© 2019 by Ketan Mittal. All rights reserved.

HIGHLY SCALABLE SOLUTION OF INCOMPRESSIBLE NAVIER-STOKES EQUATIONS USING THE SPECTRAL ELEMENT METHOD WITH OVERLAPPING GRIDS

BY

KETAN MITTAL

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Mechanical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2019

Urbana, Illinois

Doctoral Committee:

Professor Paul Fischer, Chair
Professor Arne Pearlstein
Professor Moshe Matalon
Associate Professor Andreas Kloeckner

Abstract

We present a highly-flexible Schwarz overlapping framework for simulating turbulent fluid/thermal transport in complex domains. The approach is based on a variant of the Schwarz alternating method in which the solution is advanced in parallel in separate overlapping subdomains. In each domain, the governing equations are discretized with an efficient high-order spectral element method (SEM). At each step, subdomain boundary data are determined by interpolating from the overlapping region of adjacent subdomains. The data are either lagged in time or extrapolated to higher-order temporal accuracy using a novel stabilized predictor-corrector algorithm. Matrix stability analysis is used to determine the optimal number of corrector iterations. Stability and accuracy are further improved with an optimal mass flux correction to guarantee mass conservation throughout the domain. The method supports an arbitrary number of subdomains. A new multirate time-stepping scheme is developed (a first for incompressible flow simulations) that allows the underlying equations to be advanced with time-step sizes varying as much as an order-of-magnitude between adjacent domains. All the developments maintain the third-order temporal convergence and exponential convergence of the originating SEM framework. This dissertation also presents a mesh optimizer that has been specifically designed for meshes generated for turbulent flow problems. The optimizer supports surface mesh improvement, which minimizes geometrical approximation errors. The smoother is shown to reduce the computational cost of numerical calculations by as much as 40%. Numerous examples illustrate the effectiveness of these new technologies for analyzing challenging turbulence problems that were previously infeasible.

To my parents for their support and love.

Acknowledgments

I would like to express sincere gratitude to Prof. Fischer for patiently guiding me through graduate studies over the last four years. He has continuously supported and helped me in developing problem-solving skills that will serve me throughout my professional career. The intellectual freedom that Prof. Fischer gave me in pursuing research in the area of my interest was vital for my continued enthusiasm during this journey.

I am also grateful to my Ph.D. committee for insightful comments on my preliminary thesis proposal and the final dissertation. Their feedback has been instrumental in improving my approach to writing and presenting my research and has shaped my critical thinking as a researcher. I am also thankful to all the teachers, with whom I have taken various classes over the years.

My Ph.D. journey at the University of Illinois has also been successful due to continued support from peers and mentors whom I have interacted with. I am forever indebted to Som Dutta for mentoring me over the years and being available for discussions whenever I needed help. Som has been an invaluable source of motivation during some of the toughest times that I faced as a Ph.D. student. I would like to thank Li, Kento, Dimitrios, Nick, Pedro, and Thilina for providing me insight whenever I reached out to discuss problems in their area of expertise. I have also learned a great deal from everyone who attended the weekly Nek5000 user meetings at UIUC.

Finally, I would like to thank my friends and family for supporting me through this journey. My life at UIUC would not have been complete without the friends that I made after moving to the Chambana area, and I am thankful for each one of them. I am especially thankful for the fateful day when I was returning home from the Mechanical Engineering department during my first week at UIUC and met my now-fiancé, Kanika Narang. Without Kanika's love and support, I would not have been able to pursue Ph.D. with as much zeal and vigor as I did.

Table of Contents

Chapter 1

Introduction

Numerical solution of partial differential equations (PDEs) is central to much of today's engineering analysis and scientific inquiry. Techniques such as the finite element method (FEM), the finite volume method (FVM), and the spectral element method (SEM) are used to approximate solutions of PDEs on a collection of volumes or elements whose union constitutes a *mesh* that covers the entire computational domain, Ω. Construction of an optimal mesh (grid) is not a trivial task and often becomes a bottleneck for complex domains. Automatic mesh generation remains an open problem for certain classes of elements, particularly hexahedrons (hex) in 3D, which are otherwise attractive from an accuracy and performance standpoint, especially for high-order methods such as the SEM. In this dissertation, we present SEM-based technologies that enable high-order computational fluid dynamics (CFD) in complex domains by circumventing constraints imposed by traditional mesh construction approaches. The key components of this dissertation are SEM-based overlapping Schwarz (OS) methods (which form the majority of the work) and mesh optimization. We develop these technologies with emphasis on stability, accuracy, and parallel scalability, and demonstrate the potential of these methods on several challenging fluid-thermal applications.

1.1 Motivation

The primary requirement for numerical simulations, irrespective of the method, is a discretization. For simulations based on PDEs, the discretization is typically associated with a *mesh*, which indicates nodal placements for function values, whether used directly in the finite difference (FD) approximations of derivatives or as nodal points associated with basis functions as in the case of the FEM and SEM. When solving the PDE of interest in a domain Ω, the goal is to generate a mesh that has the minimum number of grid points needed to capture the physics of the problem and to ensure that the mesh is free of regions (e.g., elements, or patches of elements in the FEM and SEM) that adversely impact the performance of the PDE solver (e.g., elements with high skewness or aspect ratio). Typically, meshes generated by automated methods do not satisfy either of these criteria and generating an optimal mesh can be time-consuming.

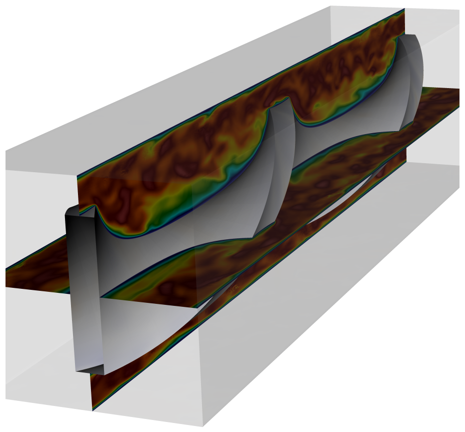


Figure 1.1: Velocity magnitude contours for flow over a bar twisted in the streamwise direction.

To illustrate some of the challenges in generating computational meshes for three-dimensional domains, we consider the deceptively simple geometry of Fig. 1.1, which shows a twisted ribbon of a rectangular cross-section in the interior of a square duct. If one simply had a square duct or just a twisted ribbon in isolation, it would be possible to generate a 3D mesh for these components by building a corresponding 2D mesh and extruding (and twisting, in the case of the ribbon) the 2D mesh. The presence of the sharp corners on each of the components (the ribbon and the duct), however, prevents the use of such a straightforward approach.

The essential difficulty with the extrusion approach is obvious when we consider a two-dimensional time analogy. Consider a 2D domain in which a rectangular rotor is turning within a square domain, as illustrated in Fig. 1.2. If we discretize the fluid (color) region with an arbitrary Lagrangian-Eulerian (ALE) formulation in which the mesh is allowed to deform, we can support only a small amount of rotation before the mesh becomes entangled or is torn apart. Because of the corners, some set of the mesh vertices are pinned to the duct walls, while others are pinned to the rotor. As the rotor spins, the edge graphs that connect the rotor to the wall must get longer and longer, leading to mesh entanglement.

The most common approach to meshing rotating machinery parts such as the toy 2D rotor of Fig. 1.2(a) is to use rotating meshes with nonconforming interfaces, as illustrated by Fig. 1.2(b). Here, we show a pair of overlapping meshes in which the outer mesh has a circular cut-out to guarantee that it will not intersect the rotor at any angle. An alternative approach is to have a pair of meshes that share a sliding circular interface, i.e., in d space dimensions, they have an interface of dimension $d-1$ that precisely matches each mesh in shape, but not necessarily in the number of mesh elements. The overlapping approach, which we pursue in this thesis, allows more flexibility in defining the individual meshes because the shared boundaries of the subdomains (here illustrated in red and black) have only minimal requirements concerning the clearance of

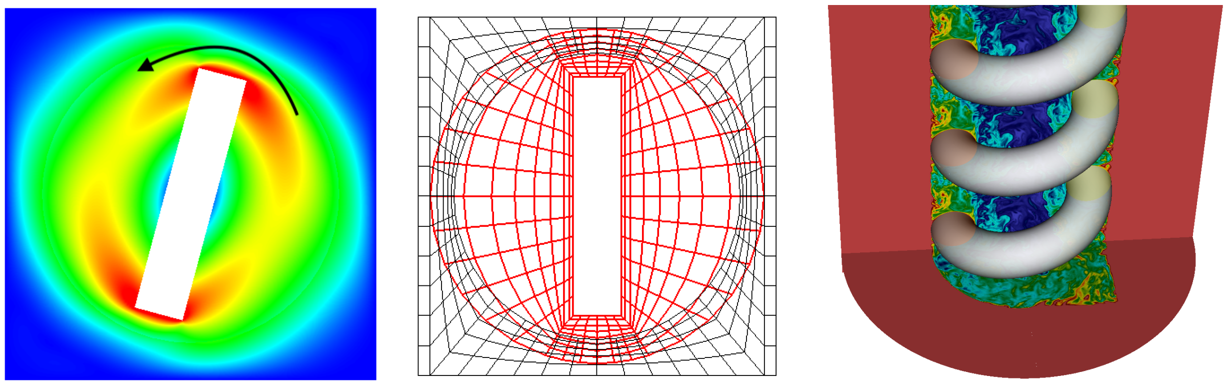


Figure 1.2: Left to right: (a) Velocity magnitude contours for a rotating bar, (b) overlapping 2D spectral element meshes, and (c) wire-coil insert inside a noncircular pipe.

the parts (e.g., the rotor) and the amount of overlap.

The relevance of our 2D example to the 3D configuration of Fig. 1.1 is that the axial direction in the 3D case is completely analogous to time in the rotating 2D case. Clearly, any attempt to extrude (or sweep) a valid 2D mesh to cover the 3D domain of Fig. 1.1 is going to suffer the same mesh tearing or entanglement issues that would occur with the ALE approach to the rotating machinery problem. By using a nonconforming approach, however, we can use a pair of simple extruded (and twisted) meshes to cover the domain in Fig. 1.1 . In fact, the results of Fig. 1.1 were generated using precisely this approach. The simple mesh pairs of Fig. 1.2(b) were swept in the axial direction and the inner mesh was twisted. A similar approach was used to mesh the geometry of Fig. 1.2(c), which shows the temperature distribution for flow through a U-channel with a wire coil insert. This challenging problem is based on an actual application at Argonne National Laboratory's advanced photon source and was not tractable before the development of the Schwarz methods presented in this thesis. We describe this problem and other applications of overlapping Schwarz methods in more detail in Chapter 9.

In the preceding examples, overlapping Schwarz resolved the challenge of meshing relatively simple twisted domains featuring sharp corners. If either of the domains had been smooth, the mesh entanglement issue would not arise because vertices on the smooth domain wall could slide along the boundary. Another class of problems where Schwarz (or other nonconforming) approaches offer significant benefits are cases where fine-scale features in the solution, which require a fine-scale mesh, are confined to a limited part of the domain.

Figure 1.3(a) shows velocity magnitude contours for a thermally buoyant plume in a stably stratified background. The physics of this problem leads to fine-scale structure restricted in the plume region, and to laminar flow in the far-field. Figure 1.3(b) shows the slice view of the axisymmetric conforming mesh used to model the buoyant plume. This monodomain mesh is suboptimal because the resolution needed to model

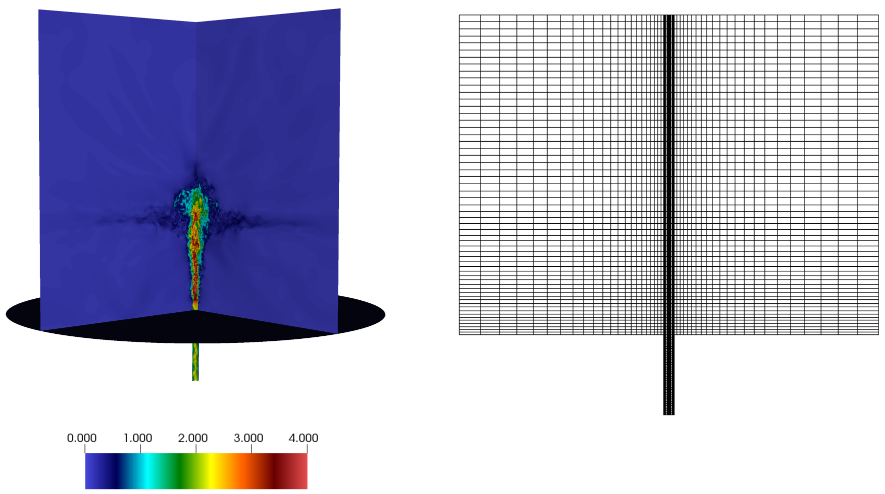


Figure 1.3: (left to right) (a) Velocity magnitude contours for a buoyant plume in oceanic environment, (b) slice view of the axisymmetric mesh for the plume.

the fine-scale structures of the plume leads to unwanted resolution in the far-field (e.g., away from the plume in the axial and radial directions), which increases the computational cost of modeling this problem.

In contrast to the buoyant plume example, where the difference in the scales of fluid motion adversely impacts the mesh quality, Fig. 1.4 shows an example of flow over shark skin denticles where the geometric features of the domain lead to a suboptimal mesh. The miniature roughnesses (denticles) on shark skin are located within the viscous sublayer of the flow and require more elements to discretize the geometry of the denticles than are needed to resolve the flow. Consequently, meshes generated for discretizing flow over these denticles typically lead to a much higher resolution than needed away from the denticles.

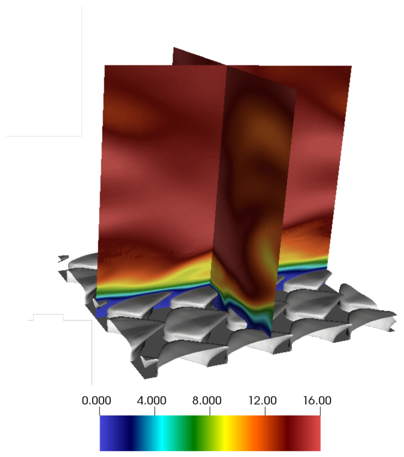


Figure 1.4: Velocity magnitude contours for flow over denticles on shark skin.

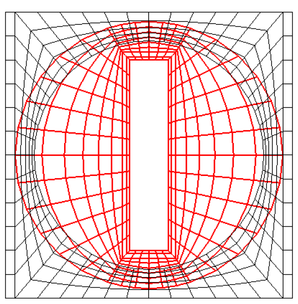
1.1.1 Nonconforming Schwarz methods

As indicated by the preceding examples, a major challenge in meshing complex domains is posed by needing to have a mesh that matches resolution requirements (as in the shark's skin and plume cases) and geometric constraints (e.g., as in the twisted ribbon and U-channel/wire-coil insert cases). Here, we consider a class of *nonconforming* meshes based on subdomain overlap. Throughout this dissertation, a *conforming mesh* will be considered to be a union of hexahedral *curvilinear brick* elements that cover a domain, $\Omega \subset \mathbb{R}^d$, $d = 2$ or 3, or subdomain, $\Omega^j \subset \Omega$ The elements in a conforming mesh do not overlap. Adjacent elements will share in their entirety a common face (curvilinear quadrilateral), edge (curvilinear line segment), or vertex (point). These local adjacency constraints are ultimately the source of difficulty in meeting the global resolution constraints.

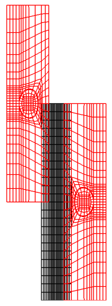
Nonconforming methods allow adjacent subdomains $\Omega^i \neq \Omega^j$ to be meshed without satisfying all the conforming mesh constraints. Nonconforming methods may be *overlapping* or *nonoverlapping.* For the latter, we require that if $\Omega \subset \mathbb{R}^d$, then $\Omega^i \cap \Omega^j \subset \mathbb{R}^{d-1}$. Some examples of the nonconforming methods are adaptive mesh refinement (AMR) [1, 2, 3], mortar element method (MEM) [4, 5, 6, 7], and the sliding mesh method (SMM) [8, 9]. AMR and MEM are based on nonoverlapping grids, but allow $h-$refinement (dividing an existing element into two or more elements) or $p-$refinement (increasing the polynomial order used for quadrature on an element) to provide additional resolution where needed. While generally effective for problems where features of interest are restricted to specific regions of the domain, AMR and MEM still lack the flexibility of OS methods in mesh generation for complicated domains such as those shown in Fig. 1.2. The effectiveness of the sliding mesh method is also limited because it does not allow different subdomains to overlap.

Nitsche's method relaxes the constraint of generating meshes with matching interfaces by allowing use of overlapping grids (e.g., $\Omega := \Omega^1 \bigcup \Omega^2$) to generate a nonconforming nonoverlapping discretization of the domain [10,11]. The nonoverlapping discretization is obtained by computing intersection of overlapping grids, and partitioning the domain into a union of disjoint subsets (e.g., $\Omega := \tilde{\Omega}^1 \bigcup \Omega^2$, where $\tilde{\Omega}^1 := \Omega^1 \backslash (\Omega^1 \cap \Omega^2)$). Thus, Nitsche's method is more effective (in terms of mesh generation) than AMR, MEM, and SMM for the applications that we are targeting. A drawback of Nitsche's method, however, is that splitting the overlapping grids into nonoverlapping subsets leads to arbitrary shaped elements at the interface of these different subsets. Consequently, Nitsche's method is undesirable for SEM-based implementations since the SEM relies on hexahedral elements (to realize fast operator evaluation).

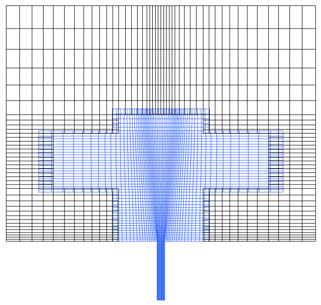
Nonconforming OS-based methods circumvent issues posed by conformal grids and nonconforming nonoverlapping grids, by allowing the domain to be represented as the union of simpler subdomains, each of which



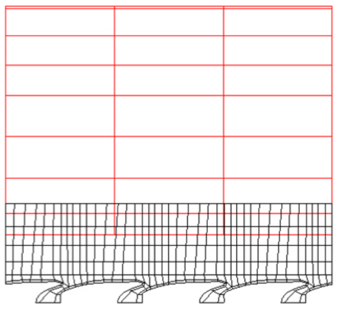
(a) Bar with a spanwise twist inside a square channel.



(b) Wire-coil insert inside a circular pipe.



(c) Thermal plume in a stratified environment.



(d) Denticles on shark skin.

Figure 1.5: 2D slices of different meshes demonstrating the potential of overlapping subdomains for different fluid-thermal problems.

can be meshed independently with relatively simple mesh constructions. The nonconforming union of these meshes allows combinations of local mesh topologies that are otherwise incompatible, which is a feature of particular importance for complex 3D domains.

Figure 1.5(a) shows the example of nonconforming overlapping meshes used for the rotor example. A moving mesh is used to discretize the rotor, and it is overlapped with a static background mesh for the square channel. Figure 1.5(b) shows a slice view of the overlapping meshes for a wire-coil insert inside a pipe. A 2D mesh is extruded along the helical wire-coil, and the singularity at the pipe centerline is avoided by overlapping it with a mesh for modeling the central flow channel.

Figure 1.5(c) and 1.5(d) demonstrate the potential of OS-based methods for reducing the element count based on the flow structures (for the buoyant plume) or geometric features (for the denticles on shark skin) in different parts of the domains.

In each of these cases, we see that overlapping grids can provide an efficient means to simulate fluid-thermal flow, with minimal time spent in mesh generation. Using OS-based methods to solve the incompressible Navier-Stokes equations accurately, however, is not trivial, and we will discuss some of the critical issues that must be addressed, later in this chapter.

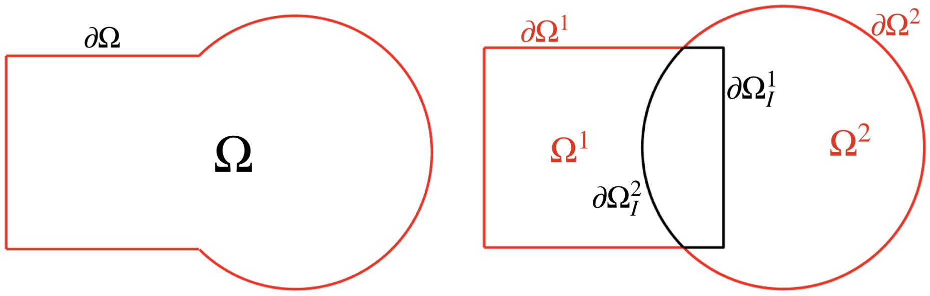


Figure 1.6: (left to right) (a) Composite domain Ω (b) modeled by overlapping rectangular (Ω^1) and circular (Ω^2) subdomains. $\partial\Omega_I^s$ denotes the segment of the subdomain boundary $\partial\Omega^s$ that is interior to another subdomain Ω^r.

1.2 Overlapping Schwarz Method for Solving PDEs

In the preceding section, we presented several examples that demonstrate the potential of overlapping grids for different fluid-thermal applications. In order to motivate how the incompressible Navier-Stokes equations can be solved on overlapping grids, we introduce the overlapping Schwarz method for solving the Poisson equation using a two-dimensional example. We choose the Poisson equation for simplicity, with the OS method readily extending to more general equations in three-dimensional domains.

Figure 1.6 shows the example of a composite domain Ω, which is partitioned into a rectangle (Ω^1) and a circle (Ω^2) with nonzero overlap such that $\partial\Omega_I^1 := \partial\Omega^1 \subset \Omega^2$ and $\partial\Omega_I^2 := \partial\Omega^2 \subset \Omega^1$. We use $\partial\Omega_I^s$ to denote the "interdomain boundary", namely the segment of the subdomain boundary $\partial\Omega^s$ that is interior to another subdomain. The interdomain boundaries $\partial\Omega_I^1$ and $\partial\Omega_I^2$ are highlighted in Fig. 1.6(b). Assuming that we are solving the Poisson equation $-\nabla^2 u = f$ with Dirichlet boundary condition ($u = u_b$) on $\partial\Omega$, the Schwarz alternating method for solving the PDE on overlapping subdomains Ω^1 and Ω^2 is

$$\begin{aligned} -\nabla^2 u^{1,[q]} &= f \text{ in } \Omega^1, \quad u^{1,[q]} = u_b \text{ on } \partial\Omega^1 \backslash \partial\Omega_I^1, \quad u^{1,[q]} = u^{2,[q-1]} \text{ on } \partial\Omega_I^1, \\ -\nabla^2 u^{2,[q]} &= f \text{ in } \Omega^2, \quad u^{2,[q]} = u_b \text{ on } \partial\Omega^2 \backslash \partial\Omega_I^2, \quad u^{2,[q]} = u^{1,[q]} \text{ on } \partial\Omega_I^2, \end{aligned} \tag{1.1}$$

where $u^{s,[q]}$ refers to the solution u in Ω^s at the qth Schwarz iteration. Starting with an initial condition $u^{s,[0]}$, the Poisson equation is solved sequentially in each subdomain with interdomain boundary data exchange before each Schwarz iteration ($q = 1 \dots Q$). The number of Schwarz iterations Q can either be specified before starting the iterative process, or the Schwarz iterations can be repeated until the solution converges to within a desired tolerance in the overlap.

Since the Schwarz alternating method solves the PDE sequentially in each subdomain (e.g., first Ω^1 and then Ω^2 in the example above), a drawback of this approach is that it restricts the parallelism of the method as the number of subdomains, S, increases [12]. As an alternative, one can consider the *simultaneous* Schwarz method, which overcomes the sequential dependencies of the alternating method by solving the PDE simultaneously on all subdomains and exchanging interdomain boundary data prior to the start of each iteration. The simultaneous Schwarz method for solving the Poisson equation using $q = 1 \ldots Q$ Schwarz iterations is

$$\begin{aligned} -\nabla^2 u^{1,[q]} &= f \text{ in } \Omega^1, \quad u^{1,[q]} = u_b \text{ on } \partial\Omega^1 \backslash \partial\Omega_I^1, \quad u^{1,[q]} = u^{2,[q-1]} \text{ on } \partial\Omega_I^1, \\ -\nabla^2 u^{2,[q]} &= f \text{ in } \Omega^2, \quad u^{2,[q]} = u_b \text{ on } \partial\Omega^2 \backslash \partial\Omega_I^2, \quad u^{2,[q]} = u^{1,[q-1]} \text{ on } \partial\Omega_I^2. \end{aligned} \tag{1.2}$$

As we can see in (1.2), the boundary data on $\partial\Omega_I^s$ are interpolated from the previous Schwarz iteration, and the PDE is solved simultaneously on all subdomains. Naturally, the advantage of simultaneous Schwarz is its parallelism, and all the work presented in this dissertation is based on the simultaneous Schwarz scheme.

We note that due to the use of overlapping grids, the solution to the PDE is double-valued in the overlap ($\Omega^1 \cap \Omega^2$), and converges to the same function with Schwarz iteration.

In order to understand how the error varies in each subdomain due to these Schwarz iterations, we subtract the exact solution $\tilde{u}$ of the boundary value problem to obtain the equation for error $e^{s,[q]} = u^{s,[q]} - \tilde{u}$:

$$\begin{aligned} -\nabla^2 e^{1,[q]} &= 0 \text{ in } \Omega^1, \quad e^{1,[q]} = 0 \text{ on } \partial\Omega^1 \backslash \partial\Omega_I^1, \quad e^{1,[q]} = e^{2,[q-1]} \text{ on } \partial\Omega_I^1, \\ -\nabla^2 e^{2,[q]} &= 0 \text{ in } \Omega^2, \quad e^{2,[q]} = 0 \text{ on } \partial\Omega^2 \backslash \partial\Omega_I^2, \quad e^{2,[q]} = e^{1,[q-1]} \text{ on } \partial\Omega_I^2. \end{aligned} \tag{1.3}$$

Since simultaneous Schwarz iterations use the solution from the previous iteration in each subdomain to obtain interdomain boundary data, the boundary condition for error depends on the error at the previous Schwarz iteration. It is straightforward to derive from the BVP in (1.3) that in accordance with the maximum principle, the error is maximum at the interdomain boundary in each domain, and goes to zero away from the interface.

To demonstrate the effectiveness of (1.2) for solving the Poisson equation $-\nabla^2 u = f$, we consider a 1D example in a domain $\Omega \in [0, \pi]$ with homogeneous boundary conditions $u(0) = u(\pi) = 0$. For $f = 2e^{-x}cos(x)$, the exact solution to this Poisson problem is $\tilde{u} = e^{-x}sin(x)$.

We use two overlapping grids to partition this domain, $\Omega^1 \in [0, 0.6\pi]$ and $\Omega^2 \in [0.4\pi, \pi]$, with a grid overlap of $\theta = 0.2\pi$. Starting with an initial guess $u^{s,[0]} = 5$ in each subdomain, the BVP (1.2) is solved. Figure 1.7(a) shows how the solution varies in each subdomain at the $q = 1$ (red), 2 (blue), 10 (pink) and 15 (black) iterations. The interdomain boundaries of the overlapping grids are indicated by dashed black lines in the figure. We can see in Fig. 1.7(a) that after the first iteration, the solution $u^{s,[1]}$ satisfies the inhomogeneous boundary condition on $\partial\Omega_I^s$ obtained from the solution in the corresponding overlapping domain at the previous Schwarz iteration ($u^{r,[0]}$) (assuming $\partial\Omega_I^s$ overlaps Ω^r). Similarly, $u^{s,[2]}$ satisfies

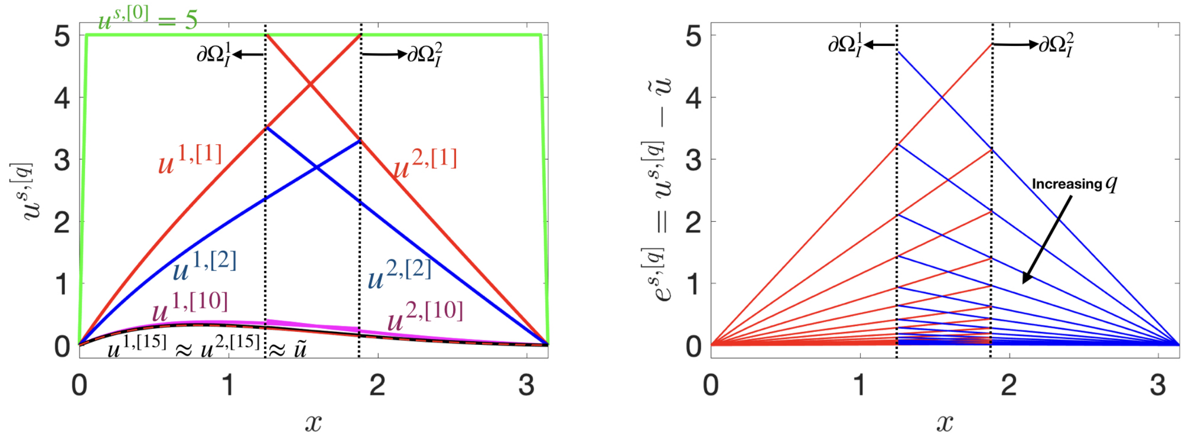


Figure 1.7: Evolution of (left) the solution ($u^{s,[q]}$) and (right) the error ($e^{s,[q]} = u^{s,[q]} - \tilde{u}$) for the Poisson equation in two overlapping grids using simultaneous Schwarz iterations.

the inhomogeneous boundary condition obtained from $u^{r,[1]}$, and so on. With each Schwarz iteration, the numerical solution converges towards the exact solution in each subdomain, and by the 15th Schwarz iteration, the solution is within a tolerance of 10^{-2} with $\tilde{u}$, i.e. $\max_{s=1\dots S} ||u^{s,[15]} - \tilde{u}||_\infty \leq 10^{-2}$.

Figure 1.7 also shows how the error varies in each subdomain with each Schwarz iteration. The equation for error satisfies (1.3), which is maximum at the interdomain boundary and goes to 0 at the domain boundary in each subdomain. After 15 Schwarz iterations, the error is reduced to less than 10^{-2} in both subdomains.

The convergence of the solution due to the simultaneous Schwarz iterations (1.2) depends on the decay of the Green's functions associated with the PDE of interest and the width of the overlap between different subdomains. For the 1D Poisson problem considered here, we can use (1.3) to show that the maximum error in each subdomain at the qth Schwarz iteration is

$$||e^{s,[q]}||_\infty = \left(\frac{L-\theta}{L}\right) ||e^{r,[q-1]}||_\infty, s \neq r, \tag{1.4}$$

where $L = 0.6\pi$ is the length of each subdomain and $\theta = 0.2\pi$ is the grid overlap width. (1.4) shows that increasing the grid overlap (θ) increases the rate at which the solution of the Poisson problem converges to the exact solution. This observation is important as it impacts important design choices for the SEM-based OS framework (Schwarz-SEM) to solve the incompressible Navier-Stokes equations (Chapter 3).

In this section, we have used a simple 1D example to demonstrated how the Overlapping Schwarz method can be used to solve a boundary value problem. The basic principles of the Schwarz method discussed here readily extend to higher dimensions ($d = 2, 3$) for general boundary value problems, and the reader is referred to [12] for a detailed discussion on the OS-based methods. We note that (1.4) describes the convergence of the simultaneous Schwarz iterations for the 1D Poisson problem considered here, and similar expressions could be derived, albeit not as straightforward, for more general PDEs (e.g., Navier-Stokes equations) in higher space-dimensions.

The subject of this dissertation is the use of overlapping Schwarz methods for time advancement of the solution of the incompressible Navier-Stokes equations (INSE) on nonconforming overlapping meshes. This application of OS methods should not be confused with the more common context in which overlapping Schwarz is used, which is to develop domain-decomposition-based preconditioners for solving the system of equations resulting from discretizing the PDE of interest on a given mesh [13, 14].

1.3 Literature Survey

In this section, we present a historical overview of the overlapping Schwarz and a survey of issues that arise when implementing these methods for the numerical solution of PDEs. We note that *overset grid* or *chimera* schemes fall within the Schwarz framework and research in these methods has addressed many of the principal concerns.

The overlapping Schwarz method was introduced by Schwarz in 1870 to prove the solvability of Laplace's equation on the union of two simply shaped overlapping subdomains [15]. The decomposition for Schwarz's initial model problem is illustrated in Fig. 1.6. The OS method started gaining popularity around 1960 when it was further developed for solving the Laplace's equation on arbitrarily shaped overlapping subdomains with smooth boundaries [16, 17], and was used to numerically solve elliptic PDEs on computers [18, 19]. These were some of the earliest works on OS-based methods, and they led to the development of overlapping grid techniques for solving time-dependent boundary value problems [20, 21, 22].

Over the last few decades, OS-based methods have become popular for solving different classes of problems on overlapping grids, such as the incompressible flow [23, 24, 25, 26], compressible flow [27, 28, 29, 30, 31], electromagnetics [32, 33], heat transfer [34, 35, 36], and particle tracking [37], and have been implemented using the FD, FEM, FVM and SEM. Some of the popular implementations of OS methods are included in commercial and research codes such as Star-CCM [26], OpenFOAM [38, 39], elsA [28, 40], Nalu [41], Overflow [30, 42], and Overture [43].

Despite the different class of problems and implementations, the fundamental approach for OS remains the same; a domain is decomposed into subdomains with a finite overlap and the PDE of interest in then solved on the mesh discretizing each subdomain.

1.3.1 Interdomain boundary data interpolation

Since overlapping grids introduce interdomain boundaries $\partial\Omega_I^s$, they require boundary data to be interpolated from the corresponding overlapping subdomain Ω^r. In the context of FEM and SEM, this interpolation requires the identification of elements in the overlapping subdomain. For example, Fig. 1.8 shows the interface boundary $\partial\Omega_I^1$ overlapping elements of Ω^2. For each grid point[1] ($\mathbf{x}^*$) on $\partial\Omega_I^1$, the *donor* element Ω^{2,e^*} (element number e^* in the mesh used for Ω^2) must be identified, and then the reference space coordinates ($\boldsymbol{\xi}^* = \{\xi^*, \eta^*, \zeta^*\}$) inside that element must be determined (Fig. 1.8). These reference space coordinates are used to interpolate boundary data for the incompressible Navier-Stokes equations at each time-step. For moving or deforming meshes, donor elements must be re-identified after each time-step since the location of $\partial\Omega_I^s$ can change with respect to the subdomain Ω^r that it overlaps. We note that the reason why interpolation depends on reference space coordinates will be clear through the discussion in the next chapter.

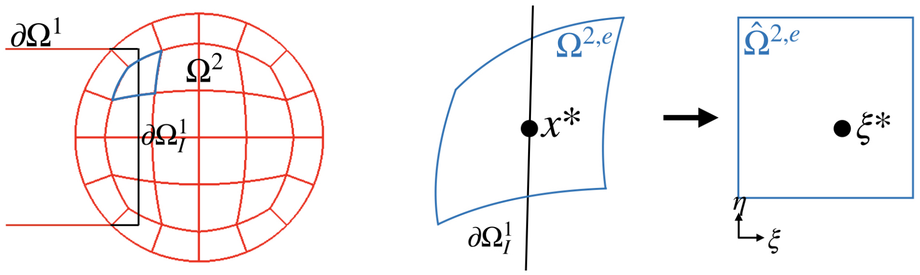


Figure 1.8: (left) Interface boundary ($\partial\Omega_I^1$) overlapping with spectral elements of Ω^2. (right) For each grid point discretizing $\partial\Omega_I^1$, the element $\Omega^{2,e}$ (spectral element with element number e in Ω^2) and reference space coordinates ($\boldsymbol{\xi}^* = \{\xi^*, \eta^*\}$) inside that element must be determined.

In our framework, we first use a hash table-based search for identifying the donor element and then use an optimization-based approach for identifying the reference-space coordinates inside the isoparametric mapping of the donor element. The objective for the optimization method is to minimize the Euclidean distance between the sought point $\mathbf{x}^*$ and the point $\tilde{\mathbf{x}}^*$ in physical-space that corresponds to the reference-space coordinates ($\tilde{\boldsymbol{\xi}}^*$) in the isoparametric mapping of the donor element. In our framework, identification of computational coordinates and high-order interpolation is done using *findptslib* [44], a set of routines that are a part of an open-source general-purpose communication library *gslib* [45]. TIOGA [46], Suggar++ [47], PUNDIT [48], and PEGASUS [49] are some of the popular libraries used by different OS-based methods for determining interdomain boundary data exchange information between overlapping subdomains.

We note that *findptslib* has two key routines, *findpts* and *findpts_eval*. Given a list of points, *findpts* determines the computational coordinates for each point. Using these computational coordinates, *findpts_eval*can interpolate any given scalar function. *findpts* and *findpts_eval* were originally designed for monodomain

[1]We use the term grid point to refer to the Gauss-Lobatto-Legendre points of the spectral element mesh.

meshes and, as a result, could not be directly integrated into the Schwarz-SEM framework. In Chapter 3, we describe the changes that we have made to *findptslib* for enabling computational coordinate identification for interdomain boundary grid points in an arbitrary number of overlapping grids. Here, we also describe how we have improved the computational efficiency of *findpts_eval* by a factor of 3X.

1.3.2 Spatial accuracy

Since OS-based methods rely on interpolation for interdomain boundary data, the spatial and temporal accuracy of these interpolated data impacts the overall accuracy of the solution. Most of the existing OS-based methods are based on the FD method or the FVM and are at most fourth-order accurate in space [23, 33, 37, 46, 50, 51]. Recently, sixth-order finite difference based methods have been presented in [30, 52], and a sixth-order finite volume-based scheme has been presented in elsA [28]. Most OS-based methods, thus, rely on low-order interpolation, but recently there has been a shift towards polynomial based interpolation to increase the spatial accuracy of the solution [24, 38, 46, 53]. A spectral element based Schwarz method presented by Merrill et al. [24] has demonstrated exponential convergence (with the order of the polynomial used for quadrature on each element), which is the starting point of the work that is presented in this dissertation. In this Schwarz-SEM framework, we use Lagrange interpolants, which are the basis function for SEM, of the same order (N) as the solution to interpolate the interdomain boundary data. This approach ensures that our OS method maintains the exponential convergence of the underlying SEM.

Spatial interpolation of the interdomain boundary data also impacts the overall mass conservation in each subdomain. For incompressible flow, it is critical to ensure that the interpolation method for obtaining the interdomain boundary data is conservative. This means that the net mass flux through the boundaries of each subdomain must be zero. Flux correction techniques have been used for FVM-based OS methods in [53, 54] and velocity field correction for mass conservation has been used for FD-based methods in [23].

In Chapter 4, we present an approach for imposing a correction on the interpolated velocity field via a constrained minimization problem. This approach ensures that the corrected velocity field is similar to the interpolated velocity field, and conserves mass in each subdomain. The projection of the interpolated velocity field onto a divergence-free space has demonstrated that, in addition to improving the accuracy of the method, it leads to improved stability of the Schwarz-SEM framework and requires fewer Schwarz iterations when high-order extrapolation is used for the interdomain boundary data.

1.3.3 Temporal accuracy

The temporal convergence of OS-based methods is constrained by the temporal accuracy of interdomain boundary data. Since the solution is known only up to the previous time-step, the interdomain boundary data has an inherent temporal error of $O(\Delta t)$ if it is directly interpolated from the overlapping domain. To improve accuracy, Peet and Fischer [55] used time-extrapolated data (of order m) from m previous time-steps for interdomain boundaries. However, stability analysis showed that this method was unstable for $m > 1$, and needed 1 to 3 Schwarz iterations at each time-step for increased stability. These Schwarz (corrector) iterations used the solution from the most recent iteration for interdomain boundaries. This predictor-corrector (PC) method was used in the Schwarz-SEM framework presented by Merrill et al. [24] to demonstrate third-order temporal convergence. We note that in this PC method, each subdomain integrates the solution of the PDE using the same time-step size, and this is referred to as a single-rate method.

In the current work, we use this high-order predictor-corrector scheme to maintain the temporal accuracy of the underlying SEM solver. In Chapter 5, we present stability analysis that gives novel insight into the impact of the number of corrector iterations on the stability of the PC scheme. We also describe how this stability analysis has led to the development of a PC scheme with improved stability properties.

1.3.4 Global integration and distance function generator

Use of OS framework for solving different fluid-thermal applications on overlapping subdomains requires us to answer questions that are not as trivial as they are in the context of a single conforming domain (monodomain).

For example, integration in a single conforming domain is straightforward to effect in Galerkin methods via the *mass matrix.* In overlapping grids, however, global integration is not straightforward, and grid points in the overlap region must be weighted appropriately. In implementations based on Nitsche's method, overlapping grids are partitioned into nonoverlapping discretizations, and the resulting arbitrary shaped elements at grid interfaces are split into regular triangles/tetrahedra for integration [11]. This approach allows integration to be performed on nonoverlapping subsets. Another approach for integration on overlapping meshes is to use the quadrature in nonoverlapping regions as usual, and remeshing the overlap region with regular-shaped elements and then performing the quadrature as usual [56]. Each of these approaches relies on remeshing, which is guaranteed only for tetrahedra or triangular elements. Additionally, these approaches are computationally expensive and complicated to implement in parallel, since they rely on identifying intersections of different overlapping grids. In our framework, global integration is achieved via partition-of-unity functions that ensure that grid points in the overlapping region are appropriately weighted. We explain our approach for global integration in Chapter 4.

Similar to the issue of global integration, obtaining a distance field (a field that specifies the shortest distance from each element to a given point/line/surface) or maintaining a fixed flow rate through overlapping subdomains is not straightforward, and requires coupling between subdomains. To the best of our knowledge, existing OS-based methods do not address these problems (or they use methods that do not apply to our framework), and we have developed an OS-based approach for generating a distance field and maintaining a fixed flow rate through overlapping subdomains.

1.3.5 Scalability

OS methods rely on general (i.e., off-grid) interpolation and communication between different overlapping subdomains for solving the PDE of interest. Efficient algorithms to effect general interpolation and communication, which is not a standard component of most numerical PDE solvers, is therefore of paramount importance, particularly in a parallel context where data locality and load balance are of concern. With the integration of *findptslib* into the Schwarz-SEM framework, we can now use an arbitrary number of overlapping subdomains to solve the INSE in parallel on high-performance computers. We demonstrate the strong scaling of the Schwarz-SEM framework on a high number of MPI ranks ($P > 40,000$) in Chapter 8.

The preceding text has focused on the fundamental concerns for the implementation of OS-based methods to solve the incompressible Navier-Stokes equations. In addition to systematically addressing these issues, as will be explained in the following chapters, we have identified methods to improve the computational performance of both monodomain and overlapping grid calculations.

1.3.6 Multirate time-stepping

One of the novel aspects of this dissertation is the introduction of multirate time-stepping for incompressible flow simulations. The idea behind multirate time-stepping is to allow subdomains to independently advance the governing time-dependent PDE at a rate that is optimal (e.g., either stability or accuracy constrained) for each individual subdomain, with periodic updates to synchronize the solutions.

Multirate time-stepping methods were introduced for a system of ODEs by Rice [57], and have since been developed for parabolic and hyperbolic PDEs [58, 59]. Multirate time-steppers are virtually nonexistent for the incompressible Navier-Stokes equations because the solution is very sensitive to the pressure, which satisfies an elliptic Poisson or pseudo-Poisson problem at every time-step. In Chapter 6, we introduce an SEM-based multirate time-stepping scheme that preserves exponential convergence of the SEM, demonstrates third-order temporal convergence, and marches all subdomains simultaneously using a novel predictor-corrector scheme. We also expand our stability analysis framework developed for the single-rate PC scheme to understand how the time-step size ratio between different subdomains impacts the stability of the multirate time-stepping scheme.

1.4 Mesh optimization

While OS-based methods generally yield relatively simple meshes in individual subdomains, mesh optimization is still of value in both the mono- and multidomain cases. Mesh optimization can greatly improve the computational efficiency of numerical simulations. For INSE, this includes improving elements that constrain the performance of the pressure Poisson solver and limit the maximum time-step size due to the Courant-Friedrichs-Lewy (CFL) constraint. Figure 1.3(b) and Fig. 1.5(c) show an example of how mesh optimization improves the quality of performance-degrading elements by getting rid of high aspect ratio elements away from the region of interest. In this example of an incompressible flow model of a thermal plume, numerical experiments showed that mesh smoothing allowed the use of a threefold-large time-step size for the smoothed mesh as compared to the original mesh.

Mesh smoothing methods have been developed over several decades and various methods of mesh improvement, such as Laplacian smoothing [60,61,62], constrained Laplacian smoothing [63], optimization [61,64,65], and untangling [66, 67] have been developed for finite element (FE) meshes. However, these methods were mostly developed for linear meshes, and the shape metrics did not address hex meshes [68, 69, 70]. Recently, high-order mesh optimization methods have been presented in [71, 72].

We have developed a mesh smoother for the SEM [73] keeping in mind key aspects of automated mesh generation methods. High-order meshes are typically generated by generating a linear mesh using a block decomposition approach [74], followed by projection onto the actual high-order geometry surface [75, 76, 77, 78]. As a result, most elements in a mesh are linear or quadratic, except for the surface conforming high-order elements. Additionally, meshes generated for simulating incompressible flow have boundary layer resolving elements to capture near-wall physics, which, due to their anisotropy, can lead to high aspect-ratio elements in the far-field. These high aspect-ratio elements adversely impact the condition of the system for the pressure Poisson equation (PPE) and the CFL of the grid. Lastly, since the quality of the surface mesh constrains mesh quality in the interior of a domain, it is important to be able to smooth the surface mesh in order to maximize the quality of the given mesh.

Existing mesh smoothing methods, to the best of our knowledge, do not address these issues, primarily because traditional mesh optimization methods were designed for application to solid mechanics (where there is typically no boundary layer). Mesh improvement methods that support surface smoothing, typically rely on a parametric representation of the original geometry in order to smooth the surface mesh. This approach is not possible in the FEM or the SEM, where only a discrete representation of the original geometry is available through the original mesh. In this dissertation, we present a mesh smoother that improves the quality of a mesh while maintaining the mesh quality at critical places such as boundary layers and improves the quality of CFL constraining elements. We also introduce a novel approach for surface mesh smoothing, and show how improving the mesh quality improves the condition of the system for pressure Poisson equation (PPE), which leads to a reduction in the cost of solving the PPE.

1.5 Organization and Thesis Contribution

In this dissertation, we develop the Schwarz-SEM method and a mesh optimizer to enable high-order CFD in complex domains. The starting point of this dissertation is the Schwarz-SEM framework presented in [24], and our goal is to extend this framework to accurately solve a wide variety of challenging fluid-thermal problems that are intractable with the monodomain SEM.

In Chapter 2, we provide a background of the SEM, and the spatial and temporal discretization of the incompressible Navier-Stokes equations. Here, we also summarize how the solution of the incompressible Navier-Stokes equations is advanced in time on a monodomain grid. All the discussion in this chapter is based on the existing literature, and this chapter is important for understanding how the INSE are solved in the Schwarz-SEM framework.

Chapter 3 presents the methodology for time-advancing the solution of the INSE on overlapping subdomains, using the predictor-corrector scheme presented in [24, 55]. Here, we describe how we have introduced discriminators in *findpts*for enabling donor element search for interdomain boundary grid points in an arbitrary number of overlapping grids. We also describe how we have reduced the computational cost of *findpts_eval*by eliminating redundant communication costs associated with interpolating multiple scalar functions for a given set of computational coordinates. In Chapter 3, we explain our methodology for generating a global distance function across overlapping subdomains, a capability that is crucial for enabling Reynolds-Averaged Navier-Stokes (RANS) based calculations in the Schwarz-SEM framework. Using an example with $S = 2$ and 3 overlapping subdomains, we demonstrate that the Schwarz-SEM framework maintains the spatial and temporal convergence of the underlying SEM-based solver that is described in

Chapter 2.

Chapter 4 describes the velocity correction method that we have developed to ensure that the use of interpolation for interdomain boundary data does not violate the divergence-free constraint, which is critical for the incompressible Navier-Stokes equations. In this chapter, we also describe the strategy for maintaining a fixed flow rate through overlapping subdomains, a capability that is crucial for modeling internal flows in periodic domains. Since the determination of quantities such as the flow rate and Nusselt number requires global integration in overlapping subdomains, which is not trivial, we develop an approach for global integration in subdomains with arbitrary overlap. This novel approach is based on the generation of partition-of-unity functions that ensure that grid points in the overlap region are weighted appropriately. Most of the methods developed in Chapters 3 and 4 appear in a recent publication in the Computers & Fluids journal [79].

In Chapter 5, we use a simple 1D time-dependent boundary value problem to empirically analyze the stability behavior of the predictor-corrector scheme for time-advancing the solution of the INSE in the Schwarz-SEM framework. The stability analysis presented in this chapter gives us novel insight that the PC scheme is more stable when the number of corrector iterations is odd as compared to when it is even, for $m > 1$. In this chapter, we also develop an improved predictor-corrector scheme that does not exhibit this odd-even pattern in stability with respect to the number of corrector iterations.

Chapter 6 presents our predictor-corrector approach for multirate time-stepping in two overlapping subdomains, which is a first in the context of incompressible flows. The PC scheme described here allows subdomains to independently time-advance the solution of INSE at a rate that is optimal for each individual subdomain. In this chapter, we also extend the stability analysis framework developed for the single-rate time-stepping scheme (Chapter 5) to understand how the time-step size ratio between different subdomains impacts the stability of the multirate time-stepping scheme. Using an example with $S = 2$ overlapping subdomains, we demonstrate that our multirate time-stepping method maintains the spatial and temporal convergence of the underlying SEM-based solver that is described in Chapter 2.

In Chapter 7, we describe our approach for mesh smoothing that is based on a combination of Laplacian smoothing and optimization-based strategy. This mesh smoother has been designed to improve the quality of elements that constrain the computational performance of turbulent flow calculations (e.g., high aspect-ratio elements) while preserving the shape of the elements at critical places such as boundary layers. In our mesh smoother, we allow elements on the surface of the mesh to be smoothed by the Laplacian smoother and mesh optimizer, followed by projection of the smoothed surface elements on to the surface of the original mesh. This novel surface smoothing approach does not rely on any CAD parameterization for mesh smoothing and does not introduce geometrical approximation errors at the surface of the mesh. The mesh smoothing methodology presented in this chapter has been published in the Journal of Scientific Computing [73].

We present the timing and scaling of several key aspects of the Schwarz-SEM framework in Chapter 8. The results in this chapter help us understand how the number of Schwarz iterations at each time-step affect the total time to solution, and how the PC-based multirate time-stepping scheme reduces the computational cost of a calculation in comparison to the PC-based single-rate time-stepping scheme. We also look at strong scaling results for the Schwarz-SEM framework in 3D problems to understand how much time is spent in donor element identification and interpolation in comparison to the total time to the solution at each time step. The results presented in this chapter are essential for understanding the benefits and limitations of the Schwarz-SEM framework, and in order to effectively use it for different fluid-thermal applications in the future.

In chapter 9, we present several examples that demonstrate the effectiveness of the Schwarz-SEM framework in solving fluid-thermal applications. We start with two 3D problems having nontrivial solutions to benchmark the Schwarz-SEM framework and then solve problems that are intractable with the monodomain SEM framework. Here, we also demonstrate the effectiveness of the multirate time-stepping method in accurately capturing the physics of turbulent flow in a thermally buoyant plume.

1.6 Use of Notation

In this thesis, we bring together results that have been published in different journals. Due to the multidisciplinary nature of the work, we do not attempt to develop a consistent notation, and certain variables can take different meanings across different chapters, depending on the context. We do, however, take care to define notation as it is introduced.

Chapter 2

SEM for the Incompressible Navier-Stokes Equations

The goal of this dissertation is to develop the Schwarz-SEM framework for solving the incompressible Navier-Stokes equations in complex domains (e.g., Fig. 1.5). Since the Schwarz-SEM framework builds upon the monodomain SEM framework, this chapter presents relevant details of the monodomain SEM. Using a model problem, we describe how the weighted residual method (WRM) is used to discretize the Helmholtz equation, which leads to a linear system that is solved using efficient, matrix-free, iterative methods in a high performance computing (HPC) setting.[1] The concepts developed for this model problem readily extend to three-dimensional domains for more general partial differential equations. Since we are interested in solving the INSE, we also describe our spatial and temporal discretization for INSE in this chapter.

2.1 Spectral Element Method

The spectral element method (SEM) is a high-order weighted residual method that was introduced by Patera [80] and has been used to solve a variety of challenging fluid dynamics and heat transfer problems [81, 82, 83]. Here, we introduce the fundamentals of SEM and explain how it is used to solve PDEs. The first step in solving PDEs in a domain, Ω, using the SEM, is domain decomposition of Ω into (conforming and nonoverlapping) spectral elements. The PDE of interest is then transformed into the weak form using the weighted residual method (WRM), followed by use of basis functions to transform the weak form into a linear system of equations. To demonstrate this solution process, we use the domain shown in Fig. 2.1(a) to solve the Helmholtz equation

$$-\nabla^2 u + u = f \text{ in } \Omega, \left.\frac{\partial u}{\partial \hat{\mathbf{n}}}\right|_{\partial\Omega} = 0, \tag{2.1}$$

where $u(\mathbf{x})$ is the desired solution, f is a known function, and homogeneous Neumann conditions are imposed on the domain boundary.

[1]Iterative methods are the *most* efficient solvers for unstructured discretizations of elliptic PDEs in 3D.

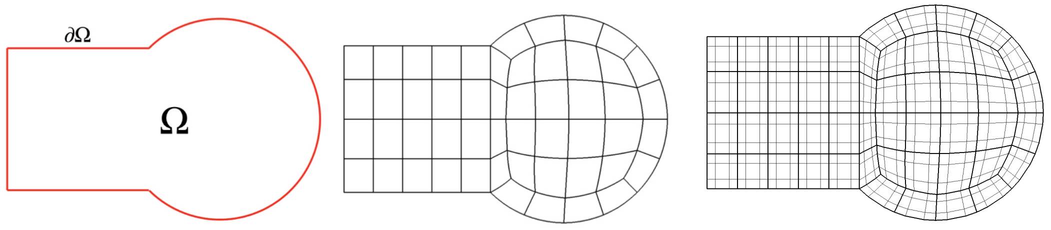


Figure 2.1: (left to right) (a) Domain Ω, (b) the spectral element mesh used to model Ω, and (c) the spectral element mesh with Gauss-Lobatto-Legendre points ($N = 3$) in each element.

2.1.1 Spectral element mesh

In the standard SEM, the computational domain Ω is the union of nonoverlapping elements, Ω^e, $e = 1 \dots E$, where each spectral element Ω^e is an isoparametric mapping of an element in the reference domain $\hat{\Omega} := [-1, 1]^d$ in d space dimensions[2]. The reference domain $\hat{\Omega}$ is a square for $d = 2$ and a cube for $d = 3$. Within a domain (or within a subdomain for overlapping meshes), we assume that elements are conforming, which means that any point on an element edge is coincident with at least one other point in an adjacent element, unless that edge (or face or vertex) is on the subdomain boundary, $\partial\Omega$ or $\partial\Omega_I$. Figure 2.1(b) shows the conforming spectral element mesh for Ω with $E = 52$ spectral elements, along with the corresponding Gauss-Lobatto-Legendre (GLL) points for each element in Fig. 2.1(c).

Figure 2.2 shows the coordinate transformation of a spectral element for $N = 4$ from reference-space ($\hat{\Omega}$) to Cartesian-space (Ω) [84], along with the corresponding GLL points. The GLL points in each spectral element are a tensor-product of the 1D GLL points (in reference-space), which are solutions to the equation

$$(1 - \xi^2) L_N^{'}(\xi) = 0, \tag{2.2}$$

where $L_N^{'}$ is the derivative of the Legendre polynomial of degree N. The GLL points, which serve as nodal points for the SEM, are also quadrature points, which allows efficient approximation of the integrals used in the WRM.

2.1.2 Weighted residual method for SEM

The weighted residual method is a popular technique for solving partial differential equations by representing the solution using a set of basis functions and satisfying the PDE in a weighted sense. The SEM is based on the Galerkin method in which the weight functions (or test functions) are chosen to be the same as those

[2]Here, we use superscript e to indicate the element number. In the multidomain case, we use superscript s to indicate the subdomain number. There should be no confusion as the usage will be clear from the context.

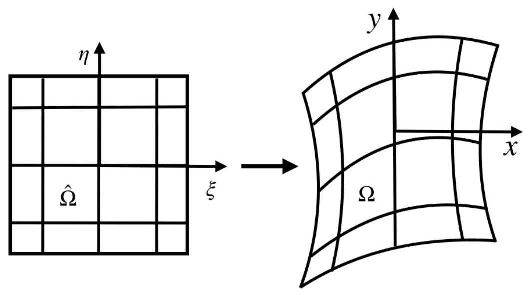


Figure 2.2: Coordinate transformation of a spectral element for $N = 4$ from the reference space ($\hat{\Omega}$) to the physical space (Ω).

used to represent the solution of the PDE. In this section, we will restrict our discussion to the Helmholtz equation (2.1), but we note that the weighted residual method readily extends to more general PDEs (e.g., INSE), and the reader is referred to [85] for a detailed discussion on the WRM.

Following standard practice, we derive the weak form of (2.1)

$$\int_{\Omega} \nabla v \cdot \nabla u \, d\mathbf{x} + \int_{\Omega} vud\mathbf{x} = \int_{\Omega} vf \, d\mathbf{x} + \int_{\partial\Omega} v(\nabla u) \cdot \hat{\mathbf{n}} \, d\mathbf{x}, \tag{2.3}$$

for all test functions $v \in \mathcal{H}^1$, where L^2 is the Hilbert space of square-integrable functions, and the Sobolev space $\mathcal{H}^1$ is the usual space of functions for solution to second-order PDEs, which are in L^2 and whose derivatives are also in L^2. Since we assume homogeneous Neumann conditions on $\partial\Omega$, the surface integral in (2.3) vanishes, and the weak form of (2.1) simplifies to

$$a(v, u) + (v, u) = (v, f), \tag{2.4}$$

where $(v, u) := \int_{\Omega} v \cdot u \, d\mathbf{x}$ and $a(v, u) := \int_{\Omega} \nabla v : \nabla u \, d\mathbf{x}$ are respective L^2 and energy inner products. The next step in solving the Helmholtz equation is the basis functions that are used to represent functions on GLL points in a spectral element mesh.

Basis functions

The basis functions in the spectral element method are a tensor-product of Nth-order Lagrange polynomials on the GLL points in each element. Consequently, any scalar function u is represented on each element as

$$u(\boldsymbol{\xi})|_{\Omega^e} = \sum_{i=0}^{N} \sum_{j=0}^{N} \sum_{k=0}^{N} u^e_{ijk} l_i(\xi) l_j(\eta) l_k(\zeta), \tag{2.5}$$

where $l_i(\xi)l_j(\eta)l_k(\zeta)$ is the tensor-product of the 1D Lagrange interpolants in each space dimension, u^e_{ijk} is the nodal basis coefficient, and $\boldsymbol{\xi} := \{\xi, \eta, \zeta\}$ are the coordinates of the GLL points in the reference domain $\hat{\Omega} = [-1, 1]^d$. A consequence of Lagrange interpolants is that the basis coefficients, u^e_{ijk}, are the nodal values of u on the tensor-product of GLL points in element e.

Similar to the solution (2.5), we represent the geometry of each element Ω^e with an isoparametric mapping, meaning the geometry shares the same functional form as the solution,

$$\mathbf{x}(\boldsymbol{\xi})|_{\Omega^e} = \sum_{i=0}^{N}\sum_{j=0}^{N}\sum_{k=0}^{N} \mathbf{x}^e_{ijk} l_i(\xi)l_j(\eta)l_k(\zeta). \tag{2.6}$$

In the SEM, the nodal (GLL) values u^e_{ijk} (or $u^e_{i,j,k}$) are numbered lexicographically inside each element, and constitute the local element-by-element vector $\underline{u}$. Consequently, the local form for a function u in a spectral element mesh with E elements and polynomial order N is

$$\underline{u} = [\underline{u}^1 \underline{u}^2 \underline{u}^3 \ldots \underline{u}^E]^T, \tag{2.7}$$

$$\underline{u}^e = \underline{u}^e_{ijk}, i = 0 \ldots N, j = 0 \ldots N, k = 0 \ldots N \tag{2.8}$$

where $\underline{u}^e$ represents the local vector for Ω^e.

A consequence of using the local form for function representation is that while nodal values on element interfaces have a unique representation on the grid with global degrees of freedom (DOF), the local form leads to multiple values for DOFs that are shared between elements. Figure 2.3 illustrates the global ($\underline{u}_g$) and local ($\underline{u}$) degrees of freedom in a spectral element mesh with $E = 2, N = 2$. As we can see, the three global degrees of freedom shared between the two elements are duplicated in the local element-by-element form and, thus, can be multivalued across the element interface.

Thus, before we use the basis functions for discretization in (2.4), we address how function continuity (e.g., for global nodes 3, 6 and 9 in Fig. 2.3) is enforced in the monodomain SEM framework.

2.1.3 Function continuity in spectral element mesh

C^0 continuity across element interfaces requires that coincident GLL points have the same function values.

$$\mathbf{x}^e_{ijk} = \mathbf{x}^{\hat{e}}_{\hat{i}\hat{j}\hat{k}} \implies u^e_{ijk} = u^{\hat{e}}_{\hat{i}\hat{j}\hat{k}}. \tag{2.9}$$

For any function $u \in X^N \subset \mathcal{H}^1$, we have a pair $(\underline{u}, \underline{u}_g)$ satisfying $\underline{u} = V\underline{u}_g$, where $\underline{u}$ is the vector of (redundantly-stored) *local* basis coefficients and $\underline{u}_g$ is the corresponding (uniquely-defined) *global* represen-

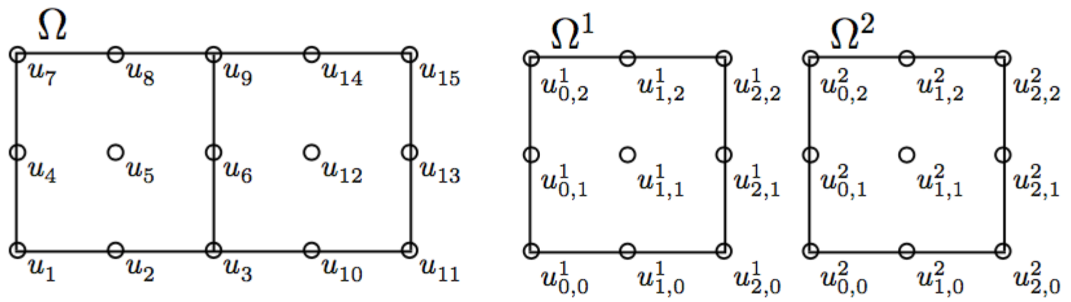

Ω
u_7 u_8 u_9 u_{14} u_{15}
u_4 u_5 u_6 u_{12} u_{13}
u_1 u_2 u_3 u_{10} u_{11}
Ω^1
$u^1_{0,2}$ $u^1_{1,2}$ $u^1_{2,2}$
$u^1_{0,1}$ $u^1_{1,1}$ $u^1_{2,1}$
$u^1_{0,0}$ $u^1_{1,0}$ $u^1_{2,0}$
Ω^2
$u^2_{0,2}$ $u^2_{1,2}$ $u^2_{2,2}$
$u^2_{0,1}$ $u^2_{1,1}$ $u^2_{2,1}$
$u^2_{0,0}$ $u^2_{1,0}$ $u^2_{2,0}$

Figure 2.3: (left) Global ($\underline{u}_g$) form and (right) local ($\underline{u}$) numbering of two spectral elements with $N = 2$. Image taken from [84].

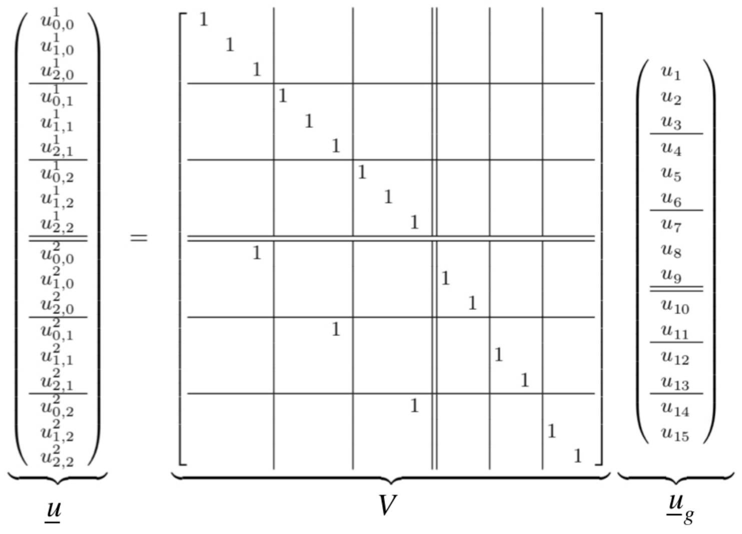


Figure 2.4: Gather operation on a global vector $\underline{u}_g$ to obtain the local element-by-element vector $\underline{u}$. Image taken from [84].

tation. For our weighted-residual formulation, we require a local-to-global map that is given by $\underline{u}_g = V^T\underline{\tilde{u}}$, where $\underline{\tilde{u}}$ represents a set of functions that may be multivalued at element interfaces. This gather operation, V^T, leads to a summation of shared values and VV^T is, thus, not idempotent. Using the gather-scatter operator, continuity is enforced on a multivalued function as $\underline{u} = VV^T\underline{\tilde{u}}$.

Figure 2.4 shows the action of gather operator in distributing the vector of global values $\underline{u}_g$ to a vector of local values $\underline{u}$, summing up the shared values in the process.

The gather-scatter operator, VV^T, is often used in conjunction with a diagonal counting matrix C that is used to generate averaged values as $\underline{u} = C^{-1}VV^T\underline{\tilde{u}}$. The diagonal weighting matrix, $C = w_{ii}\delta_{ij}$, would have entries $w_{ii} = w_i$ corresponding to the vector $\underline{w} = VV^T\underline{e}$, where $\underline{e}$ is the vector having unity for each GLL point of each element. $C^{-1}VV^T$ is thus an idempotent projector into the space of continuous functions, which has the effect of averaging values shared across the interfaces. The $C^{-1}VV^T$ operator is useful for methods such as mesh optimization, where each element is optimized individually, and then the nodal displacement determined by the optimizer for each GLL point is then averaged with other elements that share that GLL point.

In practice, the matrix V is never explicitly formed, and instead, the action of V and V^T is implemented using indirect addressing. In our framework, the nearest-neighbor exchange VV^T is implemented in *gslib* [86], an open-source communication library.

2.1.4 Linear system resulting from the variational form

In the preceding section, we have discussed various ingredients that are important for solving a PDE using the SEM. Using the tensor-product of Lagrange polynomials as our basis functions, we now transform (2.4) into a system of the form $Ax = b$.

Assuming that the known function f on the right-hand side of (2.1) is already represented on the spectral element mesh in a form similar to (2.5), we write the right-hand side of (2.4) as

$$\begin{aligned} (v, f) &:= \int_{\Omega} v f d\mathbf{x}, && (2.10)\\ &:= \sum_{e=1}^{E} \int_{\Omega^e} v f d\mathbf{x}, && (2.11)\\ &:= \sum_{e=1}^{E} \sum_{i=0}^{N^2} \sum_{j=0}^{N^2} v_i^e \, (\int_{\Omega^e} \phi_i(\mathbf{x}) \phi_j(\mathbf{x}) \, d\mathbf{x}) \, f_j^e, && (2.12)\\ &= \sum_{e=1}^{E} (\underline{v}^e)^T B^e \underline{f}^e, && (2.13)\\ &= \underline{v}^T B \underline{f}, && (2.14) \end{aligned}$$

where $\phi_i(\mathbf{x})$ and $\phi_j(\mathbf{x})$ are a tensor-product of basis functions for v and f, respectively. The mass matrix B^e for each element Ω^e is a function of the quadrature weights of the GLL points in e and the Jacobian of the transformation associated with the isoparametric mapping (2.6) (e.g., see pg. 173 in [84]). We note that in (2.14), B is the unassembled global mass matrix, which is block diagonal with each block corresponding to a spectral element,

$$B = \begin{bmatrix} B^1 & & & & & \\ & B^2 & & & & \\ & & B^3 & & & \\ & & & \ddots & & \\ & & & & \ddots & \\ & & & & & B^E \end{bmatrix}. \tag{2.15}$$

Due to the choice of basis functions (tensor-product of Lagrange interpolants), the mass-matrix B^e of each element Ω^e is diagonal, and consequently, B is also diagonal.

