}$$

With the expression of $\tilde{B}_c^N(k,m)$, we take the inverse discrete Fourier transform to obtain $B_c(n,m)$

$$B_c(n,m) = \frac{1}{N}\sum_{k=1-N/2}^{N/2} e^{2\pi i k n/N}\tilde{B}_c^N(k,m) = \frac{1}{N}\sum_{k=1-N/2}^{N/2} e^{2\pi i k n/N}\frac{1}{K - 1/K}\frac{1}{K^{|m|}}. \tag{2.44}$$

Now, we are ready to take N to infinity. To do so, define

$$\theta_k = 2\pi k/N, \quad \Delta\theta = 2\pi/N. \tag{2.45}$$

The expression of $B_c(n,m)$ becomes

$$B_c(n,m) = \Delta\theta \sum_{k=1-N/2}^{N/2} e^{in\theta_k}\frac{1}{K - 1/K}\frac{1}{K^{|m|}} \tag{2.46}$$

where

$$K = \frac{\phi_N + \sqrt{\phi_N^2 - 4}}{2}, \quad \phi_N = \lambda + c^2 - 2\alpha_1 \cos(\theta_k). \tag{2.47}$$

Note that now the summands are composed entirely of θ_k, without direct involvement of N. By taking N to infinity, we are creating a Riemann sum. Since the function defining the summands is bounded and continuous, the Riemann sum converges, and we write

$$\lim_{N\to\infty} \Delta\theta \sum_{k=1-N/2}^{N/2} e^{in\theta_k}\frac{1}{K - 1/K}\frac{1}{K^{|m|}} = \frac{1}{2\pi}\int_{-\pi}^{\pi} e^{in\theta}\frac{1}{K - 1/K}\frac{d\theta}{K^{|m|}} \tag{2.48}$$

where

$$K = \frac{\phi + \sqrt{\phi^2 - 4}}{2}, \quad \phi = \lambda + c^2 - 2\alpha_1 \cos(\theta). \tag{2.49}$$

As a result, can write

$$B_c(n, m) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \frac{e^{in\theta}}{K - 1/K} \frac{d\theta}{K^{|m|}}. \tag{2.50}$$

Since we have taken N to infinity, $B_c(n, m)$ does not have to be periodic anymore. □

Remark 2.4.1.1. *The proof of Eq. 2.4.1 generalizes the techniques presented by Buneman (1971), where the author only considered the case of $c^2 = 0$ and $\alpha_1 = 1$. In that case, K only has one root, so there is no need to identify the correct root.*

Using similar techniques, one can show that, when $c^2 = 0$ and $\alpha_1 < 1$, the corresponding LGF is

$$B_0(n, m) - B_0(0, 0) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left(\frac{e^{in\theta}}{K^{|m|}} - 1 \right) \frac{d\theta}{K - 1/K}, \tag{2.51}$$

where

$$K = \frac{\phi + \sqrt{\phi^2 - 4}}{2}, \quad \phi = \lambda - 2\alpha_1 \cos(\theta). \tag{2.52}$$

The proof follows from the proof of Thm. 2.4.1.

Remark 2.4.1.2. *In contrast to Eq. 2.8, the integral in Eq. 2.31 has a finite integration domain and an integrand consisting of elementary functions only. As a result, numerical integrating Eq. 2.31 is faster and more stable.*

2.5 Convergence rate of the trapezoidal rule approximation

We cannot reduce the one-dimensional integral presentation of $B_c(n, m)$ any further. Thus, we need to evaluate it numerically. It turns out, however, that the trapezoidal rule approximation yields an exponential convergence rate for this particular integral. To show this, we first invoke the following general theorem on the convergence rate of trapezoidal rule approximations (Trefethen and Weideman, 2014):

Theorem 2.5.1. *(Trefethen and Weideman, 2014) Let*

$$I = \int_{-\pi}^{\pi} v(\theta) d\theta. \tag{2.53}$$

For any positive integer N, define the trapezoidal rule approximation:

$$I_N = \frac{2\pi}{N} \sum_{k=1}^{N} v(\theta_k) \tag{2.54}$$

where $\theta_k = 2\pi k/N - \pi$. *Suppose* v *is* 2π *periodic and analytic and satisfies* $|v(\theta)| < M$ *in the strip* $-\gamma < \Im(\theta) < \gamma$ *for some* $\gamma > 0$. *Then for any* $N \geq 1$,

$$|I_N - I| \leq \frac{4\pi M}{e^{\gamma N} - 1}, \tag{2.55}$$

and the constant 4π *is as small as possible.*

Using the above theorem, we can show the following result regarding the convergence rate of the trapezoidal rule approximation for Eq. 2.31.

Theorem 2.5.2. *Let* $\alpha_1 \in (0, 1)$ *and*

$$v(\theta) = \frac{e^{i\theta n}}{K^{|m|}} \frac{1}{K - 1/K}, \quad \phi = \lambda + c^2 - 2\alpha_1 \cos(\theta), \quad K = \frac{\phi + \sqrt{\phi^2 - 4}}{2}. \tag{2.56}$$

Let

$$I = \int_{-\pi}^{\pi} v(\theta) d\theta \tag{2.57}$$

and I_N *be its trapezoidal rule approximation. Then for any positive real number,* γ_c, *satisfying*

$$\gamma_c < \log\left(1 + \frac{c^2}{2\alpha_1} + \sqrt{\left(1 + \frac{c^2}{2\alpha_1}\right)^2 - 1}\right), \tag{2.58}$$

for any $N \geq 1$, *we have*

$$|I_N - I| \leq \frac{4\pi M}{e^{\gamma_c N} - 1}, \tag{2.59}$$

where M *is*

$$M = \sup_{|\Im(\theta)| < \gamma_c} \left| \frac{e^{i\theta n}}{K^{|m|}} \frac{1}{K - 1/K} \right|. \tag{2.60}$$

Before proving the above theorem, we first prove the following technical lemma

Lemma 2.5.3. *For* $\theta \in \mathbb{C}$,

$$|\alpha_1 \cos(\theta)| < \lambda/2 + c^2/2 - 1 \Rightarrow \phi^2 - 4 \notin \mathbb{R}_{<0}. \tag{2.61}$$

Proof of Lemma 2.5.3. We prove this lemma by contradiction. Assume

$$\phi^2 - 4 = v \in \mathbb{R}_{<0}. \tag{2.62}$$

We rewrite this equation as:

$$(\lambda + c^2 - 2\alpha_1 \cos(\theta))^2 = 4 + v \tag{2.63}$$

We solve for $\alpha_1 \cos(\theta)$:

$$\alpha_1 \cos(\theta) = \frac{1}{2}\left[\lambda + c^2 \pm \sqrt{4+v}\right]. \tag{2.64}$$

Since

$$|\alpha_1 \cos(\theta)| = \sqrt{\left(\frac{1}{2}(\lambda + c^2) \pm \frac{1}{2}\Re(\sqrt{4+v})\right)^2 + \left(\frac{1}{2}\Im(\sqrt{4+v})\right)^2}, \tag{2.65}$$

we have

$$|\alpha_1 \cos(\theta)| \geq \frac{1}{2}(\lambda + c^2) - \frac{1}{2}|\Re(\sqrt{4+v})| > \frac{1}{2}(\lambda + c^2) - 1. \tag{2.66}$$

Thus, we have a contradiction. □

Proof of Thm. 2.5.2. To use Thm. 2.5.1, we need to find a strip within which our specific $v(\theta)$ is analytic

$$v(\theta) = \frac{e^{i\theta n}}{K^{|m|}} \frac{1}{K - 1/K}, \quad \phi = \lambda + c^2 - 2\alpha_1 \cos(\theta), \quad K = \frac{\phi + \sqrt{\phi^2 - 4}}{2}. \tag{2.67}$$

Inspecting the above expression, we know that $v(\theta)$ is analytic in a strip if, in which, $K - 1/K \neq 0$ and $\phi^2 - 4 \notin \mathbb{R}_{<0}$.

We first find a strip in which $K - 1/K \neq 0$. To do so, we only need to pick any finite γ such that

$$K - 1/K \neq 0 \quad \forall \theta \in \mathbb{C} : |\Im(\theta)| < \gamma. \tag{2.68}$$

We have

$$K - 1/K = \sqrt{\phi^2 - 4}, \tag{2.69}$$

so

$$K - 1/K = 0 \iff \phi^2 = 4 \iff \phi = \pm 2. \tag{2.70}$$

Directly plugging in the expression of ϕ, we obtain:

$$\cos(\theta) = \frac{1}{\alpha_1}\left[\frac{\lambda}{2} \pm 1 + \frac{1}{2}c^2\right]. \tag{2.71}$$

To facilitate the discussion, we denote the two possible values on the RHS as:

$$\phi_m^+ = \frac{1}{\alpha_1}\left[\frac{\lambda}{2} + 1 + \frac{1}{2}c^2\right], \quad \phi_m^- = \frac{1}{\alpha_1}\left[\frac{\lambda}{2} - 1 + \frac{1}{2}c^2\right]. \tag{2.72}$$

A necessary condition for Eq. 2.71 to be satisfied is

$$\exp(\Im(\theta)) = \phi_m^+ \pm \sqrt{(\phi_m^+)^2 - 1} \text{ or } \exp(\Im(\theta)) = \phi_m^- \pm \sqrt{(\phi_m^-)^2 - 1} \tag{2.73}$$

In addition, we notice that

$$\frac{1}{\phi_m^\pm + \sqrt{(\phi_m^\pm)^2 - 1}} = \frac{\phi_m^\pm - \sqrt{(\phi_m^\pm)^2 - 1}}{(\phi_m^\pm + \sqrt{(\phi_m^\pm)^2 - 1})(\phi_m^\pm - \sqrt{(\phi_m^\pm)^2 - 1})} = \phi_m^\pm - \sqrt{(\phi_m^\pm)^2 - 1}. \tag{2.74}$$

Consequently,

$$\log(\phi_m^\pm + \sqrt{(\phi_m^\pm)^2 - 1}) = -\log(\phi_m^\pm - \sqrt{(\phi_m^\pm)^2 - 1}). \tag{2.75}$$

Since the logarithm function is a monotonically increasing function and that $|\log(\phi_m^\pm + \sqrt{(\phi_m^\pm)^2 - 1})| = |\log(\phi_m^\pm - \sqrt{(\phi_m^\pm)^2 - 1})|$, to ensure analyticity within the strip, we need

$$|\Im(\theta)| < \gamma := \log(\phi_m^- + \sqrt{(\phi_m^-)^2 - 1}). \tag{2.76}$$

We then focus on the second condition regarding K, i.e.

$$\phi^2 - 4 \notin \mathbb{R}_{<0}. \tag{2.77}$$

Since the function $e^x + e^{-x}$ monotonically increases with x when $x \geq 0$, within the strip of $|\Im(\theta)| < \gamma$, we have

$$\begin{aligned} |\cos(\theta)| &\leq |e^{\Im(\theta)}|/2 + |e^{-\Im(\theta)}|/2 < \frac{1}{2}(e^\gamma + e^{-\gamma}) \\ &= \frac{1}{2}\left(\phi_m^- + \sqrt{(\phi_m^-)^2 - 1} + \phi_m^- - \sqrt{(\phi_m^-)^2 - 1}\right) = \phi_m^-. \end{aligned} \tag{2.78}$$

Thus, $|\alpha_1 \cos(\theta)| < (\lambda/2 - 1 + c^2/2)$, so by Lemma 2.5.3, $\phi^2 - 4 \notin \mathbb{R}_{<0}$. Considering $K - 1/K \neq 0$, we know that $v(\theta)$ is analytic within the strip. If we pick any $\gamma_c < \gamma$, we have

$$|v(\theta)| \leq M = \sup_{|\Im(\theta)| < \gamma_c} \left| \frac{e^{i\theta n}}{K^{|m|}} \frac{1}{K - 1/K} \right|. \tag{2.79}$$

We write

$$\phi_m^- = \frac{1}{\alpha_1}\left[\frac{\lambda}{2} - 1 + \frac{1}{2}c^2\right] = \frac{1}{\alpha_1}\left[1 + \alpha_1 - 1 + \frac{1}{2}c^2\right] = 1 + \frac{c^2}{2\alpha_1}. \tag{2.80}$$

Thus, we can find γ to be:

$$\gamma = \log\left(1 + \frac{c^2}{2\alpha_1} + \sqrt{\left(1 + \frac{c^2}{2\alpha_1}\right)^2 - 1}\right). \tag{2.81}$$

And by Thm. 2.5.1, we conclude with the desired result. □

Remark 2.5.3.1. *The essence of Thm. 2.5.2 is to show that the integrand in Eq. 2.31 is analytic with the strip $|\Im(\theta)| \leq \gamma_c$ for any γ_c satisfying*

$$\gamma_c < \gamma = \log\left(1 + \frac{c^2}{2\alpha_1} + \sqrt{\left(1 + \frac{c^2}{2\alpha_1}\right)^2 - 1}\right). \tag{2.82}$$

In this sense, a more concise version of Thm. 2.5.2 can be stated. However, we prefer the verbose version due to its central role in developing the algorithms that will be introduced in the subsequent sections.

Remark 2.5.3.2. *Investigating Thm. 2.5.2 further, we can understand the effect of n in the error in Eq. 2.60 better. We can write*

$$M = \sup_{|\Im(\theta)|<\gamma_c} \left| \frac{e^{i\theta n}}{K^{|m|}} \frac{1}{K - 1/K} \right| \leq e^{\gamma_c n} \left[\sup_{|\Im(\theta)|<\gamma_c} \left| \frac{1}{K^{|m|}} \frac{1}{K - 1/K} \right| \right]. \tag{2.83}$$

Then the error estimate becomes

$$|I_N - I| \leq \frac{4\pi e^{\gamma_c n} \overline{M}}{e^{\gamma_c N} - 1} = \frac{4\pi \overline{M}}{e^{\gamma_c (N-n)} - e^{-\gamma_c n}} \leq \frac{4\pi \overline{M}}{e^{\gamma_c (N-n)} - 1} \tag{2.84}$$

where

$$\overline{M} = \sup_{|\Im(\theta)|<\gamma_c} \left| \frac{1}{K^{|m|}} \frac{1}{K - 1/K} \right|. \tag{2.85}$$

Thus, we only need to increase N as fast as n to maintain the same accuracy.

Corollary 2.5.3.1. *Let $\eta \in (0, c^2/\alpha_1)$, $N > n \geq 0$. Following the definitions of Thm.2.5.1, we have*

$$|I_N - I| \leq \frac{4\pi M_\eta}{e^{\gamma_\eta (N-n)} - e^{-\gamma_\eta n}}, \tag{2.86}$$

where

$$\gamma_\eta = \log\left(1 + \frac{\eta}{2} + \sqrt{\left(1 + \frac{\eta}{2}\right)^2 - 1}\right), \quad M_\eta = \frac{1}{2\sqrt{c^2/\alpha_1 - \eta}}. \tag{2.87}$$

Proof of Corollary2.5.3.1. Since $\eta \in (0, c^2/\alpha_1)$, we have

$$\gamma_\eta = \log\left(1 + \frac{\eta}{2} + \sqrt{\left(1 + \frac{\eta}{2}\right)^2 - 1}\right) < \log\left(1 + \frac{c^2}{2\alpha_1} + \sqrt{\left(1 + \frac{c^2}{2\alpha_1}\right)^2 - 1}\right) \tag{2.88}$$

By Thm. 2.5.2 and Rmk. 2.5.3.2, we have

$$|I_N - I| \leq \frac{4\pi M'_\eta}{e^{\gamma_\eta (N-n)} - e^{-\gamma_\eta n}} \tag{2.89}$$

where M'_η is

$$M'_\eta = \sup_{|\Im(\theta)|<\gamma_\eta} \left| \frac{1}{K^{|m|}} \frac{1}{K - 1/K} \right|. \tag{2.90}$$

We can first put an upper bound on $|\cos(\theta)|$ within the strip $|\Im(\theta)| < \gamma_\eta$ as

$$|\cos(\theta)| \leq |e^{\gamma_\eta}/2| + |e^{-\gamma_\eta}/2| = 1 + \eta/2. \tag{2.91}$$

By Lemma 2.5.3, $\phi^2 - 4 \notin \mathbb{R}_{<0}$ within the strip $|\Im(\theta)| < \gamma_\eta$. Thus, for $\sqrt{\phi^2 - 4}$, we only take the principal branch. That is,

$$\sqrt{\phi^2 - 4} = |\phi^2 - 4|^{1/2} e^{i \operatorname{Arg}(\phi^2 - 4)/2}. \tag{2.92}$$

Since $\operatorname{Arg}(\phi^2 - 4) \in (-\pi, \pi)$, we have

$$\Re(\sqrt{\phi^2 - 4}) > 0. \tag{2.93}$$

Also,

$$\Re(\phi) \geq \lambda + c^2 - 2|\alpha_1 \cos(\theta)| \geq \lambda + c^2 - 2\alpha_1(1 + \eta/2) = 2 + (c^2 - \eta\alpha_1) > 2. \tag{2.94}$$

Combined with the previous result, we obtain a lower bound of $|K|$

$$\Re(\phi + \sqrt{\phi^2 - 4}) > 2, \quad \text{and} \quad |K| = \left| \frac{\phi + \sqrt{\phi^2 - 4}}{2} \right| > 1. \tag{2.95}$$

Then we can write

$$M'_\eta = \sup_{|\Im(\theta)|<\gamma_\eta} \left| \frac{1}{K^{|m|}} \frac{1}{K - 1/K} \right| \leq \sup_{|\Im(\theta)|<\gamma_\eta} \left| \frac{1}{K - 1/K} \right| = \sup_{|\Im(\theta)|<\gamma_\alpha} \left| \frac{1}{\sqrt{\phi^2 - 4}} \right|. \tag{2.96}$$

Further, we can put a lower bound on $|\phi^2 - 4|$, i.e.

$$\begin{aligned}
|\phi^2 - 4| &= |\phi - 2||\phi + 2| \\
&= |\lambda + c^2 - 2\alpha_1 \cos\theta - 2||\lambda + c^2 - 2\alpha_1 \cos\theta + 2| \\
&\geq |\lambda - 2 + c^2 - 2\alpha_1(1 + \eta/2)||\lambda + 2 + c^2 - 2\alpha_1(1 + \eta/2)| \\
&= (c^2 - \eta\alpha_1)(4 + c^2 - \eta\alpha_1) \\
&> 4(c^2 - \eta\alpha_1).
\end{aligned}$$

Thus, we can bound M'_η using

$$M'_\eta = \sup_{|\Im(\theta)|<\gamma_\eta} \left| \frac{1}{K^{|m|}} \frac{1}{K - 1/K} \right| \leq \frac{1}{2\sqrt{c^2 - \eta\alpha_1}} := M_\eta. \tag{2.97}$$

Finally, we conclude with

$$|I_N - I| \leq \frac{4\pi M'_\eta}{e^{\gamma_\eta(N-n)} - e^{-\gamma_\eta n}} \leq \frac{4\pi M_\eta}{e^{\gamma_\eta(N-n)} - e^{-\gamma_\eta n}}. \tag{2.98}$$

□

Remark 2.5.3.3. *Although the bound in Corollary 2.5.3.1 is looser than the one given in Thm. 2.5.2, it provides us with an a priori estimation of the error from the trapezoidal rule approximation of B_c, depending only on $N - n$ and η. This bound will be useful when we introduce the numerical framework to evaluate the trapezoidal rule approximation in the next section.*

2.6 Fast Fourier Transform method for solving the lattice Green's function

As we have established the convergence rate of the trapezoidal rule approximation of B_c, we also notice that the specific form of the trapezoidal rule approximation of Eq. 2.31 is precisely the inverse discrete Fourier transform. As a result, one can utilize the inverse Fast Fourier Transform (Cooley and Tukey, 1965) to efficiently evaluate the values of B_c. In this section, we introduce this algorithm.

2.6.1 A priori error estimate

Fix $m \geq 0$. To evaluate $B_c(n, m)$ for arbitrary n, for a prescribed error tolerance ϵ, we need to estimate the size of our trapezoidal rule approximation. Let $B_c^N(n, m)$ denote the N-term trapezoidal rule approximation of $B_c(n, m)$. By Corollary 2.5.3.1, for any $\eta \in (0, c^2/\alpha_1)$, we can have an error estimate:

$$|B_c(n, m) - B_c^N(n, m)| \leq \frac{2M_\eta}{e^{\gamma_\eta(N-n)} - e^{-\gamma_\eta n}} \approx \frac{2M_\eta}{e^{\gamma_\eta(N-n)}}, \tag{2.99}$$

where

$$\gamma_\eta = \log\left(1 + \frac{\eta}{2} + \sqrt{\left(1 + \frac{\eta}{2}\right)^2 - 1}\right), \quad M_\eta = \frac{1}{2\sqrt{c^2/\alpha_1 - \eta}}. \tag{2.100}$$

Without loss of generality, we assume $n = 0$. Theoretically, one can optimize over both η and N to use the fewest number of quadrature points, $N_{opt}(\epsilon)$, to satisfy the error tolerance using the trapezoidal rule approximation. Given an error tolerance ϵ, to solve for $N_{opt}(\epsilon)$, we solve the following optimization problem:

$$\begin{aligned} \arg\min_{N,\eta} \quad & N^2 \\ \text{subject to} \quad & \log\left(\frac{2M_\eta}{\epsilon e^{\gamma_\eta N}}\right) \leq 0 \\ & \eta \in \left(0, \frac{c^2}{\alpha_1}\right). \end{aligned} \tag{2.101}$$

If we denote the resulting solution as $\hat{N}_{opt}(\epsilon)$, then $N_{opt}(\epsilon) = \lceil \hat{N}_{opt}(\epsilon) \rceil$. This problem can be hard to solve numerically. For example, we found that using MATLAB's `fmincon` function with default parameters sometimes diverges. Meanwhile, extensive parameter tuning is undesirable in applications.

Alternatively, one can approximate $N_{opt}(\epsilon)$ using a function $N_{ap}(\epsilon)$ by fixing a small parameter $\delta > 0$ and let $\eta = (1-\delta)^2 c^2/\alpha_1$. The corresponding minimum N to satisfy a prescribed error tolerance ϵ is

$$N_{ap}(\epsilon) = \lceil \hat{N}_{ap} \rceil, \quad \hat{N}_{ap} = \frac{1}{\gamma_\eta} \log\left(\frac{1}{\epsilon (c/\sqrt{\alpha_1}) \sqrt{2\delta - \delta^2}} \right) \tag{2.102}$$

We can bound $\hat{N}_{ap}(\epsilon)$ using $\hat{N}_{opt}(\epsilon)$. Let η_{opt} denote the value of η for which $\hat{N}_{opt}(\epsilon)$ is obtained, and $\eta_{ap} = (1-\delta)^2 c^2/\alpha_1$. We consider two cases: $\eta_{opt} > \eta_{ap}$ and $\eta_{opt} \le \eta_{ap}$.

If $\eta_{opt} > \eta_{ap}$, we have:

$$\begin{aligned} \hat{N}_{opt}(\epsilon) &= \frac{1}{\gamma_{\eta_{opt}}} \log\left(\frac{1}{\epsilon \sqrt{c^2/\alpha_1 - \eta_{opt}}} \right) \\ &\ge \frac{1}{\gamma_{\eta_{opt}}} \log\left(\frac{1}{\epsilon \sqrt{c^2/\alpha_1 - \eta_{ap}}} \right) = \frac{\gamma_{\eta_{ap}}}{\gamma_{\eta_{opt}}} \hat{N}_{ap}(\epsilon) \end{aligned} \tag{2.103}$$

Since $\log(1+x)$ is a concave function, for any $r_\beta \in [0,1]$:

$$\begin{aligned} & r_\beta \log(1+x) = \left(1 - r_\beta\right) \log(1) + r_\beta \log(1+x) \le \log(1 + r_\beta x) \\ \Rightarrow & \frac{\log(1+x)}{\log(1+r_\beta x)} \le \frac{1}{r_\beta}. \end{aligned} \tag{2.104}$$

Since $\eta_{ap} < c^2/\alpha_1$, we have

$$\frac{\gamma_{\eta_{opt}}}{\gamma_{\eta_{ap}}} \le \frac{\gamma_{c^2/\alpha_1}}{\gamma_{\eta_{ap}}} \le \frac{\frac{c^2/\alpha_1}{2} + \sqrt{\left(1 + \frac{c^2/\alpha_1}{2}\right)^2 - 1}}{\frac{\eta_{ap}}{2} + \sqrt{\left(1 + \frac{\eta_{ap}}{2}\right)^2 - 1}} \le \frac{1}{(1-\delta)^2}. \tag{2.105}$$

Thus, we have

$$\hat{N}_{ap}(\epsilon) \le \frac{1}{(1-\delta)^2} \hat{N}_{opt}(\epsilon). \tag{2.106}$$

If $\eta_{opt} \le \eta_{ap}$, we have:

$$\hat{N}_{opt}(\epsilon) \ge \frac{1}{\gamma_{\eta_{ap}}} \log\left(\frac{1}{\epsilon \sqrt{c^2/\alpha_1}} \right) = \frac{\log\left(\frac{1}{\epsilon \sqrt{c^2/\alpha_1}} \right)}{\log\left(\frac{1}{\epsilon \sqrt{c^2/\alpha_1}} \right) + \log\left(\frac{1}{\sqrt{2\delta - \delta^2}} \right)} \hat{N}_{ap}(\epsilon). \tag{2.107}$$

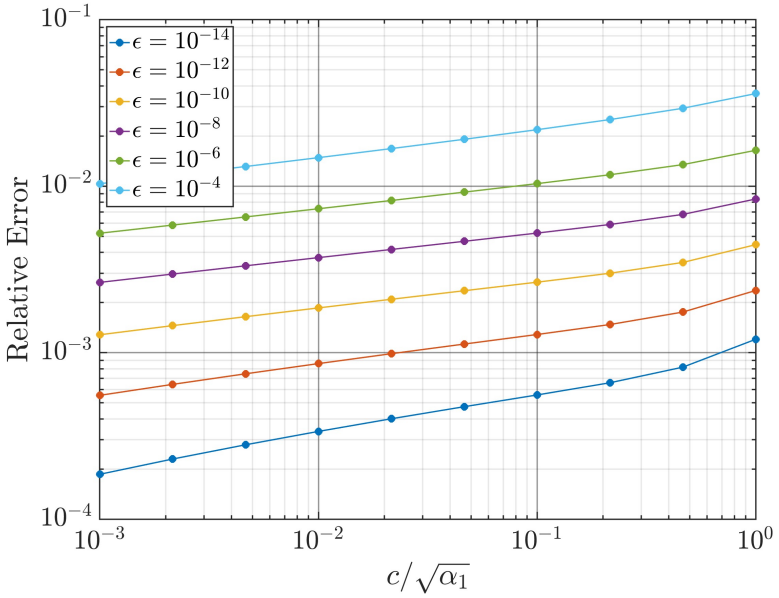


Figure 2.2: Relative error of $\hat{N}_{ap}(\epsilon)$, i.e. $(\hat{N}_{ap}(\epsilon) - \hat{N}_{opt}(\epsilon))/\hat{N}_{opt}(\epsilon)$, across a range of ϵ and $c/\sqrt{\alpha_1}$.

Thus, for this case, we have

$$\hat{N}_{ap}(\epsilon) \leq \left(1 + \frac{\log\left(\frac{1}{\sqrt{2\delta-\delta^2}}\right)}{\log\left(\frac{1}{\epsilon\sqrt{c^2/\alpha_1}}\right)}\right)\hat{N}_{opt}(\epsilon) \tag{2.108}$$

As $\epsilon\sqrt{c^2/\alpha_1} \to 0$, the ratio between the two log functions in the above inequality converges to 0 logarithmically.

In summary, we have obtained that

$$\hat{N}_{ap} \leq \hat{N}_{opt}(\epsilon) \times \max\left(1 + \frac{\log\left(\frac{1}{\sqrt{2\delta-\delta^2}}\right)}{\log\left(\frac{1}{\epsilon\sqrt{c^2/\alpha_1}}\right)}, \frac{1}{(1-\delta)^2}\right). \tag{2.109}$$

In practice, $\hat{N}_{ap}$ is much closer to $\hat{N}_{opt}$. We tuned MATLAB's `fmincon` function to obtain $\hat{N}_{opt}(\epsilon)$ for a set of parameters and compared them with $\hat{N}_{ap}(\epsilon)$ at $\delta = 0.01$. The resulting relative error is shown in Figure 2.2. The figure shows that the relative error decreases as ϵ decreases and $c/\sqrt{\alpha_1}$ decreases. Specifically, practical applications normally require $\epsilon \leq 10^{-8}$. Within this range, the relative error is within 1%.

When $c/\sqrt{\alpha_1}$ becomes small, we can give a straightforward estimate of how $N_{ap}(\epsilon)$ changes as $c/\sqrt{\alpha_1}$ varies. Assuming $c/\sqrt{\alpha_1} \ll 1$, we have

$$\gamma_\eta = \log\left(1 + \frac{\eta}{2} + \sqrt{\left(1+\frac{\eta}{2}\right)^2 - 1}\right) \approx \frac{\eta}{2} + \sqrt{\left(1+\frac{\eta}{2}\right)^2 - 1} \approx \sqrt{\eta}. \tag{2.110}$$

Then we can write

$$\hat{N}_{ap}(\epsilon) \approx \frac{1}{(c/\sqrt{\alpha_1})(1-\delta)}\left(\log\left(\frac{1}{\epsilon\sqrt{2\delta-\delta^2}}\right) + \log\left(\frac{1}{c/\sqrt{\alpha_1}}\right)\right). \tag{2.111}$$

As we can see from the equation above, we have

$$\hat{N}_{ap}(\epsilon) \sim O\left(\frac{-\log(c/\sqrt{\alpha_1})}{(c/\sqrt{\alpha_1})}\right), \tag{2.112}$$

and similarly for $N_{ap}(\epsilon)$.

2.6.2 Fast Fourier Transform based fast evaluation algorithm

Now, we introduce the algorithm to compute B_c. Suppose we would like to compute all the values of $B_c(n,m)$ for a fixed m and a range of $n \in [0, L]$ within some error tolerance ϵ. We first define:

$$f(\theta) = \frac{1}{K^{|m|}}\frac{1}{K - 1/K}, \quad \phi = \lambda + c^2 - 2\alpha_1\cos(\theta), \quad K = \frac{\phi + \sqrt{\phi^2-4}}{2}. \tag{2.113}$$

With these, the algorithm to compute that set of B_c is shown as Algorithm 1. Since $B_c(n,m)$ is real for all n and m, we can utilize the inverse real Fast Fourier

Algorithm 1 FFT-Based LGF Evaluation

procedure TRAPZOIDAL RULE WITH FFT($c, \alpha_1, L, m, \epsilon$)
  Compute $N'_{pts}(\epsilon) = \left\lceil \frac{1}{2}\left(\frac{1}{\gamma_\alpha}\log\left(\frac{1}{\epsilon(c/\sqrt{\alpha_1})\sqrt{2\delta-\delta^2}}\right) + L\right)\right\rceil$
  $N_{pts} = \max(N'_{pts}(\epsilon), L)$
  Declare $\boldsymbol{v} \in \mathbb{R}^{N_{pts}}$
  for k = 0,1,..., $N_{pts} - 1$ **do**
    $\boldsymbol{v}(k) = f(\pi k/N_{pts})$
  end for
  $\boldsymbol{B}$ = irFFT($\boldsymbol{v}$, $2N_{pts}$)
  for k = 0,1,..., $N_{pts} - 1$ **do**
    $\boldsymbol{B}(k) = \boldsymbol{B}(k) + (-1)^k f(\pi)/(2N_{pts})$
  end for
  Return $\boldsymbol{B}$
end procedure

Transform (`irFFT`). Note that to correctly compute B_c using Algorithm 1, we need to set the correct number of output in the `irFFT` function to $2N_{pts}$. Otherwise, the resulting FFT algorithm is different from the trapezoidal rule approximation, and the numerical results do not converge to B_c exponentially. Also, the correction term in the second for-loop results from comparing the formula of `irFFT` against the formula of the trapezoidal rule approximation. The difference, r, is

$$r = \frac{1}{2\pi}\frac{2\pi}{2N_{pts}}\frac{e^{ik\pi}}{K^{|m|}}\frac{1}{K-1/K} = \frac{1}{2N_{pts}}\frac{(-1)^k}{K^{|m|}}\frac{1}{K-1/K} = \frac{(-1)^k}{2N_{pts}}f(\pi). \tag{2.114}$$

We note that for a fixed (n,m), $n \neq L$ and a fixed error tolerance ϵ, the resulting $B_c(n,m)$ value from Algorithm 1 is different from the values obtained by directly evaluating the trapezoidal rule. This is because the number of quadrature points used to evaluate $B(n,m)$ using Algorithm 1 is higher than $N_{ap}(\epsilon)$. The higher number of quadrature points ensures that $B_c(L,m)$ satisfies the required error tolerance.

Algorithm 1 is best applicable when $N_{ap} \approx L$. In that case, the average operation count to evaluate an entry of B_c is

$$OC_{avg} \sim \frac{O(L\log(N_{ap}))}{L} \sim O(\log(N_{ap})). \tag{2.115}$$

By using FFT, one can take advantage of the ubiquity of the highly optimized FFT libraries such as FFTW (Frigo and Johnson, 1998) and cuFFT (NVIDIA Corporation, 2024). Thus, we not only speed up our computation in terms of reduced computational complexity but also benefit from the optimization in the software and hardware aspects. In the case that $N_{ap} \gg L$, depending on the computer architecture, it might be more efficient to directly evaluate the trapezoidal rule approximation term by term. In that case, the average computational complexity is $O(N_{ap})$.

2.7 Numerical experiments

In this section, we assess the performance of the trapezoidal rule approximation with FFT (Algorithm 1) and the direct trapezoidal rule approximation (without FFT) by comparing them to two existing methods: evaluating Appell's double hypergeometric function representation in Thm. 2.2.1 and numerically integrating Eq. 2.8 using Gauss-Kronrod quadrature (Doncker, 1978). All the computations are done on an Apple Silicon M1 chip. The code used in this section is available online*.

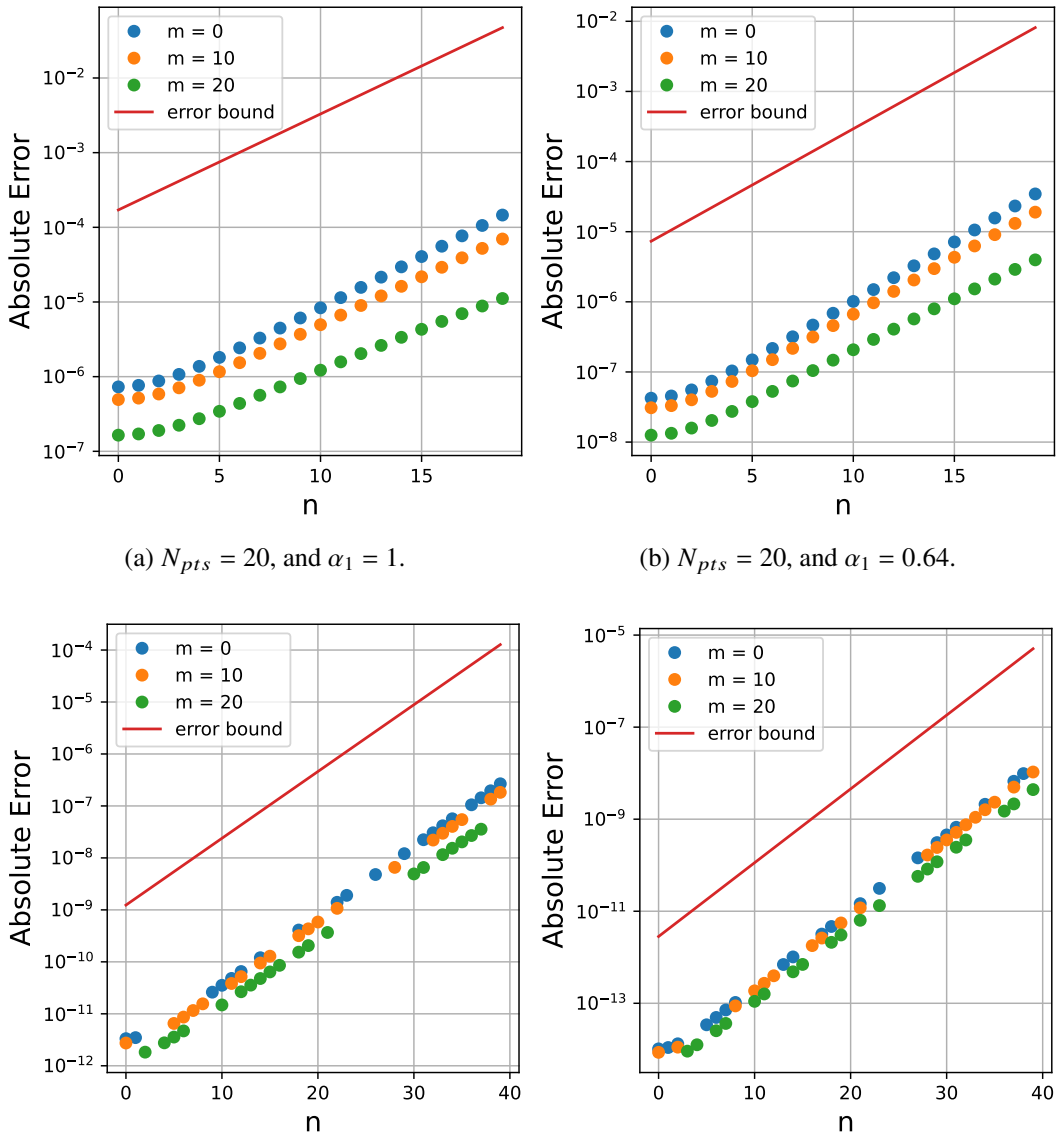
*The code for all the numerical experiments in this section can be found in https://github.com/WeiHou1996/Fast-Screened-Poisson-LGF.

First of all, we demonstrate the error bound Eq. 2.99 of the trapezoidal rule approximation. In Figure 2.3, we compare the trapezoidal rule approximation with the Appell's double hypergeometric function representation Eq. 2.5 at $c = 0.3$ for a range of N_{pts} (the number of quadrature points) and α_1. With only 40 quadrature points in the trapezoidal rule approximation, the error converges to less than 10^{-7} across all the cases. The small errors establish the validity of the trapezoidal rule approximation and our implementation. When $c < 0.3$, we cannot evaluate Appell's double hypergeometric function within a reasonable amount of time. Thus, in that case, we use the trapezoidal rule approximation with a sufficiently high number of quadrature points as the reference value. In Figure 2.4, we demonstrate the error of the trapezoidal rule approximation at $c/\sqrt{\alpha_1} = 0.01$. In two of the subfigures, the absolute error violates the error bound when n is small. However, those errors are below 10^{-13}, indicating the effects of the finite precision arithmetic.

In the exercise above, we used Python's `mpmath` (The mpmath development team, 2023) package to evaluate the Appell's double hypergeometric function, and we used `NumPy` (Harris et al., 2020) to evaluate the direct trapezoidal rule approximation. We do not directly compare the runtime of these two methods as the underlying numerical packages are implemented using different programming languages. However, as a point of reference, the time to evaluate $B_c(n, m)$ for all $(n, m) \in [0, 9]^2$ at $c = 0.3$ using the Appell's double Hypergeometric function representation takes 7.22 seconds while evaluating the trapezoidal rule approximation (without FFT) to an absolute error below 10^{-10} takes 0.00216 seconds.

We also compare our algorithms with numerically integrating the Bessel function representation using Eq. 2.8. Specifically, we used `SciPy`'s `scipy.integrate.quad` function and `scipy.special.ive` function to numerically integrate Eq. 2.8. Integrals over finite ranges are computed using the Gauss-Kronrod quadrature. Integrals with infinite ranges are first mapped onto a finite interval and then evaluated using the Gauss-Kronrod quadrature (Doncker, 1978). We measure the performance of the trapezoidal rule approximation with and without FFT by defining a speedup factor compared to evaluating the LGF using the Bessel function representation. Given a set of values of LGF to compute, the speedup factor for a specific method, $\mathcal{M}$, is:

$$\text{Speedup Factor} = \frac{\text{Runtime using the Bessel function representation}}{\text{Runtime using method } \mathcal{M}}. \tag{2.116}$$



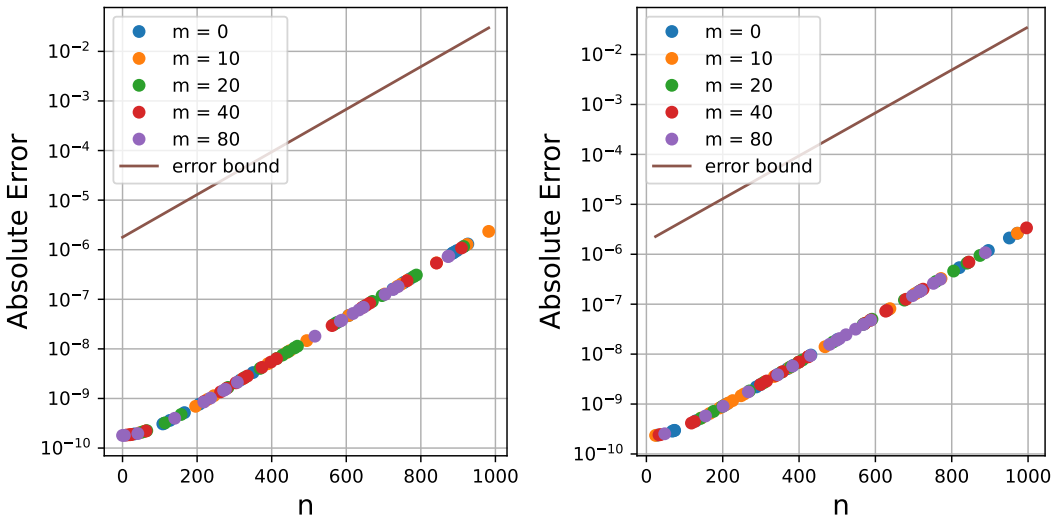
(a) $N_{pts} = 20$, and $\alpha_1 = 1$.

(b) $N_{pts} = 20$, and $\alpha_1 = 0.64$.

(c) $N_{pts} = 40$, and $\alpha_1 = 1$.

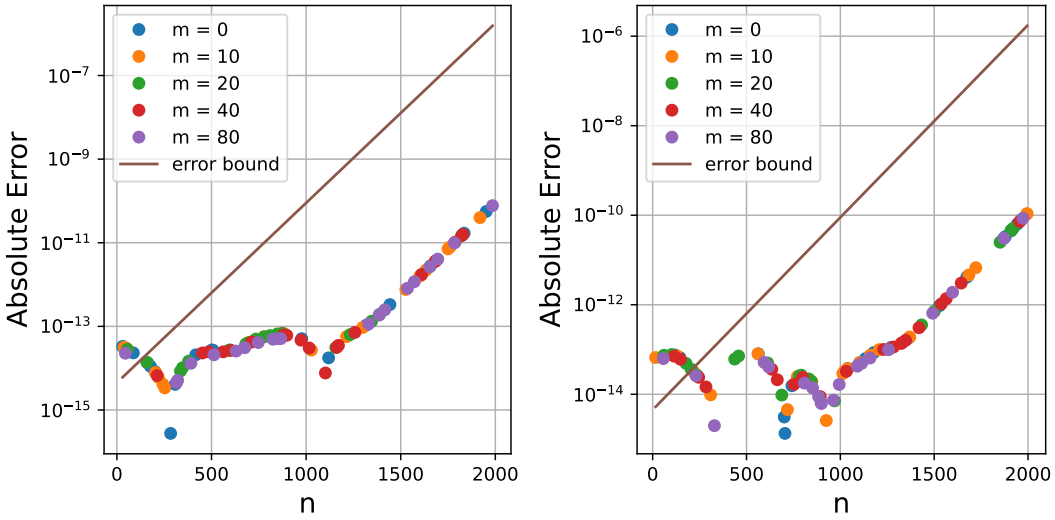
(d) $N_{pts} = 40$, and $\alpha_1 = 0.64$.

Figure 2.3: Error of the trapezoidal rule approximation of $B_c(n, m)$ with various N_{pts}, α_1, n, and m. Across all the cases, $c = 0.3$. The error is computed by referencing the analytical expression using Eq. 2.5. The error bound is computed using Eq. 2.99.



(a) $N_{pts} = 1000$, $\alpha_1 = 1$.

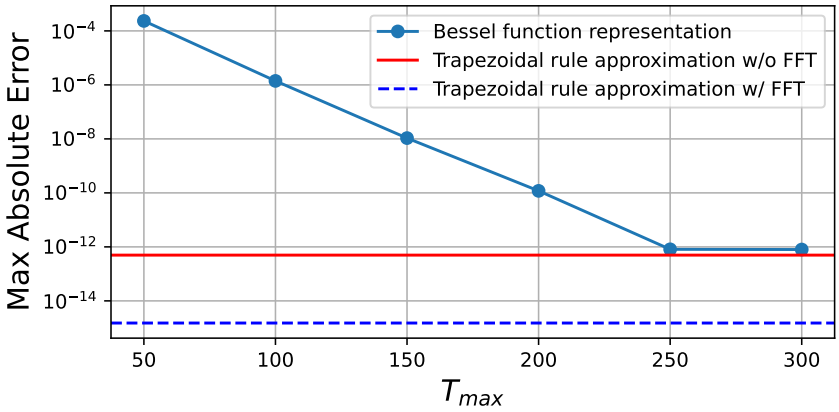
(b) $N_{pts} = 2000$, $\alpha_1 = 0.64$.



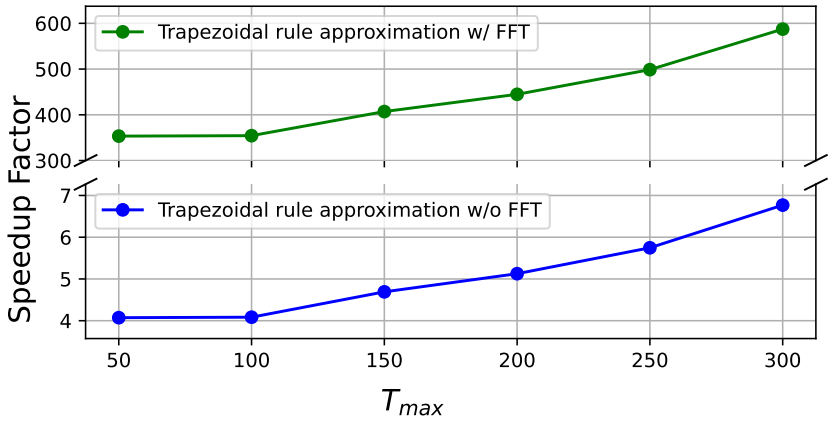
(c) $N_{pts} = 1000$, $\alpha_1 = 1$.