Since (2.14) is based on the local form of functions, we cast this system into the global form using the gather-scatter operator in order to satisfy continuity requirements for the global solution,

$$(v, f) \quad = \quad (V\underline{v}_g)^T B(V\underline{f}_g), \tag{2.16}$$

$$= \quad \underline{v}_g^T V^T BV\underline{f}_g, \tag{2.17}$$

where $V^T BV$ is the global assembled mass matrix. Similar to (2.17), the left hand side of (2.4) simplifies to

$$a(v, u) + (v, u) \quad := \quad \int_\Omega \nabla v \cdot \nabla u d\mathbf{x} + \int_\Omega vud\mathbf{x}, , \tag{2.18}$$

$$= \quad \underline{v}_g^T V^T AV\underline{u}_g + \underline{v}_g^T V^T BV\underline{u}_g, \tag{2.19}$$

where A is a stiffness matrix, and the reader is referred to Chapter 4 of [84] for a detailed derivation of the system in (2.19). We note that the stiffness matrix A, similar to B, is a block diagonal matrix with each block corresponding to the local stiffness matrix, A^e for element Ω^e. Unlike B^e, however, A^e is full when the geometry is deformed, leading to a nominal number of nonzeros for A of $n_{nz} = E(N+1)^6$, which prohibits practical formation of A for $N > 3$.

Using (2.19) and (2.17), the system in (2.4) is written as

$$\underline{v}_g^T (V^T AV + V^T BV)\underline{u}_g = \underline{v}^T V^T BV\underline{f}_g, \tag{2.20}$$

$$H\underline{u}_g = \underline{b}_g, \tag{2.21}$$

where $H = V^T AV + V^T BV$ and $\underline{b}_g = V^T BV\underline{f}_g$.

Matrix-free iterative solution of SEM systems

The system (2.21) is typically solved using an iterative method such as the conjugate gradient (CG) method or generalized minimum residual (GMRES) method (e.g., [87]), with an appropriate preconditioner. For solving a system of the form $A\underline{x} = \underline{b}$, iterative methods such as CG and GMRES construct approximate solutions in the Krylov subspace $\{\underline{r}, A\underline{r}, \ldots A^{k-1}\underline{r}\}$, where $\underline{r} = b - A\underline{x}_0$ depends on an initial guess $\underline{x}_0$. The essential step for these algorithms is matrix-vector products of the form

$$\underline{w}_g = V^T A V \underline{p}_g, \tag{2.22}$$

where $\underline{w}$ and $\underline{p}$ are intermediate vectors in the iteration algorithm. In the SEM, we store only *local* vectors. Hence, the first operation in (2.22) is superfluous, given that we already have $\underline{p} = V\underline{p}_g$. Also, since we wish only to store $\underline{w}$, we recast (2.22) as

$$\underline{w} = V\underline{w}_g = VV^T A\underline{p}. \tag{2.23}$$

Since A is block-diagonal with each block corresponding to a spectral element, (2.23) can be represented as

$$\underline{w} = VV^T[(A^1\underline{p}^1)^T(A^2\underline{p}^2)^T \ldots (A^e\underline{p}^e)^T]^T, \tag{2.24}$$

where A^e and $\underline{p}^e$ represent the operator and vector corresponding to element Ω^e. Equation (2.24), thus, consists of local element-by-element operator evaluation, followed by the communication-intensive gather-scatter operation, VV^T. An advantage of (2.24) is that it avoids packing and unpacking of $\underline{p}$ and $\underline{w}$ from their global to local forms and concentrates the communication of VV^T into a single data exchange.

(2.24) is an important result because it shows that we can solve a PDE such as (2.1) by discretizing a domain Ω into spectral elements, and using element-by-element operators instead of global operators that are computationally expensive to store and evaluate.

We reiterate that this approach of using the SEM to solve the Helmholtz equation readily extends to more general PDEs such as the incompressible Navier-Stokes equations. We seek to solve the INSE as large eddy simulations (LES) or direct numerical simulations (DNS), which require enough grid points to adequately resolve all flow structures in the domain, and can have anywhere between $E = 1000$ and 10^6 spectral elements with polynomial order N between 6 and 12, based on the domain and PDE of interest. Simulating time-dependent flows, thus, requires us to solve a system of equations similar to (2.21) at each time-step for each function of interest (e.g., velocity and pressure), which in turn depends on our ability to do local element-by-element matrix-vector products (e.g., (2.24)) for all the spectral elements in the mesh.

2.1.5 Parallel Implementation

In the previous section, we described our methodology of transforming the weak form of a PDE into a linear system of the form $Ax = b$, which is solved using element-by-element matrix-vector products. In order to expedite the solution process for the INSE, which can be expensive due to high-resolution requirements of LES and DNS, we use parallel computing based on the distributed computing model, where each processor (MPI rank) has its private memory. A spectral element mesh with E elements is partitioned onto P processors, $p = 0, \ldots, P-1$, where $P \leq E$ (each processor must have at least 1 spectral element), and each element is mapped to a processor. Each processor then uses its private memory to store all functions of interest for the elements that it has in local vectors of length $E_{(p)}(N+1)^d$, where $E_{(p)}$ is the number of elements assigned to process (or processor number) p.

The mesh partitioning, coupled with the private-memory model, leads to a natural work decomposition in the distributed-memory parallel computing paradigm. The majority of the computational effort in the SEM is in evaluation of the matrix-vector products (2.24), which are automatically parallel owing to the block-diagonal structure of global operators (e.g., A and B in (2.20)). If a balanced partition is used, the amount of work per processor scales as $\lceil E/P \rceil$. During a linear-system solve, after local element-by-element operator evaluation is done via fast tensor-product based matrix-vector products, global assembly is effected via VV^T, which ensures function continuity at interfaces that are shared between elements. See for example, (2.24).

In our framework, VV^T is effected in parallel by *gslib*, which has scaled to over six million MPI ranks (processors) using highly optimized routines having a communication overhead of at most $\log P$. All communication among MPI ranks is handled by *gslib* through a single MPI communicator (`Intracomm`), which all MPI ranks associated with a spectral element mesh belong to. *gs_crystal*, the MPI communication driver in *gslib*, is based on the *crystal router* algorithm of [88], and is available open-source [45].

In order to demonstrate how a spectral element mesh is mapped to MPI ranks in parallel, we use an example of a mesh with 5 spectral elements, shown in Figure 2.5(left). Figure 2.5(right) shows the partitioning of this spectral element mesh on 4 MPI ranks ($p = 0-3$), along with local element numbering on each MPI rank. For this example, $E_{(0)} = 2$ and $E_{(1)} = E_{(2)} = E_{(3)} = 1$.

Using $\underline{e}_g$ to represent the global element numbering in the spectral element mesh, $\underline{p}_l$ for the processor number each element is mapped to, and $\underline{e}_l$ to represent the local element numbering, the mapping for the

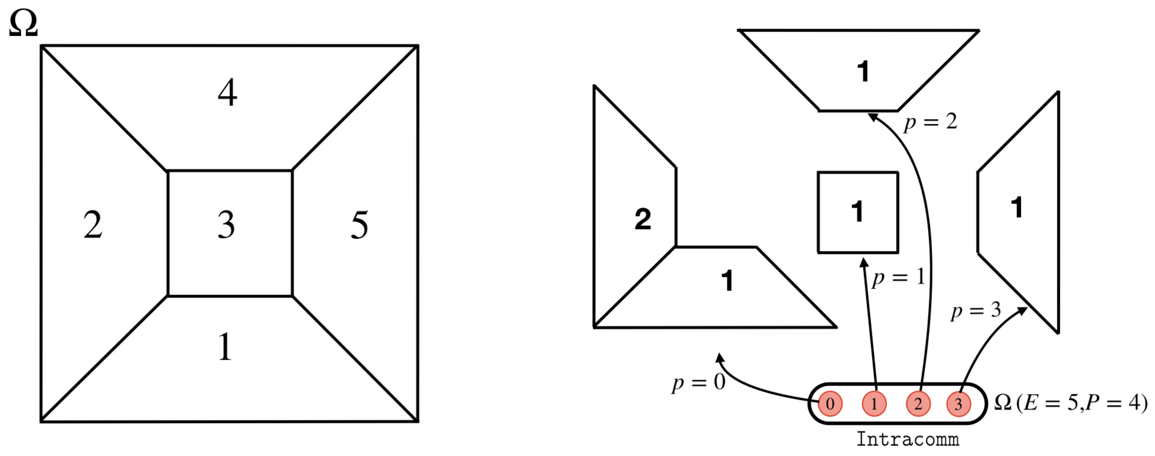


Figure 2.5: (left) Spectral element mesh with 5 elements along with global element number of each spectral element, and (right) the mesh partitioned onto 4 MPI ranks ($p = 0 - 3$) with local element numbering for each MPI rank. The plot on the right also shows a pictorial representation of the MPI communicator, `Intracomm`, that all MPI ranks associated with a spectral element mesh use to communicate via *gslib*.

spectral element mesh in Fig. 2.5 is:

$$\begin{aligned} \underline{e}_g &= [1, 2, 3, 4, 5] \leftarrow \text{global element number}, \\ \underline{p}_l &= [0, 0, 1, 2, 3] \leftarrow \text{MPI rank for each element}, \\ \underline{e}_l &= [1, 2, 1, 1, 1] \leftarrow \text{local element number}, \end{aligned} \tag{2.25}$$

This global-to-local element mapping in (2.25) is used to set up the gather-scatter operator in *gslib* to account for GLL points that are shared between elements.

Using the private memory model and local element-by-element approach for operator evaluation, the monodomain SEM framework has been used to solve large-scale incompressible flow problems on high performance computers (HPC) [81, 82, 83]

2.2 Incompressible Navier-Stokes Equations

In the preceding section, we summarized the key ingredients of the SEM for solving PDEs in parallel using the WRM. Discussing the SEM in more comprehensive detail is beyond the scope of this dissertation, and the reader is referred to [84] for further details. In this section, we focus on the INSE and describe our spatial and temporal discretization for time-advancing the solution of the INSE in the monodomain SEM framework.

Consider the solution of the constant-density incompressible Navier-Stokes equations in a given computational domain $\Omega(t)$ in $\mathbb{R}^d$ at time t,

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p + \frac{1}{Re} \nabla^2 \mathbf{u} + \mathbf{f}, \tag{2.26}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{2.27}$$

where $\mathbf{u}(\mathbf{x}, t)$ and $p(\mathbf{x}, t)$ represent the velocity and pressure[3] solution as a function of position $\mathbf{x} \in \mathbb{R}^d$ and time t, and $Re = LU/\nu$ is the Reynolds number based on the characteristic length scale L, velocity scale U, and kinematic viscosity of the fluid ν. $\mathbf{f}(\mathbf{x}, t)$ is a nondimensional forcing that is a function of position and time. The solution of (2.26) and (2.27) also depends on the initial (for time-dependent problems) and boundary conditions, which we discuss in Section 2.2.2.

2.2.1 Spatial & temporal discretization

We solve the unsteady NSE in velocity-pressure form using semi-implicit BDFk/EXTk time-stepping in which the time derivative is approximated by a kth-order backward difference formula (BDFk), the nonlinear terms (and any other forcing) are treated with a kth-order extrapolation (EXTk)[4], and the viscous and pressure terms are treated implicitly. This approach leads to a linear unsteady Stokes problem to be solved at each time-step, which is split into independent viscous and pressure (Poisson) updates [89].

Assuming a constant time-step size Δt for all time-steps[5], we compute a tentative velocity field comprising contributions from the BDFk and the explicit terms, at time t^n, as

$$\acute{\mathbf{u}}^n \;\; = \;\; -\sum_{j=1}^{k} \beta_j \mathbf{u}^{n-j} + \Delta t \sum_{j=1}^{k} \alpha_j (-\mathbf{u} \cdot \nabla \mathbf{u} + \mathbf{f})^{n-j}, \tag{2.28}$$

where the superscript $n - j$ indicates quantities evaluated at earlier time-steps, t^{n-j}, and β_j and α_j are the BDF and EXT coefficients, respectively. $\acute{\mathbf{u}}^n$ constitutes the nonlinear update but does not account for the divergence-free constraint or viscous effects. The divergence-free constraint (2.27) is enforced through a pressure correction. A pressure Poisson equation is obtained by taking the divergence of the momentum equation, assuming the solution is divergence-free at time level t^n, $\nabla \cdot \mathbf{u}^n = 0$, and using the identity $\nabla^2 \mathbf{u}^n = \nabla(\nabla \cdot \mathbf{u}^n) - \nabla \times \nabla \times \mathbf{u}^n$:

[3]Here, we use p to denote the pressure solution in the INSE. In context of parallel computing, we use p to denote the processor number. There should be no confusion as the usage will be clear from the context.

[4]From here on, we will use k to represent the order of accuracy of our temporal discretization, unless otherwise stated.

[5]We will consider a constant time-step size for all the methods and applications presented in this dissertation.

$$\begin{aligned} -\nabla^2 p^n &= -\frac{\nabla\cdot\hat{\mathbf{u}}^n}{\Delta t} + \frac{1}{Re}\nabla\cdot\sum_{j=1}^{k}\alpha_j(\nabla\times\boldsymbol{\omega}^{n-j}), && (2.29)\\ \implies -\nabla^2 p^n &= \nabla\cdot\mathbf{f}_p, && (2.30)\end{aligned}$$

where $\boldsymbol{\omega}^n = \nabla\times\mathbf{u}^n$, and

$$\mathbf{f}_p = -\frac{\hat{\mathbf{u}}^n}{\Delta t} + \frac{1}{Re}\sum_{j=1}^{k}\alpha_j(\nabla\times\boldsymbol{\omega}^{n-j}). \tag{2.31}$$

The advantage of using the curl-curl form for the viscous term to decouple the velocity and pressure solve is that the equation governing the error in divergence ($\nabla\cdot\mathbf{u}^n$) is an elliptic PDE instead of a parabolic PDE. As a result, this formulation is stable with the splitting-induced divergence errors that are only $\mathcal{O}(\Delta t^k)$ [89,90].

Substituting the pressure solution p^n in (2.26), $\mathbf{u}^n$ is obtained by solving the Helmholtz equation

$$\frac{\beta_0}{\Delta t}\mathbf{u}^n - \frac{1}{Re}\nabla^2\mathbf{u}^n = -\nabla p^n + \frac{\hat{\mathbf{u}}^n}{\Delta t}. \tag{2.32}$$

Similar to Section 2.1.2, spatial discretization of (2.29) and (2.32) is based on variational projection operators. (2.32) is recast in weak form: *Find* $\mathbf{u}^n \in X_b^N$ *such that*

$$\frac{\beta_0}{\Delta t}(\mathbf{v},\mathbf{u}^n) + \frac{1}{Re}a(\mathbf{v},\mathbf{u}^n) = (\mathbf{v},\mathbf{f}_v^n), \tag{2.33}$$

for all test functions $\mathbf{v}\in X_0^N$, where $X^N\subset\mathcal{H}^1$, X_0^N is the set of basis functions that vanish on the domain boundary wherever Dirichlet conditions are prescribed, and X_b^N is the set of spectral element basis functions in $\mathcal{H}^1$ that satisfy the Dirichlet conditions on $\partial\Omega$ (and interdomain boundary $\partial\Omega_I$ in the case of overlapping grids). In (2.33), $\mathbf{f}_v$ is the right-hand side of (2.32).

We use $\partial\Omega_D$ to denote the subset of domain boundary $\partial\Omega$ on which Dirichlet conditions are imposed on velocity, and $\partial\Omega_N$ for the subset (e.g., outflow) on which pressure is prescribed. As expected, surfaces that have Dirichlet conditions for velocity have Neumann conditions for pressure, and vice-versa. Equation (2.33) is free of surface integrals because homogeneous Neumann conditions are used for velocity.

Similar to (2.33), the pressure Poisson equation (2.29) is cast in the weak form, which leads to surface integrals on $\partial\Omega_D$ due to integration by parts and Green's theorem: *Find* $p^n\in X_{0,p}^N\subset\mathcal{H}_{0,p}^1$ *such that*

$$a(\psi,p^n) = -(\mathbf{f}_p,\nabla\psi) + \int_{\partial\Omega_D}\psi\nabla p^n\cdot\hat{\mathbf{n}}\,dA + \int_{\partial\Omega_D}\psi\mathbf{f}_p\cdot\hat{\mathbf{n}}\,dA, \tag{2.34}$$

for all test functions $\psi \in X_{0,p}^N$, and $X_{0,p}^N$ is the set of basis functions vanishing on the domain boundary. The surface integrals in (2.34) vanish on the domain boundary where outflow (homogeneous pressure) boundary conditions are applied.

The use of Lagrange interpolants on GLL points allows integrals in (2.33) and (2.34) to be computed accurately using pointwise quadrature. The quadrature is performed on N^d GLL points in each element, save for the nonlinear terms, which are evaluated on $(3N/2)^d$ Gauss points to preserve skew-symmetry and ensure stability [91]. Using an approach similar to that explained in Section 2.1.4, the spatial discretization of (2.32) leads to a symmetric positive definite (SPD) system matrix for each component of the velocity, which is solved with diagonally-preconditioned conjugate gradient iteration. Spatial discretization of (2.29) results in an SPD Poisson system which is solved using p-multigrid accelerated by GMRES [92, 93]. Additional details concerning the SEM formulation for the incompressible Navier-Stokes equations are in [84, 89, 90].

Thermal Energy equation

When simulating heat transfer with incompressible flow, we also have the energy equation,

$$\frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T = \frac{1}{Pe} \nabla^2 T + q_T, \tag{2.35}$$

where $T(\mathbf{x}, t)$ is the temperature as a function of space $\mathbf{x} \in \mathbb{R}^d$ and time t, and q_T is an energy source term. $Pe = 1/(Re \cdot Pr)$ is the Peclet number. The Prandlt number $Pr = \nu/\alpha$ is the ratio of the momentum diffusivity (ν) and the thermal diffusivity (α). Similar to (2.32), using implicit treatment of the diffusion term and explicit treatment of the advection term, the solution T^n for temperature at time t^n is obtained as

$$\frac{\beta_0}{\Delta t} T^n - \frac{1}{Pe} \nabla^2 T^n = \sum_{j=1}^{k} \frac{\beta_j}{\Delta t} \mathbf{u}^{n-j} + \sum_{j=1}^{k} \alpha_j (-\mathbf{u} \cdot \nabla T + q_T)^{n-j}, \tag{2.36}$$

and the weak form of (2.36) leads to the Helmholtz equation

$$\frac{\beta_0}{\Delta t} (\varphi, T^n) + \frac{1}{Pe} a(\varphi, T^n) = (\varphi, \mathbf{f}_T^n) + \int_{\partial \Omega_{NT}} \varphi \nabla T^n \cdot \hat{\mathbf{n}} \, dA, \tag{2.37}$$

for all test functions $\varphi \in X_{b,T}^N$, where $X_{b,T}^N$ is the set of basis functions that satisfy the Dirichlet conditions on $\partial\Omega$ (and interdomain boundary $\partial\Omega_I$ in the case of overlapping grids), and $\mathbf{f}_T^n$ represents the terms on the right-hand side of (2.32). In (2.37), $\partial\Omega_{NT}$ is the set of surfaces that have a Neumann condition for temperature. Equation (2.35) is an advection-diffusion equation that can be solved for passive scalars advected by the flow. However, care must be taken to translate parameters such as the Prandtl number to the ratio of appropriate diffusivities for each scalar.

2.2.2 Boundary conditions

In our SEM formulation of Navier-Stokes, we typically impose either essential (Dirichlet) boundary conditions or natural (Neumann) boundary conditions on a surface for velocity. The boundary conditions for pressure are opposite to that of velocity, that is, a Dirichlet surface for velocity is Neumann for pressure, and vice-versa. In this section, we describe the different boundary conditions that we impose. We note that all boundary conditions can be a function of position and time, $\mathbf{u}(\mathbf{x},t)$ and $p(\mathbf{x},t)$ $\forall$ $\mathbf{x} \in \partial\Omega$, but for brevity, we simply represent boundary conditions as $\mathbf{u}$ and p.

Dirichlet conditions for velocity can be imposed on any surface as

$$\mathbf{u} = \mathbf{u}_b, \tag{2.38}$$

where $\mathbf{u}_b = 0$ for nonmoving walls due to the no-slip condition (continuum mechanics hypothesis), and $\mathbf{u}_b$ can be an arbitrary function depending on the problem of interest. The stress-free condition (symmetry) states that

$$\mathbf{u} \cdot \hat{\mathbf{n}} = 0,\ \nabla\mathbf{u} \cdot \hat{\mathbf{n}} = 0, \tag{2.39}$$

where $\hat{\mathbf{n}}$ is the unit vector normal to the surface. All outflow surfaces are Neumann for velocity and have homogeneous Dirichlet condition for pressure

$$\nabla\mathbf{u} \cdot \hat{\mathbf{n}} = 0,\ p = 0. \tag{2.40}$$

Since Dirichlet surfaces for velocity are treated as Neumann for pressure, an immediate concern is boundary condition (surface integrals) for pressure in (2.34). The Neumann condition on pressure is obtained by taking the dot product of the momentum equation (2.26) with the normal vector ($\hat{\mathbf{n}}$) to the surface

$$\frac{\partial p^n}{\partial n} = -\frac{\partial(\hat{\mathbf{n}} \cdot \mathbf{u})}{\partial t} - \hat{\mathbf{n}} \cdot \sum_{j=1}^{k} \alpha_j (\mathbf{u} \cdot \nabla\mathbf{u} - \mathbf{f})^{n-j} - \frac{1}{Re}\hat{\mathbf{n}} \cdot \sum_{j=1}^{k} \alpha_j (\nabla \times \boldsymbol{\omega}^{n-j}), \tag{2.41}$$

which simplifies to

$$\begin{aligned}\frac{\partial p^n}{\partial n} + \mathbf{f}_p \cdot \hat{\mathbf{n}} &= -\frac{\beta_0}{\Delta t}\mathbf{u}^n \cdot \hat{\mathbf{n}}, \qquad (2.42)\\ &= -\frac{\beta_0}{\Delta t}\mathbf{u}_b^n \cdot \hat{\mathbf{n}} \text{ on } \partial\Omega_D. \qquad (2.43)\end{aligned}$$

As we can see, for surfaces which have a Dirichlet condition on velocity, the surface integral (Neumann condition on pressure) in (2.34) is cast in terms of the Dirichlet velocity condition ($\mathbf{u}_b^n$). For surfaces that are Neumann for velocity (e.g., outflow), a Dirichlet condition is prescribed for pressure, and the surface integrals vanish because the test function $\psi \subset \mathcal{H}^1_{0,p}$.

In addition to (2.26), an arbitrary number of scalars can be solved for using the advection-diffusion equation (2.35). For the thermal energy equation, there are Dirichlet and Neumann boundary conditions as well. A Dirichlet condition for temperature ($T(\mathbf{x},t)$) can be imposed as

$$T = T_b. \qquad (2.44)$$

A Neumann condition for temperature is imposed as

$$\frac{\partial T}{\partial n} = T_q, \qquad (2.45)$$

where T_q is a user-specified flux, and $T_q = 0$ for insulated surfaces.

We note that in addition to the Dirichlet and Neumann conditions, we can also have periodic boundary conditions

$$\begin{aligned}\mathbf{u}(\mathbf{x},t) &= \mathbf{u}(\mathbf{x}+\mathbf{p},t),\\ p(\mathbf{x},t) &= p(\mathbf{x}+\mathbf{p},t), \qquad (2.46)\\ T(\mathbf{x},t) &= T(\mathbf{x}+\mathbf{p},t),\end{aligned}$$

where we assume that $\mathbf{p}$ is the vector describing the location of two periodic surfaces with respect to each other in the Cartesian-space. From an implementation perspective, periodic boundary conditions are imposed in the SEM framework using the gather-scatter operator by assuming that GLL points that are periodic to each other are coincident.

2.2.3 Summary of Navier-Stokes time advancement

To set the stage for our Schwarz-based Navier-Stokes solver, we summarize the Navier-Stokes time advancement of the preceding subsection. Using $\boldsymbol{\phi}^n = [p^n, \mathbf{u}^n]^T$ to represent the pressure and velocity solution in Ω at time t^n, and assuming that the solution is known up to time t^{n-1}, the solution to the INSE is time-advanced as:

1. Compute the tentative velocity field $\acute{\mathbf{u}}^n$ using (2.28), which accounts for the BDFk and time extrapolated nonlinear terms (EXTk terms).

2. Solve the *linear* Stokes subproblems (2.29) and (2.32) to compute the velocity-pressure pair, $\boldsymbol{\phi}^n = [\mathbf{u}^n, p^n]^T$

$$\mathbf{S}\boldsymbol{\phi}^n = \mathbf{r}^n, \quad \mathbf{u}^n|_{\partial\Omega_D} = \mathbf{u}_b^n, \quad p^n|_{\partial\Omega_N} = 0, \tag{2.47}$$

where $\mathbf{r}^n$, determined using $\acute{\mathbf{u}}^n$, accounts for all inhomogeneities for both pressure and velocity, given on the right-hand sides of (2.29) and (2.32), respectively:

$$\begin{aligned} \mathbf{r}^n &= [r_v^n, \mathbf{r}_p^n]^T, \\ \mathbf{r}_v^n &= -\nabla p^n + \frac{\acute{\mathbf{u}}^n}{\Delta t}, \\ r_p^n &= -\frac{\nabla \cdot \acute{\mathbf{u}}^n}{\Delta t} + \frac{1}{Re}\nabla \cdot \sum_{j=1}^{k} \alpha_j (\nabla \times \boldsymbol{\omega}^{n-j}). \end{aligned} \tag{2.48}$$

In (2.47), $\mathbf{u}_b^n$ is the prescribed velocity on all Dirichlet surfaces $\partial\Omega_D$ of the domain, homogeneous Dirichlet conditions are imposed for pressure on outflow surfaces $\partial\Omega_N$, and homogeneous Neumann conditions are imposed for velocity on $\partial\Omega_N$.

We note that in the Navier-Stokes time advancement process summarized above, Step 1 is used to compute a tentative velocity field that depends on the solution at previous time-steps. Only Step 2 includes the solution of a boundary value problem, and thus we can anticipate that time-advancing the solution to the INSE on overlapping subdomains will require us to iterate on this Schwarz subproblem (2.47), which constitutes a system of elliptic PDEs (a Poisson problem for pressure and a Helmholtz equation for each velocity component). We discuss our formulation for solving the INSE on overlapping subdomains in the next chapter.

2.3 Spatial and Temporal Convergence of SEM

We demonstrate the spatial and temporal accuracy of the monodomain SEM formulation by considering the exact Navier-Stokes eigenfunctions in a periodic domain $\Omega = [0, 2\pi]^2$, derived by Walsh [94]. We will

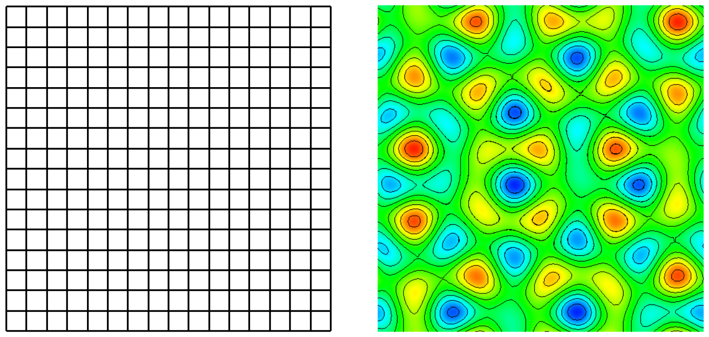


Figure 2.6: (left) Spectral element mesh for discretizing the periodic domain $\Omega = [0, 2\pi]^2$ with 256 elements, and (right) vorticity contours at $T_f = 1$ for $N = 15$.

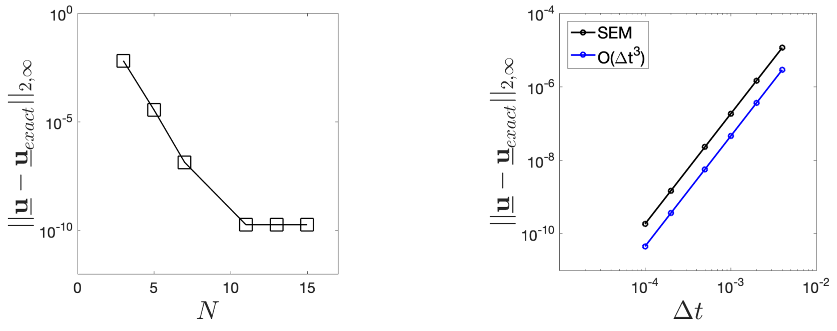


Figure 2.7: (left) Spatial and (right) temporal convergence plots, with final error computed at $T_f = 1$.

also use this example in later sections to demonstrate that our multidomain Schwarz formulation retains the exponential spatial and third-order temporal convergence properties of the monodomain formulation.

Walsh introduced families of eigenfunctions that can be defined using linear combinations of $\cos(px)\cos(qy)$, $\sin(px)\cos(qy)$, $\cos(px)\sin(qy)$, and $\sin(px)\sin(qy)$, for all integer pairs (p, q) satisfying $\lambda = -(p^2 + q^2)$. Taking as an initial condition the eigenfunction $\hat{\mathbf{u}} = (-\tilde{\psi}_y, \tilde{\psi}_x)$, a solution to the NSE is $\mathbf{u} = e^{\nu\lambda t}\hat{\mathbf{u}}(\mathbf{x})$. Here, $\tilde{\psi}$ is the streamfunction resulting from the linear combinations of eigenfunctions. Interesting long-time solutions can be realized by adding a relatively high-speed mean flow $\mathbf{u}_0$ to the eigenfunction, in which case the solution is $\mathbf{u}_{exact} = e^{\nu\lambda t}\hat{\mathbf{u}}[\mathbf{x} - \mathbf{u}_0 t]$, where the brackets imply that the argument is modulo 2π in x and y. As a result, this problem lets us test the advection and diffusion component of NSE.

Figure 2.6 shows the spectral element mesh used for this periodic domain $\Omega = [0, 2\pi]^2$ with 256 equally-sized spectral elements. The flow parameters are $\nu = 0.05$, $\mathbf{u}_0 = (1, 0.3)$, $\tilde{\psi} = (1/5)sin(5y) + (1/5)cos(5x) - (1/4)sin(3x)sin(4y)$, and $\lambda = -25$. The flow is integrated up to time $T_f = 1$ convective time units (CTU) with a fixed $\Delta t = 10^{-4}$. Figure 2.6 also shows the vorticity contours at $T_f = 1$.

Exponential convergence of the velocity error with respect to N is demonstrated in Fig. 2.7. The error is computed as $\underline{\mathbf{e}} = \underline{\mathbf{u}} - \underline{\mathbf{u}}_{exact}$, and the norm is the 2-norm of the point-wise maximum of the vector field, i.e., $||\underline{\mathbf{e}}||_{2,\infty} := ||\tilde{\mathbf{e}}||_2$, where $\tilde{\mathbf{e}} = [||\underline{e}_1||_\infty, ||\underline{e}_2||_\infty]$. For spatial convergence, N is varied from 3 to 15, and Δt is fixed at 10^{-4}. For temporal convergence, $N = 13$ and the solution is integrated up to $T_f = 1$ for various Δt. As we see in Fig. 2.7, exponential convergence with respect to N, and third-order convergence with respect to Δt are achieved using the spatial and temporal discretization described in this chapter.

2.4 Summary

In this chapter, we have described the key ingredients of the SEM, along with how a PDE is transformed from its weak form into a linear system of equations. We have also described the spatial and temporal discretization of the incompressible Navier-Stokes equations along with the different boundary conditions that are used in the monodomain SEM framework. We have also summarized how we advance the solution to the INSE in time. The following chapter will extend the monodomain SEM framework to the Schwarz-SEM framework for overlapping grids.

Chapter 3

Schwarz-SEM for the Incompressible Navier-Stokes Equations

In Section 1.2, we have presented how the overlapping Schwarz method can be used to solve a PDE in overlapping subdomains, and in Section 2.2.3, we have discussed our methodology for time-advancing the solution to the incompressible Navier-Stokes equations in a single conforming domain. In this chapter, we describe the Schwarz-SEM framework for time-advancing the solution of INSE on overlapping subdomains. We also discuss how we improve and integrate *findpts* in the Schwarz-SEM framework to enable use of an arbitrary number of overlapping subdomains.

Important note on notation: In Chapter 2, we used Ω^e to denote the element with (local) element number e in the spectral element mesh used to model Ω. From here on, we will reserve the superscript (typically s) to denote the subdomain number (e.g., Ω^s), unless otherwise stated. We also note that we will use $\Omega^{s,e}$ to represent the spectral element with element number e in Ω^s.

3.1 Schwarz-SEM for Navier-Stokes

The strategy for time-advancing the solution of INSE in overlapping subdomains is similar to that for a single conforming domain. For notational purposes, we introduce $\partial\Omega^s_D := \partial\Omega^s \cap \partial\Omega_D$ as the set of surfaces that have user-prescribed Dirichlet conditions for velocity, $\partial\Omega^s_N := \partial\Omega^s \cap \partial\Omega_N$ as the set of surfaces that have (homogeneous) Neumann conditions for velocity, and $\partial\Omega^s_I := \partial\Omega^s \backslash (\partial\Omega^s_D \cup \partial\Omega^s_N)$ as the interdomain boundary

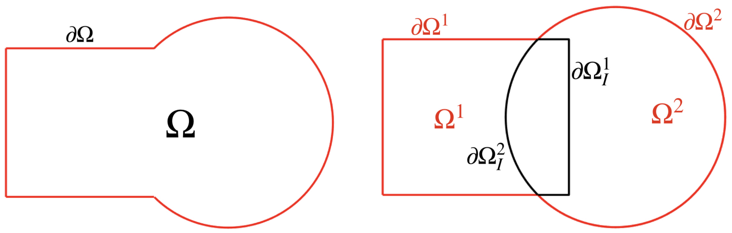


Figure 3.1: (left) Composite domain Ω discretized into (right) overlapping rectangular (Ω^1) and circular (Ω^2) subdomain.

surfaces that obtain their boundary condition via interpolation from an overlapping domain $\Omega^r \neq \Omega^s$. Figure 3.1 shows an example of a composite domain modeled with a rectangular domain overlapping with a circular domain.

Recall from Section 2.2.3 that we use $\boldsymbol{\phi}^{n-1} = [\mathbf{u}^{n-1}, p^{n-1}]^T$ to represent our numerical velocity-pressure solution at time-level t^{n-1}. In the Schwarz context, we further define $\boldsymbol{\phi}^{s,n-1,[q]}$ as the qth iterate of the Schwarz update on subdomain Ω^s at time level t^{n-1}. We typically run with a predefined upper bound on q, which we denote as Q. Thus, assuming that that the solution is known up to time t^{n-1} and has been converged using Schwarz iterations at the previous time-step, $\boldsymbol{\phi}^{s,n-1,[Q]}$ represents our solution at time t^{n-1}. With this notation, and assuming a constant time-step size Δt (which is equal for all overlapping grids), we define the Schwarz update procedure as follows:

1. Compute the tentative velocity field $\acute{\mathbf{u}}$ using (2.28) with the solution from the Q-th Schwarz iteration at k (for BDFk/EXTk scheme) previous time-steps in each subdomain $\Omega^s, s = 1 \dots S$:

$$\acute{\mathbf{u}}^{s,n} \quad = \quad -\sum_{j=1}^{k} \beta_j \mathbf{u}^{s,n-j,[Q]} + \Delta t \sum_{j=1}^{k} \alpha_j (-\mathbf{u} \cdot \nabla \mathbf{u} + \mathbf{f})^{s,n-j,[Q]}, \tag{3.1}$$

where $\acute{\mathbf{u}}^{s,n}$ has contributions from the BDFk and EXTk terms.

2. Solve the *linear* Stokes subproblems (2.29) and (2.32) to compute the velocity-pressure pair, $\boldsymbol{\phi}^{s,n,[q]} = [\mathbf{u}^{s,n,[q]}, p^{s,n,[q]}]^T$ in each subdomain using Q simultaneous Schwarz iterations. We use $q = 0$ to denote the solution that is based on the interdomain boundary data interpolated from the solution at previous time-step, and $q = 1 \dots Q$ to denote the subsequent Schwarz iterations at that time-step:

$$q = 0: \ \mathbf{S}\boldsymbol{\phi}^{s,n,[q]} \quad = \quad \mathbf{r}^{s,n,[q]}, \ \mathbf{u}^n|_{\partial\Omega^s_D} = \mathbf{u}_b^{s,n}, \ \mathbf{u}^n|_{\partial\Omega^s_I} = \mathcal{I}(\mathbf{u}^{r,n-1,[Q]}), \ p^n|_{\partial\Omega^s_N} = 0, \tag{3.2}$$

$$q = 1 \dots Q: \ \mathbf{S}\boldsymbol{\phi}^{s,n,[q]} \quad = \quad \mathbf{r}^{s,n,[q]}, \ \mathbf{u}^n|_{\partial\Omega^s_D} = \mathbf{u}_b^{s,n}, \ \mathbf{u}^n|_{\partial\Omega^s_I} = \mathcal{I}(\mathbf{u}^{r,n,[q-1]}), \ p^n|_{\partial\Omega^s_N} = 0, \tag{3.3}$$

where

$$\begin{aligned}
\mathbf{r}^{s,n,[q]} &= [\mathbf{r}_v^{s,n,[q]}, r_p^{s,n,[q]}]^T, \\
\mathbf{r}_v^{s,n,[q]} &= -\nabla p^{s,n,[q]} + \frac{\acute{\mathbf{u}}^{s,n}}{\Delta t}, \\
r_p^{s,n,[q]} &= -\frac{\nabla \cdot \acute{\mathbf{u}}^{s,n}}{\Delta t} + \frac{1}{Re} \nabla \cdot \sum_{j=1}^{k} \alpha_j (\nabla \times \boldsymbol{\omega}^{s,n-j,[Q]}).
\end{aligned} \tag{3.4}$$

We note that in (3.2) and (3.3), the contribution from $\acute{\mathbf{u}}^{s,n}$ in $\mathbf{r}^{s,n,[q]}$ does not need to be updated at each Schwarz iteration because it depends only on the solution at previous time-steps. Consequently, we do not use the superscript q in $\acute{\mathbf{u}}^{s,n}$. Similarly, the boundary condition for user-specified Dirichlet surfaces $(\partial\Omega^s_D)$

also does not need to be updated at each Schwarz iteration, i.e., $\mathbf{u}^n|_{\partial\Omega^s_D} = \mathbf{u}_b^{s,n}$. Only the contribution from the interdomain boundary ($\partial\Omega^s_I$) needs to be updated at each iteration by interpolating the solution from the overlapping subdomain Ω^r, using the interpolation operator $\mathcal{I}$ that we will explain in the next section.

In (3.2), we interpolate the interdomain boundary data from the solution at the previous time-step, and the resulting solution is, thus, only $\mathcal{O}(\Delta t)$ accurate. In (3.3), the Schwarz iterations $q = 1 \dots Q$ interpolate the solution from most recent iteration and increase the temporal accuracy of the solution (up to order at most k, which is the order of accuracy of the driving BDFk/EXTk temporal discretization). Numerical experiments show that it can take more than $Q = 30$ Schwarz iterations to get temporal accuracy of $\mathcal{O}(\Delta t^3)$. Since each Schwarz iteration requires solution of the pressure Poisson problem and the Helmholtz equation for each component of velocity, one would like to minimize the number of Schwarz iterations needed at each time-step.

Peet and Fischer [55] showed that the temporal accuracy of the solution at the first Schwarz iteration could be increased from $\mathcal{O}(\Delta t)$ to $\mathcal{O}(\Delta t^m)$ ($m \leq k$) by temporally extrapolating the boundary data from m preceding steps. Such an approach can formally elevate the accuracy to the level of the underlying BDFk/EXTk scheme. Stability analysis, however, shows that $m > 1$ leads to an unstable scheme. To recover stability, we consider a predictor-corrector (PC) scheme in which the initial guess for data on $\partial\Omega^s_I$ is based on mth-order temporal extrapolation (from neighboring subdomain Ω^r), and subsequent corrector values are obtained from Schwarz iterations of the linear Stokes system. This (PC) scheme for advancing the solution of INSE in overlapping subdomains is:

1. Compute the tentative velocity field $\acute{\mathbf{u}}$ using (2.28) with the solution from Qth Schwarz iteration at k (for BDFk/EXTk scheme) previous time-steps in each subdomain $\Omega^s, s = 1 \dots S$:

$$\acute{\mathbf{u}}^{s,n} \quad = \quad -\sum_{j=1}^{k} \beta_j \mathbf{u}^{s,n-j,[Q]} + \Delta t \sum_{j=1}^{k} \alpha_j (-\mathbf{u}\cdot\nabla\mathbf{u} + \mathbf{f})^{s,n-j,[Q]}, \tag{3.5}$$

   where $\acute{\mathbf{u}}^{s,n}$ has contributions from the BDFk and EXTk terms.

2. Solve the *linear* Stokes subproblems (2.29) and (2.32) to compute the velocity-pressure pair, $\boldsymbol{\phi}^{s,n,[q]} = [\mathbf{u}^{s,n,[q]}, p^{s,n,[q]}]^T$ in each subdomain using Q simultaneous Schwarz iterations. The initial solution $q = 0$ is now based on the interdomain boundary data extrapolated in time using the solution at previous time-steps, and each subsequent Schwarz iteration ($1 \leq q \leq Q$) uses the solution from the

most recent iteration:

$$q = 0 : \mathbf{S}\boldsymbol{\phi}^{s,n,[0]} \quad = \quad \mathbf{r}^{s,n,[q]},\ \mathbf{u}^n|_{\partial\Omega^s_D} = \mathbf{u}^{s,n}_b,\ \mathbf{u}^n|_{\partial\Omega^s_I} = \mathcal{I}\bigg(\sum_{j=1}^{m} \tilde{\alpha}_j\, \mathbf{u}^{r,n-j,[Q]}\bigg),\ p^n|_{\partial\Omega^s_N} = 0, \quad (3.6)$$

$$q = 1 \dots Q : \mathbf{S}\boldsymbol{\phi}^{s,n,[q]} \quad = \quad \mathbf{r}^{s,n,[q]},\ \mathbf{u}^n|_{\partial\Omega^s_D} = \mathbf{u}^{s,n}_b,\ \mathbf{u}^n|_{\partial\Omega^s_I} = \mathcal{I}(\mathbf{u}^{r,n,[q-1]}),\ p^n|_{\partial\Omega^s_N} = 0, \quad (3.7)$$

where the definition of $\mathbf{r}^{s,n,[q]}$ stays unchanged from (3.4), m is the order of extrapolation for interdomain boundary data, and $\tilde{\alpha}_j$ are the corresponding extrapolation weights.

It is clear that the difference between (3.6) and (3.2) is the use of mth-order extrapolation in time for the interdomain boundary data, which increases the temporal accuracy of the solution and reduces the number of Schwarz iterations needed at each time-step.

Using stability analysis on a 1D model problem, Peet and Fischer demonstrated that $1 \leq Q \leq 3$ was sufficient for ensuring a stable solution when $m > 1$. For the INSE, Merrill et al. [24] demonstrated that this PC approach yields up to third-order temporal accuracy for Schwarz-SEM flow applications. In Chapter 5, we develop a stability analysis framework to empirically analyze the stability of this PC scheme for time-advancing the solution of the INSE in overlapping subdomains, and we also describe a PC scheme that requires fewer Q than (3.6,3.7) to ensure a stable solution.

3.2 Interdomain Boundary Data Interpolation

Interpolation is central to the overlapping Schwarz formulation. For a given subdomain, Ω^s, the essential operation is to find the value of a field $u(\mathbf{x})$ at all points $\mathbf{x}^* \in \partial\Omega^s_I$, where u is to be evaluated in the adjacent subdomain $\Omega^r \neq \Omega^s$. Given the spectral element representation for the function u (2.5) and spatial coordinates $\mathbf{x}$ (2.6) of the mesh in the physical space, the task at hand is to use (2.6) to identify *computational* coordinates, $(e^*, \xi^*, \eta^*, \zeta^*)$, which indicate the element Ω^{r,e^*} that $\mathbf{x}^*$ overlaps, and the reference-space coordinates $\boldsymbol{\xi}^* = \{\xi^*, \eta^*, \zeta^*\}$ inside Ω^{r,e^*}. With these computational coordinates, (2.5) can be used to evaluate $u^* = u(\mathbf{x}^*)$[1].

Figure 3.2 shows the example from Chapter 1 with a rectangular domain overlapping with a circular domain. The overlap between the two domains is nontrivial, and grid points discretizing the interdomain boundary of each subdomain will rely on the corresponding overlapping subdomain for boundary conditions. Thus, the first challenge for OS-based methods is to accurately identify the donor element and reference-space coordinates ($\boldsymbol{\xi} = \xi, \eta, \zeta$) inside that element, for each interdomain boundary point $\mathbf{x}^* \in \partial\Omega^s_I$, as shown

[1] Technically, according to (2.6)–(2.5), it would be $u^* = u(\boldsymbol{\xi}^*(\mathbf{x}^*))$, but we will use the less cumbersome form $u^* = u(\mathbf{x}^*)$.

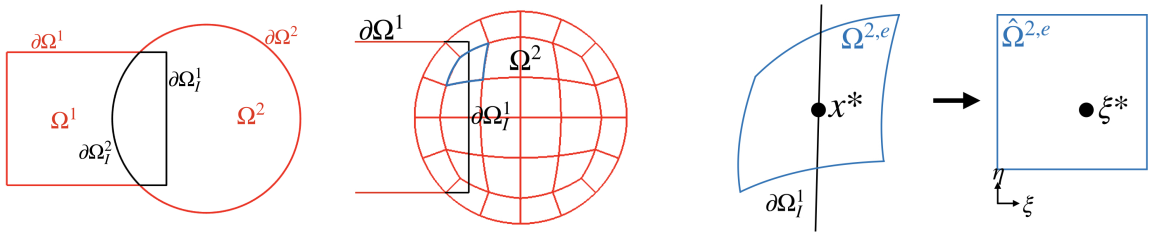


Figure 3.2: (left) Overlapping rectangular (Ω^1) and circular (Ω^2) subdomains, (center) interdomain boundary $\partial\Omega^1_I$ overlapping spectral elements of Ω^2, and (right) grid point $\mathbf{x}^* \in \partial\Omega^1_I$ overlapping the element $\Omega^{2,e}$.

in Fig. 1.8. Since each subdomain is partitioned onto many processors, we also need to know the process p owning the donor element e corresponding to each interdomain boundary grid point. Using $q^*_j = (p^*, e^*, \boldsymbol{\xi}^*)_j$ for each grid point $(\mathbf{x}^*_j)$, we can interpolate any scalar function using (2.5). In our framework, speed and accuracy for donor-element identification and interpolation, comes via integration of *findptslib*, a set of high-order interpolation routines in *gslib*.

3.2.1 High-order interpolation in a single conforming mesh using *findpts*

findptslib was originally developed for data interrogation and Lagrangian particle tracking in a single conforming domain for up to $P = 10^6$ processors. High fidelity interpolation for highly curved elements, like the ones supported by SEM, is quite challenging. Thus, *findptslib* was designed with the principles of robustness (i.e., it should never fail) and speed (i.e., it should be fast). Here, we summarize how *findpts* enables high-order interpolation in spectral element meshes using *findpts* and *findpts_eval*, and the reader is referred to [44] for a detailed discussion on *findptslib*.

findptslib provides two key capabilities. Using *findpts*, it first determines computational coordinates $\mathbf{q}^*_j = (e^*, p^*, \xi^*, \eta^*, \zeta^*)_j$ (local element number e^* on process p^*, and reference-space coordinates $\boldsymbol{\xi}^* = (\xi^*, \eta^*, \zeta^*)$) for any given set of points $\underline{\mathbf{x}}^* = (\mathbf{x}^*_1, \mathbf{x}^*_2 \ldots \mathbf{x}^*_b)$, where b is the number of points to be found, and $\mathbf{x}^*_j = (x^*, y^*, z^*)_j \in \mathbb{R}^3$. To find $\mathbf{q}^*$ for a given point $\mathbf{x}^*$, *findpts* first uses a hash table to identify processors that could potentially have the element inside which the target point $\mathbf{x}^*$ is located. A call to *gs_crystal* then exchanges copies of the $\mathbf{x}^*$ entries between the source processor that is looking for $\mathbf{x}^*$, and potential processors. Once there, element-wise bounding boxes further discriminate to determine potential elements on each processor that could contain $\mathbf{x}^*$. At that point, a trust-region based Newton optimization is used to solve $\boldsymbol{\xi}^* = \text{argmin}||\mathbf{x}^e(\boldsymbol{\xi}) - \mathbf{x}^*||^2$, where $\mathbf{x}^e(\boldsymbol{\xi})$ represents the isoparametric mapping from $\hat{\Omega}^e$ to Ω^e. Once the Newton search is complete and $\mathbf{x}^*$ is found inside element Ω^{e^*}, its computational coordinates $\mathbf{q}^*$ are returned to the source processor. Additionally, *findpts* also indicates whether a point was found inside an element, on the edge/surface of an element, or if it was not found within the mesh. We note that due to the use of local form (2.5) for representing functions, and the use of the private memory model (Section 2.1.5), *findpts* requires the local element number e^* (not the global element number) on the process p^* on which the point is found. The second key feature of *findptslib* is the routine *findptslib* that interpolates any given field for a given set of computational coordinates (determined from *findpts*) using (2.5).

Because interpolation is nonlocal (a point $\mathbf{x}_j^*$ may originate from any processor), *findpts* and *findpts_eval* require interprocessor coordination and must be called by all processors in a given communicator. For efficiency reasons, interpolation calls are typically batched with all queries posted in a single call, but the work can become serialized if a single processor ultimately owns all target points. A detail discussion of *findpts* is in [44].

3.2.2 *findpts* for Schwarz-SEM

Since *findpts* has demonstrated excellent scaling in parallel for finding computational coordinates of a given point [95], it is natural to use *findpts* for determining computational coordinates in the context of the Schwarz-SEM framework. However, we must keep in mind that *findpts* was originally developed for monodomain meshes, and cannot discriminate between elements of different subdomains. In order to integrate *findpts* in the Schwarz-SEM framework, we have to address two key issues.

First, to find donor elements for grid points on $\partial\Omega_I^s$, we want to ignore the elements in Ω^s. Second, in case of multiple overlapping meshes ($S > 2$), we want to pick the donor element from the subdomain that minimizes error due to Schwarz iterations. Consider the example for $S = 3$ overlapping domains, shown in Fig. 3.3, where we illustrate the meshes for Ω^1 (red) and Ω^3 (black), and the subdomain boundary of Ω^2 ($\partial\Omega_I^2$ is shown in green). In order to determine donor elements for a grid point $\mathbf{x}^* \in \partial\Omega_I^2$, we want to ignore the elements in Ω^2, and pick the donor element from Ω^1 or Ω^3 that minimizes the error associated with Schwarz iteration. To address these issues, we have added discriminators to *findpts* to enable donor element search in an arbitrary number of overlapping grids.

For the Schwarz-SEM framework, donor element search is done simultaneously for all subdomains. All elements in each subdomain Ω^s are tagged with the subdomain number s, and this subdomain number is passed to *findpts* as a discriminator. When *findpts* is used to find the donor element for an interdomain boundary grid point $\mathbf{x}^* \in \partial\Omega^r$, the subdomain number r is passed to *findpts* along with $\mathbf{x}^*$. *findpts* then uses the subdomain number r of the queried point and the subdomain number of spectral elements to only search elements that are in $\Omega\backslash\Omega^r$. This subdomain number based discriminator is sufficient for determining donor elements when there are only two subdomains ($S = 2$).

For $S > 2$, it is still possible to have multiple subdomains claim ownership of a given boundary point

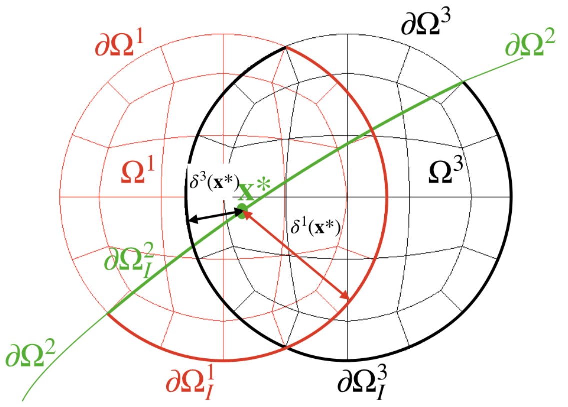


Figure 3.3: For determining the donor element for $\mathbf{x}^* \in \partial\Omega_I^2$., *findpts* ignores elements in Ω^2 and considers distance from the interface boundaries ($\delta^1(\mathbf{x}^*)$ in Ω^1 and $\delta^3(\mathbf{x}^*)$ in Ω^3). For clarity, we have shown only the domain boundary of Ω^2 overlapping with subdomains Ω^1 and Ω^3.

$\mathbf{x}^*$. To resolve such conflicts, we associate with each subdomain Ω^s a local distance function, $\delta^s(\mathbf{x})$, which indicates the minimum distance from any point $\mathbf{x} \in \Omega^s$ to $\partial\Omega_I^s$. The owner of any boundary point $\mathbf{x}^* \in \partial\Omega_I^r$ among two or more domains Ω^s, $s \neq r$, is taken to be the domain that maximizes $\delta^s(\mathbf{x}^*)$. This choice is motivated by the standard Schwarz arguments, which imply that errors decay *away* from the interface, in accordance with the decay of the associated Green's functions (Section 1.2). We illustrate this situation in Fig. 3.3, where $\mathbf{x}^* \in \partial\Omega_I^2$ is interior to both Ω^1 and Ω^3. In this case, interpolated values (from *findpts_eval*) will come from Ω^1 because $\mathbf{x}^*$ is "more interior" to Ω^1 than Ω^3.

By adding the subdomain number and distance function based discriminator to *findpts*, we are able to use it for identifying computational coordinates for interdomain boundary grid points in the Schwarz-SEM framework. The distance function δ is essential for integrating *findpts* in the Schwarz-SEM framework, and we will describe our method for generating this distance function in Section 3.4.

3.2.3 Improvement to high-order interpolation in *findpts_eval*

Since *findpts_eval* relies on computational coordinates for interpolation, no changes are required to use *findpts_eval* for interpolating interdomain boundary data in overlapping grids. Here, we describe how we have reduced the computational cost of the interpolation routine *findpts_eval*.

In the original implementation, *findpts_eval* interpolates a given scalar field at a set of computational coordinates by sending computational coordinates ($\mathbf{q}^*$) to the processor (process number p^*) on which each point ($\mathbf{x}^*$) is found. Each processor then sorts the list of points that it has received, element-wise, in order to efficiently compute Lagrange coefficients for interpolation. This sorted list of points is then used to interpolate the given scalar field, element-by-element, and the interpolated values are sent back to the source processor that had originally issued the query.

In the Schwarz-SEM framework, *findpts_eval* is called at each Schwarz iteration multiple times, and using the original *findpts_eval* implementation leads to unnecessary sorting and communication at each iteration for the same list of computational coordinates. To reduce the computational cost, we have improved *findpts_eval* such that it saves the list of computational coordinates (sorted element-wise) on the processor that the point is found on, instead of the source processor that owns the point. As a result, every time *findpts_eval* is called, the scalar field is directly interpolated and communicated to the source processor that has the interdomain boundary grid point. With this improvement, we have observed a 3X speed-up in performance of *findpts_eval*. The performance result comparing the original and improved *findpts_eval* are presented in Section 8.3, where we discuss the timing and scaling for different components of the Schwarz-SEM framework.

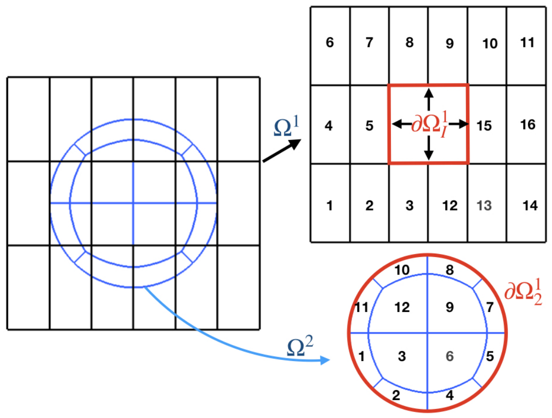
3.3 Multidomain Partitioning

Section 3.2.2 and 3.2.3 describe the changes that we have made to *findptslib* to integrate it in the Schwarz-SEM framework. With the improvements that we have made to *findpts* for enabling donor element search in overlapping subdomains, the extension of the monodomain SEM framework to the multidomain (Schwarz-SEM) framework is straightforward for parallel implementation.

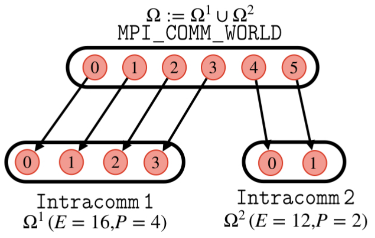
In the multidomain case with S subdomains, all MPI ranks are initialized within a single communicator (`MPI_COMM_WORLD`). These MPI ranks are then partitioned into S subsets, each with their own MPI communicator (`Intracomm`), and elements in Ω^s are distributed across MPI ranks within `Intracomm` s. With this partitioning strategy, each subdomain s individually advances the solution to INSE, using *gslib* at the level of its local subdomain communicator `Intracomm` s to effect the gather-scatter operator.

Since boundary conditions for interdomain boundaries rely on overlapping grids, however, as described in Section 3.1, *findpts* cannot be used at the level of the local subdomain communicator `Intracomm` s. Instead, *findpts* is used at the level of `MPI_COMM_WORLD` for computational coordinate search, as `MPI_COMM_WORLD` allows communication between MPI ranks associated with different subdomains. To demonstrate how MPI ranks are partitioned for multidomain cases at the level of the global communicator, `MPI_COMM_WORLD`, and local subdomain communicator, `Intracomm`, we consider a simple example of $S = 2$ overlapping grids.

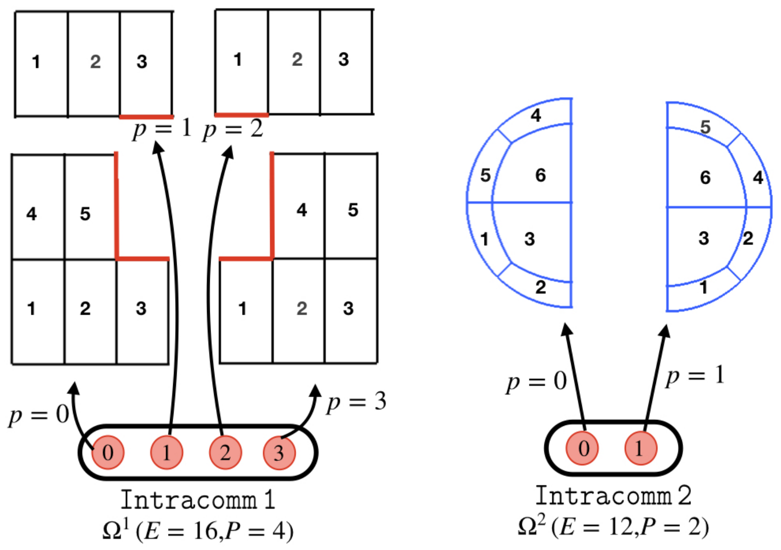
Fig. 3.4(a) shows an example of overlapping grids with a background mesh that has a hole in the center and a circular mesh which overlaps the background mesh. The interdomain boundaries $\partial\Omega_I^s$ for each mesh are also indicated in the figure. For this example, we consider that $P = 4$ for Ω^1, which has $E = 16$ spectral elements, and $P = 2$ for Ω^2, which has $E = 12$ spectral elements. In the Schwarz-SEM framework, all



(a) Overlapping spectral element meshes.


$\Omega := \Omega^1 \cup \Omega^2$
MPI_COMM_WORLD
Intracomm 1
$\Omega^1\,(E = 16, P = 4)$
Intracomm 2
$\Omega^2\,(E = 12, P = 2)$

(b) Partitioning of MPI ranks from `MPI_COMM_WORLD` to `Intracomm`.



(c) Partitioning of each spectral element mesh to MPI ranks.

Figure 3.4: Pictorial representation of two overlapping grids partitioned on to their own MPI communicators.

the 6 MPI ranks will, thus, be initialized within a single communicator (`MPI_COMM_WORLD`), as shown in Fig. 3.4(b). The 6 MPI ranks will then be split into $S = 2$ subsets, and each subset will be assigned its own MPI communicator, `Intracomm` s. Elements of each subdomain s will then be partitioned onto MPI ranks in `Intracomm` s using the model described in Fig. 2.5. Figure 3.4(c) shows the two overlapping grids partitioned onto their own MPI ranks, along with the local element numbering on each MPI rank. Since *findpts* needs to effect interdomain communication at the level of global communicator (`MPI_COMM_WORLD`), and *gslib* needs to effect the gather-scatter operator at the level of local communicator (`Intracomm` s for each subdomain Ω^s), we need a mapping, similar to (2.25), telling how each spectral element of each subdomain maps to the local and global communicator.

Using $\underline{e}_g^s$ to denote the global element number for elements in Ω^s, $\underline{p}_w^s$ to denote the vector that tells what MPI rank in `MPI_COMM_WORLD` each element is mapped to, $\underline{p}_l^s$ to denote the MPI rank in `Intracomm` s that each element is mapped to, and $\underline{e}_l^s$ to denote the local element number on the MPI rank that the element is on, the mapping for the example in Fig. 3.4 is

$$\begin{aligned}
\underline{e}_g^1 &= [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16], \leftarrow \text{global element number} \\
\underline{p}_w^1 &= [0,0,0,0,0,1,1,1,2,2,2,3,3,3,3,3], \leftarrow \text{MPI rank at } \texttt{MPI\_COMM\_WORLD} \text{ level} \\
\underline{p}_l^1 &= [0,0,0,0,0,1,1,1,2,2,2,3,3,3,3,3], \leftarrow \text{MPI rank at } \texttt{Intracomm} \text{ 1 level} \\
\underline{e}_l^1 &= [1,2,3,4,5,1,2,3,1,2,3,1,2,3,4,5], \leftarrow \text{local element number}
\end{aligned} \tag{3.8}$$

for Ω^1, and

$$\begin{aligned}
\underline{e}_g^2 &= [1,2,3,4,5,6,7,8,9,10,11,12], \leftarrow \text{global element number} \\
\underline{p}_w^2 &= [4,4,4,4,4,4,5,5,5,5,5,5], \leftarrow \text{MPI rank at } \texttt{MPI\_COMM\_WORLD} \text{ level} \\
\underline{p}_l^2 &= [0,0,0,0,0,0,1,1,1,1,1,1], \leftarrow \text{MPI rank at } \texttt{Intracomm} \text{ 2 level} \\
\underline{e}_l^2 &= [1,2,3,4,5,6,1,2,3,4,5,6], \leftarrow \text{local element number}
\end{aligned} \tag{3.9}$$

for Ω^2. This approach allows us to map each element at the level of `MPI_COMM_WORLD` for interdomain boundary data exchange, and at the level of `Intracomm` s for the unsteady Stokes solve.

Using this approach, the single domain SEM framework (100K lines of code) has been extended to the Schwarz-SEM framework for multiple overlapping meshes, with minimal change.

3.4 Distance Field Generator

Distance fields are needed in multiple contexts within our Schwarz INSE solver and with INSE solvers, in general. First, as noted earlier in Section 3.2.2, we use distance from interior domain boundaries, $\partial\Omega_I^s$, as a discriminator in cases where multiple neighboring domains, $\Omega^r \neq \Omega^s$, share a given point on $\partial\Omega_I^s$. (We favor the more interior point, that is, the one for which the distance function is largest.) Second, as we describe in Section 4.3, distance functions are useful for efficient generation of partition-of-unity functions that are critical for computing global integrals (to determine mass flux, etc.). Third, *global* distance functions from an object (such as a cylinder or other subset of $\partial\Omega$) are often required in, for example, RANS formulations based on wall models. In this latter case, coupling between overlapping subdomains is required to accurately compute the distance function from the object of interest.

Consider the example of Fig. 3.5, which shows the monodomain spectral element mesh and overlapping spectral element meshes for modeling flow over an oscillating cylinder, using an unsteady RANS formulation. To allow the cylinder to oscillate in the monodomain case, an ALE formulation is used to move the GLL points for all elements. In contrast, the Schwarz-SEM framework allows us to model this domain using a static background mesh, overlapped with an oscillating mesh for the cylinder, thus eliminating the need for ALE. Using an unsteady RANS formulation, however, requires the distance from the cylinder, which is not readily available for the background grid since it is not connected to the cylinder. Thus, we need a mechanism for coupling the overlapping grids in order to accurately generate a distance field for the entire domain.

In this section, we first describe our methodology for determining the distance field in a single conforming mesh. This methodology is used in the overlapping grid framework to generate the distance field on each overlapping subdomain to determine the distance from the interdomain boundaries, which is used as a discriminator in *findpts* (Section 3.2.2), and in generating partition-of-unity functions for global integration (Section 4.3). Next, we extend this monodomain spectral element mesh based distance function generator to the Schwarz-SEM framework using simultaneous Schwarz iterations (1.2), for applications where a global distance field is needed based on a surface that could be located in any of the overlapping subdomains.

For a single conforming mesh, we construct the distance field based on the distance between neighboring GLL points. The distance field $\underline{\delta}$ is initialized by assigning a very large number (e.g., domain diameter that can be computed as the maximum length of the domain in all space directions) to all the GLL points in the interior and 0 to all the GLL points on the domain boundary of interest. An iterative loop then updates the distance of all GLL points. For each GLL point i in the spectral element mesh, we set $\delta_i = \min\limits_{\forall j \in \mathcal{N}_i} (\delta_i^0, \delta_i^0 + d_{ij})$, where δ_i^0 is the estimate of the distance at that GLL point from previous iteration, $\mathcal{N}_i$ is the set of GLL points that are connected to the GLL point with index i, and d_{ij} is the Euclidean distance between the GLL point i and GLL point j in its neighborhood. After each iteration, *gslib* communicates information between

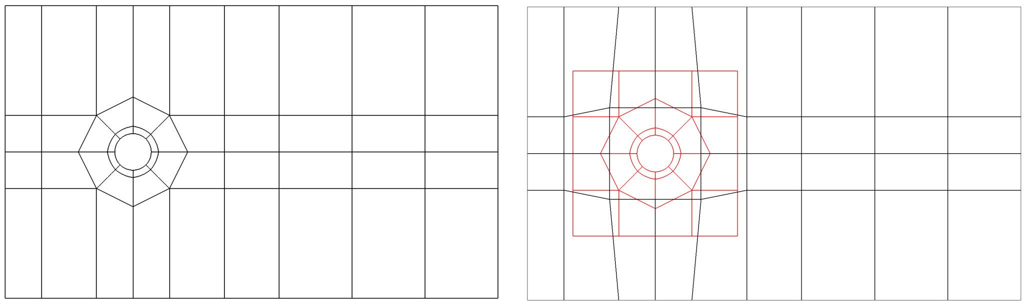


Figure 3.5: (left) Monodomain and (right) overlapping spectral element meshes for flow over an oscillating cylinder.

GLL points shared between elements (which could be located on different processors). The iterative loop is terminated when the distance is not updated for any GLL point in a mesh during the most recent iteration.

Figure 3.6(left) shows the distance field computed in the monodomain mesh using the approach described above, based on the distance from the cylinder. As expected, the distance field is 0 at the cylinder and increases away from the cylinder. We discuss the accuracy of this approach later in this section.

The extension of the distance field generator from monodomain SEM to Schwarz-SEM is straightforward using simultaneous Schwarz iterations (1.2). The distance field is initialized to 0 at the boundary of interest (e.g., cylinder surface) and interdomain boundaries, and to a large value everywhere else. An iterative loop, the same as that used in the monodomain grid framework, is used to update the distance field in each subdomain. Once the distance field converges on each subdomain, interdomain boundary data is exchanged between overlapping grids via *findpts* and *findpts_eval*. The iterative loop is then again used to update the distance field in each subdomain, before exchanging the interdomain boundary data. These Schwarz iterations are used until the distance field converges to a desired tolerance on the interdomain boundaries.

Figure 3.6(right) shows the distance field generated using the simultaneous Schwarz iterations on overlapping grids, and we can see a good comparison in the distance field generated for the monodomain grid and overlapping grids. In this example, six Schwarz iterations were needed to compute the distance field with a relative tolerance of 10^{-4} for the interdomain boundaries.

Fig. 3.7 shows the error in distance field computed using the nearest-neighbor approach for monodomain and overlapping grids. While the exact distance field is not trivial to compute for complicated domains, we can compute it for the geometry considered here since the cylinder of radius 0.5 is centered at the origin. Thus, $\delta_{exact}(\mathbf{x}) = \sqrt{x^2+y^2} - 0.5$, and we compute the error as $\underline{e} = \underline{\delta} - \underline{\delta}_{exact}$. Our nearest-neighbor based approach gives a good approximation of the distance field for GLL points near the cylinder surface, in the monodomain SEM framework. The use of simultaneous Schwarz iterations readily extends this method to overlapping grids, and we see a good agreement in the results for the monodomain and the multidomain

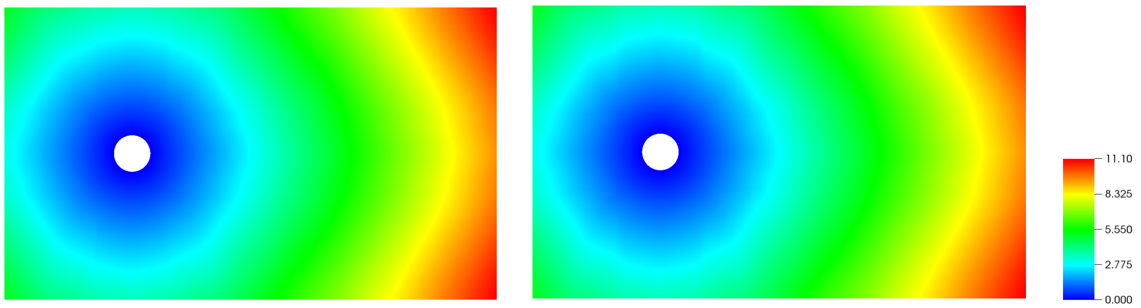


Figure 3.6: Distance field for the (left) monodomain mesh and (right) overlapping meshes, based on the distance from the cylinder.

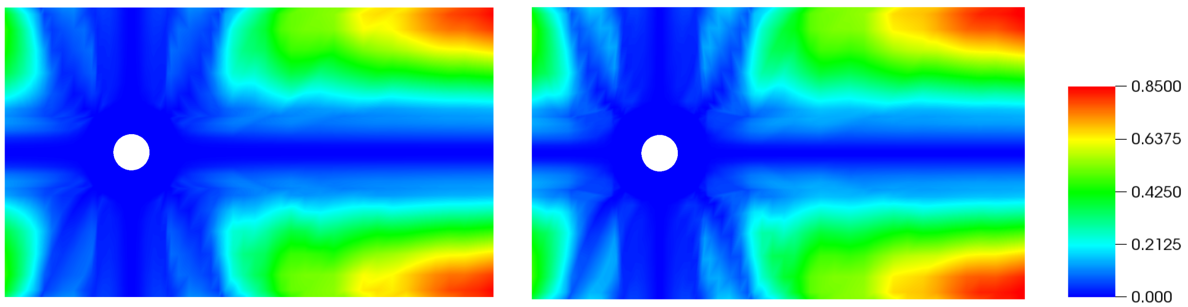


Figure 3.7: Error in distance field ($\underline{e} = \underline{\delta} - \underline{\delta}_{exact}$) computed using the nearest-neighbor approach discussed above for the (left) monodomain mesh and (right) the overlapping spectral element meshes.

case.

We note that in our experience, the nearest-neighbor based approach for distance function generation has been sufficiently accurate for the applications (Section 3.2.2 and Section 4.3) that we consider here. It is clear from Fig. 3.7, however, that a drawback of our methodology is that the error in the distance field increases away from the surface of interest, where the mesh geodesics are not aligned with the *true* distance field (which increases radially away from the cylinder, for the example considered here). This dependence of accuracy on mesh geodesics is not desirable since mesh quality impacts the error in the distance function, and this error cannot be reduced by a simple h- or p-type refinement.

In future work, we will look at more accurate ways of generating the distance function. One of the options that we plan to explore is solving the time-dependent Eikonal equation (for e.g., [96])

$$\frac{\partial \delta}{\partial \tau} + \left(\tilde{T}(\delta) \frac{\nabla \delta}{|\nabla \delta|} \right) \cdot \nabla \delta = \tilde{T}(\delta), \text{where } \tilde{T}(\delta) = \frac{\delta}{\sqrt{\delta^2 + |\nabla \delta|^2 h(\mathbf{x})}}, \tag{3.10}$$

for generating the distance field. In (3.10), $h(\mathbf{x})$ is a mesh dependent parameter. Starting with an approximation of the distance function, (3.10) can be time-advanced to steady state to get the exact distance function. Preliminary studies show that the nonlinear nature of (3.10) coupled with lack of smoothness in distance functions complicates the solution process, and we aim to develop fast and stable methods for solving (3.10) in future work. We note that methods such as the fast marching method [97] or fast sweeping method [98] are popular in the literature for generating the distance function, but they do not readily extend to the high-order SEM framework that we present here.

3.5 Validating the Schwarz-SEM Framework

With integration of *findptslib* in the Schwarz-SEM framework, we can now solve the incompressible Navier-Stokes equations in an arbitrary number of overlapping subdomains. In order to validate our Schwarz-SEM

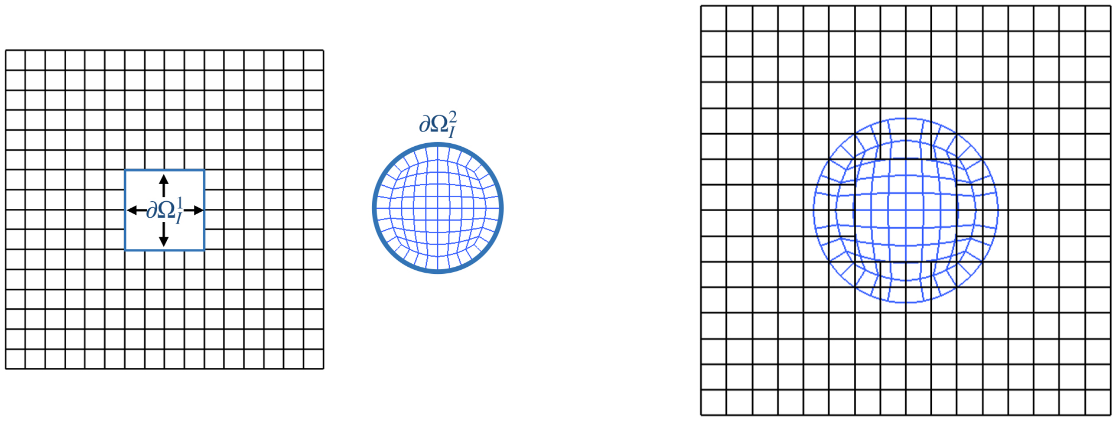


Figure 3.8: (left) Spectral element mesh for each overlapping subdomain. The background mesh has 240 elements, and is covered by a circular mesh with 96 elements. (right) Overlapped background and circular mesh.

framework, we use Nek5000, an open-source SEM-based incompressible flow solver that scales to millions of MPI ranks. It is used by researchers around the world to study turbulent flow phenomenon at high Reynolds number [81, 82, 83]. All the methods presented in this dissertation have been implemented and validated in Nek5000. Additionally, most of the theoretical developments presented here have already been made available open-source through the Nek5000 Github repo [99].

Here, we revisit the example from Section 2.3, where we demonstrated exponential convergence with N, and third-order temporal convergence with Δt for the monodomain SEM framework, using the Navier-Stokes eigenfunctions by Walsh. We consider two cases with the Schwarz-SEM framework. In the first case, we use two overlapping grids ($S = 2$), shown in Fig. 3.8(left). The periodic background mesh is modified from the monodomain case such that it has a hole in the center, with a total of 240 elements. A circular mesh with 96 elements is overlapped with the background mesh to cover the hole, as shown in Fig. 3.8(right).

In the second case, we use three overlapping grids to test the multidomain ($S > 2$) capability of the Schwarz-SEM framework. The background mesh is the same from $S = 2$ case, but now we use two circular meshes to cover the hole in the background mesh. The three overlapping grids are shown in Fig. 3.9.

The initial conditions, boundary conditions, and parameters such as time-step size (Δt), polynomial order (N) and Reynolds number (Re) are the same as for the monodomain case. We set $m = 3$ and $Q = 1$ for both, the $S = 2$ and $S = 3$ cases. Figure 3.10 shows the exponential convergence with N (left) and third-order temporal convergence (right) for the two configurations considered here. The results in Fig. 3.10 indicate that the improvements that we have made to *findpts* support multidomain calculations ($S > 2$) in the Schwarz-SEM framework.

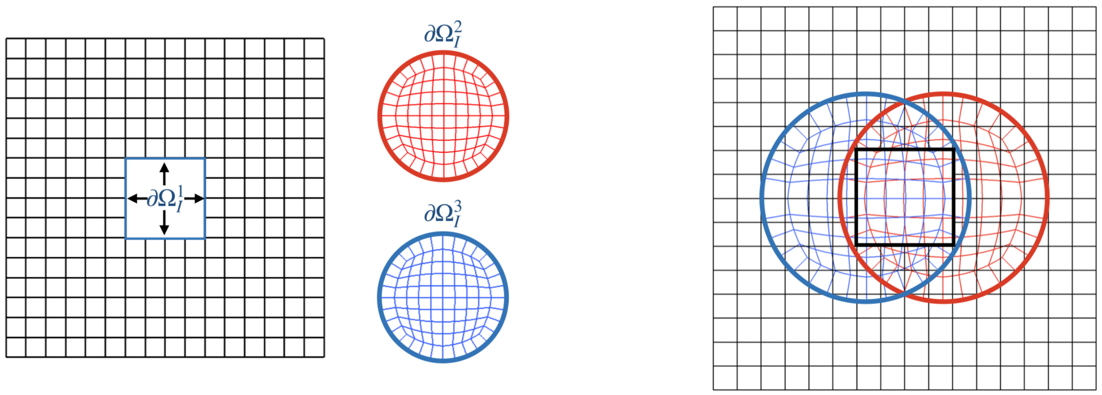


Figure 3.9: (left) Spectral element mesh for each overlapping subdomain. The background mesh has 240 elements, and is covered by two circular meshes, each with 96 elements. (right) Overlapped background and circular meshes.

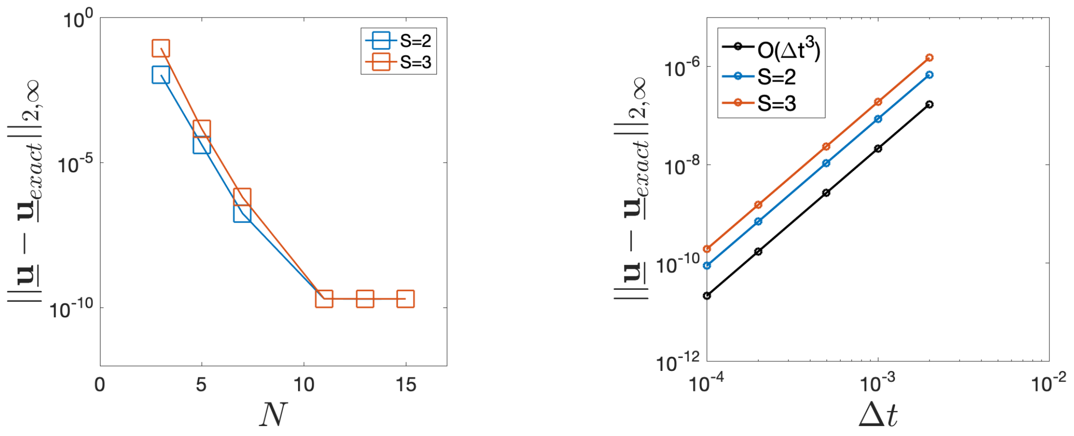


Figure 3.10: (left) Spatial and (right) temporal convergence for the multidomain case with $S = 2$ and $S = 3$ overlapping grids.

3.6 Summary

In this chapter, we described the key ingredients of the Schwarz-SEM framework that we have developed. We presented our methodology for extending the high-order interpolation library *findptslib* by adding a pair of discriminators for computational coordinate identification in *findpts* with an arbitrary number of overlapping subdomains. These discriminators are based on the subdomain number of each overlapping mesh, and the distance of GLL points in each mesh from its interdomain boundary. We have also identified ways to improve the computational complexity of *findpts_eval*, the high-order interpolation routine in *findptslib*. We empirically showed that using the extension of *findptslib* to multidomain calculations, we are able to maintain the exponential convergence with polynomial order N, and third-order temporal convergence of the underlying SEM-based incompressible Navier-Stokes solver.

Chapter 4

Mass Conservation and Fixed Flow Rate through Overlapping Subdomains

In this chapter, we describe the impact of interpolating interdomain boundary data on the net mass flux in the domain. Using a simple 2D example, we describe a velocity correction scheme that ensures that the boundary conditions for each unsteady Stokes solve are consistent. We also discuss our method for maintaining a fixed mass flow rate through overlapping subdomains, a capability crucial for modeling internal flows in periodic domains. This method also brings forth the need for a framework for global integration on overlapping subdomains, which is not as trivial as in a monodomain grid. Overlapping grid-based methods require that grid points in the overlap region are weighted appropriately for integration, and we have developed a novel method for generating partition-of-unity functions in subdomains with arbitrary overlap.

4.1 Mass Conservation

Accurate treatment of boundary conditions is crucial for the stability and accuracy of numerical simulations. For the incompressible Navier-Stokes equations, the pressure solution in the domain is tightly coupled to the divergence-free constraint, and violation of mass balance can lead to spurious oscillations in the pressure solution. Thus, mass conservation is necessary to ensure stability and accuracy in the SEM. This requires special care in the Schwarz-SEM framework because we must ensure that the boundary conditions enforced on $\partial\Omega_I^s$ are consistent with the boundary conditions on $\partial\Omega^s \backslash \partial\Omega_I^s$, i.e., the net mass flux through $\partial\Omega^s$ should be zero.

Let us consider the channel with constrictions shown in Fig. 4.1, which is modeled using two overlapping domains. The domain on the left (Ω^1) has inhomogeneous Dirichlet boundary conditions on the inflow surface on the left, and homogeneous Dirichlet boundary condition on the top, bottom and constriction walls. The interdomain boundary ($\partial\Omega_I^1$) is also Dirichlet for velocity, and the boundary data for $\partial\Omega_I^1$ is interpolated from the domain on the right (Ω^2). For Ω^2, there is a Neumann boundary condition for velocity via outflow on the right boundary and an interdomain boundary ($\partial\Omega_I^2$) on the other end. Since Ω^2 has an outflow on the right boundary, the net mass flux through the subdomain boundaries in Ω^2 is automatically

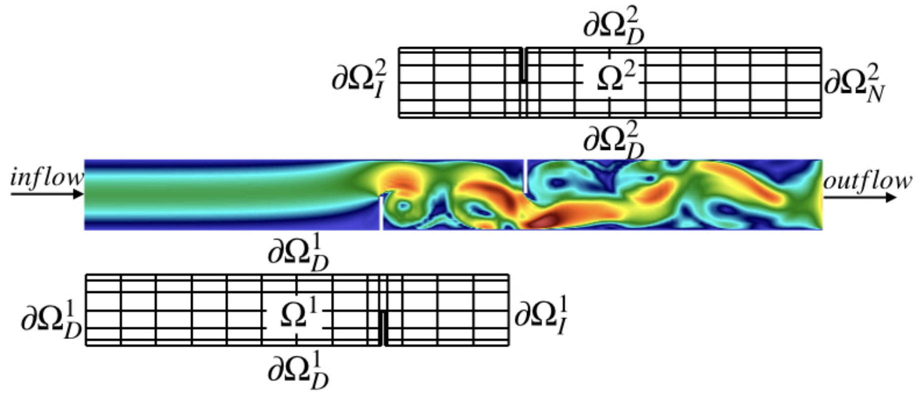


Figure 4.1: Velocity magnitude of flow through a channel with constrictions, along with overlapping meshes used to model the domain.

balanced, i.e., $\int_{\partial\Omega^2} \mathbf{u}\cdot\hat{\mathbf{n}} = 0$. For Ω^1, however, there is no guarantee that the net mass flux through $\partial\Omega_I^1$ will balance the mass inflow through the inflow boundary ($\partial\Omega_D^1$). Numerical experiments show that the mass flux imbalance across Ω^1 can lead to numerical instabilities, and we use this example to derive a velocity correction scheme to address the mass imbalance due to interpolation for interdomain boundary data.

For each subdomain Ω^s, the mass conservation statement is

$$\int_{\partial\Omega^s} \mathbf{u}\cdot\hat{\mathbf{n}}\, dA \quad = \quad 0, \tag{4.1}$$

where $\hat{\mathbf{n}}$ represents the outward pointing unit normal vector on $\partial\Omega^s$. (4.1) is a compatibility condition for the incompressible NSE and must be satisfied at each time-step. Even though we use high-order interpolation (order N) to obtain interdomain boundary data, there is no guarantee that (4.1) is exactly satisfied. This issue is especially exacerbated when grids of varying resolution are used. Our goal is to find a nearby correction to the interpolated velocity field ($\hat{\mathbf{u}}$) such that it satisfies the compatibility condition.

Let $\partial\Omega_D$ denote the subset of the domain boundary $\partial\Omega$ corresponding to Dirichlet velocity conditions and $\partial\Omega_N$ be the Neumann (outflow) subset. If $\partial\Omega_N \cap \partial\Omega^s = 0$ (as is the case for Ω^1), then there is a potential to fail to satisfy (4.1) because the interpolated fluxes on $\partial\Omega^s$ may not integrate to zero. Let $\hat{\mathbf{u}}$ denote the tentative velocity field defined on $\partial\Omega^s$ through prescribed data on $\partial\Omega_D^s := \partial\Omega^s \cap \partial\Omega_D$ and interpolation on $\partial\Omega_I^s := \partial\Omega^s \backslash \partial\Omega_D^s$. Assuming that $\upsilon = \int_{\partial\Omega^s} \hat{\mathbf{u}}\cdot\hat{\mathbf{n}}\, dA$ is the net mass flux through the domain boundary, our goal is to find a correction $\tilde{\mathbf{u}}$ to the interdomain boundary data such that $\mathbf{u} = \hat{\mathbf{u}} + \tilde{\mathbf{u}}$ and $\int_{\partial\Omega^s} \mathbf{u}\cdot\hat{\mathbf{n}} = 0$. Since we do not want to modify the prescribed boundary data on $\partial\Omega_D^s$, we set $\tilde{\mathbf{u}}|_{\partial\Omega_D^s} = 0$. The velocity field correction problem is, thus, represented as

$$\underset{\tilde{\mathbf{u}}}{\arg\min} \qquad \|\mathbf{u} - \hat{\mathbf{u}}\|_T^2 \qquad \text{subject to} \quad \int_{\partial\Omega^s} \mathbf{u}\cdot\hat{\mathbf{n}}\, dA = 0 \qquad \text{and } \tilde{\mathbf{u}}|_{\partial\Omega_D^s} = 0, \tag{4.2}$$

where $||\mathbf{u}-\hat{\mathbf{u}}||_T$ is the trace norm (or L^2 norm in the trace space) in the space of functions used to discretely represent $\mathbf{u}$ and $\hat{\mathbf{u}}$ in the SEM. The advantage of (4.2) is that it maintains the prescribed boundary data on $\partial\Omega^s_D$, and projects the boundary data on $\partial\Omega^s_I$ to a divergence-free space such that $\int_{\partial\Omega^s} \mathbf{u}\cdot\hat{\mathbf{n}}\,dA = 0$.

To define our velocity correction in the Schwarz-SEM framework, we introduce the following notation: $\underline{\hat{\mathbf{u}}}$ represents the discrete velocity field on $\partial\Omega^s$, $\underline{\tilde{\mathbf{u}}}$ represents the velocity field on $\partial\Omega^s_I$, R is the standard restriction operator of size $n_I \times n_O$ that restricts data from $\partial\Omega^s$ to $\partial\Omega^s_I$, and n_I and n_O are the number of GLL points on $\partial\Omega^s_I$ and $\partial\Omega^s$, respectively. The corrected velocity field is thus,

$$\underline{\mathbf{u}} = \underline{\hat{\mathbf{u}}} + R^T\underline{\tilde{\mathbf{u}}}, \tag{4.3}$$

and the minimization problem (4.2) is

$$\min \qquad \underline{\tilde{\mathbf{u}}}^T R B_O R^T \underline{\tilde{\mathbf{u}}} \qquad \text{subject to} \quad (\underline{\hat{\mathbf{u}}} + R^T\underline{\tilde{\mathbf{u}}})^T B_O \underline{\hat{\mathbf{n}}} = 0, \tag{4.4}$$

where $\underline{\tilde{\mathbf{u}}}^T R B_O R^T \underline{\tilde{\mathbf{u}}}$ is the trace norm determined using the diagonal matrix (B_O) consisting of the product of surface quadrature weights with surface Jacobians (e.g., [84], pg.187).

Defining λ as a Lagrange multiplier for this constrained optimization problem, the Lagrangian function is

$$\mathcal{L}(\underline{\tilde{\mathbf{u}}},\lambda) = \underline{\tilde{\mathbf{u}}}^T R B_O R^T \underline{\tilde{\mathbf{u}}} + \lambda \upsilon + \lambda \underline{\tilde{\mathbf{u}}}^T R B_O \underline{\hat{\mathbf{n}}} = 0, \tag{4.5}$$

where $\upsilon = \underline{\hat{\mathbf{u}}}^T B_O \underline{\hat{\mathbf{n}}}$ is the net mass flux through the domain boundaries of Ω^s determined using the interpolated velocity field on $\partial\Omega^s_I$ and imposed Dirichlet boundary condition on $\partial\Omega^s_D$. The gradient of the Lagrangian is

$$\nabla\mathcal{L} = \begin{bmatrix} \nabla_{\underline{\tilde{\mathbf{u}}}} L \\ \nabla_\lambda L \end{bmatrix} = \begin{bmatrix} 2RB_OR^T\underline{\tilde{\mathbf{u}}} + \lambda RB_O\underline{\hat{\mathbf{n}}} \\ \upsilon + \underline{\tilde{\mathbf{u}}}^T RB_O\underline{\hat{\mathbf{n}}} \end{bmatrix}, \tag{4.6}$$

and solving this minimization problem by setting the gradient of the Lagrangian to 0, we get

$$\begin{bmatrix} 2RB_OR^T & RB_O\hat{\mathbf{n}} \\ (RB_O\hat{\mathbf{n}})^T & 0 \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{u}} \\ \lambda \end{bmatrix} = \begin{bmatrix} 0 \\ -\upsilon \end{bmatrix}. \tag{4.7}$$

Using Gaussian elimination, the system becomes

$$\begin{bmatrix} 2RB_OR^T & RB_O\underline{\hat{\mathbf{n}}} \\ 0 & -0.5\underline{\hat{\mathbf{n}}}R^TRB_O\underline{\hat{\mathbf{n}}} \end{bmatrix} \begin{bmatrix} \underline{\tilde{\mathbf{u}}} \\ \lambda \end{bmatrix} = \begin{bmatrix} 0 \\ -\upsilon \end{bmatrix}, \tag{4.8}$$

which gives

$$\lambda = \frac{2\upsilon}{\underline{\hat{\mathbf{n}}}R^TRB_O\underline{\hat{\mathbf{n}}}}, \tag{4.9}$$

and

$$\begin{aligned} 2RB_OR^T\underline{\tilde{\mathbf{u}}} &= -RB_O\underline{\hat{\mathbf{n}}}\lambda, \\ \implies 2RB_OR^T\underline{\tilde{\mathbf{u}}} &= -RB_O\underline{\hat{\mathbf{n}}}\frac{2\upsilon}{\underline{\hat{\mathbf{n}}}R^TRB_O\underline{\hat{\mathbf{n}}}}, \\ \implies RR^T\underline{\tilde{\mathbf{u}}} &= -R\underline{\hat{\mathbf{n}}}\frac{\int_{\partial\Omega^s}\hat{\mathbf{u}}\cdot\hat{\mathbf{n}}\,dA}{\int_{\partial\Omega^s_I}\hat{\mathbf{n}}\cdot\hat{\mathbf{n}}\,dA}, \\ \implies \underline{\tilde{\mathbf{u}}}|_{\partial\Omega^s_I} &= -\frac{\int_{\partial\Omega^s}\hat{\mathbf{u}}\cdot\hat{\mathbf{n}}\,dA}{\int_{\partial\Omega^s_I}\hat{\mathbf{n}}\cdot\hat{\mathbf{n}}\,dA}\underline{\hat{\mathbf{n}}}, \end{aligned} \tag{4.10}$$

The correction, $\underline{\tilde{\mathbf{u}}}$, derived here is important for ensuring boundary condition consistency when overlapping grids are used. In the Schwarz-SEM framework, the velocity correction (4.3) is imposed before each Schwarz iteration of the unsteady Stokes solve (3.6,3.7).

A drawback of the correction (4.3) is that the corrected velocity field ($\underline{\tilde{\mathbf{u}}}$) will not satisfy the continuity requirement $\underline{\tilde{\mathbf{u}}} \in \mathcal{H}^1$ if the domain boundary is not smooth or if $\partial\Omega^s_D \neq \partial\Omega^s$. Numerical experiments, however, show that (4.10) does not adversely impact the accuracy of the solution, mainly because the velocity field in the interior of the domain is still in $\mathcal{H}^1$, due to the SEM formulation that we discussed in Chapter 2. In future work, we will explore methods to improve our current approach such that $\underline{\tilde{\mathbf{u}}} \in \mathcal{H}^1$ and $\int_{\partial\Omega^s}\mathbf{u}\cdot\hat{\mathbf{n}}\,dA = 0$.

An important consequence of using (4.10) is that it overcomes the inability of the original Schwarz-SEM formulation in ensuring mass flux balance through the boundaries of each overlapping subdomain. This shortcoming becomes clear if we consider the example in Fig. 4.1, with an initial condition $\mathbf{u} = 0$ and a prescribed nonzero inflow boundary condition on Ω^1. In this case, the lagged interface velocity ($m = 1$ in (3.6)) would imply that $\mathbf{u}|_{\partial\Omega^1_I} = 0$ during the first time-step (or first substep in the iterated case) and consequently, $\int_{\partial\Omega^1}\mathbf{u}\cdot\hat{\mathbf{n}}\,dA \neq 0$. With the mass flux based correction, the velocity on $\partial\Omega^1_I$ will be nonzero *prior* to the Stokes substep on Ω^1, and the overall solution process will be consistent since $\int_{\partial\Omega^1}\mathbf{u}\cdot\hat{\mathbf{n}}\,dA = 0$.

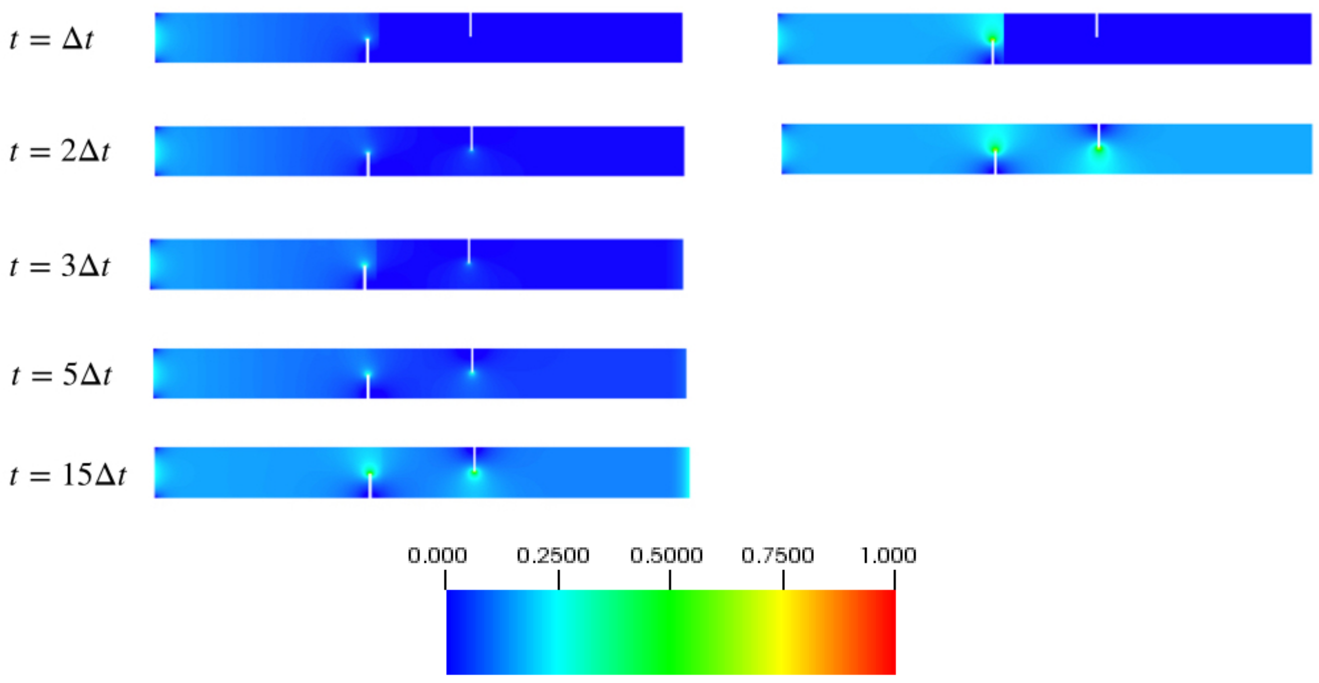


Figure 4.2: Velocity contours comparing flow in a channel with constrictions, (left) without and (right) with mass flux based correction.

If the subdomains form a chain of length S, and no subiterations are used, it will take S time-steps for the flow to exit the chain, but each subdomain solve will be self-consistent.

The geometry of Fig. 4.1 is an example of a chain of length 2. In Fig. 4.2(right), we show how the mass flux based correction leads to a rapid resolution of the global mass conservation, in contrast to the uncorrected case, which is shown on the left. Figure 4.3, which shows the mean velocity and mean divergence as a function of time in Ω^1 of Fig. 4.1, quantifies the impact of the mass flux based correction for this model problem. As we can see, without the velocity correction, the mean velocity in the streamwise direction varies even though the inflow velocity is constant. This result implies that without the mass flux based correction, mass is not conserved in Ω^1. We also see spikes in the divergence of the velocity field in the domain in Fig. 4.3(right). These spikes are associated with ejections of vortices, due to constrictions, passing through the interdomain boundaries. Using the mass flux based correction helps maintain boundary data consistency, and reduces divergence errors.

Turbulent flow calculations using the Schwarz-SEM framework have shown that using the correction (4.10) improves the stability of the PC scheme, and requires fewer corrector iterations when $m > 1$, as compared to calculations that are done without (4.10). We remark that (4.10) is analogous to flux corrections used in FV and FD approaches (e.g., [23, 53]).

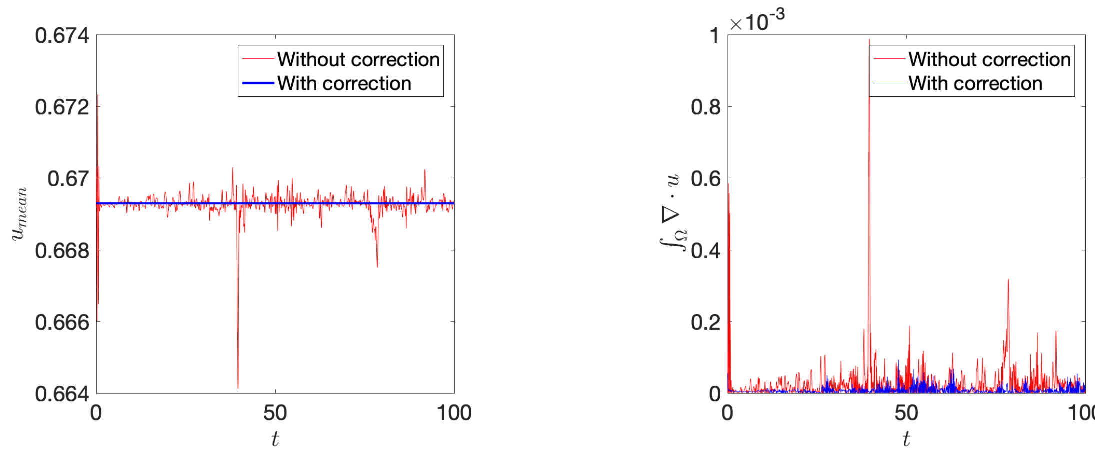


Figure 4.3: Comparison of the (left) mean velocity, and (right) divergence of the velocity $\int_\Omega \nabla \cdot \mathbf{u}$ for the channel with constrictions, with and without mass flux based correction.

4.2 Fixed Flow Rate through Overlapping Subdomains

Engineering applications featuring internal flows are often modeled in periodic domains with flow driven by a fixed pressure-gradient or a fixed flow rate. The goal of this section is to describe our methodology for maintaining a fixed flow rate ($\int_\Omega \mathbf{u}^n dV = \tilde{Q}$) in a periodic monodomain and overlapping grid-based simulations. We first describe our methodology for maintaining a fixed flow rate in the monodomain framework (which was developed prior to this dissertation) and then extend it to the Schwarz-SEM framework. This capability is essential for modeling problems such as heat transfer augmentation in pipes due to wire-coil inserts, discussed in Section 9.3.

Our strategy for time-advancing the solution of the Incompressible Navier-Stokes equations on monodomain grids is discussed in Section 2.2.3. If there is no forcing (or incorrect forcing) specified in (2.26), the INSE solution will not satisfy the desired flow rate $\tilde{Q}$ i.e., $\int_\Omega \mathbf{u}^n dV \neq \tilde{Q}$. Thus, we need to apply a forcing at every time-step in order to account for the energy dissipation due to the viscous effects and maintain the required flow rate.

Recalling our notation for monodomain SEM framework from Chapter 2, we use $\boldsymbol{\phi}^n = [\mathbf{u}^n, p^n]^T$ to describe the velocity solution ($\mathbf{u}^n$) and the pressure solution (p^n) at time t^n in monodomain SEM framework. The unsteady Stokes problem that we need to solve at each time-step is

$$\mathbf{S}\boldsymbol{\phi}^n = \mathbf{r}^n + \mathbf{r}_g^n, \quad \mathbf{u}^n|_{\partial\Omega_D} = \mathbf{u}_b^n, \quad p^n|_{\partial\Omega_N} = 0, \tag{4.11}$$

which is similar to (2.47), with $\mathbf{r}^n$ retaining its definition from (2.48), and we have introduced $\mathbf{r}_g^n$ (to be explained) to account for the inhomogeneities due to the forcing applied at each time-step for maintaining the desired flow rate.

Since (4.11) is a linear system that has to be solved at each time-step, we use the principle of superposition to split its solution into two components,

$$\begin{aligned}
\boldsymbol{\phi}^n &= \hat{\boldsymbol{\phi}}^n + \upsilon \boldsymbol{\phi}_g^n, \text{ s.t. } \int_\Omega \mathbf{u}^n dV := \tilde{Q}, && (4.12)\\
S\hat{\boldsymbol{\phi}}^n &= \hat{\mathbf{r}}^n, \quad \hat{\mathbf{u}}^n|_{\partial\Omega_D} = \mathbf{u}_b^n, \quad \hat{p}^n|_{\partial\Omega_N} = 0, && (4.13)\\
S\boldsymbol{\phi}_g^n &= \mathbf{r}_g^n, \quad \mathbf{u}_g|_{\partial\Omega_D} = 0, \quad p_g^n|_{\partial\Omega_N} = 0. && (4.14)
\end{aligned}$$

where υ is a scalar to be defined shortly, and $\hat{\boldsymbol{\phi}}^n = [\hat{p}^n, \hat{\mathbf{u}}^n]^T$ in (4.13) is the solution to the unsteady Stokes solve corresponding to (2.47), which accounts for the BDFk and EXTk terms without any forcing function. Equation (4.13) also accounts for the specified Dirichlet boundary conditions on $\partial\Omega_D$. $\hat{\mathbf{r}}^n$ in (4.13), thus, has the same definition as that in (2.48).

$\boldsymbol{\phi}_g^n = [\mathbf{u}_g^n, p_g^n]^T$ in (4.14) is the solution to the unsteady Stokes solve corresponding to a unit-forcing vector ($\mathbf{f}_g$) in the streamwise direction, with homogeneous Dirichlet boundary conditions on $\partial\Omega_D$ and homogeneous initial conditions. Splitting the solution of (4.11) using the approach described here has the advantage that the final solution can be determined using (4.12), where υ is chosen at each time-step such that $\boldsymbol{\phi}^n$ satisfies the desired flow rate,

$$\begin{aligned}
\upsilon &= \frac{\int_\Omega \mathbf{u}^n dV - \int_\Omega \hat{\mathbf{u}}^n dV}{\int_\Omega \mathbf{u}_g^n dV}, && (4.15)\\
\implies \upsilon &= \frac{\tilde{Q} - \int_\Omega \hat{\mathbf{u}}^n dV}{\int_\Omega \mathbf{u}_g^n dV}, && (4.16)
\end{aligned}$$

Since we assume that the density (ρ) and viscosity (μ) are constant in the domain, it is clear from the following discussion that (4.14) does not depend on time and can be solved as a pre-processing step for a given geometry or time-step size. The correction (4.12) can then be applied at each time-step using the pre-computed solution of $\boldsymbol{\phi}_g^n$. In the sequel, we will drop the superscript n from $\boldsymbol{\phi}_g^n$ because the latter is time-independent for the applications that we consider in this dissertation.

Using the approach (and following the notation) in Section 2.2.2, we obtain the pressure Poisson equation and Helmholtz equation for velocity for $\boldsymbol{\phi}_g$:

$$\begin{aligned}
-\nabla^2 p_g &= -\nabla \cdot \mathbf{f}_g, && (4.17)\\
\frac{\beta_0}{\Delta t}\mathbf{u}_g - \frac{1}{Re}\nabla^2 \mathbf{u}_g &= -\nabla p_g + \mathbf{f}_g, && (4.18)
\end{aligned}$$

where (4.17) and (4.18) are similar to (2.29) and (2.32) respectively, from Section 2.2.2, and account for the unit-forcing vector $\mathbf{f}_g$ in the direction of the flow. Similar to (2.34), casting (4.17) in variational form leads to

$$a(\psi, p_g) = -(\mathbf{f}_g, \nabla\psi) + \int_{\partial\Omega_D} \psi \nabla p_g \cdot \hat{\mathbf{n}}\, dA + \int_{\partial\Omega_D} \psi \mathbf{f}_g \cdot \hat{\mathbf{n}}\, dA, \tag{4.19}$$

where the surface integrals on $\partial\Omega_D$ simplify to

$$\left.\frac{\partial p_g}{\partial n}\right|_{\partial\Omega_D} + \left.\mathbf{f}_g \cdot \hat{\mathbf{n}}\right|_{\partial\Omega_D} = -\frac{\beta_0}{\Delta t}\mathbf{u}_g \cdot \hat{\mathbf{n}}, \tag{4.20}$$

and thus vanish because we assume homogeneous Dirichlet conditions, i.e., $\mathbf{u}_g|_{\partial\Omega_D} = 0$. Similarly, casting (4.18) in variational form leads to

$$\frac{\beta_0}{\Delta t}(\mathbf{v}, \mathbf{u}_g) + \frac{1}{Re} a(\mathbf{v}, \mathbf{u}_g) = (\mathbf{v}, \mathbf{f}_{vg}), \tag{4.21}$$

where $\mathbf{f}_{vg}$ represents terms on the right of (4.18).

Using (4.17) and (4.18), $\mathbf{r}_g$ in (4.14) is defined as

$$\begin{aligned} \mathbf{r}_g &= [\mathbf{r}_{vg}, r_{pg}]^T, \\ \mathbf{r}_{vg} &= -\nabla p_g + \mathbf{f}_g, \\ r_{pg} &= -\nabla \cdot \mathbf{f}_g, \end{aligned} \tag{4.22}$$

and the solution for $\boldsymbol{\phi}_g = [\mathbf{u}_g, p_g]^T$ is obtained using (4.14) and (4.22). Once $\boldsymbol{\phi}_g$ is determined for a given geometry and time-step size (Δt), $\boldsymbol{\phi}^n$ is updated at each time-step with (4.12) and (4.16) to maintain the desired flow rate. This method for maintaining fixed flow rate has been used for highly turbulent flow simulations in the monodomain SEM framework [100].

The extension of this method for maintaining fixed flow rate through overlapping subdomains is straightforward if we solve (4.13) and (4.14) using simultaneous Schwarz iterations (1.2). We have already discussed our methodology for solving (4.13) in Section 3.1 using the PC scheme (3.6,3.7), which corresponds to the unsteady Stokes solve (minus any forcing). Our goal here is to describe how we can solve (4.14) in the Schwarz-SEM framework.

Based on the discussion of the OS-based methods, we can anticipate that solving (4.14) will require us to use simultaneous Schwarz iteration (1.2). Consequently, in addition to the homogeneous boundary conditions on subdomain boundaries $\partial\Omega_D^s$, we will have inhomogeneous boundary conditions on interdomain boundaries $\partial\Omega_I^s$, that will be obtained from the corresponding overlapping grid. Using $\boldsymbol{\phi}_g^{s,[q]}$ to denote the solution $\boldsymbol{\phi}_g$ in Ω^s at the qth Schwarz iteration, our methodology for solving (4.14) with Q_g simultaneous Schwarz iterations is

$$\begin{aligned} q_g = 0 :\ \mathbf{S}\boldsymbol{\phi}_g^{s,[0]} &= \mathbf{r}_g^{s,[0]}, \quad \mathbf{u}_g|_{\partial\Omega_D^s} = 0, \quad \mathbf{u}_g|_{\partial\Omega_I^s} = 0,, \quad p_g^n|_{\partial\Omega_N} = 0, \qquad (4.23)\\ 1 \le q_g \le Q_g :\ \mathbf{S}\boldsymbol{\phi}_g^{s,[q_g]} &= \mathbf{r}_g^{s,[q_g]}, \quad \mathbf{u}_g|_{\partial\Omega_D^s} = 0, \quad \mathbf{u}_g|_{\partial\Omega_I^s} = \mathcal{I}(\mathbf{u}_g^{r,[q_g-1]}), \quad p_g^n|_{\partial\Omega_N} = 0, \qquad (4.24)\end{aligned}$$

where $\mathbf{r}_g^s$ accounts for all inhomogeneities for both pressure and velocity, similar to (4.22). We note that we start with homogeneous boundary conditions on the interdomain boundaries (4.23), and then update the boundary data for $\partial\Omega_I^s$ using interpolation from the corresponding overlapping grid, as indicated in (4.24).

Thus, recalling our notation that $\boldsymbol{\phi}^{s,n,[Q]}$ is the solution $\boldsymbol{\phi}$ in Ω^s at the Qth Schwarz iteration at time t^n, the solution to the incompressible Navier-Stokes equations can be updated at the end of each time-step as

$$\boldsymbol{\phi}^{s,n,[Q]} = \hat{\boldsymbol{\phi}}^{s,n,[Q]} + \upsilon\boldsymbol{\phi}_g^{s,[Q_g]}, \qquad (4.25)$$

where $\hat{\boldsymbol{\phi}}^{s,n,[Q]}$ is obtained at each time-step using the approach outlined in Section 3.1, and υ is chosen such that $\mathbf{u}^n$ satisfies the desired flow rate $\tilde{Q}$.

Numerical experiments show that typically 10-20 Schwarz iterations are sufficient for obtaining an accurate solution of $\boldsymbol{\phi}_g^{s,[Q_g]}$. Using this method, we have used the Schwarz-SEM framework to simulate internal flows in complex domains, which were intractable with monodomain SEM.

We note that (4.16) requires the computation of integrals over the entire domain Ω, rather than over individual subdomains Ω^s. Global integration is not trivial on overlapping subdomains because grid points in the overlap region must be weighted appropriately. In Section 4.3, we will describe our methodology of constructing partition-of-unity functions for enabling global integration in subdomains with arbitrary overlap.

While the approach of fixed flow rate described in this section is effective for simulating internal flow in periodic domains using overlapping grids, there is a caveat. In the current Schwarz-SEM framework, we only support intradomain periodicity, i.e., surfaces that are periodic to each other must be part of the same subdomain. Consequently, interdomain periodicity is not supported. Figure 4.4 shows an example of a periodic domain, which is used to study turbulent flow at a fixed flow rate, and Fig. 4.4(b) shows a valid configuration for modeling this domain with periodic boundary conditions on overlapping grids. Here, each subdomain has a surface that is periodic to another surface in the same subdomain. In contrast, Fig. 4.4(c) shows a configuration that requires interdomain periodicity, which we do not support in the current

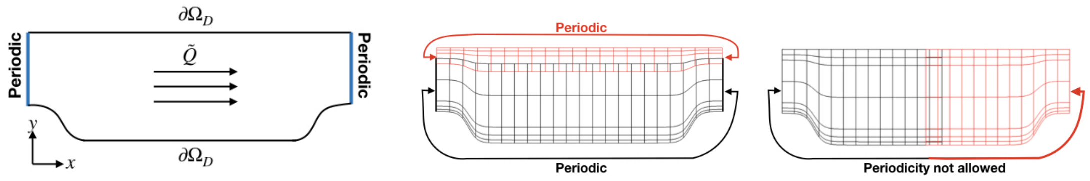


Figure 4.4: (left to right) (a) Periodic domain with flow at a fixed rate $\tilde{Q}$ in the streamwise direction (x). (b) A valid domain decomposition strategy for using two overlapping grids, each with periodic boundary conditions, to model the domain, and (c) an invalid configuration with interdomain periodicity i.e., surfaces that are periodic to each other are on different subdomains. Interdomain periodicity is not currently supported in the Schwarz-SEM framework.

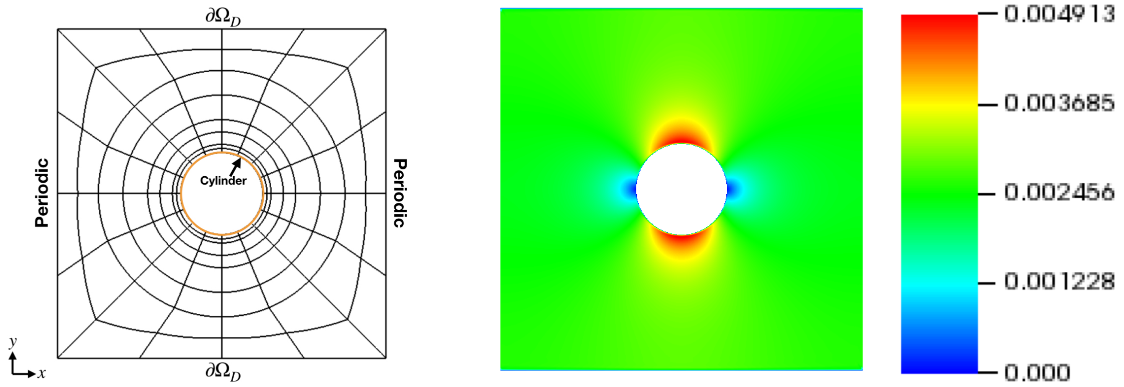


Figure 4.5: (left) Spectral element mesh for flow past a cylinder, and (right) velocity magnitude plot $||\underline{\mathbf{u}}_g||$.

Schwarz-SEM framework.

4.2.1 Validation

We validate the fixed-flow rate formulation of the preceding section by comparing the results for monodomain and multidomain simulations of flow past a cylinder in a periodic box. Figure 4.5 shows the spectral element mesh generated for the single domain calculation. The square domain $\Omega := [-2.0, 2.0]^2$ with periodic boundary conditions in the streamwise direction (x) and homogeneous Dirichlet in y. The nondimensional diameter of the cylinder is $D_{nd} = 1$. For this numerical experiment, we set the nondimensional flow rate $\tilde{Q} = U_{nd}L_{y,nd}$ to 40, where $L_{y,nd} = 4$ is the nondimensional length of the domain in y direction and $U_{nd} = 10$ is the nondimensional mean flow speed in the streamwise direction. The Reynolds number of the flow is $Re = U_{nd}D_{nd}/\nu_{nd} = 1000$.

We split this domain Ω into two overlapping grids by modifying the monodomain mesh. The radius of the inner mesh is 1.5 for $\partial\Omega_I^1$ and 1 for $\partial\Omega_I^2$ for the outer mesh. As a result, the overlap width is 0.5. Figure 4.6 shows the two spectral element meshes generated for Schwarz-SEM calculation. After $Q_g = 10$ simultaneous Schwarz iterations, the maximum difference in the solution for $\boldsymbol{\phi}_g$ between the monodomain and overlapping grids is 10^{-4}. Figure 4.6(right) shows the plot of magnitude of the velocity solution $\mathbf{u}_g$.

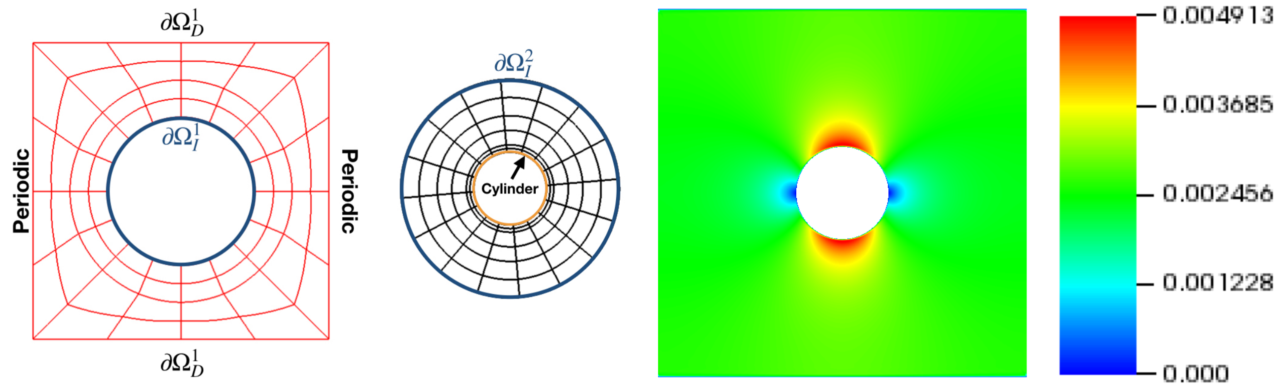


Figure 4.6: (left) Overlapping spectral element meshes for flow past a cylinder, and (right) velocity magnitude plot $||\underline{\mathbf{u}}_g||$ determined using the simultaneous Schwarz iterations.

4.3 Integration on Subdomains with Arbitrary Overlap

The computation of global quantities (such as mean flow velocity, flow rate, Nusselt number, etc.) on overlapping grids is not as straightforward as on monodomain grids. In monodomain SEM, integration is effected via the mass matrix (2.15). This mass matrix cannot be directly used for global integration on overlapping grids because we need to ensure that grid points in the overlap region are weighted correctly. In this section, we first describe why some of the methods that are used in the existing literature do not apply to our Schwarz-SEM framework. We then describe our methodology for enabling integration in subdomains with arbitrary overlap.

In implementations based on Nitsche's method, global integration is enabled by partitioning overlapping grids into nonoverlapping subsets. The arbitrary shaped elements at the interface of different subsets are further partitioned into regular shaped tetrahedrons/triangles. Using this approach of splitting the domain into nonoverlapping subsets, numerical integration can be effected via the mass matrix [11]. Another approach for integration on overlapping meshes is to remesh the overlap region with regular-shaped elements. Numerical quadrature can then be used to integrate the grid points in nonoverlapping and overlapping region, as usual [56]. Each of the above approaches relies on remeshing, which is guaranteed only for tetrahedrons or triangular elements. Additionally, none of these approaches is straightforward to implement in parallel since each relies on identifying intersections of different overlapping grids. Thus, we need a different approach for global integration that weights grid points in the overlap region correctly. The challenge of such an approach is that we need a method that supports subdomains with arbitrary overlap, where we have no a priori knowledge of the shape of different subdomains.

Our method for global integration is to construct a partition-of-unity function (weight) in each subdomain.

In order to define the properties of this function, we introduce the following notation,

$$\tilde{\Omega}^r = \Omega \backslash \Big(\bigcup_{s \neq r} \Omega^s \Big), \tag{4.26}$$

where $\tilde{\Omega}^r$ represents the part of Ω^r that is not overlapped by any other subdomain. Thus, for a GLL point $\tilde{\mathbf{x}}$ in Ω, we want the partition-of-unity function (χ^s) to satisfy the following properties

$$\begin{aligned} \sum_{s=1}^{S} \chi^s(\tilde{\mathbf{x}}) = 1 \qquad & \forall \tilde{\mathbf{x}} \in \Omega, \\ \chi^s(\tilde{\mathbf{x}}) = 1 \qquad & \forall \tilde{\mathbf{x}} \in \tilde{\Omega}^s, \\ \chi^s(\tilde{\mathbf{x}}) = 0 \qquad & \forall \tilde{\mathbf{x}} \in \partial\Omega_I^s. \end{aligned} \tag{4.27}$$

These properties of partition-of-unity function are based on the principle that the error due to Schwarz iterations is highest at the interdomain boundaries, and decreases away from $\partial\Omega_I^s$. In order to describe our approach for constructing this partition-of-unity function, we revisit the distance field (based on the distance from interdomain boundaries) that we described in Section 3.2.2.

In Chapter 3, we described our approach of generating a distance function. The distance function based on the interdomain boundaries, δ^s, is used as a discriminator in *findpts* for donor element search (Section 3.2.2). Since this distance function is 0 for grid points on $\partial\Omega_I^s$ and increases away from $\partial\Omega_I^s$, it seems natural to use it for determining χ in each subdomain. Our methodology for constructing the partition-of-unity function for a GLL point $\tilde{\mathbf{x}}$ in Ω^s, using the distance function δ^s, is

$$\chi^s(\tilde{\mathbf{x}}) \quad = \frac{\delta^s(\tilde{\mathbf{x}})}{\sum_{r=1}^{S} \delta^r(\tilde{\mathbf{x}})}, \tag{4.28}$$

where $\delta^r(\tilde{\mathbf{x}})$ is determined by first using *findpts* to find $\tilde{\mathbf{x}}$ in Ω^r. Next, if *findpts* returns that the point is not found in Ω^r, $\delta^r(\tilde{\mathbf{x}})$ is set to 0. Otherwise, the computational coordinates returned by *findpts* are used to interpolate the distance field δ using *findpts_eval*. (4.28) is used to determine the partition-of-unity weights for all the GLL points of each subdomain, and it ensures that all the requirements in (4.27) are satisfied.

A limitation of the current approach (4.28) is that the partition-of-unity functions depend on the distance function (based on the interdomain boundaries), which in turn depends on the mesh geodesics in each subdomain. The accuracy (and convergence properties) of these partition-of-unity functions is, thus, constrained by the accuracy of our nearest-neighbor based distance function generation approach (Section 3.4). Additionally, distance functions are typically not smooth (i.e., $\delta^s \notin \mathcal{H}^1$), and determination of χ^s requires interpolation of this distance function in each subdomain. As a result, we do not expect to get exponential convergence with polynomial order N for global integration using (4.28), and we will look at more accurate methods of distance function generation in future work. In the following section, however, we will discuss numerical experiments that show that for practical purposes, our current approach of global integration gives sufficiently accurate results (relative error less than 10^{-4}) even for subdomains with complex overlap.

4.3.1 Validation

To validate our approach for partition-of-unity function generation, we use the example of three overlapping grids for discretizing the periodic domain $\Omega = [0, 2\pi]^2$ from Chapter 3, shown here in Fig. 4.7. Since the overlap between the different subdomains is not trivial, constructing the partition-of-unity function manually for each subdomain is not feasible/desirable. (4.28) allows us to effortlessly determine the partition-of-unity function for each subdomain, which is shown in Fig. 4.8. As we can see, the partition-of-unity function is unity in the nonoverlapping region and goes to 0 on the interdomain boundary. In order to ensure that these weights indeed add to unity everywhere, we use the partition-of-unity weighted mass matrix to determine the total area ($4\pi^2$) of the domain. Our results indicate that the relative error for this approach is less than 10^{-5}, when $N = 8$.

Figure 4.9 shows how the relative error in the area varies with the polynomial order N. As expected, we do not get exponential convergence in global integration with polynomial order N due to dependence of χ^s on the nearest-neighbor based distance function. We do note, however, that the relative error in global integration is less than 10^{-4} for the multidomain cases considered here, even when $N = 3$, which is sufficient for our target applications.

Our approach for global integration is being used for the overlapping grid calculations ($S = 3$) at the

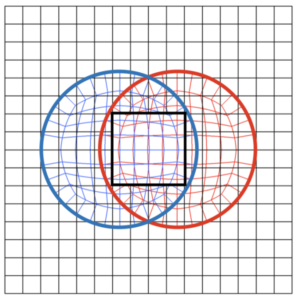


Figure 4.7: 3 overlapping spectral element meshes for discretizing the periodic domain $\Omega = [0, 2\pi]^2$. Two circular meshes (96 elements each) are used to cover a hole in the background mesh (240 elements.

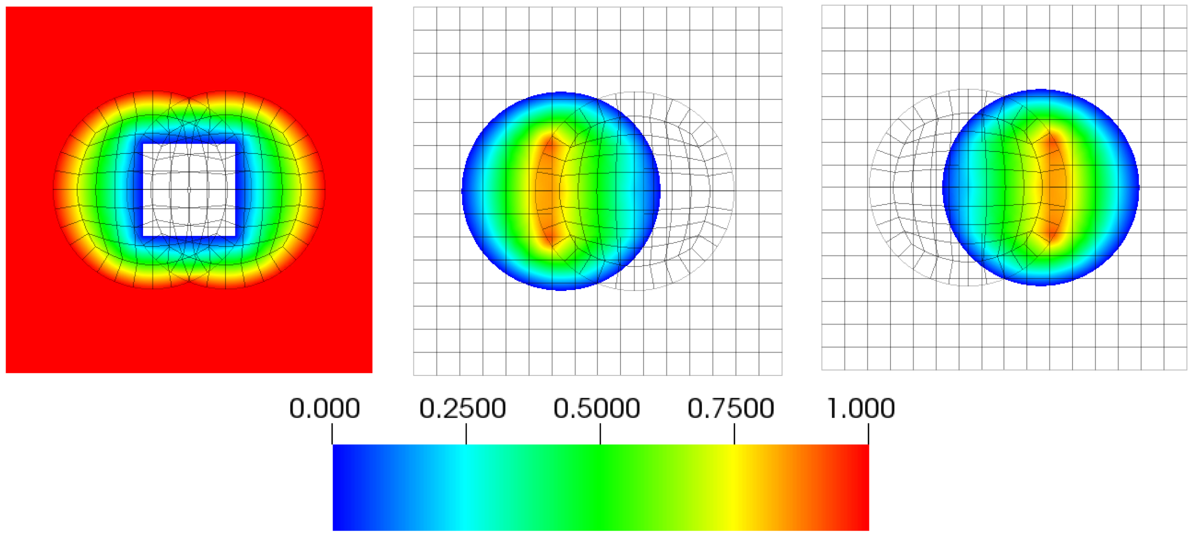


Figure 4.8: Partition-of-unity weights ($N = 8$) for each subdomain along with the grids that overlap it. The partition-of-unity weight is unity for GLL points in the nonoverlapping regions and goes to 0 on the interdomain boundary.

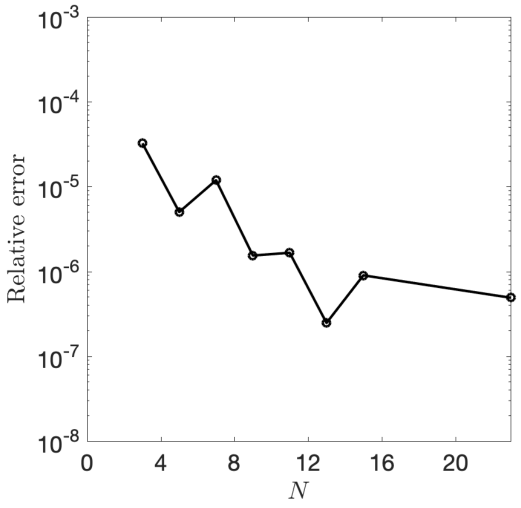


Figure 4.9: Spatial convergence in relative error with polynomial order N, for global integration in the example with $S = 3$ overlapping subdomains.

Argonne National Lab to simulate flow through the intake valve of an internal combustion engine. Figure 4.10 shows the overlapping subdomains used to model the intake valve. Numerical experiments show that (4.28) can be used for global integration to determine the volume of the domain with a relative error less than 10^{-4}. Figure 4.11 shows the partition-of-unity weights determined for each overlapping subdomain.

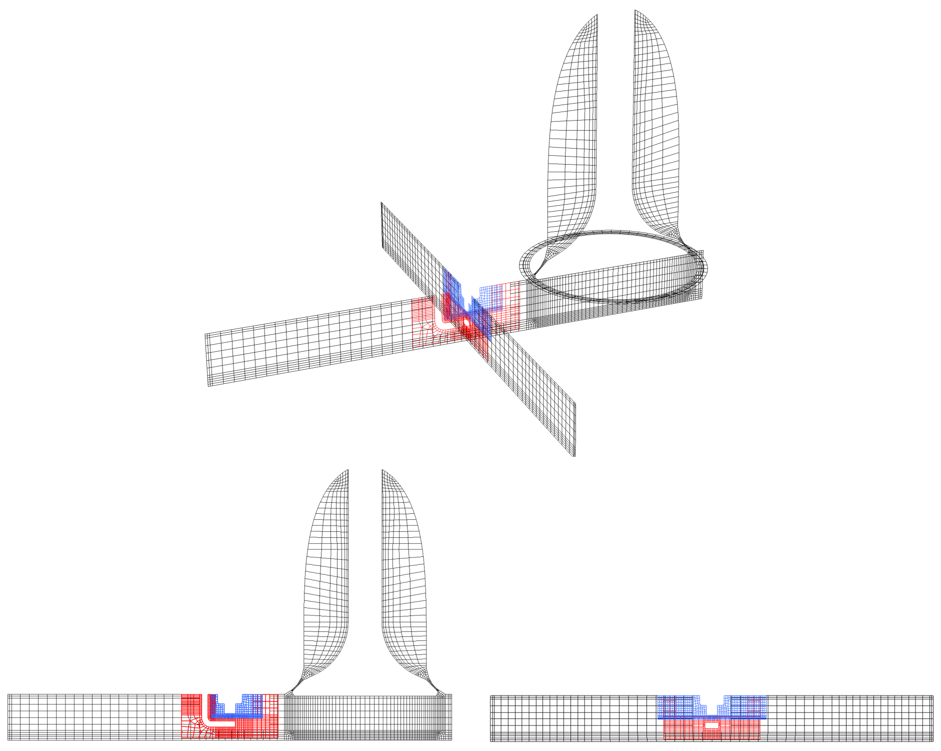


Figure 4.10: (top) Three-slice view of the overlapping grids for modeling the intake valve of an internal combustion engine, and (bottom) slice view showing the cross-section of the overlapping grids.

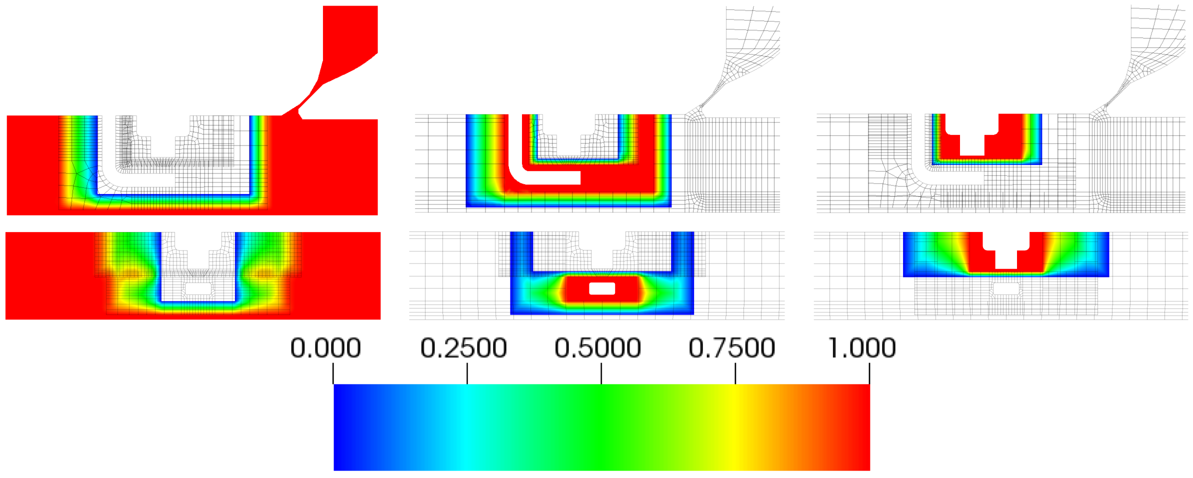


Figure 4.11: Slice views of the cross-section along two different directions, showing distribution of the partition-of-unity function ($N = 8$) near the overlap region for the three subdomains discretizing the intake valve. The error due to global integration using these weights is less than 0.01%.

4.4 Summary

In this chapter, we presented our methodology for ensuring that interpolation of boundary condition data on interdomain boundaries does not violate the divergence-free constraint. This approach ensures boundary data consistency for the solution of the INSE. Numerical experiments indicate that the mass flux based correction significantly reduces the overall divergence in the domain, and also reduces the number of Schwarz iterations required for using high-order extrapolation of interdomain boundary data in certain cases. We have also described our method for maintaining fixed flow rate in overlapping subdomains, which is important for modeling internal flows, and are already using it in a production level calculation for understanding how heat transfer is augmented in a pipe due to a wire-coil insert (Section 9.3). In this chapter, we also addressed the issue of global integration on overlapping subdomains, which is much more complicated in comparison to integration on monodomain grids. Global integration is effected in the Schwarz-SEM framework using partition-of-unity functions that ensure that grid points in the overlap region are weighted appropriately.