(d) $N_{pts} = 2000$, $\alpha_1 = 0.64$.

Figure 2.4: Error of the trapezoidal rule approximation of $B_c(n, m)$ with various N_{pts}, α_1, n, and m. Across all the cases, $c/\sqrt{\alpha_1} = 0.01$. The error is computed by referencing the trapezoidal rule approximation with $10,000$ quadrature points.



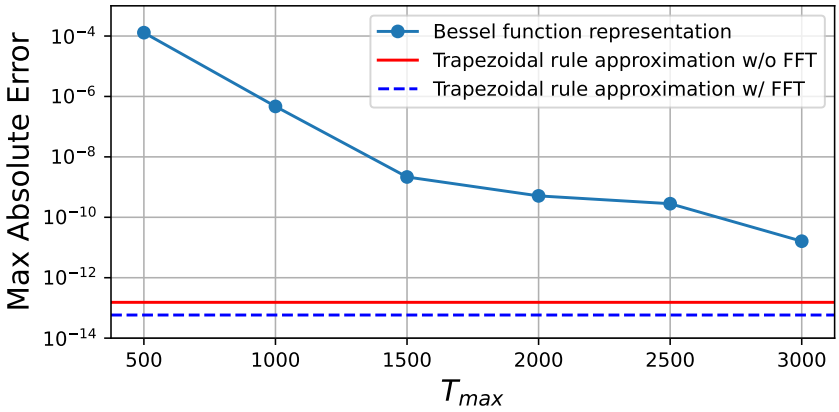
(a) Average absolute error.



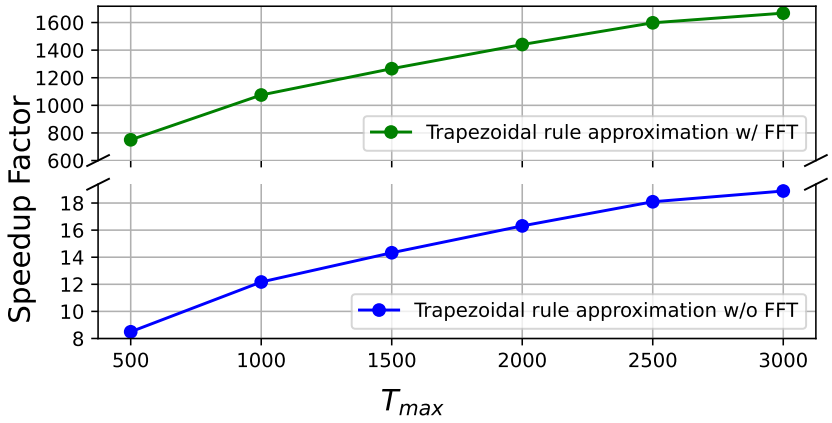
(b) Runtime speedup.

Figure 2.5: The maximum absolute error and speedup factors of computing $B_c(n,m)$ when $c = 0.3$ and $\alpha_1 = 0.5$. The error is computed by comparing the numerical integration results to the trapezoidal rule approximations with $2N_{ap}(\epsilon)$ quadrature points.

In Figs. 2.5, 2.6, and 2.7, we compare our algorithms with numerically integrating the Bessel function representation with a large integration upper bound, presenting both error and speedup. We set $c = 0.3, 0.1, 0.01$ and $\alpha_1 = 0.5$ and evaluate all $B_c(n,m)$ for $(n,m) \in [0,99]^2$ using three methods: numerically integrating the Bessel function representation with large and varying integration upper bounds (T_{max}), evaluating the trapezoidal rule approximation directly without FFT, and evaluating the values of the LGF in batch using FFT (Algorithm 1). In all these methods, the absolute error tolerance is set to 10^{-10}. When $c = 0.3$, the speedup factor is around 6 for the trapezoidal rule approximation without FFT and around 500 for the trapezoidal rule approximation with FFT. When $c = 0.1$, the advantage of the trapezoidal rule approximation is more prominent. The speedup factors are



(a) Average absolute error.

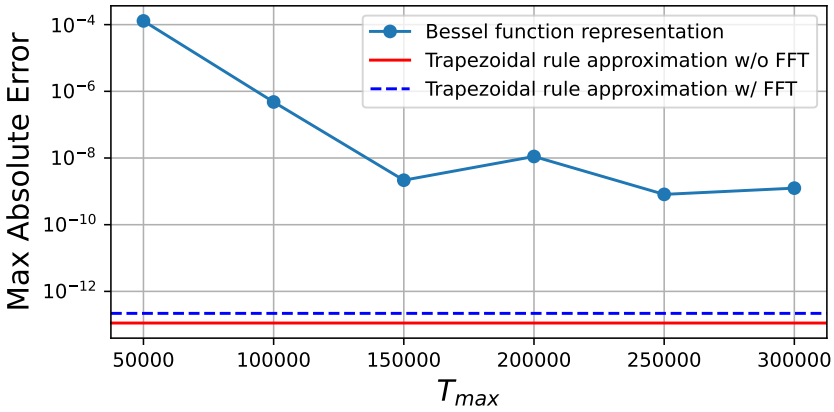


(b) Runtime speedup.

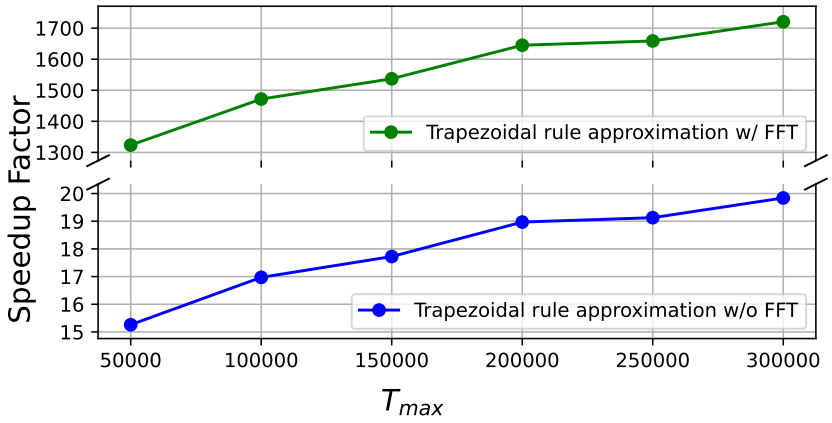
Figure 2.6: The maximum absolute error and speedup factors of computing $B_c(n, m)$ when $c = 0.1$ and $\alpha_1 = 0.5$. The error is computed by comparing the numerical integration results to the trapezoidal rule approximations with $2N_{ap}(\epsilon)$ quadrature points.

around 20 and 1650 without and with FFT, respectively. At $c = 0.01$, numerically integrating the Bessel function representation cannot reach satisfactory accuracy. In contrast, the trapezoidal rule approximations are able to reach the desired accuracy, with significant speedup factors. These three cases demonstrate the efficiency and robustness of the trapezoidal rule approximation and Algorithm 1.

Finally, in Figure 2.8, we present the error and speedup of our algorithm compared to numerically evaluating the Bessel function by mapping the infinite integration interval to a finite one (Doncker, 1978). We fix $\alpha_1 = 0.5$ and vary c between 0.001 and 0.2. We evaluate the values of B_c within the square $[0, 99]^2$ with an absolute error tolerance of 10^{-10}. When evaluating B_c using the Bessel function represen-



(a) Maximum absolute error.

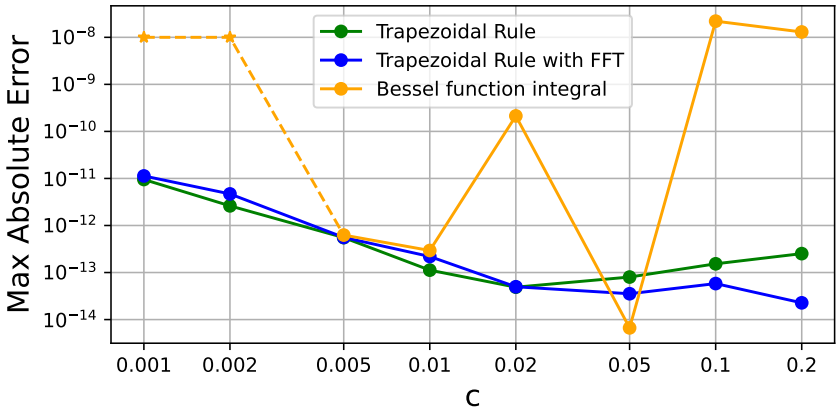


(b) Runtime speedup.

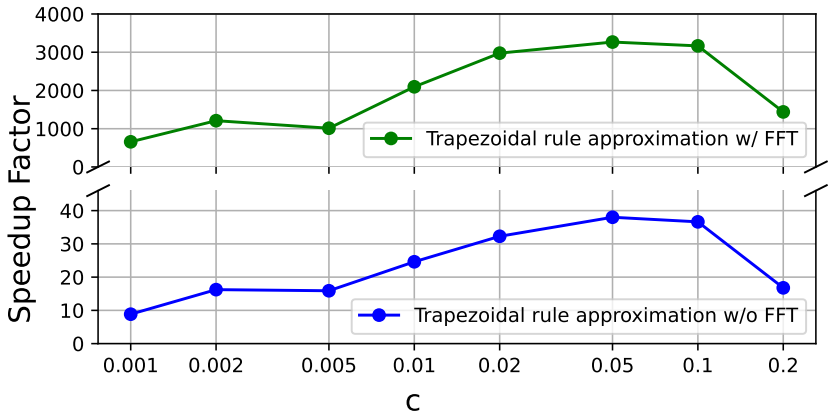
Figure 2.7: The maximum absolute error and speedup factors of computing $B_c(n, m)$ when $c = 0.01$ and $\alpha_1 = 0.5$. The error is computed by comparing the numerical integration results to the trapezoidal rule approximations with $2N_{ap}(\epsilon)$ quadrature points.

tation, the numerical quadrature can diverge when c is small. However, this does not happen with the trapezoidal rule approximation. Also, even when converged, numerically integrating the Bessel function representation does not always satisfy the prescribed error tolerance. In contrast, the trapezoidal rule approximations not only consistently satisfy the error tolerance but also greatly reduce the runtime. The trapezoidal rule approximation reaches a speedup factor of at least 15 without using FFT and a speedup factor of at least 1000 when using FFT.

We note that numerically integrating the Bessel function representation struggles to satisfy the required error tolerance, especially when c is small. The reason is that the integrand in the Bessel function representation, Eq. 2.8, decays at the



(a) Maximum absolute error.



(b) Runtime speedup.

Figure 2.8: The maximum error and speedup factors when computing $B_c(n, m)$ using the trapezoidal rule with FFT, without FFT, and numerically integrating the Bessel function representation using the transform proposed in (Doncker, 1978). The maximum absolute error is obtained by comparing the values from evaluating the trapezoidal rule approximation of B_c using $2N_{ap}(\epsilon)$ quadrature points. The stars and the dashed line in the maximum absolute error plot indicate that some values in the numerical integration did not converge.

rate of $\exp(-c^2 t)/t$ and is highly oscillatory (Abramowitz and Stegun, 1948). As c becomes small, the integral becomes increasingly pathological and difficult to evaluate numerically.

2.8 Application 1: lattice Green's function of the three-dimensional Poisson equation with one periodic direction

An immediate application of B_c is solving the discretized three-dimensional Poisson equation with one periodic direction. Consider a discretization of the three-

dimensional Poisson equation with constant spatial resolutions Δx_1, Δx_2, and Δx_3 in each of the three spatial dimensions. The resulting discretized Poisson equation reads

$$\sum_{i=1}^{3}\left(\frac{2}{\Delta x_i^2}u(\boldsymbol{n})-\frac{1}{\Delta x_i^2}u(\boldsymbol{n}+\boldsymbol{e}_i)-\frac{1}{\Delta x_i^2}u(\boldsymbol{n}-\boldsymbol{e}_i)\right)=f(\boldsymbol{n}). \tag{2.117}$$

In addition, we assume that the solution is periodic in the third direction and unbounded in the first two directions. To solve the Poisson equation, we can find a corresponding LGF and apply discrete convolution. The LGF satisfies:

$$\sum_{i=1}^{3}\left(\frac{2}{\Delta x_i^2}G(\boldsymbol{n})-\frac{1}{\Delta x_i^2}G(\boldsymbol{n}+\boldsymbol{e}_i)-\frac{1}{\Delta x_i^2}G(\boldsymbol{n}-\boldsymbol{e}_i)\right)=\delta^{\mathbb{Z}^3}(\boldsymbol{n}) \tag{2.118}$$

where $\delta^{\mathbb{Z}^d}:\mathbb{Z}^d\to\mathbb{R}$ is defined as:

$$\delta^{\mathbb{Z}^d}(\boldsymbol{n})\begin{cases}1 & \text{if } \boldsymbol{n}=\boldsymbol{0}\\ 0 & \text{otherwise.}\end{cases} \tag{2.119}$$

This equation can be readily solved if we can solve the following equation

$$\sum_{i=1}^{3}\left(2\alpha_i G(\boldsymbol{n})-\alpha_i G(\boldsymbol{n}+\boldsymbol{e}_i)-\alpha_i G(\boldsymbol{n}-\boldsymbol{e}_i)\right)=\delta^{\mathbb{Z}^3}(\boldsymbol{n}) \tag{2.120}$$

where $\alpha_2=1$, $\alpha_1=\Delta x_2^2/\Delta x_1^2$, and $\alpha_3=\Delta x_2^2/\Delta x_3^2$.

Suppose the solution is assumed to be N_p periodic in n_3, with $\boldsymbol{n}=[n_1,n_2,n_3]$, we can write

$$G(\boldsymbol{n})=G(n_1,n_2,n_3)=G(n_1,n_2,n_3+N_p). \tag{2.121}$$

We define the discrete Fourier transform of a N_p periodic discrete function f

$$\tilde{f}_k=\mathcal{F}[f](k)=\sum_{n=0}^{N_p-1}f(n)e^{-i2\pi kn/N_p} \tag{2.122}$$

and its inverse

$$f(n)=\mathcal{F}^{-1}[\tilde{f}](n)=\frac{1}{N_p}\sum_{k=0}^{N_p-1}\tilde{f}_k e^{i2\pi kn/N_p}. \tag{2.123}$$

Thus, there exists a set of Fourier coefficients $\{\tilde{G}_k\}_k$ such that

$$G(n_1,n_2,n_3)=\frac{1}{N_p}\sum_{k=0}^{N_p-1}\tilde{G}_k(n_1,n_2)e^{i2\pi kn_3/N_p}. \tag{2.124}$$

And the LHS of Eq. 2.120 can be written as:

$$\frac{1}{N_p}\left[\sum_{k=0}^{N_p-1} e^{i2\pi k n_3/N_p} L_{\kappa(k)} \tilde{G}_k(n_1, n_2)\right] \tag{2.125}$$

where

$$\kappa(k) = \sqrt{2\alpha_3 - 2\alpha_3 \cos(2\pi k/N_p)}. \tag{2.126}$$

Applying discrete Fourier transform to both sides of Eq. 2.120 gives

$$L_{\kappa(k)} \tilde{G}_k(n_1, n_2) = \delta_{0n_1}\delta_{0n_2}. \tag{2.127}$$

By definition, $\tilde{G}_k(n_1, n_2) = B_{\kappa(k)}(n_1, n_2)$. Thus, we find

$$G(n_1, n_2, n_3) = \frac{1}{N_p}\sum_{k=0}^{N_p-1} e^{i2\pi k n_3/N_p} B_{\kappa(k)}(n_1, n_2). \tag{2.128}$$

Using Algorithm 1 and the approximation in Eq. 2.19, one can efficiently compute the values of $B_{\kappa(k)}$ and evaluate G using inverse Fast Fourier Transform. We use this result to solve a Poisson equation $\nabla^2\phi = -f$ with the following analytical solution

$$\phi(x, y, z) = \frac{\exp(-64x^2 - 4y^2)}{2 - \cos(z)}. \tag{2.129}$$

We obtain the source term f by taking $-\nabla^2\phi$. The computational domain is $[-1, 1]\times[-4, 4]\times[0, 2\pi]$. The convergence result is shown in Figure 2.9.

2.9 Application 2: random walks with killing on a two-dimensional rectangular lattice

Consider a random walk with killing on a two-dimensional rectangular lattice (Madras, 1989; Lawler and Limic, 2010). When the walker is at location (n, m), it can behave in five different ways with certain probabilities. It can either move up, down, left, or right for one step. It can also decide to stay at (n, m) forever (i.e. killed). We assume that the probabilities are all strictly positive and defined by:

$$\begin{aligned}
&\mathbb{P}((n+1, m)) = \mathbb{P}((n-1, m)) = p_1,\\
&\mathbb{P}((n, m+1)) = \mathbb{P}((n, m-1)) = p_2,\\
&\mathbb{P}(\text{stay at } (n, m) \text{ forever}) = p_k := 1 - 2p_1 - 2p_2.
\end{aligned} \tag{2.130}$$

We can compute the probability of a random walk starting at an arbitrary point (n, m) and eventually returning to the origin. Let this probability be denoted as $\rho(n, m)$. We can write

$$\rho(n, m) = p_1\rho(n+1, m) + p_1\rho(n-1, m) + p_2\rho(n, m-1) + p_2\rho(n, m+1) \tag{2.131}$$

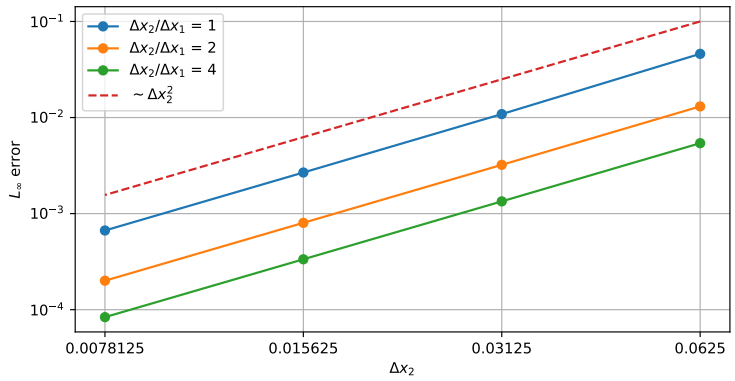


Figure 2.9: Convergence study of solving the Poisson equation using the three-dimensional Poisson LGF with one periodic direction. The ratio $\Delta x_3/\Delta x_2 = 2\pi$ is held constant across all cases. Within each series, the ratio between Δx_1 and Δx_2 is fixed. Different series have different ratios of Δx_2 and Δx_1. The dashed line indicates the expected second-order convergence rate.

with the terminal condition $\rho(0,0) = 1$. The above equation is satisfied everywhere in $\mathbb{Z}^2$ except at the origin. At the origin, we have

$$\rho(0,0) = p_1\rho(1,0) + p_1\rho(-1,0) + p_2\rho(0,-1) + p_2\rho(0,1) + C(p_1,p_2) \quad (2.132)$$

where $C(p_1,p_2)$ is an undetermined function to satisfy the condition $\rho(0,0) = 1$.

With $\alpha_1 = p_1/p_2$, we can rewrite the governing equation of $\rho(n,m)$ as

$$L_{\kappa(p_1,p_2)}\rho(n,m) = \frac{1}{p_2}\delta_{0n}\delta_{0m}C(p_1,p_2) \quad (2.133)$$

where

$$\kappa(p_1,p_2) = \sqrt{\frac{1-2p_1-2p_2}{p_2}}. \quad (2.134)$$

By definition, we have

$$\rho(n,m) = \frac{1}{p_2}C(p_1,p_2)B_{\kappa(p_1,p_2)}(n,m). \quad (2.135)$$

To determine $C(p_1,p_2)$, we use the terminal condition of $\rho(0,0) = 1$ and Eq. 2.132 to find

$$C(p_1,p_2) = \frac{1}{1 + \frac{2p_1}{p_2}B_{\kappa(p_1,p_2)}(1,0) + 2B_{\kappa(p_1,p_2)}(0,1)}. \quad (2.136)$$

In the equation above, we can compute $B_{\kappa(p_1,p_2)}(1,0)$ and $B_{\kappa(p_1,p_2)}(0,1)$ using the integral in Thm. 2.4.1 through the trapezoidal rule approximation in Eq. 2.54. Then

we can compute the return probability at all other locations using either direct trapezoidal rule approximation or Algorithm 1. A sample return probability (with $p_1 = 0.2(1 - p_k), p_2 = 0.3(1 - p_k)$) with various p_k is shown in Figure 2.10.

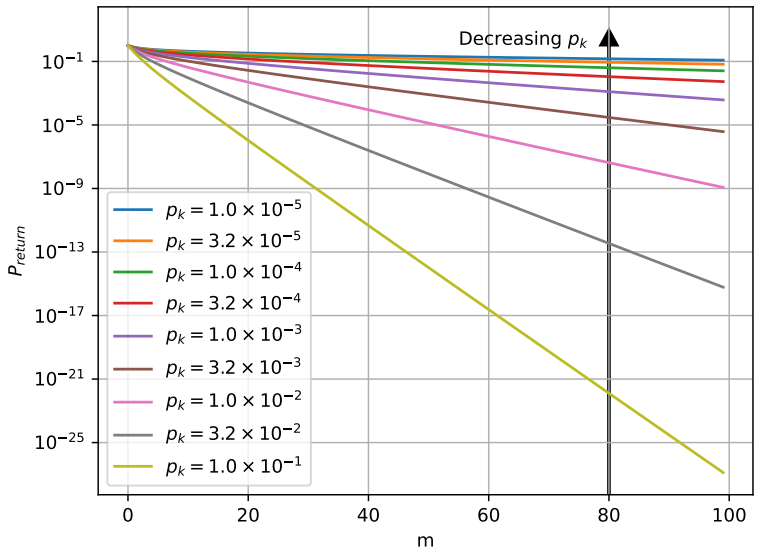
2.10 Conclusion

In this chapter, we studied the two-dimensional lattice Green's function (LGF) of the screened Poisson equation on rectangular lattices. In particular, we proposed two efficient ways to compute the LGF, depending on the c^2 term.

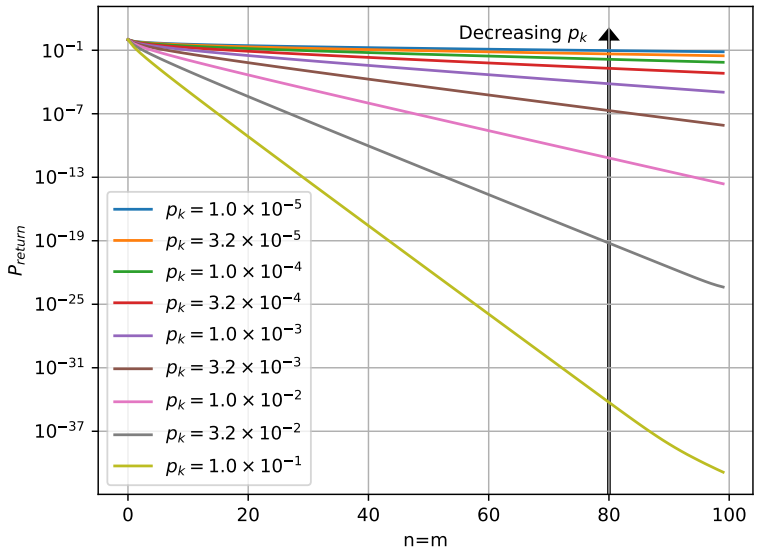
When c^2 is large, we conducted an asymptotic expansion to give an approximation formula of the LGF. We showed that this approximation exponentially converges towards the true values of the LGF. Using the approximation formula, we also established the decay rate of the LGF towards infinity.

Although the asymptotic expansion exponentially converges toward the entries of the LGF, when c^2 is small, approximating LGF using the asymptotic expansion becomes prohibitively expensive. To remedy this, we derived a one-dimensional integral representation of the LGF. In addition, we showed that the trapezoidal rule approximates this one-dimensional integral exponentially fast. By exploiting the properties of the integrand and the trapezoidal rule approximation, we devised a fast algorithm for batch-evaluating the values of the LGF using the Fast Fourier Transform. To enhance the algorithm's robustness, we proposed a simple yet accurate estimate of the minimum number of quadrature points needed for a prescribed error tolerance. Compared to existing formulations such as the Appell's double hypergeometric function representation and the Bessel function representation, the resulting algorithm demonstrates high robustness and efficiency when evaluating the LGF.

Finally, we demonstrated how our algorithms can be efficiently used to tabulate the LGF and solve two application problems – the three-dimensional Poisson equation with two unbounded directions and one periodic direction, and the return probability of a random walk with killing on a rectangular lattice.



(a) Return probability at various m with $n = 0$.



(b) Return probability on the diagonal ($n = m$).

Figure 2.10: The return probability, P_{return}, at various n, m at different kill probabilities p_k.

Chapter 3

AN ADAPTIVE LATTICE GREEN'S FUNCTION METHOD FOR EXTERNAL FLOWS WITH TWO UNBOUNDED AND ONE HOMOGENEOUS DIRECTIONS

This chapter is adapted from Hou and Colonius (2024a). In this chapter, we describe a flow simulation algorithm specially designed for external flows with spanwise periodicity. We leverage the lattice Green's function, adaptive mesh refinement, and immersed boundary method to create an algorithm suitable for handling flows around complex geometries and resolving a large range of length scales from the flow structures.

3.1 Introduction

The lattice Green's function (LGF) is the analytical inverse of a discrete elliptic operator on an unbounded domain (lattice). Due to its value in numerical applications (Liska and Colonius, 2016; Cserti, 2000), its computation (Buneman, 1971; Martinsson and Rodin, 2002; Katsura and Inawashiro, 1971) and asymptotic behavior (Martinsson and Rodin, 2002; Katsura and Inawashiro, 1973) have been studied thoroughly. In addition, the LGF can be adopted to solve the Poisson equation with various boundary conditions (Gabbard and Rees, 2024b; Caprace, Gillis, and Chatelain, 2021; Balty, Chatelain, and Gillis, 2023) and in various dimensions (Caprace, Gillis, and Chatelain, 2021; Balty, Chatelain, and Gillis, 2023). In particular, for incompressible flows, the LGF can be combined with the immersed boundary method (IB) (Peskin, 2002) to create an efficient and parallel algorithm (Liska and Colonius, 2016). Efficiency can be further enhanced with a multilevel LGF framework for adaptive mesh refinement (AMR) (Dorschner et al., 2020; Yu, 2021). In these methods, LGF is used to solve the pressure-Poisson equation and/or vorticity-streamfunction equation. The use of LGF ensures that the solution is defined everywhere in the free space (without imposition of artificial boundary conditions) yet only a finite set of active cells is needed to time-step the flow. This yields a snug, adaptive domain that encloses only the evolving vorticity field, truncated at a small threshold value at the boundary.

However, LGF methods have not been applied to solve the incompressible Naiver-Stokes equations with spanwise periodicity. In practice, a wide class of interesting

geometries and flows exhibit such span-wise periodicity. Among these flows are the flow past bodies with infinite spans and a constant two-dimensional cross-section, such as circular cylinders and unswept airfoils. Although a wide variety of other methods have been developed for these flows (e.g. Dong and Karniadakis (2005), Mittal and Balachandar (1997), and Lehmkuhl et al. (2013)), the multilevel LGF method promises greater computational efficiency while exactly preserving the asymptotic structure of the irrotational outer solution.

In this chapter, we extend the multilevel framework that combines LGF, IB, and AMR (Yu, 2021) to solve fully 3D flows with one homogeneous direction. We exploit the spanwise periodicity by using a Fourier expansion of the flow variables (velocity, pressure, and IB forcing) and derive the evolution equations of the corresponding Fourier coefficients. This formulation enables us to compute the nonlinear convective term via the (dealiased) Fast Fourier Transform (FFT). We develop a staggered-grid strategy that hybridizes the second-order finite-volume discretization for the inhomogenous directions with the Fourier expansion in the homogeneous one, while maintaining desired discrete conservation and other mimetic properties associated with the original 3D finite-volume discretization. In addition, we adaptively truncate Fourier coefficients to make the spanwise resolution consistent with the finite-volume AMR grid.

With one periodic direction, the pressure is determined by a discrete screened Poisson equation for which we derive formulas and algorithms to evaluate the LGF. This particular LGF poses a unique challenge in two aspects. First, the discrete screened Poisson equation involves a continuous coefficient such that the corresponding LGF varies nonlinearly with it. A large number of these LGFs would thus be required, and it is desirable to have a fast way to evaluate them at runtime. In contrast to the regular Poisson equation, the existing polyharmonic asymptotic expansion does not apply to the LGF of the screened Poisson equation (Duffin, 1953; Duffin and Shelly, 1958; Martinsson and Rodin, 2002; Gabbard and Rees, 2024b). Thus, we need to directly compute the LGF through numerical integration. To address both challenges, we adapt the spectrally convergent trapezoidal rule approximation presented in Chapter 2 for fast evaluation of this LGF. Similar challenges are also present in the handling of the viscous term. Thus, we derive and compute the LGF for the appropriate integrating factor for the viscous term. In addition, we provide algorithms for LGF of the integrating factor for the viscous Laplacian (the heat equation kernel), which allows us to employ a half-explicit Runge-Kutta (IF-HERK) method

for time advancement (Liska and Colonius, 2017).

The chapter is arranged as follows. We introduce the Fourier-transformed Navier-Stokes equations with IB forcing in Section 3.2. Then, in Section 3.3, we derive a spatial discretization in terms of corresponding discrete operators. In Section 3.4, we develop LGFs for the screened Poisson operator and integrating factor. In the next sections, we adapt several previous algorithms to the spanwise homogenous case, specifically the time marching method (Section 3.5), multilevel application of the LGF (Section 3.6), and the domain and mesh adaptation strategies (Section 3.7). The overall algorithm for the incompressible Navier-Stokes equations is then summarized in Section 3.8. Subsequently, in Section 3.9, we describe the parallelization strategy computational efficiency. In Section 3.10, we empirically demonstrate the (overall first-order) convergence of the scheme. Finally, in Section 3.11, we validate the algorithm by computing flow past a cylinder at $Re = 300$, and we highlight the ability of our algorithm to tackle large problems by computing the turbulent flow past a cylinder at $Re = 12,000$.

3.2 Governing equations and Fourier expansion

Physically, the problem under consideration is an infinite-span cylinder (axis z) of arbitrary cross-section moving (including acceleration) in the $x-y$ plane through an otherwise quiescent, viscous, incompressible fluid. Invoking homogeneity, we restrict z to a periodic section of a specified length c. For real-valued $f = f(x, y, z, t)$, we write the truncated Fourier series

$$f(x,y,z,t) \approx \tilde{\boldsymbol{f}}_0(x,y,t) + \sum_{k=1}^{N/2} \left[\tilde{\boldsymbol{f}}_k(x,y,t) e^{2\pi i k z/c} + \overline{\tilde{\boldsymbol{f}}_k}(x,y,t) e^{-2\pi i k z/c} \right], \quad (3.1)$$

where

$$\tilde{f}_k(x,y,t) = \mathcal{F}_k[f] := \frac{1}{c} \int_{-c/2}^{c/2} f(z) e^{-iz\frac{2\pi k}{c}} dz. \quad (3.2)$$

Let $\boldsymbol{u}$ and $\boldsymbol{\omega} = \nabla \times \boldsymbol{u}$ be the velocity and vorticity, and p be the pressure, all nondimensionalized with respect to a specified velocity scale, length scale, and density. In physical space, the incompressible Navier-Stokes equations with the IB formulation are (Liska and Colonius, 2017)

$$\begin{aligned}
\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{\omega} \times \boldsymbol{u}_a &= -\nabla \Pi + \frac{1}{Re}\nabla^2 \boldsymbol{u} + \int_{\Gamma(t)} \boldsymbol{f}_\Gamma(\boldsymbol{\xi}, t)\delta(\boldsymbol{X}(\boldsymbol{\xi}, t) - \boldsymbol{x})d\boldsymbol{\xi}, \\
\nabla \cdot \boldsymbol{u} &= 0, \\
\boldsymbol{u}_\Gamma(\boldsymbol{\xi}, t) &= \int_{\mathbb{R}^3} \boldsymbol{u}(\boldsymbol{x}, t)\delta(\boldsymbol{x} - \boldsymbol{X}(\boldsymbol{\xi}, t))d\boldsymbol{x}.
\end{aligned} \tag{3.3}$$

Here, $\boldsymbol{x}$ and $\boldsymbol{x}_a = \boldsymbol{x} - \boldsymbol{R}(t)$ denote the coordinates in the inertial reference frame and those in the non-inertial reference frame, respectively. The non-inertial frame translates with the trajectory $\boldsymbol{R}(t)$ and rotates with angular velocity $\boldsymbol{\Omega}(t)$. $\boldsymbol{u}$ is the velocity vectors in the inertia reference frame. $\boldsymbol{u}_a$ is the velocity vector with respect to the non-inertial reference frame. The two velocities are related through $\boldsymbol{u} = \boldsymbol{u}_a + \boldsymbol{u}_r$ where $\boldsymbol{u}_r = \frac{d\boldsymbol{R}(t)}{dt} + \boldsymbol{\Omega}(t) \times \boldsymbol{x}_a := \boldsymbol{U}(t) + \boldsymbol{\Omega}(t) \times \boldsymbol{x}_a$. In this equation, $\frac{\partial}{\partial t}$ is the derivative in t with $\boldsymbol{x}_a$ held constant, and ∇ the gradients with respect to $\boldsymbol{x}_a$. Correspondingly, $\Pi = p - \frac{1}{2}|\boldsymbol{u}_r|^2 - \frac{1}{2}|\boldsymbol{u}_a|^2$ where p is the pressure.

If we denote the immersed boundary points in the non-inertial frame as $\boldsymbol{X}_a(\boldsymbol{\xi}, t)$, we can rewrite the boundary condition

$$\boldsymbol{u}_{\Gamma,a}(\boldsymbol{\xi}, t) + \boldsymbol{U}(t) + \boldsymbol{\Omega}(t) \times \boldsymbol{X}_a(\boldsymbol{\xi}, t) = \int_{\mathbb{R}^3} \boldsymbol{u}(\boldsymbol{x}, t)\delta(\boldsymbol{x} - \boldsymbol{X}(\boldsymbol{\xi}, t))d\boldsymbol{x}. \tag{3.4}$$

Note that the convolution integral is taken in the inertial coordinates. The Dirac delta function uses the relative position between the immersed boundary surface and the coordinates in the inertial reference frame. Thus, we are only sampling inertial frame velocity on the immersed boundary and equate the values to the inertial frame velocity boundary condition, on each single point parameterized by $\boldsymbol{\xi}$ and t.

In Fourier space, these equations read

$$\begin{aligned}
\frac{\partial \tilde{\boldsymbol{u}}_k}{\partial t} + \mathcal{F}_k[\boldsymbol{\omega} \times \boldsymbol{u}_a] &= -\widetilde{\nabla}_k \tilde{\Pi}_k + \frac{1}{Re}\widetilde{\nabla^2}_k \tilde{\boldsymbol{u}}_k \\
&\quad + \int_{\Gamma(t)_{2D}} \tilde{\boldsymbol{f}}_{\Gamma,k}(\boldsymbol{\xi}_{2D}, t)\delta_{2D}(\boldsymbol{X}_{2D}(\boldsymbol{\xi}_{2D}, t) - \boldsymbol{x}_{2D})d\boldsymbol{\xi}_{2D}, \\
\widetilde{\nabla}_k . \tilde{\boldsymbol{u}}_k &= 0, \\
\widetilde{\boldsymbol{u}}_{\Gamma,k}(\boldsymbol{\xi}_{2D}, t) &= \int_{\mathbb{R}^2} \tilde{\boldsymbol{u}}_k(\boldsymbol{x}_{2D}, t)\delta_{2D}(\boldsymbol{x}_{2D} - \boldsymbol{X}_{2D}(\boldsymbol{\xi}_{2D}, t))d\boldsymbol{x}_{2D},
\end{aligned} \tag{3.5}$$

where

$$\widetilde{\nabla}_k . \tilde{\boldsymbol{u}}_k = \frac{\partial \tilde{u}_k}{\partial x} + \frac{\partial \tilde{v}_k}{\partial y} + \frac{2\pi i k}{c}\tilde{w}_k, \tag{3.6}$$

$$\widetilde{\nabla}_k \tilde{\Pi}_k = \left[\frac{\partial \tilde{\Pi}_k}{\partial x}, \frac{\partial \tilde{\Pi}_k}{\partial y}, \frac{2\pi i k}{c}\tilde{\Pi}_k\right]^T, \tag{3.7}$$

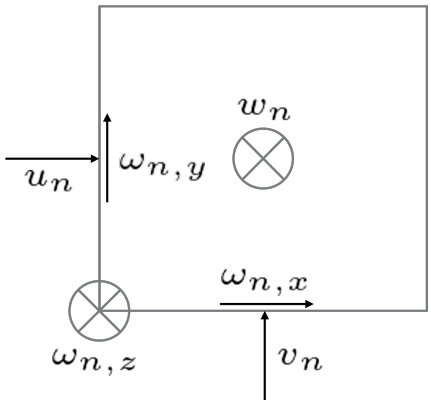


Figure 3.1: Variable placement in the $x-y$ plane for a Fourier interpolation sampling point in the z direction.

$$\widetilde{\nabla}_k^2 \tilde{u}_k = \frac{\partial^2 \tilde{u}_k}{\partial x^2} + \frac{\partial^2 \tilde{u}_k}{\partial y^2} - \left(\frac{2\pi k}{c}\right)^2 \tilde{u}_k. \tag{3.8}$$

Details for the Fourier transform of the IB terms are given in Appendix A.

Since the body and flow are homogeneous in the z direction, we simplify the boundary condition equations to

$$\begin{aligned} \boldsymbol{u}_\Gamma(\boldsymbol{\xi}_{2D}, t) &= \int_{\mathbb{R}^2} \tilde{\boldsymbol{u}}_0(\boldsymbol{x}_{2D}, t)\delta_{2D}(\boldsymbol{x}_{2D} - \boldsymbol{X}_{2D}(\boldsymbol{\xi}_{2D}, t))d\boldsymbol{x}_{2D} \quad k = 0, \\ 0 &= \int_{\mathbb{R}^2} \tilde{\boldsymbol{u}}_k(\boldsymbol{x}_{2D}, t)\delta_{2D}(\boldsymbol{x}_{2D} - \boldsymbol{X}_{2D}(\boldsymbol{\xi}_{2D}, t))d\boldsymbol{x}_{2D} \quad k \neq 0. \end{aligned} \tag{3.9}$$

We can evaluate the nonlinear term using a de-aliased pseudo-spectral approach (Orszag, 1971b), i.e. we discretize the inverse transform to the DFT, form the product in physical space via padded inverse transforms, and transform the product back to its truncated Fourier coefficients. Padding via the $3/2$ rule is sufficient since the equations involve at most quadratic nonlinearity.

3.3 Spatial discretization

To develop a framework that is best suited for the hybridized discretization, we place all finite-volume cells with their centers aligned with the evenly-spaced sampling points for Fourier interpolation. The x and y components of velocity are placed on the faces, and the z component of velocity is at the cell center. One can visualize the resulting data field as a collection of evenly-spaced slices that appear as a 2D finite-volume method depicted in Figure 3.1.

With this staggered mesh, we now define discrete operators and enumerate some of their properties. In this section, we use boldfaced variables, e.g. $\boldsymbol{g} = [g_1, g_2, g_3]^T$,

to denote 3-component vector fields and non-boldfaced variables, e.g. g, to denote scalar fields. The operators are:

- Divergence on k^{th} Fourier coefficient:

$$D_k \boldsymbol{g} = \mathcal{D}_x g_1 + \mathcal{D}_y g_2 + (2\pi i k/c) g_3. \tag{3.10}$$

- Gradient on k^{th} Fourier coefficient:

$$G_k g = [-\mathcal{D}_x^T g, -\mathcal{D}_y^T g, (2\pi i k/c) g]^T. \tag{3.11}$$

- Curl on k^{th} Fourier coefficient:

$$C_k \boldsymbol{g} = [-\mathcal{D}_y^T g_3 - (2\pi i k/c) g_2, (2\pi i k/c) g_1 + \mathcal{D}_x^T g_3, -\mathcal{D}_x^T g_2 + \mathcal{D}_y^T g_1]^T. \tag{3.12}$$

- Laplacian on k^{th} Fourier coefficient:

$$L_k g = -\mathcal{D}_x^T \mathcal{D}_x g - \mathcal{D}_y^T \mathcal{D}_y g - (2\pi k/c)^2 g. \tag{3.13}$$

In the equations above, $\mathcal{D}$ denotes a forward finite difference derivative, for example, $\mathcal{D}_x g(i,j) = [g(i+1,j) - g(i,j)]/\Delta x$. The operators mimic some properties of the continuous operators, namely $D_k = -G_k^*$ and $L_k = -G_k^* G_k$ (where the superscript * denotes the conjugate transpose).

Let $\boldsymbol{u}_{a,n} = [u_n, v_n, w_n]^T$ be the velocity in the non-inertial frame at the n^{th} slice and $\boldsymbol{\omega}_n = [\omega_{n,x}, \omega_{n,y}, \omega_{n,z}]^T$ be the vorticity on that slice, we discretize the nonlinear advection terms in rotational form by defining $\boldsymbol{N}_n(i,j) = [N_{n,x}(i,j), N_{n,y}(i,j), N_{n,z}(i,j)]^T$, and writing

$$\begin{aligned}
N_{n,x}(i,j) =& \frac{1}{2}\omega_{n,y}(i,j)[w_n(i,j) + w_n(i-1,j)] \\
&- \frac{1}{4}\omega_{n,z}(i,j)[v_n(i,j) + v_n(i-1,j)] \\
&- \frac{1}{4}\omega_{n,z}(i,j+1)[v_n(i,j+1) + v_n(i-1,j+1)], \\
N_{n,y}(i,j) =& \frac{1}{4}\omega_{n,z}(i,j)[u_n(i,j) + u_n(i,j-1)] \\
&+ \frac{1}{4}\omega_{n,z}(i+1,j)[u_n(i+1,j) + u_n(i+1,j-1)] \\
&- \frac{1}{2}\omega_{n,x}(i,j)[w_n(i,j) + w_n(i,j-1)], \\
N_{n,z}(i,j) =& \frac{1}{2}[\omega_{n,x}(i,j) v_n(i,j) + \omega_{n,x}(i,j+1) v_n(i,j+1)] \\
&- \frac{1}{2}[\omega_{n,y}(i,j) u_n(i,j) + \omega_{n,y}(i+1,j) u_n(i+1,j)].
\end{aligned} \tag{3.14}$$

Let $\boldsymbol{N}(\boldsymbol{\omega}, \boldsymbol{u})$ be the collection of the nonlinear advection term across all slices, evaluated using $\boldsymbol{\omega}, \boldsymbol{u}$ with Eq. 3.14. Inserting the discrete spatial operators in Eq. 3.5, we obtain a system of (index 2) differential-algebraic equations (DAE)

$$\begin{aligned} \frac{d\tilde{\boldsymbol{u}}_k}{dt} + \mathcal{F}_k[\boldsymbol{N}(\boldsymbol{\omega}, \boldsymbol{u})] &= -G_k \tilde{q}_k + \frac{1}{Re} L_k \tilde{\boldsymbol{u}}_k + P(t)^T \tilde{\boldsymbol{f}}_k, \\ D_k \tilde{\boldsymbol{u}}_k &= 0, \\ P(t)\tilde{\boldsymbol{u}}_k &= \widetilde{\boldsymbol{u}}_{\Gamma,k}, \end{aligned} \tag{3.15}$$

where q is a pressure-like variable. In this equation, the Fourier coefficients of the nonlinear term are evaluated with the pseudo-spectral approach discussed above. $P(t)$ is the IB interpolation operator. There are second- and higher-order accurate discretizations of this operator (Gabbard, Gillis, et al., 2022; Ji, Gabbard, and Rees, 2023; Gabbard and Rees, 2024a; Verzicco, 2023), but considering compatibility with the exact imposition of the boundary conditions and mimetic properties of the finite-volume discretization, together with the availability of AMR, we prefer the first-order-accurate discrete delta function approach (Peskin, 2002). In addition, when the immersed body is time-invariant, the linear system associated with the IB forcing possesses the desired Hermitian property (later shown in Section 3.5 and Appendix C). Any discrete delta function can be used in the formulation; the calculations below utilized a three-point delta function (Roma, Peskin, and Berger, 1999)

$$\phi(r) = \begin{cases} 1 + \sqrt{-3r^2 + 1}, |r| < 0.5, \\ \frac{1}{6}(5 - 3|r| - \sqrt{1 - 3(1 - |r|)^2}), |r| \in [0.5, 1.5), \\ 0 \text{ otherwise.} \end{cases} \tag{3.16}$$

3.4 Lattice Green's functions

3.4.1 Lattice Green's function of L_k

In solving Eq. 3.15, substituting the momentum equation in the divergence-free constraint gives rise to an inhomogeneous screened Poisson equation, $L_k u = f$ that must be solved at each time sub-step. We utilize the lattice Green's function (LGF) on a formally infinite grid to solve this system. For each k, we can find a LGF, $B_k : \mathbb{Z}^2 \to \mathbb{R}$, of the operator L_k such that

$$(L_k B_k)(n, m) = (\Delta x)^2 \delta^{\mathbb{Z}}(n - m), \quad \lim_{n,m \to \infty} B_k(n, m) = 0 \tag{3.17}$$

where $\delta^{\mathbb{Z}} : \mathbb{Z} \to \{0, 1\}$ is the Kronecker delta function and defined as:

$$\delta^{\mathbb{Z}}(n) = \begin{cases} 1 & \text{if } n = 0, \\ 0 & \text{if } n \neq 0. \end{cases} \tag{3.18}$$

As a result, we can solve the inhomogeneous screened Poisson equation using this LGF (Katsura and Inawashiro, 1971; Martinsson and Rodin, 2002):

$$L_k u = f \implies u(n, m) = (L_k^{-1} f)(n, m) := \sum_{a,b} (\Delta x)^2 B_k(n - a, m - b) f(a, b). \tag{3.19}$$

We call the sequence $\{B_k(n, m)\}_{n,m}$ the LGF kernel.