Chapter 5

Stability of the Predictor-Corrector Scheme in the Schwarz-SEM Framework

In Chapter 3, we have described our methodology for time-advancing the solution of INSE in overlapping grids. An important concern for this predictor-corrector based time advancement strategy (3.6,3.7) is the number of corrector iterations (Q) that are required at each time-step. Due to the work presented in [24] and [55], the implications for the choice of interface extrapolation order (m) and number of corrector iterations (Q) in the Schwarz-SEM framework are

- $m = 1$, $Q = 0$ - Unconditionally stable, $\mathcal{O}(\Delta t)$ accurate.
- $m = 1$, $Q > 0$ - Unconditionally stable, more than $\mathcal{O}(\Delta t)$ accurate depending upon Q.
- $m > 1$, $Q = 0$ - Unstable.
- $m > 1$, $Q > 0$ - Conditionally stable[1], $\mathcal{O}(\Delta t^m)$ accurate.

We have additionally noticed in the Schwarz-SEM framework that for some cases where $m > 1$, the calculation is stable only for *odd* values of Q but not for *even* values. This result is counter-intuitive since one would anticipate that increasing Q would lead to increased stability and accuracy. For example, for the Navier-Stokes eigenfunctions test case ($S = 2$) presented in Fig. 3.8, if we set $\Delta t = 2 \times 10^{-3}$ and $N = 7$, we observe that using $Q = 2$ or 4 leads to instabilities for $m = 3$, whereas using $Q = 1$ or 3 leads to a stable and accurate solution. Figure 5.1 shows the error in solution versus time for different Q, for this example. As we can see, the solution is stable for $Q = 1$ and 3 but unstable for $Q = 2$ and 4.

5.1 Methodology

To investigate the difference in stability of the PC scheme for odd- and even-Q, we use a matrix stability framework, similar to [55], applied to the unsteady heat equation. We start with a 1D grid and develop an FD-based scheme for time-advancing the solution of the unsteady heat equation. We then extend this framework to solve the unsteady heat equation in two overlapping grids and cast the time-advancement of the solution into a system of the form $\underline{z}^n = G\underline{z}^{n-1}$. Matrix stability analysis indicates that a scheme

[1]Based on stability analysis and numerical experiments, $Q = 3$ is typically sufficient.

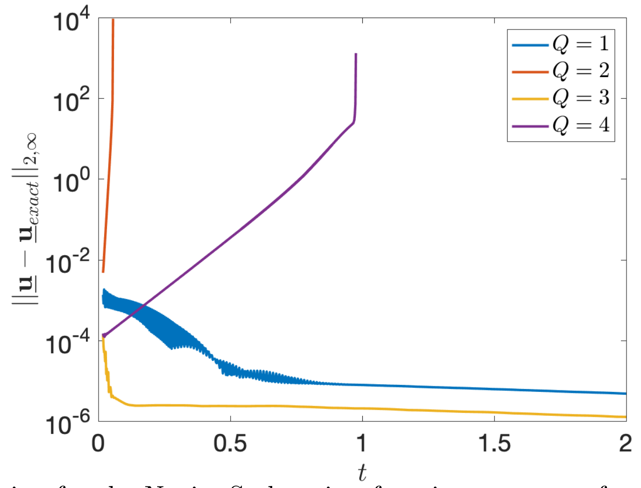


Figure 5.1: Error variation for the Navier-Stokes eigenfunctions test case from Chapter 3 with different Q using $\Delta t = 2 \times 10^{-3}$ and $N = 7$.

of the form $\underline{z}^n = G\underline{z}^{n-1}$ is asymptotically stable if the spectral radius of G, $\rho(G)$, is strictly less than 1 (e.g., [101])[2]. This framework will allow us to analyze the stability properties of the PC scheme for different m and Q, as we will see in the next section.

We note that use of the unsteady heat diffusion problem with an FD-based scheme, instead of the INSE in higher space-dimensions with an SEM-based formulation, significantly simplifies the stability analysis. This approach allows us to focus on the qualitative impact of key parameters such as Q, m, grid resolution, and overlap width on the stability of the high-order PC scheme. As we will see in this chapter, this simplified stability analysis has helped us in qualitatively capturing stability behavior that we have observed in the Schwarz-SEM framework for solving the INSE (Figure 5.1). Additionally, this analysis has also helped us in determining a potential generalized stability behavior of high-order PC schemes for solving ODEs and PDEs (similar to the 1D unsteady heat equation in this chapter or the INSE in the Schwarz-SEM framework).

5.1.1 Unsteady diffusion with a monodomain grid

For the monodomain case, consider the solution $u(x,t)$ for the unsteady diffusion equation

$$u_t = \nu u_{xx}, \quad x \in [0,1], \quad \nu > 0, \tag{5.1}$$

where we model the domain with $\tilde{M} + 2$ uniformly-spaced grid points (Fig. 5.2), such that $\Delta x = \frac{1}{\tilde{M}+1}$. A homogeneous boundary condition is imposed at the left boundary, $u(0,t) = 0$, and a time-dependent

[2]Use of ρ to denote the spectral radius should not confused with the use of ρ for density in the incompressible Navier-Stokes equations.

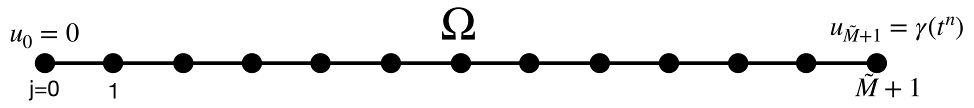


Figure 5.2: Monodomain grid ($\tilde{M} = 11$)

inhomogeneous boundary condition is prescribed at the right boundary, $u(1, t) = \gamma(t^n)$.

For notational purposes, we introduce u_j^n to represent the solution u at jth grid point at time t^n, R as the standard restriction matrix (5.2) that casts the solution from the $\tilde{M}+1$ grid points $\underline{\bar{u}}^n = [u_1^n, u_2^n, \dots u_{\tilde{M}+1}^n]^T$ to the $\tilde{M}$ degrees of freedom $\underline{u}^n = [u_1^n, u_2^n, \dots u_{\tilde{M}}^n]^T$, and $\underline{\bar{u}}_b^n = [0, 0, \dots, 0, \gamma(t^n)]^T$ as a vector of length $\tilde{M}+1$ with zeros and the inhomogeneous boundary condition $\gamma(t^n)$. We have omitted u_0^n from our vectors ($\underline{\bar{u}}^n, \underline{u}^n$, and $\underline{\bar{u}}_b$) due to the homogeneous boundary condition at $x = 0$. The $\tilde{M} \times \tilde{M}+1$ restriction operator R is an identity matrix, with a column of zeros appended to it, and is defined as

$$R = \begin{bmatrix} 1 & 0 & \ddots & \ddots & \ddots & 0 & 0 \\ 0 & 1 & \ddots & \ddots & \ddots & 0 & 0 \\ \ddots & \ddots & \ddots & \ddots & \ddots & 0 & 0 \\ \ddots & \ddots & \ddots & \ddots & \ddots & 0 & 0 \\ \ddots & \ddots & \ddots & \ddots & \ddots & 0 & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & 1 & 0 \end{bmatrix}. \tag{5.2}$$

The standard 2nd-order accurate central finite difference operator for u_{xx} is of size $\tilde{M}+1 \times \tilde{M}+1$, and is defined as $\bar{A}_{ii} = 2/\Delta x^2$ and $\bar{A}_{i-1,i} = \bar{A}_{i+1,i} = -1/\Delta x^2$, i.e.,

$$\bar{A} = \frac{1}{\Delta x^2} \begin{bmatrix} 2 & -1 & \ddots & \ddots & \ddots & 0 \\ -1 & 2 & -1 & \ddots & \ddots & 0 \\ \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ \ddots & \ddots & \ddots & \ddots & \ddots & -1 \\ 0 & \ddots & \ddots & \ddots & -1 & 2 \end{bmatrix}. \tag{5.3}$$

It is straightforward to derive that using a BDFk scheme for discretizing u_t, and a 2nd-order accurate central finite difference approximation for u_{xx}, the solution for the unsteady diffusion equation can be time-

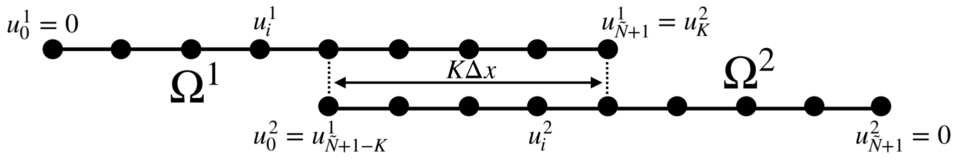


Figure 5.3: Overlapping grids ($\tilde{N} = 7, K = 4$)

advanced from t^{n-1} to t^n using the solution up to t^{n-1} as

$$\underline{u}^n = -\sum_{l=1}^{k} \beta_l H^{-1} \underline{u}^{n-l} - \nu \Delta t H^{-1} R \bar{A} \underline{\bar{u}}_b^n, \tag{5.4}$$

where Δt is the time-step size (assumed to be the same at all time-steps), β_l are coefficients for the BDFk scheme, $H = \beta_0 I + \nu \Delta t A$ is the Helmholtz matrix, I is a $\tilde{M} \times \tilde{M}$ identity matrix, and $A = R\bar{A}R^T$. The system in (5.4) also depends on the time-dependent inhomogeneous boundary condition, $\gamma(t^n)$.

We can now extend this approach to overlapping grids using a high-order PC scheme, similar to (3.7).

5.1.2 Unsteady diffusion with overlapping grids

For solving the unsteady diffusion equation with overlapping grids, we split the monodomain grid (Ω) into two grids (Ω^1 and Ω^2) with equal number of grid points ($\tilde{N} + 2$) and overlap width $K\Delta x$, such that the grid points in the overlap $\Omega^1 \cap \Omega^2$ coincide. Figure 5.3 shows the overlapping grids obtained from the monodomain grid of Fig. 5.2, and the overlapping grids are setup such that $\tilde{M} = 2\tilde{N} - K + 1$. Algebraically, this decomposition is realized through restriction matrices, R_i that extract $\tilde{N}$ of the $\tilde{M}$ values from a given vector $\underline{u}$ on Ω as $\underline{u}^i = R_i \underline{u}$. Here, we we introduce the notation $\underline{u}^i$ to represent the solution at the $\tilde{N}$ degrees of freedom of Ω^i, and $u_j^{i,n}$ to represent the solution at jth grid point of Ω^i at t^n.

$$R_1 = \overbrace{\begin{bmatrix} 1 & 0 & \ddots & \ddots & \ddots & 0 & 0 & \dots & 0 \\ 0 & 1 & \ddots & \ddots & \ddots & 0 & 0 & \dots & 0 \\ \ddots & \ddots & \ddots & \ddots & \ddots & 0 & 0 & \dots & 0 \\ \ddots & \ddots & \ddots & \ddots & \ddots & 0 & 0 & \dots & 0 \\ \ddots & \ddots & \ddots & \ddots & \ddots & 0 & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & 1 & 0 & \dots & 0 \end{bmatrix}}^{\tilde{M}}_{\underbrace{\qquad\qquad\qquad}_{\tilde{N}}\underbrace{\qquad}_{\tilde{M}-\tilde{N}}} \tag{5.5}$$

$$R_2 = \overbrace{\begin{bmatrix} 0 & \dots & 0 & 1 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & 0 & 1 & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & 0 & \ddots & \ddots & \ddots & \ddots & 1 \end{bmatrix}}^{\tilde{M}} \tag{5.6}$$
$$\underbrace{\hphantom{0 \dots 0}}_{\tilde{M}-\tilde{N}}\underbrace{\hphantom{0 \quad 1 \quad 0 \quad \ddots \quad \ddots \quad \ddots \quad 0}}_{\tilde{N}}$$

For simplicity, we impose homogeneous boundary conditions at the left boundary of Ω^1 ($u_0^{1,n} = 0$) and right boundary of Ω^2 ($u_{\tilde{N}+1}^{2,n} = 0$). These boundary conditions allow us to use the method developed for the monodomain grid (5.4), for overlapping grids, with the difference that the boundary data for the interdomain boundary grid points ($u^1_{\tilde{N}+1}$ and u^2_0) is obtained from the corresponding overlapping grid in each subdomain. To effect this interdomain exchange, we define an interpolation operator, $B_{ij} = (I - R_i^T R_i) R_j^T$, that extracts the value from Ω^j at $\partial\Omega_I^i \cap \Omega^j$ and maps it to Ω^i. The operator B_{ij} serves the same purpose as *findptslib* in the Schwarz-SEM framework for interpolating the interdomain boundary data (Section 3.2.2).

Similar to Section 3.1, since we are solving for $\underline{u}^{i,n}$ in both subdomains simultaneously, the information at interface boundaries must be extrapolated from the solution at previous time-steps. Using a kth-order accurate BDFk scheme for u_t, and an mth-order accurate EXTm scheme for extrapolating the interface boundary term, the solution in each subdomain is

$$\underline{u}^{i,n} = -\sum_{l=1}^{k} \beta_l H_i^{-1} \underline{u}^{i,n-l} + \sum_{l=1}^{m} \tilde{\alpha}_l H_i^{-1} J_{ij} \underline{u}^{j,n-l}, \tag{5.7}$$
$$H_i = \beta_0 I_i + \nu\Delta t A_i, \quad J_{ij} = -\nu\Delta t R_i A B_{ij}, \quad A_i = R_i A R_i^T,$$

where β_l and $\tilde{\alpha}_l$ are coefficients for the BDFk and the EXTm scheme, respectively, and I_i is the identity matrix. All the matrices in (5.7) are of size $\tilde{N} \times \tilde{N}$ except the restriction operator R_i (5.5,5.6).

To ensure stability for a high-order EXTm scheme, we extrapolate the interface data obtained from the overlapping subdomain at the first iteration, and then iterate using the latest solution at each corrector iteration. Using the notation $\underline{u}^{i,n,q}$ for the solution at the qth corrector iteration[3], the PC scheme for

[3]For the sake of convenience in representing the matrices that are to follow in this section, we have dropped the brackets around the Schwarz iteration index $[q]$ in this chapter.

time-advancing the solution in overlapping grids is

$$\begin{aligned} q = 0 : \underline{u}^{i,n,0} &= -\sum_{l=1}^{k} \beta_l H_i^{-1} \underline{u}^{i,n-l,Q} + \sum_{l=1}^{m} \tilde{\alpha}_l H_i^{-1} J_{ij} \underline{u}^{j,n-l,Q}, \qquad (5.8) \\ q = 1 \dots Q : \underline{u}^{i,n,q} &= -\sum_{l=1}^{k} \beta_l H_i^{-1} \underline{u}^{i,n-l,Q} + H_i^{-1} J_{ij} \underline{u}^{j,n,q-1}. \qquad (5.9) \end{aligned}$$

To understand the stability properties of this PC scheme, we cast (5.8) and (5.9) into a system of the form $\underline{z}^n = G\underline{z}^{n-1}$, where $G = C^Q P$ is a product of the matrix P corresponding to the predictor step that uses the EXTm scheme (5.8) and matrix C corresponding to the Q corrector steps (5.9).

Defining

$$\underline{z}^{n,q} = [\underline{u}^{1,n,q^T}\ \underline{u}^{2,n,q^T}\ \underline{u}^{1,n-1,q^T}\ \underline{u}^{2,n-1,q^T}\ \underline{u}^{1,n-2,q^T}\ \underline{u}^{2,n-2,q^T}\ \underline{u}^{1,n-3,q^T}\ \underline{u}^{2,n-3,q^T}]^T, \qquad (5.10)$$

the predictor step ($q = 0$) is

$$\underbrace{\begin{bmatrix} \underline{u}^{1,n,0} \\ \underline{u}^{2,n,0} \\ \underline{u}^{1,n-1,0} \\ \underline{u}^{2,n-1,0} \\ \underline{u}^{1,n-2,0} \\ \underline{u}^{2,n-2,0} \\ \underline{u}^{1,n-3,0} \\ \underline{u}^{2,n-3,0} \end{bmatrix}}_{\underline{z}^{n,0}} = \underbrace{\begin{pmatrix} -\beta_1 H_1^{-1} & \tilde{\alpha}_1 H_1^{-1} J_{12} & -\beta_2 H_1^{-1} & \tilde{\alpha}_2 H_1^{-1} J_{12} & -\beta_3 H_1^{-1} & \tilde{\alpha}_3 H_1^{-1} J_{12} & 0 & 0 \\ \tilde{\alpha}_1 H_2^{-1} J_{21} & -\beta_1 H_2^{-1} & \tilde{\alpha}_2 H_2^{-1} J_{21} & -\beta_2 H_2^{-1} & \tilde{\alpha}_3 H_2^{-1} J_{21} & -\beta_3 H_2^{-1} & 0 & 0 \\ I_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & I_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & I_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & I_2 & 0 & 0 \end{pmatrix}}_{P} \underbrace{\begin{bmatrix} \underline{u}^{1,n-1,Q} \\ \underline{u}^{2,n-1,Q} \\ \underline{u}^{1,n-2,Q} \\ \underline{u}^{2,n-2,Q} \\ \underline{u}^{1,n-3,Q} \\ \underline{u}^{2,n-3,Q} \\ \underline{u}^{1,n-4,Q} \\ \underline{u}^{2,n-4,Q} \end{bmatrix}}_{\underline{z}^{n-1,Q}} \qquad (5.11)$$

and the corrector step (with $q = 1 \dots Q$) is

$$\underbrace{\begin{bmatrix} \underline{u}^{1,n,q} \\ \underline{u}^{2,n,q} \\ \underline{u}^{1,n-1,q} \\ \underline{u}^{2,n-1,q} \\ \underline{u}^{1,n-2,q} \\ \underline{u}^{2,n-2,q} \\ \underline{u}^{1,n-3,q} \\ \underline{u}^{2,n-3,q} \end{bmatrix}}_{\underline{z}^{n,q}} = \underbrace{\begin{pmatrix} 0 & H_1^{-1} J_{12} & -\beta_1 H_1^{-1} & 0 & -\beta_2 H_1^{-1} & 0 & -\beta_3 H_1^{-1} & 0 \\ H_2^{-1} J_{21} & 0 & 0 & -\beta_1 H_2^{-1} & 0 & -\beta_2 H_2^{-1} & 0 & -\beta_3 H_2^{-1} \\ 0 & 0 & I_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & I_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & I_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & I_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_2 \end{pmatrix}}_{C} \underbrace{\begin{bmatrix} \underline{u}^{1,n,q-1} \\ \underline{u}^{2,n,q-1} \\ \underline{u}^{1,n-1,q-1} \\ \underline{u}^{2,n-1,q-1} \\ \underline{u}^{1,n-2,q-1} \\ \underline{u}^{2,n-2,q-1} \\ \underline{u}^{1,n-3,q-1} \\ \underline{u}^{2,n-3,q-1} \end{bmatrix}}_{\underline{z}^{n,q-1}} \qquad (5.12)$$

Using (5.11) and (5.12), the spectral radius of $G = C^Q P$ can be used to determine the stability of the PC scheme for different grid sizes, overlap widths, extrapolation orders, and corrector iterations. We note that using the parameters $\tilde{N}$ and K, the grid overlap width can be calculated as $K\Delta x = K/(2\tilde{N} - K + 2)$.

5.2 Stability Results

To understand the stability properties of the OS-based PC scheme, we start with $\tilde{N} = 32$ and $K = 5$. For different BDFk/EXTm schemes, we vary the number of corrector iterations Q to see how the stability of the scheme changes with the nondimensional time-step size ($\nu\Delta t/\Delta x^2$).

Figure 5.4 shows the spectral radius for the BDFk/EXTm schemes with different Q. We observe that the BDFk/EXT1 schemes are unconditionally stable, and the use of high-order extrapolation ($m > 1$) for interdomain boundary data requires correct iterations for stability. These results are also indicated in [55]. However, our analysis suggests that the BDFk/EXT2 and BDFk/EXT3 schemes are more stable when Q is odd. To the best of our knowledge, this behavior has not been observed in the current literature, and it corresponds to the stability behavior (e.g., Fig. 5.1) that we have observed in our Schwarz-SEM framework.

5.2.1 Effect of increasing subdomain overlap

The subdomain overlap width has an impact on the convergence of Schwarz-based methods (1.4). For practical purposes, one would like to minimize the overlap width to minimize the total number of elements needed for modeling a domain. Thus, we look at the impact of overlap width on the stability of the PC scheme. Since we are mainly interested in the high-order extrapolation scheme ($m = 1$ is unconditionally stable as shown in the previous section), we look at the results for the BDF2/EXT2, BDF3/EXT2, and BDF3/EXT3 schemes, with $\tilde{N} = 32$, and change the grid overlap parameter, K.

Figure 5.5-5.7 show that increasing the overlap width increases the stability range over which a given scheme is stable for a specified number of corrector iterations Q. We also notice that high-order extrapolation schemes are less stable as compared to their low order counterparts, e.g., the BDF3/EXT3 scheme is less stable than the BDF3/EXT2 scheme, for a given Q. We also notice that changing the overlap width K makes a significant difference in the stability of a scheme.

5.2.2 Effect of increasing grid resolution while keeping overlap fixed

For certain applications, such as the spanwise twisting bar shown in Fig. 1.1, geometric constraints can limit the maximum allowable overlap width between different subdomains. In these cases, if the overlap

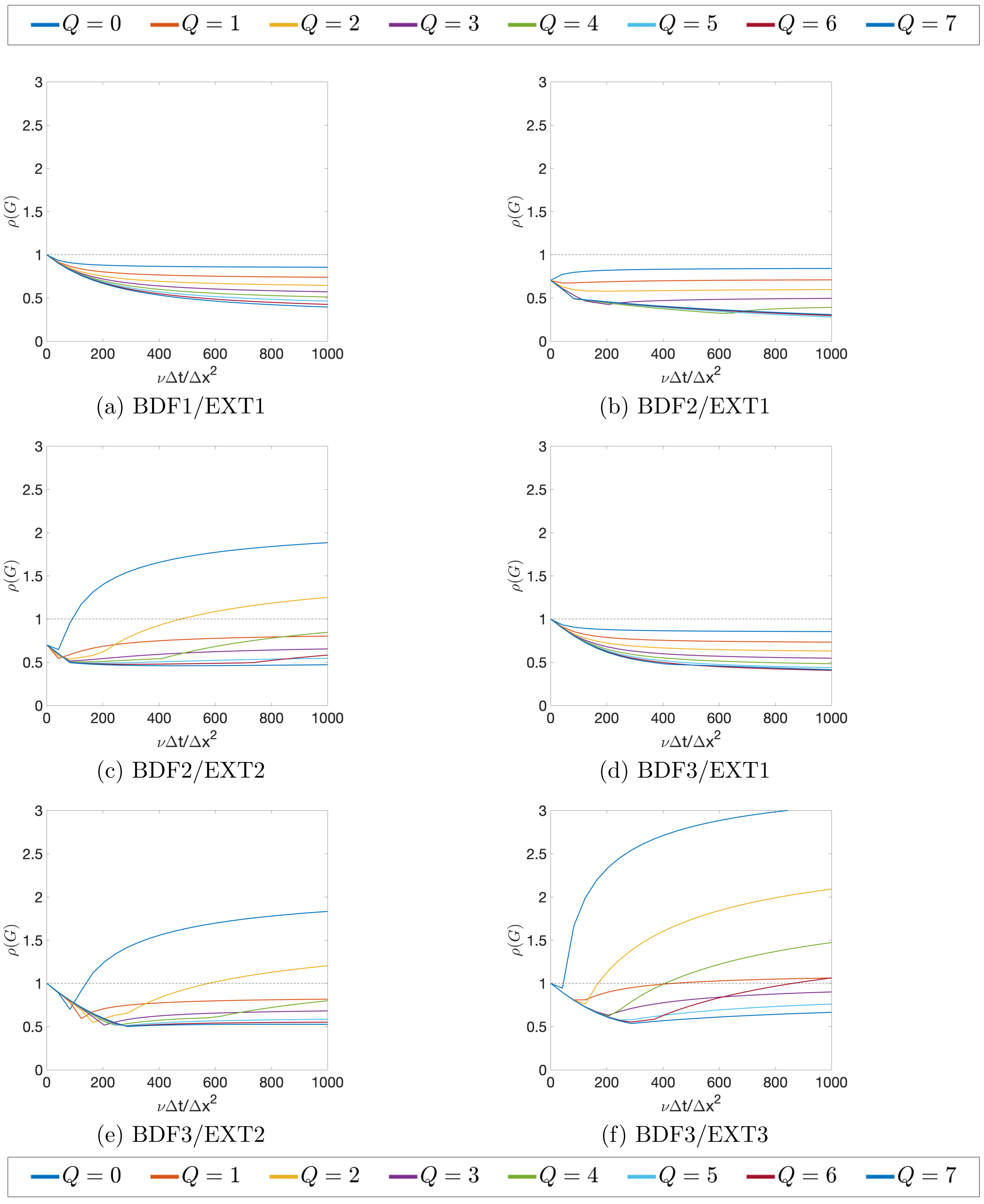
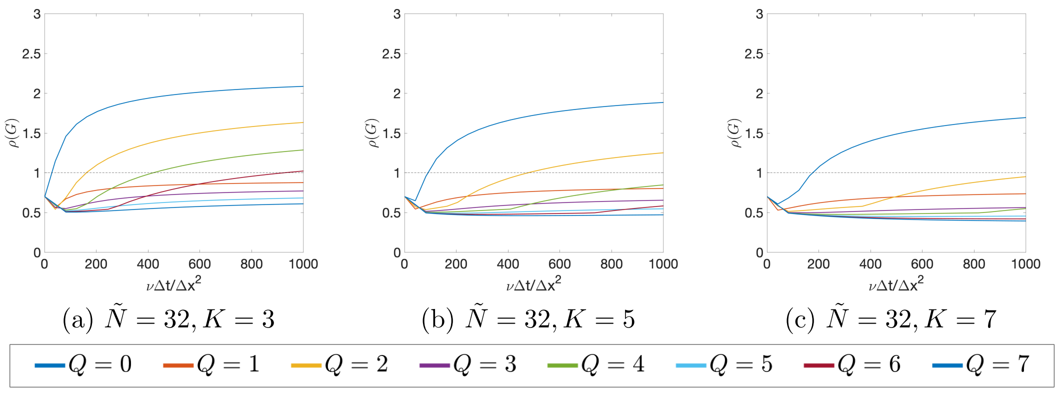


Figure 5.4: Spectral radius $\rho(G)$ versus nondimensional time $\frac{\nu\Delta t}{\Delta x^2}$ for different BDFk/EXTm schemes with $Q = 0\ldots7$, $\tilde{N} = 32$ and $K = 5$.

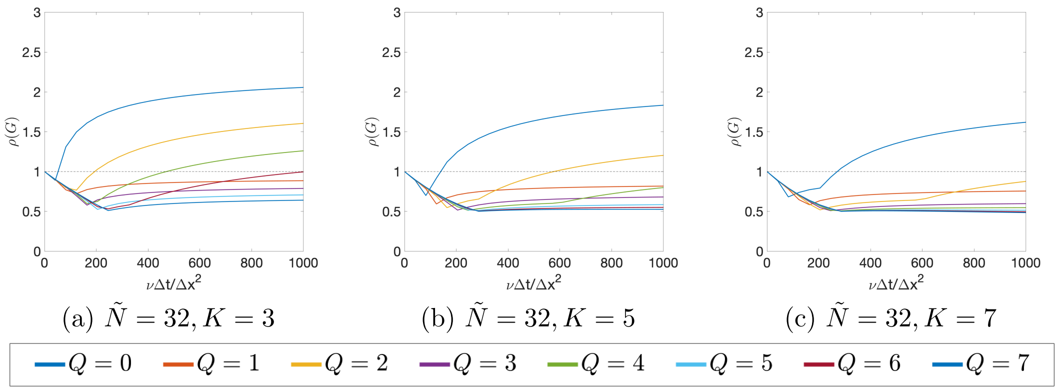
width is not enough for a stable predictor-corrector scheme with $m > 1$, the application of the Schwarz-SEM framework is limited. Thus, we look at the impact of increasing the grid resolution, while keeping the overlap width fixed, in order to understand if increasing the grid resolution can help stabilize the PC scheme.

Figure 5.8-5.10 shows the impact of increasing the grid resolution while keeping the overlap width $(K\Delta x)$



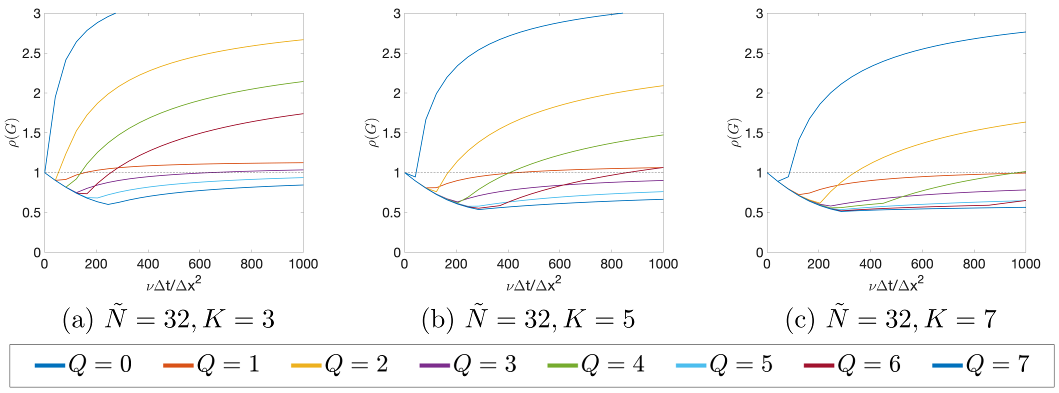
(a) $\tilde{N} = 32, K = 3$ (b) $\tilde{N} = 32, K = 5$ (c) $\tilde{N} = 32, K = 7$

Figure 5.5: Spectral radius $\rho(G)$ versus nondimensional time $\frac{\nu \Delta t}{\Delta x^2}$ for the BDF2/EXT2 scheme with $\tilde{N} = 32$, and varying K.



(a) $\tilde{N} = 32, K = 3$ (b) $\tilde{N} = 32, K = 5$ (c) $\tilde{N} = 32, K = 7$

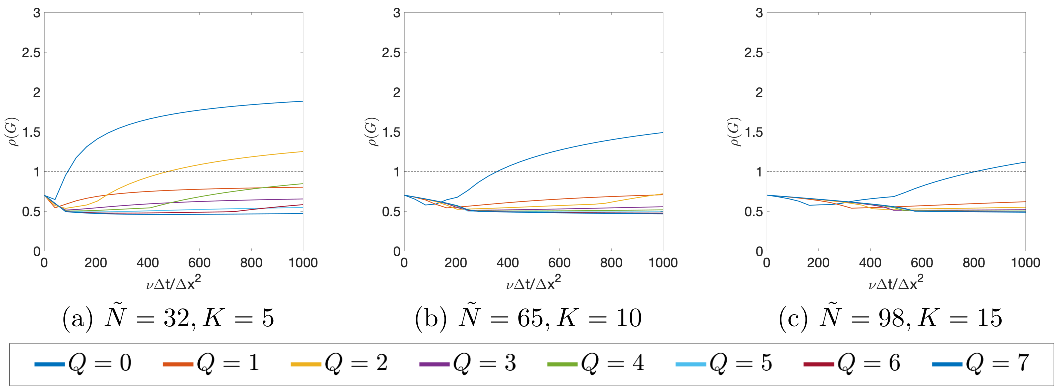
Figure 5.6: Spectral radius $\rho(G)$ versus nondimensional time $\frac{\nu \Delta t}{\Delta x^2}$ for the BDF3/EXT2 scheme with $\tilde{N} = 32$, and varying K.



(a) $\tilde{N} = 32, K = 3$ (b) $\tilde{N} = 32, K = 5$ (c) $\tilde{N} = 32, K = 7$

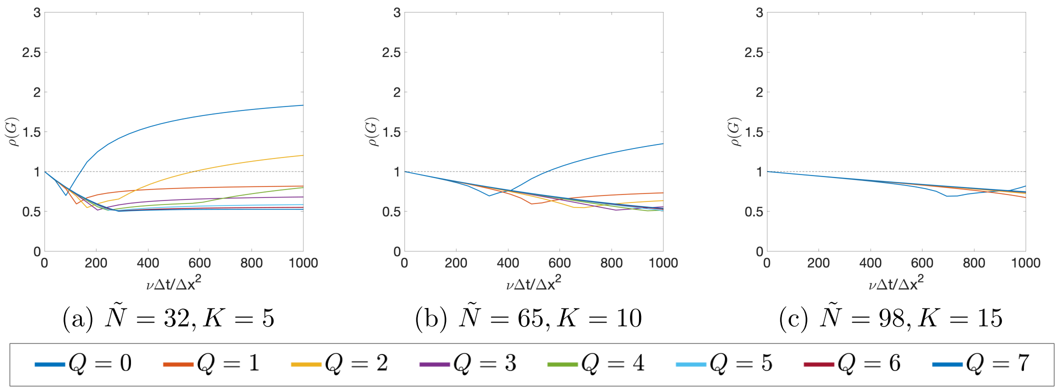
Figure 5.7: Spectral radius $\rho(G)$ versus nondimensional time $\frac{\nu \Delta t}{\Delta x^2}$ for the BDF3/EXT3 scheme with $\tilde{N} = 32$, and varying K.

fixed. As we can see, increasing the grid resolution has a stabilizing effect on the PC scheme.



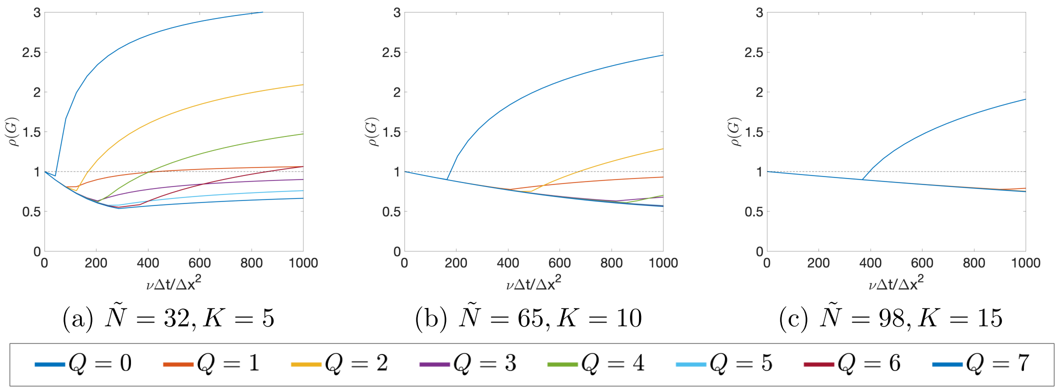
(a) $\tilde{N} = 32, K = 5$ (b) $\tilde{N} = 65, K = 10$ (c) $\tilde{N} = 98, K = 15$

Figure 5.8: Spectral radius $\rho(G)$ versus nondimensional time $\frac{\nu\Delta t}{\Delta x^2}$ for the BDF2/EXT2 scheme with $\tilde{N}$ and K varying such that $K\Delta x$ is fixed.



(a) $\tilde{N} = 32, K = 5$ (b) $\tilde{N} = 65, K = 10$ (c) $\tilde{N} = 98, K = 15$

Figure 5.9: Spectral radius $\rho(G)$ versus nondimensional time $\frac{\nu\Delta t}{\Delta x^2}$ for the BDF3/EXT2 scheme with $\tilde{N}$ and K varying such that $K\Delta x$ is fixed.



(a) $\tilde{N} = 32, K = 5$ (b) $\tilde{N} = 65, K = 10$ (c) $\tilde{N} = 98, K = 15$

Figure 5.10: Spectral radius $\rho(G)$ versus nondimensional time $\frac{\nu\Delta t}{\Delta x^2}$ for the BDF3/EXT3 scheme with $\tilde{N}$ and K varying such that $K\Delta x$ is fixed.

5.3 Discussion on Stability for Odd- and Even-Corrector Iterations

The stability analysis presented in this chapter show that the high-order PC scheme is relatively more stable when Q is odd as compared to when Q is even. This analysis was done using the unsteady diffusion equation and qualitatively captures the stability behavior that has been observed in the Schwarz-SEM framework. This decrease in stability due to *even* number of corrector iterations is not expected, and there is evidence of similar behavior presented by Stetter in 1968 [102] for a predictor-corrector scheme for solving an ODE.

In [102], Stetter discusses the stability of a scheme that uses the third-order Adam-Bashforth (AB3) method for the predictor step and the second-order Adam-Moulton (AM2) method for Q corrector steps to solve an ODE of the form $\frac{dy}{dt} = \lambda y$. While it is clear from the results presented by Stetter that there is a difference in the stability behavior between even- and odd-Q for the AB3-AM2 PC scheme, Stetter does not comment on it. Instead, Stetter describes a general method for improving the stability of the predictor-corrector scheme for any number of corrector iterations. In this section, we first derive the stability polynomials of the AB3-AM2 scheme and show the stability diagram of this PC scheme for a different number of corrector iterations. Using these stability polynomials, we show that the AB3-AM2 scheme leads to a difference in the stability of odd- and even-Q, which is similar to what we have observed for solving the unsteady heat equation. Next, we show how the stability analysis of the AB3-AM2 PC scheme for solving an ODE is connected to the BDFk/EXTm PC scheme for solving the unsteady heat equation, which could possible explain why we observe this odd-even stability behavior in the Schwarz-SEM framework.

5.3.1 Predictor-corrector scheme for ODE

The AB3-AM2 predictor-corrector scheme for solving $\frac{dy}{dt} = \lambda y$ is

$$\tilde{y}_{n+1} = y_n + \frac{h}{12}(23y_n^{'} - 16y_{n-1}^{'} + 5y_{n-2}^{'}), \tag{5.13}$$

$$y_{n+1} = y_n + \frac{h}{2}(\tilde{y}_{n+1}^{'} + y_n^{'}), \tag{5.14}$$

where (5.13) is the predictor step using AB3, and (5.14) is the corrector step using AM2. We use y_n to represent the solution y at time level t^n, h for the constant time-step size, $y_n^{'}$ for the derivative $\frac{dy_n}{dt}$, and $\tilde{y}_{n+1}$ as the solution from the predictor step. Such a scheme is called PECE, short for predictor-evaluate-corrector-evaluate. The terms on the right hand side of (5.13) are evaluated using the solution from (5.14) at the previous time-step.

Substituting $y' = \lambda y$, we can write the AB3-AM2 system as

$$\tilde{y}_{n+1} = y_n + \frac{h\lambda}{12}(23y_n - 16y_{n-1} + 5y_{n-2}), \tag{5.15}$$

$$y_{n+1} = y_n + \frac{h\lambda}{2}(\tilde{y}_{n+1} + y_n), \tag{5.16}$$

and substituting $\tilde{y}_{n+1}$ from (5.15) into (5.16), we get

$$y_{n+1} = y_n + \frac{h\lambda}{2}\left(2y_n + \frac{h\lambda}{12}(23y_n - 16y_{n-1} + 5y_{n-2})\right). \tag{5.17}$$

Assuming that z_n is the exact solution at t^n, which is free of any truncation or rounding errors (unlike $\tilde{y}^n$ and y^n), the equation for error $\epsilon_n = z_n - y_n$ becomes

$$\epsilon_{n+1} = \epsilon_n + \frac{h\lambda}{2}\left(2\epsilon_n + \frac{h\lambda}{12}(23\epsilon_n - 16\epsilon_{n-1} + 5\epsilon_{n-2})\right). \tag{5.18}$$

At this point, we can assume that the error is of the form $\epsilon_n = \rho^n$, and simplify (5.18),

$$\rho^{n+1} = \rho^n + \frac{h\lambda}{2}\left(2\rho^n + \frac{h\lambda}{12}(23\rho^n - 16\rho^{n-1} + 5\rho^{n-2})\right). \tag{5.19}$$

Substituting $\kappa = h\lambda$, and dividing both sides by ρ^{n-2}, we get

$$\rho^3 = \rho^2 + \frac{\kappa}{2}\left(2\rho^2 + \frac{\kappa}{12}(23\rho^2 - 16\rho + 5)\right). \tag{5.20}$$

(5.20) gives us the stability polynomial describing the relationship between κ and ρ for the PC scheme with 1 corrector iteration.

For a generalized predictor-corrector scheme with Q corrector iterations, P(EC)^{Q}E, we can write the AB3-AM2 system as

$$q = 0 : y^0_{n+1} = y^Q_n + \frac{\kappa}{12}(23y^Q_n - 16y^Q_{n-1} + 5y^Q_{n-2}), \tag{5.21}$$

$$0 < q \leq Q : y^q_{n+1} = y^Q_n + \frac{\kappa}{2}(y^{q-1}_{n+1} + y^Q_n), \tag{5.22}$$

where y^q_n represents the qth iterate of the solution y at t^n. Equation (5.21) and (5.22) can be used to generalize the stability polynomials for different Q,

$$\rho^3 = \rho^2\left(1 + \sum_{i=1}^{Q}\frac{\kappa^i}{2^{i-1}} + \frac{23}{12(2^Q)}\kappa^{Q+1}\right) - \frac{16}{12(2^Q)}\rho\kappa^{Q+1} + \frac{5}{12(2^Q)}\kappa^{Q+1}. \tag{5.23}$$

Figure 5.11 shows the stability diagram for the AB3-AM2 predictor-corrector scheme for different Q, and the stability curve for each Q is the $\rho = 1$ contour. As we can see, for real values of $\kappa = \lambda h$, which are represented by the x-axis in Fig. 5.11, the P(EC)^{Q}E scheme is more stable for odd-Q as compared to the corresponding $Q + 1$ case. Here, we are interested in negative real values of $\kappa = \lambda h$ because negative real κ corresponds to negative real values of λ (the time-step size h is always real and positive), which is related to our stability analysis of the unsteady heat equation.

In Fig. 5.11, we have used dashed lines to indicated different negative real values of κ, specifically

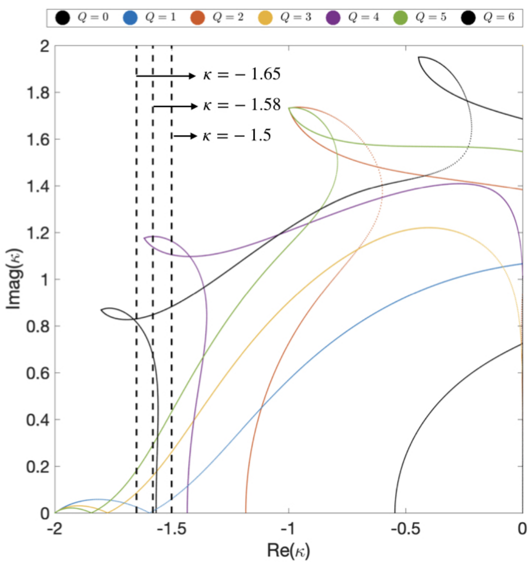
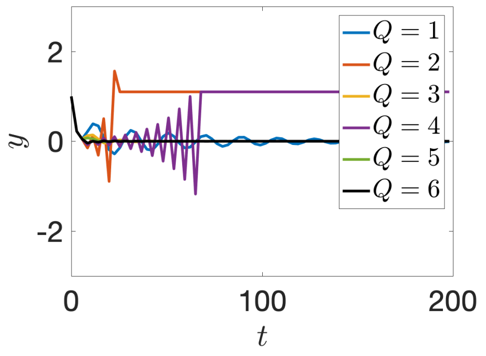


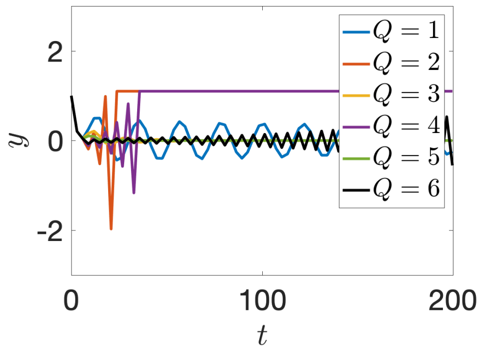
Figure 5.11: Stability diagram for the AB3-AM2 predictor-corrector scheme for different number of corrector iterations. The also indicate $\kappa = -1.5$, -1.58 and -1.65 in the plot, which we use to validate these stability analyses.

$\kappa = -1.5$, -1.58 and -1.65. From the stability diagram, we expect that for $\kappa = -1.5$, $Q = 1$, 3, 5 and 6 lead to a stable scheme, for $\kappa = -1.58$, $Q = 1$, 3 and 5 lead to a stable scheme, and for $\kappa = -1.65$, only $Q = 3$ and 5 lead to a stable scheme. To numerically validate these results, we solve $\frac{dy}{dt} = \lambda y$ with step size $h = 1$ and $\lambda = \kappa = -1.5$, -1.58 and -1.65. Figure 5.12 shows the solution of the ODE $y' = \lambda y$ for different values of λ for different number of corrector iterations. As we can see, these results validate the stability results of the P(EC)^{Q}E AB3-AM2 scheme shown in Fig. 5.11.

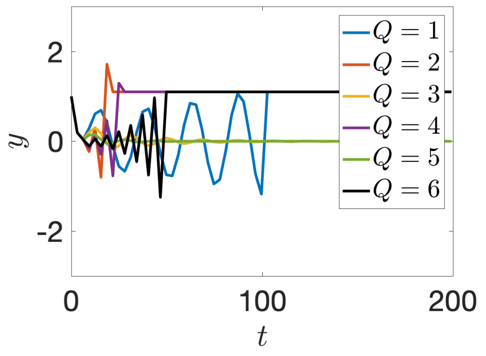
Now that we have validated our stability analysis of the AB3-AM2 PC scheme, we can connect this stability analysis with the analysis of the unsteady heat equation.



(a) $\kappa = -1.5$



(b) $\kappa = -1.58$



(c) $\kappa = -1.65$

Figure 5.12: Solution y for the ODE $\frac{dy}{dt} = \lambda y$ for different $\kappa = \lambda \Delta t$ using the AB3-AM2 predictor-corrector scheme.

5.3.2 Similarity in stability behavior of high-order predictor-corrector methods for ODEs and PDEs

In the previous section, we have presented a stability diagram for the P(EC)^{Q}E AB3-AM2 scheme for solving the ODE $y' = \lambda y$. Specifically, we have focussed on the stability behavior of this scheme for different Q for negative real values of λ and observed that odd-Q is more stable than even-Q. We focus on the negative real values of λ for the ODE $y' = \lambda y$, because the boundary value problem that we have considered with overlapping grids in Section 5.1.2 ($u_t = \nu u_{xx}$ with $\nu > 0$ and homogeneous boundary conditions) has negative real eigenvalues as well.

Based on the stability analyses presented in this chapter and the numerical experiments that we have done with the Schwarz-SEM framework, we conjecture that for BVP with negative real eigenvalues, high-order PC schemes lead to a difference in the stability between odd- and even-corrector iterations. In future work, we will continue this work to determine the fundamental reasoning behind this odd-even pattern, and look at the stability of high-order PC methods for general boundary value problems, including the unsteady Stokes problem (2.47) in our SEM-based framework.

5.4 Improving the Stability for Even-Corrector Iterations

In the predictor-corrector scheme discussed so far, since odd-Q is more stable than even-Q, we have to increase Q by 2 (e.g., increase $Q = 1$ to $Q = 3$) if the number of corrector iterations is not sufficient. When using a predictor-corrector scheme, one wants to minimize the number of corrector iterations for reducing the computational cost of a calculation. In this section, we develop an improved predictor-corrector scheme to address this issue.

5.4.1 Stability of the improved predictor-corrector scheme for an ODE

The paper by Stetter discusses how the stability of the AB3-AM2 predictor-corrector scheme can be improved by using a linear combination of solution from each corrector iteration to determine the final solution as

$$y_{n+1} = \sum_{q=0}^{Q} \gamma_q y_{n+1}^{q}, \qquad \sum_{q=0}^{Q} \gamma_q = 1, \tag{5.24}$$

where γ_q is some weight corresponding to the solution of qth corrector iteration at each time-step. Using this approach, Stetter shows that the stability region for the PC scheme could be extended. However, Stetter does not comment on the difference in stability for odd- and even- number of corrector iterations. We extend Stetter's idea to improve the stability for when Q is even. In this improved scheme, if Q is even, instead of using y_{n+1}^{Q-1} to compute y_{n+1}^{Q} in (5.14), we use a linear combination of y_{n+1}^{Q-1} and y_{n+1}^{Q-2}. The new predictor-corrector scheme is

$$q = 0 : y_{n+1}^{0} = y_n^Q + \frac{h\lambda}{12}(23y_n^Q - 16y_{n-1}^Q + 5y_{n-2}^Q), \tag{5.25}$$

$$q = 1 \dots Q-1 : y_{n+1}^{q} = y_n^Q + \frac{h\lambda}{2}(y_{n+1}^{q-1} + y_n^Q), \tag{5.26}$$

$$q = Q : y_{n+1}^{q} = y_n^Q + \frac{h\lambda}{2}(\gamma y_{n+1}^{Q-1} + (1-\gamma) y_{n+1}^{Q-2} + y_n^Q), \tag{5.27}$$

where γ is a constant. $\gamma = 1$, if Q is odd, and $0 < \gamma < 1$, if Q is even. As we can see $\gamma = 1$ for odd-Q recovers the original scheme (5.21,5.22).

The rationale behind this new predictor-corrector scheme (5.25-5.27) is that we do not want to modify the convergence properties of the original PC scheme if Q is odd. Thus, we modify only the last corrector iteration (y_{n+1}^{Q}), when Q is even. We also note that for simplicity, we currently use a linear combination of only the two most recent solutions (y_{n+1}^{Q-1} and y_{n+1}^{Q-2}) instead of additional older solutions ($y_{n+1}^{1} \dots y_{n+1}^{Q-1}$).

Stability analysis show that the new predictor-corrector scheme (5.25-5.27) has better stability for even-Q in comparison to the original scheme (5.21,5.22), if we consider negative real κ. Figure 5.13 shows how the stability varies for the predictor-corrector scheme for $Q = 2$ with γ =0.25, 0.5, 0.75 and 1, $Q = 1$ and $Q = 3$. $Q = 1$ is equivalent to using $Q = 2$ with $\gamma = 0$, and $Q = 2$ with $\gamma = 1$ corresponds to the original scheme for $Q = 2$. As we can see, $Q = 2$ can be more stable than $Q = 1$, depending on the value of γ. In this example, we see that $\gamma = 0.5$ yields the most stable scheme.

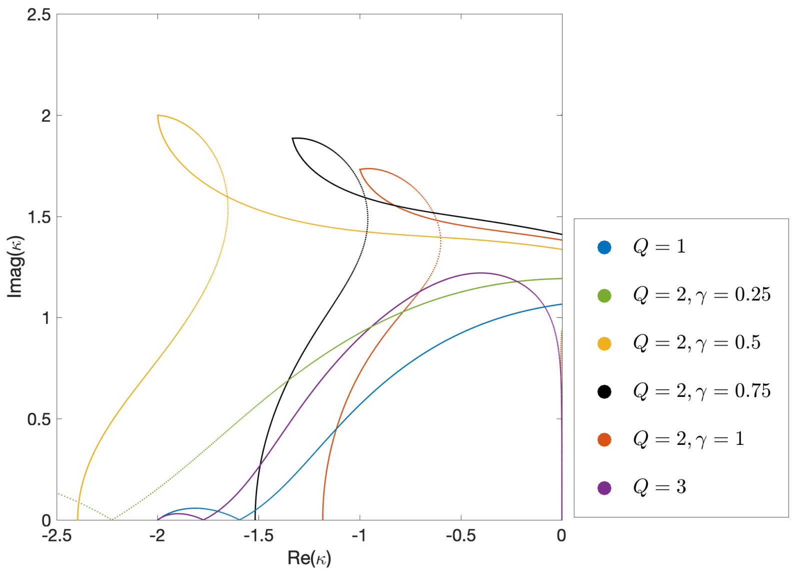


Figure 5.13: Stability diagram for the improved AB3-AM2 predictor-corrector scheme for different γ.

Figure 5.14 shows the stability diagram of the original scheme for $Q = 1, 2, 3$, and 4, and the stability diagram of the new scheme for $Q = 2$ and 4 with $\gamma = 0.5$. As we can see, the new PC scheme has better stability for even-Q, in comparison to the original scheme.

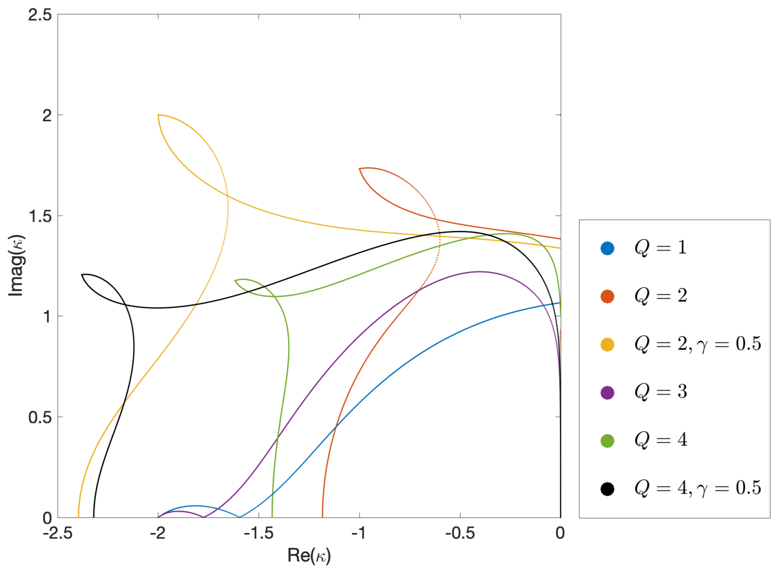


Figure 5.14: Stability diagram for the improved AB3-AM2 predictor-corrector scheme for different γ.

In the PC scheme presented in this section, we use a linear combination of two most recent solutions for the last corrector iteration to improve the stability behavior of the PC scheme for even-Q. In future work, we will look at additional ways to improve the stability of this PC scheme. For example, we will explore methods to incorporate more solutions (e.g., $y^1_{n+1} \dots y^{Q-1}_{n+1}$) at last corrector iteration such that y^Q_{n+1} can be determined with improved stability and accuracy.

5.4.2 Stability of the improved predictor-corrector scheme for the unsteady heat equation

Here, we extend the idea of the improved predictor-corrector scheme from the previous section to the stability analysis framework that we have developed for the unsteady heat equation (Section 5.1.2). The proposed predictor-corrector scheme is

$$q = 0 : \underline{u}_i^{n,0} = -\sum_{l=1}^{k} \beta_l H_i^{-1} \underline{u}^{i,n-l} + \sum_{l=1}^{m} \tilde{\alpha}_l H_i^{-1} J_{ij} \underline{u}_j^{n-l}, \tag{5.28}$$

$$q = 1 \dots Q-1 : \underline{u}_i^{n,q} = -\sum_{l=1}^{k} \beta_l H_i^{-1} \underline{u}^{i,n-l} + H_i^{-1} J_{ij} \underline{u}_j^{n,q-1}, \tag{5.29}$$

$$q = Q : \underline{u}_i^{n,q} = -\sum_{l=1}^{k} \beta_l H_i^{-1} \underline{u}^{i,n-l} + H_i^{-1} J_{ij} \Big(\gamma \underline{u}_j^{n,Q-1} + (1-\gamma) \underline{u}_j^{n,Q-2} \Big), \tag{5.30}$$

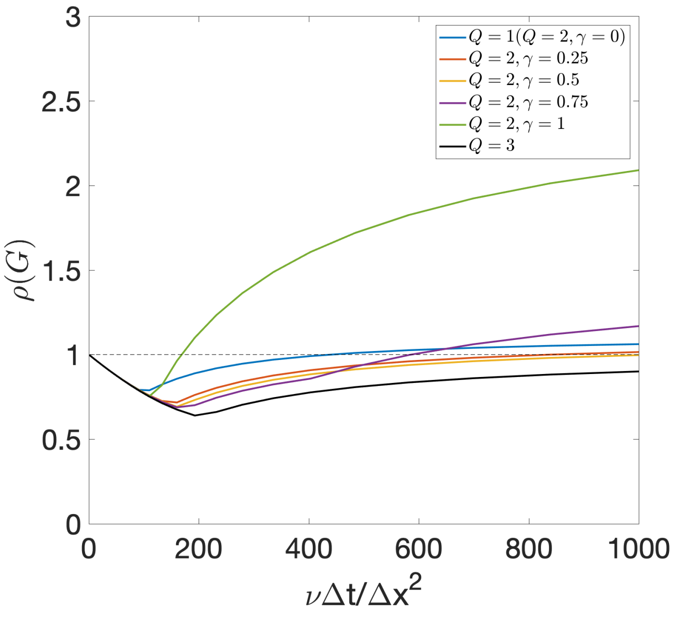


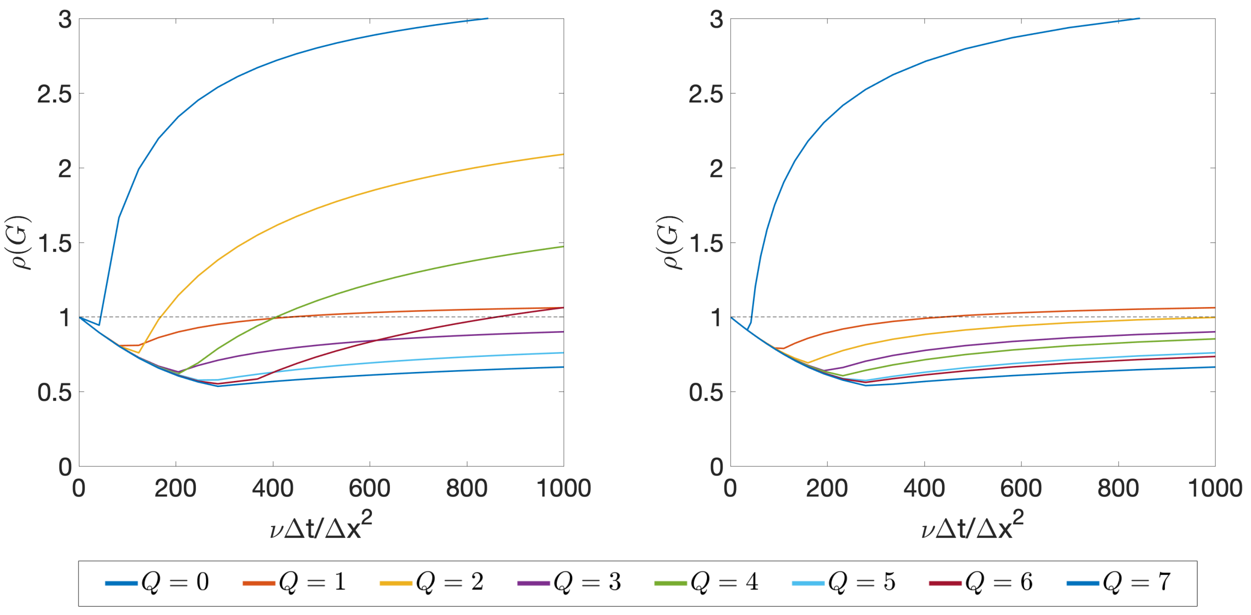
Figure 5.15: Spectral radius $\rho(G)$ versus nondimensional time $\frac{\nu\Delta t}{\Delta x^2}$ for the BDF3/EXT3 scheme with $\tilde{N} = 32$, $K = 5$ for different γ.

where γ is a parameter that can be optimized for improved stability. If Q is odd, we set $\gamma = 1$ to use the original scheme. If Q is even, however, we set $0 \leq \gamma \leq 1$.

Figure 5.15 shows the stability plot for the modified predictor-corrector schemes for different values of γ for $Q = 2$, and $Q = 3$ with $\gamma = 1$. The parameters for grid size are $\tilde{N} = 32$ and $K = 5$. We can see that $Q = 2$ with $\gamma = 1$ leads to the original scheme and $Q = 2$ with $\gamma = 0$ gives the behavior equivalent to $Q = 1$. It is also apparent that the stability of the PC scheme has drastically improved for $Q = 2$, in comparison to the original scheme (Fig. 5.4), and it is no longer more unstable than $Q = 1$ when $\gamma = 0.25$ or 0.5. In fact, we see in Fig. 5.16 that this idea works for $Q = 4$ and 6 as well, and leads to improved stability in comparison to $Q = 3$ and 5, respectively. Figure 5.16 compares the stability plots for the original and improved predictor-corrector scheme. For the improved predictor-corrector scheme, we use $\gamma = 0.5$ for even-Q.

5.5 Validation of the Improved Predictor-Corrector Scheme with Schwarz-SEM Framework

The improved predictor-corrector scheme for even-Q can be readily extended to the Schwarz-SEM framework. In this improved scheme, if Q is even, the last corrector iteration uses a combination of solution from the



(a) Original PC scheme (b) Improved PC scheme

Figure 5.16: Spectral radius $\rho(G)$ versus nondimensional time $\frac{\nu\Delta t}{\Delta x^2}$ for the BDF3/EXT3 scheme with $\tilde{N}=32$ and $K=5$ comparing the original and improved predictor-corrector scheme. $\gamma=0.5$ for even-Q in the improved PC scheme.

previous two iterations ($q=Q-1$ and $Q-2$) instead of just $q=Q-1$. Using this approach, we see that for the Navier-Stokes eigenfunctions test case (Section 3.5), the Schwarz-SEM framework stays stable for even-Q. Fig. 5.17 shows the error variation for $Q=1,2$, 3 and 4 with the original scheme, and compares it to the improved PC scheme with $Q=2$ and 4 for $\gamma=0.5$.

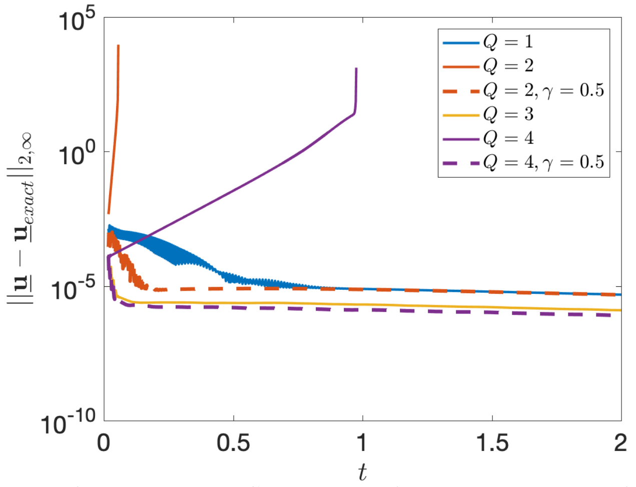
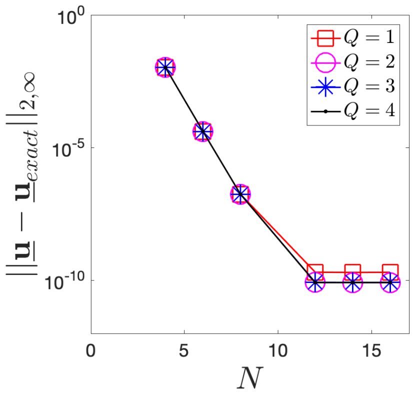


Figure 5.17: Error variation for the Navier-Stokes eigenfunctions test case from Chapter 3 ($S=2$) with different Q using $\Delta t=2\times10^{-3}$ and $N=7$.

Figure 5.18 shows the spatial and third-order temporal convergence for $Q = 1$, 2, 3, and 4 with $m = 3$ using the same boundary conditions and flow parameters that were used for validation in Chapter 3. As we can see, the improved predictor-corrector scheme maintains the spatial and temporal convergence of the SEM solver.



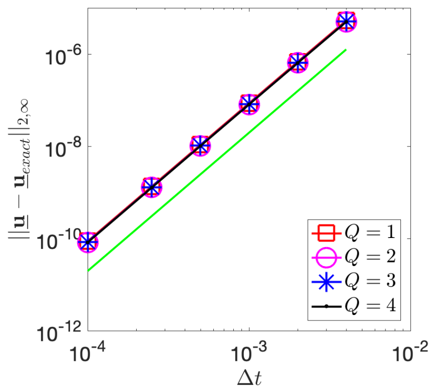


Figure 5.18: (left) Spatial convergence with $\Delta t = 10^{-4}$, and (right) temporal convergence at $N = 13$ for different Q. $\gamma = 0.5$ for $Q = 2$ and $Q = 4$.

5.6 Summary

In this chapter, we have analyzed the stability properties of a predictor-corrector scheme for solving the 1D unsteady heat equation using the finite difference method. The stability analysis shows that even-Q is less stable than odd-Q when a high-order scheme is used to extrapolate interdomain boundary data. These results have helped us understand the behavior that we have observed in the Schwarz-SEM framework. We have also found that a similar difference between even- and odd-Q is observed in the stability analysis of the AB3-AM2 based predictor-corrector scheme for solving an ODE [102]. By extending the ideas discussed in [102], we have improved the stability of the PC scheme for solving the unsteady diffusion equation, for even-Q. We have also implemented this improved PC scheme in our Schwarz-SEM framework and observed that in addition to stabilizing the INSE solver, it maintains the exponential convergence with polynomial order N, and third-order temporal convergence. In future work, we will continue the stability analysis presented in this chapter to understand the fundamental reasoning behind the difference in stability for even- and odd-Q. We will also look at ways to extend the FD-based stability analysis framework to an SEM-based formulation in higher-space dimensions to better understand the stability properties of the PC scheme for solving the INSE in the Schwarz-SEM framework.

Chapter 6

Multirate Time-Stepping Scheme for the Schwarz-SEM framework

In previous chapters, we have described the Schwarz-SEM framework for time-advancing the solution of the incompressible Navier-Stokes equations using a predictor-corrector scheme. An advantage of the Schwarz-SEM framework is that it allows us to use grids of varying densities, depending on the scales of fluid structures or geometry in each subdomain. In this chapter, we describe how we can exploit the difference in the resolution between different grids, to develop a novel multirate time-stepping strategy for the INSE. We start with a description of a method for solving the INSE with different time-step sizes in different subdomains. Next, we extend the stability framework from the previous chapter to analyze the stability of our multirate time-stepping strategy.

6.1 Motivation

In a monodomain SEM framework, the time-step size used for advancing the solution of the INSE is based on the maximum CFL number in the spectral element mesh, which is determined as

$$\text{CFL } = \forall_{i\in\Omega}\left|\frac{c_i \Delta t}{\Delta x_i}\right|_{\infty}, \tag{6.1}$$

where Δt is the time-step size, c_i and Δx_i are the velocity and local grid spacing size associated with GLL point i, respectively. At each time-step, Δt should be such that the CFL number of the grid is below the (stability constrained) maximum allowable CFL number that is determined by the temporal discretization (BDFk/EXTk in our framework). An advantage of using the Schwarz-SEM framework is that it allows us to use grids of varying resolution depending on the scales of fluid structures or geometry in each subdomain, as described in Chapter 1 (e.g., Fig. 1.5). Due to this difference in the physics of the flow (which impacts velocity $\mathbf{u}$) and resolution of the grid (which impacts Δx), the CFL number (6.1) can be very different between different overlapping grids. In the current PC scheme, the time-step size is the same for all the overlapping subdomains in a calculation. This strategy is referred to as a *single-rate* time-stepping scheme, and it can lead to an unnecessarily small time-step in the subdomains that have slower time-scales of fluid motion or coarser grid resolution, in comparison to subdomains with faster time-scales.

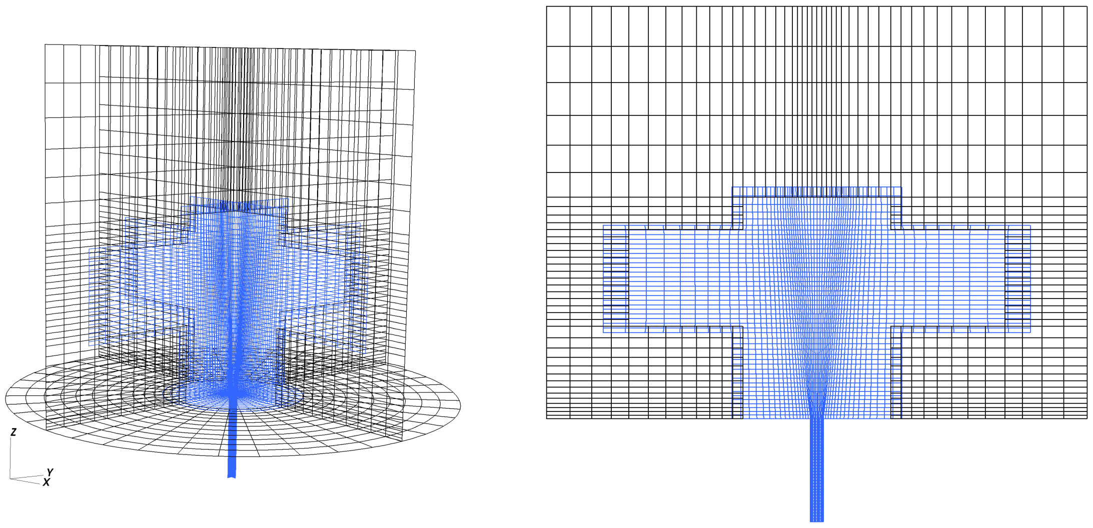


Figure 6.1: (left) Three-slice view and (right) slice view of the axisymmetric overlapping grids used to model thermal plume in a stably stratified environment.

Consider the axisymmetric overlapping grids shown in Fig. 6.1, which are used to model a buoyant plume in a stably stratified environment. Figure 6.1(left) shows a three-slice view and Fig. 6.1(right) shows a slice view of the axisymmetric grids. Figure 6.2 shows a snapshot of the velocity magnitude, where the fluid enters the tank through a pipe at its bottom. The incoming fluid has a lower density in comparison to its surroundings at the entrance. Consequently, the plume rises due to buoyancy effects, until it reaches a height at which the density of the plume is similar to the density of the background fluid. The plume settles at this *trapping* height and diffuses outwards. This problem is discussed in detail in Section 9.7.

Due to the physics of the flow in this buoyant plume, the fluid velocity is maximum near the plume entrance inside the tank, and adverse pressure-gradient at the entrance of the tank leads to fine-scale turbulent structures in that region. In the far-field, away from the plume, the fluid velocity decreases and the flow becomes less chaotic. Figure 6.3 shows a snapshot of local CFL number at each GLL point in the grid used for the monodomain calculation (Fig. 1.3). As expected, the CFL number is maximum near the plume entrance and is two orders of magnitudes smaller in the far-field.

Overlapping grids are an effective method for this problem due to the disparate difference in fluid scales between different parts of the domain. The Schwarz-SEM framework that we have developed so far, however, does not allow us to leverage the difference in the CFL constraint between the overlapping grids.

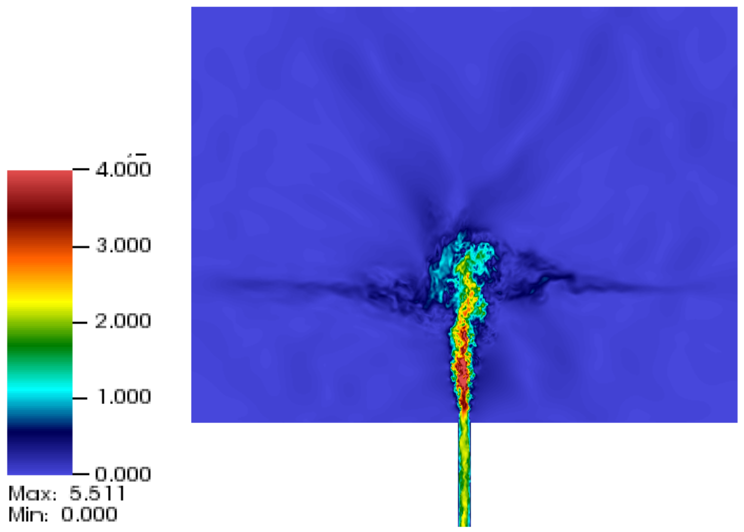


Figure 6.2: Velocity magnitude plot from the monodomain calculation of the thermal plume in a stably stratified environment.

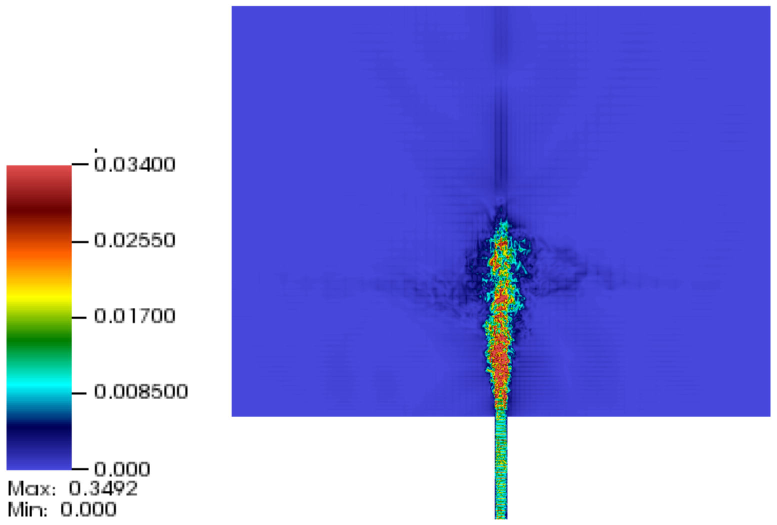


Figure 6.3: CFL plot from the monodomain calculation of the thermal plume in a stably stratified environment.

6.2 Background

Multirate time-stepping methods were first presented in the pioneering work by Rice in 1960 [57]. Therein, Rice presented a method of integrating a system of two ordinary differential equations with different time step sizes using the third-order Runge-Kutta method. The stability of multirate methods for ODEs was analyzed by Gear [103], who showed how coupling between the slow and fast moving components of the ODEs impacted the stability of the solution process. Since then, much work has been done on developing multirate methods for ODEs and PDEs, especially for parabolic and hyperbolic problems [58, 59, 104, 105, 106, 107, 108, 109, 110, 111]. In the context of Navier-Stokes equations, Mikida et al. [107] have introduced an FD-based multirate time-stepping method for compressible flow.

Most of the existing multirate based methods split the solution of ODEs into fast and slow moving components, and solve the ODEs sequentially using a fastest-first or slowest-first approach [103, 105, 106, 107, 108]. While generally useful, a drawback of slowest-first or fastest-first schemes is that they limit the parallelism of the calculation since subdomains integrate the PDE sequentially (similar to the alternating Schwarz method described in Section 1.2). Another popular approach for multirate time-stepping is to partition the elements in a domain into different groups, based on criteria such as the element size [58,59,111], and use a different time-step size for each group. This approach is not straightforward to apply in the monodomain SEM framework because the pressure is coupled to the divergence-free constraint for the INSE (2.29).

Multirate time-steppers are virtually nonexistent for the incompressible Navier-Stokes equations because the solution is very sensitive to the pressure, which satisfies an elliptic Poisson or pseudo-Poisson problem at every timestep. While multirate methods may not be a feasible approach for a single conforming domain, overlapping grids allows us to develop a multirate scheme that leverages the difference in the physics and grid sizes between different subdomains.

Here, we present a novel approach for multirate time-stepping for overlapping grids, which is parallel (integrates all subdomains simultaneously) and maintains the spatial and temporal convergence properties of the underlying SEM solver.

6.3 Methodology

In this section, we describe the multirate time-stepping scheme that we have developed for the Schwarz-SEM framework. We start with an example where the time-step ratio between two subdomains is 2, and describe our strategy for time-advancing the solution of the INSE. We then show that this multirate time-stepping strategy can be extended to an arbitrary step size ratio. For simplicity, we restrict the time-step ratio to be an integer in the Schwarz-SEM framework.

Consider the schematic in Fig. 6.4, which shows the discrete time values for subdomains with coarser/slower (Ω^c) and faster (Ω^f) time-scales[1]. We assume in this example that the physics in Ω^f and Ω^c is such that the maximum CFL in each subdomain constraints the time-step size to Δt_f and Δt_c, respectively, with a

[1]In this chapter, we reserve the superscripts/subscripts c and f to represent parameters associated with subdomains with slower and faster time-scales.

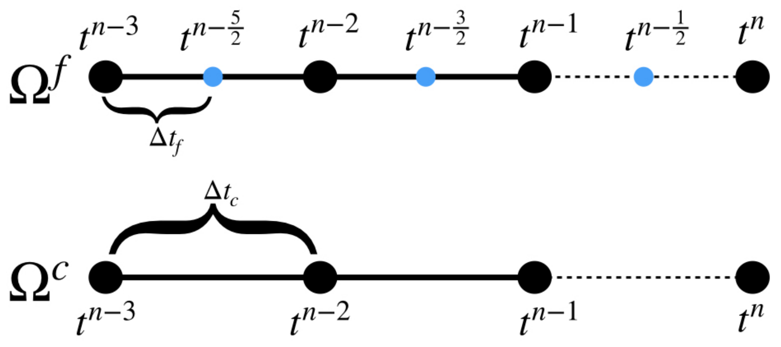


Figure 6.4: Schematic showing different time levels for subdomains with slower (Ω^c) and faster (Ω^f) time-scales.

time-step ratio $\eta = \Delta t_c/\Delta t_f = 2$. Here, assuming that the solution is known up to discrete time t^{n-1}, the INSE is solved twice in Ω^f with a time-step size of Δt_f and once in Ω^c with a time-step of Δt_c, to obtain the solution at t^n. This strategy includes, for Ω^f, the solution at time $t^{n-\frac{1}{2}}$.

In a slowest- or fastest-first method, the solution would be advanced in either of the domains (e.g., say Ω^f) to obtain the solution at time t^n, which would then be used to obtain interdomain boundary data for advancing the solution in the other domain (e.g., Ω^c). For a parallel multirate scheme, we wish to simultaneously advance the solution in Ω^f and Ω^c. As a result, the interdomain boundary data will be exchanged prior to starting the solution process such that the sub-time-steps in Ω^f can be completed independently of Ω^c.

Similar to the single-rate time-stepping scheme (e.g., see (3.6)), high-order temporal accuracy will be

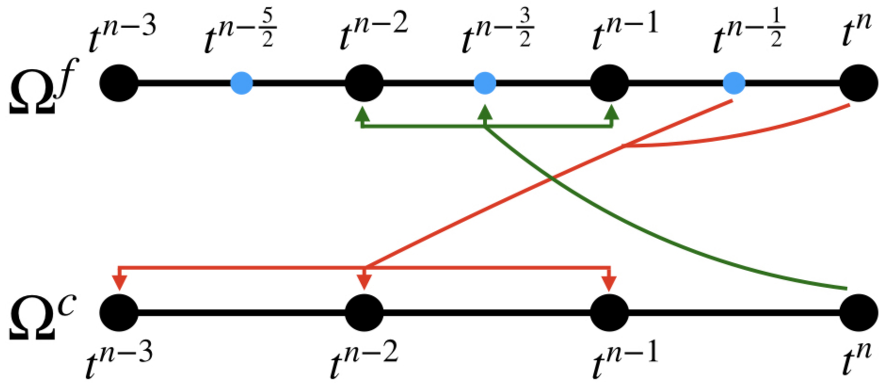


Figure 6.5: Schematic showing interdomain boundary data dependence between Ω^f and Ω^c for the predictor step in the multirate time-stepping scheme.

achieved in the multirate setting by extrapolating interdomain boundary data from the solution at previous time-steps. The key difference in the multirate time-stepping scheme is that the subdomain with faster time-scales has multiple sub-time-steps at each Schwarz iteration.

Figure 6.5 shows how interdomain boundary data is extrapolated for high-order temporal accuracy at the predictor step in the multirate time-stepping scheme. Assuming $\eta = 2$, the interdomain boundary data to solve for time-levels $t^{n-\frac{1}{2}}$ and t^n in Ω^f is interpolated from the known solution in Ω^c at time-levels t^{n-1}, t^{n-2}, and t^{n-3}. Simultaneously, the interdomain boundary data for the solution at t^n in Ω^c is interpolated from the known solution in Ω^f at t^{n-1}, $t^{n-3/2}$, and t^{n-2}.

Once the solution is known at time t^n, correction iterations are needed (similar to the single-rate time-stepping scheme) in order to stabilize the solution if high-order extrapolation is used for interdomain boundary data during the predictor step. Again, the difference in these corrector iterations in comparison to the corrector iterations for single-rate time-stepping is that Ω^f has multiple sub-time-steps at each corrector iteration. Figure 6.6 shows a schematic of how the interdomain boundary data can be interpolated in Ω^f and Ω^c for the Q corrector iterations when $\eta = 2$. In Ω^f, the interdomain boundary data for the solution at $t^{n-\frac{1}{2}}$ comes from the solution in Ω^c at the most recent iteration at time t^n and the converged solution at previous time-steps, t^{n-2} and t^{n-1}. For the solution at time t^n, both subdomains can directly exchange the solution from the most recent iteration corresponding to time t^n.

Using this approach, we can now describe our multirate time-stepping scheme for time-advancing the solution in two overlapping grids with a time-step ratio of $\eta = 2$. Recall our notation for single-rate time-stepping from Chapter 3, we use $\boldsymbol{\phi}^{s,n,[q]}$ to denote the solution $\boldsymbol{\phi}$ in Ω^s at the qth Schwarz iteration at time

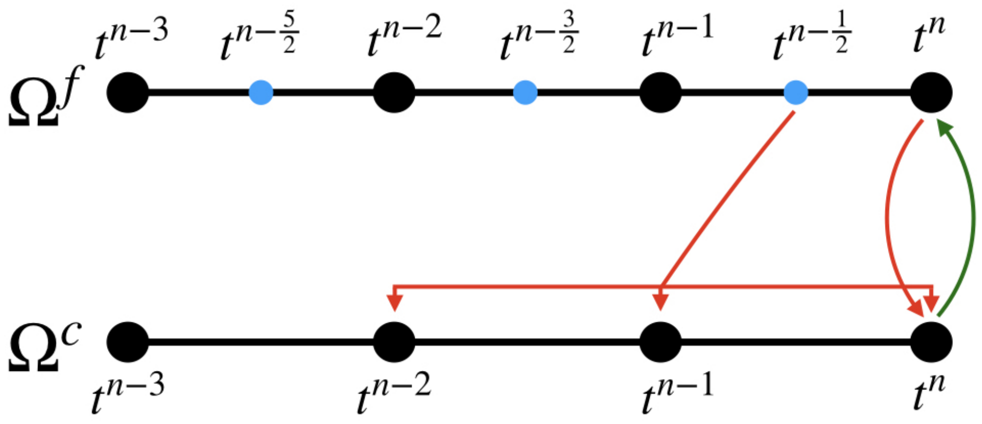


Figure 6.6: Schematic showing interdomain boundary data dependence between Ω^f and Ω^c for corrector iterations in the multirate time-stepping scheme.

t^n. For multirate time-stepping, we use $\boldsymbol{\phi}^{f,n,[q]}$ and $\boldsymbol{\phi}^{c,n,[q]}$ for solution in Ω^f and Ω^c, respectively. Assuming that the solution is known up to time t^{n-1}, the multirate time-stepping strategy for INSE is

1. For the predictor step ($q = 0$), compute the tentative velocity field $\acute{\mathbf{u}}$ using (2.28), and solve the linear Stokes problem in each subdomain:

   - Unsteady Stokes solve for the first sub-time-step in Ω^f:

$$\acute{\mathbf{u}}^{f,n-\frac{1}{2}} = -\sum_{j=1}^{k} \beta_j \mathbf{u}^{f,n-\frac{j+1}{2},[Q]} + \Delta t_f \sum_{j=1}^{k} \alpha_j (-\mathbf{u}\cdot\nabla\mathbf{u} + \mathbf{f})^{f,n-\frac{j+1}{2},[Q]},$$
$$\mathbf{S}\boldsymbol{\phi}^{f,n-\frac{1}{2},[0]} = \mathbf{r}^{f,n-\frac{1}{2},[0]},\ \mathbf{u}^{n-\frac{1}{2}}|_{\partial\Omega^f_D} = \mathbf{u}_b^{f,n-\frac{1}{2}},\ p^{n-\frac{1}{2}}|_{\partial\Omega^f_N} = 0, \mathbf{u}^{n-\frac{1}{2}}|_{\partial\Omega^f_I} = \mathcal{I}\Bigg(\sum_{j=1}^{m} \tilde{\alpha}_{1j}\, \mathbf{u}^{c,n-j,[Q]}\Bigg). \tag{6.2}$$

   - Unsteady Stokes solve for the second sub-time-step in Ω^f:

$$\begin{aligned}\acute{\mathbf{u}}^{f,n} &= -\sum_{j=1}^{k} \beta_j \mathbf{u}^{f,n-\frac{j}{2},[Q]} + \Delta t_f \sum_{j=1}^{k} \alpha_j (-\mathbf{u}\cdot\nabla\mathbf{u} + \mathbf{f})^{f,n-\frac{j}{2},[Q]},\\ \mathbf{S}\boldsymbol{\phi}^{f,n,[0]} &= \mathbf{r}^{f,n,[0]},\ \mathbf{u}^{n}|_{\partial\Omega^f_D} = \mathbf{u}_b^{f,n},\ p^{n}|_{\partial\Omega^f_N} = 0, \mathbf{u}^{n}|_{\partial\Omega^f_I} = \mathcal{I}\Bigg(\sum_{j=1}^{m} \tilde{\alpha}_{2j}\, \mathbf{u}^{c,n-j,[Q]}\Bigg).\end{aligned} \tag{6.3}$$

   - Unsteady Stokes solve for the only time-step in Ω^c:

$$\begin{aligned}\acute{\mathbf{u}}^{c,n} &= -\sum_{j=1}^{k} \beta_j \mathbf{u}^{c,n-j,[Q]} + \Delta t_c \sum_{j=1}^{k} \alpha_j (-\mathbf{u}\cdot\nabla\mathbf{u} + \mathbf{f})^{c,n-j,[Q]},\\ \mathbf{S}\boldsymbol{\phi}^{c,n,[0]} &= \mathbf{r}^{c,n,[0]},\ \mathbf{u}^{n}|_{\partial\Omega^c_D} = \mathbf{u}_b^{c,n},\ p^{n}|_{\partial\Omega^c_N} = 0, \mathbf{u}^{n}|_{\partial\Omega^c_I} = \mathcal{I}\Bigg(\sum_{j=1}^{m} \tilde{\alpha}_{j}\, \mathbf{u}^{f,n-\frac{j+1}{2},[Q]}\Bigg).\end{aligned} \tag{6.4}$$

In (6.2), we first compute the tentative velocity field as we did for single-rate time-stepping (3.5), and then solve for $\boldsymbol{\phi}^{f,n-\frac{1}{2},[0]}$ using the interdomain boundary data extrapolated from the solution at previous time-steps in Ω^c. The definition of $\mathbf{r}^{f,n-\frac{1}{2},[q]}$ is unchanged from (3.4) for single-rate time-stepping scheme, and we have introduced the notation $\tilde{\alpha}_{ij}$ to denote the coefficients that are used to extrapolate the interdomain boundary data at the ith sub-time-step for Ω^f. For example, $\tilde{\alpha}_{1j}$ corresponds to coefficients for extrapolating interdomain boundary data using $\mathbf{u}^{c,n-j,[Q]}$,$j = 1 \dots m$, to obtain the solution at the first sub-time-step for Ω^f, i.e., $\boldsymbol{\phi}^{f,n-\frac{1}{2},[0]}$.

After $\boldsymbol{\phi}^{f,n-\frac{1}{2},[0]}$ is computed in Ω^f, $\boldsymbol{\phi}^{f,n,[0]}$ can be computed using (6.3), which completes the predictor step for advancing the solution of INSE in Ω^f. Parallel to (6.2) and (6.3), (6.4) is used to solve for the solution at time t^n in Ω^c. We note that we have used $\tilde{\alpha}_j$ to represent the extrapolation weights for the only time-step for Ω^c.

From an implementation perspective, the extrapolation coefficients used in (6.2)-(6.4) are computed based on the time-levels (e.g., t^{n-1}, $t^{n-\frac{3}{2}}$, etc.) and the order of extrapolation m, using the routines described in [112].

Once the predictor step, $q = 0$, is complete, $q = 1 \dots Q$ corrector iterations can be done to improve the accuracy of the solution or stabilize the scheme.

2. For the corrector iterations ($q = 1 \dots Q$), compute the tentative velocity field $\acute{\mathbf{u}}$ again in Ω^f, and solve the linear Stokes problem in each subdomain:

   - Unsteady Stokes solve for the first sub-time-step in Ω^f:

$$\begin{aligned} \acute{\mathbf{u}}^{f,n-\frac{1}{2}} &= -\sum_{j=1}^{k} \beta_j \mathbf{u}^{f,n-\frac{j+1}{2},[Q]} + \Delta t_f \sum_{j=1}^{k} \alpha_j (-\mathbf{u}\cdot\nabla\mathbf{u} + \mathbf{f})^{f,n-\frac{j+1}{2},[Q]}, \\ \mathbf{S}\boldsymbol{\phi}^{f,n-\frac{1}{2},[q]} &= \mathbf{r}^{f,n-\frac{1}{2},[q]},\ \mathbf{u}^{n-\frac{1}{2}}|_{\partial\Omega_D^f} = \mathbf{u}_b^{f,n-\frac{1}{2}},\ p^{n-\frac{1}{2}}|_{\partial\Omega_N^f} = 0, \\ \mathbf{u}^{n-\frac{1}{2}}|_{\partial\Omega_I^f} &= \mathcal{I}\Big(\tilde{\gamma}_{11}\mathbf{u}^{c,n,[q-1]} + \tilde{\gamma}_{12}\mathbf{u}^{c,n-1,[Q]} + \tilde{\gamma}_{13}\mathbf{u}^{c,n-2,[Q]}\Big), \end{aligned} \tag{6.5}$$

   - Unsteady Stokes solve for the second sub-time-step in Ω^f:

$$\begin{aligned} \acute{\mathbf{u}}^{f,n} &= -\sum_{j=1}^{k} \beta_j \mathbf{u}^{f,n-\frac{j}{2},[Q]} + \Delta t_f \sum_{j=1}^{k} \alpha_j (-\mathbf{u}\cdot\nabla\mathbf{u} + \mathbf{f})^{f,n-\frac{j}{2},[Q]}, \\ \mathbf{S}\boldsymbol{\phi}^{f,n,[q]} &= \mathbf{r}^{f,n,[q]},\ \mathbf{u}^{n}|_{\partial\Omega_D^f} = \mathbf{u}_b^{f,n},\ p^{n}|_{\partial\Omega_N^f} = 0,\ \mathbf{u}^{n}|_{\partial\Omega_I^f} = \mathcal{I}\Big(\mathbf{u}^{c,n,[q-1]}\Big), \end{aligned} \tag{6.6}$$

   - Unsteady Stokes solve for the only time-step in Ω^c:

$$\mathbf{S}\boldsymbol{\phi}^{c,n,[q]} = \mathbf{r}^{c,n,[q]},\ \mathbf{u}^{n}|_{\partial\Omega_D^c} = \mathbf{u}_b^{c,n},\ p^{n}|_{\partial\Omega_N^c} = 0,\ \mathbf{u}^{n}|_{\partial\Omega_I^c} = \mathcal{I}\Big(\mathbf{u}^{f,n,[q-1]}\Big), \tag{6.7}$$

In (6.5), we compute $\boldsymbol{\phi}^{f,n-\frac{1}{2},[q]}$ using the interdomain boundary data obtained from Ω^c. To maintain high-order temporal accuracy, this boundary data is interpolated from the most recent solution at current time-step, $\boldsymbol{\phi}^{c,n,[q-1]}$, and the converged solution at previous time-steps, $\boldsymbol{\phi}^{c,n-1,[Q]}$ and $\boldsymbol{\phi}^{c,n-2,[Q]}$. The corresponding coefficients for this temporal interpolation are represented by $\tilde{\gamma}_{1j}$ in (6.5), and they are computed using the routines in [112]. In our framework, $\tilde{\gamma}_{ij}$ is computed assuming linear interpolation when $m = 1$ or 2, and quadratic interpolation when $m = 2$. This approach ensures that the desired temporal accuracy $\mathcal{O}(\Delta t^m)$ is maintained.

After (6.5) is used to compute $\boldsymbol{\phi}^{f,n-\frac{1}{2},[q]}$ in Ω^f, (6.6) is used to compute $\boldsymbol{\phi}^{f,n,[q]}$. Simultaneous to the corrector iterations for Ω^f, $\boldsymbol{\phi}^{c,n,[q]}$ is computed in Ω^c using (6.7).

We note that in the single-rate time-stepping scheme, the tentative velocity field ($\acute{\mathbf{u}}$) was computed only once for the Q corrector iterations (3.5). In contrast, we recompute the tentative velocity field in (6.5,6.6) at each corrector iteration for Ω^f, because the solution process spans η multiple sub-time-steps. Saving $\acute{\mathbf{u}}$ for each of the η sub-time-steps is not a scalable approach (e.g., $\eta = 100$ will require us to save tentative velocity field for 100 sub-time-steps), and, thus, we recompute $\acute{\mathbf{u}}$ at each corrector iteration of Ω^f.

Using (6.5)-(6.7), Q simultaneous Schwarz iterations can be used to determine the solution in Ω^f and Ω^c at time t^n.

Using the multirate time-stepping strategy described in this section, two overlapping ($S = 2$) subdomains, each with its time-step size, can be used to time-advance the solution of the INSE. We note that the current multirate time-stepping scheme does not apply to more than two overlapping subdomains with different time-step sizes, and we will look into this capability in the future.

6.3.1 Multirate time-stepping for arbitrary time-step ratio

Now that we have described our multirate time-stepping strategy for a time-step ratio of $\eta = 2$, we can extend this approach for an arbitrary integer η. We can anticipate from the discussion in the previous section that in this multirate scheme, the unsteady Stokes problem will be solved for η sub-time-steps in Ω^f and for only one time-step in Ω^c. The multirate time-stepping strategy for an arbitrary integer η is:

1. As earlier (6.2)-(6.5), compute the tentative velocity field and solve the linear Stokes problem in each subdomain for $q = 0$

   - Unsteady Stokes solve for the $i = 1 \dots \eta$ sub-time-steps of Ω^f:

$$\begin{aligned}
\acute{\mathbf{u}}^{f,n-1+\frac{i}{\eta}} &= -\sum_{j=1}^{k} \beta_j \mathbf{u}^{f,n-1+\frac{i-j}{\eta},[Q]} + \Delta t_f \sum_{j=1}^{k} \alpha_j (-\mathbf{u}\cdot\nabla\mathbf{u} + \mathbf{f})^{f,n-1+\frac{i-j}{\eta},[Q]}, \\
\mathbf{S}\boldsymbol{\phi}^{f,n-1+\frac{i}{\eta},[0]} &= \mathbf{r}^{f,n-1+\frac{i}{\eta},[0]},\ \mathbf{u}^{n-1+\frac{i}{\eta}}|_{\partial\Omega^f_D} = \mathbf{u}_b^{f,n-1+\frac{i}{\eta}},\ p^{n-1+\frac{i}{\eta}}|_{\partial\Omega^f_N} = 0, \qquad (6.8) \\
\mathbf{u}^{n-1+\frac{i}{\eta}}|_{\partial\Omega^f_I} &= \mathcal{I}\Bigg(\sum_{j=1}^{m} \tilde{\alpha}_{ij}\, \mathbf{u}^{c,n-j,[Q]} \Bigg),
\end{aligned}$$

- Unsteady Stokes solve for the only time-step of Ω^c:

$$\begin{aligned}
\acute{\mathbf{u}}^{c,n} &= -\sum_{j=1}^{k}\beta_j \mathbf{u}^{c,n-j,[Q]} + \Delta t_c \sum_{j=1}^{k}\alpha_j(-\mathbf{u}\cdot\nabla\mathbf{u}+\mathbf{f})^{c,n-j,[Q]},\\
\mathbf{S}\boldsymbol{\phi}^{c,n,[0]} &= \mathbf{r}^{c,n,[0]},\ \mathbf{u}^n|_{\partial\Omega^c_D} = \mathbf{u}_b^{c,n},\ p^n|_{\partial\Omega^c_N} = 0, \mathbf{u}^n|_{\partial\Omega^c_I} = \mathcal{I}\bigg(\sum_{j=1}^{m}\tilde{\alpha}_j\,\mathbf{u}^{f,n-1-\frac{j-1}{\eta},[Q]}\bigg),
\end{aligned} \tag{6.9}$$

In (6.8), we compute the sub-time-step solution for Ω^f, sequentially from $i = 1\ldots\eta$, and in (6.9), we compute the solution in Ω^c at time t^n.

2. Once the predictor step is complete, $q = 1\ldots Q$ corrector iterations are done as

- Unsteady Stokes solve for the $i = 1\ldots\eta$ sub-time-steps of Ω^f:

$$\begin{aligned}
\acute{\mathbf{u}}^{f,n-1+\frac{i}{\eta}} &= -\sum_{j=1}^{k}\beta_j \mathbf{u}^{f,n-1+\frac{i-j}{\eta},[Q]} + \Delta t_f \sum_{j=1}^{k}\alpha_j(-\mathbf{u}\cdot\nabla\mathbf{u}+\mathbf{f})^{f,n-1+\frac{i-j}{\eta},[Q]},\\
\mathbf{S}\boldsymbol{\phi}^{f,n-1+\frac{i}{\eta},[q]} &= \mathbf{r}^{f,n-1+\frac{i}{\eta},[q]},\ \mathbf{u}^{n-1+\frac{i}{\eta}}|_{\partial\Omega^f_D} = \mathbf{u}_b^{f,n-1+\frac{i}{\eta}},\ p^{n-1+\frac{i}{\eta}}|_{\partial\Omega^f_N} = 0,\\
\mathbf{u}^{n-1+\frac{i}{\eta}}|_{\partial\Omega^f_I} &= \mathcal{I}\bigg(\tilde{\gamma}_{i1}\mathbf{u}^{c,n,[q-1]} + \tilde{\gamma}_{i2}\mathbf{u}^{c,n-1,[Q]} + \tilde{\gamma}_{i3}\mathbf{u}^{c,n-2,[Q]}\bigg),
\end{aligned} \tag{6.10}$$

- Unsteady Stokes solve for the only time-step of Ω^c:

$$\mathbf{S}\boldsymbol{\phi}^{c,n,[q]} = \mathbf{r}^{c,n,[q]},\ \mathbf{u}^n|_{\partial\Omega^c_D} = \mathbf{u}_b^{c,n},\ p^n|_{\partial\Omega^c_N} = 0, \mathbf{u}^n|_{\partial\Omega^c_I} = \mathcal{I}\bigg(\mathbf{u}^{f,n,[q-1]}\bigg), \tag{6.11}$$

Using the high-order multirate time-stepping strategy in (6.8)-(6.11), the solution to the incompressible Navier-Stokes equations can be advanced in time for an arbitrary integer η. This approach has been used for analyzing the buoyant plume shown in Fig. 6.2, with $\eta = 5$ and 50, and will be discussed in detail in Section 9.7.

6.4 Stability of the multirate time-stepping scheme

In Chapter 5, we developed a framework for analyzing the stability of the single-rate predictor-corrector scheme for overlapping grids. Using the stability analysis framework, we had empirically shown how parameters such as the number of corrector iterations (Q) and the extrapolation order of interdomain boundary data (m) impact the stability of the high-order PC scheme. Since the stability analysis in Chapter 5 has helped us in qualitatively capturing behavior that we have observed in the Schwarz-SEM framework, we seek to extend that framework to the PC-based multirate time-stepping scheme.

In order to understand the stability properties of the multirate time-stepping scheme, we use the predictor-corrector scheme for solving the INSE (6.8-6.10) and combine it with our methodology of solving the unsteady diffusion equation using single-rate time-stepping method in (5.8) and (5.9).

Figure 6.7 shows an example of two overlapping grids with grid parameters $\tilde{N} = 7$ and $K = 4$. For the purposes of multirate time-stepping, we assume that $\Omega^f = \Omega^1$ and $\Omega^c = \Omega^2$. Assuming that the solution is know up to time t^{n-1}, the multirate time-stepping scheme for time-advancing the solution to the unsteady heat equation is:

1. Similar to (5.8), the predictor step $q = 0$ can be described in Ω^f and Ω^c as

$$\begin{aligned} i = 1 \dots \eta \rightarrow \underline{u}^{f,n-1+\frac{i}{\eta},[0]} &= -\sum_{l=1}^{k} \beta_l H_f^{-1} \underline{u}^{f,n-1+\frac{i-l}{\eta},[Q]} + \sum_{l=1}^{m} \tilde{\alpha}_{il} H_f^{-1} J_{fc} \underline{u}^{c,n-l,[Q]}, \\ \underline{u}^{c,n,[0]} &= -\sum_{l=1}^{k} \beta_l H_c^{-1} \underline{u}^{c,n-l,[Q]} + \sum_{l=1}^{m} \tilde{\alpha}_l H_c^{-1} J_{cf} \underline{u}^{f,n-1-\frac{l-1}{\eta},[Q]}, \end{aligned} \tag{6.12}$$

   where the unsteady heat equation is solved in Ω^f at η sub-time-steps and once in Ω^c, for obtaining the solution at time t^n. We note that the definition of H, J, β and $\tilde{\alpha}$, and the notation $\underline{u}$ is unchanged from Chapter 5

2. Similar to (5.9), $q = 1 \dots Q$ corrector iterations can be done as

$$\begin{aligned} i = 1 \dots \eta \rightarrow \ \underline{u}^{f,n-1+\frac{i}{\eta},[q]} &= -\sum_{l=1}^{k} \beta_l H_f^{-1} \underline{u}^{f,n-1+\frac{i-l}{\eta},[Q]} + \\ & H_f^{-1} J_{fc} \Big(\tilde{\gamma}_{i1} \underline{u}^{c,n,[q-1]} + \tilde{\gamma}_{i2} \underline{u}^{c,n-1,[Q]} + \tilde{\gamma}_{i3} \underline{u}^{c,n-2,[Q]} \Big), \\ \underline{u}^{c,n,[q]} &= -\sum_{l=1}^{k} \beta_l H_c^{-1} \underline{u}^{c,n-l,[Q]} + H_c^{-1} J_{cf} \underline{u}^{f,n,[q-1]}, \end{aligned} \tag{6.13}$$

   The extrapolation weights ($\tilde{\alpha}_{il}$ for Ω^f and $\tilde{\alpha}_l$ for Ω^c) are determined from the extrapolation order m, and similarly, the interpolation weights ($\tilde{\gamma}_{il}$) are chosen to maintain mth-order temporal accuracy. Both, the extrapolation and interpolation weights are generated using the routines described in [112].

6.4.1 Propagation matrix for stability analysis

In the preceding section, we have described the multirate time-stepping scheme (6.12-6.13) for time-advancing the solution of the unsteady heat equation. Here, we seek to cast this time-advancement scheme into a system of the form $\underline{z}^n = G\underline{z}^{n-1}$, in order to extend the stability analysis of the single-rate time-stepping scheme from Chapter 5.

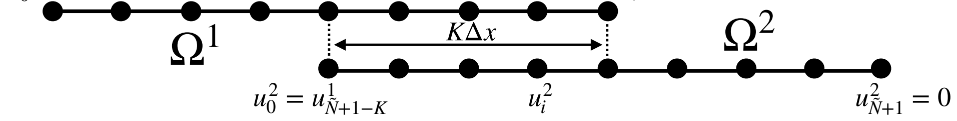


Figure 6.7: Overlapping grids ($\tilde{N} = 7, K = 4$)

In the multirate time-stepping scheme, the unsteady heat equation is solved in the subdomain with faster time scales (Ω^f) at multiple sub-time-steps in order to advance the solution from t^{n-1} to t^n. Consequently, casting the multirate time-stepping scheme into a system of the form $\underline{z}^n = G\underline{z}^{n-1}$ is not as straightforward as it was for the single-rate scheme. Here, we cast (6.12-6.13) into a system of the form $\underline{z}^n = G\underline{z}^{n-1}$ for $\eta = 2$, and this approach can then be readily extended for arbitrary η.

For notational purposes, we define

$$\underline{z}^n = [\underline{u}^{c,n^T}\ \underline{u}^{f,n^T}\ \underline{u}^{f,n-1/2^T}\ \underline{u}^{c,n-1^T}\ \underline{u}^{f,n-1^T}\ \underline{u}^{f,n-3/2^T}\ \underline{u}^{c,n-2^T}\ \underline{u}^{f,n-2^T}\ \underline{u}^{f,n-5/2^T}\ \underline{u}^{c,n-3^T}\ \underline{u}^{f,n-3^T}]^T, \quad (6.14)$$

where $\underline{u}^f$ and $\underline{u}^c$ are the solution vectors for subdomain Ω^f and Ω^c, respectively, and we use $\underline{z}^{n,[q]}$ to represent the vector with solutions at qth corrector iteration. (6.14) is different in comparison to the solution vector for the single-rate scheme (5.10), because Ω^f has solution vectors at sub-time-steps in $\underline{z}^n$.

In order to account for these sub-time-steps in Ω^f, we have to modify our methodology of building the predictor and corrector matrices. The predictor matrix will now be a product of η matrices, of which $\eta - 1$ matrices will correspond to the sub-time-steps of Ω^f, and 1 matrix will be for the last sub-time-step of Ω^f and the only step of Ω^c. For $\eta = 2$, the predictor matrix is $P = P_2P_1$ where P_1 generates the solution $\underline{u}^{f,n-\frac{1}{2},[0]}$ and P_2 generates the solution $\underline{u}^{c,n,[0]}$ and $\underline{u}^{f,n,[0]}$. The matrices P_2 and P_1 can be defined as

$$\begin{bmatrix} u^{f,n-\frac{1}{2},[0]} \\ u^{c,n-1,[0]} \\ u^{f,n-1,[0]} \\ u^{f,n-\frac{3}{2},[0]} \\ u^{c,n-2,[0]} \\ u^{f,n-2,[0]} \\ u^{f,n-\frac{5}{2},[0]} \\ u^{c,n-3,[0]} \\ u^{f,n-3,[0]} \end{bmatrix} = \underbrace{\begin{bmatrix} \tilde{\alpha}_{11}H_f^{-1}J_{fc} & -\beta_1H_f^{-1} & -\beta_2H_f^{-1} & \tilde{\alpha}_{12}H_f^{-1}J_{fc} & -\beta_3H_f^{-1} & 0 & \tilde{\alpha}_{13}H_f^{-1}J_{fc} & 0 & 0 & 0 & 0 \\ I_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & I_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & I_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & I_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & I_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_2 & 0 & 0 & 0 \end{bmatrix}}_{P_1} \underbrace{\begin{bmatrix} u^{c,n-1,[Q]} \\ u^{f,n-1,[Q]} \\ u^{f,n-\frac{3}{2},[Q]} \\ u^{c,n-2,[Q]} \\ u^{f,n-2,[Q]} \\ u^{f,n-\frac{5}{2},[Q]} \\ u^{c,n-3,[Q]} \\ u^{f,n-3,[Q]} \\ u^{f,n-\frac{7}{2},[Q]} \\ u^{c,n-4,[Q]} \\ u^{f,n-4,[Q]} \end{bmatrix}}_{\underline{z}^{n,[Q]}},$$

$$\underbrace{\begin{bmatrix} u^{c,n,[0]} \\ u^{f,n,[0]} \\ u^{f,n-\frac{1}{2},[0]} \\ u^{c,n-1,[0]} \\ u^{f,n-1,[0]} \\ u^{f,n-\frac{3}{2},[0]} \\ u^{c,n-2,[0]} \\ u^{f,n-2,[0]} \\ u^{f,n-\frac{5}{2},[0]} \\ u^{c,n-3,[0]} \\ u^{f,n-3,[0]} \end{bmatrix}}_{\underline{z}^{n,[0]}} = \underbrace{\begin{bmatrix} 0 & -\beta_1H_c^{-1} & \tilde{\alpha}_{21}H_c^{-1}J_{cf} & \tilde{\alpha}_{22}H_c^{-1}J_{cf} & -\beta_2H_c^{-1} & \tilde{\alpha}_{23}H_c^{-1}J_{cf} & 0 & -\beta_3H_c^{-1} & 0 \\ -\beta_1H_f^{-1} & \tilde{\alpha}_1H_f^{-1}J_{fc} & -\beta_2H_f^{-1} & -\beta_3H_f^{-1} & \tilde{\alpha}_2H_f^{-1}J_{fc} & 0 & 0 & \tilde{\alpha}_3H_f^{-1}J_{fc} & 0 \\ I_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & I_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & I_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & I_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & I_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_2 \end{bmatrix}}_{P_2} \begin{bmatrix} u^{f,n-\frac{1}{2},[0]} \\ u^{c,n-1,[0]} \\ u^{f,n-1,[0]} \\ u^{f,n-\frac{3}{2},[0]} \\ u^{c,n-2,[0]} \\ u^{f,n-2,[0]} \\ u^{f,n-\frac{5}{2},[0]} \\ u^{c,n-3,[0]} \\ u^{f,n-3,[0]} \end{bmatrix}.$$

Thus, the predictor step which time-advances the solution in Ω^f and Ω^c from t^{n-1} to t^n is

$$\underbrace{\begin{bmatrix} u^{c,n,[0]} \\ u^{f,n,[0]} \\ u^{f,n-\frac{1}{2},[0]} \\ u^{c,n-1,[0]} \\ u^{f,n-1,[0]} \\ u^{f,n-\frac{3}{2},[0]} \\ u^{c,n-2,[0]} \\ u^{f,n-2,[0]} \\ u^{f,n-\frac{5}{2},[0]} \\ u^{c,n-3,[0]} \\ u^{f,n-3,[0]} \end{bmatrix}}_{\underline{z}^{n,[0]}} = P_2 P_1 \underbrace{\begin{bmatrix} u^{c,n-1,[Q]} \\ u^{f,n-1,[Q]} \\ u^{f,n-\frac{3}{2},[Q]} \\ u^{c,n-2,[Q]} \\ u^{f,n-2,[Q]} \\ u^{f,n-\frac{5}{2},[Q]} \\ u^{c,n-3,[Q]} \\ u^{f,n-3,[Q]} \\ u^{f,n-\frac{7}{2},[Q]} \\ u^{c,n-4,[Q]} \\ u^{f,n-4,[Q]} \end{bmatrix}}_{\underline{z}^{n,[Q]}}.$$

Similarly, the system for corrector iterations is

$$\begin{bmatrix} u^{c,n,[q-1]} \\ u^{f,n,[q-1]} \\ u^{f,n-\frac{1}{2},[q]} \\ u^{c,n-1,[q]} \\ u^{f,n-1,[q]} \\ u^{f,n-\frac{3}{2},[q]} \\ u^{c,n-2,[q]} \\ u^{f,n-2,[q]} \\ u^{f,n-\frac{5}{2},[q]} \\ u^{c,n-3,[q]} \\ u^{f,n-3,[q]} \end{bmatrix} \underbrace{\begin{bmatrix} I_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & I_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \tilde{\gamma}_{11} H_f^{-1} J_{fc} & 0 & 0 & \tilde{\gamma}_{12} H_f^{-1} J_{fc} & -\beta_1 H_f^{-1} & -\beta_2 H_f^{-1} & \tilde{\gamma}_{13} H_f^{-1} J_{fc} & -\beta_3 H_f^{-1} & 0 & 0 & 0 \\ 0 & 0 & 0 & I_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & I_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & I_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_2 \end{bmatrix}}_{C_1} \underbrace{\begin{bmatrix} u^{c,n,[q-1]} \\ u^{f,n,[q-1]} \\ u^{f,n-\frac{1}{2},[q-1]} \\ u^{c,n-1,[q-1]} \\ u^{f,n-1,[q-1]} \\ u^{f,n-\frac{3}{2},[q-1]} \\ u^{c,n-2,[q-1]} \\ u^{f,n-2,[q-1]} \\ u^{f,n-\frac{5}{2},[q-1]} \\ u^{c,n-3,[q-1]} \\ u^{f,n-3,[q-1]} \end{bmatrix}}_{\underline{z}^{n,[q-1]}},$$

$$\underbrace{\begin{bmatrix} u^{c,n,[q]} \\ u^{f,n,[q]} \\ u^{f,n-\frac{1}{2},[q]} \\ u^{c,n-1,[q]} \\ u^{f,n-1,[q]} \\ u^{f,n-\frac{3}{2},[q]} \\ u^{c,n-2,[q]} \\ u^{f,n-2,[q]} \\ u^{f,n-\frac{5}{2},[q]} \\ u^{c,n-3,[q]} \\ u^{f,n-3,[q]} \end{bmatrix}}_{\underline{z}^{n,[q]}} \underbrace{\begin{bmatrix} 0 & H_c^{-1} J_{cf} & 0 & -\beta_1 H_c^{-1} & 0 & 0 & -\beta_2 H_c^{-1} & 0 & 0 & -\beta_3 H_c^{-1} & 0 \\ H_f^{-1} J_{fc} & 0 & -\beta_1 H_f^{-1} & 0 & -\beta_2 H_f^{-1} & -\beta_3 H_f^{-1} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & I_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & I_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & I_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_2 \end{bmatrix}}_{C_2} \begin{bmatrix} u^{c,n,[q-1]} \\ u^{f,n,[q-1]} \\ u^{f,n-\frac{1}{2},[q]} \\ u^{c,n-1,[q]} \\ u^{f,n-1,[q]} \\ u^{f,n-\frac{3}{2},[q]} \\ u^{c,n-2,[q]} \\ u^{f,n-2,[q]} \\ u^{f,n-\frac{5}{2},[q]} \\ u^{c,n-3,[q]} \\ u^{f,n-3,[q]} \end{bmatrix},$$

and thus, the corrector step is

$$\underbrace{\begin{bmatrix} u^{c,n,[q-1]} \\ u^{f,n,[q-1]} \\ u^{f,n-\frac{1}{2},[q-1]} \\ u^{c,n-1,[q-1]} \\ u^{f,n-1,[q-1]} \\ u^{f,n-\frac{3}{2},[q-1]} \\ u^{c,n-2,[q-1]} \\ u^{f,n-2,[q-1]} \\ u^{f,n-\frac{5}{2},[q-1]} \\ u^{c,n-3,[q-1]} \\ u^{f,n-3,[q-1]} \end{bmatrix}}_{\underline{z}^{n,[q]}} = C_2C_1 \underbrace{\begin{bmatrix} u^{c,n-1,[q-1]} \\ u^{f,n-1,[q-1]} \\ u^{f,n-\frac{3}{2},[q-1]} \\ u^{c,n-2,[q-1]} \\ u^{f,n-2,[q-1]} \\ u^{f,n-\frac{5}{2},[q-1]} \\ u^{c,n-3,[q-1]} \\ u^{f,n-3,[q-1]} \\ u^{f,n-\frac{7}{2},[q-1]} \\ u^{c,n-4,[q-1]} \\ u^{f,n-4,[q-1]} \end{bmatrix}}_{\underline{z}^{n,[q-1]}}$$

We note that the solution $u^{f,n-1/2,[q]}$ is effected via C_1, and $u^{f,n,[q]}$ and $u^{c,n,[q]}$ are determined using C_2. For Q corrector iterations, the time-advancement scheme is $\underline{z}^n = G\underline{z}^{n-1}$, where $G = C^QP$, $C = C_2C_1$, and $P = P_2P_1$, for $\eta = 2$.

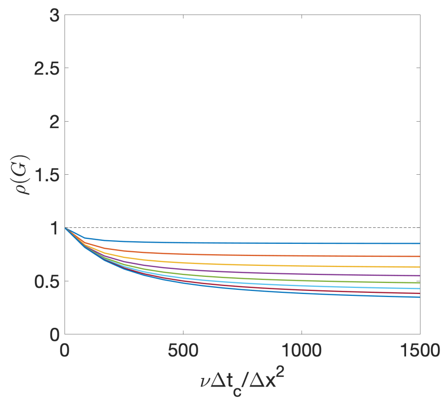
This methodology can readily be extended for arbitrary η where the predictor matrix is $P = P_\eta \dots P_1$ and the corrector matrix is $C = C_\eta \dots C_1$. For example, for $\eta = 3$, the predictor matrix P will be $P = P_3P_2P_1$, where P_1 determines $u^{f,n-1/3,[0]}$, P_2 determines $u^{f,n-2/3,[0]}$, and P_3 determines $u^{f,n,[0]}$ and $u^{c,n,[0]}$. Similarly the corrector matrix C will be $C = C_3C_2C_1$ where C_1 determines $u^{f,n-1/3,[q]}$, C_2 determines $u^{f,n-2/3,[q]}$, and C_3 determines $u^{f,n,[q]}$ and $u^{c,n,[q]}$.

Using this approach, we can determine the growth matrix $G = C^QP$ for any arbitrary η. The spectral radius of G can then be used to understand the stability behavior of the multirate time-stepping scheme for different parameters such as the extrapolation order of interdomain boundary data during the predictor step (m), grid resolution ($\tilde{N}$), overlap width (K), and the number of corrector iterations (Q).

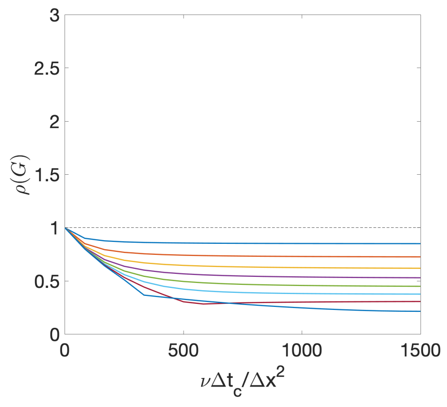
6.4.2 Stability results

In this section, we present the spectral radius ($\rho(G)$) version nondimensional time plots ($\nu\Delta t_c/\Delta x^2$) for different time-step size ratios. We start with $\eta = 2$, and then look at increasing values of η.

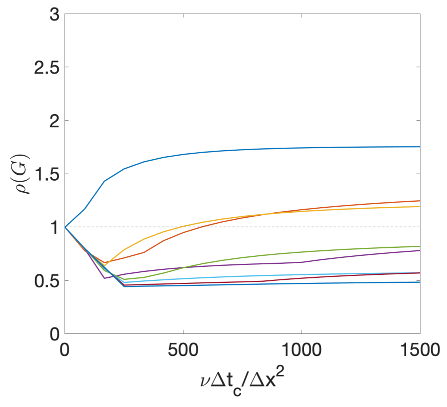
Figure 6.8 shows the stability plots for BDFk/EXTm scheme for different number of corrector iterations (Q) when $\eta = 2$, with grid parameters $\tilde{N} = 32$ and $K = 5$.



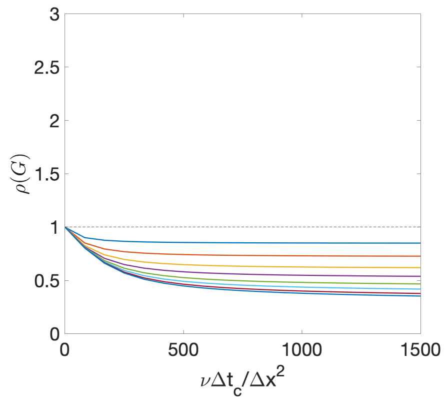
(a) $k = 1, m = 1$



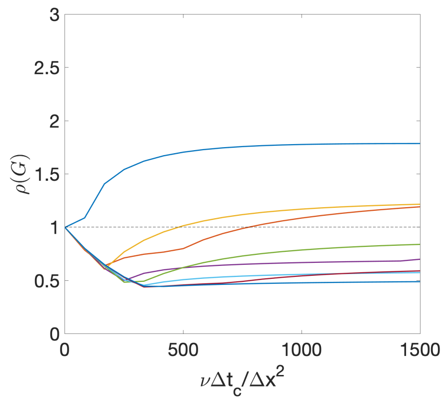
(b) $k = 2, m = 1$



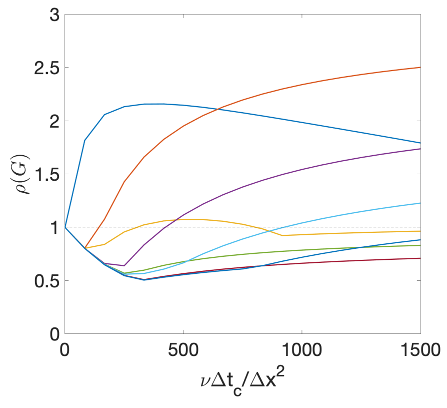
(c) $k = 2, m = 2$



(d) $k = 3, m = 1$



(e) $k = 3, m = 2$



(f) $k = 3, m = 3$

Figure 6.8: Spectral radius $\rho(G)$ versus nondimensional time $\frac{\nu\Delta t_c}{\Delta x^2}$ for different BDFk/EXTm schemes for $\eta = 2$ with $Q = 0 \ldots 7$, with grid parameters $\tilde{N} = 32$ and $K = 5$.

Based on the results shown here, we can conclude that using high-order extrapolation for interdomain boundary data (m) decreases the nondimensional time at which the spectral radius of G is greater than 1. This result is similar to the stability results for the single-rate time-stepping scheme (Fig. 5.4). We also see that using an odd-Q decreases the stability of the scheme, which is the opposite of what is observed for the single-rate time-stepping method ($\eta = 1$) where an even-Q is less stable than an odd-Q. In order to determine if the behavior of odd- and even-Q is universal for all η, we look at the results for $\eta > 2$. Additionally, since we are interested in third-order temporal accuracy, we will focus on the high-order scheme with $k = 3$ and $m = 3$.

Figure 6.9 shows the spectral radius versus nondimensional time plot for $\eta = 1, 2, 3, 4, 5$, and 10 with different Q. The plots in Fig. 6.9 use a semi-log scale for the x-axis to show the stability behavior of the multirate time-stepping scheme for a large range of nondimensional time-step size $\nu \Delta t_c / \Delta x^2$. We observe in Fig. 6.9 that for $\eta = 1$, odd-Q is more stable than even-Q, and for $\eta = 2$, even-Q is more stable than odd-Q. For $\eta \geq 3$, however, we observe that the odd-even pattern goes away for a large nondimensional time-step size ($\nu \Delta t_c / \Delta x^2 > 2 \times 10^4$). We also observe in Fig. 6.9 that the single-rate time-stepping scheme ($\eta = 1$) requires fewer corrector iterations to guarantee unconditional stability in comparison to the multirate time-stepping scheme. Here, unconditional stability means that $\rho(G) < 1$ irrespective of the nondimensional time-step size. Fig. 6.9 shows that for the grid parameters considered here ($\tilde{N} = 32$ and $K = 5$), $\eta = 1$ requires $Q = 3$ and $\eta \geq 2$ requires $Q = 6$ for unconditional stability, to solve the unsteady heat equation using overlapping grids with third-order temporal accuracy in the FD-based framework.

We notice similar behavior in stability if we increase the grid overlap by changing the grid parameter $K = 5$ to $K = 10$. Fig. 6.10(b), similar to Fig. 6.9(b), shows that even-Q is more stable than odd-Q for $\eta = 2$. For $\eta \geq 3$, we observed that the odd-even pattern in stability goes away, as evident in Fig. 6.9. We also see, as expected, that increasing the grid overlap makes the PC scheme more stable.

In the following section, we will show that the stability behavior that we have observed for large nondimensional time-step size in the 1D model problem, qualitatively captures the general stability behavior of the PC-based multirate time-stepping scheme for solving the INSE in the Schwarz-SEM framework. In future work, we will extend this analysis to make more rigorous predictions and establish theoretical bounds on the stability of the PC-based multirate time-stepping scheme. This will require us to understand how the nondimensional time-step size of the 1D model problem is related to the time-step size for the unsteady Stokes problem in the Schwarz-SEM framework. We will also investigate why the odd-even stability pattern manifests for $\eta = 1$ and 2, and not for $\eta \geq 3$.

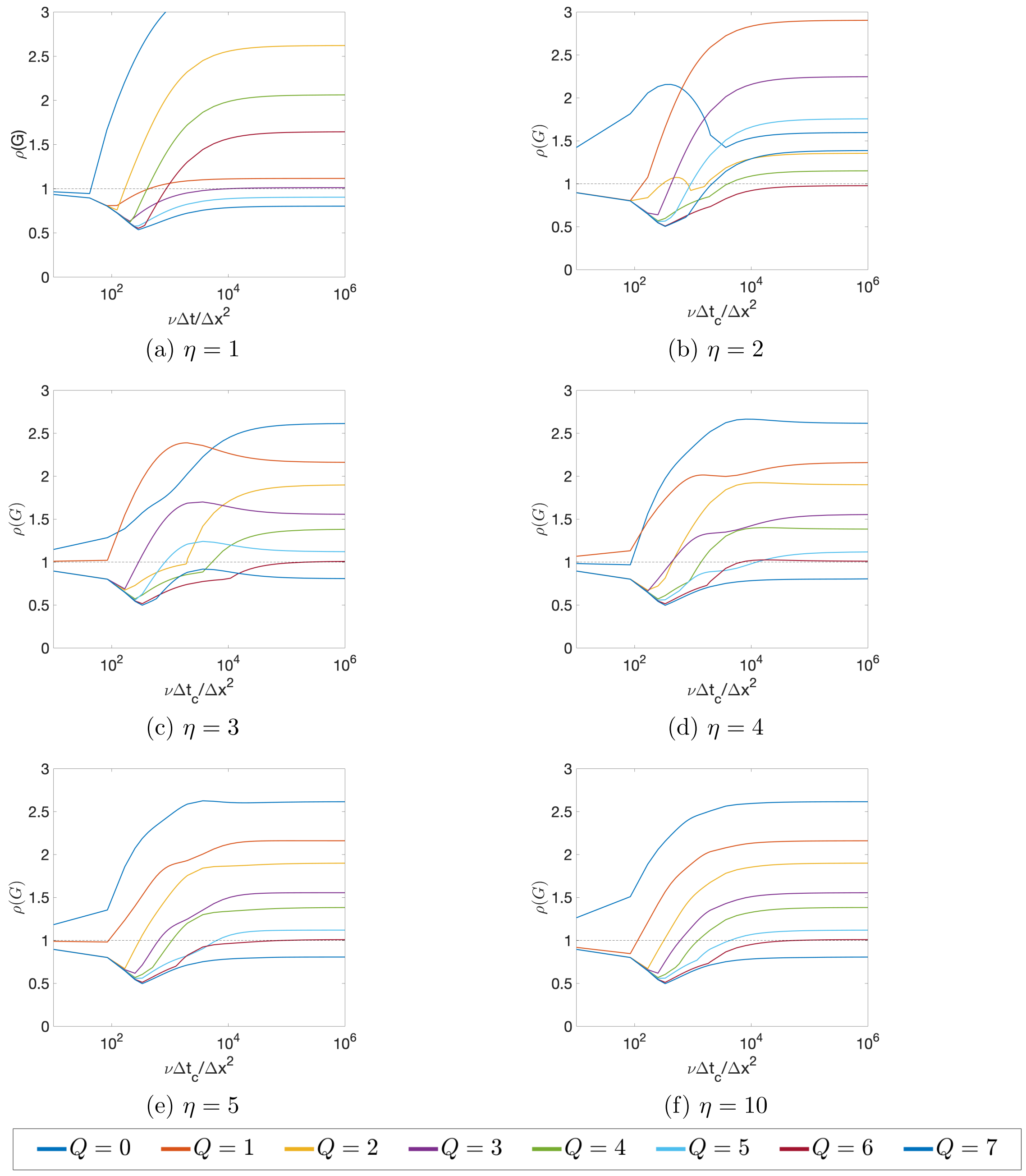


Figure 6.9: Spectral radius $\rho(G)$ versus nondimensional time $\frac{\nu \Delta t_c}{\Delta x^2}$ for different BDF3/EXT3 schemes with $Q = 0 \ldots 7$, with grid parameters $\tilde{N} = 32$ and $K = 5$.

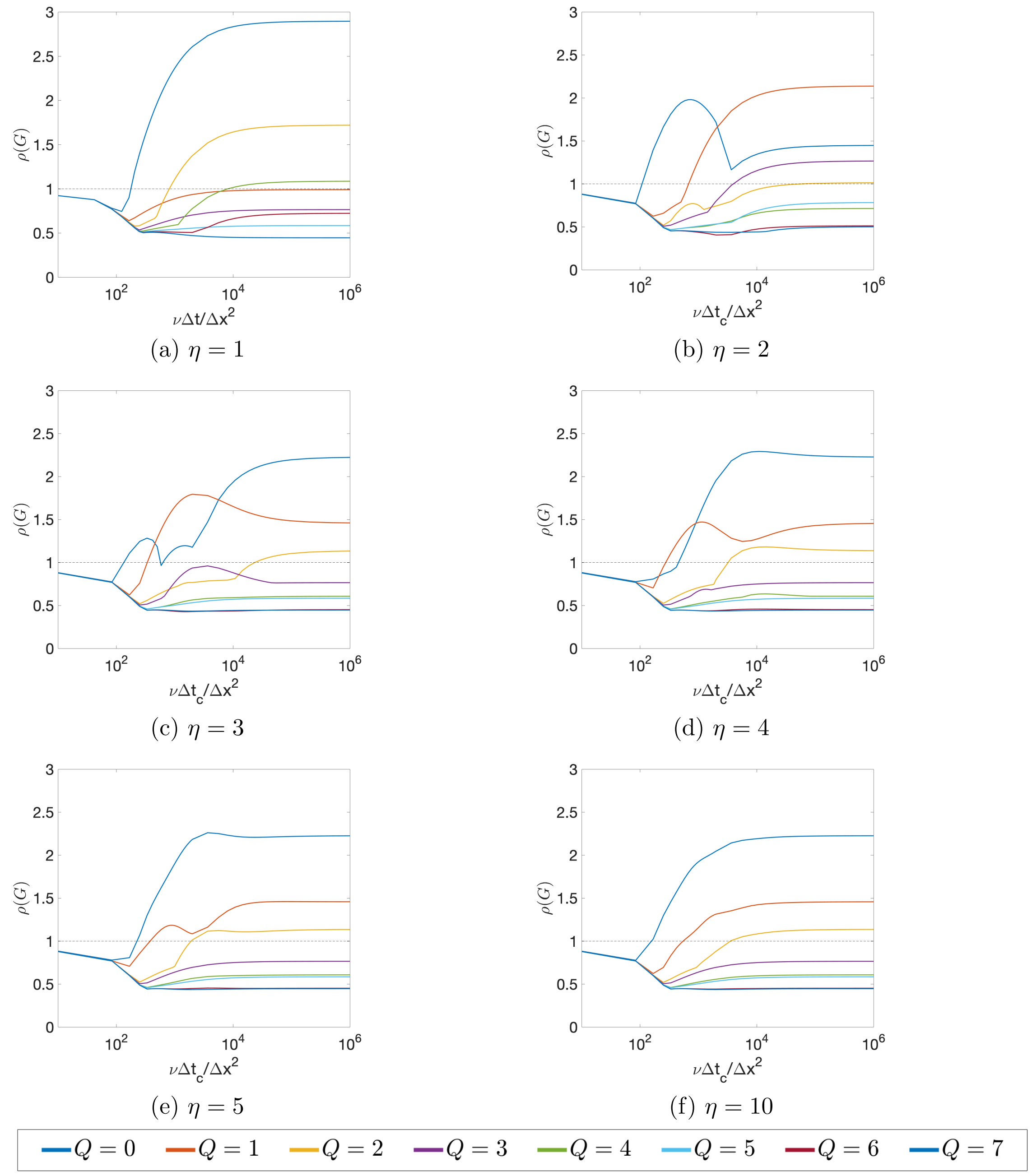


Figure 6.10: Spectral radius $\rho(G)$ versus nondimensional time $\frac{\nu\Delta t_c}{\Delta x^2}$ for different BDF3/EXT3 schemes with $Q = 0 \dots 7$, with grid parameters $\tilde{N} = 32$ and $K = 10$.

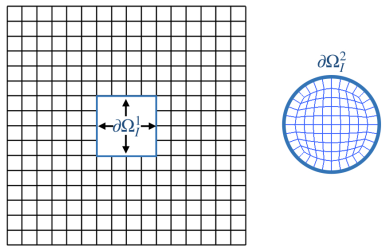


Figure 6.11: Overlapping spectral element meshes for discretizing the periodic domain $\Omega = [0, 2\pi]^2$ with (left) an outer mesh with 240 elements, and (right) a circular inner mesh with 96 elements.

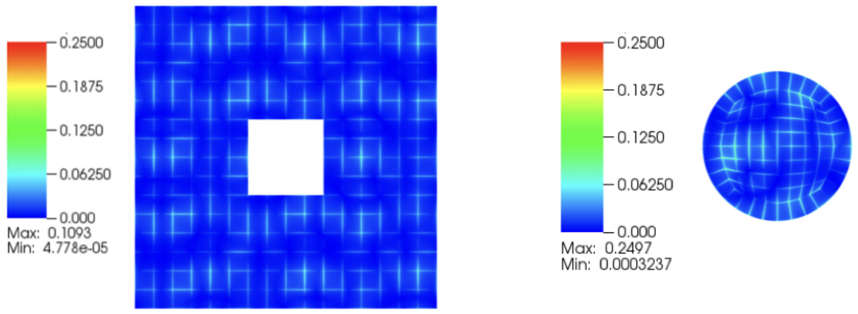


Figure 6.12: CFL comparison for the overlapping spectral element mesh for same time-step size. (left) Ω^c and (right) Ω^f. The CFL is minimum (4.78×10^{-5} in Ω^c and 3.22×10^{-4} in Ω^f) where the ratio of flow velocity to grid spacing is lowest in each subdomain.

6.5 Validation with Schwarz-SEM Framework

In order to validate the PC-based multirate time-stepping scheme and the stability analysis that we have done in this chapter, we revisit the Navier-Stokes eigenfunctions by Walsh that we had described in Chapter 2.

Figure 6.11 shows the $S = 2$ overlapping grids generated for the periodic domain ($\Omega = [0, 2\pi]^2$), with 240 elements for the background mesh and 96 elements for the circular mesh. In Fig. 6.12, we show a snapshot of the local grid CFL for these grids for a fixed time-step size. Due to the difference in the grid sizes, the maximum CFL number in the circular mesh (CFL=0.2497) is twice as much as the maximum CFL in the background mesh (CFL=0.1093). Thus, we can anticipate that in the context of multirate time-stepping, we will use a larger time-step size (Δt_c) for the background mesh and a smaller time-step size (Δt_f) for the circular mesh.

Here, we will first compare the stability behavior of the multirate time-stepping scheme in the Schwarz-

SEM framework with the results from the stability analysis of the 1D model problem. Next, we will empirically show that the PC-based multirate time-stepping scheme maintains the spatial and temporal convergence of the underlying SEM.

6.5.1 Stability Properties

To understand the stability properties of the PC-based multirate time-stepping scheme in the Schwarz-SEM framework, we set $\Delta t_c = 5 \times 10^{-3}$, $N = 7$, $k = 3$, and $m = 3$ with different time-step ratio ($\eta = 2 - 4$) and corrector iterations ($Q = 1 - 5$). Similar to the results in Fig. 5.1 for single-rate time-stepping ($\eta = 1$), we look at whether the multirate time-stepping scheme shows the same stability behavior in the Schwarz-SEM framework that we have observed in our FD-based analysis.

Preliminary results using the Schwarz-SEM framework show that we observe the stability behavior that we had expected from the analysis of the 1D model problem. Figure 6.13(a) shows that for $\eta = 2$, odd-Q is less stable than even-Q, as we had expected from the results in Fig. 6.9(b) and 6.10(b). Figure 6.13(b) and (c) shows that for $\eta = 3$ and 4, respectively, we do not observe the odd-even stability pattern in the Schwarz-SEM framework, which we had observed for large nondimensional time-step size in the FD-based framework (Fig. 6.9(c) and (d)). We note that we have observe this same behavior for similar numerical experiments that we have done in the Schwarz-SEM framework for different N.