When $k \neq 0$, we can write the LGF kernel as (Buneman, 1971):

$$B_k(n, m) - B_k(0, 0) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left(1 - \frac{e^{i\theta m}}{K^{|n|}}\right) \frac{d\theta}{K - 1/K} \tag{3.20}$$

where

$$K = \frac{a + \sqrt{a^2 - 4}}{2}, \tag{3.21}$$

and

$$a = \left(4 + \left(\frac{2\pi k \Delta x}{c}\right)^2 - 2\cos(\theta)\right). \tag{3.22}$$

Finally,

$$B_k(0, 0) = \frac{1}{2b} {}_2F_1\left(\frac{1}{2}, \frac{1}{2}; 1; \left(\frac{2}{b}\right)^2\right), \tag{3.23}$$

where $b = 2 + 2\left(\frac{\pi k \Delta x}{c}\right)^2$. We note that the integral in Eq. 3.20 is increasingly oscillatory with increasing m. However, as the integrand is periodic, it can be approximated with spectral convergence using the trapezoidal rule (Trefethen and Weideman, 2014). Further, one can show that the number of quadrature points needed to evaluate this integral at most increases linearly with m. However, we do note that the spectral convergence rate of the numerical approximation is dictated by a. Specifically, as we show in Chapter 2, the approximation error of an N-point trapezoidal rule approximation (ϵ_N) is bounded by

$$|\epsilon_N| \leq \frac{2M}{e^{\gamma_c N} - 1}, \tag{3.24}$$

where γ_c is any positive real number satisfying:

$$\gamma_c < \log\left(1 + \frac{\alpha^2}{2} + \sqrt{\left(1 + \frac{\alpha^2}{2}\right)^2 - 1}\right) := \kappa_c, \qquad \alpha = \frac{2\pi k \Delta x}{c}, \tag{3.25}$$

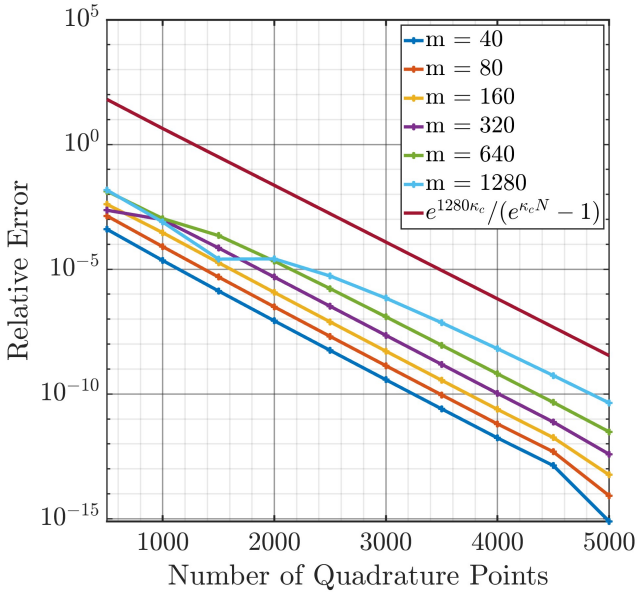


Figure 3.2: Convergence of $B(n,m)$ for $n = m = 1280$ and various n compared with the analytical convergence rate.

and M is

$$M = \sup_{|\Im(\theta)|<\gamma_c} \left| (1 - \frac{e^{i\theta m}}{K^{|n|}}) \frac{1}{K - 1/K} \right|. \tag{3.26}$$

In our implementation, this integral is evaluated for each k, m, and n using the adaptive trapezoidal rule (Schäling, 2011) with relative tolerance of 10^{-10}. This algorithm halves the integration step size (i.e. double the quadrature points) until the tolerance is achieved. The convergence of this approximation is empirically demonstrated in Figure 3.2. The threshold 10^{-10} is chosen to achieve sufficient precision in subsequent computations with the LGF without excessive computational expense or the need to invoke quad-precision arithmetic. Due to the periodicity of the integrand, the relative error is typically much less than the threshold 10^{-10}, often reaching the machine epsilon (Schäling, 2011).

To determine the required number of quadrature points for practical computations, we consider a worst-case scenario. First, consider that K monotonically increases with α, which also monotonically increases with $k\Delta x/c$. We thus consider the case where $k = 1$ and $c/\Delta x = 1200$ which provides a conservative estimate for the lowest value of $k\Delta x/c$ likely to be encountered in applications. To further simplify the matter, we observe that $B(n,m) = B(|n|,|m|)$, so, without loss of generality, we can assume $n, m \geq 0$. In addition, since $K > 1$, the higher the value n is, the greater $K^{|n|}$, and the smaller the oscillatory term in the integrand. Thus, the worst case happens

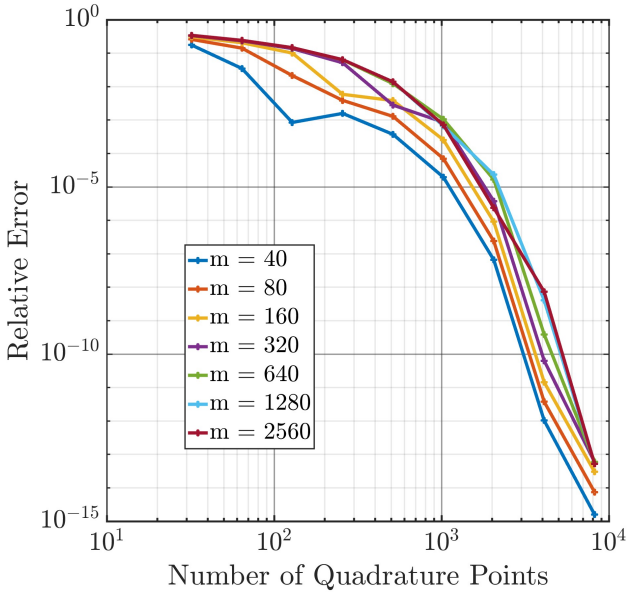


Figure 3.3: Spectral convergence of $B(m, m)$ for various m.

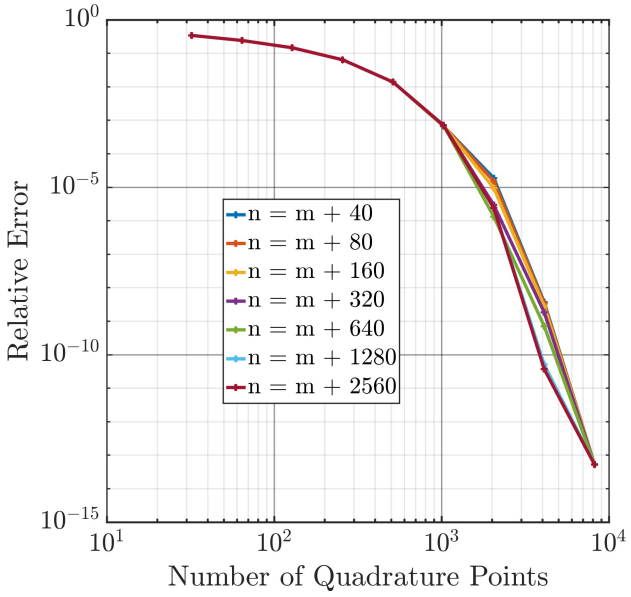


Figure 3.4: Convergence of $B(n, m)$ for $m = 1280$ and various n.

when m is large and n is small. However, since $B(m, n) = B(n, m)$ (Katsura and Inawashiro, 1971), we can always write $B(n, m) = B(\max(n, m), \min(n, m))$. Thus the oscillation of the integrand is the most severe when $m = n$ and they are both large. Computed errors for a range of m are shown in Figure 3.3. To further illustrate the point that increasing n when holding m constant will not exacerbate the effects of oscillation during numerical integration, we hold $m = 1280$ and vary n in Figure 3.4.

Returning now to the case $k = 0$, the far-field boundary condition imposed on B_k is not achievable, since the fundamental solution logarithmically diverges in 2D (Martinsson and Rodin, 2002). However, eliminating this term provides an inverse that is unique up to a constant. This constant can be absorbed into the pressure and need never be determined explicitly. However, when we introduce the multilevel LGFs (Section 3.6), we shall need to impose a compatibility constraint so that the arbitrary constant is the same regardless of resolution. Let B_0^l denote the kernel on the l^{th} refinement level (0 is the coarsest grid), we can write the compatibility condition as the following:

$$B_0^l(\boldsymbol{n}) = B_0^0(\boldsymbol{n}) - \frac{l}{2\pi}\log(2) \qquad \forall \boldsymbol{n} \in \mathbb{Z}^2. \tag{3.27}$$

We detail the derivation of this compatibility condition in Appendix B.

With a compact source term, L_k^{-1} provides the solution at any point on an infinite lattice. However, to march the solution (and the source) to the next time, we only need to evaluate the action of applying L_k^{-1} on the support of its source (including a buffer region to allow the solution to adapt). To ensure the accuracy of the solution and adapt to the evolving vortical flow region, we adopt the domain adaptation and "velocity refresh" algorithms developed for the 3D inhomogeneous case (Liska and Colonius, 2016; Liska and Colonius, 2017). Further details of these techniques will be provided in Section 3.7.

To accelerate the application of L_k^{-1}, we adopt an interpolation-based kernel-independent fast multipole method on a block-wise decomposed grid (Liska and Colonius, 2014). This algorithm not only achieves linear complexity but also lends itself to efficient parallelization.

3.4.2 Integrating factors ($E_{k,n}^i$)

The availability of the LGF provides an opportunity to use an integrating factor to march the viscous term without an associated time step restriction, enabling the application of an RK-type explicit DAE solver (Liska and Colonius, 2016). To implement this in the present method requires finding the LGF for the integrating factor, $H_k(t)$, that solves the following linear ODE system on an infinite lattice:

$$\frac{dH_k(t)}{dt} = \frac{1}{Re} L_k H_k(t), \quad H_k(0) = I, \tag{3.28}$$

where I is the identity operator.

We first denote the kernel of $H_0(t)$ as $A_0(t)$, which can be written as(Liska and Colonius, 2017):

$$A_0(t)(\boldsymbol{n}) = \frac{1}{4\pi^2}\int_{\Pi} \exp\left(-i\boldsymbol{n}.\boldsymbol{\xi} + \frac{\sigma(\boldsymbol{\xi})t}{Re\Delta x^2}\right) d\boldsymbol{\xi}$$
$$= \prod_{q\in\boldsymbol{n}}\left[\exp\left(\frac{-2t}{Re\Delta x^2}\right) I_q(\frac{2t}{Re\Delta x^2})\right], \tag{3.29}$$

where $\sigma(\boldsymbol{\xi}) = 2\cos(\xi_1) + 2\cos(\xi_2) - 4$, $\Pi = (-\pi,\pi)^2$, and $I_n(z)$ is the modified Bessel function of the first kind of order n. For $k \neq 0$, H_k can be found in terms of H_0. The resulting expression is

$$H_k(t) = \exp\left[-\left(\frac{2\pi k}{c}\right)^2 \frac{t}{Re}\right] H_0(t). \tag{3.30}$$

This solution can be verified as follows. The IC follows by evaluating the expression at $t = 0$

$$H_k(0) = \exp(0)H_0(0) = I, \tag{3.31}$$

and Eq. (3.28) follows by

$$\frac{dH_k}{dt} = \exp\left[-\left(\frac{2\pi k}{c}\right)^2 \frac{t}{Re}\right] \frac{dH_0}{dt} - \left(\frac{2\pi k}{c}\right)^2 \frac{1}{Re} \exp\left[-\left(\frac{2\pi k}{c}\right)^2 \frac{t}{Re}\right] H_0 \tag{3.32}$$
$$= \exp\left[-\left(\frac{2\pi k}{c}\right)^2 \frac{t}{Re}\right] \frac{1}{Re} L_0 H_0 - \left(\frac{2\pi k}{c}\right)^2 \frac{1}{Re} H_k \tag{3.33}$$
$$= \frac{1}{Re} L_0 H_k - \left(\frac{2\pi k}{c}\right)^2 \frac{1}{Re} H_k \tag{3.34}$$
$$= \frac{1}{Re}\left[L_0 - \left(\frac{2\pi k}{c}\right)^2 I\right] H_k, \tag{3.35}$$
$$\equiv \frac{1}{Re} L_k H_k. \tag{3.36}$$

Note that the kernel associated with $H_k(t)$ decays faster than any exponential, which can be exploited in the fast multipole solution by restricting the source of any target to its vicinity (Liska and Colonius, 2016).

3.5 Temporal discretization

To discretize Eq. 3.15 in time, by imposing the integrating factor, we can rewrite the system by denoting $\tilde{\boldsymbol{v}}_k = H_k(t)\tilde{\boldsymbol{u}}_k$ and $\tilde{b}_k = H_k(t)\tilde{q}_k$, so we have:

$$\begin{aligned} \frac{d\tilde{\boldsymbol{v}}_k}{dt} + H_k(t)\mathcal{F}_k[\boldsymbol{N}(\boldsymbol{\omega},\boldsymbol{u})] &= -G_k\tilde{b}_k + H_k(t)P(t)^T\tilde{\boldsymbol{f}}_k, \\ D_k\tilde{\boldsymbol{v}}_k &= 0, \\ P(t)H_k^{-1}(t)\tilde{\boldsymbol{v}}_k &= \widetilde{\boldsymbol{u}}_{\Gamma,k}. \end{aligned} \tag{3.37}$$

We adopt a half-explicit Runge-Kutta (IF-HERK) method for these DAE (Brasey and Hairer, 1993; Liska and Colonius, 2017). For the present method, this can be stated as

1. Initialize: set $\tilde{\boldsymbol{u}}^0_{k,n} = \tilde{\boldsymbol{u}}_{k,n}$ and $t^0_n = t_n$
2. Multistage: for $i = 1, 2, ..., s$, solve the linear system

$$\begin{bmatrix} (E^i_k)^{-1} & (Q^{(i-1)}_{k,n})^* \\ Q^i_{k,n} & 0 \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{u}}^i_{k,n} \\ \hat{\lambda}^i_{k,n} \end{bmatrix} = \begin{bmatrix} \boldsymbol{r}^i_{k,n} \\ \zeta^i_{k,n} \end{bmatrix}. \tag{3.38}$$

3. Finalize: set $\tilde{\boldsymbol{u}}_{k,n+1} = \tilde{\boldsymbol{u}}^s_{k,n}$, $\lambda_{k,n+1} = (\tilde{a}_{s,s}\Delta t)^{-1}\hat{\lambda}^s_{k,n}$, and $t_{n+1} = t^s_n$

where

$$Q^i_{k,n} = \begin{bmatrix} G^*_k \\ P^i_n \end{bmatrix}, \qquad \hat{\lambda}^i_{k,n} = \begin{bmatrix} \tilde{q}^i_{k,n} \\ \tilde{f}^i_{k,n} \end{bmatrix}, \qquad \zeta^i_{k,n} = \begin{bmatrix} 0 \\ (\widetilde{\boldsymbol{u}}_{\Gamma,k})^i_n \end{bmatrix}. \tag{3.39}$$

The terms appearing in the linear system are

$$\boldsymbol{u}^{i-1}_n = \mathcal{F}^{-1}(\{\tilde{\boldsymbol{u}}^{i-1}_{k,n}\}_k), \qquad \boldsymbol{\omega}^{i-1}_n = \mathcal{F}^{-1}(\{C_k\tilde{\boldsymbol{u}}^{i-1}_{k,n}\}_k) \tag{3.40}$$

where $\mathcal{F}^{-1}$ is evaluated using (de-aliased) inverse Fast Fourier Transform (iFFT). Furthermore

$$E^i_k = H_k\left(\frac{(\tilde{c}_i - \tilde{c}_{i-1})\Delta t}{(\Delta x)^2 Re}\right), \qquad g^i_{k,n} = -\tilde{a}_{i,i}\Delta t\mathcal{F}_k[\boldsymbol{N}(\boldsymbol{\omega}^{i-1}_n, \boldsymbol{u}^{i-1}_n)] \tag{3.41}$$

where for all k, $\mathcal{F}_k[\boldsymbol{N}(\boldsymbol{\omega}^{i-1}_n, \boldsymbol{u}^{i-1}_n)]$ is evaluated using FFT. Then the following are recursively computed for $i > 1$ and $j > i$ using:

$$h^i_{k,n} = E^{i-1}_k h^{i-1}_{k,n}, \qquad h^1_{k,n} = \tilde{\boldsymbol{u}}^0_{k,n} \tag{3.42}$$

$$w^{i,j}_{k,n} = E^{i-1}_k w^{i-1,j}_{k,n}, \qquad w^{i,i}_{k,n} = (\tilde{a}_{i,i}\Delta t)^{-1}\left(g^i_{k,n} - Q^{i-1}_{k,n}\hat{\lambda}^i_{k,n}\right). \tag{3.43}$$

0	0	0	0
1/3	1/3	0	0
1	-1	2	0
	0	3/4	1/4

Table 3.1: Runge-Kutta scheme Butcher Tableau used in our implementation

And we compute:

$$r^i_{k,n} = h^i_{k,n} + g^i_{k,n} + \Delta t \sum_{j=1}^{i-1} \tilde{a}_{i,j} w^{i,j}_{k,n}, \qquad t^i_n = t_n + \tilde{c}_i \Delta t. \tag{3.44}$$

With this time discretization, at the n^{th} time step in i^{th} stage for the k^{th} Fourier coefficient, we have the following linear system:

$$M^i_{k,n} \begin{bmatrix} \tilde{\boldsymbol{u}}^i_{k,n} \\ \tilde{q}^i_{k,n} \\ \tilde{f}^i_{k,n} \end{bmatrix} = \begin{bmatrix} (E^i_k)^{-1} & G_k & (P^{(i-1)}_n)^T \\ G^*_k & 0 & 0 \\ P^i_n & 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{u}}^i_{k,n} \\ \tilde{q}^i_{k,n} \\ \tilde{f}^i_{k,n} \end{bmatrix} = \begin{bmatrix} \boldsymbol{r}^i_{k,n} \\ 0 \\ (\widetilde{\boldsymbol{u}}_{\Gamma,k})^i_n \end{bmatrix} \tag{3.45}$$

where $S^i_{k,n} = P^i_n E^i_k (I - G_k L^{-1}_k D_k)(P^{i-1}_n)^T$, which we will show, in Appendix C, that it is Hermitian when $P^i_n = P^{i-1}_n$. E^i_k refers to the integrating factor associated with L_k. L^{-1}_k refers to the lattice Green's Function (LGF) of L_k. We will explain how to apply them in subsequent sections.

We can solve the linear system arising from the IF-HERK method using a block LU decomposition. As a result, the steps to solve this system are:

$$\begin{aligned} \tilde{q}^*_k &= -L^{-1}_k G^*_k \boldsymbol{r}^i_{k,n}, \\ S^i_{k,n} \tilde{f}^i_{k,n} &= P^i_k E^i_k [\boldsymbol{r}^i_{k,n} - G_k \tilde{q}^*_k] - (\widetilde{\boldsymbol{u}}_{\Gamma,k})^i_n, \\ \tilde{q}^i_{k,n} &= \tilde{q}^*_k + L^{-1}_k G^*_k (P^i_n)^T \tilde{f}^i_{k,n}, \\ \tilde{\boldsymbol{u}}^i_{k,n} &= E^i_k [\boldsymbol{r}^i_{k,n} - G_k \tilde{q}^i_{k,n} - (P^{i-1}_n)^T \tilde{f}^i_{k,n}]. \end{aligned} \tag{3.46}$$

Note that this block LU decomposition method does not have splitting error due to the use of the integrating factor (Liska and Colonius, 2017). The advantage is that the divergence-free constraint and the boundary conditions are satisfied exactly (Taira and Colonius, 2007; Liska and Colonius, 2017). In our current implementation of this algorithm, we use a 3rd order scheme with the Butcher Tableau shown in Table 3.1.

Apart from the second equation of Eq. 3.46, the remaining equations can be solved directly through the application of LGF and the integrating factor. The second

equation in Eq. 3.46 corresponds to the projection step to compute the IB forcing in order to satisfy the boundary condition (Taira and Colonius, 2007; Liska and Colonius, 2017). In similar algorithms designed for general 3D flows (Liska and Colonius, 2017; Yu, Dorschner, and Colonius, 2022), the conjugate gradient method is employed to solve for the IB forcing. In that case, it is estimated that the linear system associated with IB forcing is a $3N_L$ by $3N_L$ dense system where N_L is the total number of IB points, and the constant 3 arises from the three velocity components. In the case of general 3D flows, the number of operations needed to solve such a linear system is $O(N_L^3)$ (Liska and Colonius, 2017). In addition, due to the sequential nature of matrix factorization and back-substitution (Liska and Colonius, 2017), directly solving for IB forcing using numerical factorization becomes less desirable than the conjugate gradient method.

For flows with one homogeneous direction, the immersed body has a uniform 2D cross-section across the span-wise direction giving $N_L = N_z N_{2D}$, where N_z is the number of Fourier coefficients in the truncated Fourier series, and N_{2D} is the number of IB points used to represent the 2D cross-section. To solve for the Fourier coefficients of the IB forcing using the evolution equations of the Fourier coefficients (Eq. 3.5), we solve for N_z independent $3N_{2D}$ by $3N_{2D}$ dense linear systems instead of one $3N_L$ by $3N_L$ dense linear system. Thus, the operation count for using a direct solver in our scenarios decreases to $O(N_z(N_{2D})^3) = O(N_L^3/N_z^2)$. More importantly, due to the independence among those N_z linear systems, the application of numerical factorization and back-substitution can be efficiently parallelized. Thus, unlike the method to solve for IB forcing in the general 3D flow solver algorithm, we solve the IB forcing using direct LU factorization instead of the conjugate gradient method. In our implementation, the dense linear system is solved using ScaLAPACK(Blackford et al., 1997) wrapper from PETSc(Balay et al., 2022). According to our numerical experiments, solving for IB forcing directly takes less than 3% of the total computational time when using the LU factorization.

3.6 Multilevel mesh

To resolve thin boundary layers, particularly with the IB method, adaptive mesh refinement is needed (Mittal and Balachandar, 1997; Lehmkuhl et al., 2013; Yu, 2021). For Cartesian grids, a wide range of adaptive mesh refinement (AMR) algorithms have been proposed to locally refine the computational mesh (Berger and Oliger, 1984; Berger and Colella, 1989; Popinet, 2003; Burstedde, Wilcox, and Ghattas, 2011; Offermans et al., 2020; Gillis and Van Rees, 2022). These local refinement

methods focus on refining patches of the computational mesh according to specific refinement criteria. To enhance the scalability of the mesh refinement process, octree-based methods have been proposed (Popinet, 2003; Burstedde, Wilcox, and Ghattas, 2011). Further, these octree-based methods can be combined with wavelet methods to enable more efficient multiresolution methods (Schneider and Vasilyev, 2010; Gillis and Van Rees, 2022).

For our solver, the main requirement for AMR methods is compatibility with the application of the LGF. Thus, we adapted an existing adaptive mesh refinement approach that has proven efficient and accurate when combined with LGF (Yu, 2021). This AMR approach uses an octree structure (Burstedde, Wilcox, and Ghattas, 2011; Gillis and Van Rees, 2022) to recursively refine the fully 3D computational mesh. We implemented a quadtree counterpart of this approach to apply to each 2D slice in the $x - y$ plane for each Fourier coefficient. Adaptation of this AMR algorithm to our hybrid method is detailed in the remainder of this section.

3.6.1 Multilevel mesh on a Cartesian grid

We first review the salient features of the algorithm of Yu (2021). We start by constructing a composite grid consisting of multiple computational grids with increasing resolutions, $\{\mathbb{R}_0^Q, \mathbb{R}_1^Q, ..., \mathbb{R}_N^Q\}$, wherein $\mathbb{R}_k^Q$ is generated by evenly dividing each computational cell in $\mathbb{R}_{k-1}^Q$ into 2^d smaller cells, d being the physical dimension of the problem. The composite grid is the Cartesian product of this series of grids:

$$\overline{\mathbb{R}^Q} = \bigotimes_{k=0}^{N} \mathbb{R}_k^Q. \tag{3.47}$$

Then the AMR grid is defined by partitioning the entire computational domain Ω into $N+1$ smaller pairwise-disjoint domains $\{\Omega_0, \Omega_1, ...\Omega_N\}$. Define the restriction functional as:

$$\Gamma_k(f)(\boldsymbol{n}) = \begin{cases} f(\boldsymbol{n}) & \text{if } \boldsymbol{n} \in \Omega_k, \\ 0 & \text{if } \boldsymbol{n} \notin \Omega_k \end{cases}, \tag{3.48}$$

and the values on the AMR computational grid are defined through the Cartesian product of these restriction functionals

$$\overline{\Gamma} = \bigotimes_{k=0}^{N} \Gamma_k. \tag{3.49}$$

The operator $\overline{\Gamma}$ restricts the region we need to compute the numerical solutions to

$$\overline{\Omega} = \bigotimes_{k=0}^{N} \mathbb{R}_k^Q \cap \Omega_k. \tag{3.50}$$

Thus, the solution is defined in the subspace $\overline{\Omega}$. To communicate the information across different levels of mesh, we also define interpolation and coarsening operators from level l to level k as $P_{l\to k}$ (interpolation when $l < k$ and coarsening when $l > k$). To estimate the information f on level k given the information across the AMR grid $\hat{f}$, we compute:

$$f_k = \sum_{l=0}^{N} P_{l\to k}\hat{f}_l. \tag{3.51}$$

Then to apply LGF on the AMR mesh from a source term $\hat{f}$, we use the following (Yu, 2021):

$$\phi(\boldsymbol{n}) = \otimes_{j=0}^{N}\left[\left(\sum_{i=0}^{j-1}\overline{P}^{Q}_{i\to j}L^{-1}_{0,i}\hat{f}_i\right) + L^{-1}_{0,j}\left(\sum_{i=j}^{N}P^{Q}_{i\to j}\hat{f}_i\right)\right], \tag{3.52}$$

where $\overline{P}^{Q}_{i\to j}$ is a commutative projection operator (see Eq. 30-32 in Yu, Dorschner, and Colonius (2022)). We use $L^{-1}_{0,l}$ to denote the action of applying LGF of Laplacian on level l in the refinement mesh. In this equation, at level j, the first term recursively computes the solution at level j induced from the solution at coarser levels. The second term computes the solution induced by the source on level j and the source interpolated from finer levels. The first term is accumulated when computing the solution from the base level to the finest level. We can directly use this method to apply L^{-1}_{k} for each Fourier coefficient. To achieve additional speedup when applying LGFs, a fast and parallel multilevel elliptic equation solution method is employed (Dorschner et al., 2020; Ying, Biros, and Zorin, 2004; Liska and Colonius, 2014).

3.6.2 Multilevel in Fourier space

To exploit the multilevel mesh in the $x-y$ plane, we must also locally truncate the Fourier series such that the resolution in z is comparable to the local resolution in the $x-y$ plane. Considering the spectral convergence in z, compared to the low-order convergence of the finite volume discretization in the $x-y$ plane, it is expected that the mesh in the z direction needs to be refined, at most, as fast as the rate we refine the mesh in the $x-y$ plane.

However, for spanwise homogeneous flows, the flow field is not homogeneous everywhere in the presence of the immersed body, especially when the boundary layer is laminar. In the far wake, the length scales tend toward homogeneity, but, near the immersed body, variation in the $x-y$ plane can be much more rapid than that of the z-direction (Smith, 1986; Williamson, 1996). One example of such

inhomogeneity is the flow past a circular cylinder. In the shear-layer transition regime ($Re \sim 1,000 - 200,000$), two shear layers are forming from the side of the cylinder. Those shear layers and the associating Kelvin-Helmholtz instability are essentially 2D. Thus, to resolve the shear layers, we only need to refine the computational grid in the $x - y$ plane. However, downstream of the shear layers, the flow transitions to three-dimensional turbulence, and comparable resolution is required in all three directions. To optimally treat these situations, we modify the spanwise refinement. Far from the body, if there are l_{ref} refinement levels, on level l, we retain $N_l = N_0 \times 2^l$ Fourier coefficients, where N_0 is the number for the base (coarsest) mesh. Near the body, we cap the number of Fourier coefficients, even as we refine the $x - y$ grid by an additional l_{add} levels.

We now elaborate on how we apply the LGF on the multilevel mesh in both Fourier space and the $x - y$ plane. For the k^{th} Fourier coefficient, we find an l such that $N_0 2^{l-1} \leq k < N_0 2^l$. We know that computational cells on level $l - 1$ need only retain the first $N_0 2^{l-1}$ Fourier coefficients. That is, we may assume the k^{th} Fourier coefficient is zero for all computational cells on level $l-1$. Similarly, the k^{th} Fourier coefficient is zero for all computational cells on levels 0 through $l - 1$. Thus, the k^{th} Fourier coefficients and the associating source terms $\hat{f}^k$ are zero on those levels. We do not need to apply L_k^{-1} on levels 0 through $l - 1$, nor need we consider source terms from those levels since they are to be truncated. With this strategy, we can simplify the procedure that applies L_k^{-1} in Eq. 3.52 so that the index of the first term begins at $i = l$ rather than $i = 0$, resulting in significant computational savings.

Conversely, for the k^{th} Fourier coefficient, the corresponding coarsest level that L_k^{-1} need be applied is given by:

$$l = \max\left(\left\lceil \log_2\left(\frac{k+1}{N_0}\right)\right\rceil, 0\right) = \max\left(l_{ref} - \left\lfloor \log_2\left(\frac{N_0 \times 2^{l_{ref}}}{k+1}\right)\right\rfloor, 0\right). \tag{3.53}$$

Similar to applying L_k^{-1}, applying discrete operators (divergence, gradient, curl), interpolation operators, and the integrating factor E_k^i follows a similar strategy. For an operator operating on the k^{th} Fourier coefficient, the coarsest level, l, the operator needs to be applied on is also determined by Eq. 3.53. Thus. that operator only needs to be applied to grid points on levels greater than or equal to l.

3.7 Adaptation

So far, we have introduced the steps to time integrate the discretized Navier-Stokes equations using the IF-HERK method and LGF. In addition, the computational grid

spatially adapts to the vortical region, which we term base-level adaptivity, and adaptively refines the mesh in a block-wise fashion. For adaptation, we adopt the strategies developed for the fully inhomogeneous case (Liska and Colonius, 2016; Yu, 2021); we provide a brief summary of the algorithms here.

3.7.1 Base-level (domain) adaptation

The assumption that allows us to constrain our computational domain to a finite one is that the vortical region is in general compact (exponentially decaying). The strategy is to add or delete cells (block-wise) when the vorticity near the boundary exceeds or falls below a threshold value. However, the surface defining the threshold value must adapt in time as the vorticity evolves in the boundary layer and wake regions. Additionally, it is sometimes pertinent to alter the threshold in the far wake as, for long times, the vorticity decays slowly.

To these ends, we denote the active cell region in the base level

$$\Omega_0^{supp} = \left\{ \boldsymbol{x} \in \mathbb{R}^2 : \frac{\sum_i |D_i \boldsymbol{r}_i(\boldsymbol{x})|^2}{\max_{\boldsymbol{x}} \sum_i |D_i \boldsymbol{r}_i(\boldsymbol{x})|^2} \geq \epsilon_0 \right\}, \tag{3.54}$$

where $\boldsymbol{r}_i$ is the nonlinear term (Lamb vector) in the time-discretized equations (Eq. 3.44) and D_k the discrete divergence operator for the k^{th} Fourier coefficient. ϵ_0 is some prescribed cutoff threshold. The $\sum_i$ is the sum across all Fourier coefficients. Thus, the term $\sum_i |D_i \boldsymbol{r}_i(\boldsymbol{x})|^2$ is approximately the magnitude of the source term for the pressure-Poisson equation at location $\boldsymbol{x}$ across all Fourier coefficients. To ensure that Ω_0^{supp} is captured by our computational domain, we additionally put a region (Ω_{buff}) of buffering computational cells around Ω_0^{supp}. These additional computation cells are those at a fixed distance N_b from $\partial\Omega_0^{supp}$. We periodically update Ω_0^{supp} by incorporating grid cells from Ω_{buff} to satisfy Eq. 3.54. The detail of how to choose N_b is discussed in (Liska and Colonius, 2016).

Each time we incorporate computational cells from Ω_{buff} into Ω_0^{supp}, a new $\partial\Omega_0^{supp}$ is effectively defined, and we need to compute the velocity field in the newly incorporated region Ω_{buff}. To do so, we solve the vorticity-streamfunction equation to fill in the velocity in Ω_{buff}. Let $\tilde{\boldsymbol{u}}_k^u$ be the k^{th} Fourier coefficient of velocity before solving the vorticity-streamfunction equation and $\tilde{\boldsymbol{u}}_k^a$ be the values after, we solve the velocity within Ω_{buff} using:

$$\begin{aligned} \tilde{\boldsymbol{\omega}}_k &= C_k \tilde{\boldsymbol{u}}_k^u, \\ \tilde{\boldsymbol{u}}_k^a &= -C_k^* L_k^{-1} \mathbb{1}_{\Omega_0^{supp}} \tilde{\boldsymbol{\omega}}_k. \end{aligned} \tag{3.55}$$

where $\mathbb{1}_{\Omega_0^{supp}}$ is the indicator function of the set Ω_0^{supp}. Namely, when fill in the velocity in the buffering region, we only use the vorticity within Ω_0^{supp}. We term the above procedure as velocity padding.

3.7.2 Velocity refresh

When applying the integrating factor E_k^i through convolution, the support of the associating kernel is unbounded. However, we can still accurately evaluate this convolution with Ω_0^{supp} using only the values within Ω_0^{supp} and Ω_{buff}. To that end, we use two facts: the integrating factor decays faster than any exponential (Liska and Colonius, 2016), and velocity in Ω_0^{supp} can be obtained by solving a vorticity-streamfunction equation. Thus, we employ a two-step approximation to evaluate the action of E_k^i within Ω_0^{supp}.

First, we truncate the integrating factor kernel to have a compact support by thresholding the value of the kernel. Due to the fast decay, we can accurately approximate the action of E_k^i by only applying the integrating factor kernel within this compact region.

Second, near $\partial\Omega_0^{supp}$, the source region of the approximated E_k^i operator extends outside of Ω_0^{supp}. We assume that this additional source region is contained in some Ω_{IF}. Since this region is outside of Ω_0^{supp}, the vorticity within Ω_{IF} is negligible. Thus, we can compute the velocity in this region using vorticity-streamfunction equation as in Eq. 3.55. This step is called velocity refresh.

Combining these two steps, we can accurately approximate the action of E_k^i within Ω_0^{supp} by carefully defining Ω_{IF} and the source region of E_k^i. The specific procedure of properly truncating the integrating factor kernel and defining Ω_{IF} can be found in (Liska and Colonius, 2016). We note that the vorticity-streamfunction equation only needs to be solved periodically to ensure accurate simulation. In fact, the frequency we need to solve the vorticity-streamfunction equation depends on Ω_{IF}.

It is shown that we can overlap Ω_{IF} with the buffering region we defined for velocity padding (Liska and Colonius, 2016). That is $\Omega_{\text{IF}} = \Omega_{\text{buff}}$. As a result, we can compute a corresponding maximum time step n_r such that we only need to conduct velocity refresh every n_r time step.

3.7.3 Adaptive refinement

As the flow develops, the high vorticity and high velocity gradient regions change. As a result, the computational grid needs to adapt to the evolution of the flow. In our

algorithm, this step is done by tracking the high vorticity region and locally refining the mesh accordingly. Recall that the AMR grid decomposes the entire computational domain into a sequence of pairwise disjoint domains $\{\Omega_l\}_l$. To adaptively refine the computational mesh, we partition each Ω_l into smaller computational blocks. As shown in Section 3.6, the adaptive refinement in the spanwise direction is determined by the adaptive refinement in the $x-y$ plane. Thus, it is sufficient to consider the refinement criterion for the 2D plane formed by the inhomogeneous directions. In this part, the vorticity is used as the criterion for adaptation. We specify a refinement factor $\alpha \in (0,1)$ and a deletion factor $\beta \in (0,1)$. When a computational block is on level l, we refine the block if any point $\boldsymbol{x}$ in that block satisfies:

$$S(\boldsymbol{x}) = \sum_k ||\tilde{\boldsymbol{\omega}}_k(\boldsymbol{x})||_2^2 > \alpha^{l_{max}-l} S_{max}, \tag{3.56}$$

and coarsen the block if every point $\boldsymbol{x}$ in that block satisfies:

$$S(\boldsymbol{x}) = \sum_k ||\tilde{\boldsymbol{\omega}}_k(\boldsymbol{x})||_2^2 < \beta\alpha^{l_{max}-l} S_{max}, \tag{3.57}$$

where $||\tilde{\boldsymbol{\omega}}_k(\boldsymbol{x})||_2$ is the 2-norm of the k^{th} Fourier coefficient of vorticity vector at point $\boldsymbol{x}$. Using Parseval's identity, we have:

$$\begin{aligned} &\frac{1}{c}\int_{-c/2}^{c/2} \sum_{i\in\{1,2,3\}} \omega_i^2(x,y,z)dz \\ &= \sum_{k=-\infty}^{\infty} ||\tilde{\boldsymbol{\omega}}_k(\boldsymbol{x})||^2 \\ &\approx \sum_{k=-N/2}^{N/2} ||\tilde{\boldsymbol{\omega}}_k(\boldsymbol{x})||^2. \end{aligned} \tag{3.58}$$

Thus, $S(\boldsymbol{x})$ approximates the squared L_2 norm of the vorticity at each $\boldsymbol{x}$ location. In addition, we solve the vorticity-streamfunction equations to pad velocity when new blocks are refined using Eq. 3.55. This is to fill in the Fourier coefficients of velocities in the newly refined blocks, as those Fourier coefficients were previously set to zero due to the truncation of the Fourier series from the multilevel nature in Fourier space.

3.8 Algorithm summary

Algorithm 2 summarizes the required steps to march the solution forward by N time steps. Let n_a be the desired frequency (number of steps) to adapt the domain

and/or resolution, and let n_r be the desired frequency (number of steps) to conduct velocity refresh. When simulating fluid flows with this algorithm, n_a should be chosen according to the resolution requirements, and n_r should be chosen according to the procedure detailed in (Liska and Colonius, 2017).

Algorithm 2 Time Marching using IF-HERK, IB, and LGF

procedure TIME MARCHING($\tilde{\boldsymbol{u}}_{k,0}, t_f$)
  $n = 0$
  while $n < N$ **do**
    if $n\%n_a = 0$ **then**
      Perform domain adaptation
      Perform velocity padding using Eq. 3.55
    else if $n\%n_r = 0$ **then**
      Perform velocity refresh using Eq. 3.55
    end if
    set $\tilde{\boldsymbol{u}}^0_{k,n} = \tilde{\boldsymbol{u}}_{k,n}$ and $t^0_n = t_n$
    for each stage $i \in \{1, 2, 3\}$ **do**
      Compute $\boldsymbol{u}^{i-1}_n$ and $\boldsymbol{\omega}^{i-1}_n$ using inverse FFT
      Compute $g^i_{k,n}$, $r^i_{k,n}$ according to Eq. 3.41 and Eq. 3.44
      Solve the system of equations shown by Eq. 3.45 using the block LU decomposition as detailed in Eq. 3.46
    end for
    Setting $\tilde{\boldsymbol{u}}_{k,n+1} = \tilde{\boldsymbol{u}}^3_{k,n}$, $\lambda_{k,n+1} = (\tilde{a}_{3,3}\Delta t)^{-1}\hat{\lambda}^3_{k,n}$, and $t_{n+1} = t^3_n$
    $n = n + 1$
  end while
end procedure

3.9 Parallelization and performance

We adopted a server-client model for parallelization based on decomposing the domain into pencils that correspond to blocks in the $x - y$ plane. That is, all Fourier coefficients (regardless of the number) are stored on the same processor, which avoids data transfer to accomplish the FFT. Each block is assigned a computational load according to their roles during the time-stepping routine, and a load balancing algorithm distributes those blocks into different processors (Yu, Dorschner, and Colonius, 2022).

However, since direct solvers are used to solve for IB forcings, we need to devise a corresponding parallelization strategy. Suppose we are solving systems with N Fourier coefficients and M parallel client processes. Two separate parallelization strategies are devised for the case when $N > M$ and $N \leq M$, respectively.

- When $N > M$, each linear system is only solved using one process, and each process is tasked with solving one or more linear systems. Specifically, $N\%M$ processes are allocated to handle $\lceil N/M \rceil$ dense linear systems, and the rest processes are allocated to handle $\lfloor N/M \rfloor$ dense linear systems.

- When $N \leq M$, multiple processes are allocated to solve one linear system, and each process is assigned only one linear system. Specifically, $M\%N$ linear systems are each solved by a group of $\lceil M/N \rceil$ processes, while the rest linear systems are each solved by a group of $\lfloor M/N \rfloor$ processes.

We report a modest scaling test for a varying number of leaf octants (16000, 31360, and 64000), spreading across 3 refinement levels ($l_{ref} = 1$, $l_{add} = 1$, $l_{max} = 2$). Each octant is a 6 by 6 grid cell. The blocks on the finest level have 16 complex Fourier modes. We define parallel efficiency as the speedup divided by the number of computational nodes, benchmarked against the runtime obtained using 4 computational nodes. We computed the parallel efficiency by evaluating one RK3 step using various numbers of computational nodes. The resulting parallel efficiency is shown in Figure 3.5. The strong scaling is consistent with the corresponding fully inhomogeneous LGF method (Dorschner et al., 2020; Yu, Dorschner, and Colonius, 2022). The simulation size is restricted by the memory required by the algorithm. Further refinements to the parallelization will be implemented in the future.

As discussed previously (Liska and Colonius, 2014; Yu, Dorschner, and Colonius, 2022), the LGF approach to the Poisson inversion is extremely efficient given the complex and adaptive domain. On a per-point basis, only purely FFT based algorithms are likely to be more efficient, but the required rectangular domain would waste many points for the flows we compute.

For the specific case of incompressible flow solver with one Fourier diagonalizable direction, we can make a direct comparison with the Jacobi and block-Jacobi Preconditioned Conjugate Gradient methods (JPCG and bJPCG) employed in an unstructured-mesh solver (Borrell et al., 2011). The authors also reported solution times for a direct Schur-complement based decomposition method (DSD), but we refrain from comparisons as such a method requires precomputing the Cholesky factorization and would be prohibitive in an adaptive algorithm. Our method is compared to JPCG and bJPCG in Table 3.2, where the computational rate for the Poisson solution is reported*. For our code, the test is performed in the context of

*The simulations by the bJPCG and JPCG are done using PowerPC 970MP 2.3GHz CPUs(Borrell

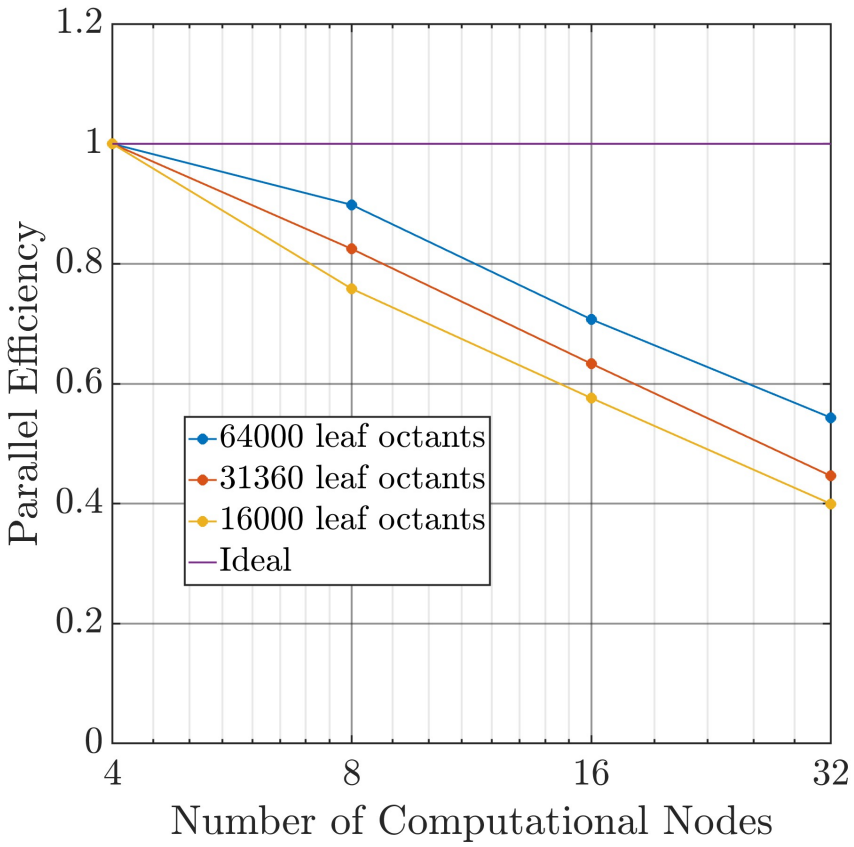


Figure 3.5: Parallel efficiency for a varying number of computational nodes for computational meshes of various sizes. The parallel efficiency is measured by solving one RK3 step using different numbers of computational nodes. Each node contains 48 computational cores. The simulations are conducted on TACC Stampede3 supercomputer (*Stampede3 User Guide* n.d.) with Intel Xeon Platinum 8380 2.3GHz CPUs.

the $Re = 300$ and $Re = 12,000$ cylinder flows[†] to be discussed in Section 3.11, which were computed using 256 CPU cores (2 computational nodes on Bridges-2 supercomputer) and 2000 CPU cores (25 computational nodes on Stampede2 supercomputer), respectively. Our algorithm is about an order of magnitude faster for $Re = 300$ case, and 4 times faster for the $Re = 12,000$ case. The latter case was impacted by deteriorating parallel performance on the associated large grid of about 400 million cells.

In addition, for the more general case of simulating incompressible external flows

et al., 2011).

[†]The $Re = 300$ cylinder flow is simulated on the Bridges-2 supercomputer (Brown et al., 2021) with AMD EPYC 7742 2.25GHz CPUs. The $Re = 12,000$ cylinder flow is simulated on the Stampede2 supercomputer (Stanzione et al., 2017) using Intel Xeon Platinum 8380 2.3GHz CPUs.

Case	Computational Rate (cpu $\times\mu$s/pts)
JPCG $\xi = 0$	83.1
bJPCG $\xi = 0$	79.8
Nek5000	55.6
Present ($Re = 300$)	7.1
Present ($Re = 12,000$)	19.4

Table 3.2: Efficiency comparison between methods for the 2D screened Poisson problems (JPCG and bJPCG (Borrell et al., 2011)) and the 3D Poisson problem (Nek5000 (Fischer, Lottes, and Tufo, 2007; Hosseini et al., 2016)) in incompressible flow. The JPCG and bJPCG values are based on those reported (Borrell et al., 2011) for the $\xi = 0$ parameter value in their screened Poisson problem, which represents the worst case. The Nek5000 value is based on the time to run one GMRES iteration (Hosseini et al., 2016) in their Poisson solver and the expected number of iterations for the GMRES algorithm to converge, estimated from the number of iterations for the JPCG method to converge (Borrell et al., 2011).

with one periodic direction, we can compare with the spectral element incompressible flow solver Nek5000 (Fischer, Lottes, and Tufo, 2007) simulating the flow past a wing section (Hosseini et al., 2016). In (Hosseini et al., 2016), the runtime of one GMRES (generalized minimal residual method) iteration for their Poisson solver is reported[‡]. However, solving the Poisson equation requires many GMRES iterations. As indicated by (Borrell et al., 2011), the number of iterations for the JPCG method to solve the Poisson equation is 217. In addition, the conjugate gradient method (used by the JPCG method) has better convergence properties than GMRES (Trefethen and Bau, 2022). Thus, the number of iterations for the JPCG method to converge can serve as a lower-bound estimate of the number of iterations required for the GMRES algorithm to converge. As such, we compare the computational rate for Nek5000 to compute 217 GMRES iterations to the computational rate of our method, also in Table 3.2. Our method is roughly 8 times faster than Nek5000 for the $Re = 300$ case and roughly three times faster for the $Re = 12,000$ case.

The computational efficiency reported here could potentially be further improved with enhancements to the parallelization strategy and other optimizations, but as it stands we believe our algorithm is competitive with (and potentially faster than) other state-of-the-art incompressible flow solvers.

[‡]The GMRES iteration is computed on the Cray-XC40 computer Beskow at PDC (KTH) using Intel Xeon E5-2698v3 Haswell 2.3 GHz CPUs (Hosseini et al., 2016).

3.10 Verification

As discussed above, the flow solver described in this paper is an extension of a fully 3D incompressible flow solver (Liska and Colonius, 2017; Yu, 2021). In these papers, the authors verified the method by solving the flow past an impulsively starting sphere. To verify the modified solver for one homogeneous and two unbounded directions, we provide two examples: flow past an impulsively starting cylinder and the evolution of an Oseen vortex (Panton, 2024).

We compute the flow past a cylinder with diameter D and $Re = 100$ using 16 complex Fourier coefficients (31 terms when evaluating Fourier series) and the following initial vorticity distributions:

$$\omega_{n,z} = \exp(-n^2 - |r|^2/D^2). \tag{3.59}$$

We obtain an initial velocity by solving the discrete 2D Poisson equation and the screened Poisson equations from the vorticity-streamfunction equation. We run the simulation for $1.024\, tU_\infty/D$ and used a uniform grid simulation with $\Delta x_0/D = 0.005$ as the reference solution. We consider cases where the base spatial resolutions Δx_{base} satisfies $\Delta x_{base}/\Delta x_0 \in \{4, 8, 16, 32, 64\}$ and the finest level resolution Δx_{fine} satisfies $\Delta x_{fine}/\Delta x_0 \geq 4$. In each mesh topology, $l_{max} = l_{ref} = N_l$, $l_{add} = 0$. On l^{th} level, we refine a squared region centered at the origin with an edge length of $3.84D/2^l$. Mathematically, on l^{th} level, we refine the region defined by the following set:

$$\Omega_l^r = \left\{ (x, y) : \left|\frac{x}{D}\right| < \frac{1.92}{2^l}, \left|\frac{y}{D}\right| < \frac{1.92}{2^l} \right\}. \tag{3.60}$$

The error is shown in Figure 3.6. We normalize L_∞ error by the L_∞ norm of the reference solution and L_2 error by the L_∞ norm of the reference solution and the size of the domain. Both L_∞ and L_2 error show a first-order convergence, as expected for our 2nd-order finite-volume scheme with first-order immersed boundary method treatment (Tornberg and Engquist, 2004; Mori, 2008; Taira and Colonius, 2007; Colonius and Taira, 2008).

Second, to verify that our flow solver converges to the solution of the Navier-Stokes equations, we compare the evolution of an Oseen vortex to the exact solution (Panton, 2024)

$$u_\theta(t, r) = \frac{\eta_c^2 + 2}{\eta_c^2 r}\left[1 - \exp\left(\frac{r^2 Re}{4t}\right)\right]. \tag{3.61}$$

To verify our solver, we initialized the velocity profile at t_0 and marched for t_c time units. The numerical solution is then compared against the analytical solution to

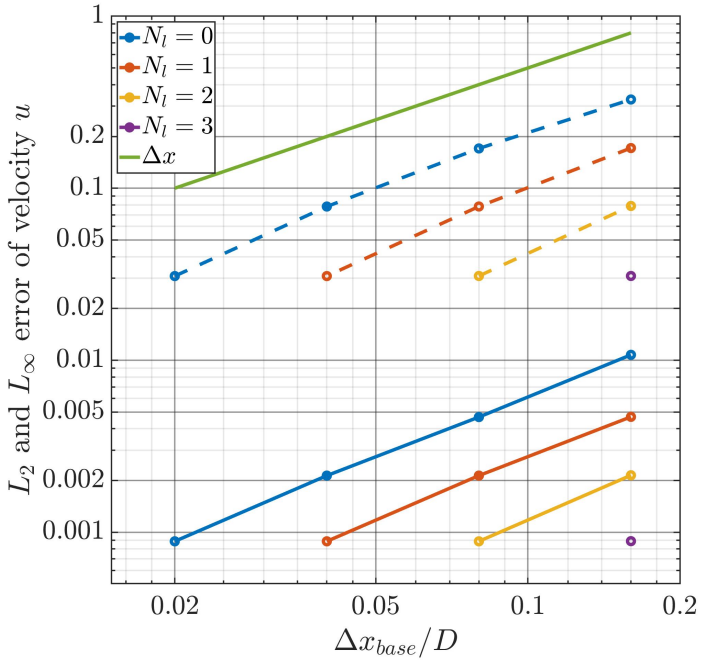


Figure 3.6: Error in the streamwise velocity compared to the base solution using different numbers of refinement levels ($0 \leq N_l \leq 3$). The solid lines represent the L_2 error, and the dashed lines represent the L_∞ error.

obtain the error. Our specific choice of parameters is:

$$t_0 = \frac{Re}{\eta_c^2}, \quad t_c = 0.5, \quad Re = 100, \quad \eta_c = 2.24181. \tag{3.62}$$

The above parameters are chosen such that the maximum velocity is 1, which is obtained when $t = t_0$ and $r = \sqrt{x^2 + y^2} = 1$. We hold the mesh topology fixed with three levels of refinement. Following the notation of the previous example, at level l, the refinement region is defined as

$$\Omega_l^r = \left\{ (x, y) : |x| < \frac{11.2}{2^l}, |y| < \frac{11.2}{2^l} \right\}. \tag{3.63}$$

The set of spatial resolution we consider is $\Delta x_{base} \in \{0.08, 0.04, 0.02, 0.01\}$. The time-step size is fixed at $\Delta t = 0.00625$. The L_2 and L_∞ errors of the x-component velocity are shown in Figure 3.7 and demonstrate the expected second-order convergence.

3.11 Transitional and turbulent flow over a circular cylinder

To demonstrate the new algorithm on transitional and turbulent flows we consider flow over a circular cylinder at $Re = 300$ and 12,000.

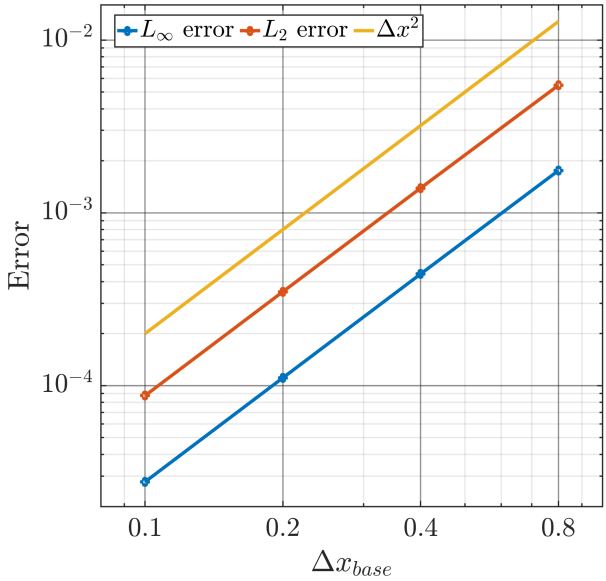


Figure 3.7: Error convergence of the numerical solution from our solver. The error is computed by simulating the evolution of an Oseen vortex for $t_c = 0.5$ and compared against the analytical solution.

3.11.1 $Re = 300$

Flow over a circular cylinder at $Re = 300$ has been extensively studied numerically and experimentally (Mittal and Balachandar, 1997; Kravchenko, Moin, and Shariff, 1999; Norberg, 2003). At still lower Re, the flow undergoes a series of bifurcations and by $Re = 300$, the most prominent instability is termed mode-B and consists of vortex shedding modulated by three-dimensional streamwise vortex pairs. These form horseshoe-shaped vortices downstream that are stretched in the streamwise direction (Williamson, 1996). Furthermore, the number of horseshoe vortices decreases downstream due to a subharmonic instability (Mittal and Balachandar, 1997). Previous numerical studies (Mittal and Balachandar, 1995; Mittal and Balachandar, 1997) determined that sufficient spanwise resolution is imperative to obtain accurate estimates of the lift and drag coefficients and the Strouhal number of vortex shedding.

In our simulation, we use a spanwise period of $c = 12D$ where D is the diameter of the cylinder, with 288 Fourier coefficients at the finest refinement level. Three levels of refinement ($l_{max} = l_{ref} = 3$, $l_{add} = 0$) are used, and thus the number of Fourier coefficients for computational cells on the coarsest level is 36. The base resolution is set to be $x_{base}/D = 0.08$. The mesh at each refinement level is increasingly refined with a factor of 2. Thus, the resolution on the finest level is $\Delta x_3/D = 0.01$. The

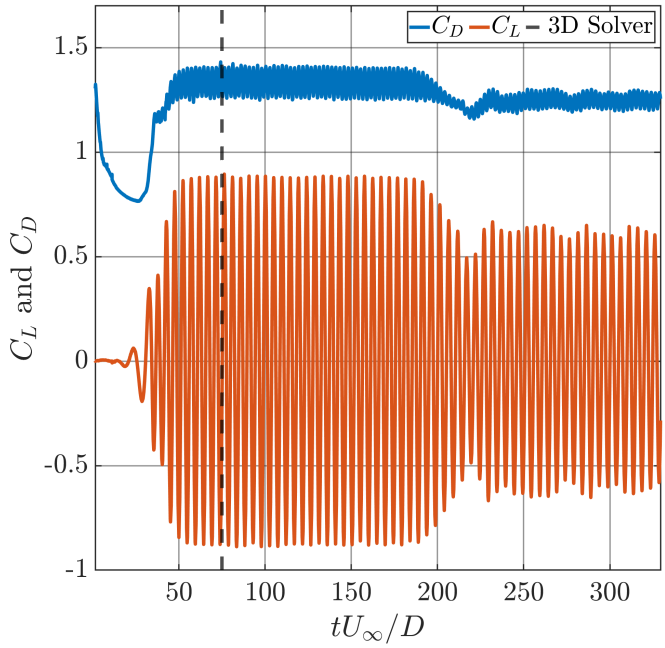


Figure 3.8: Lift and drag coefficient history of the simulation

adaptive mesh refinement algorithm locally refines and coarsens the computational domain with a refinement factor of $\alpha = 0.25$ and a deletion factor of $\beta = 0.7$. The time step size is chosen as $\Delta t U_\infty/\Delta x_3 = 0.75$.

To efficiently simulate this flow, we initialized the simulation by first computing the flow in 2D. After we reach a temporally periodic solution, we initialize the 3D simulation using the 2D solution as the zeroth Fourier coefficient. At the beginning of the 3D simulation, a small (on the scale of 10^{-5}) random vortical perturbation is introduced, with the expectation that the resulting flow becomes independent of the specific perturbation (Mittal and Balachandar, 1997). Integrating forward in time results in the lift C_L and drag C_D coefficients shown in Figure 3.8, where the dashed line denotes the initiation of the 3D simulation.

The three-dimensional instability is slow to develop, reaching a significant amplitude only by $tU_\infty/D = 175$ and saturating thereafter. The flow is (approximately) stationary after $tU_\infty/D = 225$. These dynamics are similar to what has been previously observed (Mittal and Balachandar, 1995; Mittal and Balachandar, 1997).

The vorticity magnitude and streamwise vorticity at $tU_\infty/D = 367.5$ are shown in Figures 3.9 and 3.10, respectively. We can clearly observe the three-dimension mode-B vortices forming in the wake near the cylinder and the elongated horseshoe

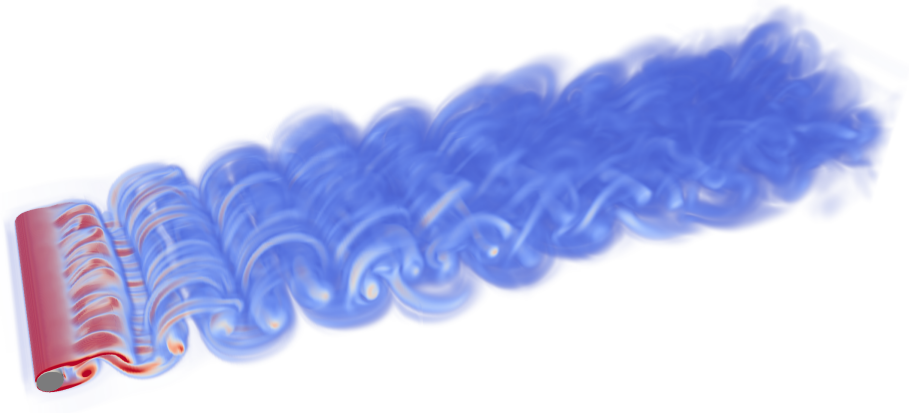


Figure 3.9: Vorticity magnitude at $tU_\infty/D = 367.5$. The non-dimensionalized vorticity, $\omega D/U_\infty$, magnitude ranges from 0 (blue) to 5 (red).

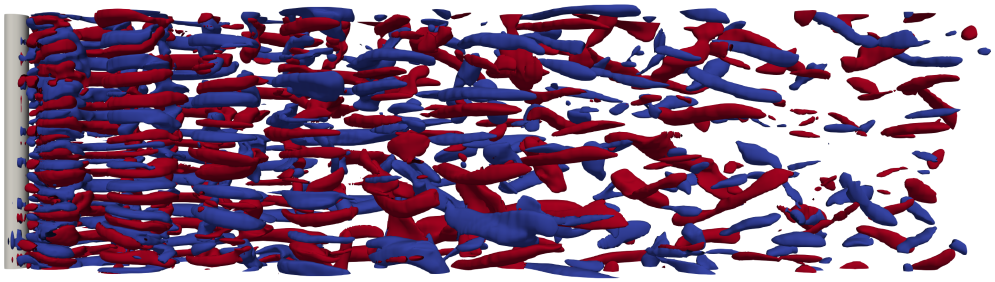


Figure 3.10: Streamwise vorticity contour plot at $tU_\infty/D = 367.5$ at $\omega_x D/U_\infty = 0.5$ (red) and $\omega_x D/U_\infty = -0.5$ (blue).

vortices further downstream, with a continual decrease in the number of horseshoe vortices as the flow progresses. The formation of the horseshoe vortices and the reduction of the number of horseshoe vortices downstream indicates the subharmonic instability of the three-dimensional vortices generated by mode-B instability (Williamson, 1996; Mittal and Balachandar, 1997; Kravchenko, Moin, and Shariff, 1999).

In addition to these qualitative observations, we computed the average drag coefficient ($\overline{C_D}$), root mean squared lift coefficient ($C_{L,rms}$), and Strouhal number (St) of this simulation. The mean is computed starting at $tU_\infty/D = 245.7$, which is more than 12 shedding cycles after the onset of three-dimensional instability. The statistics were accumulated for the next 14 shedding cycles. In Table 3.3, we report those

Study	$\overline{C_D}$	$C_{L,rms}$	St
KMS99 (num.)	1.28	0.40	0.203
MB97 (num.)	1.26	0.38	0.203
Experimental	1.22	0.45	0.203
Present	1.25	0.44	0.203

Table 3.3: Comparison of lift and drag statistics with previous studies. Numerical results from KMS99 - Kravchenko, Moin, and Shariff (1999), MB97 - Mittal and Balachandar (1997). The experimental results are from Wieselsberger (1922) and Norberg (2003).

statistics and compare the values to previous numerical and experimental studies. All quantities are in reasonable agreement with the reference values.

3.11.2 $Re = 12,000$

We now consider the fully turbulent cylinder flow at $Re = 12,000$. We carried out the simulation using 128 Fourier coefficients and spanwise periodic length $c = 3D$. In this simulation, we used a base resolution $\Delta x_{base}/D = 0.04$ with four levels of adaptive refinement, yielding the finest resolution of $\Delta x_4/D = 0.0025$. The refinement factor is $\alpha = 0.25$, and the deletion factor is $\beta = 0.7$. At $Re = 3,900$, the average Kolmogorov scale ($\bar{\eta}$) in the near wake ($x/D < 5$) is $\bar{\eta}/D = 0.02$ (Lehmkuhl et al., 2013). According to the $3/4$ scaling of the Kolmogorov scale, we estimate that $\bar{\eta}/D = 0.0086$ at $Re = 12,000$. The ratio of the Kolmogorov scale to the second finest level is $\Delta x_3/\bar{\eta} = 0.58$. Thus, the turbulence is expected to be fully resolved. The finest level is required to resolve the thin laminar boundary layers on the cylinder prior to separation and the shear layer attached to the cylinder.

To efficiently reach the fully developed turbulent flow, we adopted a step-wise strategy. Specifically, we first initialize a 2D simulation of flow past a cylinder at $Re = 5,000$. After vortex shedding initiates, we use that flow profile as the zeroth Fourier coefficient to initialize a fully 3D simulation with 64 spanwise Fourier coefficients. We perturb this 3D simulation with a small random vortical perturbation (on the level 10^{-5}) to trigger the spanwise instability. After the flow becomes fully turbulent (at $tU_\infty/D \approx 25$), we increase the Reynolds number to 12,000 and increase the number of spanwise Fourier coefficients to 128 to continue our simulation. The simulation is then continued for more than $125tU_\infty/D$. Throughout the simulation, the time step size is chosen to be $\Delta tU_\infty/\Delta x = 0.5$. The value is chosen to satisfy the CFL criterion (Courant, Friedrichs, and Lewy, 1928).

Figure 3.11 shows the isosurface of the Q-criterion at $tU_\infty/D = 145.75$. We can

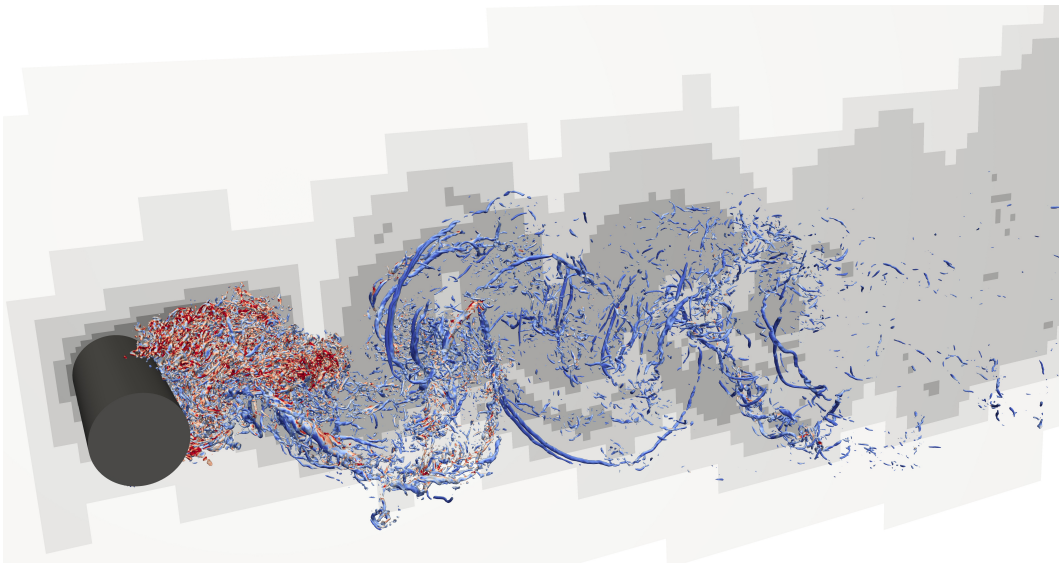


Figure 3.11: Q-criterion isosurface at $Q = 100U_\infty/D$. The isosurface is colored by the vorticity magnitude from $||\boldsymbol{\omega}||D/U_\infty = 20$ (blue) to 50 (red). The corresponding $x-y$ computational mesh is shown in the background. The mesh gets increasingly fine as the gray darkens. The computational domain is truncated in this figure. The full computational domain is adaptive and extends to 23D downstream.

clearly observe the rib structures in the wake that are also present in the wake of the flow past a flat plate (Rai, 2013). We also show the $x-y$ mesh topology in the same figure. The computational grid tracks the vortical region of the flow and adaptively refines in the high vorticity region. We computed statistics using the data within the time period $tU_\infty/D \in [105, 145]$, corresponding to approximately 8 vortex-shedding cycles. When estimating a Strouhal number, we conduct a Fourier transform of the lift coefficient data in time. The resulting Fourier spectrum is shown in Figure 3.12. The peak of the Fourier spectrum centers around $St = 0.198$.

The drag coefficient and lift coefficient time evolution data for $tU_\infty/D \in [100, 145]$ are shown in Figure 3.13. The mean drag and Strouhal number are compared to experimental data in Table 3.4. The Strouhal number and $C_{L,rms}$ are computed from an empirical formula obtained from experimental data (Norberg, 2003) when $Re \in [1600, 150000]$ for St and $Re \in [5400, 220000]$ for $C_{L,rms}$:

$$St = 0.1853 + 0.0261 \times \exp(-0.9 \times x^{2.3}), \qquad x = \log(Re/1600), \tag{3.64}$$

$$C_{L,rms} = 0.52 - 0.06 \times x^{-2.6}, \qquad x = \log(Re/1600). \tag{3.65}$$

The drag coefficient and Strouhal number agree with the reference values within uncertainty associated with the empirical formula (see Figures 1 and 2 in (Norberg,

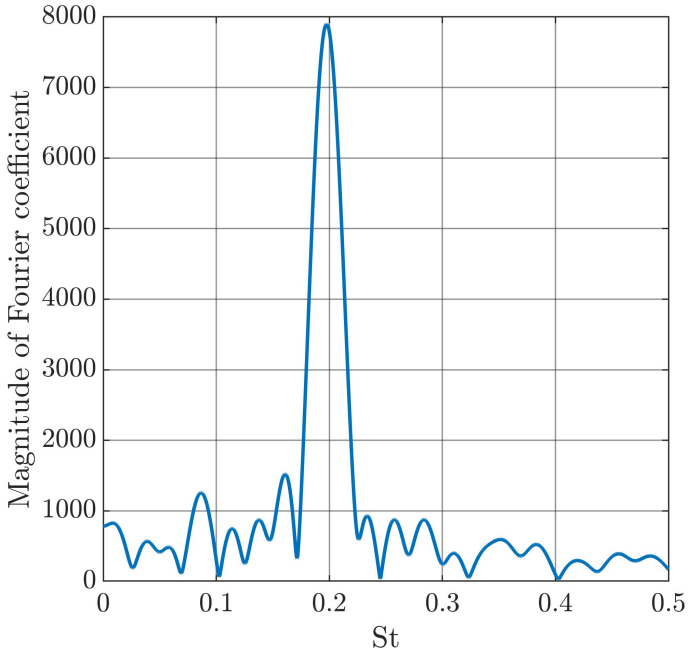


Figure 3.12: Fourier spectrum of lift coefficient.

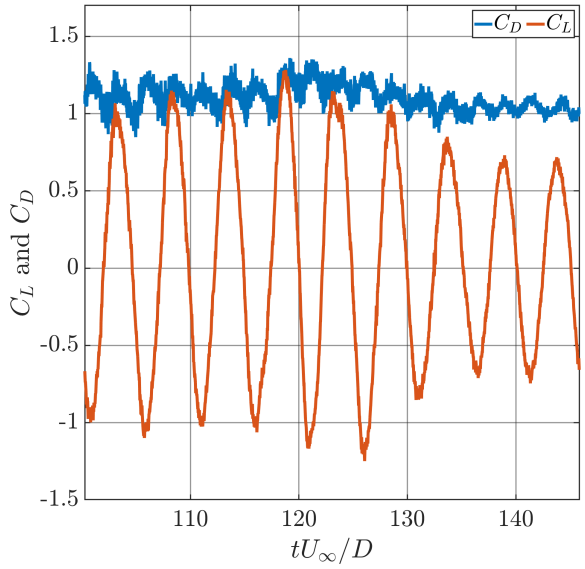


Figure 3.13: Drag coefficient and lift coefficient time evolution during $tU_\infty/D \in [100, 145]$.

Study	$\overline{C_D}$	$C_{L,rms}$	St
Norberg (2003)	-	0.435	0.199
Wieselsberger (1922)	1.15	-	-
Present	1.12	0.67	0.198

Table 3.4: Drag coefficient, lift coefficient, and Strouhal number comparison between present numerical method and experimental data for the flow past a cylinder at $Re = 12,000$.

2003)). On the other hand, $C_{L,rms}$ is about 50% higher than the formula. As can be observed in Figure 3.13, there is a more significant variation in the lift from cycle to cycle, and the discrepancy is likely attributable to the statistic not having converged.

3.12 Summary

We have extended the IB-LGF-AMR approach previously developed for unbounded three-dimensional flows to solve external flows around cylinders of arbitrary cross-sections. To this end, we hybridized a mimetic, staggered-mesh finite-volume discretization for the inhomogeneous directions with a Fourier spectral method for the homogeneous spanwise direction, using a pseudo-spectral approach for the nonlinear term. Some innovations include the implementation of AMR on the Fourier coefficients, a direct LU solution of the system of equations for the IB forcing, and new algorithms for efficiently tabulating the associated LGFs for the viscous integrating factor to high precision. The validated algorithm retains the flexibility, scalability, and accuracy of the IB-LGF-AMR approach, as demonstrated by computing the fully turbulent flow past a circular cylinder at $Re = 12,000$.

Chapter 4

THREE-DIMENSIONAL STABILITY AND RESOLVENT ANALYSES OF EXTERNAL FLOWS OVER SPANWISE-HOMOGENEOUS IMMERSED BODIES

This chapter is adapted from Hou and Colonius (2023). In this chapter, we introduce a numerical framework to conduct stability and resolvent analysis of spanwise periodic flows. This numerical framework leverages lattice Green's function, adaptive mesh refinement, and immersed boundary method to create an efficient and versatile method to conduct stability analysis and resolvent analysis. The focus of this chapter is to introduce this numerical framework and also show some validating examples.

4.1 Introduction

Stability and resolvent analyses are important flow analysis tools. Stability analysis is used to study the evolution of disturbances to (typically) steady, laminar flows and is conducted by analyzing the eigenvalues of the linearized Navier-Stokes equations (Taira, Brunton, et al., 2017) to determine if an equilibrium is stable. Recently, numerical methods for stability analysis have been developed to study more complicated flow configurations (Theofilis, 2011). However, linear stability analysis fails to predict the onset of instability for many flows, such as Poiseuille flow (Orszag, 1971a). To remedy this discrepancy, resolvent analysis is developed to study the transient growth of perturbations to an equilibrium (Trefethen, Trefethen, et al., 1993). This transient growth is used to explain the flow instability that occurs even when stability analysis shows that the equilibrium point is stable. Moreover, resolvent analysis performed around the mean flow profile is useful to study both laminar and turbulent (mean) flow fields for characterizing coherent structures as high-gain responses to generic nonlinear forcing (Taira, Brunton, et al., 2017; McKeon and Sharma, 2010). In addition, resolvent analysis can also be used to perform optimization on aerodynamic performance (Yeh and Taira, 2019).

Both analyses require the construction of the linearized Navier-Stokes equation (LNSE) around an appropriate base flow (Taira, Brunton, et al., 2017). Even though both analyses are vastly useful, one major challenge that limits their applications is the construction and LU decomposition of the linear operator arising from discretizing LNSE (Taira, Brunton, et al., 2017; Barthel, Gomez, and McKeon, 2022).

Especially for external flows around immersed bodies, the domain required to resolve the flow is much larger than the characteristic length of the immersed body (Pralits, Giannetti, and Brandt, 2013; Ribeiro, Yeh, and Taira, 2020). The expansive domain further exacerbates the computational cost when conducting those analyses. Considerable effort has been put into circumventing this problem using iterative methods and matrix-free methods (Bagheri et al., 2009; Martini et al., 2021). However, we tackle this problem head-on by reducing the size of the matrix. This reduction of matrix size is achieved by leveraging the IBLGF-AMR framework created by Yu (2021).

Based on the existing method of Liska and Colonius (2017), IBLGF-AMR is a numerical method developed by Yu (2021) to solve incompressible external flows around immersed bodies using lattice Green's function (LGF), adaptive mesh refinement (AMR), and immersed boundary (IB) method. This method is able to solve the external flows efficiently on multi-resolution meshes that adapt to the solution and need only have active grid points in vortical flow regions–exact far-field boundary conditions are applied via the Green's function.

In this chapter, we extend the IBLGF-AMR framework for flows with one homogeneous (periodic) direction and introduce a matrix-based framework to enable efficient solutions of the corresponding stability and resolvent problems. A key difficulty that is addressed in this work is how to retain a sparse matrix structure in the presence of the LGF approach to the Poisson equation arising in incompressible flows, such that modern multifrontal LU solvers can be efficiently employed to solve the stability and resolvent problems.

4.2 Problem statement

The flow problems we are interested in are external incompressible flows with one homogeneous direction, namely the flow over cylinders with arbitrary cross sections. In general, we can write the spatially discretized Navier-Stokes equation (NSE) as

$$\boldsymbol{M}\frac{d\boldsymbol{q}}{dt} = \boldsymbol{F}(\boldsymbol{q}), \tag{4.1}$$

where $\boldsymbol{q}$ is a vector of discretized flow variables, $\boldsymbol{M}$ is the diagonal matrix with ones for the entries corresponding to velocity variables and zeros otherwise, and $\boldsymbol{F}(\cdot)$ is the collection of all the terms in the discrete NSE except the time derivative. Linearizing this equation (around some base flow $\boldsymbol{q}_b$) and adding a forcing term $\boldsymbol{f}$ yields

$$\boldsymbol{M}\frac{d\boldsymbol{q}}{dt} = \boldsymbol{A}(\boldsymbol{q}_b)\boldsymbol{q} + \boldsymbol{f}. \tag{4.2}$$

The $\boldsymbol{A}(\boldsymbol{q}_b)$ operator is also referred as the linearized Navier-Stokes operator (Taira, Brunton, et al., 2017). For stability analysis, the linearization is about an equilibrium, $\boldsymbol{q}_b = \boldsymbol{q}_s$ satifying $\boldsymbol{F}(\boldsymbol{q}_s) = 0$ and, $\boldsymbol{f} = 0$. The equilibrium solution can be obtained using the nonlinear code directly (for stable flows) or using Newton's iterations for unstable ones

For resolvent analysis, $\boldsymbol{q}_b$ can be either an equilibrium, in which case $\boldsymbol{f} \neq 0$ is some exogenous forcing, or the mean of a turbulent flow, $\boldsymbol{q}_b = \overline{\boldsymbol{q}}$ in which case $\boldsymbol{f}$ comprises the residual nonlinear terms in the full nonlinear discretized Navier-Stokes equations.

For stability analysis, we solve the generalized eigenvalue problem

$$\lambda \boldsymbol{M}\boldsymbol{v} = \boldsymbol{A}(\boldsymbol{q}_s)\boldsymbol{v}, \tag{4.3}$$

and the fixed point is unstable if there exists at least one eigenvalue with a positive real part. More generally, we may seek to determine the full spectrum of disturbances and characterize, for stable systems, the extent of transient growth.

For resolvent analysis, on the other hand, we seek to determine, in the frequency domain, those forcings that are most amplified by the base flow, namely

$$\sigma^2 = \max \frac{< \hat{\boldsymbol{q}}, \hat{\boldsymbol{q}} >}{< \hat{\boldsymbol{f}}, \hat{\boldsymbol{f}} >}, \tag{4.4}$$

where $\hat{(\cdot)}$ refers to the (temporal) Fourier transform and $< \cdot, \cdot >$ refers to the appropriate inner product. For example, this inner product can be induced from Chu's energy norm (Kamal, Lakebrink, and Colonius, 2023) or the discrete counterpart of the continuous L_2 inner product (Towne, Schmidt, and Colonius, 2018). The inputs, $\hat{\boldsymbol{f}}$ are related to the outputs, $\hat{\boldsymbol{q}}$, through the solution of Eq. (4.2) by the resolvent operator

$$\hat{\boldsymbol{q}} = \boldsymbol{R}(\omega)\hat{\boldsymbol{f}}, \tag{4.5}$$

where

$$\boldsymbol{R}(\omega) = \boldsymbol{C}(i\omega\boldsymbol{M} - \boldsymbol{A}(\overline{\boldsymbol{q}}))^{-1}\boldsymbol{B}, \tag{4.6}$$

in which $\boldsymbol{B}$ defines the subspace of the feasible forcing $\hat{\boldsymbol{f}}$, and $\boldsymbol{C}$ defines the subspace of the output $\hat{\boldsymbol{q}}$. The solution to the maximization (Rayleigh quotient) is given by the singular value decomposition (SVD) of $\boldsymbol{R}$, namely σ is the first (largest) singular value of $\boldsymbol{R}$ and the input and output giving rise to the largest amplification are given by the corresponding left and right singular vectors. We also call this σ the resolvent

norm (Symon, 2018) since it coincides with the definition of the operator norm of the resolvent operator, i.e.,

$$\sigma = ||\boldsymbol{R}(\omega)||. \tag{4.7}$$

For spanwise-homogeneous flows, we can carry out a Fourier expansion in that homogeneous direction. We can transform Eq. 4.2 into the following form for the k^{th} Fourier mode:

$$\tilde{\boldsymbol{M}}\frac{d\tilde{\boldsymbol{q}}_k}{dt} = \tilde{\boldsymbol{A}}_k(\boldsymbol{q}_b)\tilde{\boldsymbol{q}}_k + \tilde{\boldsymbol{f}}_k. \tag{4.8}$$

The corresponding generalized eigenvalue problem for stability analysis is

$$\lambda\tilde{\boldsymbol{M}}\boldsymbol{v} = \tilde{\boldsymbol{A}}_k(\boldsymbol{q}_b)\boldsymbol{v}, \tag{4.9}$$

and the resolvent analysis becomes the SVD of the Fourier-transformed resolvent operator $\tilde{\boldsymbol{R}}_k(\omega)$

$$\tilde{\boldsymbol{R}}_k(\omega) = \boldsymbol{C}(i\omega\tilde{\boldsymbol{M}} - \tilde{\boldsymbol{A}}_k(\overline{\boldsymbol{q}}))^{-1}\boldsymbol{B}. \tag{4.10}$$

This spanwise-homogeneous formulation is also referred to as BiGlobal (Theofilis, 2011) stability (Eq. 4.9) and resolvent (Eq. 4.10) analysis. As mentioned in the previous section, the key to an efficient solution of either the stability or resolvent method is for the corresponding $\boldsymbol{M}$ and $\boldsymbol{A}$ matrices to be as sparse and structured as possible, such that modern multifrontal LU solvers achieve their best performance. Unfortunately, a direct linearization of the IBLGF-AMR solver will result in dense matrices, since the LGF represents a nonlocal operator. We pursue here an alternative strategy of using the LGF to form boundary conditions so that the forward Laplacian appears in the operators instead of the LGF.

4.3 Derivation and discretization of governing equations

The Navier-Stokes equations with immersed boundary forcing are

$$\begin{aligned}
\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{\omega}\times\boldsymbol{u} &= -\nabla p - \frac{1}{\mathrm{Re}}\nabla\times\boldsymbol{\omega} + \int_\Gamma \boldsymbol{f}_\Gamma(\boldsymbol{\xi},t)\delta(\boldsymbol{X}(\boldsymbol{\xi})-\boldsymbol{x})d\boldsymbol{\xi} \\
\nabla.\boldsymbol{u} &= 0 \\
\boldsymbol{u}_\Gamma(\boldsymbol{\xi},t) &= \int_{\mathbb{R}^3}\boldsymbol{u}(\boldsymbol{x},t)\delta(\boldsymbol{x}-\boldsymbol{X}(\boldsymbol{\xi}))d\boldsymbol{x}.
\end{aligned} \tag{4.11}$$

where

$$\boldsymbol{\omega} = \nabla\times\boldsymbol{u}, \tag{4.12}$$

and the immersed boundary Γ is parameterized by ξ. $X(\cdot)$ is that parametrization function to describe the immersed boundary from the parametrizing variable ξ. $\boldsymbol{u}_\Gamma$

is the boundary conditions that need to be satisfied on the boundary Γ, and $\boldsymbol{f}_\Gamma$ is the immersed boundary forcing induced by enforcing the boundary conditions. $\delta(\cdot)$ represents the Dirac delta distribution and is used here to enforce boundary conditions and to represent the effect of immersed boundary forcings on the flow field. Using the immersed boundary, we are able to conduct flow analysis around arbitrary geometries without changing the underlying discretization.

For a time-invariant solution, $\boldsymbol{u}_s(x, y)$ and $p_s(x, y)$ the linearization of Eq. 4.11 around this flow is

$$\begin{aligned}
\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{\omega} \times \boldsymbol{u}_s + \boldsymbol{\omega}_s \times \boldsymbol{u} &= -\nabla p - \frac{1}{\mathrm{Re}} \nabla \times \boldsymbol{\omega} + \int_\Gamma \boldsymbol{f}_\Gamma(\boldsymbol{\xi}) \delta(\boldsymbol{X}(\boldsymbol{\xi}) - \boldsymbol{x}) d\boldsymbol{\xi} \\
\nabla \cdot \boldsymbol{u} &= 0 \\
0 &= \int_{\mathbb{R}^3} \boldsymbol{u}(\boldsymbol{x}, t) \delta(\boldsymbol{x} - \boldsymbol{X}(\boldsymbol{\xi})) d\boldsymbol{x}
\end{aligned} \tag{4.13}$$

Due to the homogeneity of the base flow and the immersed body, the governing equations can be Fourier expanded in the homogeneous direction. By doing so, we obtain a set of linear evolution equations for the Fourier coefficients of the state variables, i.e. $\boldsymbol{u}$ and p. Then for the k^{th} Fourier coefficient ($\tilde{\boldsymbol{u}}_k$ and $\tilde{p}_k$), the equations is

$$\begin{aligned}
\frac{\partial \tilde{\boldsymbol{u}}_k}{\partial t} + \tilde{\boldsymbol{\omega}}_k \times \boldsymbol{u}_s + \boldsymbol{\omega}_s \times \tilde{\boldsymbol{u}}_k &= -\widetilde{\nabla}_k \tilde{p}_k - \frac{1}{\mathrm{Re}} \widetilde{\nabla}_k \times \tilde{\boldsymbol{\omega}}_k \\
&\quad + \int_{\Gamma_{2D}} \tilde{\boldsymbol{f}}_{\Gamma,k}(\boldsymbol{\xi}_{2D}) \delta_{2D}(\boldsymbol{X}_{2D}(\boldsymbol{\xi}_{2D}) - \boldsymbol{x}_{2D}) d\boldsymbol{\xi}_{2D} \\
\widetilde{\nabla}_k \cdot \tilde{\boldsymbol{u}}_k &= 0 \\
0 &= \int_{\mathbb{R}^2} \tilde{\boldsymbol{u}}_k(\boldsymbol{x}_{2D}, t) \delta_{2D}(\boldsymbol{x}_{2D} - \boldsymbol{X}_{2D}(\boldsymbol{\xi}_{2D})) d\boldsymbol{x}_{2D}
\end{aligned} \tag{4.14}$$

where

$$\widetilde{\nabla}_k = [\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, i2\pi k]^T, \tag{4.15}$$

$$\widetilde{\boldsymbol{\omega}}_k = \widetilde{\nabla}_k \times \widetilde{\boldsymbol{u}}_k. \tag{4.16}$$

4.3.1 Discretization

Since the evolution equations of the Fourier coefficients are decoupled, they are inherently 2D. Thus, we design a compatible 2D discretization scheme to solve these equations. The placement of variables in a Cartesian grid cell is shown in Figure 4.1. We define discrete operators as the following:

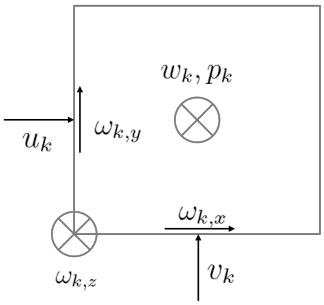


Figure 4.1: Variable placement of one cell in k^{th} Fourier interpolation sampling point.

- Divergence on k^{th} Fourier coefficient

$$\boldsymbol{D}_k\boldsymbol{g} = \mathcal{D}_x g_1 + \mathcal{D}_y g_2 + (2\pi ik/c)g_3, \tag{4.17}$$

- Gradient on k^{th} Fourier coefficient

$$\boldsymbol{G}_k g = [-\mathcal{D}_x^T g, -\mathcal{D}_y^T g, (2\pi ik/c)g]^T, \tag{4.18}$$

- Curl on k^{th} Fourier coefficient

$$\boldsymbol{C}_k\boldsymbol{g} = [\mathcal{D}_y g_3 - (2\pi ik/c)g_2, (2\pi ik/c)g_1 - \mathcal{D}_x g_3, D_x g_2 - \mathcal{D}_y g_1]^T, \tag{4.19}$$

- Laplacian on k^{th} Fourier coefficient

$$\boldsymbol{L}_k g = -\mathcal{D}_x^T\mathcal{D}_x g - \mathcal{D}_y^T\mathcal{D}_y g - (2\pi k/c)^2 g, \tag{4.20}$$

Here, the boldfaced variables, e.g. $\boldsymbol{g} = [g_1, g_2, g_3]^T$, to denote a 3-component vector field and non-boldfaced variables, e.g. g, to denote a scalar field. $\mathcal{D}$ denotes the finite difference derivative and the subscript denotes the direction that the derivative is taken. For example, $\mathcal{D}_x g(i,j) = [g(i+1,j) - g(i,j)]/\Delta x$. The resulting discretized version of linearized Navier-Stokes equation is

$$\begin{aligned}
\frac{d\tilde{\boldsymbol{u}}_k}{dt} + \tilde{\boldsymbol{\omega}}_k \times \boldsymbol{u}_s + \boldsymbol{\omega}_s \times \tilde{\boldsymbol{u}}_k &= -\boldsymbol{G}_k\tilde{p}_k - \frac{1}{\mathrm{Re}}\boldsymbol{C}_k^*\tilde{\boldsymbol{\omega}}_k + \boldsymbol{P}^T\tilde{\boldsymbol{f}}_{\Gamma,k} \\
\boldsymbol{D}_k\tilde{\boldsymbol{u}}_k &= 0 \\
\boldsymbol{P}\tilde{\boldsymbol{u}}_k &= 0 \\
\boldsymbol{C}_k\tilde{\boldsymbol{u}}_k &= \tilde{\boldsymbol{\omega}}_k
\end{aligned} \tag{4.21}$$

where $\boldsymbol{P}$ is the projection operator from the entire computational domain to the set of IB points. Thus, in matrix form, we can write

$$\frac{d}{dt}\begin{bmatrix} \boldsymbol{I} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} \tilde{\boldsymbol{u}}_k \\ \tilde{p}_k \\ \tilde{\boldsymbol{\omega}}_k \\ f_k \end{bmatrix} = \begin{bmatrix} -(\boldsymbol{\omega}_s\times) & -\boldsymbol{G}_k & -\frac{1}{\mathrm{Re}}\boldsymbol{C}_k^T + \boldsymbol{u}_s\times & \boldsymbol{P}^T \\ \boldsymbol{D}_k & 0 & 0 & 0 \\ \boldsymbol{C}_k & 0 & -\boldsymbol{I} & 0 \\ \boldsymbol{P} & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} \tilde{\boldsymbol{u}}_k \\ \tilde{p}_k \\ \tilde{\boldsymbol{\omega}}_k \\ f_k \end{bmatrix} \tag{4.22}$$

or in a more compact form

$$\boldsymbol{M}\frac{d\boldsymbol{q}_k}{dt} = \boldsymbol{A}_k\boldsymbol{q}_k, \tag{4.23}$$

where

$$\boldsymbol{q}_k = \begin{bmatrix} \tilde{\boldsymbol{u}}_k \\ \tilde{p}_k \\ \tilde{\boldsymbol{\omega}}_k \\ f_k \end{bmatrix} \tag{4.24}$$

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{I} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.25}$$

and

$$\boldsymbol{A}_k = \begin{bmatrix} -(\boldsymbol{\omega}_s\times) & -\boldsymbol{G}_k & -\frac{1}{\mathrm{Re}}\boldsymbol{C}_k^* + \boldsymbol{u}_s\times & \boldsymbol{P}^T \\ \boldsymbol{D}_k & 0 & 0 & 0 \\ \boldsymbol{C}_k & 0 & -\boldsymbol{I} & 0 \\ \boldsymbol{P} & 0 & 0 & 0 \end{bmatrix}. \tag{4.26}$$

4.3.2 Multi-resolution considerations

To construct the above operators on a multi-resolution mesh, we use the same technique employed by Yu (2021). We briefly summarize this approach.