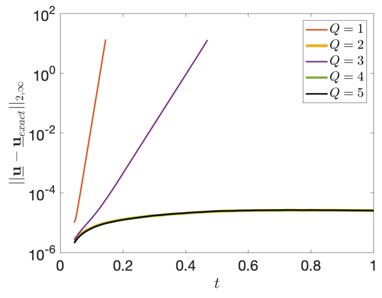
Based on numerical experiments that we have done in the Schwarz-SEM framework with the single-rate and multirate time-stepping PC scheme, we conclude that the asymptotic behavior (in terms of the nondimensional time-step size) that we observe for different Q and η in the 1D model problem, qualitatively captures the stability behavior that we observe in the Schwarz-SEM framework. In future work, we will look at methods that can allow us to make more rigorous predictions on the impact of Q and η on the stability properties of the multirate time-stepping scheme.

6.5.2 Spatial and Temporal Convergence

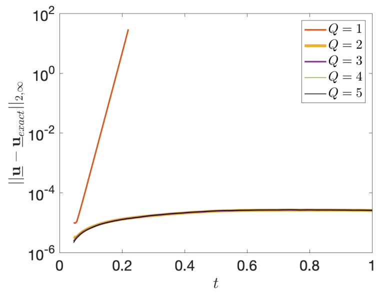
In this section, we empirically show that the PC-based multirate time-stepping scheme that we have described in this chapter maintains the spatial and temporal convergence of the underlying SEM solver.

Figure 6.14 compares the spatial and temporal convergence results from the single-rate time-stepping scheme and multirate time-stepping scheme (with $\eta = 2$ and 3). $Q = 2$ for $\eta = 2$ and $Q = 3$ for $\eta = 3$. For spatial convergence, we set $\Delta t_c = 10^{-4}$, and for temporal convergence, we set $N = 13$. The final error is computed at time $T_f = 1$.

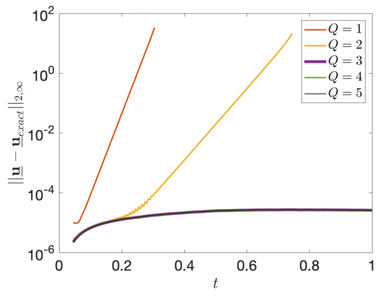
As we can see in Fig. 6.14, the multirate time-stepping scheme preserves the exponential convergence



(a) $\eta = 2$



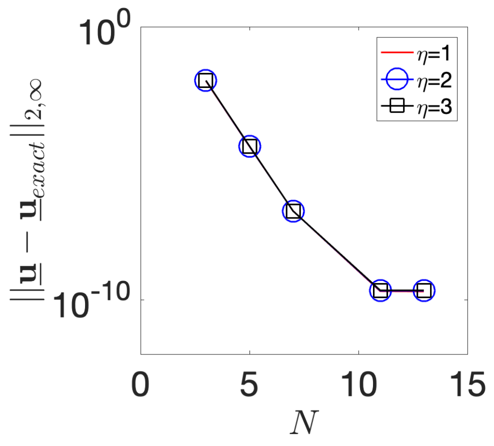
(a) $\eta = 3$



(b) $\eta = 4$

Figure 6.13: Error variation for the Navier-Stokes eigenfunctions test case from Chapter 3 with different Q for $\eta = 2 - 4$. We set $k = 3$, $m = 3$, $\Delta t_c = 5 \times 10^{-3}$, and $N = 7$ for the results presented here.

with N and third-order temporal convergence of the underlying SEM solver, and the error for $\eta = 2$ and 3 is of the same order as $\eta = 1$.



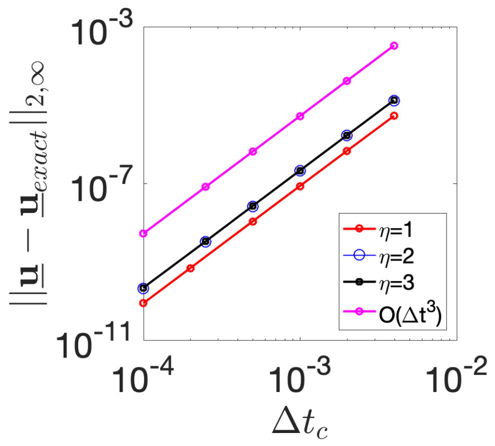


Figure 6.14: (left) Exponential convergence with polynomial order N and (right) third-order temporal convergence for $\eta = 1$, 2 and 3. The error is computed at time $T_f = 1$.

6.6 Summary

In this chapter, we have developed a multirate time-stepping scheme for the Schwarz-SEM framework. This novel scheme solves the INSE simultaneously in two overlapping subdomains using a PC scheme. We have also extended our stability analysis framework from the single-rate time-stepping scheme to determine how factors such as the time-step ratio and the interdomain boundary data extrapolation order impact the stability of the multirate time-stepping scheme. Preliminary results show that the FD-based stability analysis qualitatively describes the stability behavior for the multirate time-stepping scheme in the Schwarz-

SEM framework. Using numerical experiments, we have shown that the PC-based multirate time-stepping scheme preserves the spatial and temporal convergence of the underlying SEM solver. In future work, we will extend the multirate time-stepping scheme to more than two overlapping subdomains, and develop methods to better understand the stability properties of this PC scheme in the context of the Schwarz-SEM framework.

Chapter 7

Mesh Smoothing

In this chapter, we describe our mesh smoothing method for spectral element meshes. We start with a simple example to motivate the need for a robust mesh smoother and then discuss some of the critical aspects that have defined the design of our method. Next, we describe some of the essential tools needed to develop this smoother, followed by a discussion on our strategy for combining Laplacian smoothing and constrained optimization for improving the computational performance of spectral element meshes.

We note that the majority of the work that we present here was done before this dissertation. The work done as a part of this dissertation, however, was critical in making the mesh smoother robust and effective for fluid-thermal applications. Specifically, the contribution of this work is the development of a novel surface smoothing approach, which we describe in Section 7.2.3. We also describe our approach for monitoring the conditioning of the system for solving the pressure Poisson equation (Section 7.3) and empirically show that mesh smoothing improves the conditioning of this system. The mesh smoother presented in this chapter has been published in Journal of Scientific Computing [73].

7.1 Motivation

While overlapping Schwarz based methods generally yield relatively simple meshes in individual subdomains, mesh smoothing is of value in both, the mono- and multidomain cases. Mesh smoothing can significantly improve the computational efficiency of numerical simulations. For the INSE, this includes improving elements that constrain the performance of pressure Poisson solver and limit the maximum time-step size due to the CFL constraint.

Figure 7.1(a) shows an example of the mesh that was generated for studying buoyant plumes in a stably stratified environment, using a typical block decomposition approach. In this example, the resolution needed to capture the turbulence in the pipe at the bottom of the tank, leads to a thin band of elements in the domain (for the inner mesh shown in blue). These high aspect-ratio elements are degrading for the computational performance of the SEM solver and limit the maximum allowable time-step size due to the CFL number

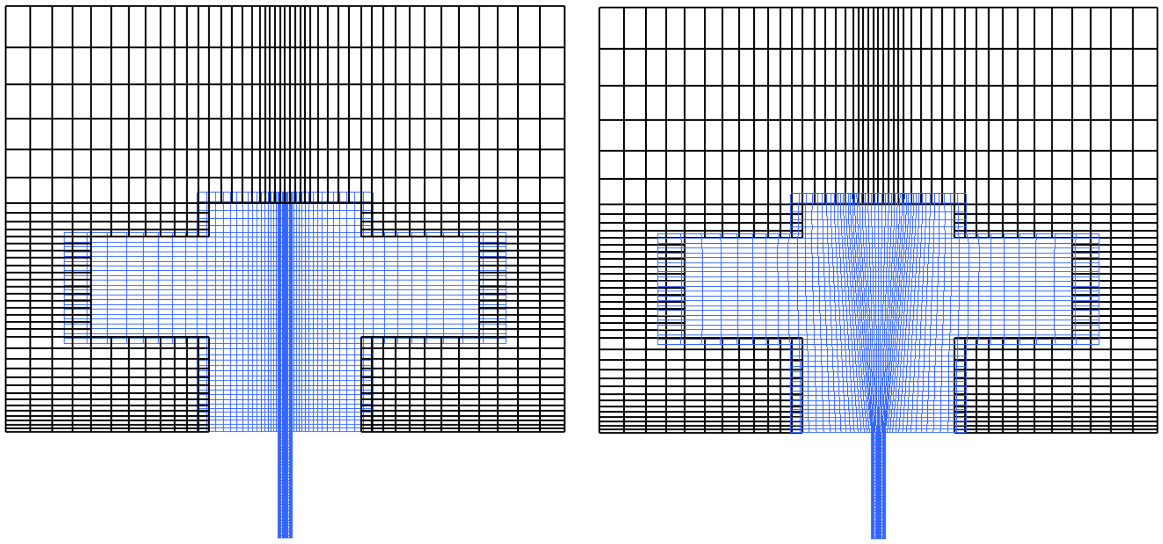

(a) Original mesh (b) Smoothed mesh

Figure 7.1: Comparison of the slice view of the original and smoothed mesh generated for studying buoyant plumes in a stably stratified environment using the Schwarz-SEM framework.

(6.1). Numerical experiments have shown that smoothing the inner mesh allows use of a 3X larger time-step size as compared to the original mesh. The smoothed mesh is shown in Fig. 7.1(b), where we can see that the mesh is free of high aspect-ratio elements away from the pipe at the bottom of the tank. We also note that in this example, surface smoothing is enabled, which allows grid points on the subdomain boundary to move along the boundary. If surface smoothing had not been used here, the mesh smoother would not have been able to eliminate high aspect ratio elements near the subdomain boundaries. The outer mesh is not smoothed in this example because the flow is laminar in the far-field (Fig. 6.2) and consequently, the computational performance of the calculation is limited by the quality of the inner mesh.

Our goal here is to describe the mesh smoother that we have developed [73], keeping in mind certain key aspects of meshes generated by automated mesh generation tools. High-order meshes are typically constructed by generating a linear mesh using a block decomposition approach [74], followed by projection onto the actual high-order geometry surface [75, 76, 77, 78]. Due to this approach, most elements in a mesh are linear or quadratic, except for the surface conforming high-order elements. Additionally, meshes generated for simulating incompressible flow have boundary layer resolving elements to capture near-wall physics, which due to their anisotropy, can lead to high aspect-ratio elements in the far-field. These high aspect-ratio elements adversely impact the condition number of the system of the pressure Poisson solver and the CFL of the grid. Lastly, since the quality of the surface mesh constrains mesh quality in the interior of a domain, it is essential to be able to smooth surface mesh in order to maximize the quality of the given mesh. Mesh improvement methods that support surface smoothing, typically rely on a CAD or a parametric representation of the original geometry in order to smooth the surface mesh. This approach is not possible in the FEM- or SEM-based framework where only a discrete representation of the original geometry is available through the original mesh. These various aspects have helped us develop a robust mesh smoothing method.

7.2 Key Tools for Mesh Smoothing

In this section, we describe a set of tools that are critical to the efficiency and impact of our smoothing algorithm.

7.2.1 Smoothing on lower polynomial order

As mentioned earlier, high-order meshes are typically generated by meshing the domain with low-order elements and then projecting the surface GLL points to the actual high-order description of the bounded domain. Thus, surface elements, of order $G = N$, eliminate geometrical approximation errors, where G represents the order of the mesh and N represents the order of the SEM. Consequently, the interior elements of most meshes are typically low-order.

Relaxation of the constraint $G = N$ to $G = 2$, allows us to smooth the mesh with each element cast in second-order representation since quadratic elements have a GLL point at the middle of every edge. Furthermore, we note here that the exponential convergence of the SEM is not dependent on the order of the interior elements, but on the spatial resolution inside the domain[1]. Consequently, smoothing the mesh at $G = 2$ does not impact the convergence of the SEM as long as regions of interest have adequate resolution and the geometry (i.e., surface elements) is defined with adequate precision. A notable exception where $G = N$ almost everywhere includes high order Lagrangian formulations [113].

Using $G = 2$, the first step in our smoothing process is to split each spectral element into 2^d micro-elements, which are represented as bilinear (in 2D) or trilinear (3D) elements (i.e., G=1). Our smoothing algorithms are applied to this new mesh comprising $E_m = 2^d E$ micro-elements. Figure 7.2 shows an example of a spectral element mesh with 16 spectral elements, and Fig. 7.2(b) shows the mesh split into 64 linear micro-elements.

When performing surface smoothing on this mesh with micro-elements, we project all surface points onto the surface of the original spectral element mesh after each smoothing sweep. At the very end of the smoothing process, all micro-elements are recombined into their parent spectral elements at $G = 2$, and the

[1]In Appendix A, we empirically demonstrate that the spatial convergence of the solution in SEM depends on the spatial resolution and not on the mesh quality.

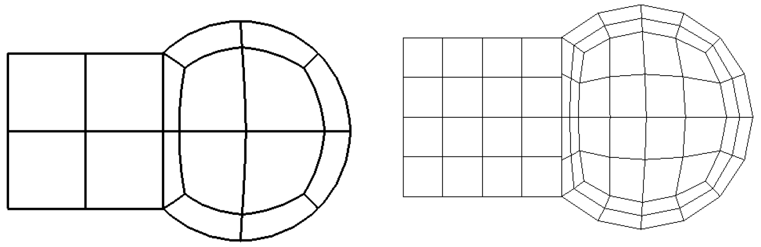


Figure 7.2: (left to right) (a) A spectral element mesh with $E = 16$ elements (b) split into $E_m = 64$ elements with $G = 1$.

mesh is interpolated from $G = 2$ to the SEM order $G = N$. Finally, all GLL points (at SEM order N) are projected onto the original geometry for elements on the domain boundary. This final projection step is accompanied by a blending process that smoothly distributes the surface displacement into the domain interior. Blending is done either on an element-by-element basis using transfinite interpolation [84,114] or by solving a global Laplace equation, with the latter approach preferred when the mesh has thin boundary-layer elements.

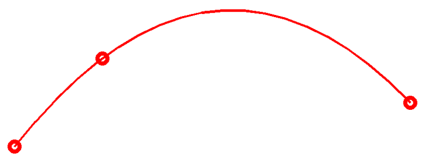
When recombining $(E_m, G = 1)$ elements into the $(E, G = 2)$ mesh it is essential to ensure that the midside nodes[2] are interpolated to the middle of the edge after recombining the mesh, to avoid inverted elements. This is a common pitfall since valid quad elements do not necessarily combine to form one valid spectral element. The interpolation of the midside node is done by translating it parallel to the line joining the two endpoints of the edge, and moving it to the center of this line, as shown below in Figure 7.3. An alternative to this approach is to fit a second-order curve through the 3 nodes and move the midside node to the middle of the curved edge. While generally effective, numerical experiments have shown that this strategy can lead to spectral elements with negative Jacobians.

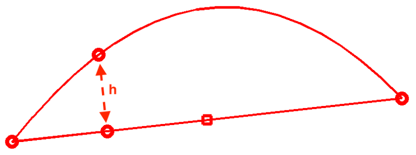
We note that for notational purposes, the use of the local form (e.g., $\underline{u}$) and global form (e.g., $\underline{u}_g$) in this chapter will refer to the functions and operators (e.g., VV^T) corresponding to the mesh with micro-elements (and not the spectral elements that are a starting point for the mesh smoothing process).

7.2.2 Weight function for boundary layer preservation

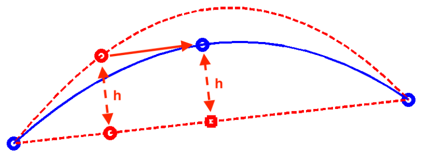
Boundary layer preservation becomes important when simulating high Reynolds number flows in computational fluid dynamics applications (the principal target of our effort), because the solution requires adequate boundary-layer resolution to capture important near-wall physics. Thus, mesh improvement strategies must ensure that the size of the boundary layer resolving elements is maintained in the wall-normal direction

[2]We use the term "midside node" to refer to the grid point that is at the center of a spectral element's edge.


(a) Edge with midside node not at the middle of the edge



(b) h, distance of the midside node from the line segment joining the two end points of the edge, and the midpoint (☐) of that line segment



(c) Modified midside node to the middle of the edge while maintaining the distance between the midside node and the line joining the two end points of the edge

Figure 7.3: Interpolation of midside node post-smoothing to the middle of the edge

during the smoothing process.

The issue of boundary layer preservation has not been sufficiently addressed in the existing literature. Canann et al. [63] proposed a method to preserve boundary-layer resolution by constraining the distance of the farthest boundary-layer node. While this method is effective, it requires the user to identify boundary-layer points that are farthest from the wall, which is not straightforward to implement for complicated meshes. Here, we adopt a different approach of using a weight function based on the minimum distance of all the grid points from the closest wall, to restrict their motion during and after the smoothing process.

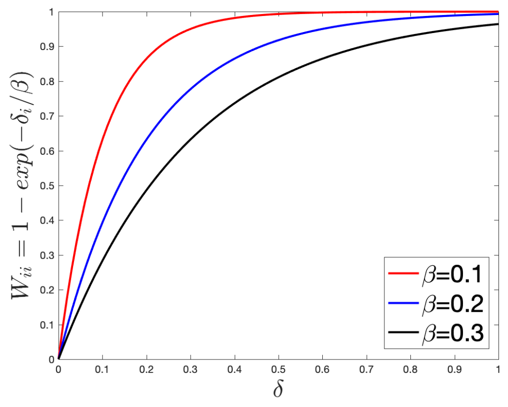
The weight function (W) is a diagonal matrix with weights corresponding to each grid point in the mesh, which is used to preserve the shape of the boundary layer elements during the smoothing process. This weight function can either be user-specified or is set by default as a function of the distance to the nearest wall for each grid point. A typical approach to determine the weight function is first to use the distance function generator described in Section 3.4 to determine the distance of all grid points in the mesh from the closest boundary surface. Once the distance function (δ) is determined, it is normalized by some chosen characteristic length scale. In the absence of any other scale information, we set the characteristic length scale to the maximum value of $\underline{\delta}$ over the grid points of micro-elements. Using the normalized distance function, the diagonal matrix with weight function (W_{ii}) is determined using the distance δ_i for each grid point i as

$$W_{ii} = 1 - e^{-\delta_i/\beta}, \tag{7.1}$$

$$\text{or } W_{ii} = 0.5(tanh(\alpha(\delta_i - \beta)) + 1) \tag{7.2}$$

or another function of user's choice. In (7.1) and (7.2), α and β are parameters that determine the shape of the weight function.

Figure 7.4 shows how the weight function given in (7.1) and (7.2) varies for different parameters α and β. We use this weight function to scale the displacements of the grid points, determined by the mesh smoother, at each iteration. This approach has an effect of reducing the movement of the grid points near the wall. We also use the weight function W at the end of the smoothing process, as shown in Algorithm 3 and illustrated in Figure 7.5, to generate the final mesh as a weighted combination of the original mesh and the smooth mesh. Our use of the weight function to restore the boundary layer resolution as the final step of the mesh smoothing process, as shown in Fig. 7.5(c), will be clear from the discussion in the next section.



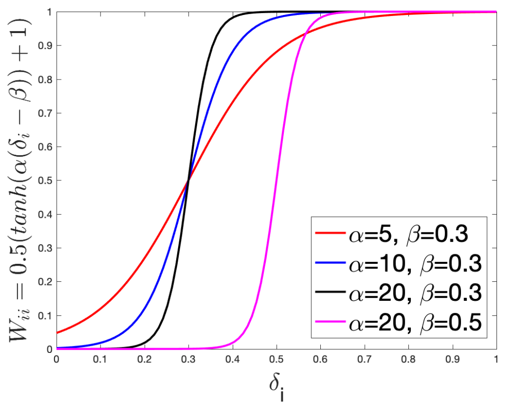
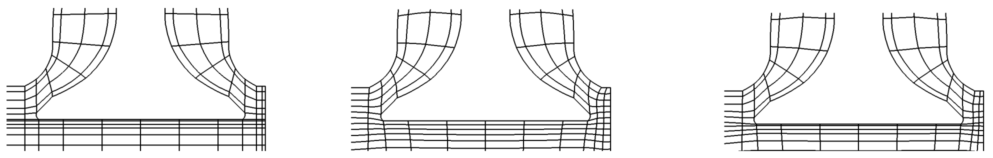


Figure 7.4: (left to right) (a) Exponential- (7.1) and (b) $tanh$-based (7.2) weight function.



(a) Original Mesh (b) Smoothed mesh (c) Boundary layer restored

Figure 7.5: Smoothing across domain decomposition zones and boundary layer preservation.

7.2.3 Surface smoothing

The quality of the surface mesh constraints the mesh quality interior to a domain. It is important, therefore, to be able to smooth the surface mesh without distorting the domain-boundary geometry. In cases where the geometry is provided analytically or through a CAD model, one can smooth the elements on the surface of the domain and project them on to the geometry after smoothing. In other cases, one can leverage the originating high-order spectral element description as a surrogate for the true geometry, under the assumption that (2.6) provides a satisfactory description of the domain surface. In our framework, we use this second scheme to project the surface grid points to the original geometry after each smoothing iteration. The essential idea behind the surface smoothing is to allow surface points to move during the volumetric smoothing process, which will be described in the next section, followed by projection onto the original geometry.

The tools that are critical to the surface smoothing procedure include a fast and robust interpolation routine and a readily computed distance function $\delta(\mathbf{x})$ that indicates the minimum distance between any point $\mathbf{x}^* \in \Omega$ and the domain boundary $\partial\Omega$. Off-grid interpolation is effected via *findptslib*, the high-order interpolation library that we have described in Section 3.2.1, and the distance function $\delta(\mathbf{x})$ is generated using the approach described in Section 3.4. We also define a diagonal mask matrix, M, before starting the smoothing process. The entries of this mask matrix are 1 or 0 for each grid point depending on whether the user wants it to move or not during the smoothing process. Where surface smoothing is desired, the mask is set to 1 such that grid points are allowed to move during each substep of the volumetric mesh optimization step.

Surface smoothing is implemented as part of the volumetric mesh smoothing process. If a surface node i, with physical-space coordinates $\mathbf{x}_i$ is moved to $\mathbf{x}_i^*$ by the smoother, the constraint for surface smoothing is that the displacement $\mathbf{x}_i^* - \mathbf{x}_i$ should lie in the tangent plane passing through the original surface location, $\mathbf{x}_i$. (The tangent-plane constraint is readily implemented using $\nabla\delta$.) For planar surfaces, this constraint suffices to keep surface grid points on the domain boundary. For curved surfaces, the constrained motion will yield an $O(||\mathbf{x}_i^* - \mathbf{x}_i||^2)$ displacement in the normal direction. Consequently, all displaced surface vertices are reprojected to the original spectral-element domain boundary after each smoothing substep. This surface projection is done by first using *findpts* to determine if the surface grid points have moved interior or exterior to the original spectral element mesh. If a surface node $\mathbf{x}_i^*$ is outside the domain, the node is mapped to the nearest surface value, $\mathbf{x}_i'$, which is automatically returned by *findpts*. If $\mathbf{x}^*$ is interior to Ω, we evaluate $[\delta(\mathbf{x}_i^*), \nabla\delta(\mathbf{x}_i^*)]$ using *findpts_eval* and set

$$\mathbf{x}_i' \quad = \quad \mathbf{x}_i^* \; - \; \nabla\delta(\mathbf{x}_i^*)\,\delta(\mathbf{x}_i^*), \tag{7.3}$$

where $\mathbf{x}'_i$ is the position of the surface node after projection on to the original geometry.

At the end of the smoothing process, when the micro-elements are combined into the coarse mesh ($E, G = 2$), which is then interpolated back to the original SEM order ($E, G = N$), all surfaces GLL points are again projected onto the original mesh. This approach ensures that no geometrical approximations are introduced for high-order meshes. Additionally, as discussed earlier, as long as there is adequate resolution inside the domain, the exponential convergence of the SEM is maintained. Figure 7.5 shows a section of mesh generated for an internal combustion engine. As we can see, surface smoothing on the cylinder helps eliminate CFL constraining elements between the valve stem and the cylinder wall. Figure 7.6 shows the effect of smoothing the surface of a mesh for a turbine blade.

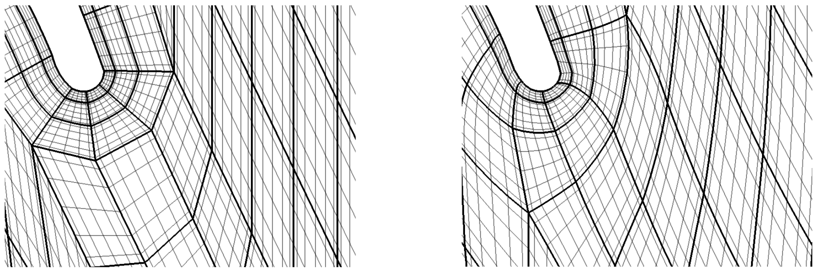


Figure 7.6: Comparison of a mesh ($N = 5$) before and after surface smoothing for a turbine blade, with GLL points

7.3 Pressure Solve and Conditioning of the Resulting System

As one of our key drivers for the smoother is computational efficiency, we briefly introduce the pressure Poisson equation that is typically a bottleneck for incompressible Navier-Stokes due to its ill-conditioning for high-resolution calculations. The discretization of the INSE for the SEM formulation has been presented in Chapter 2 in (2.33-2.34). Since the characteristic propagation speed of pressure perturbations is infinite in incompressible flows, the pressure operator is the leading contributor to the stiffness of the system resulting in unsteady flows. Following [115,116], the pressure is calculated using generalized minimal residual (GMRES) method to solve a preconditioned system $MA\underline{\mathbf{x}} = M\underline{\mathbf{b}}$. Here M is the preconditioner, and MA has properties similar to a Poisson operator [117, 118], which adds to the ill-conditioning of the overall pressure system. This preconditioner is effectively a three-level p-multigrid strategy [119, 120], and has proven to be highly effective for systems with $E > 10^6$ on over a million processors [121].

For a given preconditioned system of equations $MA\underline{\mathbf{x}} = M\underline{\mathbf{b}}$, the convergence of the solution using a Krylov-subspace projection method depends on the iterative condition number, which is the ratio of the maximum to minimum eigenvalues ($\kappa_{iter} = \frac{\lambda_{max}}{\lambda_{min}}$) of the matrix MA [122]. This convergence property holds for our problem since we solve for pressure using GMRES. For a mesh with E spectral elements, the size of this system of equations is $n_p \times n_p$, where $n_p = E(N-1)^D$, which can be very large for even calculations of modest size. Consequently, the matrix MA is never explicitly formed, and as a result, the condition number of MA cannot be monitored during the solution to understand how the quality of mesh affects this system. However, since GMRES is used, the upper Hessenberg matrix that is generated during the iterative process can be used because its extreme eigenvalues (also know as Ritz values) [123] are a good estimate of the extreme eigenvalues of MA.

We hypothesize that for our calculation of pressure correction, mesh smoothing decreases κ_{iter} which results in decreasing the number of iterations for the pressure-solve step, with λ_{max} and λ_{min} determined from the upper Hessenberg matrix generated during the GMRES iterations. We present the κ_{iter} for various cases that we have analyzed using this approach, in Section 9.9, to understand how the smoothness of mesh translates into better computational performance in terms of N_{iter}.

7.4 Mesh smoothing strategy

In this section, we describe our mesh improvement methodology that is based on a combination of Laplacian smoothing and constrained optimization and can be used in parallel on P MPI ranks. The rationale behind our approach of mesh improvement is that Laplacian smoothing is computationally cheap, but does not guarantee that that the mesh will stay valid during the smoothing process. Combining the Laplacian smoother with a constrained optimization approach allows us to ensure that the mesh stays valid everywhere in the domain while improving the quality of the performance degrading elements.

7.4.1 Laplacian smoothing

Mesh smoothing is inherently nonlocal and therefore requires interprocessor communication in a parallel implementation. Laplacian smoothing is typically performed by updating the position of each grid point as the average of all the grid points that it is connected to. In our framework, the geometry is stored in the local form, as explained in Section 2.1.2. Thus, updating the position of each element vertex as a function of other elements that it is shared with, is not an efficient approach for mesh smoothing in parallel. As a result, we smooth each element individually and then use the $C^{-1}VV^T$ operator (Section 2.1.3) to average the displacement between shared grid points. The use of the $C^{-1}VV^T$ operator ensures that we maintain the geometric continuity of the mesh.

Since we split each element in a mesh into micro-elements, we use the notation $\underline{\mathbf{x}}^e$ to represent the vector of physical-space coordinates of the vertices of each micro-element e. Similar to the spectral elements, each micro-element is also mapped to the reference-space element $\hat{\Omega}$, such that $\underline{\mathbf{x}}^e = \underline{\mathbf{x}}(\xi, \eta, \zeta) = \underline{\mathbf{x}}(\pm 1, \pm 1, \pm 1)$. Our approach to Laplacian smoothing is to shrink each element in the mesh by a user-specified factor (s_f) in the reference-space. We typically use $s_f = 0.99$ based on our experience with smoothing meshes. The benefit of shrinking elements in the reference-space is that it is a convex operation. Following the shrinking step, the displacement of the grid points is averaged using $C^{-1}VV^T$ operator. This strategy has an overall effect of making the mesh uniformly sized. Additionally, the edges/faces on the surface are shrunk tangent to the surface. This can be achieved by simply setting $s_f = 1$ for the edge/face which is on the surface. Figure 7.7 demonstrates a simple example of one iteration of the shrinking process.

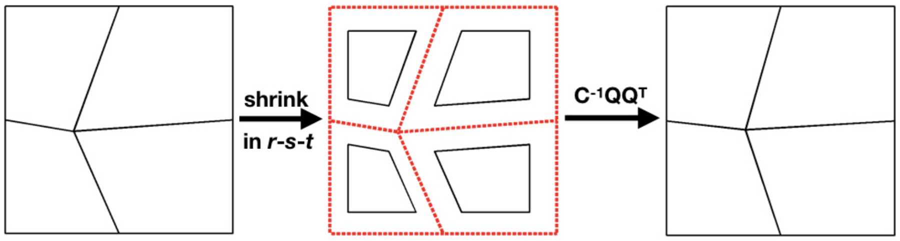


Figure 7.7: Laplacian smoothing done by shrinking each element followed by the $C^{-1}VV^T$ operator on the nodal displacements.

Algorithm 1: Laplacian smoothing algorithm

```
1: for i:=1 to $N_{lap}$ do
2:     for e:=1 to $E_m$ do
3:         $\underline{\mathbf{\Delta x}}^e = \underline{\mathbf{x}}^e(\pm s_f, \pm s_f, \pm s_f) - \underline{\mathbf{x}}^e(\pm 1, \pm 1, \pm 1)$
4:     $\underline{\mathbf{\Delta x}} = C^{-1}VV^T\underline{\mathbf{\Delta x}}$
5:     $\underline{\mathbf{\Delta x}} = M\underline{\mathbf{\Delta x}}$
6:     $\underline{\mathbf{\Delta x}} = W\underline{\mathbf{\Delta x}}$
7:     update $\underline{\mathbf{\Delta x}}$ by surface smoothing
8:     $\underline{\mathbf{x}} = \underline{\mathbf{x}} + \underline{\mathbf{\Delta x}}$
```

Algorithm 1 describes the Laplacian smoothing algorithm in our mesh improvement framework. Laplacian smoothing is computationally cheap, and the smoother typically spends only a fraction of time per iteration in Laplacian smoothing as compared to the optimizer. However, it is essential to note that Laplacian smoothing can lead to invalid elements for certain meshes, especially when aggressive values of s_f are used. Such situations warrant the use of more sophisticated smoothing techniques such as optimization which we talk about next. Additionally, if the Laplacian smoother makes the mesh invalid, the mesh is restored to its state before that iteration of Laplacian smoother, and that loop of Laplacian smoother is terminated. This approach ensures that the Laplacian smoother is only used when it improves the mesh quality.

7.4.2 Constrained optimization

Optimization is governed by an objective function that is minimized over the region of interest. Typically, mesh optimization is done on a grid point-by-grid point basis where the objective function is the sum of the local objective function for each grid point in the mesh. If the optimization relies on a gradient-based method, such as steepest-descent or conjugate gradients (CG), it requires evaluation of the gradient of the global objective function. Movement of a grid point i affects the quality of the elements that it is connected to. Calculation of the gradient thus requires communication across processors if grid points in the same neighborhood are on different processors. Similar to Laplacian smoothing, we use an element-by-element approach for effecting mesh optimization with minimal parallel communication.

The global objective function for our element-by-element smoother is determined as the sum of the shape metric for every element in the SE mesh.

$$\phi(\underline{\mathbf{x}}) = \phi(\underline{x}, \underline{y}, \underline{z}) = \sum_{e=1}^{E_m} \tilde{\phi}^e(\underline{x}, \underline{y}, \underline{z}) = \frac{1}{2^d}\sum_{e=1}^{E_m}\left[\sum_{i=1}^{2^d}(\phi_i^e)^2\right], \tag{7.4}$$

where, $\tilde{\phi}^e$ is the objective function associated with element e, and ϕ_i^e is the objective function for vertex i of element e. The vertex-based objective function is based on the normalized condition number (in the Frobenius norm) of the local Jacobian $J_{i,e}$ for vertex i of element e,

$$\phi_i^e = \frac{1}{d}\kappa(J_{i,e}) = \frac{1}{d}(\|J_{i,e}\|_F \|J_{i,e}^{-1}\|_F), \tag{7.5}$$

where $J_{i,e}$ is the Jacobian matrix associate with the transformation of from the physical space coordinates $(\mathbf{x}_i^e)$ to reference space coordinates $(\boldsymbol{\xi})$.

$$[J_{i,e}]_{j,k} = \left.\frac{\partial x_j}{\partial \xi_k}\right|_{\mathbf{x}=\mathbf{x_i^e}} j, k \in \{1, .., d\}^2 . \tag{7.6}$$

We remark that in (7.6), the Jacobian is computed based on the small bi- or trilinear micro-elements, rather than the corresponding Nth-order spectral elements. The normalized Frobenius-norm based metric was originally proposed by Knupp [65] and yields a minimum of $\tilde{\phi}_e = 1$ for an ideal element (i.e., a square in 2D or a cube in 3D).

In our mesh optimization framework, we rely on the conjugate gradients method for optimization, which is based on the gradient of the objective function. In the element-by-element framework, the gradient ($\hat{g}^e_{i,j}$) for each vertex i of micro-element e corresponding to the jth space-direction is calculated as

$$\hat{g}^e_{i,1} = \frac{\partial \tilde{\phi}^e}{\partial x_i},\ \hat{g}^e_{i,2} = \frac{\partial \tilde{\phi}^e}{\partial y_i},\ \hat{g}^e_{i,3} = \frac{\partial \tilde{\phi}^e}{\partial z_i}, \quad e = 1, \ldots, E_m, \ \ i = 1, \ldots, 2^d, \tag{7.7}$$

and we compute each component of the gradient using second-order accurate central finite differences as

$$\hat{g}^e_{i,j} = \frac{1}{2h}[\tilde{\phi}_e(\mathbf{x}_i + h\hat{e}_j) - \tilde{\phi}_e(\mathbf{x}_i - h\hat{e}_j)],\ j = 1 \ldots d,\ e = 1, \ldots, E_m,\ i = 1, \ldots, 2^d, \tag{7.8}$$

where h is a local variable, chosen to be .01 times the length of the smallest edge incident to $\mathbf{x}^e_i$, and $\hat{e}_j$ is the unit-vector corresponding to the jth space-direction.

Once the element-by-element gradient vector is computed, it is assembled for CG as,

$$\underline{\hat{\mathbf{g}}} = \begin{pmatrix} \underline{\hat{g}}_x \\ \underline{\hat{g}}_y \\ \underline{\hat{g}}_z \end{pmatrix}, \qquad \underline{\hat{g}}_j = \begin{pmatrix} \underline{\hat{g}}_j^{e=1} \\ \underline{\hat{g}}_j^{e=2} \\ \vdots \\ \underline{\hat{g}}_j^{e=E} \end{pmatrix}, \qquad \underline{\hat{g}}^e_j = \begin{pmatrix} \underline{\hat{g}}^e_{1,j} \\ \underline{\hat{g}}^e_{2,j} \\ \vdots \\ \underline{\hat{g}}^e_{2^d,j} \end{pmatrix}, \tag{7.9}$$

followed by the VV^T operation to add the gradient for shared grid points:

$$\underline{\mathbf{g}} = \begin{pmatrix} VV^T\underline{\hat{g}}_x \\ VV^T\underline{\hat{g}}_y \\ VV^T\underline{\hat{g}}_z \end{pmatrix}. \tag{7.10}$$

(7.10) is important because it ensures that the local element-by-element gradients are assembled and continuous across shared element vertices. Using this assembled gradient vector and other tools described in the previous sections, we can use CG for constrained mesh optimization with Algorithm 2.

In Algorithm 2, l_{min} is the smallest edge length in the mesh with micro-elements, $\underline{\mathbf{x}}_0$ refers to the original mesh, and the parameters α_k and β_k should not be confused with the α and β that we use in the SEM framework to denote the coefficients for the BDFk/EXTk temporal discretization.

In the element-by-element approach, the gradient calculation for a given grid point requires a total of 2

Algorithm 2: Optimization smoothing algorithm

```
$h = 0.1(l_{min})$
Determine $\phi(\underline{\mathbf{x}}_0)$
Determine $\underline{\mathbf{g}}(\underline{\mathbf{x}}_0)$
$\underline{\mathbf{\Delta x}} = -\underline{\mathbf{g}}(\underline{\mathbf{x}}_0)$
for k:=0 to $N_{opt}$ do
    Determine $\alpha_k$ using line search to minimize $\phi(\underline{\mathbf{x}}_k + \alpha \underline{\mathbf{\Delta x}}_k)$
    $\underline{\mathbf{\Delta x}}_k = C^{-1} V V^T \underline{\mathbf{\Delta x}}_k$
    $\underline{\mathbf{\Delta x}}_k = M \underline{\mathbf{\Delta x}}_k$
    $\underline{\mathbf{\Delta x}}_k = W \underline{\mathbf{\Delta x}}_k$
    update $\underline{\mathbf{\Delta x}}$ by surface smoothing
    $\underline{\mathbf{x}}_{k+1} = \underline{\mathbf{x}}_k + \alpha \underline{\mathbf{\Delta x}}_k$
    Determine $\underline{\mathbf{g}}(\underline{\mathbf{x}}_{k+1})$
    $\beta_k = \frac{\underline{\mathbf{g}}(\underline{\mathbf{x}}_{k+1})^T C^{-1} \underline{\mathbf{g}}(\underline{\mathbf{x}}_{k+1})}{\underline{\mathbf{g}}(\underline{\mathbf{x}}_k)^T C^{-1} \underline{\mathbf{g}}(\underline{\mathbf{x}}_k)}$
    $\underline{\mathbf{\Delta x}}_{k+1} = -\underline{\mathbf{g}}(\underline{\mathbf{x}}_{k+1}) + \beta_k \underline{\mathbf{\Delta x}}_k$
```

function evaluations ($\phi(\mathbf{x}_i + h\hat{e_j})$ and $\phi(\mathbf{x}_i - h\hat{e_j})$) in each direction, followed by the action of VV^T operator on it. Thus the total number of function evaluations for calculating the gradient of all the grid points in a mesh is $2dn$, where $n = 2^d E_m$ is the total number of grid points in the decomposed mesh. We also note here that the cost of each function evaluation is bounded since it is simply based on the Frobenius norm of a $d \times d$ matrix and its inverse. Since the total number of function evaluations is $\mathcal{O}(n)$, even as the number of elements increases or the mesh topology changes, the algorithm is easily parallelized and scales well. The P processor parallel time complexity for the mesh optimizer is thus $O(n/P)$, which is optimal.

7.4.3 Mesh smoothing algorithm

The cost of both Laplacian smoother and optimizer are $\mathcal{O}(n)$, which allows them to scale well as the number of elements increases in the mesh. Since the Laplacian smoother does not require any line search step or objective function evaluation for each grid point, the proportionality constant for the cost is much less than that for the optimizer. Consequently, even though the Laplacian smoother can make the mesh invalid during the smoothing process, its cheap computational cost compared to the optimization-based smoothing make it attractive to include it in the smoother. Numerical experiments conducted to compare the speed of Laplacian and optimization-based smoothing show that the Laplacian smoothing can speed up the smoothing process by as much as 3 times. Consequently, we include the Laplacian smoother in our smoother with constraints implemented to ensure that it is turned off if it makes the mesh invalid. Additionally, the combination of Laplacian and optimization based smoothers have proven to be successful in the past [61, 63, 66, 124].

User-specified iterations of Laplacian smoothing (n_{lap}) and optimization (n_{opt}) are applied to the mesh before interpolating the mid-side nodes to the middle of the parent spectral element's edge. This process is repeated for the user-specified number of iterations (n_{outer}). In case the mesh converges to the state with minimum $\phi(\underline{\mathbf{x}})$, the smoother automatically terminates the loop. The interface of the smoother is shown below. Based on our experience with numerous meshes, we have found $n_{outer} = 20$, $n_{lap} = 20$, and $n_{opt} = 40$ to be reasonable default values.

Algorithm 3: Mesh Smoother

```
1:  Interpolate mesh from $G = N$ to $G = 2$
2:  save a copy of original mesh - $\underline{\text{xc}}$,$\underline{\text{yc}}$,$\underline{\text{zc}}$
3:  generate the mask ($M$) for boundary grid points
4:  generate the weight function ($W$) based on minimum distance from wall surfaces
5:  Determine $\phi(\underline{\mathbf{x}}_0)$
6:  for k:=1 to $n_{outer}$ do
7:      $n_{lap}$ iterations of Laplacian smoothing - Algorithm 1
8:      $n_{opt}$ iterations of optimization smoothing - Algorithm 2
9:      save smooth mesh - $\underline{\text{xs}}$,$\underline{\text{ys}}$,$\underline{\text{zs}}$
10:     Determine $\phi(\underline{\mathbf{x}}_k)$
11:     if $\phi(\underline{\mathbf{x}}_k) - \phi(\underline{\mathbf{x}}_{k-1}) < tol$ , go to 12
12: ($\underline{\text{xs}}$,$\underline{\text{ys}}$,$\underline{\text{zs}}$) = ($\underline{W}$)($\underline{\text{xs}}$,$\underline{\text{ys}}$,$\underline{\text{zs}}$) + ($\underline{\text{I}}$-$\underline{W}$)( $\underline{\text{xc}}$,$\underline{\text{yc}}$,$\underline{\text{zc}}$)        This restores boundary layer
13: Interpolate mesh from $G = 2$ to $G = N$
14: Project surface grid points to the surface of the original mesh
15: Gordon-hall mapping of GLL points inside each element
```

After the mesh has been smoothed by the Laplacian smoother and optimizer for user-specified iterations, the weight function (Section 7.2.2) is used to average the original and smoothed mesh as shown in Algorithm 3 and Fig. 7.5. This weight function has the effect of favoring the grid points closer to walls in the original mesh and grid points farther away in the smoothed mesh, thus restoring the boundary layer resolution. Next, the mesh is interpolated from $(E, G = 2)$ to the SEM order $(E, G = N)$, and all the surface GLL points are projected to the surface of the original mesh (Section 7.2.3). Finally, all GLL points in each element are mapped using the algorithm by Gordon and Hall [84, 114].

7.5 Summary

In this chapter, we have presented our methodology for smoothing spectral element meshes using a Laplacian and optimization-based mesh smoother. Using our novel approach of element-by-element smoothing, we will show in Chapter 8 that the mesh smoother demonstrates good scaling even when each MPI rank has as few as ten elements. In Chapter 9, we will present various examples that show the effectiveness of our methodology of mesh smoothing in improving the computational performance of a given mesh. Through different examples, we will see that improving mesh quality leads to an improvement in the conditioning of the system for the PPE, which reduces the computational cost of a calculation. A comparison of the iterative condition number ($\kappa_{iter} = \frac{\lambda_{max}}{\lambda_{min}}$) via the upper Hessenberg matrix constructed during GMRES iterations of pressure solve has helped us establish a strong correlation between κ_{iter} and N_{iter}. For the eight cases that will be presented in Chapter 9, we see that N_{iter} decreases with κ_{iter}. This result is significant because it opens doors to better mesh smoothing strategies. In future work, we aim to exploit this correlation between the conditioning of the upper Hessenberg matrix and pressure iterations to improve our mesh smoother. Instead of just using the conditioning of the Jacobian matrix for mesh smoothing, we will investigate ways to include κ_{iter} in the global function used for mesh optimization. This strategy could potentially help us ensure that every step taken during mesh smoothing improves the conditioning of the system and hence results in decreasing the iteration count for pressure solve.

Chapter 8

Timing & Parallel Scaling

In previous chapters, we focussed on developing the Schwarz-SEM framework for solving incompressible Navier-Stokes equations in complex domains, and developing a mesh smoothing method for improving the computational performance of spectral element meshes. In addition to the accuracy and stability of these methods, their applicability for solving large-scale real-world problems also depends on their speed and scalability. In this chapter, we understand how different aspects of Schwarz-SEM framework impact its speed, and also look at the scaling of the mesh smoother. All the results presented in this section were obtained on Cetus or Mira, the IBM Blue Gene/Q machines at the Argonne Leadership Computing Facility, unless otherwise noted. Additionally, the timing results that we show here for the Schwarz-SEM framework were obtained by averaging the timing data over at-least 50 time-steps.

8.1 Impact of Corrector Iterations

Through our stability analysis results presented in Chapter 5 for single-rate methods, and in Chapter 6 for multirate methods, we have seen that increasing the number of corrector iterations (Q) increases the stability of the Schwarz-SEM framework. Since each corrector iteration requires interdomain boundary data interpolation and an unsteady Stokes solve, we look at understanding how Q impacts the cost of time-advancing the solution of the INSE using the single-rate time-stepping approach.

Here, we consider turbulent flow in a doubly-periodic channel, that we will discuss in detail in Section 9.2. The overlapping meshes used to model this domain have a total of 6304 elements and these meshes are shown in Fig. 9.3. For the timing test, we set $m = 1$ for polynomial order $N = 7$ and 9. Figure 8.1 shows how the time to solution per time-step (T_{step}) varies as we increase Q. The time to solution includes the time for solving the pressure Poisson equation and a Helmholtz equation for each component of velocity. As we can see in Fig. 8.1, the time to solution increases almost linearly for $Q > 4$. This is likely because in the SEM framework, the pressure Poisson equation is solved by using an initial guess based on the space of solutions at previous P_r time-steps. This projection method for accelerating the pressure solve was developed for the monodomain SEM framework, and is used in each subdomain in the Schwarz-SEM framework. When $Q > 0$ in the Schwarz-SEM framework, the P_r solutions that are used to generate the initial guess, essentially only span previous $P_r/(Q+1)$ time-steps. As a result, the quality of the initial approximate generated using this projection technique decreases as Q increases. We plan on looking into better projection techniques for reducing the cost of increasing Q, in future.

The results in Fig. 8.1 also show why the improved predictor-corrector scheme that was presented in Section 5.4, will help in significantly reducing the computational cost of the Schwarz-SEM framework. In contrast to the original PC scheme where we observed that even-Q was less stable than odd-Q (Chapter 5), the improved PC scheme shows that increasing Q monotonically increases the stability of the method. As a result, we can increase Q by 1 instead of 2, if the current Q is not sufficient from a stability perspective.

We note that not all cases require $Q > 0$. For examples such as the turbulent channel flow (Section 9.2) and buoyant plume (Section 9.7), we have observed that even the simple time-lagged Schwarz updates ($Q = 0$ with $m = 1$) have produced results that agree well with literature.

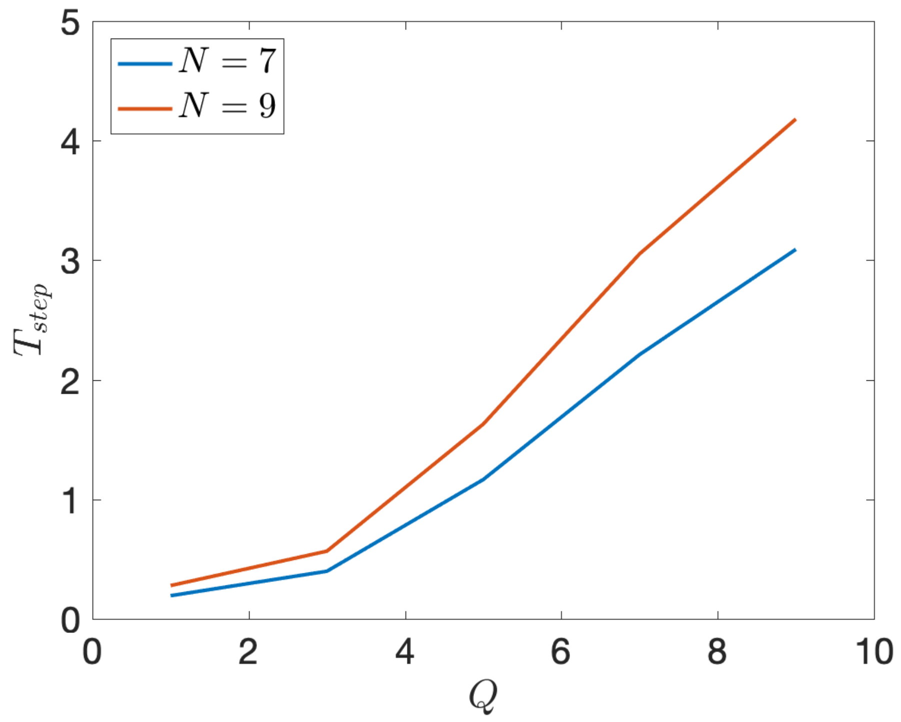


Figure 8.1: Impact of Q on the time to solution per time-step for turbulent channel flow at $N = 7$ and $N = 9$. The domain was modeled with two overlapping meshes with a total of 6304 elements (1444 elements in the lower mesh and 4860 elements in the upper mesh).

8.2 Impact of Multirate Time-Stepping

In Chapter 6, we introduced a multirate time-stepping predictor-corrector scheme for time-advancing the solution of INSE using different time-step sizes in overlapping subdomains. With multirate time-stepping, the subdomain with faster time-scales (Ω^f) uses a smaller time-step size in comparison to the subdomain with slower time-scales (Ω^c), and as a result Ω^c needs to take many fewer time-steps in comparison to Ω^f. The number of time-steps needed for each subdomain to integrate up to a fixed time depends on the time-step ratio $\eta = \Delta t_c/\Delta t_f$. Using this novel multirate time-stepping predictor-corrector scheme, we have demonstrated that it maintains the exponential convergence with N and third-order temporal accuracy of the SEM.

In Section 9.7, we will demonstrate the use of multirate time-stepping scheme for modeling a buoyant plume with two overlapping meshes of different element size. For the buoyant plume, a dense inner grid is used to resolve the fine scale structures in the plume where the flow is buoyancy driven, and a coarse outer grid is used to model the tank in the far-field. Due to the difference in the spatial and temporal scales of the overlapping grids, multirate time-stepping scheme enables us to use $\eta = 50$ for this example. In order to validate the multirate time-stepping scheme, numerical calculations were done with $\eta = 5$ and 50. Here, we use the calculation done with $\eta = 5$ to demonstrate how the multirate time-stepping reduces the computational cost in comparison to the single-rate time-stepping ($\eta = 1$) for the same problem.

For the buoyant plume problem, $E_f = 55,480$ elements for the dense inner grid (Ω^f) and $E_c = 15,560$ elements in the coarse outer grid (Ω^c). For overlapping subdomains, ideally one would partition the domain in parallel such that the time to solution per time-step (T_{step}) is similar for each subdomain. For the single-rate time-stepping scheme, a good rule of thumb to choose the number of MPI ranks for each subdomain is

$$\frac{P_c}{P_f} \approx \frac{E_c}{E_f}, \tag{8.1}$$

where P_f and P_c are the number of MPI ranks use to partition Ω^f and Ω^c, respectively. Based on E_c and E_f for this example, we can set $P_c \approx P_f/4$. For the multirate scheme, however, since Ω^c has many times fewer steps as compared to the Ω^f, the number of MPI ranks for Ω^c can be reduced even further.

Figure 8.2 compares how T_{step} varies with P_c for the single-rate and multirate time-stepping scheme, while keeping P_f fixed at 4096 MPI ranks. These calculations were done with $m = 1$, $Q = 0$, and $N = 7$. The time per time-step was obtained for multirate time-stepping scheme by monitoring the mean time taken by Ω^f for each sub-time-step, which is equivalent to a single time-step in the single-rate time-stepping

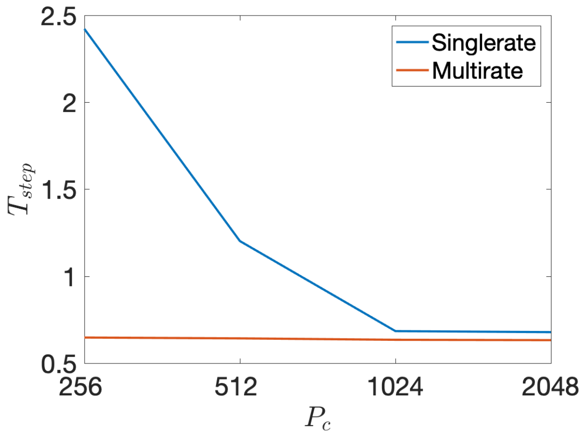


Figure 8.2: Comparison of time to solution per time-step (T_{step}) for the single-rate and multirate time-stepping scheme with number of MPI ranks (P_c) used for the subdomain Ω^c with slow time-scales. These tests were done using the buoyant plume problem (Section 9.7), where $E_f = 55,480$ and $E_c = 15,560$.

scheme. The time-step size was kept same for Ω^f for the multirate and single-rate time-stepping scheme, in order to ensure fair comparison.

As we can see, the single-rate time-stepping is most efficient when $P_c = 1024 = P_f/4$. As P_c is decreased, the time to solution increases as expected. We also notice that as P_c is increased from 1024 to 2048, the time to solution doesn't change. This is because when $P_c > P_f/4$, T_{step} is limited by Ω^f.

In contrast to the single-rate scheme, since Ω^c has to take many fewer time-steps with the multirate time-stepping scheme, $P_c = P_f/16$ is as effective as $P_c = P_f/4$. P_c cannot be reduced further because of the constraint on maximum memory that can be allocated on each MPI rank. Additionally, we see that the multirate time-stepping scheme does better than the single-rate time-stepping scheme for equivalent number of MPI ranks because it requires fewer calls to *findpts_eval*.

Based on the results presented here, we conclude that load balance can be ensured for multirate time-stepping-based calculations by choosing the MPI ranks for each subdomain such that

$$\frac{P_c}{P_f} \approx \frac{E_c}{\eta E_f}. \tag{8.2}$$

We note that (8.1) and (8.2) are based on our experience with numerical experiments, and have proven to be effective for choosing the number of MPI ranks to ensure load balance between different subdomains, due to the private-memory model (Section 2.1.5) that the Schwarz-SEM implementation is based on. (8.1) and (8.2) will likely not be effective in cases where the difference in the number of iterations needed for pressure Poisson solve or Helmholtz solve for velocity varies significantly between different subdomains.

8.3 Impact of Changes to Interpolation via *findpts_eval*

findpts_eval is used at each Schwarz iteration in the Schwarz-SEM framework to interpolate interdomain boundary data. In Section 3.2.3, we had described our methodology for improving the performance of *findpts_eval* by reducing the sorting and communication cost of the algorithm. To test our improvements, we use the monodomain spectral element mesh generated for modeling a thermally buoyant plume (Section 9.7). The monodomain mesh has 70,400 spectral elements, and we set $N = 5$ and 7 with 275,000 points distributed uniformly throughout the domain.

To test the time for interpolation in the old and new interpolation framework, *findpts* is first used to determine the computational coordinates of the 275,000 sought points, and then *findpts_eval* is used to interpolate a scalar field. Figure 8.3 compares the time for interpolation for these 275,000 points (T_{eval}) using the original and improved *findpts_eval*. As we can see, our improvements to *findpts_eval* show a 3X speed-up in comparison to the original scheme.

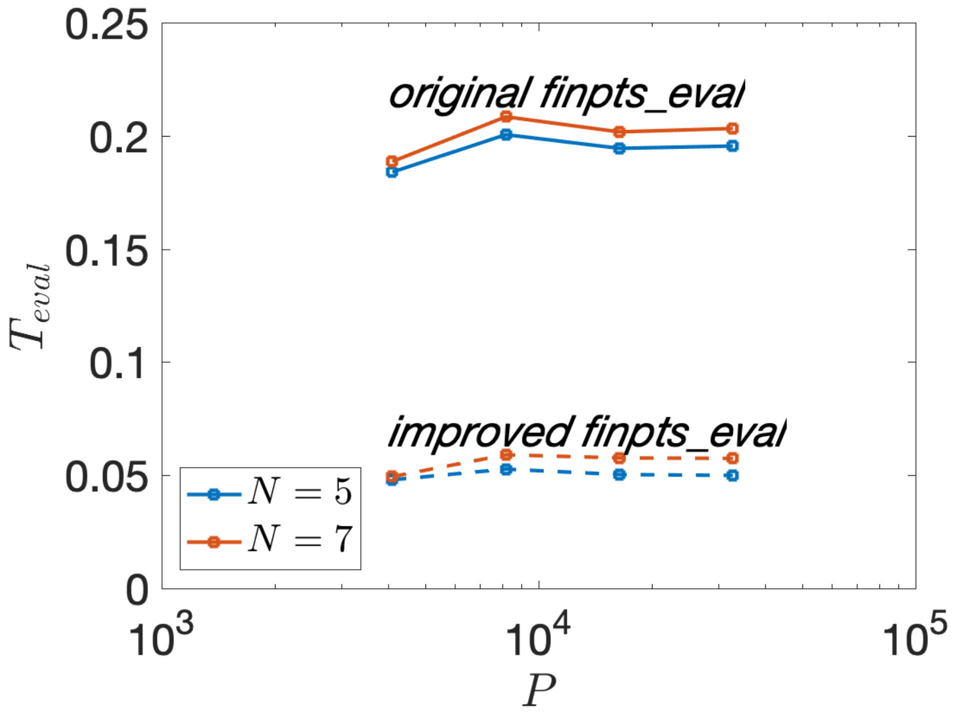


Figure 8.3: Comparison of time to interpolation (T_{eval}) via *findpts_eval* using the original and improved scheme for the buoyant plume example ($E = 70,400$ with $N = 5$ and 7.)

8.4 Strong Scaling for the Schwarz-SEM Framework

In the context of the Schwarz-SEM method, we are concerned with the cost of *findpts* for finding computational coordinates of interdomain boundary grid points, the cost of *findpts_eval* for interpolating scalar functions for these grid points, and the overall time to solution per time-step. For evaluating the parallel performance of the Schwarz-SEM framework, we present two different examples.

The first example is the turbulent channel flow case that will be discussed in Section 9.2. The domain for this example is discretized using two overlapping grids with 4860 and 1444 elements, respectively. These grids have 324 and 361 elements, respectively, for interdomain boundary surface. Figure 8.4 shows how time to solution for NS solve, time for finding donor elements of interdomain boundary points (*findpts*), and time for interpolation (*findpts_eval*) varies with number of grid points ($n = EN^D$) per MPI rank (P) for the case when $m = 1$, $Q = 0$ and $N = 7$. Here, we consider E and P as the total number of spectral elements ($E = 6304$) and MPI ranks, respectively, for the two overlapping grids. The results presented here go up to $P = 5120$, where due to the difference in the element counts between the two overlapping grids, the ratio of MPI ranks used for overlapping subdomains is $1 : 4$ (fewer MPI ranks for the grid with fewer elements). Figure 8.4(b) shows a comparison of time for different components of the Schwarz-SEM framework with the total number of MPI ranks. We can see from the results in Fig. 8.4(a) and (b) that finding interdomain boundary grid points and interpolating a scalar field are computationally much cheaper as compared to the NS solve. The scaling for *findpts* and *findpts_eval* is not ideal, as expected, due to the inherent load imbalance in overlapping grids because all interface elements might not be partitioned onto separate processors.

Figure 8.4(c) shows the parallel efficiency versus n/P and as we can see, the parallel efficiency of the calculation does not drop below 70% until $n/P \approx 2000$, which meets theoretical expectations [121].

The second example that consider here is flow over a wall-mounted cube that is used to understand how laminar boundary layer flow transitions to turbulence due to a wall-mounted roughness. This case is ideal for the Schwarz-SEM framework because fine scale structures of the flow are confined near the wall, and using a single conforming grid leads to unwanted resolution in the far-field. The overlapping grids used here have 99,840 elements in the subdomain modeling the roughness (near the surface), and 30,932 elements for the subdomain modeling the region away from the surface. These overlapping grids have 6272 and 2812 elements, respectively, for the interdomain boundary surface. The Schwarz-SEM framework reduced the total element count by 30% in comparison to the monodomain case. The application of Schwarz-SEM for the wall-mounted cube is discussed in detail in Section 9.8.

For the strong scaling study, we set $N = 7$, and the ratio of MPI ranks used for overlapping subdomains

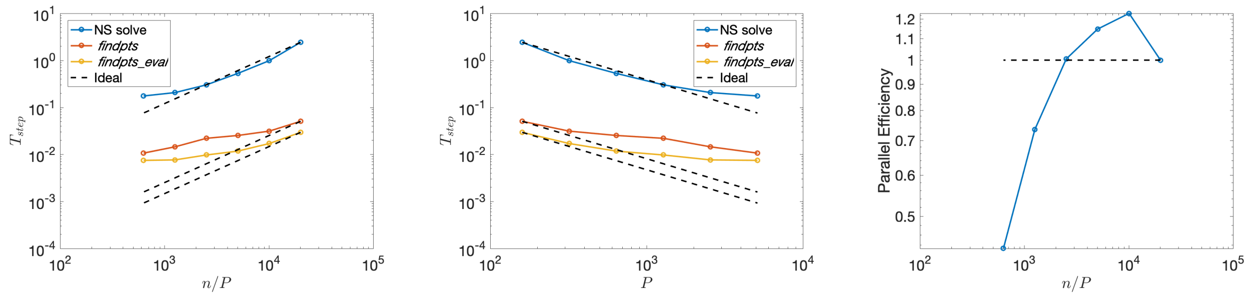


Figure 8.4: (left to right) Strong scaling plot showing total time for the NS solve, *findpts*, and *findpts_eval* versus (a) total number of points per processor (n/P) and (b) total processor P. (c) Parallel-efficiency versus n/P. The overlapping grids have a total of $E = 6304$ spectral elements and the results shown here were obtained from calculations with $m = 1$, $Q = 0$ and $N = 7$.

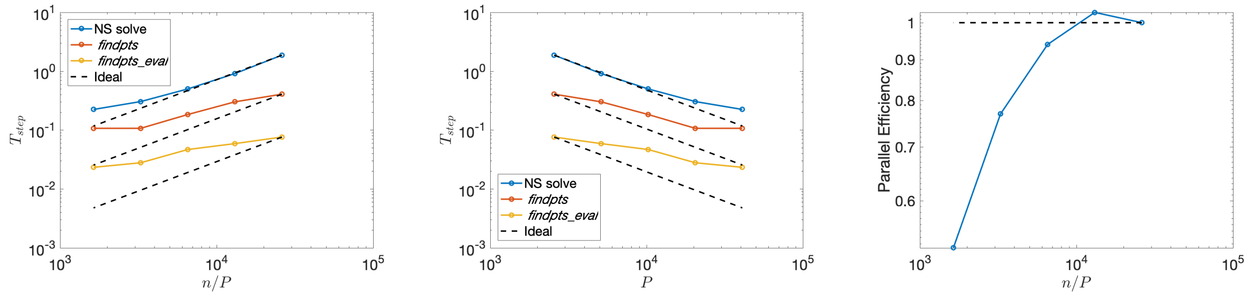


Figure 8.5: (left to right) Strong scaling plot showing total time for the NS solve, *findpts*, and *findpts_eval* versus (a) total number of points per processor (n/P) and (b) total processor P, and (c) parallel-efficiency versus n/P. The overlapping grids have a total of $E = 130,772$ spectral elements and the results shown here were obtained from calculations with $m = 1$, $Q = 0$ and $N = 7$.

is 1 : 4 due to the difference in element counts of the two grids. Fig. 8.5(a) and (b) show the strong scaling plot for time for NS solve, time for donor-element search (*findpts*), and time for interpolation (*findpts_eval*) with $m = 1$, $Q = 0$, and $N = 7$ versus n/P and P, respectively. Here, we define $n = EN^d$ with E as the total number of elements (130,772) and P as the total number of MPI ranks used for the two subdomains. The parallel scaling results presented here are for P up to 40,960.

We observe that similar to the results for channel flow (Fig. 8.4), the cost of *findpts* and *findpts_eval* is a fraction of the cost of the NS solve. We also see that the parallel efficiency of the calculation does not drop to below 70% until $n/P \approx 2000$, which matches the theoretical expectations [121].

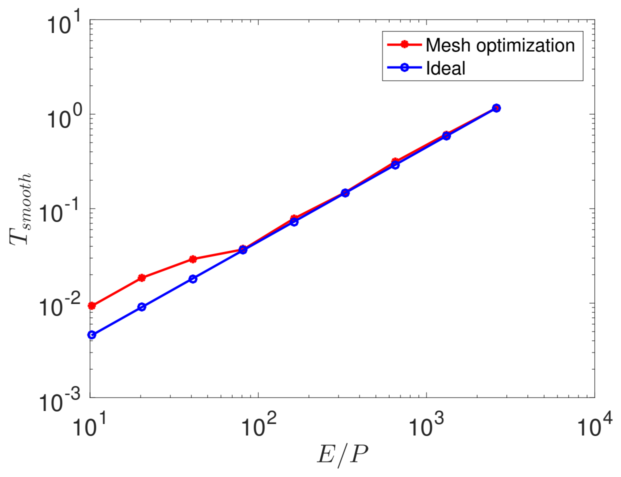


Figure 8.6: Scaling test comparing total time per iteration of smoothing with number of elements per processor. The timing results shown here were obtained from numerical experiments done on the Blue Waters Supercomputer.

8.5 Strong Scaling for Mesh Optimization

In Chapter 7, we presented our mesh smoothing framework that is based on a combination of Laplacian smoothing and function optimization. Due to the data locality and private memory model that we use in the monodomain- and Schwarz-SEM framework (Section 2.1.5), mesh smoothing is done on an element-by-element basis. This approach has an advantage that all the elements of a mesh can be smoothed simultaneously on any number of processors while keeping the communication cost to a minimum.

Here, we use the monodomain mesh constructed for analyzing oscillatory flow over a cylinder (Section 9.9.4) to test the scaling of the mesh smoother. This mesh is chosen because of its high element count, $E = 83,598$. Figure 8.6 shows a comparison of the total time it takes per smoothing iteration (T_{opt}) versus the total number of spectral elements per MPI rank (E/P). The plot indicates that the smoother shows nearly linear speed-up even at the strong scale limit where there are only 10 elements per processor.

We note that typically, the mesh smoother is used as a preprocessing step where the mesh of interest is smoothed before being used for a production level calculation. Thus, the computational resources needed for mesh smoothing are insignificant in comparison to the resources that are typically used for turbulent flow calculations on HPCs. Nonetheless, the results here show that the mesh smoother that we have developed shows good performance in parallel.

8.6 Summary

In this chapter, we presented results that show how different aspects of the Schwarz-SEM framework impact its performance. Our analysis show that the cost of each corrector iteration is significant, and it is crucial to minimize Q to ensure that the cost of Q corrector iterations does not offset the savings associated with reduction in element count due to use of overlapping grids. The results in this chapter also show that multirate time-stepping can significantly reduce the computational resources needed to solve the INSE in the subdomain with slow time-scales. The improvements that we have made to *findpts_eval* have shown that reducing the sorting and communication cost of the algorithm has led to 3X speed-up for high-order interpolation in parallel. We also presented strong scaling results for the Schwarz-SEM framework, which show that the parallel efficiency of the framework meets the theoretical expectations [121]. Finally, we also discussed strong scaling tests on mesh smoothing that show nearly linear speed-up even at the strong scale limit, when there are only 10 elements per processor.