Suppose we have a series of Cartesian grids with different resolutions $\{\mathbb{R}_i\}$, $i \in \mathbb{Z} \cap [0, N_l]$ containing $N_l + 1$ increasingly refined unbounded meshes. i refers to the refinement level. $\mathbb{R}_0$ has the coarsest spatial resolution, denoted as Δx_0. Then, $\mathbb{R}_k$ has spatial resolution of $\Delta x_k = \Delta x_0/2^k$. Then we define a partition of $\mathbb{R}^2$ using $N_l + 1$ disjoint domains, denoted as $\{\Omega_i\}$, $i \in \mathbb{Z} \cap [0, N_l]$. Each Ω_i denotes the active region in which operators on $\mathbb{R}_i$ are active. This procedure is defined using the indicator functional Γ_{Ω_i} defined as

$$(\Gamma_{\Omega_i} f)(\boldsymbol{x}) = \begin{cases} f(\boldsymbol{x}) & \boldsymbol{x} \in \Omega_i \\ 0 & \boldsymbol{x} \notin \Omega_i \end{cases}. \tag{4.27}$$

Then if $\boldsymbol{A}_i$ is the a linear operator defined on mesh $\mathbb{R}_i$, the multi-resolution version of this operator is defined as

$$\boldsymbol{A} = \sum_{i=0}^{N_l} \Gamma_{\Omega_i} \boldsymbol{A}_i \tag{4.28}$$

and the active AMR grid is defined as

$$\mathbb{R}_{AMR} = \bigcup_{i=0}^{N_l} \mathbb{R}_i \cap \Omega_i. \tag{4.29}$$

Then, for each point in $\mathbb{R}_i \backslash \Omega_i \ \forall i$, the value can be obtained using interpolation. In our approach, coarse-to-fine mapping is via the 5th-order Lagrange interpolation, and fine-to-coarse mapping is via cell-centered averaging, which is equivalent to linear interpolation in our case. Thus, the values of the state vector that we need to keep are values associated with grid points in $\mathbb{R}_{AMR}$ and those immediately next to $\mathbb{R}_{AMR}$. To facilitate discussion in the next section, we use $\boldsymbol{P}_{\rightarrow 0}$ to denote the series of averaging operations to obtain values on all of $\mathbb{R}_0$

4.3.3 Compactness assumptions and boundary conditions

Operators and algorithms defined above are valid for unbounded computational meshes. To restrict our computational mesh to a finite one, our algorithm assumes that the vorticity field is compactly supported in a domain Ω. In practice, we designate a cut-off magnitude for vorticity and only retain the part of the domain with a vorticity magnitude higher than this value. The consequences of this truncation have been assessed in detail by Liska and Colonius (2017), and it is possible to define thresholds that have a minor impact on the flow while, at the same time, resulting in snug computational domains. Combined with our formulation of linearized Navier-Stokes equations, we have the following compactness assumption

$$|\boldsymbol{\omega}_s(\boldsymbol{x})| = |\tilde{\boldsymbol{\omega}}_k(\boldsymbol{x})| = 0 \text{ if } \boldsymbol{x} \notin \Omega. \tag{4.30}$$

Thus, we only need to pose a boundary condition on $\tilde{\boldsymbol{u}}_k$ and $\tilde{p}_k$. We obtain the boundary condition of $\tilde{\boldsymbol{u}}_k$ by solving

$$-\boldsymbol{L}_k \tilde{\boldsymbol{s}}_k = \tilde{\boldsymbol{\omega}}_k, \quad \tilde{\boldsymbol{u}}_k = \boldsymbol{C}_k^* \tilde{\boldsymbol{s}}_k, \tag{4.31}$$

and the boundary condition of $\tilde{p}_k$ by solving

$$-\boldsymbol{L}_k \tilde{p}_k = \boldsymbol{D}_k (\tilde{\boldsymbol{\omega}}_k \times \tilde{\boldsymbol{u}}_k). \tag{4.32}$$

To solve these, we use Lattice Green's Function (LGF) to obtain the boundary condition. Specifically, the LGF ($\boldsymbol{L}_k^{-1}$) is the analytical inverse of the $\boldsymbol{L}_k$ operator in free space. It is defined as the following in the unbounded domain

$$\boldsymbol{L}_k u = f \Rightarrow u = \boldsymbol{L}_k^{-1} f. \tag{4.33}$$

Here $\boldsymbol{L}_k^{-1}$ is a discrete convolution operator and each entry in its kernel can be computed using a 1-D integral. We will not explain the specific derivations in detail but refer readers to (Buneman, 1971; Martinsson and Rodin, 2002).

4.3.4 Applying boundary condition through one-side fast multipole method (O-FMM)

Since the operator $\boldsymbol{L}_k^{-1}$ is a discrete convolution operator, it is a dense operator if applied directly. In numerical simulations, we can use FFT-based convolution and fast multipole method (FMM) to apply this operator in linear time (Ying, Biros, and Zorin, 2004; Liska and Colonius, 2014; Liska and Colonius, 2017). However, in the context of the stability and resolvent analysis, we need to explicitly construct this operator. Thus, FFT becomes infeasible. Instead, we can adapt FMM to sparsify this operator.

4.3.4.1 Interpolation-based kernel independent fast multipole method

In this section, we briefly introduce the FMM method created by Liska and Colonius ((Liska and Colonius, 2014)), which we later adapt to sparsify the LGF operator.

Suppose we have a kernel $K(x, y)$ and would like to apply it on a discrete source field f in some domain Ω. Directly convolution gives the following

$$u(x_i) = \sum_{y_j \in \Omega} K(x_i, y_j) f(y_j). \tag{4.34}$$

Applying this formula to all points in Ω yields $O(N^2)$ complexity, given N as the number of points in Ω.

Now, suppose we have a set of interpolation functions $[\phi]$

$$[\phi] = [\phi_0, \phi_1, ..., \phi_{n-1}], \tag{4.35}$$

then we can approximate $K(x, y)$ using the following

$$K(x, y) \approx K^n(x, y) = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} \phi_i(x) K(x_i, y_j) \phi_j(y). \tag{4.36}$$

Using this formula, a possibly infinite-dimensional kernel $K(x, y)$ is approximated, or "compressed", using a set of finite sampling points: $\{K(x_i, y_j) \mid i, j \in [0, n-1] \cap \mathbb{Z}\}$. This step is termed "kernel compression". Plugging in this approximation to the direction convolution formula gives

$$\begin{aligned} u(x_i) &= \sum_{y_j \in \Omega} K(x_i, y_j) f(y_j) \\ &\approx \sum_{y_j \in \Omega} K^n(x_i, y_j) f(y_j) \\ &= \sum_{y_j \in \Omega} \sum_{l=0}^{n-1} \sum_{k=0}^{n-1} \phi_l(x_i) K(x_l, y_k) \phi_k(y_j) f(y_j). \end{aligned} \tag{4.37}$$

Use this formula, we can simplify the application of $K(x, y)$ using the following procedure

- Compute effective source distribution

$$\tilde{f}(y_k) = \sum_{y_j \in \Omega} \phi_k(y_j) f(y_j), \tag{4.38}$$

- Apply compressed kernel

$$\tilde{u}(x_l) = \sum_{k=0}^{n-1} K(x_l, y_k) \tilde{f}(y_k), \tag{4.39}$$

- Interpolate to get the solution

$$u(x_i) \approx \sum_{l=0}^{n-1} \phi_l(x_i) \tilde{u}(x_l). \tag{4.40}$$

We can use this formula to approximate the application of LGF by partitioning the domain into smaller blocks and applying the above formula recursively for blocks with different distances. Specifically, a tree structure is used by Liska and Colonius (2014) and its multi-resolution counterpart by Dorschner et al. (2020). Using the tree structure with $l_{max} + 1$ levels, ranging from level $-l_{max}$ to 0, Liska and Colonius (2014) proposed the following procedure:

- Upward pass
  Recursively computing, from level 0 to level $-l_{max}$, $\tilde{f}_l(y_k) = \sum_{y_j \in \Omega} \phi_k(y_j) \tilde{f}_{l+1}(y_j)$ with $\tilde{f}_0(y_k) = f(y_k)$,

- Level interaction
  On each level, applying $\tilde{u}_l(x_m) = \sum_{y_k \in \mathcal{S}_0(x_m,l)} K(x_m, y_k)\tilde{f}_l(y_k)$,

- Downward pass
  Interpolating and accumulating to get results, from level $-l_{max}$ to level 0, $u_l(x_i) \approx \tilde{u}_l(x_i) + \sum_{k=0}^{n-1} \phi_k(x_i)u_{l+1}(x_k)$.

where $\mathcal{S}_0(x_m, l)$ is the set of source points to be applied on level l to point x_m according to the tree structure.

4.3.4.2 One-side FMM

In our case, however, we do not need to obtain the solution everywhere in the domain. Instead, only the values on the boundary are needed. That means we can modify the above procedure to obtain the boundary conditions more efficiently and accurately. Specifically, we compress the kernel using only one set of interpolations, i.e.

$$K(x, y) \approx \tilde{K}^n(x, y) = \sum_{j=0}^{n-1} K(x, y_j)\phi_j(y). \tag{4.41}$$

Then the solution at an arbitrary point, x_i, (retaining the same notation as the previous section) is

$$u(x_i) = \sum_{y_j \in \Omega} K(x_i, y_j)f(y_j) \approx \sum_{y_j \in \Omega} \tilde{K}^n(x_i, y_j)f(y_j) = \sum_{y_j \in \Omega} \sum_{k=0}^{n-1} K(x_i, y_k)\phi_k(y_j)f(y_j). \tag{4.42}$$

Similar to the FMM method used by Liska and Colonius (2014), a tree structure is also used to compute the coarse representations of the source field. The difference is that we no longer need to compute the downward pass. Instead, we compute the direct interaction between source points on each level in the tree and the target points on the boundary. So the procedure becomes

- Upward pass
  Recursively computing $\tilde{f}_l(y_k) = \sum_{y_j \in \Omega} \phi_k(y_j)\tilde{f}_{l+1}(y_j)$ with $\tilde{f}_0(y_k) = f(y_k)$,

- Inter-level interaction
  On level 0, applying $u(x_i) \approx \sum_{l=0}^{-l_{max}} \sum_{y_k \in \mathcal{S}(x_i,l)} K(x_i, y_k)\tilde{f}_l(y_k)$,

where $\mathcal{S}(x_i, l)$ is the set of source points to be applied on level l to point x_i according to the tree structure Then, we can integrate this into a matrix form using a set of constraining operators

$$\begin{bmatrix} 0 \\ 0 \\ \dots \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \Phi^T & -\boldsymbol{I} & 0 & \dots & 0 & 0 & 0 \\ 0 & \Phi^T & -\boldsymbol{I} & \dots & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & -\boldsymbol{I} & 0 & 0 \\ 0 & 0 & 0 & \dots & \Phi^T & -\boldsymbol{I} & 0 \\ \boldsymbol{A}_0 & \boldsymbol{A}_{-1} & \boldsymbol{A}_{-2} & \dots & \boldsymbol{A}_{-l_{max}+1} & \boldsymbol{A}_{-l_{max}} & -\boldsymbol{I} \end{bmatrix} \begin{bmatrix} \tilde{f}_0 \\ \tilde{f}_{-1} \\ \tilde{f}_{-2} \\ \dots \\ \tilde{f}_{-l+1} \\ \tilde{f}_{-l} \\ u_{BC} \end{bmatrix}, \tag{4.43}$$

where Φ is the matrix representation of the interpolation operators, and A_l is the operator to apply the compressed kernel on level l. Together, we can write the entire linearized and discretized Navier-Stokes equations as

$$\frac{d}{dt}\begin{bmatrix} \tilde{\boldsymbol{u}}_k \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} -(\boldsymbol{\omega}_s\times) & -\boldsymbol{G}_k & -\frac{1}{\mathrm{Re}}\boldsymbol{C}_k^* + \boldsymbol{u}_s\times & \boldsymbol{P}^T & 0 & 0 & \boldsymbol{M}_1 & 0 & 0 & \boldsymbol{M}_2 \\ \boldsymbol{D}_k & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \boldsymbol{D}_{k,BC} \\ \boldsymbol{C}_k & 0 & -\boldsymbol{I} & 0 & 0 & 0 & 0 & 0 & 0 & \boldsymbol{C}_{k,BC} \\ \boldsymbol{P} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\boldsymbol{P}_{\to 0}\boldsymbol{D}_k(\boldsymbol{\omega}_s\times) & 0 & \boldsymbol{P}_{\to 0}\boldsymbol{D}_k(\boldsymbol{u}_s\times) & 0 & -\boldsymbol{I} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \Phi_e^T & \boldsymbol{TM} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\boldsymbol{L}_0^{-1} & -\boldsymbol{L}_c^{-1} & -\boldsymbol{I} & 0 & 0 & 0 \\ 0 & 0 & \boldsymbol{P}_{\to 0} & 0 & 0 & 0 & 0 & -\boldsymbol{I} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \Phi_e^T & \boldsymbol{TM} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\boldsymbol{C}_k^*\boldsymbol{L}_0^{-1} & -\boldsymbol{C}_k^*\boldsymbol{L}_c^{-1} & -\boldsymbol{I} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{u}}_k \\ \tilde{p}_k \\ \tilde{\boldsymbol{\omega}}_k \\ f_k \\ \tilde{g}_0 \\ \tilde{g}_c \\ p_{BC} \\ \tilde{m}_0 \\ \tilde{m}_c \\ u_{BC} \end{bmatrix} \tag{4.44}$$

where

$$\Phi_e^T = [\Phi, 0, ..., 0]^T \tag{4.45}$$

,

$$\boldsymbol{TM} = \begin{bmatrix} \Phi^T & -\boldsymbol{I} & 0 & \dots & 0 & 0 \\ 0 & \Phi^T & -\boldsymbol{I} & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & -\boldsymbol{I} & 0 \\ 0 & 0 & 0 & \dots & \Phi^T & -\boldsymbol{I} \end{bmatrix}, \tag{4.46}$$

and $\boldsymbol{M}_1, \boldsymbol{M}_2, \boldsymbol{D}_{k,BC}, \boldsymbol{C}_{k,BC}$ are appropriate boundary condition matrices. $\boldsymbol{L}_c^{-1}$ is the matrix arising from applying LGF using One-side FMM. Explicitly, it is written as

$$\boldsymbol{L}_c^{-1} = [\boldsymbol{L}_{-1}^{-1}, ..., \boldsymbol{L}_{-l_{max}}^{-1}]. \tag{4.47}$$

With these sparse matrices, we can readily conduct stability and resolvent analysis. We will present some validating results using this formulation in the subsequent sections.

4.3.4.3 A note on scalability

The aforementioned LNSE operator can be partitioned using the block-structured mesh employed by Yu (2021). Thus, the construction of LNSE operator and subsequent operations on this operator (e.g. LU decomposition, eigenvalue computations) can be parallelized according to the same block structure and are compatible with existing parallel solvers such as MUMPS (Amestoy et al., 2001) and SLEPc (Hernandez, Roman, and Vidal, 2005). Thus, the scalability of the algorithms associated with the aforementioned LNSE operator is implied.

4.4 Results

4.4.1 Flow past a rotating cylinder

In this section, we validate our method by conducting the stability analysis of the flow past a rotating cylinder. The stability of this particular type of flow has been studied both experimentally (Barnes, 2000) and numerically (Stojković, Breuer, and Durst, 2002; Pralits, Giannetti, and Brandt, 2013). Here, we define a non-dimensional rotation rate as

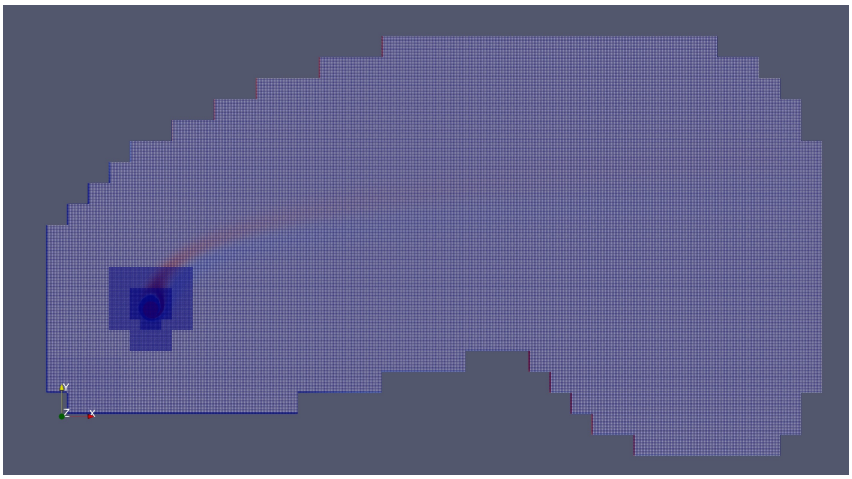
$$\alpha = \frac{\Omega D}{2U_\infty}, \tag{4.48}$$

and a non-dimensionalized span-wise wavenumber

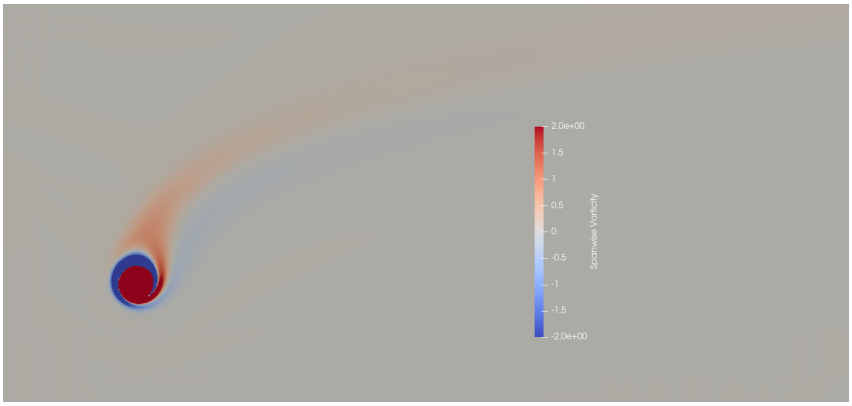
$$\lambda = \frac{2\pi k}{c}. \tag{4.49}$$

As shown by Pralits, Giannetti, and Brandt (2013), the rotational motion of the cylinder has a stabilizing effect. However, as the rotational rate further increases, unstable modes emerge again. To validate our solver, we choose two cases as detailed below.

We first apply our linear stability solver to the case of $Re = 40$, $\alpha = 5.5$. For these conditions, according to the study of Pralits, Giannetti, and Brandt (2013), there are no 2D unstable modes but only 3D unstable modes. In this case, we used the computational mesh shown in Figure 4.2a and the 2D fixed point base flow shown in Figure 4.2b. Using our 3D stability analysis solver, we find an unstable mode at $\lambda = 1.0$, and the corresponding unstable mode is shown in Figure 4.3.



(a) Mesh topology of the case at $Re = 40,\ \alpha = 5.5,\ \lambda = 1.0$

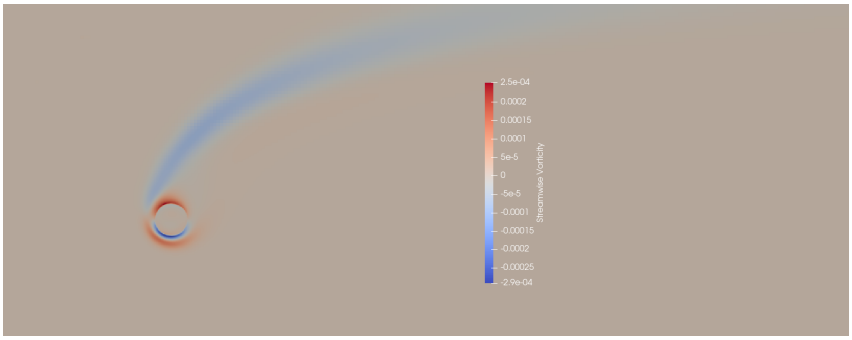


(b) Spanwise vorticity distribution of the fixed point at $Re = 40,\ \alpha = 5.5,\ \lambda = 1.0$

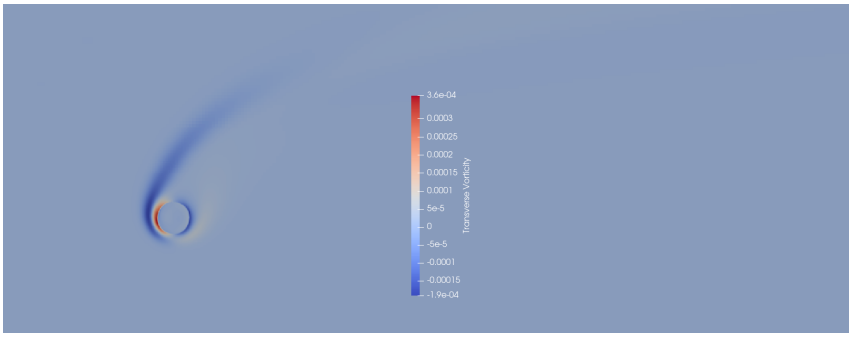
Figure 4.2: Fixed point and mesh topology used to conduct stability analysis for the case of $Re = 40,\ \alpha = 5.5,\ \lambda = 1.0$.

Then, we apply our algorithm to the case of $Re = 100,\ \alpha = 1.0$. As shown by Pralits, Giannetti, and Brandt (2013), although the most unstable mode appears at $\lambda = 0$, i.e. a 2D mode, there is a spectrum of less unstable 3D modes. The largest λ with an unstable 3D mode appears near $\lambda = 0.92$. We used the computational mesh shown in Figure 4.4a and the fixed point shown in Figure 4.4b to compute the marginally stable mode shown at $\lambda = 0.915$. The vorticity components of this marginally stable mode are shown in Figure 4.5.

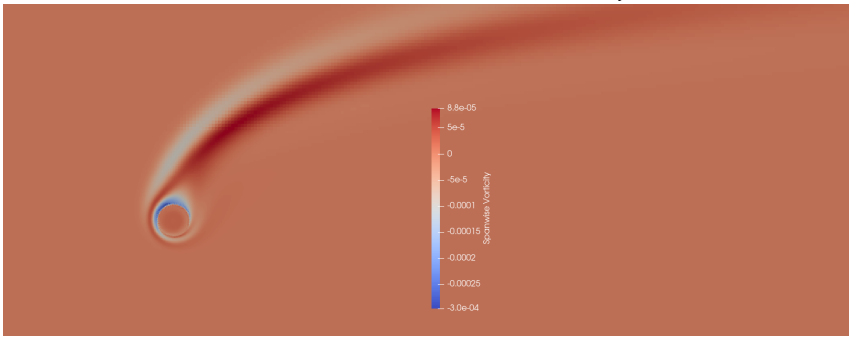
By conducting the stability analyses of the flow past a rotating cylinder at two different configurations, we demonstrated the validity of our formulation of LNSE and the O-FMM approximation on the boundary conditions.



(a) Streamwise vorticity distribution ω_x.

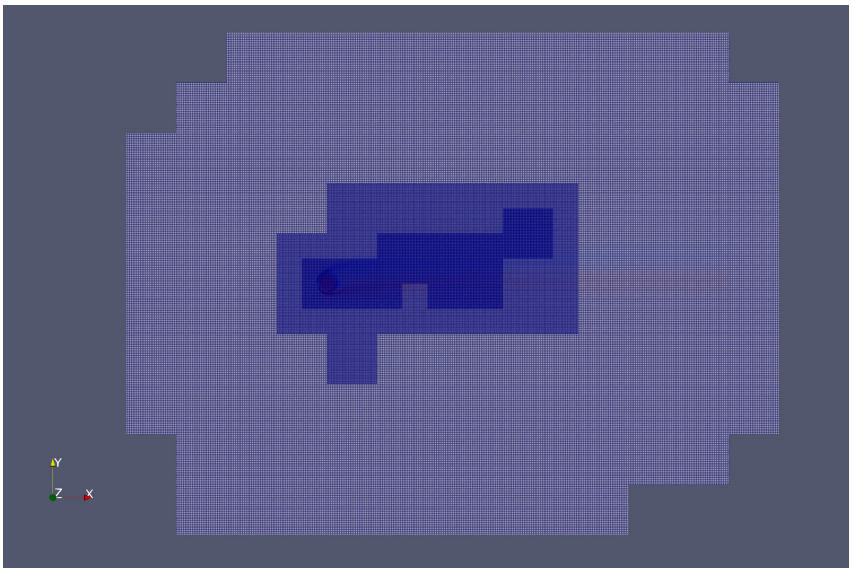


(b) Transverse vorticity distribution ω_y.

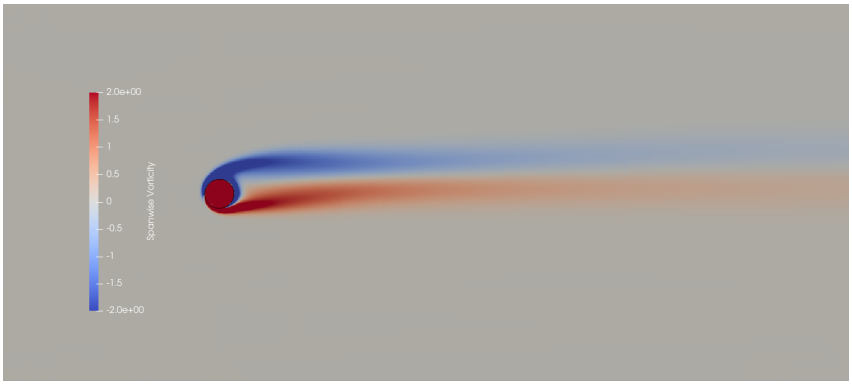


(c) Transverse vorticity distribution ω_z.

Figure 4.3: Vorticity distribution of the most unstable mode at $Re = 40$, $\alpha = 5.5$, $\lambda = 1.0$.



(a) Mesh topology of the case at $Re = 100,\ \alpha = 1.0,\ \lambda = 0.915$

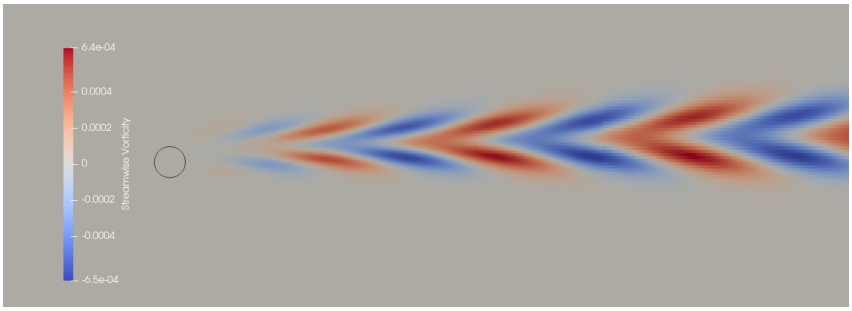


(b) Spanwise vorticity distribution of the fixed point at $Re = 100,\ \alpha = 1.0,\ \lambda = 0.915$

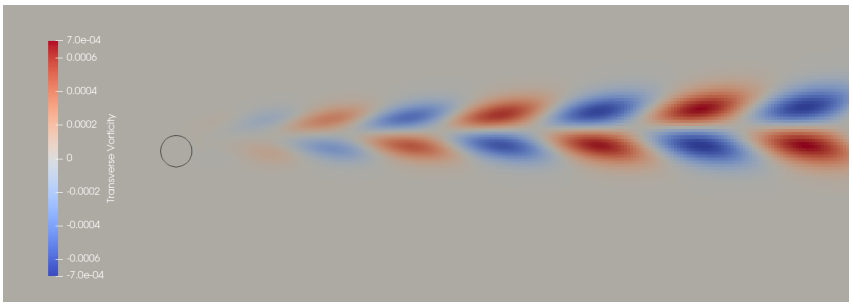
Figure 4.4: Fixed point and mesh topology used to conduct stability analysis for the case of $Re = 100,\ \alpha = 1.0,\ \lambda = 0.915$.

4.4.2 Resolvent analysis

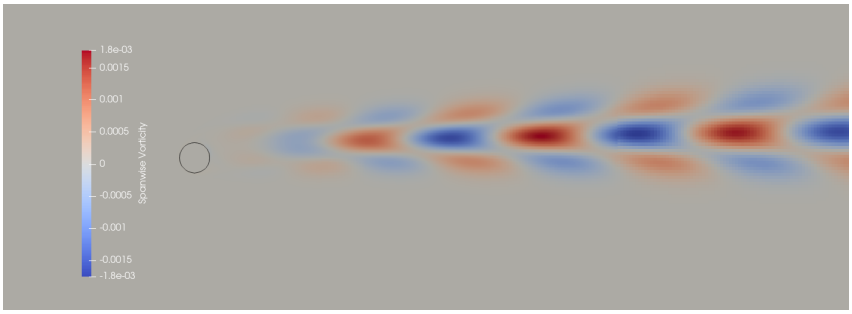
Even in linearly stable flows, disturbances can be amplified by transient growth. Such transient growth is due to the non-normality of the LNSE. In these scenarios, we can use resolvent analysis to identify the linear amplification mechanisms to deduce the instability mechanisms (Trefethen, Trefethen, et al., 1993). For unsteady flows, resolvent analysis around mean flows can be used to construct lower dimensional models that faithfully replicate the full flow dynamics (Symon, 2018). In this section, we demonstrate that the LNSE operator, induced from IBLGF-AMR and detailed in previous sections, can be used to conduct resolvent analysis. In particular, we estimated the resolvent norm and computed the most energetic resolvent mode for the flow past a cylinder at $Re = 100$.



(a) Streamwise vorticity distribution ω_x.



(b) Transverse vorticity distribution ω_y.

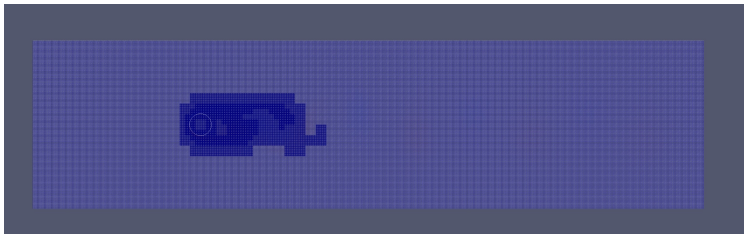


(c) Spanwise vorticity distribution ω_z.

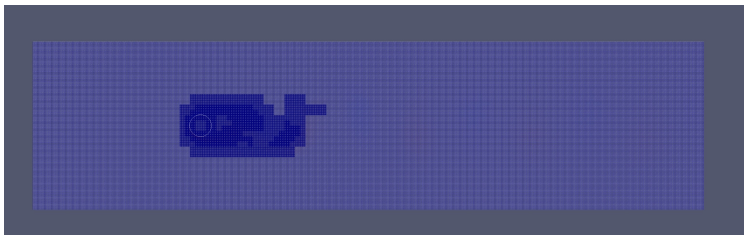
Figure 4.5: Vorticity distribution of the most unstable mode at $Re = 100$, $\alpha = 1.0$, $\lambda = 0.915$.

First, we conducted the same resolvent analysis calculation on two different meshes (shown in Figure 4.6) to demonstrate that the resolvent calculation is independent of mesh topology. The resulting resolvent norms are shown in Figure 4.7. The figure shows that two meshes produce identical resolvent norms across a range of frequencies ω, demonstrating that the resolvent analysis calculation is mesh independent.

Next, we simulated this particular flow for 375 convective time units ($375tU_\infty/c$) and used the flow data of the last 68 convective time units to construct a time average mean flow, i.e. $\bar{\boldsymbol{q}}$ in Eq. 4.6. We constructed the linearized Navier-Stokes operator around that mean flow ($\boldsymbol{A}(\bar{\boldsymbol{q}})$) and computed the resolvent norm over a range of frequencies from $\omega = 0.5$ to $\omega = 1.5$. The resulting resolvent norm is shown in



(a) Mesh 1 used to conduct resolvent analysis



(b) Mesh 2 used to conduct resolvent analysis

Figure 4.6: Two different meshes used to show the independence of mesh topology.

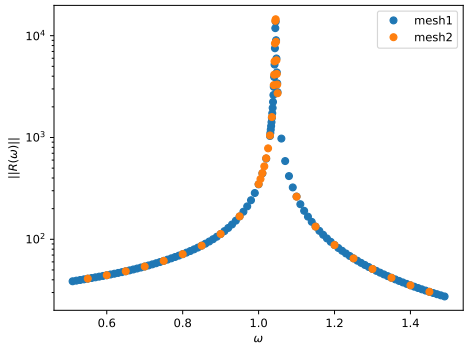


Figure 4.7: Comparison of the resolvent norm between two meshes.

Fig 4.8. In this plot, the resolvent norm peaks at $\omega = 1.0452$, which is consistent with the observations of a previous study on the same flow (Symon, 2018). In addition, we also display the velocity of the most energetic mode obtained from the resolvent analysis. The streamwise component of the forcing mode is shown in Figure 4.9, and the streamwise component of the response mode is shown in Figure 4.10. The mode shapes are consistent with the ones found by Jin et al. (Jin, Symon, and Illingworth, 2021).

4.4.3 Stability of the flow past a cylinder with a control cylinder in the wake

Strykowski and Sreenivasan (1990) showed that by placing a small cylinder (called the control cylinder) behind the main cylinder, vortex shedding can be suppressed. The stabilization associated with the control cylinder was explained by Giannetti

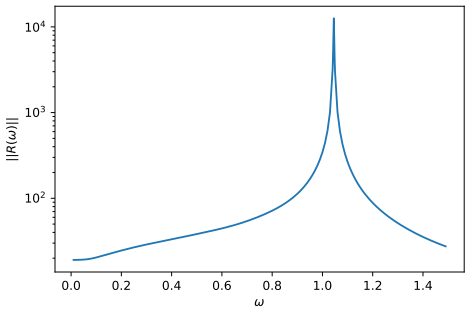


Figure 4.8: Resolvent norm $||R(\omega)||$ of flow past a circle at Re = 100 over a range of ω from 0.0 to 1.5

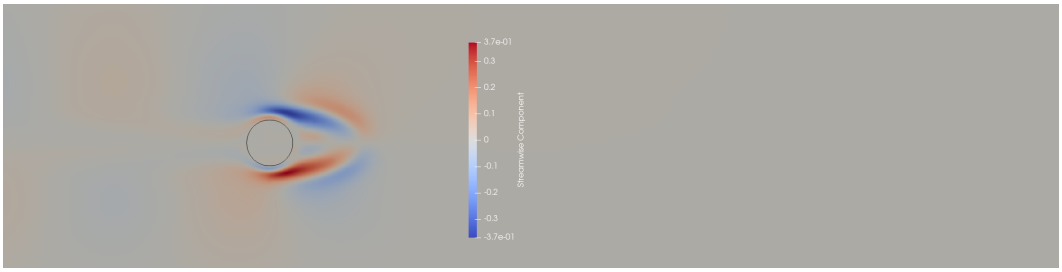


Figure 4.9: Streamwise component of optimal forcing mode at ω = 1.0452

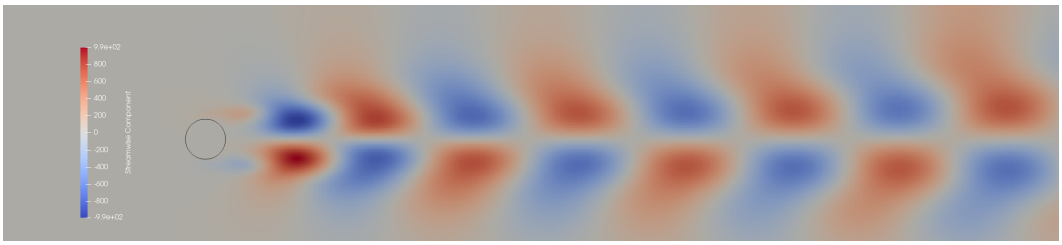


Figure 4.10: Streamwise component of optimal response mode at ω = 1.0452

and Luchini (2007) by introducing the concept of structural sensitivity, i.e. finding those points in the flow where the eigenvalues are most sensitive to perturbations. In our case, we compute the stability of the flow, including the control cylinder directly, as a numerical challenge for our solver.

The main computational challenge, in this case, is that there is a wide range of length scales (the main cylinder's diameter, D, and the control cylinder's diameter, d, differ by a factor of 10). An AMR grid resolving both cylinders is obtained by starting with a uniform resolution base mesh with $\Delta x_{base} = 0.04D$. The base mesh is then increasingly and adaptively refined (by factors of 2) based on flow simulations. We find good convergence with three levels of refinement, yielding the finest resolution as $\Delta x_{fine} = 0.005D$. The equilibrium solution and AMR grid for

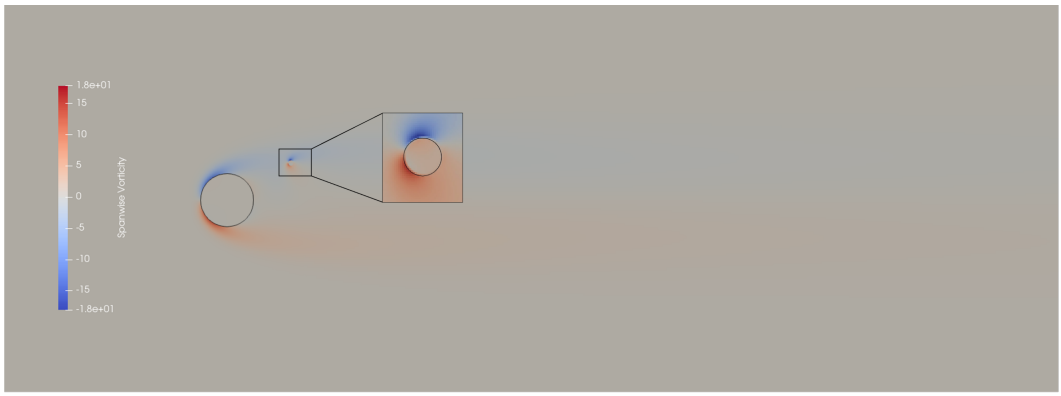


Figure 4.11: Base flow for flow past two cylinders used in the structural stability analysis. The main cylinder (large black circle) is centered around the origin, and the control cylinder (small black circle) is centered around $x_c/D = 1.2$, and $y_c/D = 0.7$ with $d/D = 0.1$. The flow around the control cylinder is magnified.

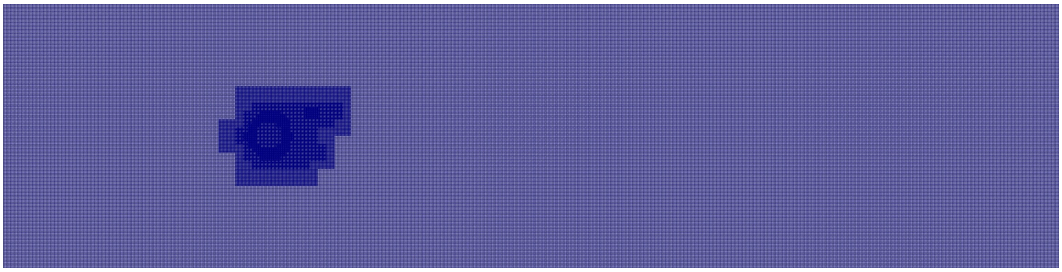


Figure 4.12: Computational mesh for the stability analysis of the flow past a cylinder with a control cylinder in the wake.

one of the configurations we study ($x_c/D = 1.2$, $y_c/D = 0.7$, $Re = 63$) are shown in Figs. 4.11 and 4.12, respectively.

The resulting matrix form of LNSE has dimension 3×10^6 by 3×10^6. We compute the equilibrium using Newton iteration and the eigenvalues using an Arnoldi-type method from SLEPc library (Hernandez, Roman, and Vidal, 2005) coupled with MUMPS solver (Amestoy et al., 2001). For each case, the entire process took around 5 hours using a total of 96 cores. We also give the example of an unstable mode found given $x_c/D = 1.2$, $y_c/D = 0.7$, and $Re = 63$ (Figure 4.13). The vortical structures downstream of the cylinders illustrate the importance of using LGF to create a compact domain to conduct the linear stability analysis.

We apply our linear stability solver to compute the eigenvalue spectra associated with a variety of control cylinder positions. Specifically, we fix $Re = 63$ and the x-coordinate of the center of the control cylinder at $x_c = 1.2D$ and vary the y-coordinate, y_c, from $0.5D$ to $1.0D$ with $0.1D$ increments. The resulting eigenvalue

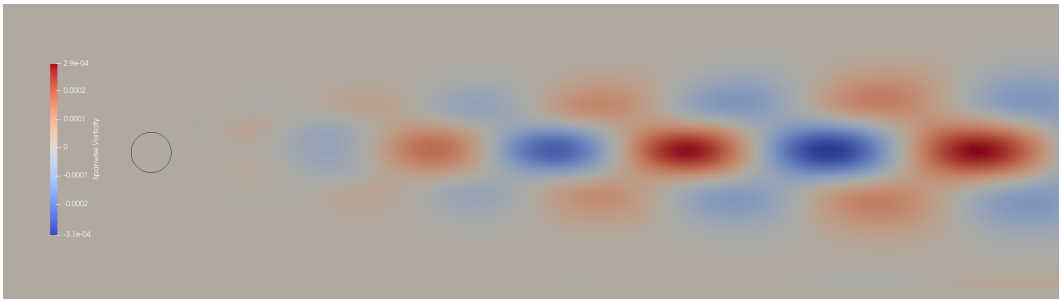


Figure 4.13: Unstable mode computed using the linear stability analysis.

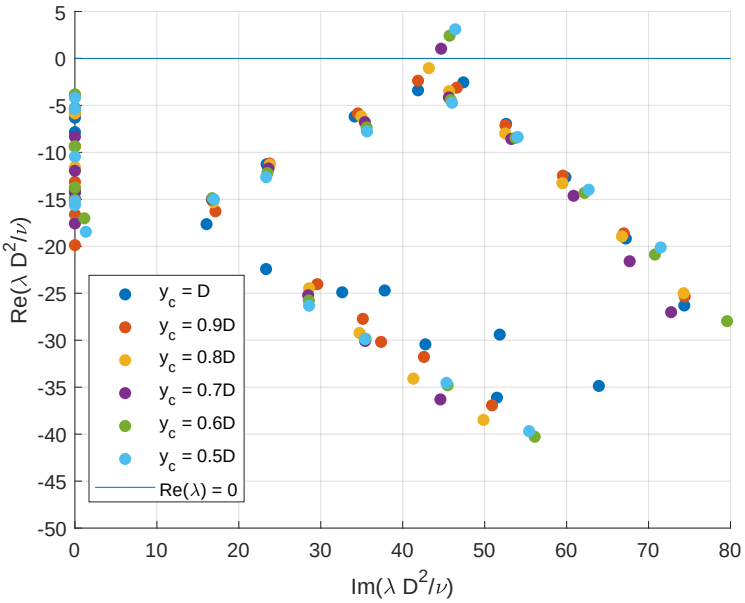


Figure 4.14: Distribution of eigenvalues of LNSE for various control cylinder locations at $x_c = 1.2D$ and $y_c \in \{0.5D, 0.6D, 0.7D, 0.8D, 0.9D, D\}$. Because the LNSE is a purely real system in this case and all eigenvalues will appear in complex conjugate pairs, we only show the eigenvalues with positive imaginary parts.

spectra are shown in Figure 4.14. In this figure, the vertical axis is the real part of the eigenvalues, indicating how fast a particular perturbation (i.e. a particular eigenvector) grows or decays. A positive real part indicates a growing perturbation, thus an unstable equilibrium. If, on the contrary, all eigenvalues have negative real parts for a particular equilibrium, all the perturbations will decay, and that equilibrium is stable. From the eigenvalue spectra, we see that for $y_c \in \{0.5D, 0.6D, 0.7D\}$, eigenvalues with positive real parts exist, and the equilibria in those cases are not stable, thus resulting in vortex shedding. In contrast, at $y_c \in \{0.8D, 0.9D, D\}$, all eigenvalues have negative real parts, indicating stable equilibria, and thus, the vortex shedding is suppressed in those cases. The stability properties of those flows are corroborated by interpolating the results from Strykowski and Sreenivasan (1990).

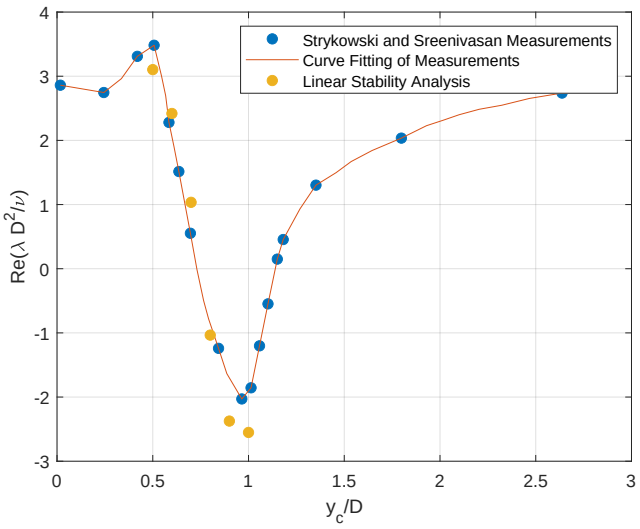


Figure 4.15: Comparison of flow perturbation growth rates between estimates from linear stability analysis and those from experimental data (Strykowski and Sreenivasan, 1990) for $Re = 63$, fixed streamwise location of the control cylinder $x_c/D = 1.2$, and varying horizontal location of the control cylinder. The growth rates obtained from linear stability analysis agree with the experimentally measured values.

According to their experimental measurements, at $Re = 63$ with the control cylinder positioned at $x_c = 1.2D$, $y_c = 0.75D$, the flow is marginally stable. Moving the control cylinder up will stabilize the flow and suppress the vortex shedding while moving the control cylinder down will have the opposite effect. This dichotomy of the stabilizing and destabilizing effects from the horizontal locations of the control cylinder confirms the results of our linear stability analysis.

For a specific equilibrium, the growth rate of flow perturbations can be estimated by the largest real part among all the real parts of eigenvalues associated with that equilibrium. To compare our growth rate estimates with the experimental results of Strykowski and Sreenivasan (1990), we non-dimensionalize all the eigenvalues with D^2/ν and overlay the estimates of flow perturbation growth rates obtained from our linear stability analysis to those obtained from the experimental measurements (Strykowski and Sreenivasan, 1990) in Figure 4.15. This figure shows that the estimated flow perturbation growth rates from the stability analyses agree with those obtained from experimental measurements. Thus, the eigenvalues from linear stability analysis accurately estimate the flow perturbation growth rates.

By investigating the relationship between the position of the control cylinder and the onset of vortex shedding, this specific computational study highlights the efficiency

of the AMR components in this algorithm when analyzing flows with a wide range of scales and also the necessity of using LGF to obtain a compact computational domain while maintaining accurate boundary conditions.