Chapter 9

Applications

The Schwarz-SEM framework is allowing us to tackle various fluid-thermal problems that are intractable with the monodomain SEM framework. In this chapter, we start with two examples (vortex breakdown in a canister and turbulent channel flow) that we have used to benchmark the Schwarz-SEM framework and establish its accuracy. We then present the application of overlapping grids for solving various fluid-thermal applications that are intractable with the monodomain SEM method.

9.1 Vortex Breakdown in a Canister

We illustrate the potential and capabilities of the Schwarz-SEM solver on a problem that has a fairly sensitive structure, namely, vortex breakdown in a circular canister with a rotating lid, as studied experimentally by Escudier [125]. Escudier considered cylinders of various height to radius ration (H/R) at different Reynolds number, $Re = \tilde{\Omega}^2 R/\nu$. $\tilde{\Omega}$ is the angular velocity of the rotating lid, and following standard nondimensionalization practice, we set $\tilde{\Omega} = R = 1$ and $\nu = 1/Re$. For the current study, $Re = 1854$ and $H/R = 2$. Here, we use the Schwarz-SEM framework to study this problem with two overlapping grids and compare our results with the experimental results of [125] and with a monodomain SEM-based solution.

The monodomain mesh has 140 elements at $N = 9$ and is generated by extruding a planar mesh with 20 elements, which is shown in Fig. 9.1(left). The overlapping meshes are generated by cutting the monodomain mesh across the cylinder axis such that $\Omega_z = [0, 1.21R]$ for the lower domain and $\Omega_z = [0.8R, 2R]$ for the upper domain ($E = 80$ for both overlapping meshes). Figure 9.1(right) shows the slice view, parallel to cylinder-axis, of the overlapping spectral element meshes. The boundary-conditions are no-slip walls for all surfaces except the top of the canister (in z-direction), where we impose a fixed Dirichlet velocity to model rotation. The calculations were run with $m = 1$ and $Q = 0$ (no corrector iterations) for 2000 convective time-units to reach steady state.

Figure 9.1 compares the axial velocity along the centerline for monodomain and overlapping grids based solution. The locations where the axial velocity (w) reverses sign, corresponds to the location of bubbles.

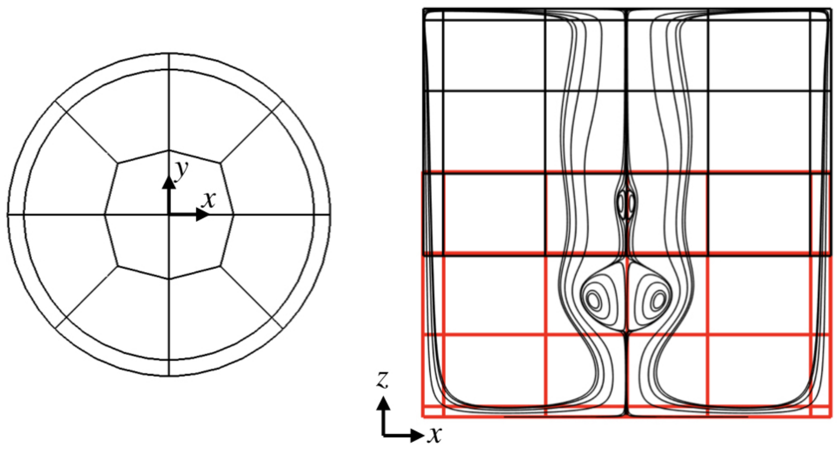


Figure 9.1: Vortex breakdown problem: (left) 2D mesh used to generate the 3D mesh, and (right) in-plane streamlines along with slice view of the two overlapping meshes used for discretizing the domain.

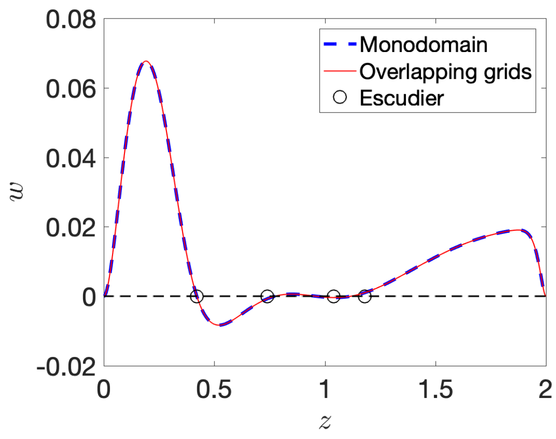


Figure 9.2: Vortex breakdown problem: w along cylinder centerline showing locations of velocity reversals.

	z_1	z_2	z_3	z_4
Overlapping	0.42	0.78	0.96	1.12
Monodomain	0.42	0.77	0.96	1.12
Escudier	0.42	0.74	1.04	1.18

Table 9.1: Zero-crossings (z) for vertical velocity along cylinder centerline for the vortex breakdown.

Figure 9.1 also shows the location of these velocity reversals that are reported by Escudier. Comparing these locations, we find that the Schwarz-SEM results are within 1 percent of the monodomain SEM results, and to within 7 percent of Escudier's experiments (see Table 9.1).

9.2 Turbulent Channel Flow

Turbulent channel flow and boundary layer flows are a class of problems where overlapping grids offer the potential for significant savings. These flows feature fine-scale structures near the wall with relatively larger scales in the far-field. As a first step to addressing this class of problems, we validate our Schwarz-SEM scheme for turbulent flow in a doubly-periodic channel, for which abundant data is available in the literature. In particular, we compare mono- and multidomain SEM results at Reynolds number $Re_\tau = u_\tau h/\nu = 180$ with direct numerical simulation (DNS) results of Moser et al. [126], who used 2.1 million grid points, and with the DNS of Vreman & Kuerten [127], who used 14.2 million grid points. Both, [126] and [127], use Fourier modes in the periodic/horizontal direction, and Chebyshev modes in the wall-normal direction. The Reynolds number Re_τ is based on the friction velocity u_τ at the wall, channel half-height h and the fluid kinematic viscosity ν, with $u_\tau = \sqrt{\tau_w/\rho}$ determined using the wall shear stress τ_w and the fluid density ρ.

Figure 9.3 shows a slice view of the overlapping meshes used to discretize the domain, and Table 9.2 lists the key parameters for the four different calculations. The monodomain grid has 5832 elements ($18\times18\times18$). Simulations were primarily conducted with polynomial order $N = 9$. Thus, the mono-domain case has 5.832 million grid points. The overlapping meshes were generated with a resolution similar to the monodomain grid. The lower mesh has 1444 elements, and the upper mesh has 4860 elements. The total number of elements in the overlapping grid calculations is higher than the monodomain calculation because the number of elements in the streamwise and spanwise direction was increased by 1 in the lower mesh to avoid elements coinciding in the overlap. Following [126] and [127], the streamwise and spanwise lengths of the channel were $4\pi h$ and $4\pi h/3$, respectively, and we set the channel half-height to $h = 1$. Statistics for all SEM calculations were collected over 50 convective time units.

	Re_τ	Ω_y/h	Grid-size
Monodomain	179.9	$[-1, 1]$	$18\times18\times18\times N^3$
Overlapping	179.9	$[-1, -0.88]$	$19\times4\times19\times N^3$
		$[-0.76, 1]$	$18\times15\times18\times N^3$
Moser	178.1	$[-1, 1]$	$128\times129\times128$
Vreman	180	$[-1, 1]$	$384\times193\times192$

Table 9.2: Parameters for channel flow calculations. We consider different values of N for the monodomain and overlapping grids for comparison. The second column in the table reports the actual shear Reynolds number of the simulation. This number is slightly different from the imposed 180, and can be used as a measure for lack of spatial resolution.

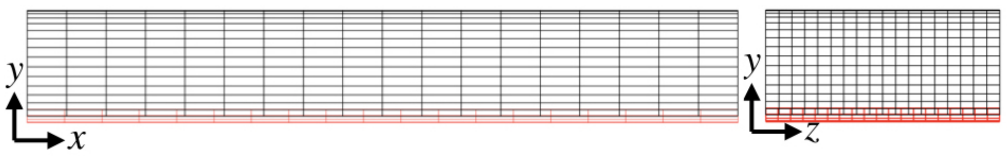


Figure 9.3: Slice view of the overlapping meshes used for the channel. The two meshes are shown in red and black, with overlap in the wall-normal (y) direction.

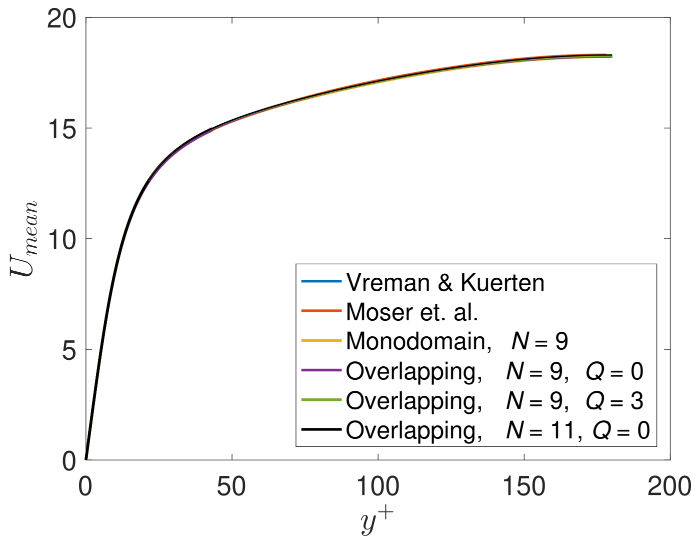


Figure 9.4: Comparison of monodomain, overlapping grid, and Moser et al. with Vreman & Kuerten for U_{mean} (left) and u_{rms}, v_{rms} and w_{rms} at $N = 9$ and $N = 11$.

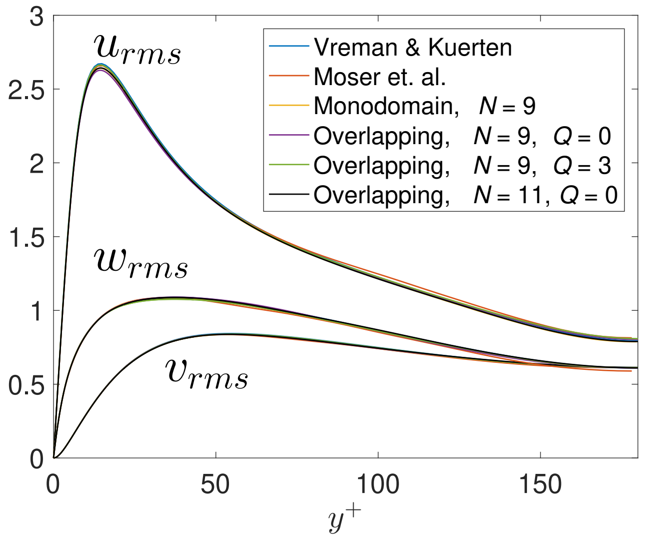


Figure 9.5: Comparison of monodomain, overlapping grid, and Moser et al. with Vreman & Kuerten for U_{mean} (left) and u_{rms}, v_{rms} and w_{rms} at $N = 9$ and $N = 11$.

Figure 9.4 shows the time-averaged mean streamwise velocity (U_{mean}) and Fig. 9.5 shows the mean turbulence intensity in the three-direction (u_{rms}, v_{rms} and w_{rms}), versus $y^+ = u_\tau y/\nu$ for each case determined using y, the distance from the nearest wall. U_{mean}, u_{rms}, v_{rms} and w_{rms} were obtained by temporally averaging the velocity field, and then spatially averaging it in the homogeneous directions. For the Schwarz case, several combinations of resolution (polynomial order N), and corrector iterations (Q) were considered. We quantify the relative error for each quantity by computing the norm of the relative percent difference from the results of Vreman & Kuerten. Error is calculated by using

$$\epsilon_\psi(y) \quad := \quad 100\frac{|\psi(y) - \psi(y)_{Vreman}|}{\psi(y)_{Vreman}} \tag{9.1}$$

and

$$||\epsilon_\psi|| \quad := \quad \frac{1}{2h}\int_{-h}^{h} \epsilon_\psi \, dy \tag{9.2}$$

for $\psi = U_{mean}$, u_{rms}, v_{rms} and w_{rms}.

Table 9.3 lists $||\epsilon_\psi||$ for different quantities and we observe that $||\epsilon_\psi||$ is within three percent of the results by [127] for $\psi = U_{mean}$, u_{rms}, v_{rms} and w_{rms}. The results in Table 9.3 demonstrate that increasing the number of Schwarz iterations (Q) leads to improved accuracy, as expected. However, even the simple time-lagged Schwarz updates (i.e., without corrector iterations) are within 3 percent of the values in the literature. These results indicate that for production-scale turbulence calculations, typically performed as large-eddy simulations, the noniterated simulations ($Q = 0$) can provide a fast and sufficiently accurate pathway to simulating turbulence in complex domains.

	U_{mean}	u_{rms}	v_{rms}	w_{rms}
Monodomain, $N = 9$	0.17	1.00	0.60	0.57
Monodomain, $N = 11$	0.16	0.46	0.49	0.71
Overlapping, $N = 7$, $Q = 0$, $m = 1$	0.32	2.83	1.63	1.78
Overlapping, $N = 9$, $Q = 0$, $m = 1$	0.43	1.69	0.89	0.77
Overlapping, $N = 9$, $Q = 3$, $m = 3$	0.21	0.92	0.60	1.05
Overlapping, $N = 11$, $Q = 0$, $m = 1$	0.17	1.58	0.74	0.31
Moser et al.	0.04	0.15	0.52	1.62

Table 9.3: Relative % difference for channel flow results compared with Vreman & Kuerten [127] for different polynomial order (N), corrector iterations (Q) and order of extrapolation (m) for interdomain boundary data.

9.3 Heat Transfer Enhancement in a Pipe with Wire-Coil Insert

This problem is being studied in collaboration with scientists at Argonne National Lab and the University of Illinois Urbana-Champaign (UIUC), who are studying heat transfer augmentation due to wire-coil inserts in pipes. Annie Goering and Kento Kaneko have done the monodomain calculations discussed here.

The effectiveness of wire-coil inserts to increase heat transfer in pipe flow has been studied through an extensive set of experiments by Collins *et al.* at Argonne National Laboratory (ANL) [128]. The goal of the experiments, done with wire-coil inserts in circular casings, was to understand how heat transfer could be augmented in high-heat load components at the Advanced Photon Source (APS) at ANL. However, the actual casings that were used at APS were noncircular, and understanding the impact of different casing shapes was not possible without reworking the experimental apparatus. Thus, the onus of studying the effect of different casing shapes on the heat transfer falls on the numerical simulations.

The domain for wire-coil insert in a circular casing is straightforward to mesh since it can be generated by twisting a 3D mesh that is created by sweeping (extruding) a 2D template. Wire-coil inserts in noncircular casings, however, are not readily accessible for the monodomain approach due to the topological constraints induced by the shape of the casing and wire-coil insert. We first use the overlapping grid approach for the circular-casing configuration and compare these results against available monodomain simulations and experimental data. We then apply the overlapping approach to the more challenging noncircular casing geometry.

Figure 9.6 shows the wire-coil insert in a circular pipe that was used for the experiments. The geometric parameters of interest are wire-coil pitch (p_c) and diameter (e_c), and pipe inner diameter (D_c). p_c and D_c are indicated in Fig. 9.6 along with the pipe outer diameter (D_{out}). The experiment compared Nusselt number (Nu) for a range of p_c, e_c, and D_c. The configuration that yields the highest Nu for a given Re is $e_c/p_c = 0.4273$ with $e_c/D_c = 0.2507$ [128]. Coils with shorter and longer pitches resulted in Nusselt numbers that were not as high, but were nonetheless higher than the straight pipe without a wire-coil insert. We refer to the configuration that yields the highest Nu as the *optimal-pitch*. As a first step towards validation, we consider the case with $e_c/p_c = 0.4273$ (*optimal-pitch*) and $e_c/p_c = 0.0940$ (*long-pitch*), and compare our results with the experimental data and monodomain SEM calculations. The Reynolds number of the flow is $UD_c/\nu = 5300$, and the Prandtl number is $\nu/\alpha_c = 5.8$, where ν is the kinematic viscosity and α_c is thermal diffusivity. The velocity U for determining the Reynolds number is based on the mean flow speed in a pipe without a wire-coil insert for the flow rate ($\tilde{Q} = \pi D_c^2 U/4$) at which the experiments were done.

The spectral element meshes were periodic in the streamwise direction, and a fixed flow rate was imposed

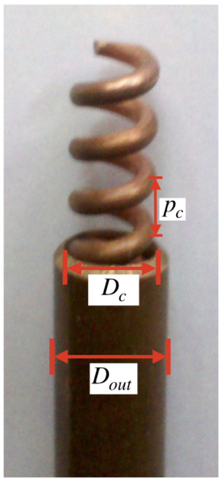


Figure 9.6: Wire-coil insert in a circular pipe used for experiments at Argonne National Lab [128]. The wire-coil pitch (p_c), pipe inner diameter (D_c) and outer diameter (D_out) are indicated in the figure.

through overlapping meshes using the method described in Chapter 4. A fixed heat flux $q^{''} = 1$ was imposed on the outer walls of the pipe, and the Nusselt number Nu was calculated by monitoring the bulk fluid temperature and the inner-wall temperature of the casing. All calculations were started at a relatively low polynomial order of $N = 5$, with $m = 1$ and $Q = 0$ to develop the flow. Eventually, the flow was spatially converged by increasing N. Additionally, the number of Schwarz iterations was set to $Q = 3$ with $m = 3$ for ensuring high-order temporal accuracy of interdomain boundary data. A detailed discussion on the boundary conditions for monodomain SEM and a description of how the Nusselt number is calculated is given in [129].

9.3.1 Wire-coil insert in a circular pipe

For the long-pitch case, $e_c/p_c = 0.0940$, a 2D mesh with 448 elements is azimuthally extruded along a helical path ($E = 10,752$ with $E_f = 5760$ for fluid domain and $E_s = 4992$ for the solid domain) to model the outer part of the domain. The singularity at the pipe centerline is avoided by using a second (overlapping) mesh ($E = 2016$), generated by extruding a planar mesh with 84 elements, to model the central flow channel. Following the monodomain SEM calculations, the outer diameter of the pipe was set to $D_{out} = 4D_c/3$ for the long-pitch case. The meshes overlapped in the radial direction between $r \in [0.15, 0.2]$, and integration weights were setup accordingly to enable calculation of flow rate and Nusselt number. Overlapping meshes avoid the topological constraint of mesh conformity and lead to better mesh quality. Mesh smoothing [73] further improves the quality of the mesh. Figure 9.7 shows a slice view of the overlapping meshes generated for the long-pitch case and Fig. 9.8 shows contours of velocity magnitude and temperature for flow through

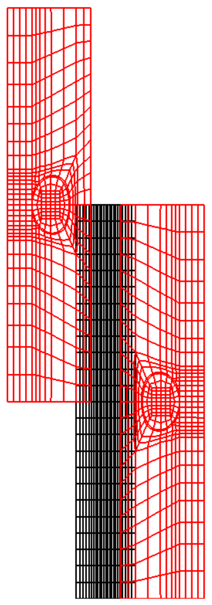
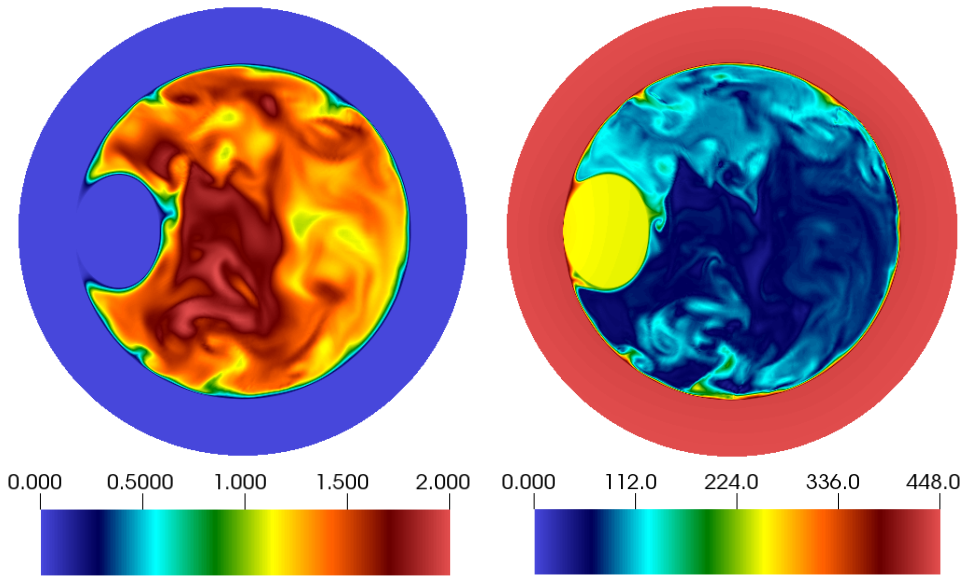


Figure 9.7: Overlapping spectral element meshes generated for the long-pitch case.



(a) Velocity (b) Temperature

Figure 9.8: Slice view of the (left) velocity magnitude and (right) temperature contours for flow through a circular pipe with a wire-coil insert. The Nusselt number for the *long-pitch* configuration is $Nu \approx 113$.

a circular pipe with a wire-coil insert.

The overlapping grid calculation for the long-pitch case, $e_c/p_c = 0.0940$, was spatially converged at $N = 11$, and determined that the Nusselt number is $Nu = 113$. The Nusselt number compares well with the experimental data ($Nu \approx 100$) and the monodomain SEM ($Nu = 113$) calculation. The difference between these results and experiments can be attributed to the uncertainty associated with the experimental apparatus.

For the *optimal-pitch* case, $e_c/p_c = 0.4273$, a 2D mesh with 440 elements is extruded azimuthally with a helical pitch ($E = 10{,}560$ with $E_f = 5952$ for fluid domain and $E_s = 4608$ for the solid domain) for the outer region, and overlapped with a mesh ($E = 2184$) generated by extruding a planar mesh with 91 elements. The Re and Pr were unchanged, but the outer diameter of the pipe was set to $D_{out} = 2D$. The diameter was increased in comparison to the long-pitch case because the wire-coil insert for optimal-pitch was also used in a noncircular pipe, which required $D_{out} = 2D$ to accommodate for the change in the shape of the pipe. Figure 9.9 shows a slice view of the overlapping meshes generated for this calculation, and Fig. 9.10 shows a slice view of the velocity magnitude and temperature contours for the optimal wire-coil insert in a pipe.

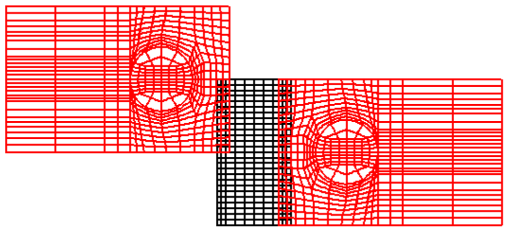
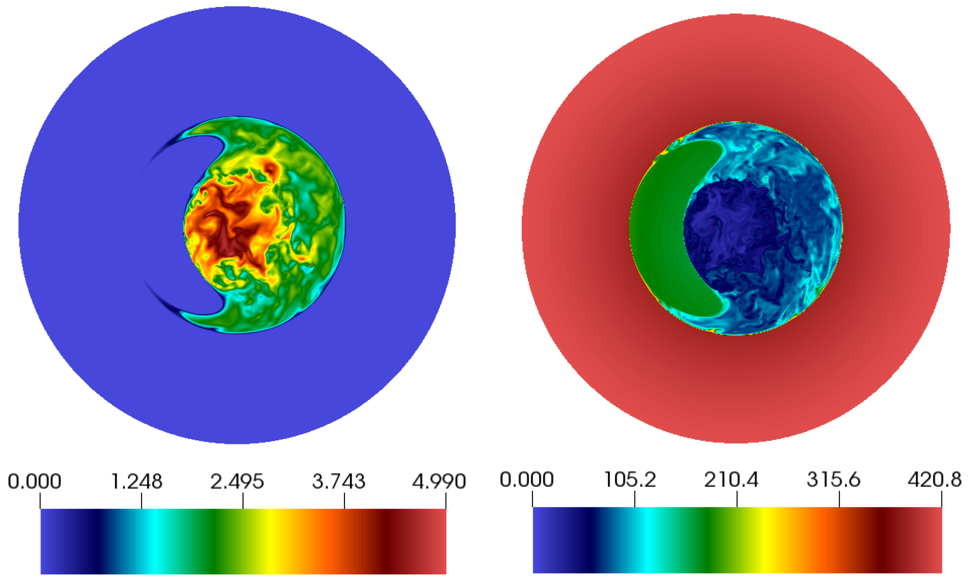


Figure 9.9: Overlapping spectral element meshes generated for the optimal-pitch case.

The Nusselt number was determined to be around 184 by the experiment, 194 by the monodomain approach, and 186 by the overlapping-grid calculation, for the optimal-pitch wire-coil insert in a circular pipe. As we can see, the Schwarz-SEM framework gives a good comparison with the experiment, and there is about a 4% relative difference in the Nusselt number obtained from the monodomain simulation. We are currently looking at the monodomain simulation to ensure that it is spatially converged, which could possibly explain why the Nusselt number is different between the monodomain and multidomain calculation. Ongoing efforts with monodomain spectral element simulations also seek to understand why the optimal-pitch case leads to the highest Nu for wire-coil insert inside a circular pipe.



(a) Velocity (b) Temperature

Figure 9.10: Slice view of the (left) velocity magnitude and (right) temperature contours for flow through a circular pipe with a wire-coil insert. The Nusselt number for the *optimal-pitch* configuration is $Nu \approx 186$.

9.3.2 Wire-coil insert in a noncircular pipe

The experiments done at ANL to understand the impact of wire-coils on heat transfer augmentation were done with a circular pipe (casing). However, the actual casings were noncircular and understanding the impact of this change in shape is not possible without reworking the original experimental apparatus. Thus, the effect of the noncircular casing on heat-transfer could only be studied numerically. As discussed in Section 1.1, wire-coil inserts in noncircular casings are not readily accessible to the monodomain SEM approach because of the combination of twisting/shearing in the inner geometry with the sharp corners of the outer casing over-constrains the meshing requirements, similar to the twisted ribbon configuration of Fig. 1.2. Using the case where the wire-coil insert is in a circular pipe, we have shown that the Schwarz-SEM framework provides an effective way for simulating turbulent flow and heat transfer in complex domains. Here, we consider two different Re_D (5300 and $10,000$) for the optimal-pitch wire-coil insert in a noncircular pipe. Figure 9.11 shows the difference in the cross section of the circular and noncircular pipe, along with velocity magnitude contours.

We note that another advantage of using overlapping grids instead of a single conforming mesh is that a monodomain grid requires twisting the mesh in the axial direction. This mesh twisting/stretching leads to a decrease in the effective resolution of the domain, which is demonstrated through a simple 2D example in Appendix A. Overlapping grids avoid this since they do not require either the inner or outer mesh to be twisted. The meshes for the noncircular casings were generated using the same approach as in the preceding examples. The principal benefit of using the Schwarz approach, in this case, is that the mesh lines for the outer part of the domain remain vertical (in contrast to the monodomain meshes for the helical pipe, where the mesh is twisted axially). With vertical mesh lines, it is possible to readily accommodate the vertical (axially aligned) corners in the outer casing. The mesh for the outer part of the domain has $E = 13{,}088$ ($E_f = 7648$ and $E_s = 5440$), and $E = 2560$ for the interior mesh, for $Re_D = 5300$. For $Re_D = 10,000$, the mesh was refined to increase the element count by roughly a factor of 2 in each direction. As a result, the exterior mesh has $E = 107,776$ ($E_f = 64,256$ and $E_s = 43520$), and $E = 15,792$ for the interior mesh.

We note that the spectral element mesh with $E = 107,776$ elements was also used with $Re_D = 5300$ to ensure that the Nusselt number obtained from the mesh with $E = 13{,}088$ elements was spatially converged (i.e., grid-independent).

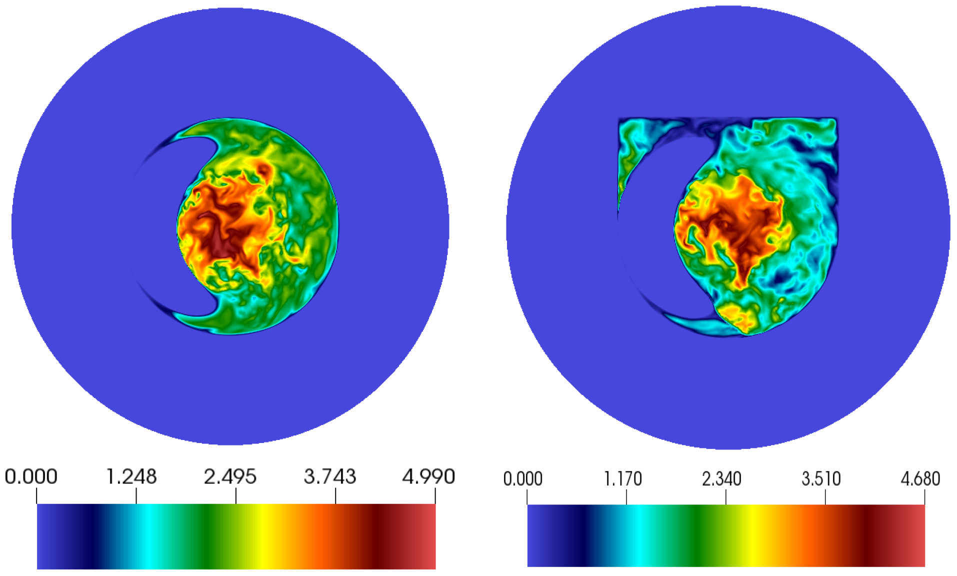


Figure 9.11: Velocity magnitude contours for the optimal-pitch wire-coil insert in (left) circular pipe, and (right) noncircular pipe.

As before, the flow was initialized at a low polynomial order ($N = 5$) without any corrector iterations ($Q = 0$ and $m = 1$). The polynomial order N was gradually increased to ensure that the calculation was spatially converged. Both calculations, $Re_D = 5300$ and $Re_D = 10,000$, were spatially converged at $N = 9$, and used $Q = 3$ with $m = 3$ for high-order temporal accuracy of the interdomain boundary data. These calculations indicate that $Nu \approx 205$ for $Re_D = 5300$, and $Nu \approx 320$ for $Re_D = 10,000$, which corresponds to an 11% increase for $Re_D = 5300$ and a 20% increase for $Re_D = 10,000$, with respect to the wire-coil insert in circular pipe. Figure 9.12 and Fig. 9.13 show slice view of the velocity magnitude and temperatures, respectively, for flow through noncircular pipe with wire-coil insert at $Re_D = 5300$ and $Re_D = 10,000$.

A potential limitation of the current approach is that we model a single pitch of the helically-periodic wire-coil insert. While calculations done for circular pipes indicate that this approach accurately captures the heat transfer properties of the flow, we will consider more than one pitch of the wire-coil insert in future work to ensure that the estimates of Nu are independent of the number of pitches of the wire-coil.

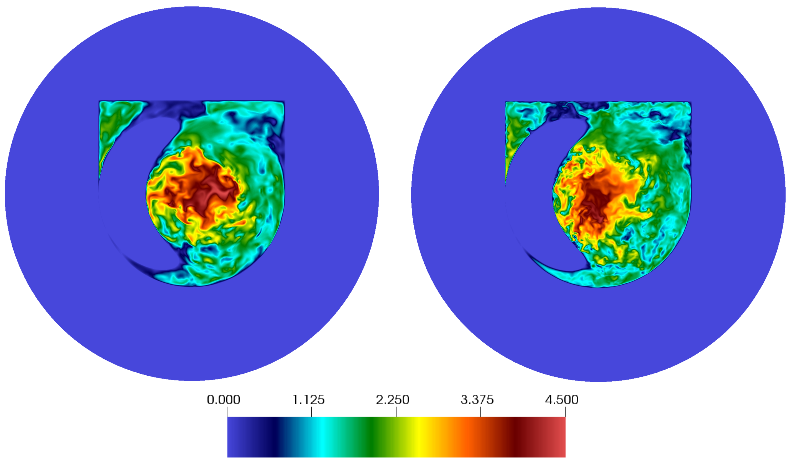


Figure 9.12: Velocity magnitude contours for the optimal-pitch wire-coil insert in noncircular pipe at (left) $Re_D = 5300$ and (right) $Re_D = 10,000$.

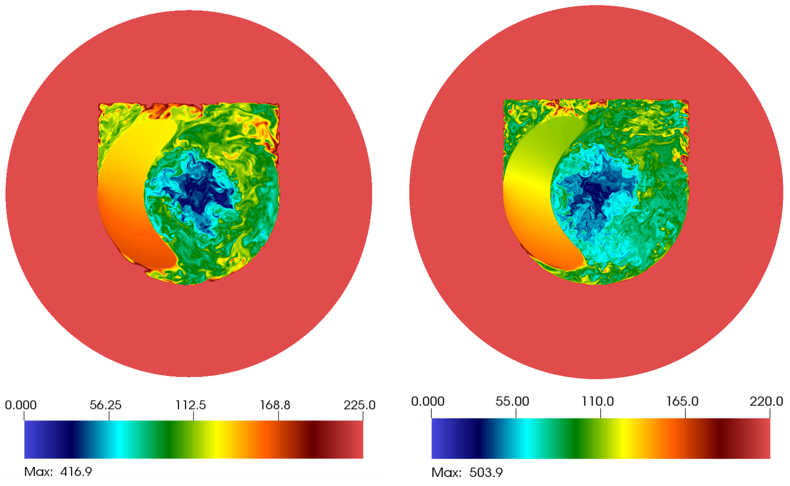


Figure 9.13: Temperature contours for the optimal-pitch wire-coil insert in noncircular pipe at (left) $Re_D = 5300$ and (right) $Re_D = 10,000$. The Nusselt number for $Re_D = 5300$ and $Re_D = 10,000$ is 205 and 320, respectively, based on the calculations done using the Schwarz-SEM framework.

9.4 Rotating nonspherical Particles

In this section, we consider simulations of nonspherical particles under rotation that are performed in collaboration with scientists at Utah State University and UIUC.
Particle-laden flows are common to engineering systems such as environmental flows (sediment transport in rivers), industrial systems (chemical processing and oil pipelines), and combustion processes, etc. Particles in these systems could rotate due to collision with other particles, or due to high vorticity and mean shear of the fluid. Thus, it is essential to understand how the shape and orientation of a rotating particle impact the flow around it, and consequently, the forces acting on it.

Through his experiments, Best [130] observed that particle rotation led to turbulence enhancement, even at moderate Reynolds number, which emphasized the importance of accounting for particle rotation for modeling particle-laden flow. Dobson et al., Kim et al., and Poon et al. have done high-resolution LES and DNS calculations to study flow around a rotating sphere for a range of Reynolds number and rotation rates [131, 132, 133]. Typically, numerical studies have modeled particles as spheres, as the rotation of a sphere can be modeled with a static mesh by imposing Dirichlet velocity on the surface of the sphere.

Modeling a rotating nonspherical particle, however, either requires remeshing in case of conformal meshes (as the mesh distorts beyond the limit of computational utility), or a nonconforming mesh method. Here, we use the Schwarz-SEM framework to model a rotating particle using a rotating inner mesh for the particle, which overlaps with a static mesh for the background. This approach allows us to compare our results with the existing literature [131, 132], which helps demonstrate the effectiveness of the Schwarz-SEM framework for modeling rotating/moving objects. Next, we also consider the impact of change in the shape of particles on the flow characteristics. These preliminary analyses will give us insight into a phenomenon that has not been studied until now. The long-term goal of this study is to model a large number of rotating particles to understand their interaction with each other and their impact on the flow.

9.4.1 Rotating spherical particle

As a first step towards validating the Schwarz-SEM framework for moving meshes, we model a rotating sphere at Reynolds number, $Re_p = U_\infty D/\nu = 300$, where U_∞ is the freestream velocity, D is the particle diameter, and ν is the kinematic viscosity of the fluid. Following Dobson, the particle rotation is set normal to the flow direction, and the nondimensional rotation rate is set to $\Omega^* = 0.5\tilde{\Omega}D/U_\infty$. Here, $\tilde{\Omega}$ is the angular velocity of the sphere. Figure 9.14 shows the schematic that indicates the direction of rotation of the particle with respect to the direction of the flow. In Fig. 9.14, a_j represents the length of the principal axis of the sphere in jth direction, and $a_x = a_y = a_z$ for a sphere.

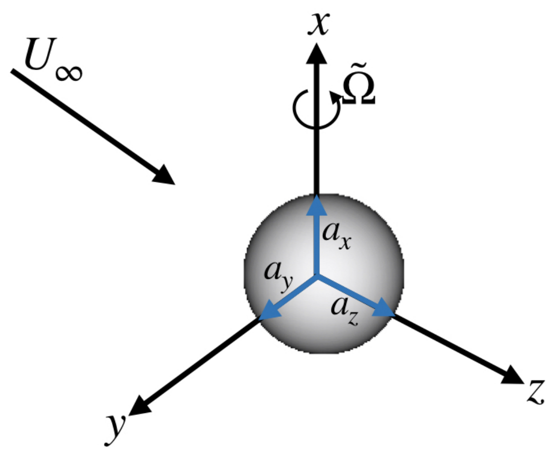


Figure 9.14: Schematic showing the direction of the rotation of particle with respect to the flow.

Figure 9.15 shows a slice view of the two overlapping spectral element meshes that are used to model the flow around a rotating sphere. A moving interior mesh with $E = 24,576$ is used for the rotating particle, and a static exterior mesh with $E = 41,216$ is used for the background flow. The particle is centered at the origin, and a uniform inflow is imposed at $z = -10D$ with the outflow at $z = 24D$. Periodic boundary conditions are imposed on the other two directions, with periodic surfaces $20D$ apart in each direction. In contrast to our problem setup, Dobson's calculations were based on a Fourier-Chebyshev spectral collocation method in spherical coordinates. The maximum domain extent in the radial direction was set to $22.5D$, and a mesh with $121 \times 100 \times 32$ points in the $r \times \theta \times \phi$ direction was used for $\Omega^* \leq 1.25$. For $\Omega^* > 1.25$, a mesh with $120 \times 100 \times 64$ points was used. Here, θ corresponds to the direction of rotation, and ϕ is the azimuthal direction.

Figure 9.16 shows contours of λ_2 vortices [134] colored by velocity magnitude (top) and a velocity magnitude plot for $Re_p = 300$ with $\Omega^* = 1$. This calculation was done with $Q = 3$ and $m = 3$ for the Schwarz iterations, and the results were spatially converged at $N = 7$. Figure 9.16 shows vortex shedding in the wake of the sphere, with the flow separating around the point of least relative velocity between the sphere and the background flow. The sphere rotation makes the vortex shedding asymmetrical.

Figure 9.17 shows a slice view of the flow near the rotating sphere. We can observe in the vector plots shown in Fig. 9.17 that the rotating sphere pulls fluid from the leeward side to the windward side[1], which is similar to the phenomena of added-mass where fast-moving/accelerating particles are known to carry additional mass of fluid around them [135]. Consequently, a shear layer forms between the opposing flows of the fluid pulled by the rotating sphere and the background flow (U_∞) along the z-axis. With the shear

[1]The windward side refers to the side of the sphere which is facing the inflow, and leeward side refers to the side that does not directly interact with the background flow.

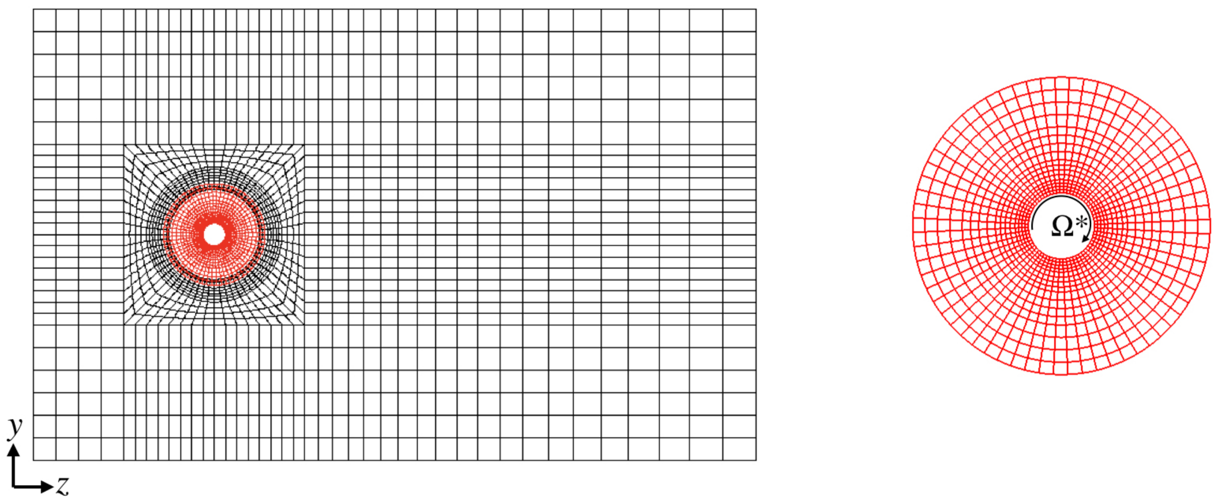


Figure 9.15: Slice view of the (left) overlapping spectral element meshes for the rotating particle and the background mesh, and (right) spectral element mesh around the particle.

layer instability growing behind the sphere, we see vortex shedding about one sphere diameter downstream of the sphere. This vortex shedding is also apparent in Fig. 9.16. Due to the fluid being pulled from the leeward to the windward side and its interaction with the background flow, a high pressure region forms at the bottom of the sphere. This relatively high pressure region at the bottom along with a relatively low pressure region on the top of the sphere leads to lift force (along the y direction) on the sphere. Similarly, a high pressure region on the windward side of the sphere, due to the background flow, and a low pressure region on the leeward side, due to the rotation of the sphere, results in the form drag (along z direction). The flow structures that we see in Fig. 9.16 and Fig. 9.17 are similar to those reported by Dobson [131] and others [132, 133].

We note that for the Schwarz-SEM framework, the boundary conditions for pressure are Neumann on surfaces that are Dirichlet for velocity (Chapter 2). Since we do not impose Dirichlet condition on pressure at the interdomain boundaries, the pressure is not continuous across overlapping subdomains. However, this pressure discontinuity across overlapping grids is acceptable because it is the pressure-gradient which needs to be continuous in order for the solution of (2.26) to be consistent. Consequently, we only show the pressure distribution for the inner mesh in the following sections.

Table 9.4 lists the lift coefficient (C_{Ly}), drag coefficient (C_{Dz}) and Strouhal number (St) determined by the Schwarz-SEM framework for the sphere at $Re_p = 300$ rotating at $\Omega^* = 1$ and 2, and compares these

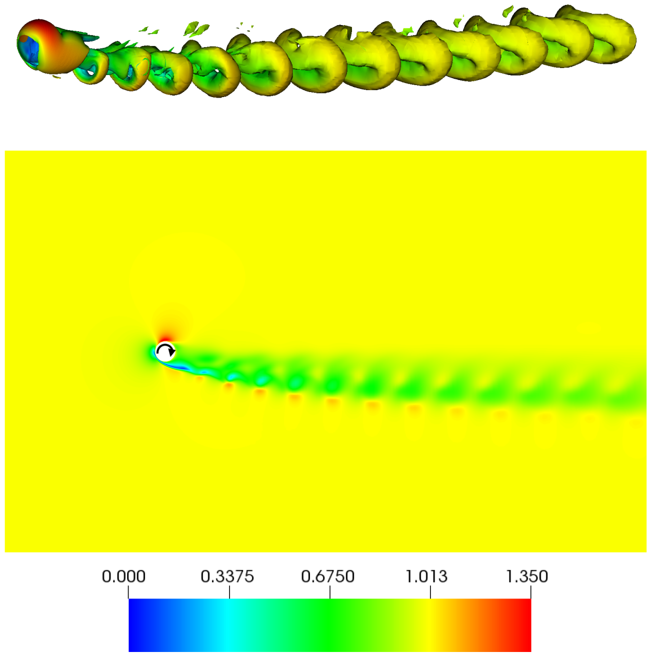


Figure 9.16: (top) Isosurfaces of λ_2 colored by velocity magnitude, and (bottom) velocity magnitude contours for flow over the rotating sphere.

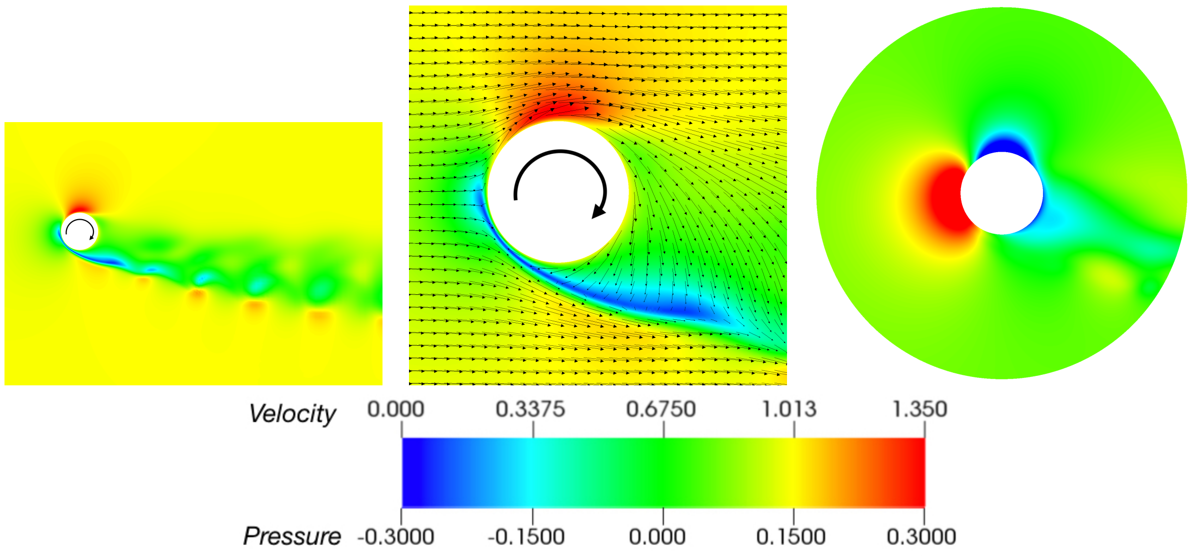


Figure 9.17: (left) Velocity magnitude contours, (center) close up of the velocity vector field near the sphere surface, and (right) pressure contours for the rotating sphere.

results with Dobson et al. (and other literature for the case of $\Omega^* = 1$). The drag coefficient was computed as $C_{Dz} = 2F_z/(\rho U_\infty^2 A_p)$, where F_z is the force on the particle in the streamwise direction and $A_p = \pi D^2/4$ is the projected area of the sphere. The lift coefficient is computed similarly using the lift force. Table 9.4

shows that our results match the data of Dobson et al. to within 1.5%.

	C_{Ly}	C_{Dz}	St
$\Omega^* = 1$			
Schwarz-SEM	0.613	0.961	0.426
Dobson et al. [131]	0.610	0.961	0.423
Kim et al. [132]	0.596	0.931	0.424
Poon et al. [133]	0.605	0.964	0.427
$\Omega^* = 2$			
Schwarz-SEM	0.589	1.019	0.243
Dobson et al.	0.582	1.012	0.240

Table 9.4: Comparison of lift coefficient (C_{Ly}), drag coefficient (C_{Dz}), and Strouhal number (St) for the Schwarz-SEM calculations, with Dobson et al. and other existing literature.

9.4.2 Nonspherical particles

Here, we extend the results of the preceding section to the case of a rotating ellipsoid. We consider two different particle shapes, where we keep $a_x = a_z = 0.5D$ unchanged from the sphere in the previous section, but modify $a_y = 0.75D$ for one particle and $a_y = 0.25D$ for the other. As before, the axis of rotation is x, and the background flow is in the positive z direction. The meshes for these nonspherical (ellipsoid) particles have been generated by morphing the inner mesh generated for the spherical particle. This morphing of mesh is straightforward to effect since the surface of the particle is described as $x^2/a_x^2 + y^2/a_y^2 + z^2/a_z^2 = 1$. Figure 9.18 contrasts the two ellipsoids with the spherical geometry of the preceding example. The position of the particles shown in Fig. 9.18 corresponds to $\theta = 0°$, and θ increases from $0°$ to $360°$ as the particle rotates clockwise around the x-axis.

Figure 9.19 shows the λ_2 [134] for the three particles considered here. It is apparent from Fig. 9.19 that the structure of the vortices being shed behind the ellipsoids is different from the structures that we had observed for the sphere. The primary reason behind this difference is that unlike in the flow over the rotating sphere, a steady shear layer never develops behind the ellipsoid. The flow keeps attaching and separating as the ellipsoid rotates, which leads to multiple vortices being shed in its wake. The primary vortex shedding mechanism is the interaction of the mean background flow with the ellipsoid, which we had observed for the spherical particle as well. For the ellipsoids, there is also a secondary mechanism due to the asymmetry in shape. This mechanism will be clear through the discussion in the next section.

Since the shape of the particle has changed, it is essential to consider what area to use for computing the drag (and lift) coefficient, $C_D = 2F/(\rho U_\infty^2 A_p)$. For the case of drag in the streamwise direction, the choice of the frontal area is straightforward. However, for the lift coefficient (normal to the direction of the flow

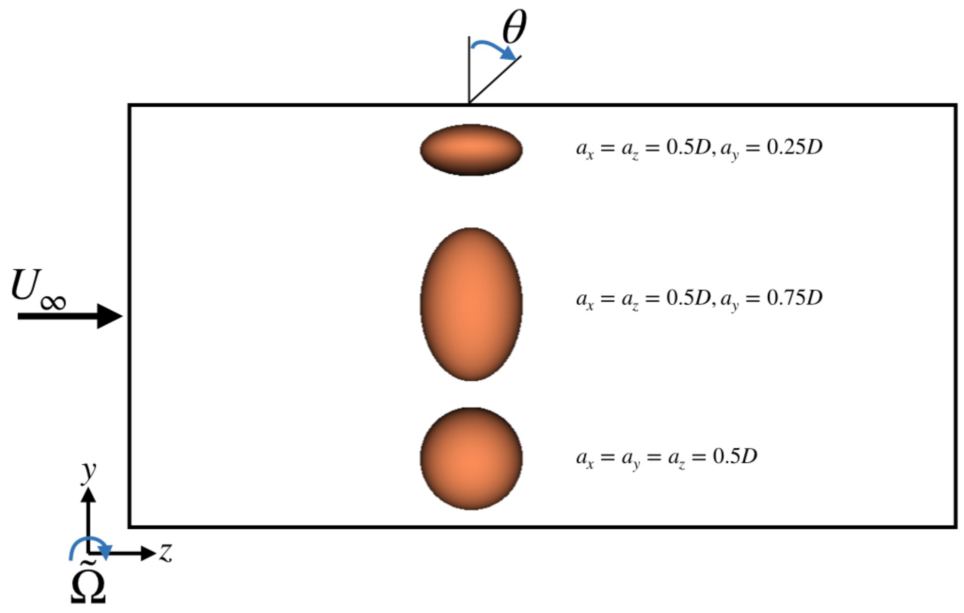
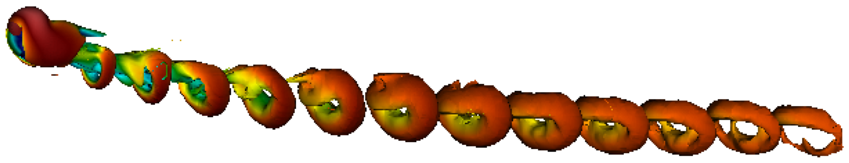


Figure 9.18: Schematic showing the direction of the rotation of particle with respect to the flow for different particles, and the three different particles considered here.

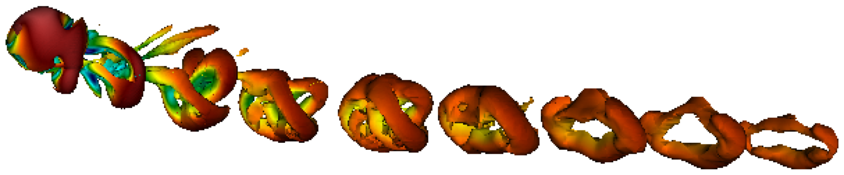
- y in Fig. 9.18), it is not clear whether the projected area normal to the direction of the flow is the best choice. Using the frontal area of the sphere (constant at $\pi D^2/4$) indicates that decreasing a_y increases the lift coefficient. However, using the actual frontal area of the ellipsoids (time-varying due to rotation) indicates that decreasing a_y decreases the lift coefficient. Thus, instead of comparing the drag and lift coefficients, we compare the nondimensionalized drag and lift forces to avoid any confusion and inconsistencies.

Figure 9.20(top) and (bottom) shows how the drag and lift, respectively, varies for the three particles with θ. The drag and lift forces on the sphere are almost constant in time because the flow is steady near the sphere surface. In contrast to the sphere, we observe that the drag and lift vary in time for the ellipsoids, and there is a phase-shift of 90° for the drag and lift time-series for the two ellipsoids. This phase-shift is expected because the major principal axis of the particle with $a_y = 0.75$ is the minor principal axis for the particle with $a_y = 0.25$. In Fig. 9.20, we have also indicated the θ at which the frontal area of the particles is maximum or minimum, using vertical golden-colored lines. An interesting observation from these results is that there is a phase difference between the θ of maximum (or minimum) frontal area and the θ at which the drag is maximum (or minimum).

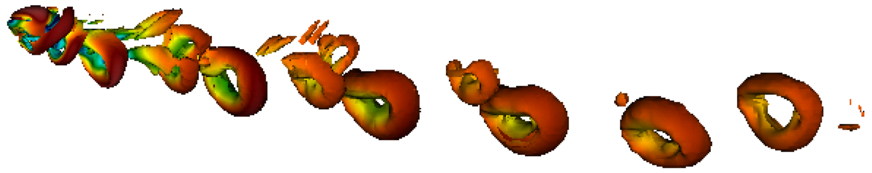
Table 9.5 lists the mean drag and lift forces on each particle for $Re_p = 300$ at $\Omega^* = 1$ along with the maximum and minimum drag and lift forces. As we can see, increasing a_y increases the mean drag on the particle. This increase is expected due to the change in the frontal area of the particle. However, we see that the lift force (F_{Ly}) has decreased for both the ellipsoids in comparison to the sphere. This decrease in



(a) $a_x = a_y = a_z = 0.5D$



(b) $a_x = a_z = 0.5D, a_y = 0.75D$



(c) $a_x = a_z = 0.5D, a_y = 0.25D$

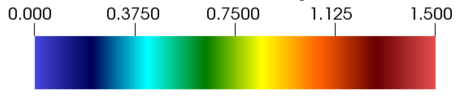


Figure 9.19: Comparison of the λ_2 vortices [134] for the three ellipsoids considered for this problem.

F_{Ly} is likely because of the repeated flow attachment and separation that occurs for the ellipsoids, which will be discussed in the next section.

	F_{Ly} $[F_{Ly,min}, F_{Ly,max}]$	F_{Dz} $[F_{Dz,min}, F_{Dz,max}]$
$a_x = a_y = a_z = 0.5D$	0.241 [0.234,0.247]	0.378 [0.367,0.387]
$a_x = a_z = 0.5D, a_y = 0.75D$	0.222 [0.046,0.515]	0.426 [0.135,0.675]
$a_x = a_z = 0.5D, a_y = 0.25D$	0.212 [0.042,0.441]	0.330 [0.085,0.563]

Table 9.5: Comparison of lift (F_{Ly}) and drag (F_{Dz}) for the Schwarz-SEM calculations.

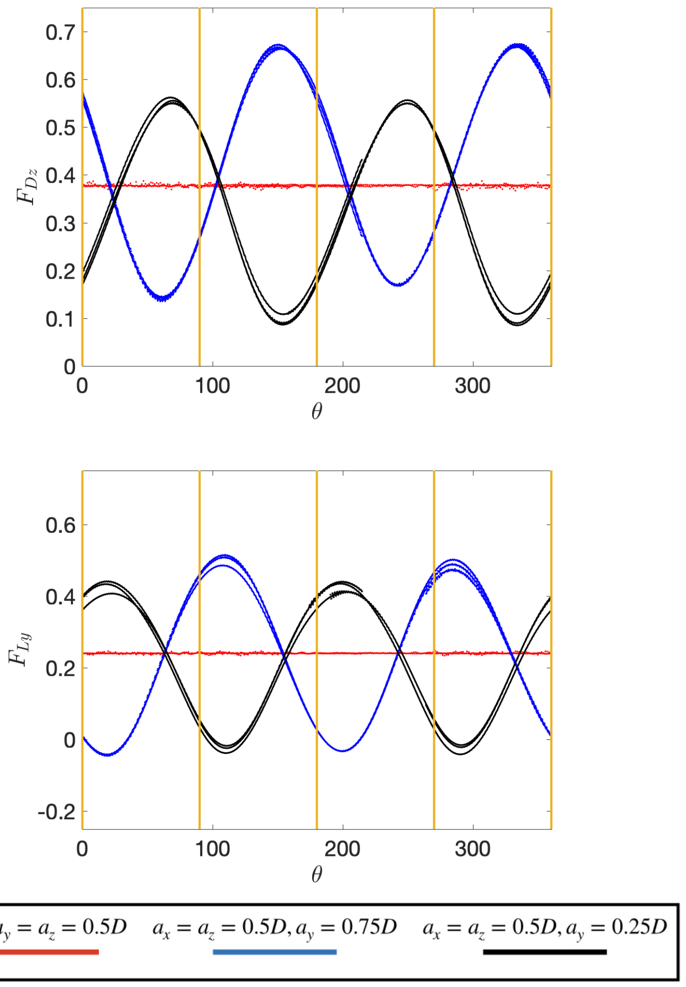


Figure 9.20: Comparison of the (top) drag and (bottom) lift forces on different particles. Golden colored vertical lines indicate θ at which the projected area of the ellipsoids is maximum or minimum.

Discussion of results

Figures 9.21 and 9.22 show the drag and lift variation, respectively, with θ for the two ellipsoids. It is clear that the θ at which the ellipsoids experiences maximum and minimum streamwise drag (F_{Dz}) and transverse lift (F_{Dy}), do not coincide with the θ at which the frontal area is maximum or minimum. We also note that the two particles exhibit similar characteristics for the evolution of drag and lift, with a phase-shift of 90°. Additionally, we observe that due to the symmetry in the shape of the particle, the drag and lift time-series repeat after 180°. In this section, we examine the velocity magnitude and pressure for flow over the ellipsoid with $a_y = 0.75$ to understand how the shape of the ellipsoid leads to this phase-shift between maximum and minimum drag (and lift) with maximum and minimum frontal area.

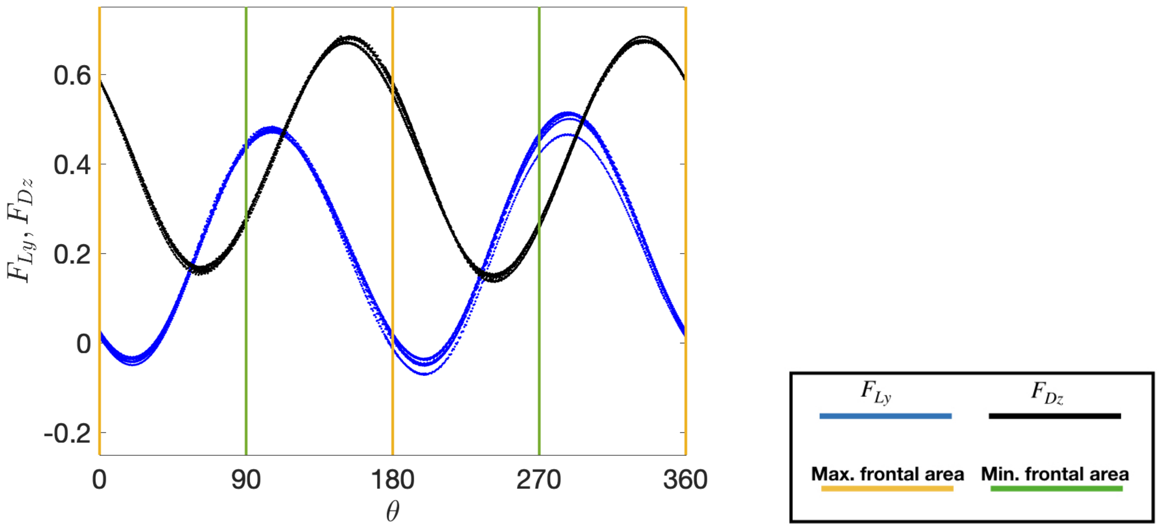


Figure 9.21: Comparison of the drag and lift with rotation angle (θ) for the ellipsoid with $a_y = 0.75$.

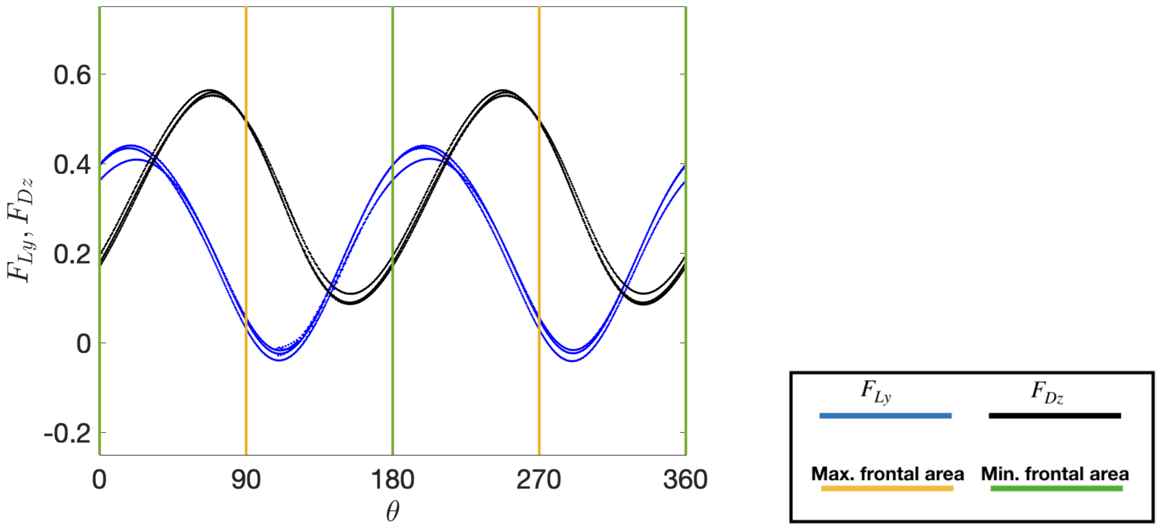


Figure 9.22: Comparison of the drag and lift with rotation angle (θ) for the ellipsoid with $a_y = 0.25$.

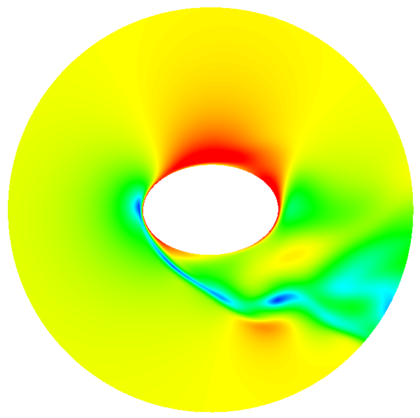
Maximum streamwise drag on the ellipsoid with $\mathbf{a_y = 0.75}$

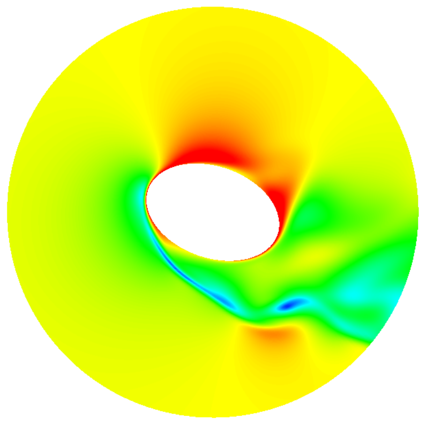
Figures 9.23 and 9.24 show the variation of velocity-magnitude and pressure near the sphere, for different angles of rotation θ (measured clockwise from the axis normal to the flow, as indicated in Fig. 9.18). Figure 9.23(f) corresponds to the instance of maximum drag and Fig. 9.23(i) corresponds to the instance of maximum frontal area. In order to understand why there is a phase-difference between the two, we look at the flow-field around the ellipsoid as it rotates from $\theta = 90°$ to $180°$, as shown in Fig. 9.23,9.24(a)-(i).

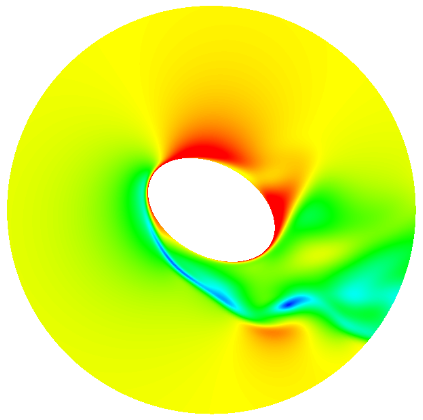
As the ellipsoid is rotating past $\theta = 90°$, a high pressure zone forms behind it (Fig. 9.23(a),9.24(a)). This high pressure zone forms due to the increase in the frontal area of the particle with respect to the background flow. Interestingly, this zone forms behind the particle and is not attached to the particle itself ($\theta = 90° - 140°$). As the particle rotates, it pulls fluid with it from the leeward side to the windward side, which leads to a low pressure (high velocity) region on the leeward side surface of the ellipsoid. With particle rotation, the high velocity zone stretches across the surface of the particle, and a low pressure zone grows on the leeward side. Though, once $\theta > 153°$, the velocity magnitude starts to drop on the leeward side of the particle. This drop in the velocity magnitude is due to an increase in blockage (frontal area) by the ellipsoid, which results in flow separation on its surface. Consequently, the pressure starts to increase on the surface of the ellipsoid as the particle rotates past $\theta = 153°$.

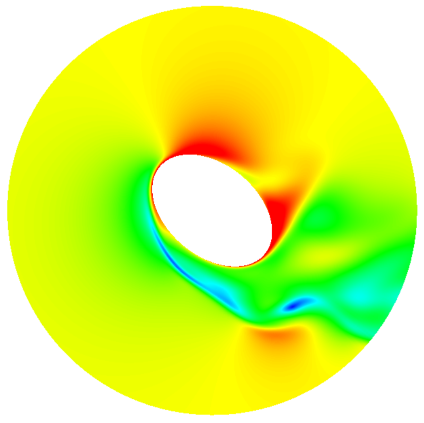
On the windward side, a high pressure zone is created at the surface of the ellipsoid due to its blockage effect. This high pressure zone is further extended on the particle surface because of the interaction of the background flow with the fluid being pulled from the leeward side (see Fig. 9.25).

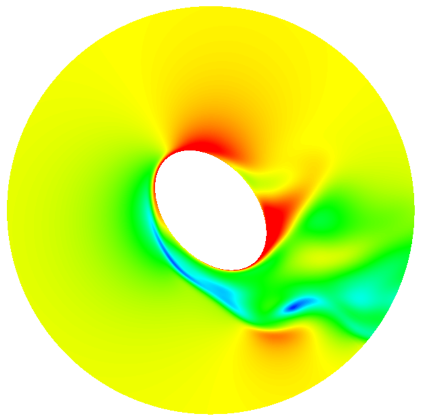
Typically, the drag on a nonrotating particle is maximum at the orientation at which the frontal area is maximum ($\theta = 180°$) because the maximum frontal area results in maximum pressure difference in the streamwise direction. Interestingly, the rotation of the particle changes this dynamic. As the rotating particle tends to pull fluid along with it (as previously described), the added-mass of the fluid results in the shift of the optimal pressure difference (in the z-direction) scenario from $\theta = 180°$ to $\theta \approx 153°$ (Fig. 9.24(i)).