4.5 Conclusions

In this chapter, we introduced a fast algorithm for stability and resolvent analysis by combining LGF, AMR, and FMM. This method conducts flow analysis by explicitly constructing the spatially discretized linearized Navier-Stokes operator as a sparse matrix. Using LGF, we were able to conduct such flow analysis on an infinite domain while using a finite computational mesh; using FMM, we sparsified the dense block of the LGF operator; using AMR, we can resolve all relevant scales while maintaining computational efficiency.

To verify our algorithm, we investigated the stability of the flow past a rotating cylinder. We were able to replicate results from the literature, namely one marginally stable mode and one unstable mode in different flow configurations. We also presented 2D resolvent analysis results for flow past a circle at Re = 100. The corresponding resolvent norm, optimal input mode, and optimal response mode are consistent with previous studies (Symon, 2018; Jin, Symon, and Illingworth, 2021). Furthermore, we examined the suppression of vortex shedding on a cylinder via the placement of a small control cylinder in its wake. We accurately resolved all length scales posed by the problem and computed the growth rates that matched the experimental measurements (Strykowski and Sreenivasan, 1990). The successful application of IBLGF-AMR and O-FMM to stability and resolvent analysis yields an efficient and scalable algorithm for flow analysis and opens up the possibility for future work with more complex flows.

Chapter 5

THE STARTING VORTICES GENERATED BY BODIES WITH SHARP AND STRAIGHT EDGES IN A VISCOUS FLUID

This chapter is adapted from Sader et al. (2024). This is the first application of the algorithm presented in Chapter 3. In this chapter, we use the numerical simulations to validate the starting vortex theorem proposed by Pullin and Sader (2021).

5.1 Introduction

The flow generated by a thin airfoil that suddenly moves at finite angle-of-attack has been widely studied over the last century (Prandtl, 1924; Wagner, 1924; Kaden, 1931; Anton, 1939; Anton, 1956; Rott, 1956; Wedemeyer, 1961; Blendermann, 1967; Pullin, 1978). A key characteristic of this flow is the generation of a 'starting vortex' that is localized to the sharp trailing edge of the airfoil, immediately after the plate starts to move. Proximity to the trailing edge causes the starting vortex to exhibit self-similar behavior in time. This property has been utilized in its calculation, based on inviscid theory, leading to the widely recognized rolled-up vortex that moves approximately normal to the airfoil for small time (Kaden, 1931; Anton, 1956; Pullin, 1978). This starting vortex has been confirmed using direct numerical simulations (DNS) of the Navier-Stokes equations (Koumoutsakos and Shiels, 1996; Krasny, 1991; Luchini and Tognaccini, 2002; Jones, 2003; Eldredge, 2007; Michelin and Llewellyn Smith, 2009; Nitsche and Xu, 2014; Xu and Nitsche, 2015; Luchini and Tognaccini, 2017; Xu, Nitsche, and Krasny, 2017) and physical experiments (Pierce, 1961; Pullin and Perry, 1980; Auerbach, 1987).

Pullin and Sader (2021), henceforth denoted PS21, recently calculated the starting vortex generated at the trailing edge of a flat plate that suddenly, and simultaneously, translates and rotates; see Figure 5.1(a). This was achieved using an inviscid vortex sheet formulation that invokes the Kutta condition at its trailing edge. The motion of the flat plate was specified by two independent power-laws in time, m and p, such that,

$$U(T) = U_0 T^m, \qquad \Omega(T) = \Omega_0 T^p, \tag{5.1}$$

where $U(T)$ and $\Omega(T)$ are the translational and angular velocities of the plate, as shown in Figure 5.1(a), U_0 and Ω_0 are the (constant and dimensional) translational

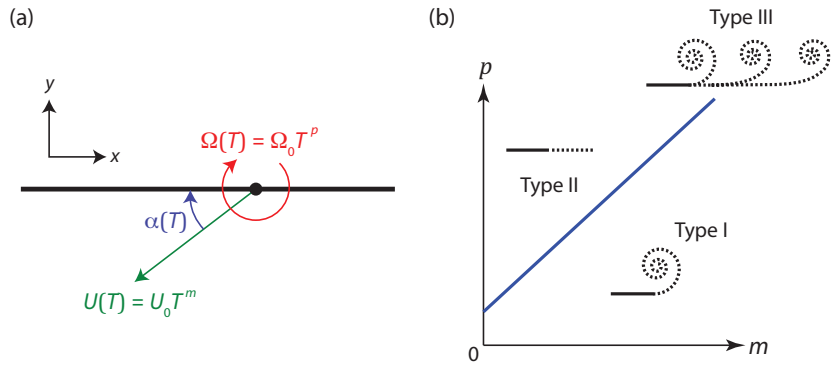


Figure 5.1: Starting vortex generated by a flat plate that moves suddenly. (a) Schematic showing the flat plate with its translational and angular velocities. The x and y-components of the Cartesian frame are always parallel and perpendicular to the plate, respectively; the origin is at the plate center (this differs from PS21). (b) Phase plane for the starting vortices generated at the trailing (right) edge of the plate for zero initial angle-of-attack, $\alpha_0 = 0$, where m and p are the translational and rotational power-laws in (5.1), respectively. Plate rotation is away from the three-quarter-chord position, i.e., $d \neq 1/2$. The critical line (solid and diagonal blue line) is (5.6). Nominal shapes of Type I, II and III vortices are illustrated (dashed lines). Type I and II vortices are independent of $\beta \equiv \Omega_0 a/U_0$, defined in (5.4), whereas Type-III vortices are swept further downstream with decreasing β.

and angular velocity scales, respectively, and the dimensionless convective time is

$$T \equiv \frac{tU_0}{a}. \tag{5.2}$$

Here, t is the dimensional time, and a is the half-chord of the plate. The rotational pivot position lies in the plane of the plate and is denoted by

$$d \equiv \frac{x_0}{a}, \tag{5.3}$$

where x_0 is its dimensional x-coordinate whose origin is at the plate center. Thus, $d = 0, -1/2$, and $1/2$ correspond to rotation about the plate center, quarter-chord, and three-quarter-chord positions, respectively. Importantly, the starting vortex is derived in the small convective time limit, i.e., $T \to 0^+$. The finite-time interval over which it holds is addressed in this study.

In addition to the rolled-up vortex sheet whose center moves approximately normal to the plate, designated a Type-I vortex by PS21, this recent study showed that two additional vortex sheet types can arise at the trailing edge. The precise vortex type depends on the relative values of the power laws, m and p. The first additional type was termed a Type-II vortex sheet. It does not roll up, i.e., there is no vortex center,

but moves strictly parallel to the plate while being convected downstream. That is, the vortex sheet rotates synchronously with the plate; convection off the plate surface dominates vortex self-induction. The second additional type was termed a Type-III vortex and combines the features of Type I and II vortices. It exhibits the competing effects of convection and vortex-sheet roll up, yet unlike Types I and II, its shape depends on the relative magnitude of the angular and translational speeds through the dimensionless 'rotation parameter',

$$\beta \equiv \frac{\Omega_0 a}{U_0}. \tag{5.4}$$

The shape of a Type-III vortex approaches that of Type II and I vortices as $\beta \to 0$ and ∞, respectively, i.e., when the respective plate motion is translation and rotation dominated. The angle-of-attack of the plate can be expressed in terms of β,

$$\alpha(T) = \alpha_0 + \frac{\beta}{1+p} T^{1+p}, \tag{5.5}$$

where α_0 is the initial angle of attack. Figure 5.1(b) summarizes the key finding of PS21 for the trailing-edge vortex of a flat plate with zero initial angle-of-attack, i.e., $\alpha_0 = 0$. This shows that the vortex shape changes discontinuously as a critical line in the (m, p)-phase plane is transversed. Type-III vortices constitute singular solutions in the (m, p)-phase plane, that occur on the critical line,

$$m = \frac{1}{3}(2p - 1), \tag{5.6}$$

provided plate rotation is away from the three-quarter-chord position. For rotation at this singular position, the critical line is $m = 1 + 2p$.

Hinton et al. (2024) generalized the study of PS21, enabling calculation of the starting vortex at any 'sharp and straight edge' of an arbitrary solid body. A 'sharp and straight edge' is one whose neighboring upper and lower surfaces are tangent to one another at the edge, e.g., a flat plate has two such edges. Several findings were reported:

1. The three vortex sheet types found for the trailing edge of a flat plate in PS21, apply to the sharp and straight edges of an arbitrary body.

2. The phase diagram for the leading-edge vortex of a flat plate is identical to Figure 5.1(b), except that (1) Type-II vortices do not exist, i.e., no vortex is shed, and (2) the critical line is $m = 1 + 2p$ for rotation about the plate's quarter-chord position.

3. Translation of the sharp and straight edge of an arbitrary body, parallel and normal to the neighboring surfaces of this edge, is the physical mechanism that drives the starting vortex. For example, rotation of a flat plate about a pivot position is a motion that generates the requisite edge translation.

4. Rotation of a flat plate can produce starting vortices of different types at its leading and trailing edges (provided rotation occurs at either its quarter or three-quarter-chord positions). This is not possible with pure plate translation. This highlights a distinction between global motion of the solid body and local translation of the edge under consideration.

5. The implication of point (ii), sub-point (1), is that the leading-edge vortex of a flat plate can be naturally suppressed through control of the plate's dynamics. The relevance of this finding to low-speed aircraft that predominantly use thin airfoils of low curvature, was discussed.

6. Explicit formulas for the lift experienced by a flat plate that is undergoing simultaneous translation and rotation were derived. This required a new treatment of the Bernoulli equation in a rotating frame.

7. The general theory was applied to Joukowski airfoils, highlighting its applicability to bodies of arbitrary shape.

Here, we examine the existence of Type I, II, and III starting vortices in a viscous fluid, for (1) a flat plate that is simultaneously translating and rotating, and (2) two symmetric Joukowski airfoils that are translating in two orthogonal directions. This is achieved using high-fidelity DNS of the Navier-Stokes equations in two spatial dimensions, employing a lattice Green's function (LGF) method that discretizes the flow in the regions of finite vorticity only. This approach enables access to start-up flows at high Reynolds number, i.e.,

$$\mathrm{Re} \equiv \frac{cU_0}{\nu} \gg 1, \tag{5.7}$$

where the viscous boundary layer and starting vortex are only resolved in the immediate vicinity of the solid body; ν is the kinematic viscosity and c is the chord of the body. This study primarily focuses on the flat plate where a comprehensive analysis of vortex formation is reported. Of particular interest is the change in shape of the starting vortices as the power laws of the plate motion, m and p, are varied. We include a brief section on Joukowski airfoils to illustrate the utility of the theory

reported by Hinton et al. (2024) in predicting the starting vortices of bodies with more complicated shapes.

The chapter is organized as follows. We begin in Section 5.2 by outlining the LGF/immersed boundary methodology used to numerically solve the Navier-Stokes equations for a flat plate and the two Joukowski airfoils. A comprehensive analysis of the vortices formed at the edges of a translating and rotating flat plate then follows. This includes analytical formulas of the maximal time for the existence of its starting vortices, which are derived in Section 5.3. These formulas are based on the inviscid theory, which was first reported for the trailing edge of a flat plate in PS21 and later extended to an arbitrary body with any number of edges by Hinton et al. (2024). This collective theory is henceforth termed the 'starting vortex inviscid theory', which is abbreviated to 'SVT'. In Section 5.4, a detailed comparison of SVT and DNS for the leading and trailing-edge vortices of a flat plate is reported. This employs the above-mentioned maximal time for the existence of a starting vortex. This comparison confirms the theoretical prediction of Hinton et al. (2024) that the leading-edge vortex of a flat plate can be suppressed dynamically. Finally, the vortices formed at the trailing edges of two symmetric Joukowski airfoils are briefly examined in Section 5.5. This illustrates the utility of SVT in predicting the starting vortices of bodies with more complex shapes. We conclude in Section 5.6 by summarizing the key findings of this study with details of the calculations relegated to Appendix D.

5.2 Numerical method for solving the Navier-Stokes equations

The starting vortices generated by a flat plate and two Joukowski airfoils, immersed in an incompressible and unbounded viscous fluid, are calculated using DNS of the two-dimensional Navier-Stokes equations,

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t}+\mathbf{u}\cdot\nabla\mathbf{u}\right)=-\nabla P+\mu\nabla^2\mathbf{u},\qquad \nabla\cdot\mathbf{u}=0, \tag{5.8}$$

where ρ and μ are the fluid density and shear viscosity, respectively, $\mathbf{u}$ is the velocity field, and P is pressure. Eq. (5.8) is solved using an immersed boundary method, combining (i) an LGF approach that tracks regions of finite vorticity, and (ii) a uniform mesh with spatial computational domain adaptation. The immersed boundary method enforces no-slip at the body's surface, and the domain adaptation is performed at each time step. Simulations, and hence discretizations, are performed in the non-inertial frame of the moving body. The details of this algorithm can be found in Chapter 3. This combined numerical method enables accurate simulation of external and unbounded incompressible flows, by requiring discretizations in the

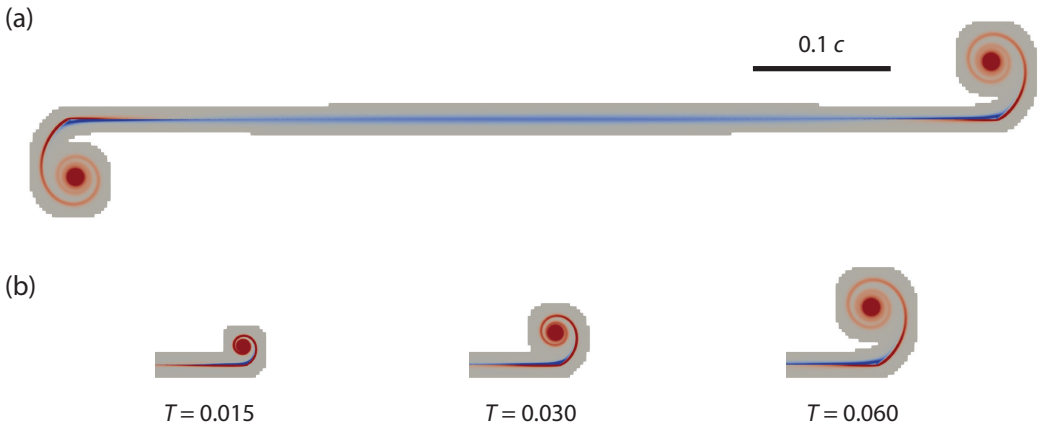


Figure 5.2: Sample computational mesh (grey region) of the DNS for a flat plate showing vorticity distribution (red and blue colors); $m = 2$, $p = 0$, $d = 0$ (rotation about plate center), $\beta = 3.175$, $\mathrm{Re} \equiv cU_0/\nu = 5{,}040$, where ν and c are the kinematic viscosity and plate chord, respectively, with uniform spatial discretization $\Delta x = 2.5 \times 10^{-4} c$. (a) Snapshot of the entire computational domain for $T = 0.060$. (b) Time evolution of the mesh near the trailing edge of the plate. The computational mesh spatially adapts to discretize regions of finite vorticity only. The scale bar applies to all plots. Further details are provided in Section 5.2.

regions of finite vorticity only. This involves a user-chosen vorticity threshold which is varied until convergence is achieved, as discussed below.

An example of the computational mesh for a flat plate and its evolution in time is given in Figure 5.2, showing it is confined to the immediate vicinity of the plate. The flow in the undiscretized region can be computed analytically using results from the discretized region, if needed. This is not required here, because we are only interested in regions of finite vorticity. This numerical approach results in a dramatic improvement in computational efficiency relative to other methods that require complete discretization of the flow domain. Consequently, the present methodology is ideally suited to start-up problems, where vorticity generation occurs at the solid surfaces.

We focus on assessing the validity of SVT. Thus, the simulation parameters are chosen using the following procedure. First, we compute the flow for a relatively low Reynolds number, $\mathrm{Re} \approx 1{,}000$. The Reynolds number is then systematically increased until viscous diffusion is small enough to study the vortical structures, given available computational resources. Spatial discretization is simultaneously increased to minimize numerical diffusion and ensure negligible numerical artifacts. The time step-size is chosen to maintain linear stability of the time marching. The

chosen vorticity cut-off threshold for meshing is 0.001% of the maximum vorticity; this is varied to ensure convergence. This threshold value is in accord with Liska and Colonius (2017) that showed it is sufficiently small for accurate flow simulation.

For further details of the numerical algorithm used in this study, the reader is referred to Liska and Colonius (2017), Yu, Dorschner, and Colonius (2022), and Hou and Colonius (2024a).

5.3 Maximal time for the existence of the starting vortices generated by a flat plate

Next, we use SVT to calculate the maximal time that vortices conforming to the Type I, II, and III description can exist at the trailing and leading edges of a flat plate. An initial angle-of-attack of $\alpha_0 = 0$, and plate rotation away from its quarter-chord and three-quarter-chord positions, i.e., $d \neq \pm 1/2$, are assumed. A similar analysis can be performed for $\alpha_0 \neq 0$ and $d = \pm 1/2$ (not shown).

We first analyze the vortex sheet at the trailing edge, which is given by

$$x_v + i y_v = a\, T^q Z(\lambda), \tag{5.9}$$

where x_v and y_v are (dimensional) Cartesian coordinates referenced to the trailing edge, q is the vortex power-law defined in Table 1 of PS21, i is the imaginary unit, with the governing equation for the self-similar vortex shape, $Z(\lambda)$, being

$$T^{q-1}\left(q\bar{Z} + \left(\frac{q}{2} + p\right)(1-\lambda)\frac{d\bar{Z}}{d\lambda}\right) = T^m - i\hat{\beta}\, T^{p-\frac{q}{2}} \frac{1 + M(Z)}{(2Z)^{\frac{1}{2}}}, \tag{5.10}$$

where

$$\hat{\beta} \equiv \beta\left(\frac{1}{2} - d\right), \qquad M(Z) \equiv \frac{\displaystyle\int_0^1 \left(\frac{1}{Z^{\frac{1}{2}} - Z'^{\frac{1}{2}}} - \frac{1}{Z^{\frac{1}{2}} + \bar{Z}'^{\frac{1}{2}}}\right) \mathrm{d}\lambda'}{\displaystyle\int_0^1 \left(\frac{1}{Z'^{\frac{1}{2}}} + \frac{1}{\bar{Z}'^{\frac{1}{2}}}\right) \mathrm{d}\lambda'}, \tag{5.11}$$

and $Z' \equiv Z(\lambda')$; all other symbols are identical to those in PS21. Eq. (5.10) is obtained by substituting (3.1) into the Birkhoff-Rott equation that defines the vortex sheet emanating from the sharp edge that is required to enforce the Kutta condition. It is obtained from equations (4.1), (4.4), (5.1), (5.2), and (5.4) of PS21, where λ is the similarity variable that varies monotonically from 0 to 1 along the vortex sheet. Type I, II, and III vortices arise when the left-hand side of (5.10) balances the second, first, and both terms on its right-hand side, respectively, as $T \to 0^+$.

In arriving at (5.10), a small-time expansion of the trigonometric functions in Eq. (4.4) of PS21 is performed, which is valid provided

$$\frac{\beta}{1+p}T^{1+p} \ll 1. \tag{5.12}$$

Specifying a multiplicative factor, $0 < \delta \ll 1$, for (5.12) to hold, gives the maximal time,

$$T_{\max}^{(1)} \approx \left(\frac{\delta\left[\frac{1}{2}-d\right]}{\hat{\beta}}\right)^{\frac{1}{1+p}}, \tag{5.13}$$

for which (5.10) is valid; where we have used the approximation, $(1+p)^{1/(1+p)} \approx 1$ for all $p \geq 0$. Because Type-III vortices arise when both terms in the right-hand side of (5.10) balance, (5.13) sets the maximal time for which this vortex type exists; the constraint in (5.13) also applies to all vortex types. This is provided the vortex remains close to the plate's sharp and straight trailing edge, which is an overriding assumption of the theory; this spatial condition applies to any solid body with sharp and straight edges. Note that all starting vortices obey the self-similar form described by (5.9).

Type I and II vortices arise when one term on the right-hand side of (5.10) dominates the other, as discussed above. This provides an additional constraint on the maximal time for which Type I and II vortices exist (derived in Appendix D),

$$T_{\max}^{(2)} \approx \left(\hat{\beta}^{\frac{2}{3}\mathrm{sgn}(\epsilon)}\delta\right)^{\frac{1}{|\epsilon|}}, \tag{5.14}$$

where

$$\epsilon \equiv m - \frac{2p-1}{3}, \tag{5.15}$$

with Type I and II vortices corresponding to $\epsilon \neq 0$. Combining (5.13) and (5.14) gives the overall maximal time for a starting-vortex type to exist at the trailing edge of a flat plate,

$$T_{\max} \approx \begin{cases} \min\left\{\left(\frac{\delta\left[\frac{1}{2}-d\right]}{\hat{\beta}}\right)^{\frac{1}{1+p}}, \left(\hat{\beta}^{\frac{2}{3}\mathrm{sgn}(\epsilon)}\delta\right)^{\frac{1}{|\epsilon|}}\right\}, & \text{Type I and II vortices} \\ \left(\frac{\delta\left[\frac{1}{2}-d\right]}{\hat{\beta}}\right)^{\frac{1}{1+p}}, & \text{Type-III vortices} \end{cases}, \tag{5.16}$$

provided the vortex remains spatially close to the edge. The corresponding expression for the leading-edge vortex is obtained by replacing $1/2-d \longrightarrow 1/2+d$ in (5.16) and in the definition of $\hat{\beta}$ in (5.11). This is because their governing equations are of identical form (Hinton et al., 2024).

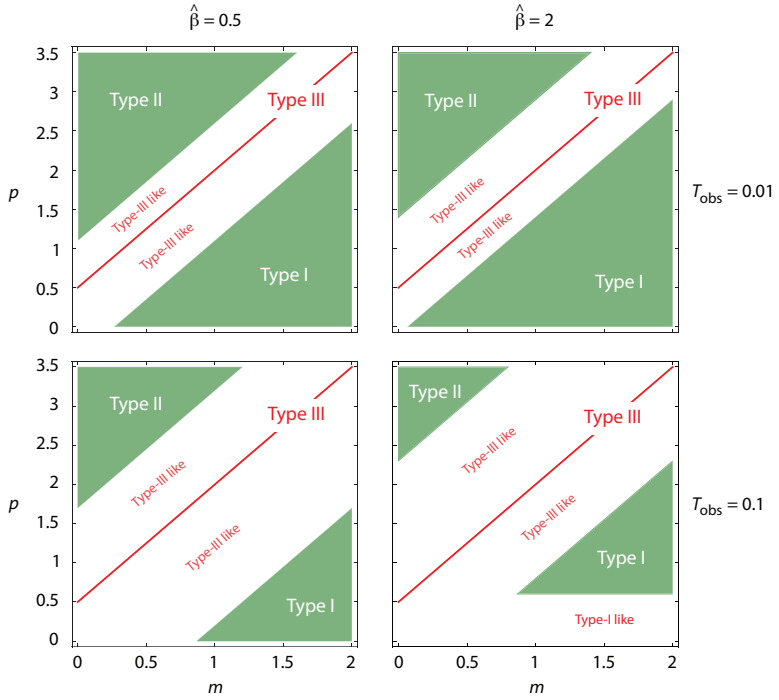


Figure 5.3: Regions of the (m, p)-phase plane where Type I, II and III vortices of a flat plate exist for $0 < T \leq T_{\text{obs}}$, using two choices of $T_{\text{obs}} = 0.01$ (first row) and 0.1 (second row); left column ($\hat{\beta} = 0.5$) and right column ($\hat{\beta} = 2$). The shaded (green) regions correspond to parameter values, (m, p), where the (small time) starting vortices—obeying the self-similar form described by (5.9)—are expected to hold at the given observation time, T_{obs}. Results given for rotation about the plate center, $d = 0$, and a nominal multiplicative factor of $\delta = 0.1(\ll 1)$ in (5.16). Type-I like and Type-III like vortices are not self-similar, but have shapes resembling those of Type I and III vortices, respectively.

To illustrate the utility of (5.16), we use it to determine regions of the (m, p)-phase plane where each vortex type exists, over an observation time, $0 < T \leq T_{\text{obs}}$. This requires $T_{\text{obs}} \leq T_{\text{max}}$, where T_{max} is defined in (5.16). Figure 5.3 gives these regions for plate rotation about its center, $d = 0$, observation times of $T_{\text{obs}} = 0.01$ and 0.1, and rotation parameters of $\hat{\beta} = 0.5$ and 2, using a nominal value of $\delta = 0.1(\ll 1)$. The diagonal (green) boundaries—within which Type I and II vortices exist (specified by (5.14))—approach the critical (red) line, $\epsilon = 0$, as T_{obs} is reduced. Thus, Figure 5.3 shows that in any finite-time simulation or physical measurement, Type I and II vortices can only be observed away from this critical line, $\epsilon = 0$. The horizontal (green) boundary in the bottom right-hand plot of Figure 5.3 is due to (5.13), and specifies the limit where the governing equation, (5.10), applies. Type-III vortices always occur on the critical (red) line, $\epsilon = 0$, for the chosen values of T_{obs}; increasing T_{obs} further can and does eliminate this

finding. Type-III like (non-similarity) vortices exist in the (white) regions between the diagonal green boundaries, because the first and second terms in the right-hand side of (5.10) are comparable in those regions. Type-I like (non-similarity) vortices exist in the white region below the horizontal green boundary.

These findings and (5.16) are used in the next section to guide a quantitative comparison of SVT with the DNS.

5.4 Direct numerical simulations of a flat plate

High-fidelity DNS are performed to resolve the starting vortices generated at the edges of a flat plate at their required small times, i.e., $T \lesssim T_{\max}$, where possible. Results are reported for Reynolds numbers ranging from Re = 5,040 to 45,255. The maximal values of the Reynolds numbers are limited by available computational resources, and are used in the comparison with SVT. We focus on a zero initial angle-of-attack, $\alpha_0 = 0$, for which SVT predicts that all three vortex types can exist. The majority of results are reported for rotation about the plate center, i.e., $d = 0$. Two sets of results are reported for rotation about the quarter-chord and three-quarter-chord positions of the plate, i.e., $d = -1/2$ and $+1/2$, respectively. The latter cases explore the theoretical prediction of SVT that vortices of different types can occur at the leading and trailing edges (Hinton et al., 2024).

All plots are presented in the frame of the flat plate. In all cases, vortices from SVT are plotted in their similarity scale, i.e., they do not appear to change size with increasing time. Instead, the corresponding spatial scale bar, referenced to the plate chord, c, varies with time and is reported in each figure. Table 5.1 summarizes the plate and DNS parameters used.

5.4.1 Type-I vortices

We first explore two cases for which SVT predicts Type-I vortices.

5.4.1.1 $m = 2$, $p = 0$, $d = 0$

Figure 5.4 provides a comparison of the DNS with the predictions of SVT, for the time power-laws, $m = 2$ and $p = 0$, and rotation about the plate center, $d = 0$. This corresponds to the plate undergoing a smooth accelerating translation from rest, with its rotational velocity starting abruptly at $T = 0^+$. SVT predicts that the time evolution of Type-I vortices depends explicitly on the rescaled time, $\hat{\beta}\,T$—rather

Vortex type	m	p	d	Re	$\Delta x/c$
I	2	0	0	5,040	2.5×10^{-4}
I	0	0	0	8,000	2.5×10^{-4}
II	0	2	0	32,000	1.25×10^{-4}
III	0	0.5	0	32,000	1.25×10^{-4}
III	1	2	0	45,255	1.25×10^{-4}
III-II	1	2	0.5	45,255	1.25×10^{-4}
None-III	1	2	−0.5	45,255	1.25×10^{-4}

Table 5.1: DNS parameters used for the flat plate. Starting-vortex type, as predicted by SVT, is in the first column. For $d = \pm 0.5$, vortices generated at the leading edge (LE) and trailing edge (TE) are specified using the format: LE-TE. Reynolds number, Re, and spatial discretization, $\Delta x/c$, used in each simulation are listed. The two symmetric Joukowski airfoils employ $\Delta x/c = 2.5 \times 10^{-4}$ and Re = 8,000.

than T alone—which from (5.2) and (5.4) gives

$$\hat{\beta}\, T = \Omega_0 \left(\tfrac{1}{2} - d \right) t. \tag{5.17}$$

That is, the evolution of Type-I starting vortices depends only on the plate rotation (not its translation). This feature is evident in Figure 5.4, because (1) the vortices predicted by SVT are precisely antisymmetric (by construction), and (2) the level of agreement between the SVT and DNS is identical at the leading and trailing edges. This finding is expected for rotation-dominated motion about the plate center.

For the smallest rescaled time of $\hat{\beta}\, T = 0.012$—see top row of Figure 5.4—agreement between the SVT and DNS is observed. This is despite the DNS vortices diffusing to a greater extent with decreasing $\hat{\beta}$ and increasing time, T, as expected. However, the level of agreement decreases with increasing $\hat{\beta}\, T$; see middle and bottom rows of Figure 5.4. This is due to deterioration of the approximation in (5.12) that underlies the governing equation in (5.10). Indeed, for $m = 2$, $p = 0$ with $\delta = 0.1$ (as chosen in Section 5.3), (5.16) gives $\hat{\beta}\, T_{\max} = \hat{\beta}\, T^{(1)}_{\max} \approx 0.05$, which is similar to $\hat{\beta}\, T = 0.048$ in the bottom row of Figure 5.4. This establishes that SVT is near the limit of its

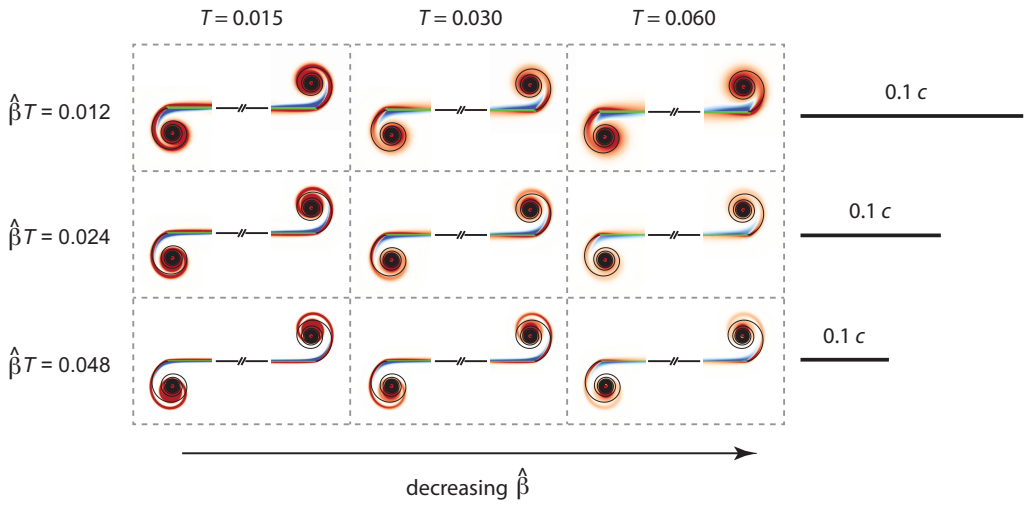


Figure 5.4: Flat plate. Type-I vortex: $m = 2$, $p = 0$, and $d = 0$ for Re = 5,040. Comparison of DNS (colored vorticity plots) with SVT (solid black lines), showing leading (left) and trailing (right) edges. The leading and trailing edges are at the ends of the green horizontal lines. The central section of the plate is not shown. Red and blue colors denote DNS vorticity regions of opposite sign. The color thresholds to white when the vorticity magnitude is less than 1% of the maximum value; this applies to all figures in this study. Scale bars of one-tenth of a chord length, i.e., $0.1c$, are given.

		$T = 0.015$	$T = 0.030$	$T = 0.060$
$\hat{\beta}T$ 0.012	=	(0.66, 0.70)	(0.32, 0.35)	(0.16, 0.17)
$\hat{\beta}T$ 0.024	=	(1.7, 1.8)	(0.83, 0.88)	(0.41, 0.44)
$\hat{\beta}T$ 0.048	=	(4.2, 4.4)	(2.1, 2.2)	(1.0, 1.1)

Table 5.2: Flat plate. Type-I vortex: $m = 2$, $p = 0$, and $d = 0$ for Re = 5,040. Total dimensionless circulation, $\overline{\Gamma}_0 \equiv \Gamma_0/(U_0 a)$, in the trailing-edge vortex for the plots reported in Figure 5.4. Comparison of results for $\overline{\Gamma}_0$ obtained by integrating the DNS vorticity distributions (first entry in parentheses) to the predictions of SVT (second entry in parentheses).

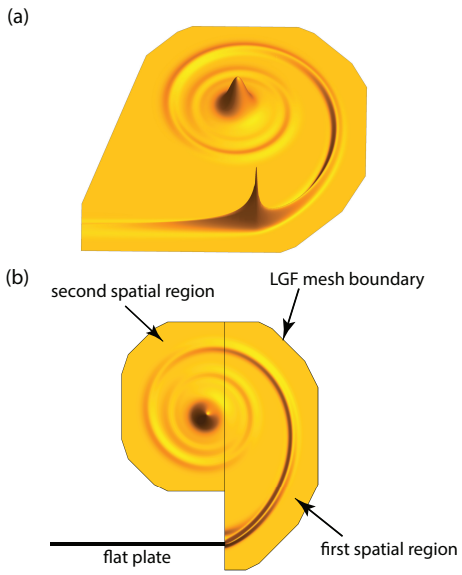


Figure 5.5: Flat plate. Type-I vortex: $m = 2$, $p = 0$, and $d = 0$ for Re = 5,040. DNS vorticity distribution in the trailing-edge wake region for $\hat{\beta}T = 0.048$ and $T = 0.015$, also plotted in the lower left-hand corner of Figure 5.4. (a) Three-dimensional perspective plot of vorticity distribution showing the trailing-edge wake and the vorticity distribution along the plate. Vorticity data is smoothed using a 3-point moving average across neighboring grid points in the direction parallel to the plate. (b) Segmentation of the wake region into two distinct spatial regions described in the text. The mesh boundary indicated is automatically generated by the LGF procedure; see Figure 5.2.

validity when $\hat{\beta}T = 0.048$, which causes a departure from the self-similar structure of the starting vortex.

Table 5.2 reports the total circulation, Γ_0, in the trailing-edge vortex for all plots in Figure 5.4. Results obtained from SVT (using equations (2.13) and (2.17) of Hinton et al. (2024)) and those obtained by numerically integrating the vorticity distributions of the DNS are reported. This integration is performed by segmenting the wake region into two regions: (1) the half space to the right of the plate's trailing edge, and (2) the spatial region to the left of this half space that omits the shear layer adjacent to the plate and the leading edge vortex. Figure 5.5 gives an example of this segmentation. There is an integrable singularity in the vorticity distribution at the plate's edge due to the sudden change in boundary condition from no-slip to free flow. This singularity does not significantly contribute to the total circulation. Agreement between SVT and DNS is observed throughout. Similar agreement is observed for the leading-edge vortex, as may be expected by symmetry (data not shown).

The results in Figure 5.4 and Table 5.2 collectively demonstrate the robustness of SVT in accurately predicting both the qualitative (shape) and quantitative (total circulation) features of the starting vortex.

5.4.1.2 $m = 0$, $p = 0$, $d = 0$

Figure 5.6 gives results for the trailing-edge vortex when $m = p = 0$. In contrast to Section 5.4.1.1, the plate undergoes an abrupt translational motion (in addition to an abrupt rotational motion) at $T = 0^+$. SVT also predicts a Type-I vortex to exist. However, the comparison in Figure 5.6 shows poor agreement in the vortex shapes between SVT and the DNS, even for the smallest time of $T = 0.018$. This is in contrast to the results in Figure 5.4, for $m = 2$ and $p = 0$, where agreement is observed for much longer times.

This difference is due to proximity of $m = p = 0$ to the critical line, $\epsilon = 0$, in the (m, p)-phase plane; as detailed in Section 5.3. For the parameters considered here, (5.16) gives $T_{\max} = T^{(2)}_{\max} \approx 2.5 \times 10^{-4}$. The computed times, T, in Figure 5.6 exceed this value by 2 to 3 orders of magnitude. This establishes that Type-I vortices are not expected in the DNS at the times shown, and indeed are not observed. This feature is due to strong competition between the two terms on the right-hand side of (5.10), which leads to simultaneous roll-up and convection of the starting vortex. The (observed) DNS vortex is not self similar but is a Type-III like vortex, as defined in Section 5.3. It is not possible to resolve the starting vortex for $T < T_{\max} \approx 2.5 \times 10^{-4}$ with our available computational resources. Despite these differences in vortex shape, the total circulation in the trailing-edge vortex is well predicted by SVT, except for the largest time of $T = 0.192$; see Table 5.3.

In contrast to the trailing-edge vortex, the leading-edge vortex exhibits a Kelvin-Helmholtz instability (data not shown). Thus, by replacing the smooth plate acceleration in Section 5.4.1.1 ($m = 2$) with an abrupt one here ($m = 0$), fore-aft antisymmetry in the DNS breaks. This is despite SVT predicting antisymmetry in the leading and trailing-edge vortices for $T < T_{\max}$, as in Section 5.4.1.1. We refrain from comparing the leading-edge vortex to SVT, because this theory is outside its regime of validity, as detailed above.

5.4.2 Type-II vortices

Next, we switch the time power-laws of m and p used in Section 5.4.1.1. That is, we now choose $m = 0$ and $p = 2$, for which Table 1 of PS21 predicts a Type-II

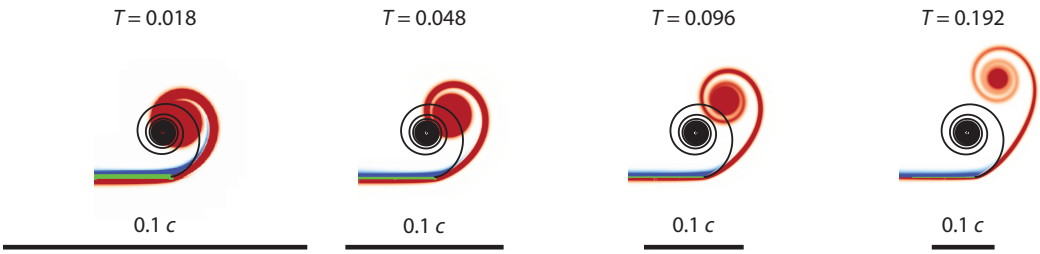


Figure 5.6: Flat plate. Type-I vortex: $m = 0$, $p = 0$, $d = 0$, $\hat{\beta} = 1$, and Re = 8,000. Description as for Figure 5.4.

$T = 0.018$	$T = 0.048$	$T = 0.096$	$T = 0.192$
$(0.93, 1.0)$	$(1.3, 1.4)$	$(1.8, 1.8)$	$(2.7, 2.2)$

Table 5.3: Flat plate. Type-I vortex: $m = 0$, $p = 0$, $d = 0$, $\hat{\beta} = 1$, and Re = 8,000. Total dimensionless circulation, $\overline{\Gamma}_0 \equiv \Gamma_0/(U_0 a)$, in the trailing-edge vortex for the plots reported in Figure 5.6. Description as per Table 5.2.

trailing-edge vortex sheet, i.e., it does not roll up. Hinton et al. (2024) predicts no starting vortex to occur at the leading edge. A Type-II vortex is yet to be observed in a (real) viscous fluid, whether it be DNS or physical experiment.

5.4.2.1 $m = 0$, $p = 2$, $d = 0$

Figure 5.7 gives DNS viscous flows generated at the leading and trailing edges, for $m = 0$, $p = 2$, $d = 0$, with Re = 32,000. A higher Reynolds number is possible here because vorticity is tightly confined to the plane of the flat plate; the LGF method discretizes the regions of finite vorticity only. Agreement is observed between SVT and the DNS. As predicted by SVT, the trailing-edge vortex moves downstream and parallel to the plate at small time, $T = 0.096$. The spatial extent of the DNS trailing-edge vortex precisely matches that of SVT. Also as predicted by SVT, no starting vortex is observed at the leading edge. Instead, flow at the leading edge resembles a classical Blasius boundary layer with downstream thickening.

Increasing time beyond $T = 0.096$, produces some discrepancy between SVT and the DNS. The greatest difference occurs in the right-hand column of Figure 5.7, for the largest value of $\hat{\beta} = 0.25$. A plate rotation of $\alpha = 0.54°$ exists here, which is incommensurately small relative to the observed vertical displacement of the wake. This does not explain the observed deviation. Instead, (5.16) predicts a maximal time of $T_{\max} = T^{(2)}_{\max} \approx 0.25$ for $\hat{\beta} = 0.25$, which is smaller than the observation time of $T = 0.384$. Thus, SVT is outside of its regime of validity, and a Type-III-like

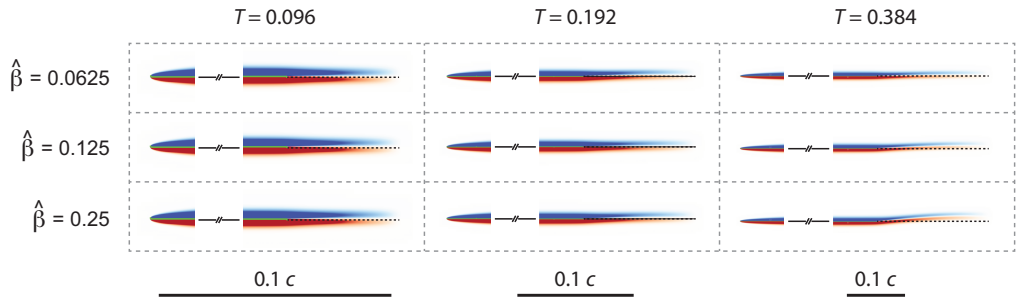


Figure 5.7: Flat plate. Type-II vortex: $m = 0$, $p = 2$, and $d = 0$ for Re = 32,000. Description as for Figure 5.4.

vortex is expected as per Section 5.3; consistent with the observed discrepancy.

Note that $T_{\max} = T^{(2)}_{\max} \approx 0.4$ and 0.63 for $\hat{\beta} = 0.125$ and 0.0625, respectively. Thus, SVT is on the edge of validity for $\hat{\beta} = 0.125$ ($T_{\max} \approx T = 0.384$), whereas it is valid for $\hat{\beta} = 0.0625$ ($T < T_{\max}$). This is also consistent with the observed level of agreement in the right-hand column of Figure 5.7.

5.4.2.2 Vorticity distribution in the trailing-edge wake

We note from Figure 5.7 that the DNS vorticity distributions in the trailing-edge wake consist of two distinct spatial regions. These contain apparently antisymmetric vorticity distributions (of opposite sign); generated by shear layers on opposite sides of the flat plate being swept downstream. However, SVT predicts a distinct asymmetry in the singular vorticity distribution, i.e., a nonzero velocity jump occurs across the vortex sheet. Figure 5.8(a) explores this prediction by plotting the vorticity distribution, ω, across the DNS trailing-edge wake; for the smallest time and rotational parameter reported in Figure 5.7. The magnitude of the vorticity decreases in the streamwise direction, as expected. SVT in (5.18) predicts that the wake will terminate at $x = 0.548c$, and this is clear in Figure 5.8(a). The data also shows that the vorticity distribution is not precisely antisymmetric about the plane of the plate, $y = 0$; a slight asymmetry about the vertical axis is evident.

To make a quantitative comparison with SVT, the DNS vorticity distributions (as in Figure 5.8(a)) are numerically integrated across the wake, i.e., in the y-direction. This produces the velocity jump across the wake, γ, which is reported in Figure 5.8(b) together with the prediction of SVT for $m = 0$ and $p = 2$,

$$\frac{\gamma}{U_0} = \frac{8\sqrt{2}}{3}\hat{\beta}\,T^{\frac{3}{2}}\left(1 - \frac{2\bar{x} - 1}{T}\right)^{\frac{3}{2}}, \qquad \frac{1}{2} \le \bar{x} \le \frac{1}{2}(1 + T), \tag{5.18}$$

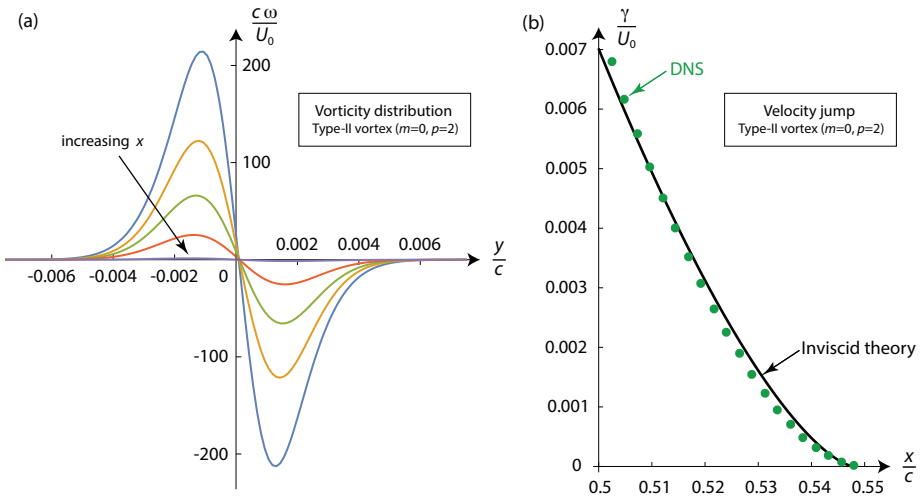


Figure 5.8: Flat plate. Type-II vortex: $m = 0$, $p = 2$, and $d = 0$ for Re = 32,000; for $T = 0.096$ and $\hat{\beta} = 0.0625$ (top left-hand entry of Figure 5.7). (a) Dimensionless vorticity distribution, $c\omega/U_0$, plotted across the trailing-edge wake at the spatial positions, $x/c = 0.51, 0.52, 0.53, 0.54, 0.548$; the last position is the end of the trailing-edge vortex as per (5.18), and the trailing edge is at $x/c = 0.5$; see Figure 5.1. (b) Dimensionless velocity jump, γ/U_0, across the trailing-edge vortex. DNS is evaluated by numerically integrating the vorticity distribution, ω, across the wake. SVT for γ/U_0 is (5.18). Note that the chord length, $c = 2a$.

where $\bar{x} \equiv x/c$ is dimensionless. This formula is obtained from equations (2.33) and (3.4) of Hinton et al. (2024); $\bar{x} = 0$ and $\bar{x} = 1/2$ correspond to the center and trailing-edge of the plate, respectively. Agreement of (5.18) with the DNS is observed, in spite of a very slight vorticity asymmetry; see Figure 5.8(b). Similar agreement is obtained for all cases reported in Figure 5.7, with the discrepancy in the velocity jump, γ, increasing slightly with increasing time (data not shown). Interestingly, the level of agreement is independent of $\hat{\beta}$ for fixed time, T, indicating that the rotational characteristics of the plate motion are decoupled from those of its translation. Integrating (5.18) over the spatial extent of the trailing-edge wake gives its total circulation,

$$\frac{\Gamma_0}{U_0 a} = \frac{16\sqrt{2}}{15}\hat{\beta}\,T^{\frac{5}{2}}, \tag{5.19}$$

which is compared to the DNS in Table 5.4, for all cases reported in Figure 5.7. Agreement is again observed. Eq. (5.19) is also recovered from Eq. (2.31) of Hinton et al. (2024).

These results provide evidence for the robustness and accuracy of SVT. It not only predicts the vortex shapes, but also the detailed quantitative characteristics of the flow. We remind the reader that SVT calculates the flow dynamics without any

	$T = 0.096$	$T = 0.192$	$T = 0.384$
$\hat{\beta} = 0.0625$	$(2.6, 2.7)\times10^{-4}$	$(1.6, 1.5)\times10^{-3}$	$(9.6, 8.6)\times10^{-3}$
$\hat{\beta} = 0.125$	$(5.3, 5.4)\times10^{-4}$	$(3.1, 3.0)\times10^{-3}$	$(1.9, 1.7)\times10^{-2}$
$\hat{\beta} = 0.25$	$(1.1, 1.1)\times10^{-3}$	$(6.2, 6.1)\times10^{-3}$	$(3.9, 3.4)\times10^{-2}$

Table 5.4: Flat plate. Type-II vortex: $m = 0$, $p = 2$, and $d = 0$ for Re = 32,000. Total dimensionless circulation, $\overline{\Gamma}_0 \equiv \Gamma_0/(U_0 a)$, in the trailing-edge vortex for the plots reported in Figure 5.7. Description as per Table 5.2.

reference to viscosity. Even so, the requisite properties of the vorticity distribution are accurately captured.

5.4.3 Type-III vortices

Finally, we explore another starting-vortex sheet type that was reported in PS21. The Type-III vortex sheet combines the features of Type I and II vortices, with roll up and convection downstream. Unlike the other vortex types, the shape of Type-III vortices depends on the rotation parameter, β, as discussed above.

5.4.3.1 $m = 0$, $p = 0.5$, $d = 0$

Figure 5.9 shows results for $m = 0$, $p = 0.5$, $d = 0$ at Re = 32,000. SVT predicts Type-III vortices at both leading and trailing-edges of the plate. Agreement with SVT is observed in Figure 5.9 for the trailing-edge vortex at the smallest time, $T = 0.048$. Eq. (5.16) gives $T_{\max} = T^{(1)}_{\max} \approx 0.68, 0.43, 0.27$ for $\hat{\beta} = 0.177, 0.354, 0.707$, respectively; these values also apply to the leading-edge vortex (which is explored below). Thus, we find that $T \lesssim T_{\max}$ for all cases in Figure 5.9, though results in the middle and bottom rows of the right-hand column are closest to SVT's limit of validity. Again, this is precisely where the greatest discrepancies are observed. Note that these vortices are no longer near the trailing edge. This further limits the validity of SVT, which is consistent with an increase in the level of disagreement. Table 5.5 reports results for the total circulation in the trailing-edge vortex, obtained from the SVT and DNS. The level of agreement is commensurate with the above discussion.

The leading-edge vortex, in contrast, exhibits the opposite trend, with the level of agreement increasing with increasing time, T. The reason for this seemingly counter-intuitive trend appears connected to the observation that the leading-edge vortex interacts strongly with the shear layer at the plate surface. The thickness of the shear layer is easily discerned from Figure 5.9; see the region downstream

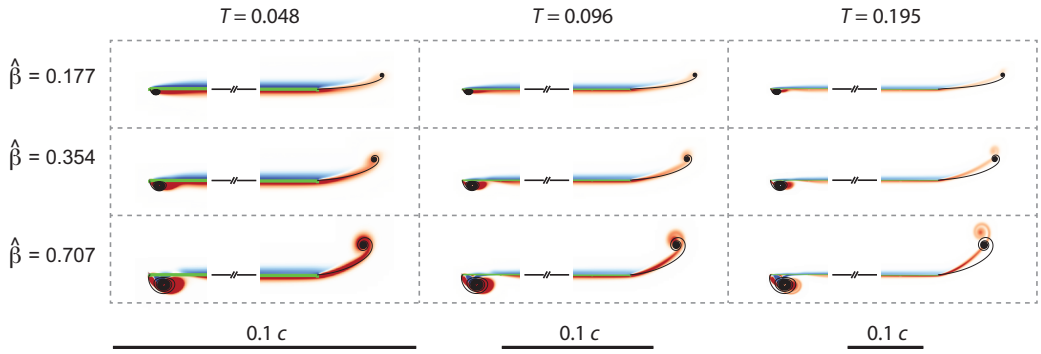


Figure 5.9: Flat plate. Type-III vortex: $m = 0$, $p = 0.5$, and $d = 0$ for Re = 32,000. Description as for Figure 5.4.

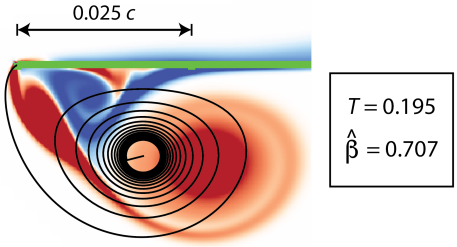


Figure 5.10: Flat plate. Zoomed-in version of leading-edge vortex in bottom right-hand plot of Figure 5.9; $m = 0$, $p = 0.5$, $d = 0$ at Re = 32,000.

	$T = 0.048$	$T = 0.096$	$T = 0.195$
$\hat{\beta} = 0.177$	$(1.9, 1, 9)\times10^{-2}$	$(4.0, 3.8)\times10^{-2}$	$(8.9, 7.7)\times10^{-2}$
$\hat{\beta} = 0.354$	$(4.0, 3.9)\times10^{-2}$	$(8.3, 7.7)\times10^{-2}$	$(1.8, 1.6)\times10^{-1}$
$\hat{\beta} = 0.707$	$(8.5, 8.2)\times10^{-2}$	$(1.8, 1.6)\times10^{-1}$	$(4.1, 3.3)\times10^{-1}$

Table 5.5: Flat plate. Type-III vortex: $m = 0$, $p = 0.5$, and $d = 0$ for Re = 32,000. Total dimensionless circulation, $\overline{\Gamma}_0 \equiv \Gamma_0/(U_0 a)$, in the trailing-edge vortex for the plots reported in Figure 5.9. Description as per Table 5.2.

from the leading-edge vortex. This interaction distorts the leading-edge vortex and appears to sweep it downstream—producing an elongated vortex relative to SVT and a secondary separation; see Figure 5.10. As time evolves, the leading-edge vortex also moves perpendicular to the plate. This weakens its interaction with the shear layer at the plate. The vortex is then observed to approach what is expected from SVT. Unlike the trailing-edge vortex, the leading-edge vortex remains close to its edge for all reported times, enabling a greater level of agreement with SVT at larger times. The Reynolds number of Re = 32,000 used in these simulations is evidently not sufficiently high to restrict interaction of the leading-edge vortex with the shear layer at the plate, at the smaller times. Available computational resources

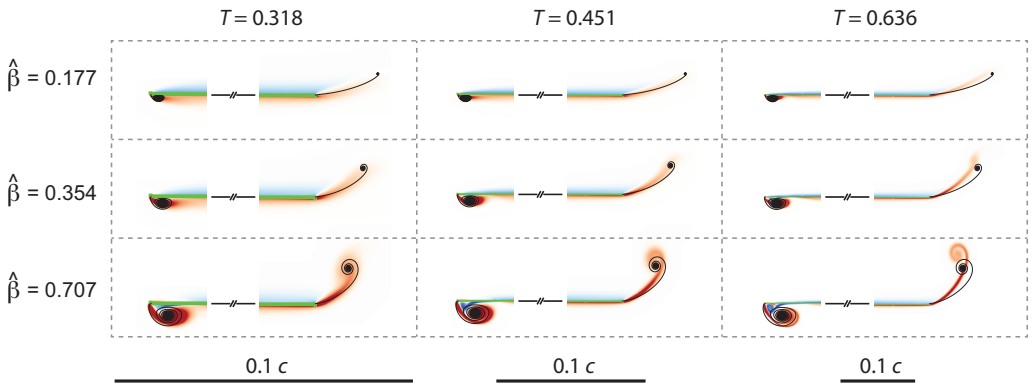


Figure 5.11: Flat plate. Type-III vortex: $m = 1$, $p = 2$, and $d = 0$ for Re = 45,255. Description as for Figure 5.4.

	$T = 0.318$	$T = 0.451$	$T = 0.636$
$\hat{\beta} = 0.177$	$(7.9, 7.7)\times10^{-3}$	$(2.4, 2.2)\times10^{-2}$	$(7.3, 6.2)\times10^{-2}$
$\hat{\beta} = 0.354$	$(1.7, 1.9)\times10^{-2}$	$(5.0, 5.5)\times10^{-2}$	$(1.6, 1.6)\times10^{-1}$
$\hat{\beta} = 0.707$	$(3.8, 4.9)\times10^{-2}$	$(1.2, 1.4)\times10^{-1}$	$(3.8, 3.9)\times10^{-1}$

Table 5.6: Flat plate. Type-III vortex: $m = 1$, $p = 2$, and $d = 0$ for Re = 45,255. Total dimensionless circulation, $\overline{\Gamma}_0 \equiv \Gamma_0/(U_0 a)$, in the trailing-edge vortex for the plots reported in Figure 5.11. Description as per Table 5.2.

do not permit simulations at higher Re. We refrain from reporting results for the total circulation in the leading-edge vortex due to its strong overlap with the plate's shear layer. This complicates the numerical separation of vorticity contributions from the leading-edge vortex and the plate's shear layer, e.g., using the procedure illustrated in Figure 5.5 or a variant.

5.4.3.2 $m = 1$, $p = 2$, $d = 0$

Figure 5.11 gives a similar comparison for $m = 1$, $p = 2$, $d = 0$ at Re = 45,255, for which SVT also predicts Type-III vortices. In this case, (5.16) gives $T_{\max} = T^{(1)}_{\max} \approx 0.82, 0.66, 0.52$ for $\hat{\beta} = 0.177, 0.354, 0.707$, respectively, for both leading and trailing edges. Thus, again we have $T \lesssim T_{\max}$ for all results in Figure 5.11. This indicates that SVT holds throughout, provided the vortex stays close to the edge in question. Agreement is observed for the trailing-edge vortices in Figure 5.11, except for the longest time, $T = 0.636$, which coincides with these vortices not being near the plate (spatial proximity to the edge is required for SVT to hold).

Interestingly, the discrepancy between SVT and the DNS for the leading-edge vortices is smaller than that observed in Section 5.4.3.1. The leading-edge vortices in

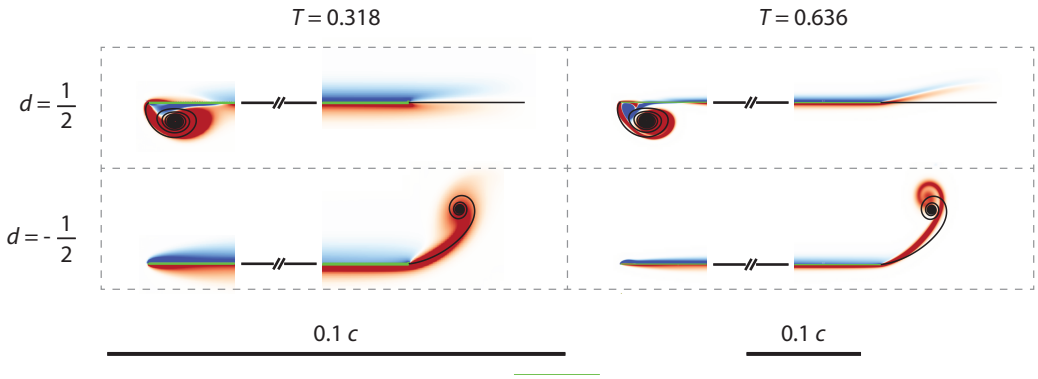


Figure 5.12: Flat plate. Different vortex types at leading and trailing edges: $m = 1$, $p = 2$, $\beta = 0.707$, and $d = \pm 1/2$ for Re = 45,255. Results for $d = 0$ correspond to $\hat{\beta} = \beta/2 = 0.354$ and are given in the middle row of Figure 5.11. Remainder of description as for Figure 5.4.

Figure 5.11 are further from the plate relative to Figure 5.9, which decreases their interaction with the plate's shear layer. This is consistent with the better level of agreement between SVT and DNS relative to Section 5.4.3.1.

Table 5.6 gives corresponding results for the total circulation in the trailing-edge vortex. Agreement between SVT and DNS is again observed, except for $\hat{\beta} = 0.707$ and $T = 0.318$. This is not expected at first sight because it occurs at the smallest time studied, where SVT is predicted to be valid; see above. However, the corresponding DNS trailing-edge vortex in the bottom left-hand plot of Figure 5.11 does not exhibit the significant vortex core present at larger times, and it displays a strong level of diffusion. In contrast, SVT predicts the vortex to have the same shape at all times. The observed diffusion in the DNS vortex may contribute to this difference in total circulation. Resolved DNS at higher Reynolds numbers are limited by available computational resources.

5.4.3.3 $m = 1$, $p = 2$, $d = \pm 1/2$

Using SVT, Hinton et al. (2024) predicted that plate rotation about either its quarter-chord and three-quarter-chord positions, i.e., $d = \pm 1/2$, can produce vortices of different types at the leading and trailing edges. Figure 5.12 shows a comparison of this inviscid prediction to the DNS.

In line with Hinton et al. (2024), plate rotation about the three-quarter-chord position, $d = 1/2$, produces Type III and II vortices at the leading and trailing edges, respectively. Moreover, the level of agreement between SVT and the DNS as time

	$T = 0.318$	$T = 0.636$
$d = \frac{1}{2}$	$(7.9, 8.2)\times10^{-4}$	$(2.2, 2.6)\times10^{-2}$
$d = -\frac{1}{2}$	$(3.6, 4.9)\times10^{-2}$	$(3.3, 3.9)\times10^{-1}$

Table 5.7: Flat plate. Total dimensionless circulation, $\overline{\Gamma}_0 \equiv \Gamma_0/(U_0 a)$, in the trailing-edge vortex for the plots reported in Figure 5.12. Description as per Table 5.2.

evolves, is commensurate with the discussion in the previous sections. Note that the theory in Section 5.3 for $T_{\max}$ does not apply to cases where $d = \pm 1/2$, as discussed.

Plate rotation about the quarter-chord position, $d = -1/2$, completely suppresses the leading-edge vortex at the smaller time of $T = 0.318$ in Figure 5.12. This again agrees with the prediction of SVT. Doubling the measurement time to $T = 0.636$ produces a small separation bubble at the leading edge. This corresponds to an instantaneous angle-of-attack of $\alpha \approx 3.5°$, which further highlights the robustness of SVT and the utility of calculating the starting vortices. The trailing edge generates a Type-III vortex, again in agreement with SVT; the agreement is superior for the smaller time, as expected. Note that the SVT trailing-edge vortices for $d = -1/2$ (bottom row of Figure 5.12) are identical to those reported in the bottom row of Figure 5.11; the corresponding DNS trailing-edge vortices are also similar in form. This validation of SVT reinforces its implications to the airfoil design of low-speed aircraft, such as MAVs and UAVs, which were discussed in Hinton et al. (2024). Dynamic suppression of the leading-edge vortex can also affect thrust generation in flapping wings (Alben, 2010; Fang et al., 2017; Heydari and Kanso, 2021).

The corresponding total circulations in the trailing-edge vortices are reported in Table 5.7. Agreement is again observed, with a larger discrepancy in the bottom left-hand entry of Table 5.7. This again coincides with significant diffusion in the DNS trailing-edge vortex in the bottom-left-hand entry of Figure 5.12; see previous paragraph and the corresponding discussion at end of Section 5.4.3.2 for an explanation of this discrepancy.

5.5 Joukowski airfoil

We conclude our study by considering the Joukowski airfoil which was studied in Section 3.5 of Hinton et al. (2024). The purpose is to assess the utility of SVT to an airfoil of more complex shape. Two symmetric Joukowski airfoils of different thickness are considered. The first is relatively thin with a Joukowski parameter

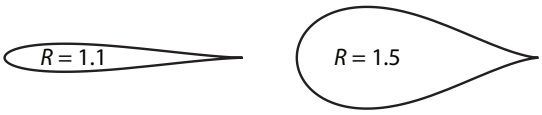


Figure 5.13: The two symmetric Joukowski airfoils considered in this study. Each has a sharp and straight trailing edge, which is evident by their upper and lower surfaces being tangent at the trailing edge. The Joukowski parameter, R, is defined in Eq. (3.29) of Hinton et al. (2024) and specifies the airfoil thickness.

of $R = 1.1$, and is streamlined. The second is deliberately chosen to be bluff in its cross-section, with $R = 1.5$. Illustrations of these two airfoils are given in Figure 5.13.

5.5.1 Starting vortex inviscid theory

In line with Section 3.5 of Hinton et al. (2024), the airfoil does not rotate but instead translates in two orthogonal directions with independent power-laws in time. Specifically, each airfoil moves with the following velocity components,

$$U(T) = U_0 \mathcal{V} T^m, \qquad V(T) = U_0 \mathcal{W} T^n, \tag{5.20}$$

where U and V are the velocity components parallel and perpendicular to the airfoil's symmetric axis, respectively, U_0 is the dimensional velocity scale, $\mathcal{V}$ and $\mathcal{W}$ are dimensionless positive constants that control the relative magnitudes of the velocity components, and the translational power-laws are $m, n \geq 0$.

We focus on the case of $m = 0$ and $n = 2$ reported in Section 3.5.1 of Hinton et al. (2024). SVT predicts that Type-II vortices form at the trailing-edge. Analytical formulas for the spatial extent of the trailing-edge vortex, the velocity jump across the trailing-edge vortex, and its total circulation are given by

$$\frac{\gamma}{U_0} = \frac{8\sqrt{2}}{3}\left(1 - \frac{x}{x_{\max}}\right)^{\frac{3}{2}} \sqrt{\frac{\mathcal{R}}{V}} \mathcal{W} T^{\frac{3}{2}}, \qquad 0 \leq x \leq x_{\max}, \tag{5.21a}$$

$$\frac{\Gamma_0}{U_0 a} = \frac{16\sqrt{2}}{15} \sqrt{\frac{\mathcal{V}}{R}} \mathcal{W} T^{\frac{5}{2}}, \tag{5.21b}$$

where the (dimensional) spatial extent of the wake is

$$x_{\max} = \frac{\mathcal{V} T}{R} a, \tag{5.22}$$

and a is the half-chord of a Joukowski airfoil with $R = 1$; $x = 0$ corresponds to the trailing edge of the Joukowski airfoil (this differs from the origin of the flat plate

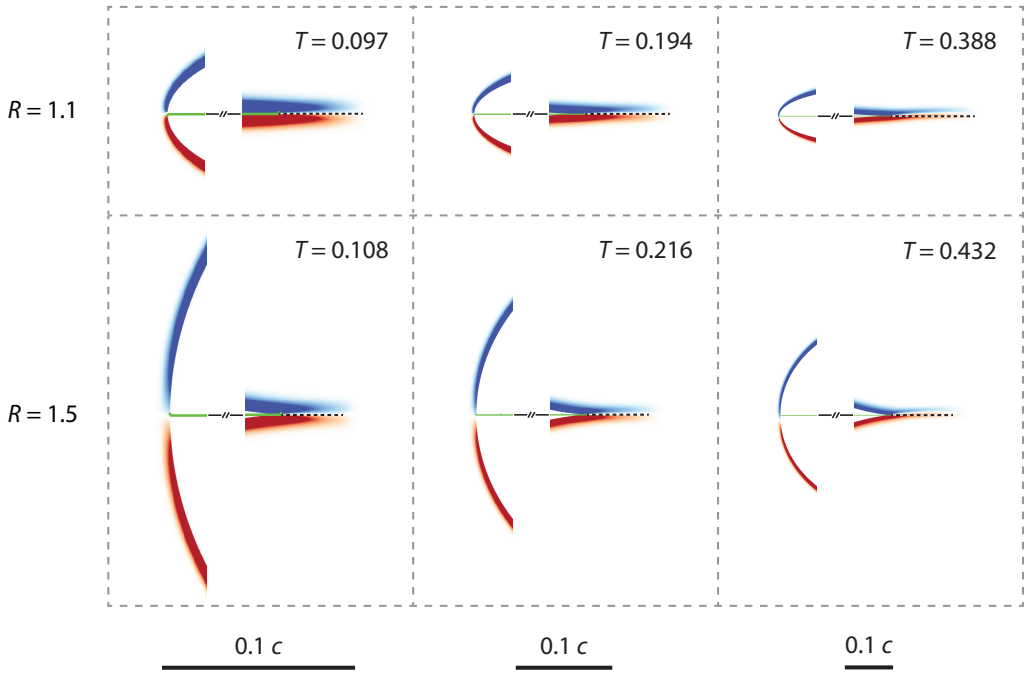


Figure 5.14: Joukowski airfoils. Type-II vortex: $m = 0$, $n = 2$, $\mathcal{V} = 1$, $\mathcal{W} = 0.25$, for Re = 8,000; here c is the chord of the airfoil. Leading (left) and trailing (right) edges of the airfoils are shown. Description as for Figure 5.4.

	$\overline{\Gamma}_0 \equiv \Gamma_0/(U_0 a)$			
$R = 1.1$	$(8.1, 10.3) \times 10^{-4}$		$(5.0, 5.8)\times10^{-3}$	$(3.0, 3.3)\times10^{-2}$
$R = 1.5$	$(8.9, 9.3)\times10^{-4}$		$(5.2, 5.3)\times10^{-3}$	$(3.0, 3.0)\times10^{-2}$

Table 5.8: Joukowski airfoil. Total dimensionless circulation, $\overline{\Gamma}_0 \equiv \Gamma_0/(U_0 a)$, in the trailing-edge vortex for the respective plots reported in Figure 5.14; where a is the half-chord of a Joukowski airfoil for $R = 1$. The entry positions in this table (and corresponding times, T) coincide with Figure 5.14. Description as per Table 5.2.

above). Note that the chord length of the symmetric Joukwoski airfoil defined in Eq. (3.29) of Hinton et al. (2024) is

$$c = \frac{2R^2}{2R-1} a. \tag{5.23}$$

The dimensionless time, T, is as defined in (5.2).

5.5.2 Direct numerical simulations

Figure 5.14 shows a comparison of the trailing-edge vortex calculated using SVT with the DNS; simulation parameters are reported in the caption of Table 5.1. Agreement in the spatial extent of this vortex is observed especially for $R = 1.1$.

Some discrepancy emerges when the airfoil's finite thickness (upstream from its trailing edge) becomes visible in the plots. In these cases, the wake's spatial extent exceeds that of the straight trailing-edge and the vortex cannot be considered close to a 'sharp and straight edge' (see definition in Section 5.1). This is a fundamental constraint of SVT and is consistent with the discrepancies observed in the second row of Figure 5.14. Increasing the Reynolds above Re = 8,000 results in separation upstream of the trailing edge for $R = 1.5$ (near the maximum width of the airfoil), which complicates further numerical investigation.

Table 5.8 reports the total circulation in the trailing-edge wake, computed by SVT and DNS. The level of agreement is similar to that in Table 5.4 for Type-II vortices generated by a flat plate (for a much higher Reynolds number, Re = 32,000). Some discrepancy exists for $R = 1.1$ with SVT overestimating the total wake circulation, especially for the smallest time. This appears to correlate with a wide and short wake, for which a vortex sheet description may exhibit some error. The wake's aspect ratio (width over length) decreases with increasing time, which is seen to align with improved agreement between SVT and the DNS. Results for $R = 1.5$ in Table 5.8 display enhanced agreement relative to $R = 1.1$, with a similar trend for increasing time, despite the elongated wake of the DNS relative to SVT; see second row of Figure 5.14. This is consistent with the vorticity generating mechanism (the airfoil's no-slip condition) being decoupled from downstream effects where the flow accelerates (not included in SVT). These collective results for the total wake circulation align with those for the velocity jump distribution across the wake in Figure 5.15. Agreement between SVT and DNS again improves slightly with increasing time (data not shown).

5.6 Conclusions

Starting vortex inviscid theory (SVT) predicts that three distinct types of starting vortex sheets—Types I, II, and III—can be generated by an arbitrary body with sharp and straight edges that suddenly moves. This study investigated the existence of these vortices in a viscous fluid for a flat plate and two symmetric Joukowski airfoils.

The motion of a translating and rotating flat plate is characterized by two respective power-laws in time, m and p, which select the vortex type. An abrupt transition between the starting-vortex types is predicted to occur in the (m, p)-phase plane; see Figure 5.1. Type-I vortex sheets had been previously reported, while PS21

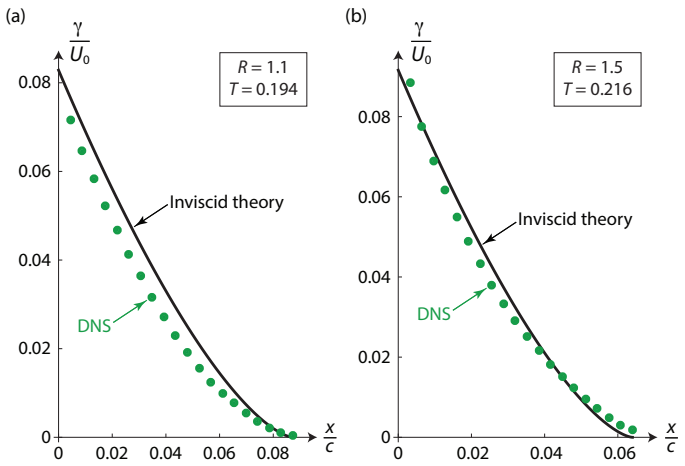


Figure 5.15: Joukouski airfoil. Type-II vortex: $m = 0$, $n = 2$, for Re = 8,000. Dimensionless velocity jump, γ/U_0, across the trailing-edge vortex. DNS is evaluated by numerically integrating the vorticity distribution across the wake. SVT for γ/U_0 is (5.21a). (a) Thinner airfoil, $R = 1.1$. (b) Thicker airfoil, $R = 1.5$, where DNS data is shown to the spatial extent predicted by SVT; DNS data extends slightly further downstream. These plots correspond to trailing-edge wakes shown in the central column of Figure 5.14.

introduced Types II and III to the literature and were yet to be observed in a (real) viscous fluid. Hinton et al. (2024) extended these findings using inviscid theory to an arbitrary body with sharp and straight edges.

In this study, we reported a detailed assessment of these inviscid predictions using high-fidelity DNS of the Navier-Stokes equations. We also used SVT to calculate the time interval over which each starting-vortex type exists. It was found that for finite-time observations of a flat plate, there is a smooth (rather than abrupt) transition between the vortex types, as m and p are varied. This is because the time interval over which Type I and II vortices exist, becomes vanishingly small as the critical (m, p)-phase plane line for Type-III vortices is approached. We compared SVT to the DNS for values of m and p that span all vortex types. Agreement was observed when SVT was predicted to be valid. This included the velocity jump across a Type-II vortex sheet and the total circulation in the wake of all vortex types. Moreover, different vortex types at the leading and trailing edges of a flat plate were observed in the DNS, for plate rotation about its quarter-chord and three-quarter-chord positions. Dynamic suppression of the leading-edge vortex for plate rotation about its quarter-chord position was confirmed. This position coincides with the aerodynamic centre of the airfoil and is relevant to the airfoil design of low-speed aircraft, such as MAVs and UAVs.

The predictions of SVT for two symmetric Joukwoski airfoils that translate in two orthogonal directions were also compared to DNS. Agreement was again observed for Type-II vortices when their spatial extent does not exceed that of its sharp and straight edge—an overarching assumption of SVT. When this does occur, SVT was found to (1) underestimate the spatial extent of the wake, due to an accelerating flow downstream of the trailing edge, but (2) accurately predict both the velocity jump across the wake and its total circulation. These comparisons highlight the utility of SVT for bodies of arbitrary shape.

This study confirms the existence of Type II and III vortices in a viscous fluid, and validates the accuracy of SVT. We emphasize that SVT is predictive with no adjustable parameters. The inviscid theory reported by PS21, and its generalization by Hinton et al. (2024) for an arbitrary body—collectively referred to as the SVT—can therefore be used with confidence to calculate starting vortices in a viscous fluid.

Chapter 6

STABILITY ANALYSIS OF THE FLOW PAST A ROTATING CYLINDER WITH A CONTROL CYLINDER IN THE WAKE

In this chapter, we demonstrate the algorithms described in Chapter 4. Specifically, we investigate an interesting flow phenomenon generated by combining the two stabilizing effects of the flow past a cylinder: the rotational motion of the main cylinder and putting a small control cylinder in the wake of the main cylinder.

6.1 Introduction

In previous chapters, we discussed two interesting phenomena involving the flow past a cylinder. The first phenomenon is the flow past a cylinder with a small control cylinder in the wake. By placing a small control cylinder in the wake of the main cylinder, we can suppress the onset of vortex shedding. The second phenomenon is the flow past a rotating cylinder. Below a certain rotational rate, we can also suppress the onset of vortex shedding. From a linear perspective, combining these two methods, i.e. placing the control cylinder in the wake of a rotating cylinder, may suppress the onset of vortex shedding at an even higher Reynolds number compared to using only one of these two methods. In this chapter, we will show that this linear perspective is not accurate: combining both methods might induce three-dimensional instability mechanisms that are stable without the control cylinder.

6.2 Problem Setting

We consider the flow past a rotating cylinder with a smaller control cylinder in its wake. We denote the diameter of the main cylinder as D and the diameter of the control cylinder as d. The ratio between the two cylinders is fixed at $D/d = 10$. The Reynolds number is defined in terms of the main cylinder and fixed at $Re = 60$ throughout this chapter:

$$Re = \frac{U_\infty D}{\nu} \tag{6.1}$$

where U_∞ is the freestream velocity and ν the kinematic viscosity. We defined the non-dimensional rotational rate as

$$\alpha = \frac{\Omega D}{2U_\infty} \tag{6.2}$$

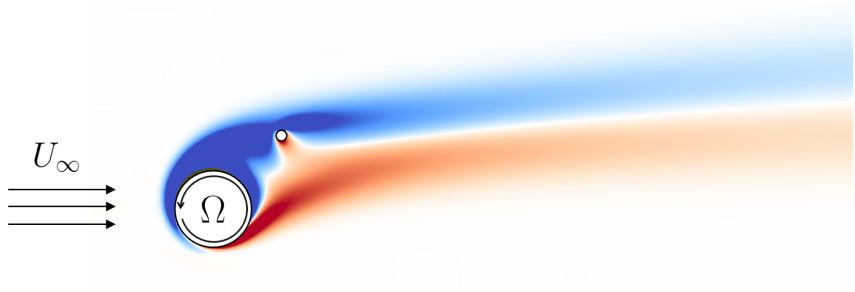


Figure 6.1: Flow past a rotating cylinder with a control cylinder in its wake.

where Ω is the angular velocity of the rotational motion. A demonstration of the problem we are studying is shown in Figure 6.1. We let the center of the main cylinder be the origin. We denote the center of the control cylinder as (x_c, y_c). For the subsequent stability analysis, we fix $x_c/D = y_c/D = 1$.

6.3 Background Problems

In this section, we briefly summarize the two background problems that appeared in earlier chapters.

Stability analysis of the flow past a cylinder with a control cylinder in its wake For the flow past a stationary cylinder, if we put a small control cylinder in the wake of the main cylinder, we can suppress the onset of vortex shedding. Specifically, for Reynolds number below 63, if we put the control cylinder in the wake of the main cylinder, we can suppress the onset of vortex shedding. This is shown by Strykowski and Sreenivasan (1990) and also demonstrated in Figure 6.2.

Stability analysis of the flow past a rotating cylinder In Chapter 4, one of the validating cases we used was the flow past a rotating cylinder. As studied by Pralits, Giannetti, and Brandt (2013), at Reynolds number of 60, rotating the cylinder will suppress the onset of vortex shedding if the rotational rate is below 4. Specifically, if the rotational rate $\alpha \in [1.5, 4] \cup [5.5, 6]$, the flow remains steady and time-invariant at $Re = 60$. Specifically, one can map out the parameter space for which the flow is unstable, as shown in Figure 6.3. Notably, Figure 6.3 presents two distinct instability mechanisms: one at lower rotational rate and one at high rotational rate. Those two instability mechanisms are termed mode I (low rotational rate) and mode II (high rotational rate) instability, respectively (Pralits, Brandt, and Giannetti, 2010).

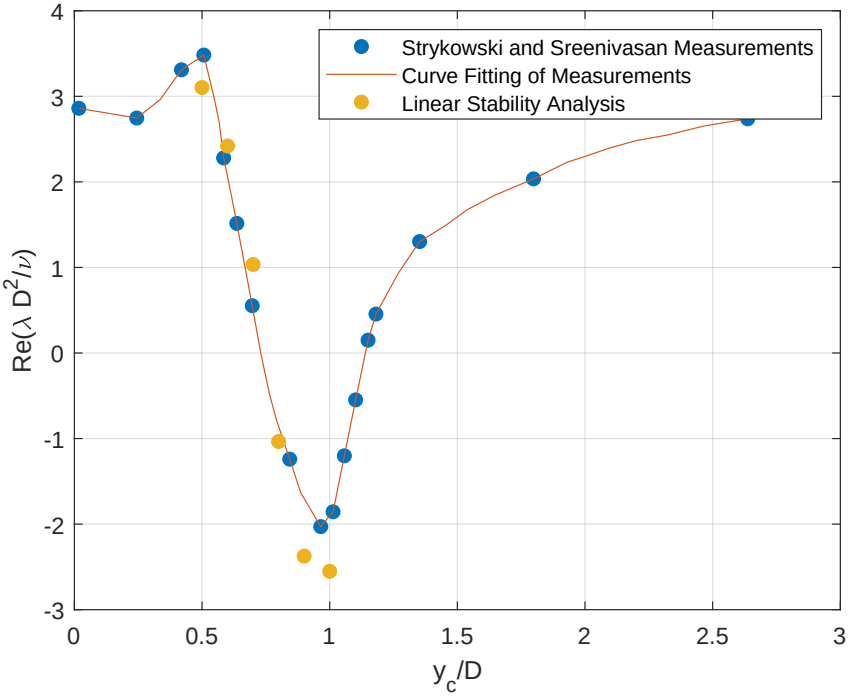


Figure 6.2: Perturbation of growth rates for variable vertical control cylinder locations with a fixed horizontal location at $x_c/D = 1.2$. Previously shown in Chapter 4.

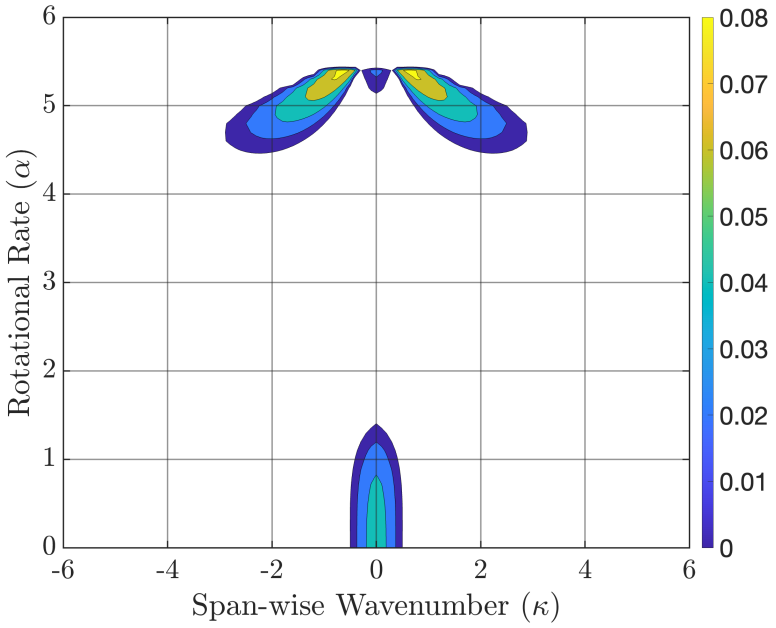


Figure 6.3: Parameters space for which the flow past a rotating cylinder is unstable at $Re = 60$. The contours are isocontours of perturbation growth rate. Contours are in increments of 0.02.

6.4 Stability Analysis

As discussed in the previous section, for the flow past a cylinder at $Re = 60$, both rotating the main cylinder and putting a small control cylinder can result in the suppression of vortex shedding. In this section, we will introduce the steps we used to conduct three-dimensional stability analysis.

Recall from Chapter 4 that the mathematical problem we solve is the eigenvalue problem of the linearized three-dimensional Navier-Stokes equations around some two-dimensional equilibrium. We then Fourier-expand the linearized Navier-Stokes equations along the span-wise direction. In its abstract form, we can write the dynamics of a Fourier coefficient with span-wise wavenumber κ linearized around some two-dimensional equilibrium $\boldsymbol{q}_s$ as the following:

$$\frac{d\boldsymbol{q}}{dt} = A(\boldsymbol{q}_s, \kappa)\boldsymbol{q} \tag{6.3}$$

where $A(\boldsymbol{q}_s, \kappa)$ is the discretized and linearized (around $\boldsymbol{q}_s$) Navier-Stokes operator. Due to the large ratio between the diameter of the main cylinder and that of the control cylinder, we can leverage the multi-resolution property of the algorithm presented in Chapter 4 to efficiently solve the problem. One of the examples of a computational grid is shown in Figure 6.4. It shows the significant length-scale difference between the small control cylinder and the large main cylinder. By only refining the computational grids around the small control cylinder and the boundary layer of the main cylinder, we can save a significant amount of computational resources. Then for a sequence of spanwise wavenumbers $\{\kappa_i\}$, we compute the corresponding eigenvalue spectrums. If there exists an eigenvalue with a positive real part, exciting the flow using the corresponding eigenvector and wavenumber will induce instability. Otherwise, the flow is linearly stable for that particular wavenumber. In particular, for $\kappa = 0$, the corresponding perturbations are two-dimensional, i.e. constant in the span-wise direction.

To conduct the stability analysis for a fixed rotational rate α, we follow a two-step process:

1. Compute the equilibrium using Newton iterations;
2. Compute the eigenvalue spectrums for a set of wavenumbers from 0 to 6 with 0.1 increments.

By collecting the results from the second step, we can find if perturbing the flow with the corresponding wavenumber will lead to instability.

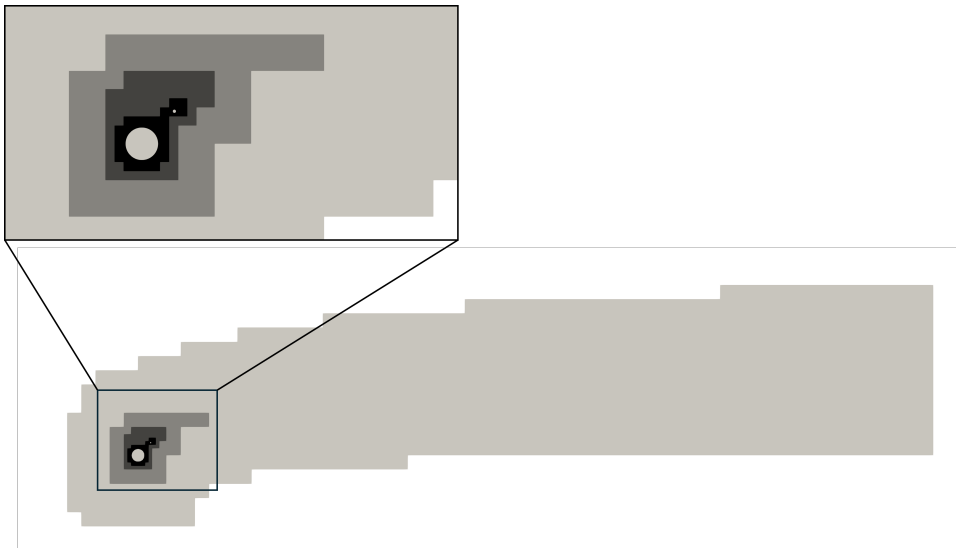


Figure 6.4: Computational grid used to solve one of the eigenvalue problems. The grid density increases as a factor of 4 as the color darkens.

6.5 Results

Similar to the study we did in Chapter 4, we compute the stability problem across a range of parameters within the range $\alpha \in [0, 6.0]$ with increments of 0.1. We compute the equilibrium solution via the Newton iterations. Then, we computed the stability problem for the wavenumbers $\kappa \in [0.0, 6.0]$ with increments of 0.1. We plot the region with positive perturbation growth rates in Figure 6.5.

We can directly compare the stability characteristics of this flow to the one without a control cylinder by comparing Figure 6.5 and 6.3. At low rotational rates, the mode I instability is suppressed entirely with the presence of the control cylinder. This is consistent with the observation by Strykowski and Sreenivasan (1990). However, at high rotational rates ($\alpha \geq 4.4$), mode II instability is slightly excited by placing a control cylinder in the wake of the main cylinder. To see this more clearly, we compare the perturbation growth rate for different wavenumbers at rotational rates $\alpha = 4.4$ and 5.0. The comparison is shown in Figure 6.6. At $\alpha = 4.4$, when there is no control cylinder, the flow is linearly stable and all the growth rates are negative across. However, when we place a small control cylinder in the wake, we see positive perturbation growth for a range of wavenumbers, indicating small unstable three-dimensional modes. At $\alpha = 5.0$, placing a small control cylinder suppresses the mode II instability but the flow is still unstable. In summary, placing a small control cylinder (in the location we have chosen) in the wake of the flow past a rotating cylinder suppresses the mode I instability mechanism but slightly

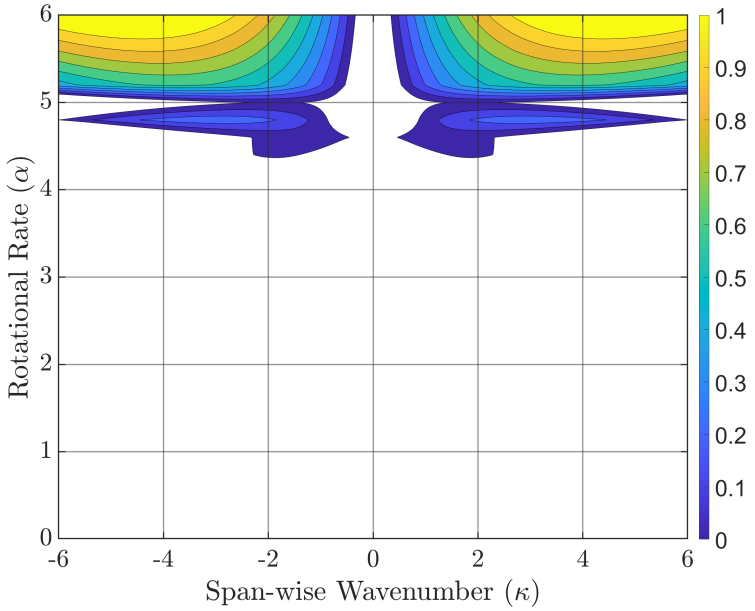
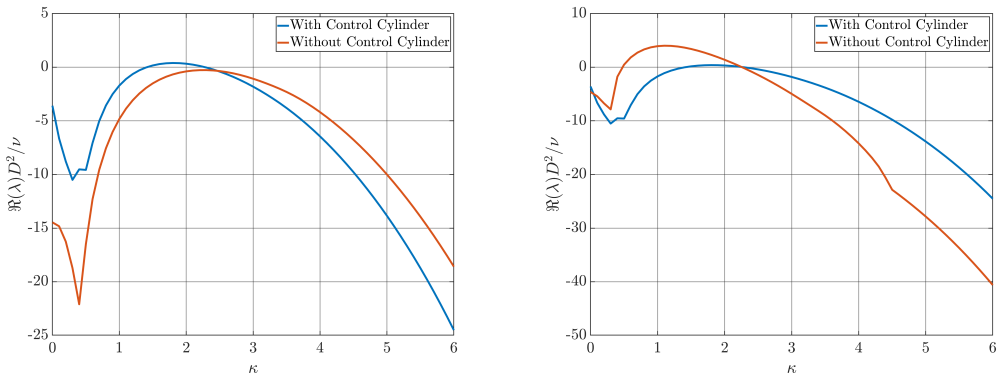


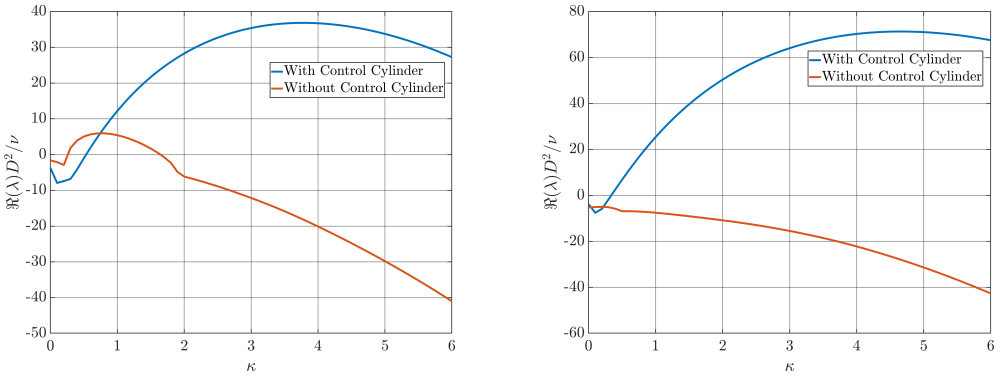
Figure 6.5: Parameters space for which the flow past a rotating cylinder is unstable at $Re = 60$ when there is a small control cylinder in the wake. The contours are isocontours of perturbation growth rate. Contours are in increments of 0.1.



(a) $\alpha = 4.4$. (b) $\alpha = 5.0$.

Figure 6.6: Comparison of the perturbation growth rate $\Re(\lambda)$ for the flow past a rotating cylinder at $Re = 60$ between with and without a small control cylinder in the wake.

excites the mode II instability mechanism. At an even higher rotational rate of $\alpha \geq 5.2$, we see a different type of instability that has a much higher perturbation growth rate and larger wavenumber than the instability at $\alpha < 5.2$. The perturbation growth rate across a range of wavenumbers is shown in Figure 6.7. In the regime, the perturbation growth rate increases as the rotational rate increases, exhibiting a different type of instability than both mode I and mode II instability.



(a) $\alpha = 5.2$. (b) $\alpha = 6.0$.

Figure 6.7: Comparison of the perturbation growth rate $\mathfrak{R}(\lambda)$ for the flow past a rotating cylinder at $Re = 60$ between with and without a small control cylinder in the wake at higher rotational rates.

6.6 Concluding Remarks

In this section, we investigated the stability criterion of the flow past a rotating cylinder with a small control cylinder in the wake of the main cylinder. We conducted a series of stability analyses to demonstrate the algorithm presented in Chapter 4. By leveraging the multi-resolution capability of this algorithm, we are able to cover a large parameter space to map out the stability characteristics of the flow. By doing so, we discovered that placing a small control cylinder in the wake (in the position we chose) suppresses the mode I instability but slightly excites the mode II instability. For rotational rates higher than 5.2, we also see the rise of another type of instability that becomes increasingly unstable as the rotational rate increases.

We note that there are further investigations that can be done to further understand the effects of placing a small control cylinder on the stability characteristics of the flow past a cylinder. To understand this phenomenon further, we need to vary the location of the control cylinder to give a more comprehensive understanding of how the small control cylinder affects the stability characteristics of the flow past a rotating cylinder.

Chapter 7

CONCLUDING REMARKS AND FUTURE WORK

In this thesis, we discussed two algorithms combining the immersed boundary (IB) method, lattice Green's function (LGF) approach, and adaptive mesh refinement (AMR) to create fast numerical algorithms for simulating and analyzing spanwise periodic external incompressible flows. We demonstrated those algorithms through various flow simulation and analysis studies.

For these algorithms to be fast, efficiently tabulating the LGF of the screened Poisson equation is a prerequisite. Thus, we first derived algorithms for tabulating these LGFs in Chapter 2. In this chapter, we extended existing techniques to tabulate the LGF for the Poisson operator to compute the entries of the LGFs of the screened Poisson operators. Through numerical analysis, asymptotic expansion, and FFT, we proposed a set of new algorithms to compute entries of the LGFs for the screened Poisson equation and provided convergence rate guarantees. We demonstrated that the resulting algorithms are significantly faster than existing methods. The algorithms derived in this chapter serve as a building block to create the subsequent flow simulation algorithm and analysis algorithm using the LGFs.

Utilizing the algorithms derived in Chapter 2, we propose an algorithm to simulate spanwise periodic external incompressible flows in Chapter 3. In this algorithm, we leveraged the spanwise periodicity by Fourier-expanding the velocity and pressure variables. Through the incompressible Navier-Stokes equations, we derived the evolution equations of the corresponding Fourier coefficients while incorporating the immersed boundary formulation. With this formulation, we are able to adapt an existing algorithm that simulates the fully 3D incompressible external flow using LGF, AMR, and the IB method by Yu (2021). In addition, we specialized this algorithm to better leverage the spanwise periodicity by introducing a parallel version of the direct IB forcing solver. The resulting algorithm is demonstrated to be both accurate and efficient when simulating spanwise periodic external incompressible flows.

In addition to the flow simulation algorithm, we also proposed an algorithm to analyze external flows via linear stability analysis and resolvent analysis. The mathematical formulations of these analyses are well-established in previous works

(Taira and Colonius, 2007). Mathematically, they are some forms of continuous operator eigenvalue problems. However, the challenge to conduct such analyses in external flow systems resides in efficiently discretizing and numerically computing the resulting eigenvalue problems. In Chapter 4, we propose an algorithm to efficiently discretize and formulate the eigenvalue problems. By leveraging the advantages of the flow solver we presented in Chapter 3, we developed a boundary integral formulation to solve the operator eigenvalue problems. In addition, using the fast multipole method (FMM), we greatly reduced the computational complexity of the eigenvalue problems. We validated the resulting algorithms using a wide range of interesting flow cases.

Using the flow simulation algorithm we presented in Chapter 3, we studied the starting vortices generated by translating and rotating bodies with sharp and straight edges in a viscous fluid. Leveraging the flow simulation algorithm, we can efficiently simulate the early time behavior of the starting vortices. We conducted direct numerical simulations (DNS) of various flow configurations. By analyzing the resulting vorticity distribution and comparing it to the starting vortex inviscid theory (SVT) by Pullin and Sader (2021), we verified that SVT still manifests in incompressible flow, even in the presence of viscosity. As a result, we confirm the validity and accuracy of SVT in predicting the early time behavior of the flow generated by translating and rotating bodies with sharp and straight edges in a viscous fluid.

To demonstrate the stability analysis algorithm we proposed in Chapter 4, we applied it to study the flow past a rotating cylinder with a control cylinder in its wake. In this chapter, we fixed the location of the control cylinder and varied the rotational rate of the main cylinder. We found that while placing a control cylinder in the wake stabilizes the two-dimensional mode I instability, it slightly excites the three-dimensional mode II instability. In addition, we observe the formation of a different type of instability mechanisms appearing at rotational rates above 5.2. To further examine this phenomenon, we can also vary the location of the control cylinder to understand the physical mechanisms underlying the change in the stability characteristics.

The potential of the LGF method combined with the IB method and AMR shows great promise in solving a wide range of computational problems in fluid mechanics, as demonstrated in this thesis. However, there is still much work can be done to further leverage the benefits these methods can offer. For external incompressible flows that are spanwise periodic, turbulence models can be implemented to facilitate

a broader range of applications of the flow simulation algorithm proposed in this thesis. In addition, from a computational perspective, further optimization can be conducted to enhance the parallel efficiency of the flow simulation algorithm.

For the flow analysis algorithms, one can leverage the flexibility of the IB method to study flows around more complicated geometries, those that are much more complicated than the ones demonstrated in this thesis. Although only briefly mentioned in this thesis, the linear analysis algorithms are readily applicable to solve resolvent analysis problems. The accuracy and efficiency brought by the flow analysis algorithm proposed in this thesis can also be used to study the turbulent flows generated around immersed bodies at high Reynolds numbers.

BIBLIOGRAPHY

Abramowitz, Milton and Irene A. Stegun (1948). *Handbook of mathematical functions with formulas, graphs, and mathematical tables*. Vol. 55. US Government Printing Office.

Alben, Silas (2010). "Passive and active bodies in vortex-street wakes". In: *Journal of Fluid Mechanics* 642, pp. 95–125.

Amestoy, P. R. et al. (2001). "A Fully Asynchronous Multifrontal Solver Using Distributed Dynamic Scheduling". In: *SIAM Journal on Matrix Analysis and Applications* 23.1, pp. 15–41.

Anton, L. (1956). *Formation of a vortex at the edge of a flat plate*. Tech. rep. NACA Tech. Memo.

Anton, Leo (1939). "Ausbildung eines Wirbels an der Kante einer Platte". In: *Ingenieur-Archiv* 10.6, pp. 411–427.

Auerbach, David (1987). "Experiments on the trajectory and circulation of the starting vortex". In: *Journal of Fluid Mechanics* 183, pp. 185–198.

Bagheri, Shervin et al. (2009). "Global stability of a jet in crossflow". In: *Journal of Fluid Mechanics* 624, pp. 33–44.

Balay, Satish et al. (2022). *PETSc/TAO Users Manual*. Tech. rep. ANL-21/39 - Revision 3.17. Argonne National Laboratory.

Balty, Pierre, Philippe Chatelain, and Thomas Gillis (2023). "FLUPS-a flexible and performant massively parallel Fourier transform library". In: *IEEE Transactions on Parallel and Distributed Systems* 34.7, pp. 2011–2024.

Barnes, F.H. (2000). "Vortex shedding in the wake of a rotating circular cylinder at low Reynolds numbers". In: *Journal of Physics D: Applied Physics* 33.23, p. L141.

Barthel, Benedikt, Salvador Gomez, and Beverley J. McKeon (2022). "Variational formulation of resolvent analysis". In: *Physical Review Fluids* 7.1, p. 013905.

Berger, Marsha J. and Phillip Colella (1989). "Local adaptive mesh refinement for shock hydrodynamics". In: *Journal of Computational Physics* 82.1, pp. 64–84.

Berger, Marsha J. and Joseph Oliger (1984). "Adaptive mesh refinement for hyperbolic partial differential equations". In: *Journal of Computational Physics* 53.3, pp. 484–512.

Blackford, L. Susan et al. (1997). *ScaLAPACK users' guide*. SIAM.

Blendermann, Werner (1967). *Der Spiralwirbel am translatorisch bewegten Kreisbogenprofil: Struktur, Bewegung und Reaktion*. Institut für Schiffbau der Univ. Hamburg.

Borrell, Rick et al. (2011). “Parallel direct Poisson solver for discretisations with one Fourier diagonalisable direction”. In: *Journal of Computational Physics* 230.12, pp. 4723–4741.

Brasey, Valérie and Ernst Hairer (1993). “Half-explicit Runge–Kutta methods for differential-algebraic systems of index 2”. In: *SIAM Journal on Numerical Analysis* 30.2, pp. 538–552.

Brown, Shawn T. et al. (2021). “Bridges-2: a platform for rapidly-evolving and data intensive research”. In: *Practice and Experience in Advanced Research Computing*, pp. 1–4.

Buneman, O. (1971). “Analytic inversion of the five-point Poisson operator”. In: *Journal of Computational Physics* 8.3, pp. 500–505.

Burstedde, Carsten, Lucas C. Wilcox, and Omar Ghattas (2011). “p4est: Scalable algorithms for parallel adaptive mesh refinement on forests of octrees”. In: *SIAM Journal on Scientific Computing* 33.3, pp. 1103–1133.

Caprace, Denis-Gabriel, Thomas Gillis, and Philippe Chatelain (2021). “FLUPS: A Fourier-based library of unbounded Poisson solvers”. In: *SIAM Journal on Scientific Computing* 43.1, pp. C31–C60.

Colonius, Tim and Kunihiko Taira (2008). “A fast immersed boundary method using a nullspace approach and multi-domain far-field boundary conditions”. In: *Computer Methods in Applied Mechanics and Engineering* 197.25-28, pp. 2131–2146.

Cooley, James W. and John W. Tukey (1965). “An algorithm for the machine calculation of complex Fourier series”. In: *Mathematics of Computation* 19.90, pp. 297–301.

Courant, Richard, Kurt Friedrichs, and Hans Lewy (1928). “Über die partiellen Differenzengleichungen der mathematischen Physik”. In: *Mathematische Annalen* 100.1, pp. 32–74.

Cserti, József (2000). “Application of the lattice Green’s function for calculating the resistance of an infinite network of resistors”. In: *American Journal of Physics* 68.10, pp. 896–906.

Delves, Richard T. and Geoff S. Joyce (2001). “On the Green function for the anisotropic simple cubic lattice”. In: *Annals of Physics* 291.1, pp. 71–133.

Doncker, Elise de (1978). “An adaptive extrapolation algorithm for automatic integration”. In: *ACM SIGNUM Newsletter* 13.2, pp. 12–18.

Dong, Suchuan and George Em Karniadakis (2005). “DNS of flow past a stationary and oscillating cylinder at Re= 10000”. In: *Journal of Fluids and Structures* 20.4, pp. 519–531.

Dorschner, Benedikt et al. (2020). "A fast multi-resolution lattice Green's function method for elliptic difference equations". In: *Journal of Computational Physics* 407, p. 109270.

Duffin, R.J. (1953). "Discrete potential theory". In: *Duke Math. J.* 20.1, pp. 233–251.

Duffin, R.J. and E.P. Shelly (1958). "Difference equations of polyharmonic type". In: *Duke Math. J.* 25.1, pp. 209–238.

Eldredge, Jeff D. (2007). "Numerical simulation of the fluid dynamics of 2D rigid body motion with the vortex particle method". In: *Journal of Computational Physics* 221.2, pp. 626–648.

Fang, Fang et al. (2017). "A computational model of the flight dynamics and aerodynamics of a jellyfish-like flying machine". In: *Journal of Fluid Mechanics* 819, pp. 621–655.

Fischer, Paul, James Lottes, and Henry Tufo (2007). *Nek5000*. Tech. rep. Argonne National Lab.(ANL), Argonne, IL (United States).

Frigo, Matteo and Steven G. Johnson (1998). "FFTW: An adaptive software architecture for the FFT". In: *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181)*. Vol. 3. IEEE, pp. 1381–1384.

Gabbard, James, Thomas Gillis, et al. (2022). "An immersed interface method for the 2D vorticity-velocity Navier-Stokes equations with multiple bodies". In: *Journal of Computational Physics* 464, p. 111339.

Gabbard, James and Wim M. van Rees (2024a). "A high-order finite difference method for moving immersed domain boundaries and material interfaces". In: *Journal of Computational Physics* 507, p. 112979.

– (2024b). "Lattice Green's Functions for High-Order Finite Difference Stencils". In: *SIAM Journal on Numerical Analysis* 62.1, pp. 25–47.

Giannetti, Flavio and Paolo Luchini (2007). "Structural sensitivity of the first instability of the cylinder wake". In: *Journal of Fluid Mechanics* 581, pp. 167–197.

Gillis, Thomas and Wim M. Van Rees (2022). "MURPHY—A Scalable Multiresolution Framework for Scientific Computing on 3D Block-Structured Collocated Grids". In: *SIAM Journal on Scientific Computing* 44.5, pp. C367–C398.

Gillman, Adrianna and Per-Gunnar Martinsson (2010). "Fast and accurate numerical methods for solving elliptic difference equations defined on lattices". In: *Journal of Computational Physics* 229.24, pp. 9026–9041.

Harris, Charles R. et al. (Sept. 2020). "Array programming with NumPy". In: *Nature* 585.7825, pp. 357–362. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2.

Hernandez, Vicente, Jose E. Roman, and Vicente Vidal (2005). "SLEPc: A scalable and flexible toolkit for the solution of eigenvalue problems". In: *ACM Transactions on Mathematical Software (TOMS)* 31.3, pp. 351–362.

Heydari, Sina and Eva Kanso (2021). "School cohesion, speed and efficiency are modulated by the swimmers flapping motion". In: *Journal of Fluid Mechanics* 922, A27.

Hinton, Edward M. et al. (2024). "Starting vortices generated by an arbitrary solid body with any number of edges". In: *Journal of Fluid Mechanics* 987, A11.

Hosseini, Seyed Mohammad et al. (2016). "Direct numerical simulation of the flow around a wing section at moderate Reynolds number". In: *International Journal of Heat and Fluid Flow* 61, pp. 117–128.

Hou, Wei and Tim Colonius (2023). "Three-dimensional stability and resolvent analysis of external flows over spanwise-homogeneous immersed bodies". In: *AIAA AVIATION 2023 Forum*, p. 3414. URL: https://arc.aiaa.org/doi/10.2514/6.2023-3414.

– (Dec. 2024a). "An adaptive lattice Green's function method for external flows with two unbounded and one homogeneous directions". In: *Journal of Computational Physics* 519, p. 113370. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2024.113370. URL: http://dx.doi.org/10.1016/j.jcp.2024.113370.

– (2024b). "Fast and robust method for screened Poisson lattice Green's function using asymptotic expansion and Fast Fourier Transform". In: *arXiv preprint arXiv:2403.03076*. URL: https://arxiv.org/abs/2403.03076.

Ji, Xinjie, James Gabbard, and Wim M. van Rees (2023). "A sharp immersed method for 2D flow-body interactions using the vorticity-velocity Navier-Stokes equations". In: *Journal of Computational Physics* 494, p. 112513.

Jin, Bo, Sean Symon, and Simon J. Illingworth (2021). "Energy transfer mechanisms and resolvent analysis in the cylinder wake". In: *Physical Review Fluids* 6.2, p. 024702.

Jones, Marvin A. (2003). "The separated flow of an inviscid fluid around a moving flat plate". In: *Journal of Fluid Mechanics* 496, pp. 405–441.

Kaden, Heinrich (1931). "Aufwicklung einer unstabilen Unstetigkeitsfläche". In: *Ingenieur-Archiv* 2.2, pp. 140–168.

Kamal, Omar, Matthew T. Lakebrink, and Tim Colonius (2023). "Global receptivity analysis: physically realizable input–output analysis". In: *Journal of Fluid Mechanics* 956, R5.

Katsura, Shigetoshi and Sakari Inawashiro (1971). "Lattice Green's functions for the rectangular and the square lattices at arbitrary points". In: *Journal of Mathematical Physics* 12.8, pp. 1622–1630.

Katsura, Shigetoshi and Sakari Inawashiro (1973). “Asymptotic form of the lattice Green’s function of the square lattice”. In: *Annals of Physics* 77.1-2, pp. 206–215.

Kazhdan, Michael and Hugues Hoppe (2013). “Screened Poisson surface reconstruction”. In: *ACM Transactions on Graphics (ToG)* 32.3, pp. 1–13.

Koster, G.F. and J.C. Slater (1954). “Simplified impurity calculation”. In: *Physical Review* 96.5, p. 1208.

Kotera, Takeyasu (1962). “Localized vibration and random walk”. In: *Progress of Theoretical Physics Supplement* 23, pp. 141–156.

Koumoutsakos, P. and D. Shiels (1996). “Simulations of the viscous flow normal to an impulsively started and uniformly accelerated flat plate”. In: *Journal of Fluid Mechanics* 328, pp. 177–227.

Krasny, Robert (1991). “Vortex sheet computations: roll-up, wakes, separation”. In: *Lectures in Applied Mathematics* 28.1, pp. 385–401.

Kravchenko, Arthur G., Parviz Moin, and Karim Shariff (1999). “B-spline method and zonal grids for simulations of complex turbulent flows”. In: *Journal of Computational Physics* 151.2, pp. 757–789.

Lai, Ming-Chih and Charles S. Peskin (2000). “An immersed boundary method with formal second-order accuracy and reduced numerical viscosity”. In: *Journal of computational Physics* 160.2, pp. 705–719.

Lawler, Gregory F. and Vlada Limic (2010). *Random walk: a modern introduction*. Vol. 123. Cambridge University Press.

Lehmkuhl, O. et al. (2013). “Low-frequency unsteadiness in the vortex formation region of a circular cylinder”. In: *Physics of Fluids* 25.8.

Liska, Sebastian and Tim Colonius (2014). “A parallel fast multipole method for elliptic difference equations”. In: *Journal of Computational Physics* 278, pp. 76–91.

– (2016). “A fast lattice Green’s function method for solving viscous incompressible flows on unbounded domains”. In: *Journal of Computational Physics* 316, pp. 360–384.

– (2017). “A fast immersed boundary method for external incompressible viscous flows using lattice Green’s functions”. In: *Journal of Computational Physics* 331, pp. 257–279.

Luchini, Paolo and Renato Tognaccini (2002). “The start-up vortex issuing from a semi-infinite flat plate”. In: *Journal of Fluid Mechanics* 455, pp. 175–193.

– (2017). “Viscous and inviscid simulations of the start-up vortex”. In: *Journal of Fluid Mechanics* 813, pp. 53–69.

Maassarani, Z. (2000). “Series expansions for lattice Green functions”. In: *Journal of Physics A: Mathematical and General* 33.32, p. 5675.

Madras, Neal (1989). “Random walks with killing”. In: *Probability Theory and Related Fields* 80, pp. 581–600.

Maradudin, Alexei A. (1960). *Green’s functions for monatomic simple cubic lattices*. Vol. 2. na.

Martini, Eduardo et al. (2021). “Efficient computation of global resolvent modes”. In: *Journal of Fluid Mechanics* 919, A3.

Martinsson, Per-Gunnar and Gregory J. Rodin (2002). “Asymptotic expansions of lattice Green’s functions”. In: *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 458.2027, pp. 2609–2622.

McKeon, Beverley J. and Ati S. Sharma (2010). “A critical-layer framework for turbulent pipe flow”. In: *Journal of Fluid Mechanics* 658, pp. 336–382.

Michelin, Sébastien and Stefan G Llewellyn Smith (2009). “An unsteady point vortex method for coupled fluid–solid problems”. In: *Theoretical and Computational Fluid Dynamics* 23, pp. 127–153.

Michta, Emmanuel and Gordon Slade (2021). “Asymptotic behaviour of the lattice Green function”. In: *arXiv preprint arXiv:2101.04717*.

Mittal, R. and S. Balachandar (1995). “Effect of three-dimensionality on the lift and drag of nominally two-dimensional cylinders”. In: *Physics of Fluids* 7.8, pp. 1841–1865.

– (1996). “Direct numerical simulation of flow past elliptic cylinders”. In: *Journal of computational physics* 124.2, pp. 351–367.

– (1997). “On the inclusion of three-dimensional effects in simulations of two-dimensional bluff-body wake flows”. In: *ASME Summer Meeting*. ASME, pp. 1–6.

Mori, Yoichiro (2008). “Convergence proof of the velocity field for a Stokes flow immersed boundary method”. In: *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences* 61.9, pp. 1213–1263.

Morita, Tohru and Tsuyoshi Horiguchi (1971). “Calculation of the lattice Green’s function for the bcc, fcc, and rectangular lattices”. In: *Journal of Mathematical Physics* 12.6, pp. 986–992.

The mpmath development team (2023). *mpmath: a Python library for arbitrary-precision floating-point arithmetic (version 1.3.0)*. `http://mpmath.org/`.

Nitsche, Monika and Ling Xu (2014). “Circulation shedding in viscous starting flow past a flat plate”. In: *Fluid Dynamics Research* 46.6, p. 061420.

Norberg, Christoffer (2003). “Fluctuating lift on a circular cylinder: review and new measurements”. In: *Journal of Fluids and Structures* 17.1, pp. 57–96.

NVIDIA Corporation (2024). *cuFFT*. https://developer.nvidia.com/cufft. Version 12.3.

Offermans, Nicolas et al. (2020). "Adaptive mesh refinement for steady flows in Nek5000". In: *Computers & Fluids* 197, p. 104352.

Orszag, Steven A. (1971a). "Accurate solution of the Orr–Sommerfeld stability equation". In: *Journal of Fluid Mechanics* 50.4, pp. 689–703.

– (1971b). "Numerical simulation of incompressible flows within simple boundaries. I. Galerkin (spectral) representations". In: *Studies in Applied Mathematics* 50.4, pp. 293–327.

Panton, Ronald L. (2024). *Incompressible flow*. John Wiley & Sons.

Peskin, Charles S. (2002). "The immersed boundary method". In: *Acta Numerica* 11, pp. 479–517.

Pierce, D. (1961). "Photographic evidence of the formation and growth of vorticity behind plates accelerated from rest in still air". In: *Journal of Fluid Mechanics* 11.3, pp. 460–464.

Popinet, Stéphane (2003). "Gerris: a tree-based adaptive solver for the incompressible Euler equations in complex geometries". In: *Journal of computational physics* 190.2, pp. 572–600.

Pralits, Jan O., Luca Brandt, and Flavio Giannetti (2010). "Instability and sensitivity of the flow around a rotating circular cylinder". In: *Journal of Fluid Mechanics* 650, pp. 513–536.

Pralits, Jan O., Flavio Giannetti, and Luca Brandt (2013). "Three-dimensional instability of the flow around a rotating circular cylinder". In: *Journal of Fluid Mechanics* 730, pp. 5–18.

Prandtl, L. (1924). "Über die Entstehung von Wirbeln in der idealen Flüssigkeit, mit Anwendung auf die Tragflügeltheorie und andere Aufgaben". In: *Vorträge aus dem Gebiete der Hydro-und Aerodynamik (Innsbruck 1922)*, pp. 18–33.

Pullin, D.I. (1978). "The large-scale structure of unsteady self-similar rolled-up vortex sheets". In: *Journal of Fluid Mechanics* 88.3, pp. 401–430.

Pullin, D.I. and A.E. Perry (1980). "Some flow visualization experiments on the starting vortex". In: *Journal of Fluid Mechanics* 97.2, pp. 239–255.

Pullin, D.I. and John E. Sader (2021). "On the starting vortex generated by a translating and rotating flat plate". In: *Journal of Fluid Mechanics* 906, A9.

Rai, Man Mohan (2013). "Flow physics in the turbulent near wake of a flat plate". In: *Journal of Fluid Mechanics* 724, pp. 704–733.

Ribeiro, Jean Hélder Marques, Chi-An Yeh, and Kunihiko Taira (2020). "Randomized resolvent analysis". In: *Physical Review Fluids* 5.3, p. 033902.

Roma, Alexandre M., Charles S. Peskin, and Marsha J. Berger (1999). "An adaptive version of the immersed boundary method". In: *Journal of Computational Physics* 153.2, pp. 509–534.

Rott, Nicholas (1956). "Diffraction of a weak shock with vortex generation". In: *Journal of Fluid Mechanics* 1.1, pp. 111–128.

Sader, John E. et al. (2024). "The starting vortices generated by bodies with sharp and straight edges in a viscous fluid". In: *Journal of Fluid Mechanics* 992, A15. URL: https://doi.org/10.1017/jfm.2024.515.

Schäling, Boris (2011). *The boost C++ libraries*. Boris Schäling.

Schneider, Kai and Oleg V. Vasilyev (2010). "Wavelet methods in computational fluid dynamics". In: *Annual review of fluid mechanics* 42.1, pp. 473–503.

Smith, F.T. (1986). "Steady and unsteady boundary-layer separation". In: *Annual Review of Fluid Mechanics* 18.1, pp. 197–220.

Stampede3 User Guide (n.d.). https://docs.tacc.utexas.edu/hpc/stampede3/. Accessed: 2024-08-02.

Stanzione, Dan et al. (2017). "Stampede 2: The evolution of an XSEDE supercomputer". In: *Proceedings of the Practice and Experience in Advanced Research Computing 2017 on Sustainability, Success and Impact*, pp. 1–8.

Steger, Joseph L., F. Carroll Dougherty, and John A. Benek (1983). "A Chimera grid scheme". In.

Stojković, Dragan, Michael Breuer, and Franz Durst (2002). "Effect of high rotation rates on the laminar flow around a circular cylinder". In: *Physics of fluids* 14.9, pp. 3160–3178.

Strykowski, P.J. and K. R. Sreenivasan (1990). "On the formation and suppression of vortex 'shedding' at low Reynolds numbers". In: *Journal of Fluid Mechanics* 218, pp. 71–107.

Symon, Sean Pearson (2018). "Reconstruction and estimation of flows using resolvent analysis and data-assimilation". PhD thesis. California Institute of Technology.

Taira, Kunihiko, Steven L Brunton, et al. (2017). "Modal analysis of fluid flows: An overview". In: *AIAA Journal* 55.12, pp. 4013–4041.

Taira, Kunihiko and Tim Colonius (2007). "The immersed boundary method: a projection approach". In: *Journal of Computational Physics* 225.2, pp. 2118–2137.

Tang, H.S., S. Casey Jones, and Fotis Sotiropoulos (2003). "An overset-grid method for 3D unsteady incompressible flows". In: *Journal of Computational Physics* 191.2, pp. 567–600.

Theofilis, Vassilios (2011). "Global linear instability". In: *Annual Review of Fluid Mechanics* 43, pp. 319–352.

Tornberg, Anna-Karin and Björn Engquist (2004). "Numerical approximations of singular source terms in differential equations". In: *Journal of Computational Physics* 200.2, pp. 462–488.

Towne, Aaron, Oliver T. Schmidt, and Tim Colonius (2018). "Spectral proper orthogonal decomposition and its relationship to dynamic mode decomposition and resolvent analysis". In: *Journal of Fluid Mechanics* 847, pp. 821–867.

Trefethen, Lloyd N. and David Bau (2022). *Numerical linear algebra*. Vol. 181. SIAM.

Trefethen, Lloyd N., Anne E. Trefethen, et al. (1993). "Hydrodynamic stability without eigenvalues". In: *Science* 261.5121, pp. 578–584.

Trefethen, Lloyd N. and J.A.C. Weideman (2014). "The exponentially convergent trapezoidal rule". In: *SIAM Review* 56.3, pp. 385–458.

Verzicco, Roberto (2023). "Immersed boundary methods: Historical perspective and future outlook". In: *Annual Review of Fluid Mechanics* 55.1, pp. 129–155.

Virtanen, Pauli et al. (2020). "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17, pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

Wagner, Herbert (1924). "Über die Entstehung des dynamischen Auftriebes von Tragflügeln". In.

Wedemeyer, E. (1961). "Ausbildung eines Wirbelpaares an den Kanten einer Platte". In: *Ingenieur-Archiv* 30.3, pp. 187–200.

Wieselsberger, Carl (1922). *Further information on the laws of fluid resistance*. Tech. rep.

Williamson, Charles H.K. (1996). "Vortex dynamics in the cylinder wake". In: *Annual Review of Fluid Mechanics* 28.1, pp. 477–539.

Williamson, Charles H.K. and Anatol Roshko (1988). "Vortex formation in the wake of an oscillating cylinder". In: *Journal of Fluids and Structures* 2.4, pp. 355–381.

Xu, Ling and Monika Nitsche (2015). "Start-up vortex flow past an accelerated flat plate". In: *Physics of Fluids* 27.3.

Xu, Ling, Monika Nitsche, and Robert Krasny (2017). "Computation of the starting vortex flow past a flat plate". In: *Procedia IUTAM* 20, pp. 136–143.

Yeh, Chi-An and Kunihiko Taira (2019). "Resolvent-analysis-based design of airfoil separation control". In: *Journal of Fluid Mechanics* 867, pp. 572–610.

Ying, Lexing, George Biros, and Denis Zorin (2004). "A kernel-independent adaptive fast multipole algorithm in two and three dimensions". In: *Journal of Computational Physics* 196.2, pp. 591–626.

Yu, Ke (2021). “Multi-resolution Lattice Green’s Function Method for High Reynolds Number External Flows”. PhD thesis. California Institute of Technology.

Yu, Ke, Benedikt Dorschner, and Tim Colonius (2022). “Multi-resolution lattice Green’s function method for incompressible flows”. In: *Journal of Computational Physics* 459, p. 110845.

Appendix A

IB TERMS FOR FOURIER COEFFICIENTS

In this part, we provide a few details regarding the Fourier transform of Eq. 3.3 leading to Eq. 3.5. The only terms that are nontrivial are the Fourier transforms of the IB forcing term and the no-slip boundary condition.

We first show the derivations of the IB forcing term. Consider the immersed boundary defined, $\Gamma(t)$. The geometry of the immersed boundary is assumed to be the extrusion of a two-dimensional boundary. Thus, we can denote that said two-dimensional boundary as $\Gamma_{2D}(t)$. With this, we can write:

$$\Gamma(t) = \Gamma_{2D}(t) \times [-c/2, c/2]. \tag{A.1}$$

Consequently, we can find some $\boldsymbol{\xi}_{2D}$ that parameterizes $\Gamma_{2D}(t)$ for all t. Correspondingly, the parameterization at time t is defined as $\boldsymbol{X}_{2D}(\boldsymbol{\xi}_{2D}, t)$. Thus, following the notation of Eq. 3.5, we can write

$$\boldsymbol{X}(\boldsymbol{\xi}, t) = (\boldsymbol{X}_{2D}(\boldsymbol{\xi}_{2D}, t), Z), \quad \boldsymbol{\xi} = (\boldsymbol{\xi}_{2D}, Z). \tag{A.2}$$

Then with $\boldsymbol{x} = (\boldsymbol{x}_{2D}, z)$, we can write the IB forcing term at $\boldsymbol{x}$ as:

$$\begin{aligned}
&\int_{\Gamma(t)} \boldsymbol{f}_{\Gamma}(\boldsymbol{\xi}, t)\delta(\boldsymbol{X}(\boldsymbol{\xi}, t) - \boldsymbol{x})d\boldsymbol{\xi} \\
&= \int_{-c/2}^{c/2} \int_{\Gamma_{2D}(t)} \boldsymbol{f}_{\Gamma}(\boldsymbol{\xi}_{2D}, Z, t)\delta_{2D}(\boldsymbol{X}_{2D}(\boldsymbol{\xi}, t) - \boldsymbol{x}_{2D})\delta_{1D}(z - Z)d\boldsymbol{\xi}_{2D}dZ
\end{aligned}$$

where δ_{1D} and δ_{2D} denote the Delta functions in 1D spaces and 2D spaces, respec-

tively. With the above notation, we write:

$$
\begin{aligned}
&\mathcal{F}_k\left[\int_{\Gamma(t)} \boldsymbol{f}_\Gamma(\boldsymbol{\xi},t)\delta(\boldsymbol{X}(\boldsymbol{\xi},t)-\boldsymbol{x})d\boldsymbol{\xi}\right]\\
=&\frac{1}{c}\int_{-c/2}^{c/2} e^{-iz\frac{2\pi k}{c}}\int_{\Gamma(t)} \boldsymbol{f}_\Gamma(\boldsymbol{\xi},t)\delta(\boldsymbol{X}(\boldsymbol{\xi},t)-\boldsymbol{x})d\boldsymbol{\xi}dz\\
=&\frac{1}{c}\int_{\Gamma(t)}\int_{-c/2}^{c/2} e^{-iz\frac{2\pi k}{c}} \boldsymbol{f}_\Gamma(\boldsymbol{\xi},t)\delta(\boldsymbol{X}(\boldsymbol{\xi},t)-\boldsymbol{x})dzd\boldsymbol{\xi}\\
=&\frac{1}{c}\int_{\Gamma_{2D}(t)}\int_{-c/2}^{c/2}\int_{-c/2}^{c/2} e^{-iz\frac{2\pi k}{c}} \boldsymbol{f}_\Gamma(\boldsymbol{\xi}_{2D},Z,t)\delta_{2D}(\boldsymbol{X}_{2D}(\boldsymbol{\xi}_{2D},t)-\boldsymbol{x}_{2D})\delta_{1D}(z-Z)dZdzd\boldsymbol{\xi}_{2D}\\
=&\frac{1}{c}\int_{\Gamma_{2D}(t)}\int_{-c/2}^{c/2} e^{-iz\frac{2\pi k}{c}} \boldsymbol{f}_\Gamma(\boldsymbol{\xi}_{2D},z,t)\delta_{2D}(\boldsymbol{X}_{2D}(\boldsymbol{\xi}_{2D},t)-\boldsymbol{x}_{2D})dzd\boldsymbol{\xi}_{2D}\\
=&\int_{\Gamma_{2D}(t)} \tilde{\boldsymbol{f}}_{\Gamma,k}(\boldsymbol{\xi}_{2D},t)\delta_{2D}(\boldsymbol{X}_{2D}(\boldsymbol{\xi}_{2D},t)-\boldsymbol{x}_{2D})d\boldsymbol{\xi}_{2D},
\end{aligned}
\tag{A.3}
$$

where $\tilde{f}_{\Gamma,k}$ is the k^{th} Fourier coefficient of the IB forcing f_Γ.

For the boundary condition, we apply $\mathcal{F}_k$ directly. The LHS, by definition, is:

$$
\begin{aligned}
&\mathcal{F}_k\left[\boldsymbol{u}_\Gamma(\boldsymbol{\xi},t)\right]\\
=&\frac{1}{c}\int_{-c/2}^{c/2} e^{-iZ\frac{2\pi k}{c}}\boldsymbol{u}_\Gamma(\boldsymbol{\xi},t)dZ\\
=&\frac{1}{c}\int_{-c/2}^{c/2} e^{-iZ\frac{2\pi k}{c}}\boldsymbol{u}_\Gamma(\boldsymbol{\xi}_{2D},Z,t)dZ\\
=&\widetilde{\boldsymbol{u}}_{\Gamma,k}(\boldsymbol{\xi}_{2D},t).
\end{aligned}
\tag{A.4}
$$

The RHS is:

$$
\begin{aligned}
&\mathcal{F}_k\left[\int_{\mathbb{R}^3} \boldsymbol{u}(\boldsymbol{x},t)\delta(\boldsymbol{x}-\boldsymbol{X}(\boldsymbol{\xi},t))d\boldsymbol{x}\right]\\
=&\frac{1}{c}\int_{-c/2}^{c/2} e^{-iZ\frac{2\pi k}{c}}\int_{\mathbb{R}^3} \boldsymbol{u}(\boldsymbol{x},t)\delta(\boldsymbol{x}-\boldsymbol{X}(\boldsymbol{\xi},t))d\boldsymbol{x}dZ\\
=&\frac{1}{c}\int_{-c/2}^{c/2} e^{-iZ\frac{2\pi k}{c}}\int_{\mathbb{R}}\int_{\mathbb{R}^2} \boldsymbol{u}(\boldsymbol{x}_{2D},z,t)\delta_{2D}(\boldsymbol{x}_{2D}-\boldsymbol{X}_{2D}(\boldsymbol{\xi}_{2D},t))\delta_{1D}(z-Z)d\boldsymbol{x}_{2D}dzdZ\\
=&\frac{1}{c}\int_{-c/2}^{c/2} e^{-iZ\frac{2\pi k}{c}}\int_{\mathbb{R}^2}\int_{\mathbb{R}} \boldsymbol{u}(\boldsymbol{x}_{2D},z,t)\delta_{2D}(\boldsymbol{x}_{2D}-\boldsymbol{X}_{2D}(\boldsymbol{\xi}_{2D},t))\delta_{1D}(z-Z)dzd\boldsymbol{x}_{2D}dZ\\
=&\frac{1}{c}\int_{-c/2}^{c/2} e^{-iZ\frac{2\pi k}{c}}\int_{\mathbb{R}^2} \boldsymbol{u}(\boldsymbol{x}_{2D},Z,t)\delta_{2D}(\boldsymbol{x}_{2D}-\boldsymbol{X}_{2D}(\boldsymbol{\xi}_{2D},t))d\boldsymbol{x}_{2D}dZ\\
=&\int_{\mathbb{R}^2}\left[\frac{1}{c}\int_{-c/2}^{c/2} e^{-iZ\frac{2\pi k}{c}}\boldsymbol{u}(\boldsymbol{x}_{2D},Z,t)dZ\right]\delta_{2D}(\boldsymbol{x}_{2D}-\boldsymbol{X}_{2D}(\boldsymbol{\xi}_{2D},t))d\boldsymbol{x}_{2D}\\
=&\int_{\mathbb{R}^2}\tilde{\boldsymbol{u}}_k(\boldsymbol{x}_{2D},t)\delta_{2D}(\boldsymbol{x}_{2D}-\boldsymbol{X}_{2D}(\boldsymbol{\xi}_{2D},t))d\boldsymbol{x}_{2D}.
\end{aligned}
\tag{A.5}
$$

Then, the boundary condition becomes:

$$
\widetilde{\boldsymbol{u}}_{\Gamma,k}(\boldsymbol{\xi}_{2D},t)=\int_{\mathbb{R}^2}\tilde{\boldsymbol{u}}_k(\boldsymbol{x}_{2D},t)\delta_{2D}(\boldsymbol{x}_{2D}-\boldsymbol{X}_{2D}(\boldsymbol{\xi}_{2D},t))d\boldsymbol{x}_{2D}. \tag{A.6}
$$

Appendix B

COMPATIBILITY CONDITION ON L_0^{-1} IN A MULTILEVEL MESH

Consider two different uniform Cartesian meshes with different resolutions Δx_1 and Δx_2 and are governed by the relationship $\Delta x_1 = N\Delta x_2$, where N a positive integer. Denote the B_0 kernel for the two grid as B_0^1 and B_0^2. The compatibility condition we impose is:

$$\lim_{|\boldsymbol{n}|\to\infty} (B_0^1(\boldsymbol{n}) - B_0^2(\boldsymbol{n}N)) = 0. \tag{B.1}$$

Since $\boldsymbol{n}\Delta x_1 = \boldsymbol{n}N\Delta x_2$, this condition means that, if B_0^1 and B_0^2 are two different discrete solutions of a discretized 2D Poisson equation induced by the Dirac delta function, both solutions should match at infinity in the physical space.

Using the asymptotic expansion of LGF (Martinsson and Rodin, 2002), we can write:

$$B_0(\boldsymbol{n}) = \frac{1}{2\pi}\log(|\boldsymbol{n}|) + C + O(1/|\boldsymbol{n}|^2). \tag{B.2}$$

Thus, by plugging in this expansion to Eq. B.1, we get:

$$\begin{aligned}
0 &= \lim_{|\boldsymbol{n}|\to\infty} (B_0^1(\boldsymbol{n}) - B_0^2(\boldsymbol{n}N)) \\
&= \lim_{|\boldsymbol{n}|\to\infty} (\frac{1}{2\pi}\log(|\boldsymbol{n}|) + C_1 + O(1/|\boldsymbol{n}|^2) - \frac{1}{2\pi}\log(N|\boldsymbol{n}|) - C_2 - O(1/|N\boldsymbol{n}|^2)) \\
&= \lim_{|\boldsymbol{n}|\to\infty} (-\frac{1}{2\pi}\log(N) + C_1 - C_2 + O(1/|\boldsymbol{n}|^2)) \\
&= -\frac{1}{2\pi}\log(N) + C_1 - C_2.
\end{aligned} \tag{B.3}$$

Thus, we obtain

$$C_1 - C_2 = \frac{1}{2\pi}\log(N). \tag{B.4}$$

That is:

$$B_0^1(\boldsymbol{n}) = B_0^2(\boldsymbol{n}) + \frac{1}{2\pi}\log(N). \tag{B.5}$$

In the context of our algorithm, we have $N = 2^l$ for some non-negative integer l. As a result, we have:

$$B_0^1(\boldsymbol{n}) = B_0^2(\boldsymbol{n}) + \frac{l}{2\pi}\log(2). \tag{B.6}$$

Appendix C

$S^i_{k,n}$ IS HERMITIAN WHEN $P^i_n = P^{i-1}_n$

Here we show that $(S^i_{k,n})^* = S^i_{k,n}$ under the assumption of $P^i_n = P^{i-1}_n = P_n$. In this case, we have $S^i_{k,n} = P_n E^i_k (I - G_k L_k^{-1} D_k) P_n^T$. Since $E^i_{k,n}$ commutes with all the discrete operators and L_k^{-1}, we have:

$$\begin{aligned}(S^i_{k,n})^* &= \left[P_n E^i_k (I - G_k L_k^{-1} D_k) P_n^T\right]^* \\ &= (P_n^T)^* (E^i_k)^* (I - G_k L_k^{-1} D_k)^* P_n^*.\end{aligned} \tag{C.1}$$

Since P_n is purely real, $P_n^* = P_n^T$. In addition, we know that $(E^i_k)^* = E^i_k$, $(L_k^{-1})^* = L_k^{-1}$. As a result, we can write the equation above as:

$$(P_n^T)^* (E^i_k)^* (I - G_k L_k^{-1} D_k)^* P_n^* = P_n E^i_k (I - D_k^* L_k^{-1} G_k^*) P_n^T. \tag{C.2}$$

Now we recall the mimetic property $D_k^* = -G_k$, and we obtain that:

$$P_n E^i_k (I - D_k^* L_k^{-1} G_k^*) P_n^T = P_n E^i_k (I - G_k L_k^{-1} D_k) P_n^T. \tag{C.3}$$

Combing all the steps together, we have shown that:

$$\begin{aligned}(S^i_{k,n})^* &= \left[P_n E^i_k (I - G_k L_k^{-1} D_k) P_n^T\right]^* \\ &= (P_n^T)^* (E^i_k)^* (I - G_k L_k^{-1} D_k)^* P_n^* \\ &= P_n E^i_k (I - D_k^* L_k^{-1} G_k^*) P_n^T \\ &= P_n E^i_k (I - G_k L_k^{-1} D_k) P_n^T \\ &= S^i_{k,n}.\end{aligned} \tag{C.4}$$

Thus, we conclude that $S^i_{k,n}$ is Hermitian.

Appendix D

ADDITIONAL TIME CONSTRAINT FOR THE EXISTENCE OF TYPE I AND II VORTICES GENERATED BY A FLAT PLATE

In this appendix, we derive the maximal time for Type I and II vortices of a flat plate to exist, which supplements the time constraint in (5.13).

For Type-I vortices, the second term on the right-hand side of (5.10) must dominate the first, i.e.,

$$i\hat{\beta}\,T^{p-\frac{q}{2}}\frac{1+M(Z)}{(2Z)^{\frac{1}{2}}} \gg T^{m}, \tag{D.1}$$

where $q = 2(p+1)/3$, as per Table 1 of PS21. Balancing the left and right-hand sides of (5.10), gives $Z = O(\hat{\beta}^{\frac{2}{3}})$, and the inequality in (D.1) becomes

$$T^{\epsilon} \ll \hat{\beta}^{\frac{2}{3}}, \tag{D.2}$$

where

$$\epsilon \equiv m - \frac{2p-1}{3}, \tag{D.3}$$

which is strictly positive for a Type-I vortex, i.e., $\epsilon > 0$; see (5.6) and Figure 5.1.

Type-II vortices occur in the opposite regime, i.e.,

$$i\hat{\beta}\,T^{p-\frac{q}{2}}\frac{1+M(Z)}{(2Z)^{\frac{1}{2}}} \ll T^{m}, \tag{D.4}$$

where $q = 1 + m$. Balancing the left and right-hand sides of (5.10), shows that $Z = O(1)$, from which (D.4) becomes

$$T^{-\frac{3}{2}\epsilon} \ll \hat{\beta}^{-1}, \tag{D.5}$$

where $\epsilon < 0$ for Type-II vortices.

Combining (D.2) and (D.5) produces

$$T^{|\epsilon|} \ll \hat{\beta}^{\frac{2}{3}\mathrm{sgn}(\epsilon)}, \tag{D.6}$$

where $\epsilon \neq 0$; note that $\epsilon = 0$ corresponds to a Type-III vortex sheet, which is not considered in this appendix. The time constraint in (D.6) is in addition to (5.12), and the overarching spatial constraint that the vortex remains close to the trailing edge.

Choosing the same multiplicative factor as (5.13) for the inequality in (D.6) to hold, i.e., $0 < \delta \ll 1$, gives the additional maximal time for existence of Type I and II vortices,

$$T_{\max}^{(2)} \approx \left(\hat{\beta}^{\frac{2}{3}\mathrm{sgn}(\epsilon)}\delta\right)^{\frac{1}{|\epsilon|}}, \tag{D.7}$$

where it is understood that $\hat{\beta}$ is finite and the parenthesized term is less than unity. This shows that Type I and II vortices, corresponding to $\epsilon > 0$ and $\epsilon < 0$, respectively, are more easily observed for large and small, $\hat{\beta}$, respectively. In Section 5.3, we combine (D.7) with (5.13) to give the overall maximal time for existence of any starting vortex.