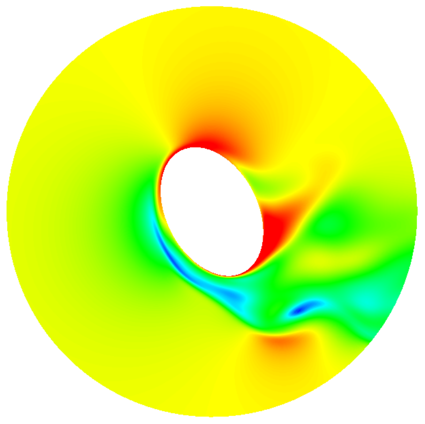

(a) $\theta \approx 90°$

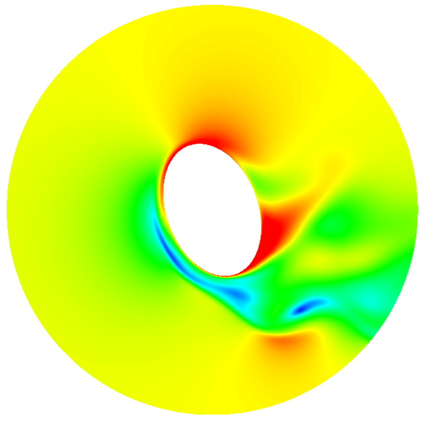

(b) $\theta \approx 110°$

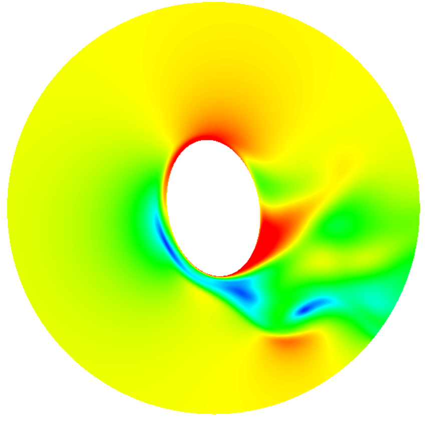

(c) $\theta \approx 120°$

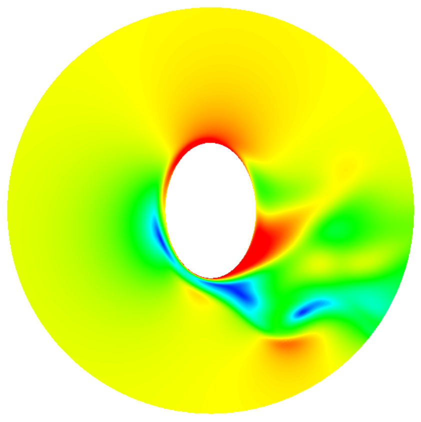

(d) $\theta \approx 130°$


(e) $\theta \approx 140°$


(f) $\theta \approx 153°$(maximum drag)


(g) $\theta \approx 160°$


(h) $\theta \approx 170°$


(i) $\theta \approx 180°$

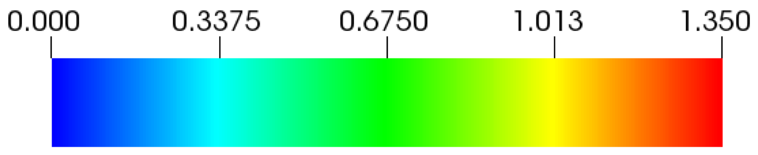


Figure 9.23: Slice view of the velocity magnitude contours at different θ for the nonspherical particle with $a_y = 0.75$.

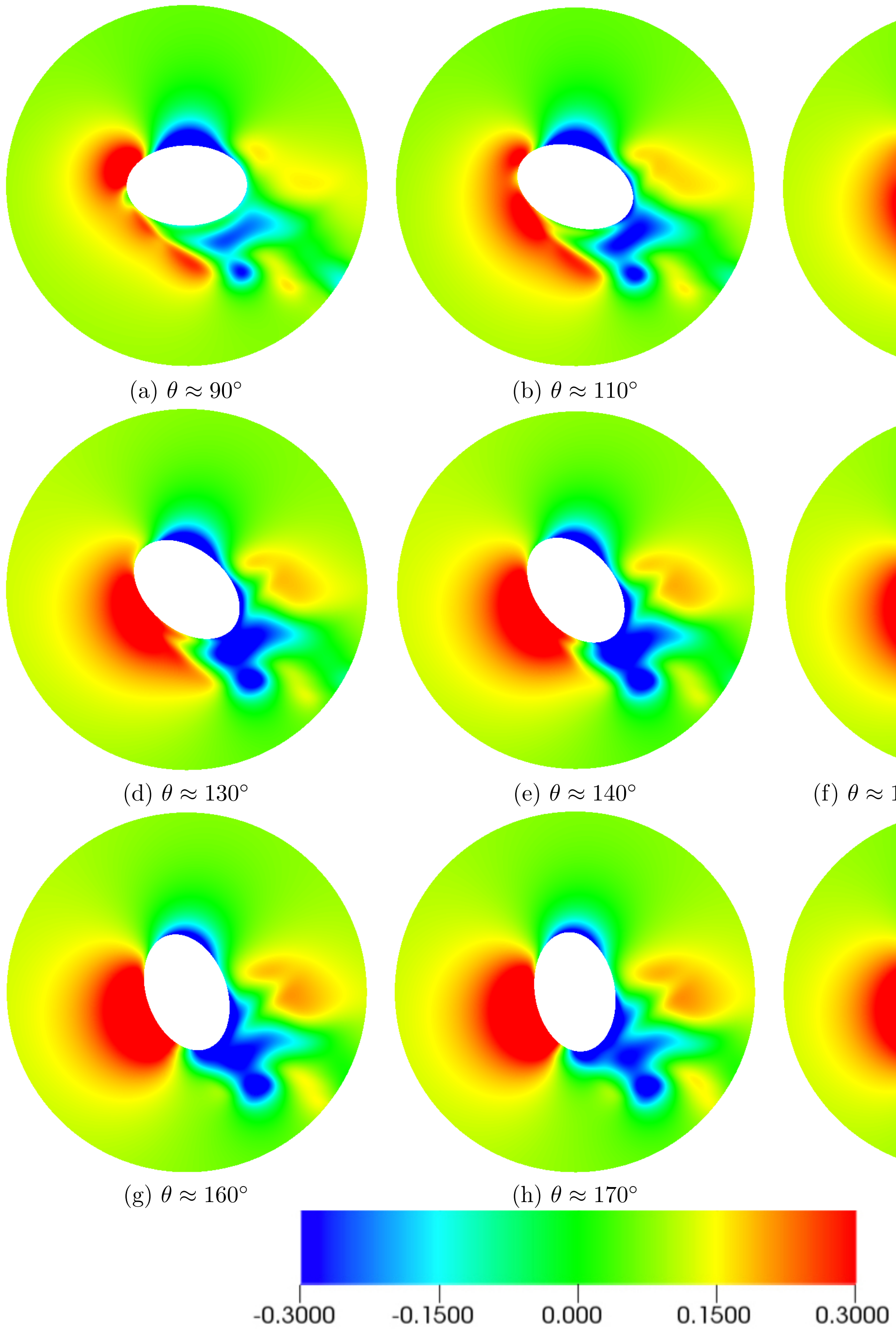
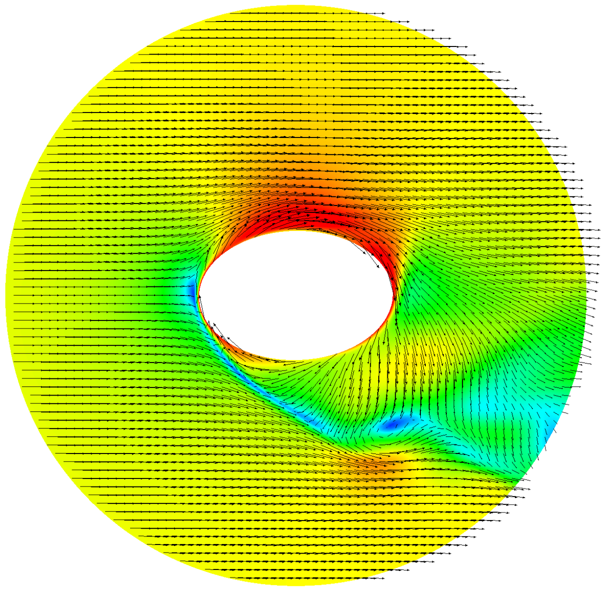
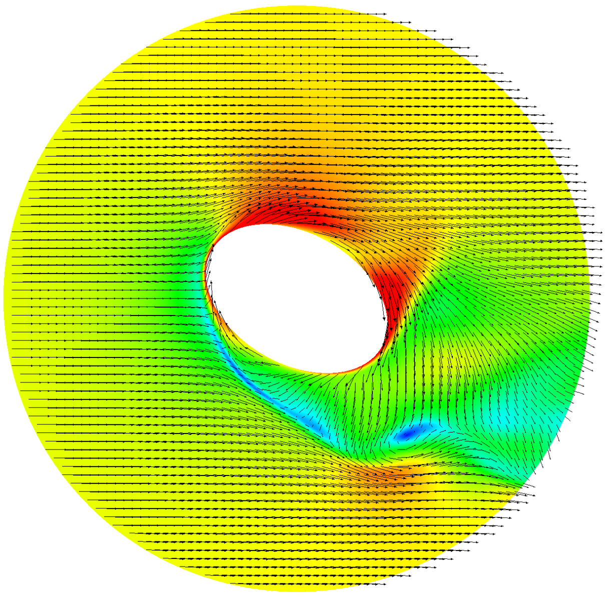
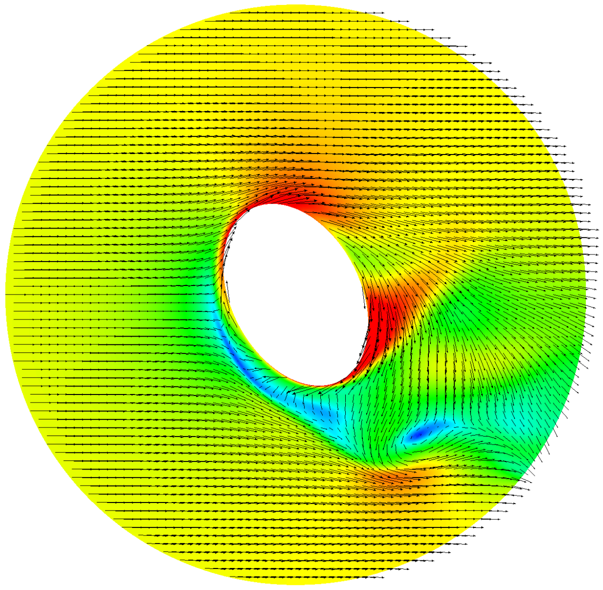
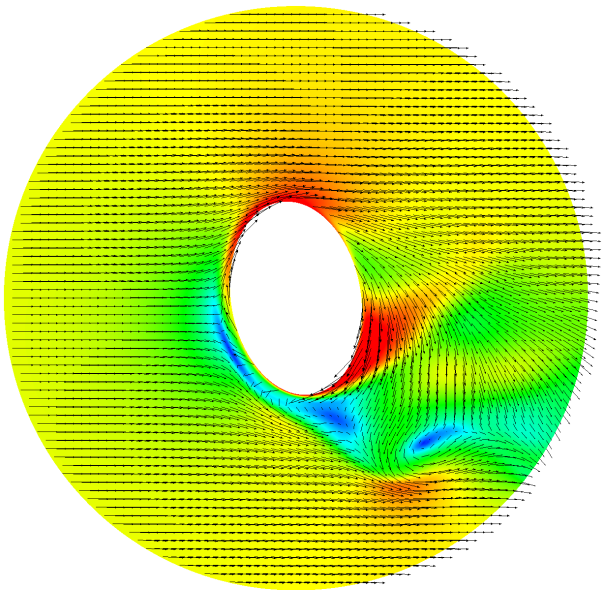


Figure 9.24: Slice view of the pressure contours at different θ for the nonspherical particle with $a_y = 0.75$.


(a) $\theta \approx 90°$


(b) $\theta \approx 120°$


(c) $\theta \approx 153°$(maximum drag)


(d) $\theta \approx 170°$

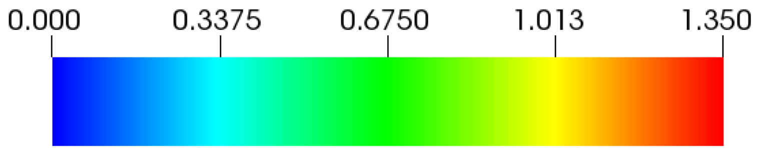


Figure 9.25: Slice view of the velocity magnitude contours with vectors indicating the direction of flow for different θ.

Minimum streamwise drag on ellipsoid with $\mathbf{a_y = 0.75}$

Here, we continue the discussion from why the maximum streamwise drag has a phase difference with the maximum frontal area, to understand the phase difference between minimum drag and minimum frontal area. Figure 9.26-9.28 show the velocity magnitude, pressure, and vector plots, respectively, for the ellipsoid with $a_y = 0.75$ for θ between 180° and 270°.

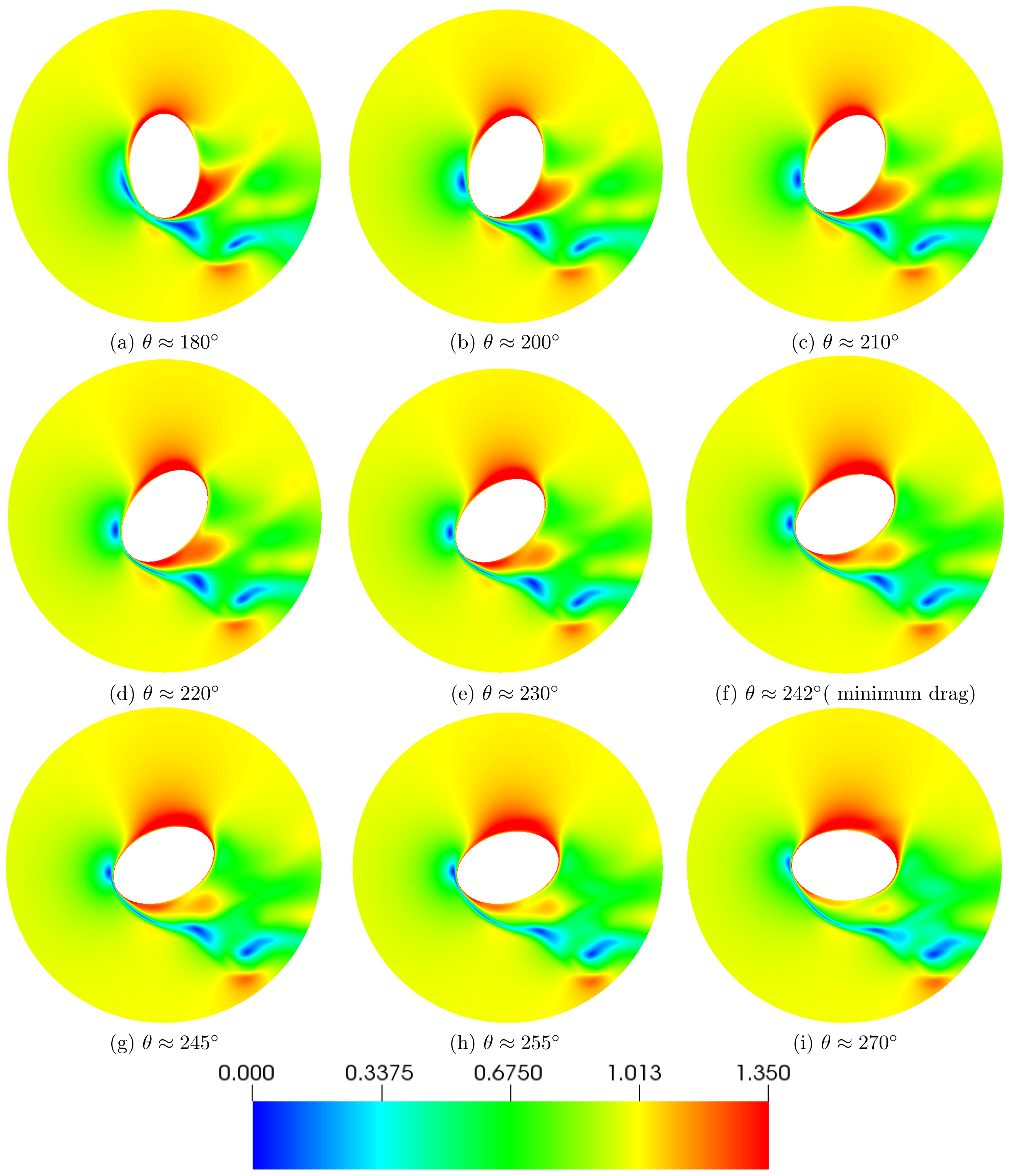
As the ellipsoid rotates beyond $\theta = 180°$, the relatively high pressure zone continues to grow on the leeward side of the ellipsoid, especially due to the momentum of the streamwise flow taking it past the ellipsoid ($\theta = 180°$ to 230° in Fig. 9.26 and Fig. 9.27). Additionally, with increase in θ from 180° to 230°, the particle becomes relatively more streamlined with the flow, which results in decrease in size of the high pressure zone on the windward side (Fig. 9.27(a)-(e)). Consequently, at about $\theta = 242°$ the pressure difference in the z-direction is minimum, which resulting in minimizing the drag on the particle; even though the frontal/projected area of the particle is not minimum.

Traditionally, the particle should experience the least drag at $\theta = 270°$, as the frontal area is minimum, and the flow around the particle is the most streamlined. Here, we observe that as the particle becomes streamlined with the flow for $\theta > 242°$, the fluid being pulled from the leeward side interacts with the background flow, which increases the pressure on the windward side. Consequently, the drag is not minimum at $\theta = 270°$. We can see the details of this interaction in Fig. 9.28(c) and 9.28(d), and in Fig. 9.26(f)-(i) and Fig. 9.27(f)-(i).

We also observed in these figures that as the particle rotates from $\theta = 180°$ to $\theta = 270°$ and the position becomes more streamlined with respect to the background flow, the flow re-attaches to the particle and a low pressure region develops on the top along with a high pressure region on the bottom[2]. This observation will be important in understanding the phase difference between the maximum lift force with the minimum frontal area.

The key result that comes forward from the analysis of streamwise drag is that the impact of rotation primarily manifests through the fluid pulled by the particle, which results in decreasing pressure on the leeward side or increasing pressure at the windward side, or sometimes both, depending on the orientation of the particle. Due to the asymmetry in the shape of the ellipsoid, a steady shear layer never develops near the particle, which is observed for the rotating sphere.

[2]Here, we use top and bottom to represent the surface of the particle in the transverse direction, i.e., y as indicated in Fig. 9.18.

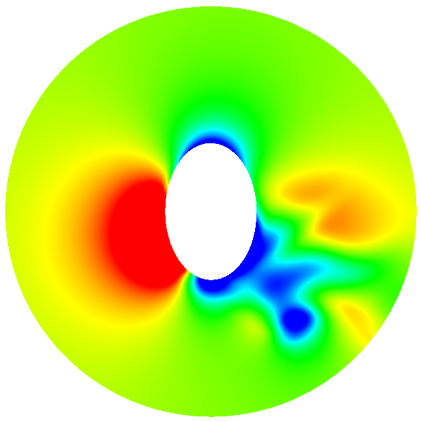


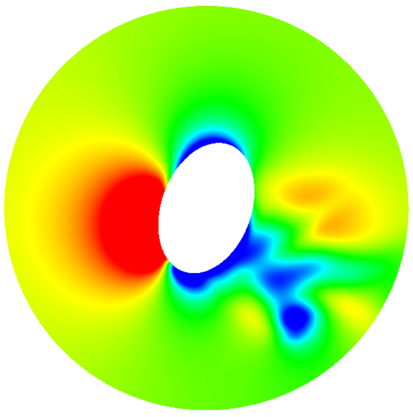
(a) $\theta \approx 180°$ (b) $\theta \approx 200°$ (c) $\theta \approx 210°$

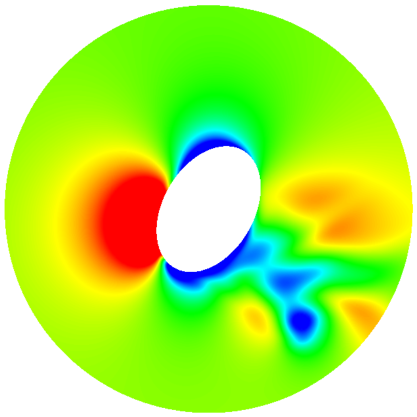
(d) $\theta \approx 220°$ (e) $\theta \approx 230°$ (f) $\theta \approx 242°$(minimum drag)

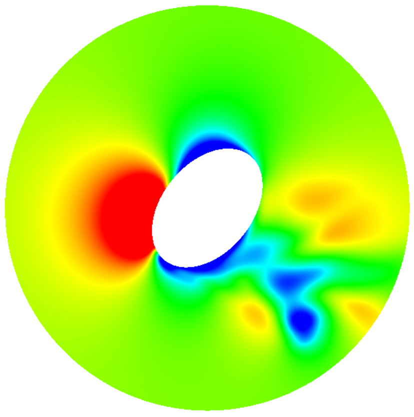
(g) $\theta \approx 245°$ (h) $\theta \approx 255°$ (i) $\theta \approx 270°$

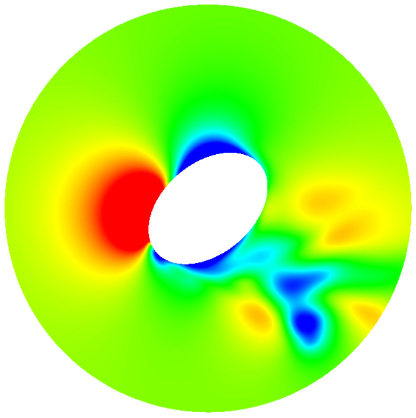
Figure 9.26: Slice view of the velocity magnitude contours at different θ for the nonspherical particle with $a_y = 0.75$.

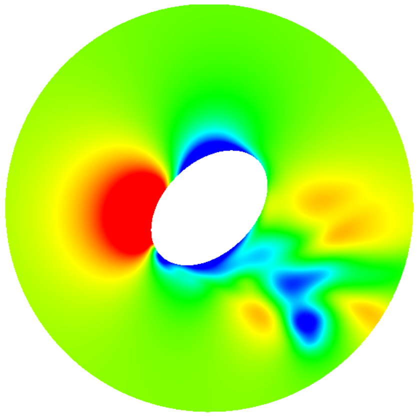

(a) $\theta \approx 180°$

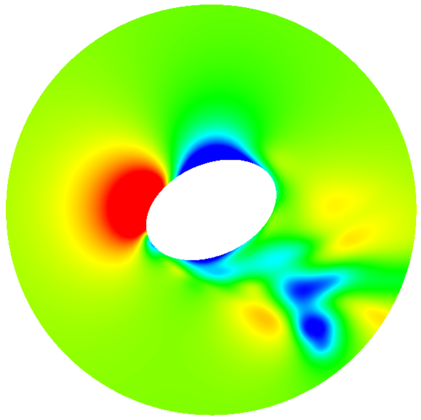

(b) $\theta \approx 200°$

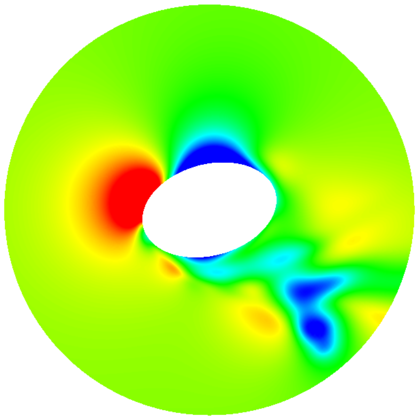

(c) $\theta \approx 210°$

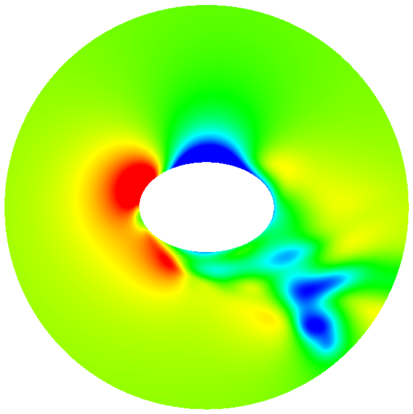

(d) $\theta \approx 220°$


(e) $\theta \approx 230°$


(f) $\theta \approx 242°$(minimum drag)


(g) $\theta \approx 245°$


(h) $\theta \approx 255°$


(i) $\theta \approx 270°$

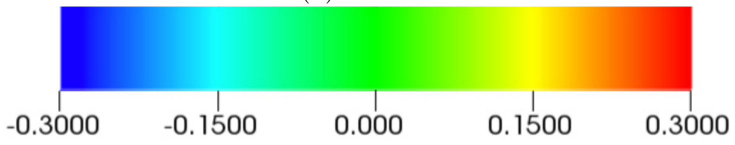
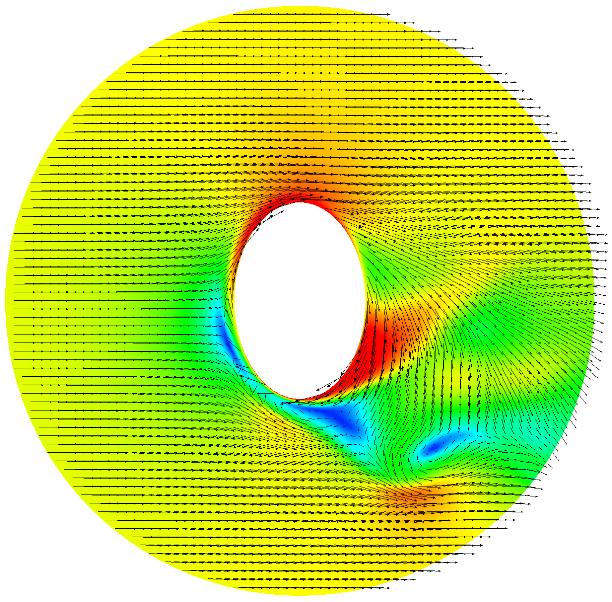
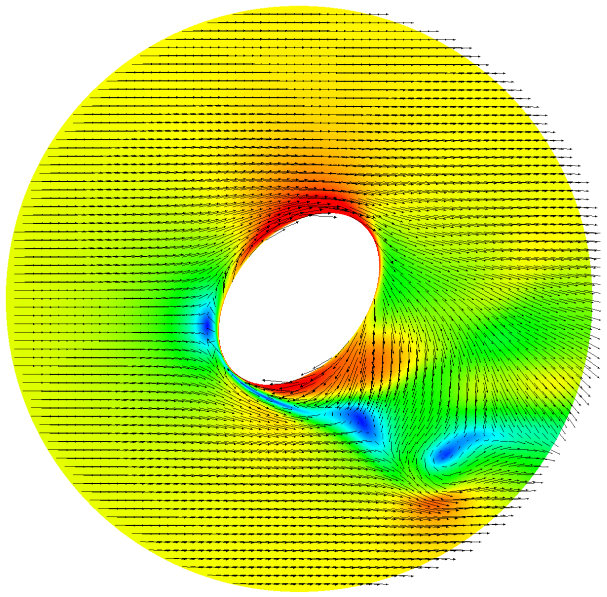


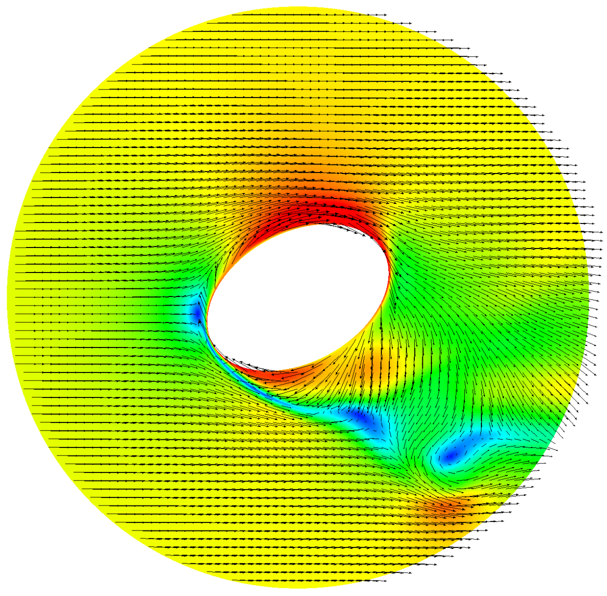
Figure 9.27: Slice view of the pressure contours at different θ for the nonspherical particle with $a_y = 0.75$.



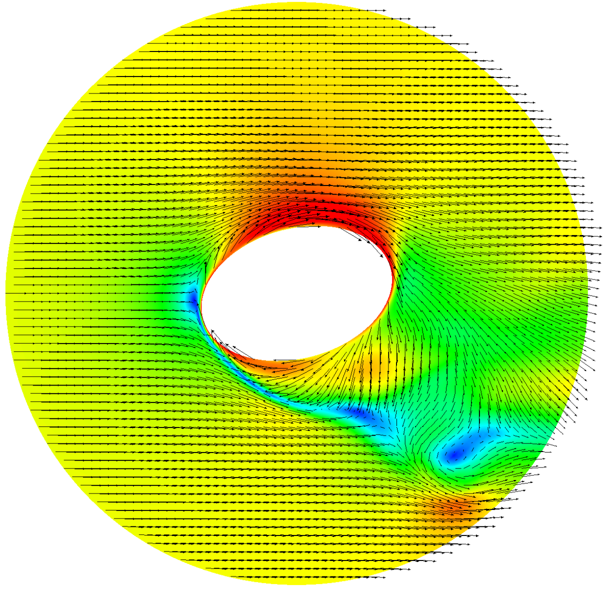
(a) $\theta \approx 180°$



(b) $\theta \approx 220°$



(c) $\theta \approx 242°$(minimum drag)



(d) $\theta \approx 255°$

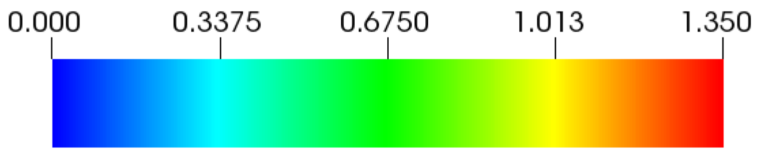


Figure 9.28: Slice view of the velocity magnitude contours with vectors indicating the direction of flow for different θ.

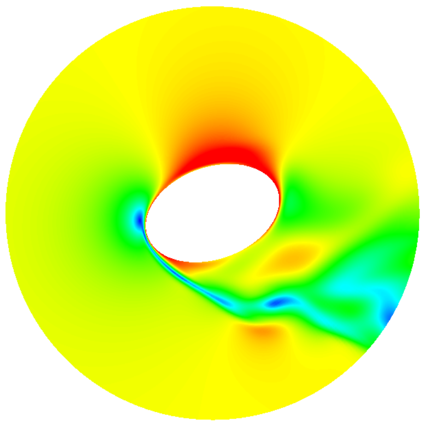
Maximum and minimum lift on the ellipsoid with $\mathbf{a_y = 0.75}$

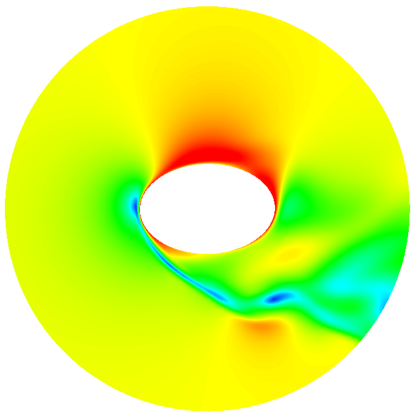
Building on our discussion on the streamwise drag, we can understand the mechanics that lead to maximum and minimum lift for the ellipsoid with $a_y = 0.75$. Figure 9.29-9.31 show the velocity magnitude, pressure, and vector plots, respectively, for the flow over the spheroid at different θ.

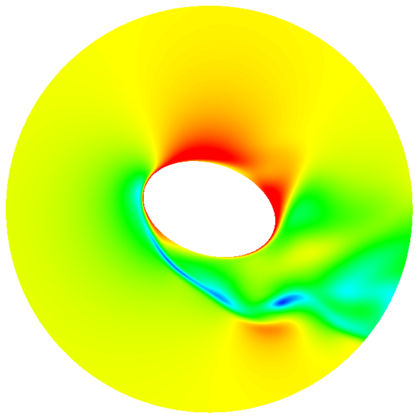
Figure 9.30(b) shows the snapshot at which the front area of the particle is minimum. As the particle is rotating past $\theta = 90°$, the flow is attached to the top of the particle, which leads to a low pressure zone. With particle rotation, the size of the low pressure zone at the top increases. The reason is the same as that discussed in the maximum-drag section. On the bottom of the particle, the fluid pulled due to the particle's rotation starts to interact with the background flow (as discussed earlier), which leads to an increase in the pressure. Due to this increasing pressure difference between the top and bottom of the particle, the lift is maximum at $\theta \approx 108°$ (also at $\theta = 288°$), as shown in Fig. 9.29,9.30(c).

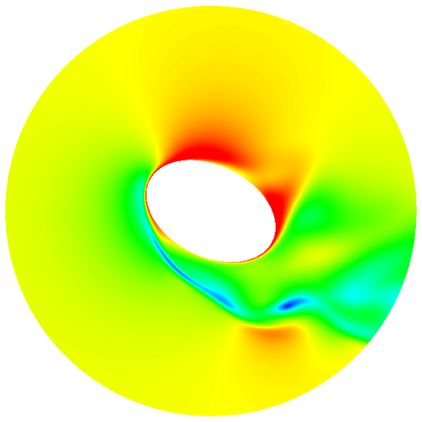
As the particle rotates further, adverse pressure-gradient on the leeward side causes the flow to separate, which increases the pressure on the leeward side. The fluid moving with the particle continues to accelerate at the bottom of the ellipsoid, whereas, the background flow continues to cause a low pressure zone on the top of the sphere ($\theta = 120° - 180°$). It is important to note here that the low pressure zone at the bottom of the particle is not due to the background flow, but due to the fluid being pulled as a consequence of particle rotation (Fig. 9.31(c)). As a result, the lift is minimized at the point when the entrained fluid has not yet started interacting with the background flow at the bottom of the particle, which happens at about $\theta \approx 200°$ (or $\theta \approx 20°$). As the particle rotates further, it becomes more aligned with the flow and the pressure increases at the bottom due to the interaction of the entrained flow with background flow. Consequently, the lift force on the particle starts increasing.

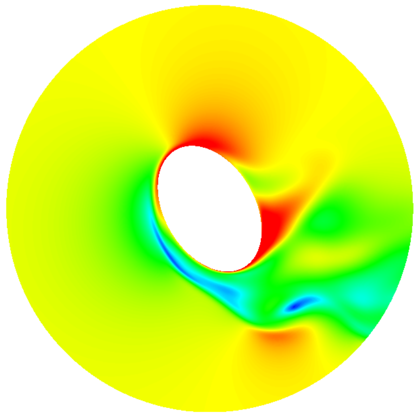
Similar to the discussion on streamwise drag, we observe that the θ at which the ellipsoid experiences maximum or minimum lift is significantly dependent on the interaction of the background flow and the fluid being pulled by the particle with its rotation. The discussion for the drag and lift forces on the particle with $a_y = 0.75$ extends to the particle with $a_y = 0.25$, where we observe similar flow characteristics with a phase-shift of $\theta = 90°$.

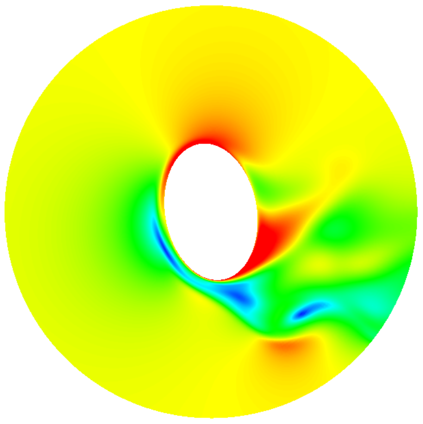

(a) $\theta \approx 70°$

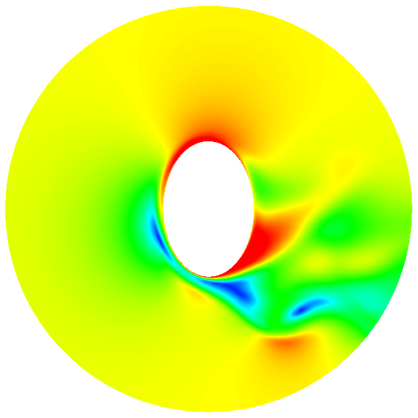

(b) $\theta \approx 90°$

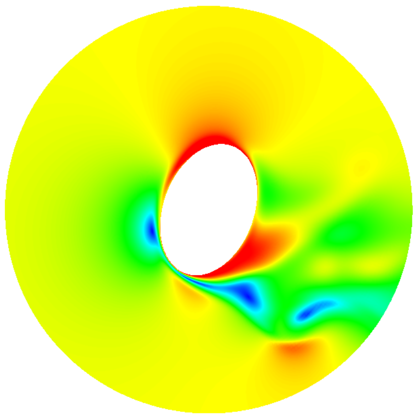

(c) $\theta \approx 108°$(maximum lift)

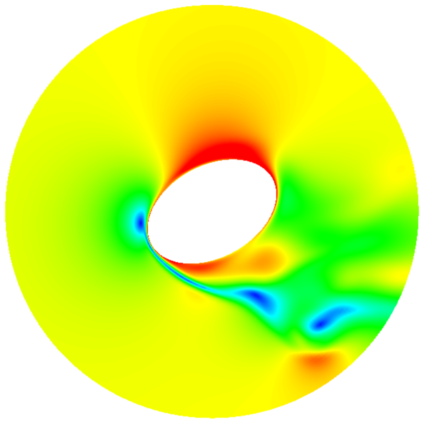

(d) $\theta \approx 120°$


(e) $\theta \approx 150°$


(f) $\theta \approx 170°$


(g) $\theta \approx 180°$


(h) $\theta \approx 200°$(minimum lift)


(i) $\theta \approx 225°$

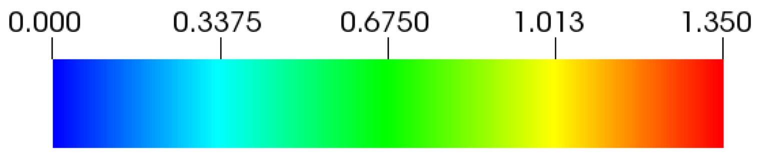
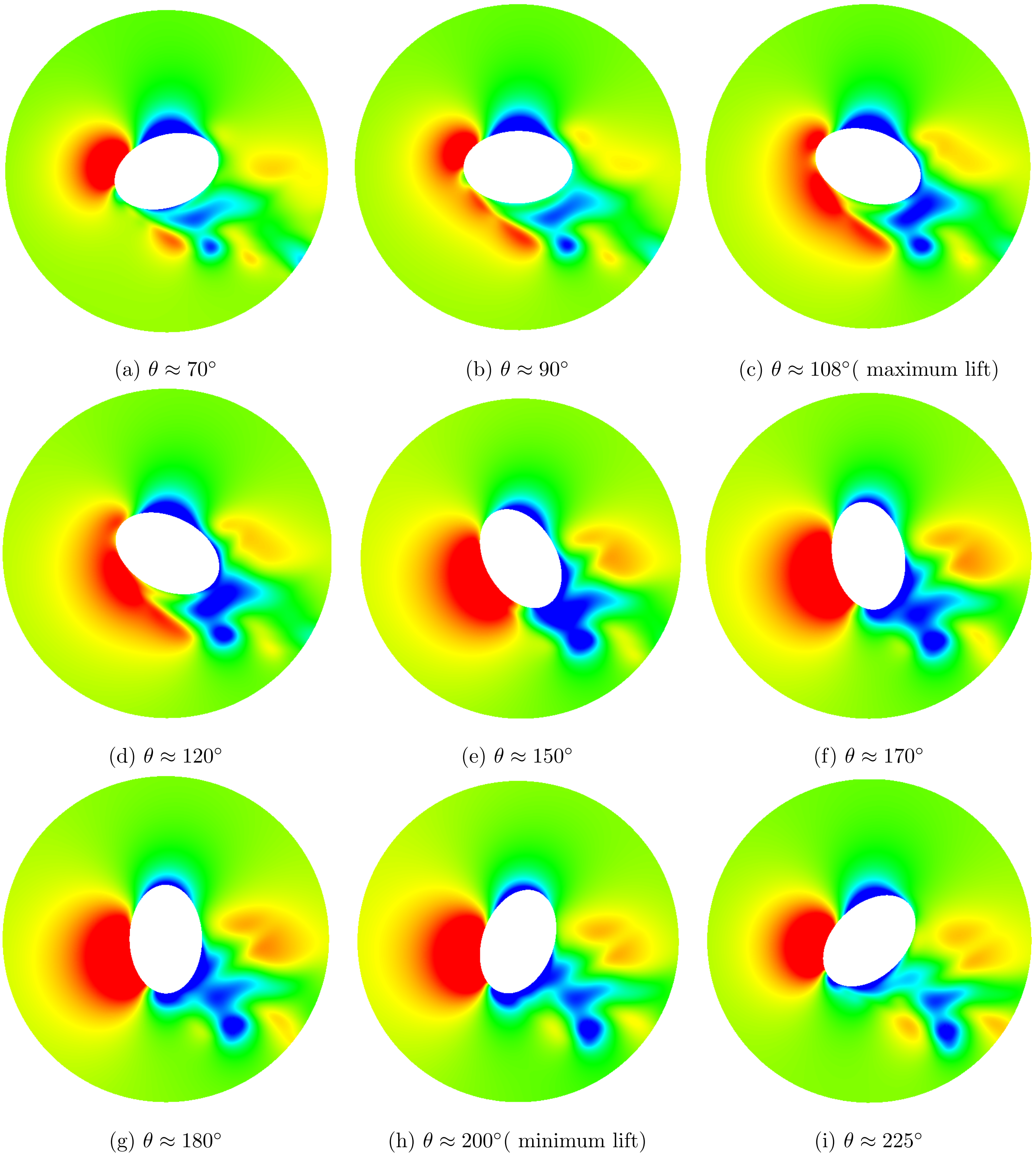


Figure 9.29: Slice view of the velocity magnitude contours at different θ for the nonspherical particle with $a_y = 0.75$.


(a) $\theta \approx 70°$

(b) $\theta \approx 90°$

(c) $\theta \approx 108°$(maximum lift)

(d) $\theta \approx 120°$

(e) $\theta \approx 150°$

(f) $\theta \approx 170°$

(g) $\theta \approx 180°$

(h) $\theta \approx 200°$(minimum lift)

(i) $\theta \approx 225°$

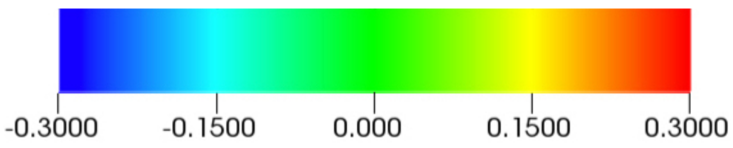
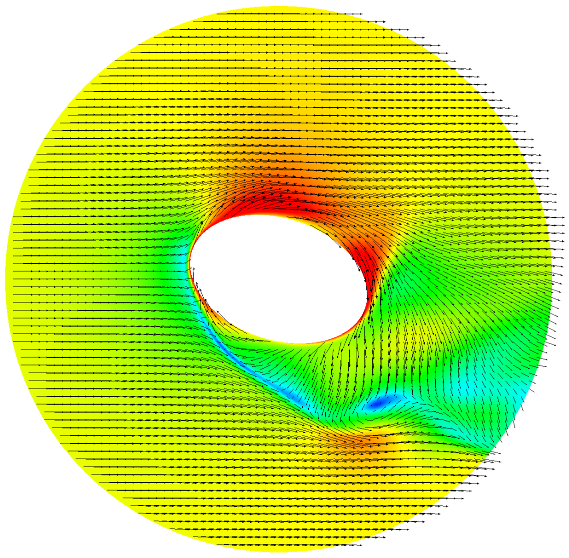
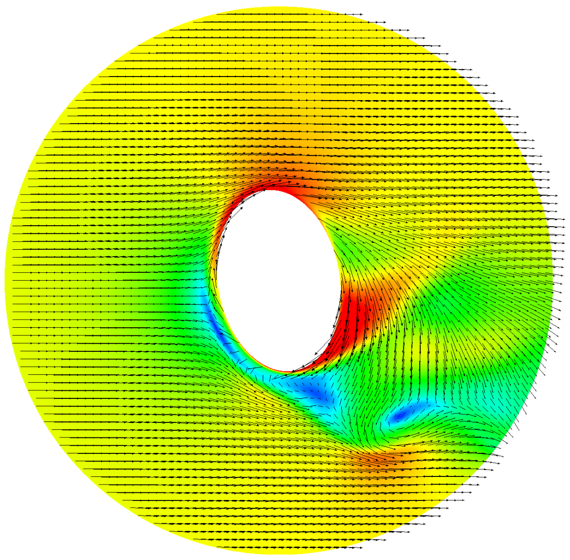


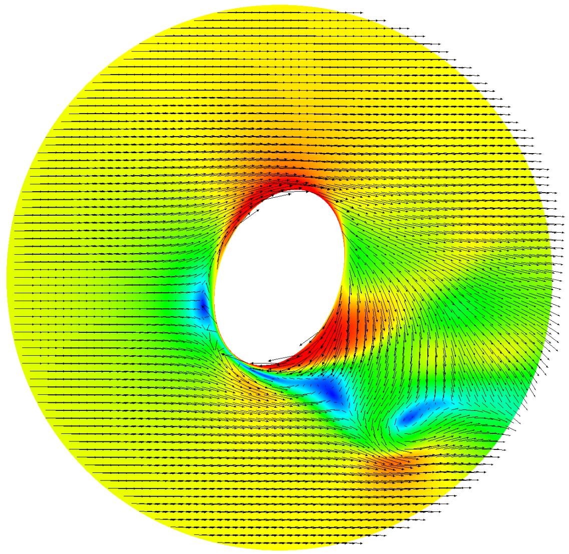
Figure 9.30: Slice view of the pressure contours at different θ for the nonspherical particle with $a_y = 0.75$.



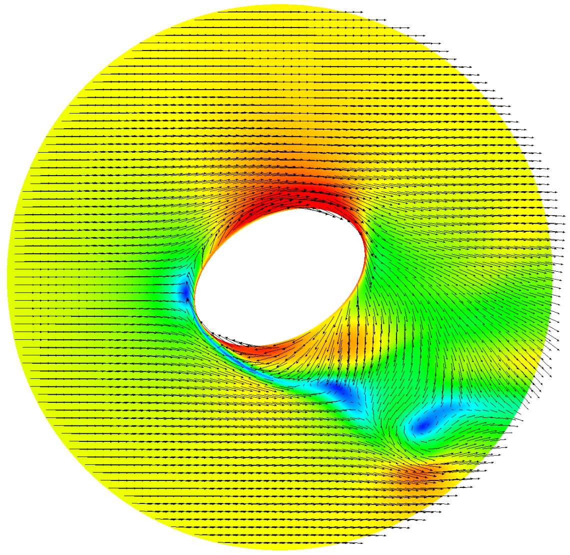
(a) $\theta \approx 108°$(maximum lift)



(b) $\theta \approx 170°$



(c) $\theta \approx 200°$(minimum lift)



(d) $\theta \approx 225°$

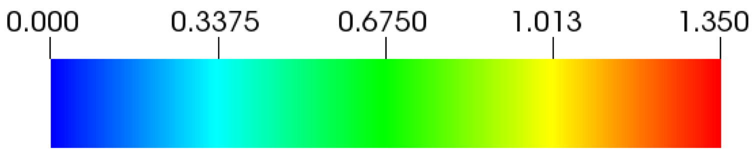


Figure 9.31: Slice view of the velocity magnitude contours with vectors indicating the direction of flow at different θ for the nonspherical particle with $a_y = 0.75$.

9.4.3 Summary

In this section, we have presented some of the preliminary results from the ongoing study on flow past rotating particles. Unlike the rotating sphere, where the flow is steady near the surface with a shear layer instability growing downstream of the sphere, the flow for a nonspherical particle experiences recurring separation and reattachment on its surface. As a result of the rotation, both spherical and nonspherical particles pull near-surface fluid on the leeward side. However, due to the difference in the major and minor axis of the particle, the orientation at which the background flow interacts with the pulled fluid varies. It is this feature of the flow that determines the angle of rotation at which the flow experiences maximum and minimum drag and lift forces. Additionally, we observe that changing the shape of the sphere leads to a decrease in the average lift of the particle and increases in the average drag on the particle, in comparison to the spherical particle. This observation is important because particles are typically modeled as spheres, and the drag and lift coefficients are usually assumed to be those associated with a sphere.

In future work, we will continue the analysis that we have started here to understand how the shape and rotational axis of particle impact the flow characteristics, and how spheres can be used to model the effect of nonspherical particles for production-scale runs with thousands of particles.

9.5 Denticles on Shark Skin

This problem is being studied in collaboration with scientists at the University of Leeds in England, who are analyzing the impact of shark skin denticles on drag reduction. Charlie Lloyd (University of Leeds) generated the spectral element mesh used in this section.

It has been hypothesized that miniature roughnesses on shark skin, known as denticles, help reduce skin-friction and allow sharks to swim faster. Despite decades of experimental and computational research in this area, the question of whether (and the mechanism) denticles reduce drag, remains inconclusive. Denticles have been compared to riblets (grooved structures aligned with the direction of the flow), which have shown to reduce drag by reducing the Reynolds stresses in the buffer region of the boundary layer [136,137,138,139,140,141].

Figure 9.32 shows the shape of riblets that are commonly used in experiments and Fig. 9.33 shows a typical drag reduction profile for a ribletted surface. As we can see, the spacing between riblets (s in Fig. 9.32) is a key factor impacting the drag forces on a riblet. The only DNS done on shark denticles is by Boomsma and Sotiropoulos [142], who observed that denticles increased drag for the geometrical configurations that they had considered. Boomsma et al. conclude in their work that denticles may behave like riblets if the spacing between denticles is just right.

Boomsma and Sotiropoulos had used immersed-boundary methods for their numerical experiments. One of the reasons that DNS has not been extensively used to study shark skin is that generation of a good-quality computational mesh for an array of denticles is difficult, and the computational cost of these calculations can be prohibitively high. The goal of our current work is to use DNS calculations at $Re_\tau = 180$ in a channel (discussed in Section 9.2) with denticles shown in Fig. 9.34. Due to the small size ($y+ = yu_\tau/nu \approx 13$) and curvature of the denticles, the mesh resolution required to model the denticles is higher than the resolution needed for the DNS of a flat channel at the Reynolds number that we are considering. Thus, we use the Schwarz-SEM framework with two overlapping grids; a dense grid near the surface that accurately models the denticles and a relatively coarser grid (yet, DNS quality) in the far-field that resolves all the flow structures.

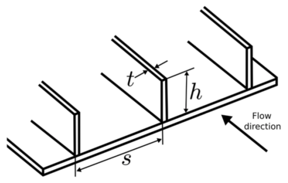


Figure 9.32: Schematic of riblets that are commonly used for experiments.

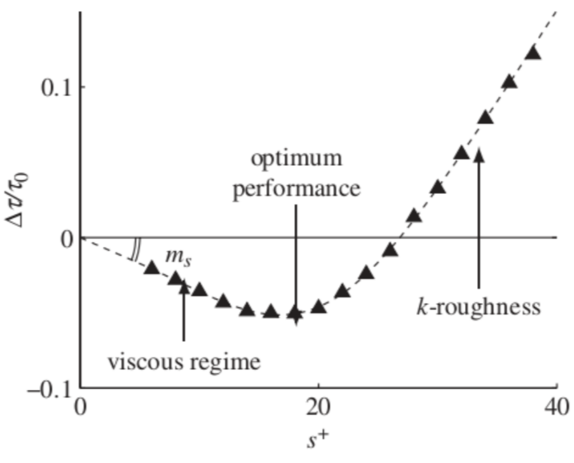


Figure 9.33: Drag reduction profile for a ribleted surface. $\Delta\tau$ is the change in shear stress due to the riblets, and τ_0 is the shear stress for a channel without riblets. s^+ is the distance between riblets in wall units. Image taken from [143].

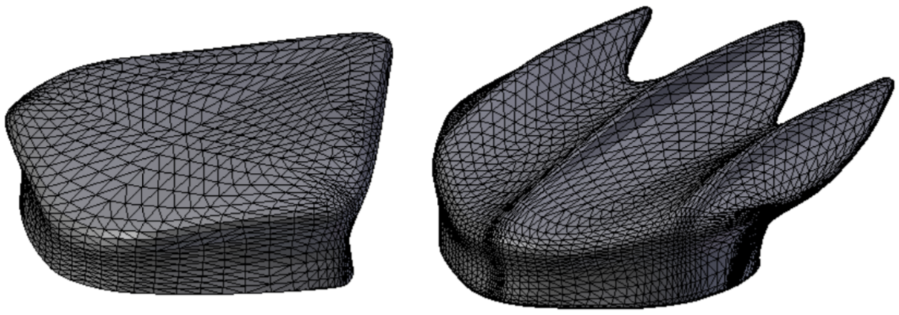


Figure 9.34: (left to right) (a) A smoothed and a (b) ribletted denticle. The ribletted denticle is based on denticles of a Shortfin mako shark, but it has been scaled to have identical aspect ratio as the smoothed denticle shown on the left. We note that these plots show the surface triangulation that is used by CAD, and should not be confused with the spectral element mesh that we use for analyzing these denticles.

9.5.1 Problem setup

Figure 9.34 shows CAD models of the two denticles that will be analyzed using the Schwarz-SEM framework. Figure 9.34(a) shows the smooth denticle that we use for the results presented in this dissertation and Fig. 9.34(b) shows a ribletted denticle. The ribletted denticle is based on the denticle of a Shortfin mako shark, which has been scaled to have the same aspect ratio as the smoothed denticle. We note that the CAD models in Fig. 9.34 show the surface triangulation that is used by CAD, and it should not be confused with the spectral element mesh that we use for analyzing these denticles.

The geometrical parameters that are used to describe the shape of denticles are shown in Fig. 9.35, and Table 9.6 lists the corresponding parameters. We note that the parameters in Table 9.6 are nondimension-

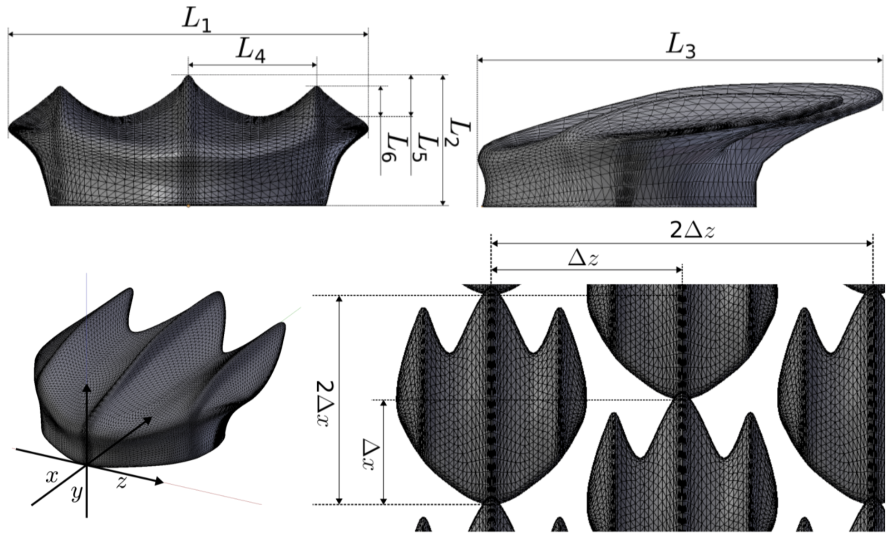


Figure 9.35: Geometrical parameters for defining the size of denticles.

	L_1/h	L_2/h	L_3/h	L_4/h	L_5/h	L_6/h	Δ_x	Δ_z
Ribletted	0.25	0.091	0.287	0.089	0.029	0.022	0.138	0.25
Smooth	0.25	0.073	0.287	-	-	-	0.138	0.25

Table 9.6: Denticle dimensions (scaled by the half channel height h) corresponding to those defined in Fig. 9.35.

alized with the half channel height (h).

For numerical calculations using overlapping grids, the target computational domain size is the same as that used for the turbulent channel (Section 9.2) for the streamwise (x) and spanwise (z) direction. For the wall-normal direction, however, the channel height is h instead of $2h$, and a stress-free boundary condition is used at the top surface. This domain size ($\Omega = [4\pi h \times h \times 4\pi h/3]$) was also used for the DNS in [142].

Using a single conforming grid for modeling this domain requires about 210,000 elements, and overlapping grids reduces the element count by about 50%. Since the computational cost of the calculation is still significant, we seek to use a domain scaled to roughly a quarter of the target channel streamwise and spanwise length, with a staggered array of denticles. The purpose of this smaller domain is to develop a better understanding of the mesh resolution required to adequately resolve all fluid structures in the flow. Additionally, this smaller domain will also allow us to use spatial correlations to determine the minimum streamwise and spanwise length required to get rid of any correlations in the velocity field.

Figure 9.36 shows the staggered array of the denticle from Fig. 9.34(a) that we consider for DNS in this dissertation. This 12×2 array is generated by replicating a single doubly-periodic block with 2 denticles (also shown in Fig. 9.36), which allows us to model a total of 48 denticles. We also show the spectral element

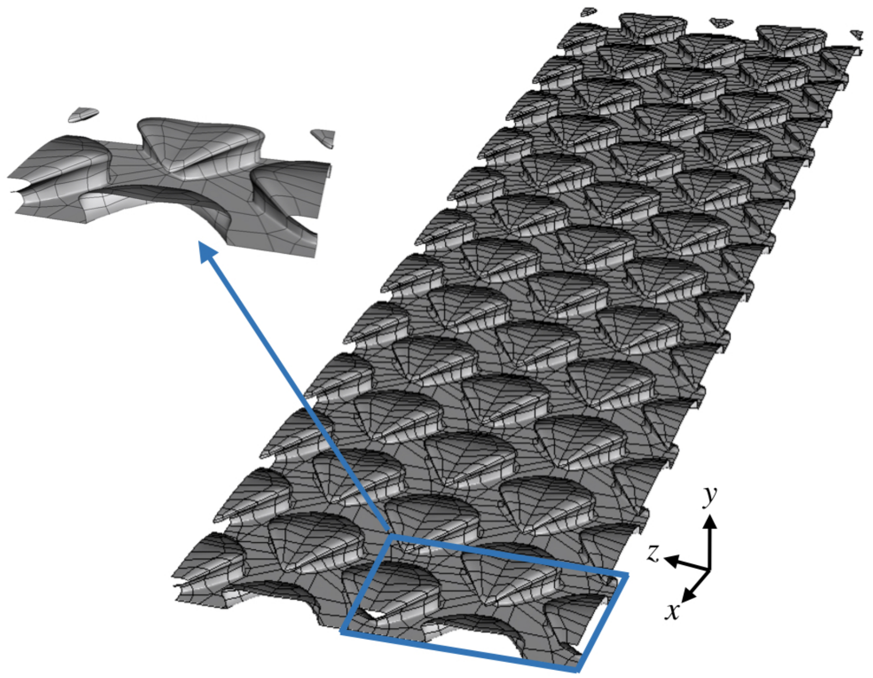


Figure 9.36: Array of denticles generated by replicating a doubly-periodic block with 2 denticles, for DNS at $Re_\tau = 180$. The spectral element mesh is shown for the surface of the denticles.

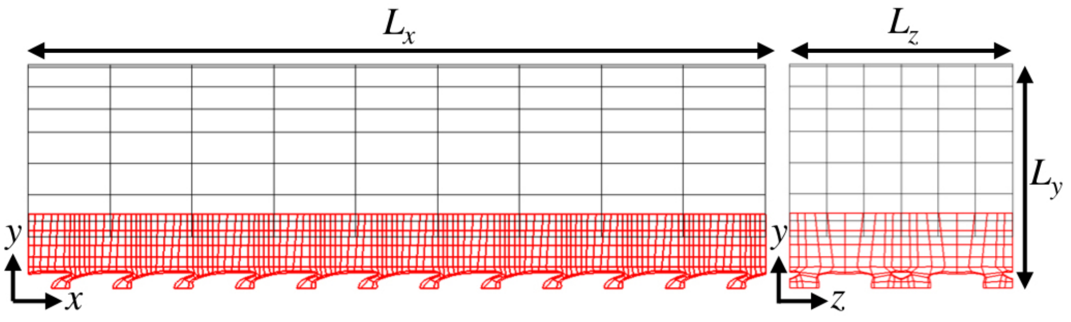


Figure 9.37: Slice view of the overlapping meshes used for array of denticles.

mesh on the surface of the denticles in Fig. 9.36.

Figure 9.37 shows a slice view of the overlapping grids used for turbulent flow over the array of denticles. The dimensions of the domain are $L_x = 3.3h$, $L_y = h$, and $L_z = h$. The lower mesh has $16,272$ elements, and the upper mesh has 672 elements. The element count is higher in the lower mesh due to the geometry of the denticles, and using a single conforming grid would have required a total of 33,072 elements. Thus, overlapping grids reduce the total element count by about 50%. The overlap for the two meshes is $0.08h$ wide between $\Omega_y = [0, 0.33h]$ for the lower mesh and $\Omega_y = [0.25h, h]$ for the upper mesh.

Following Boomsma, the calculation was done with a fixed flow rate through the domain, which was determined using the bulk velocity associated with the bulk Reynolds number ($Re_{bulk} = U_{bulk}h/\nu$) in the smooth channel flow corresponding to $Re_\tau = 180$. Based on the smooth channel flow calculation (Section 9.2), the bulk velocity is $U_{bulk} \approx 15.63$. Consequently, the flow rate is fixed to $15.63L_yL_z$. The DNS calculation was run for more than 100 flow-through times, and time-averaged statistics were collected for about 75 flow-through times. For the Schwarz iterations, $Q = 3$ and $m = 3$ to ensure third-order temporal accuracy. Additionally, in order to spatially average the results, blockwise averaging is performed to combine the statistics for the 48 denticles that were replicated from the single block containing 2 denticles (shown in Fig. 9.36).

9.5.2 Results

Figure 9.38 shows instantaneous velocity magnitude contours for the channel with denticles. The drag is monitored on the no-slip surfaces, and results indicate that the drag increases by about 50% as compared to the smooth wall channel case. These results are similar to those of Boomsma and Sotiropoulos, who have explored a similar staggered configuration for a different shaped denticle and found that the drag is increased by 44-50% in comparison to a flat channel [142].

Next, in order to understand whether the length and spanwise width of the channel is sufficient, we look at the two-point correlations (R_{ii}) along various sections of the domain. The two-point correlation is defined as

$$R_{ii} = \frac{\overline{u_i'(\mathbf{x},t)u_i'(\mathbf{x}+\mathbf{r},t)}}{\overline{u_i'^2(\mathbf{x},t)}}, \tag{9.3}$$

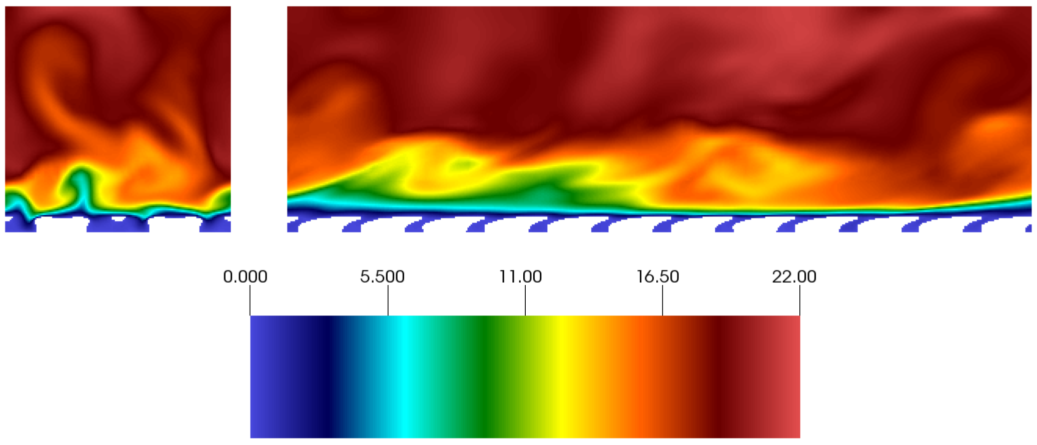
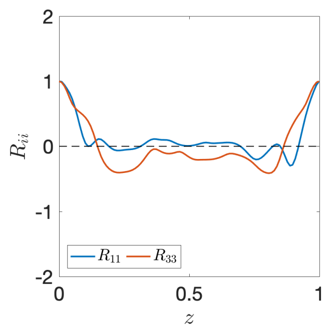
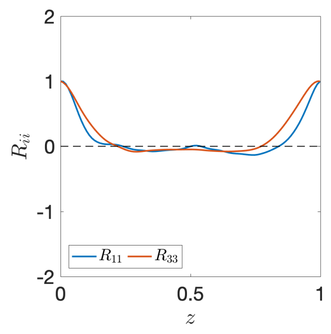


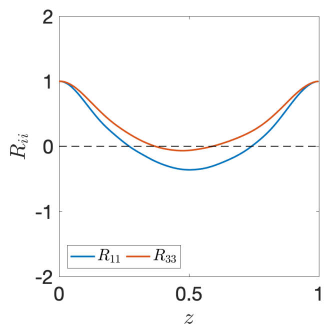
Figure 9.38: Slice view of the velocity magnitude contours with and without the denticles.



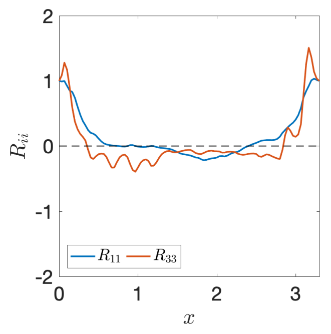
(a) $x = L_x/2, y = 0.08h, z = 0 - L_z$



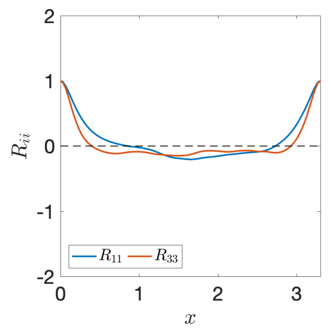
(b) $x = L_x/2, y = 0.15h, z = 0 - L_z$



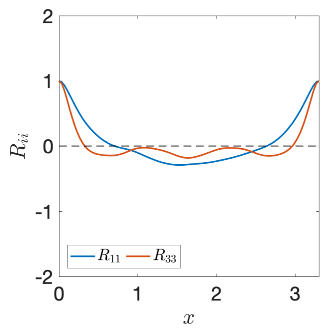
(c) $x = L_x/2, y = 0.65h, z = 0 - L_z$



(d) $x = 0 - L_x, y = 0.08h, z = L_z/2$



(e) $x = 0 - L_x, y = 0.15h, z = L_z/2$



(f) $x = 0 - L_x, y = 0.65h, z = L_z/2$

Figure 9.39: Spatial correlations R_{11} and R_{33}, corresponding to the streamwise and spanwise velocity, along different sections of the domain.

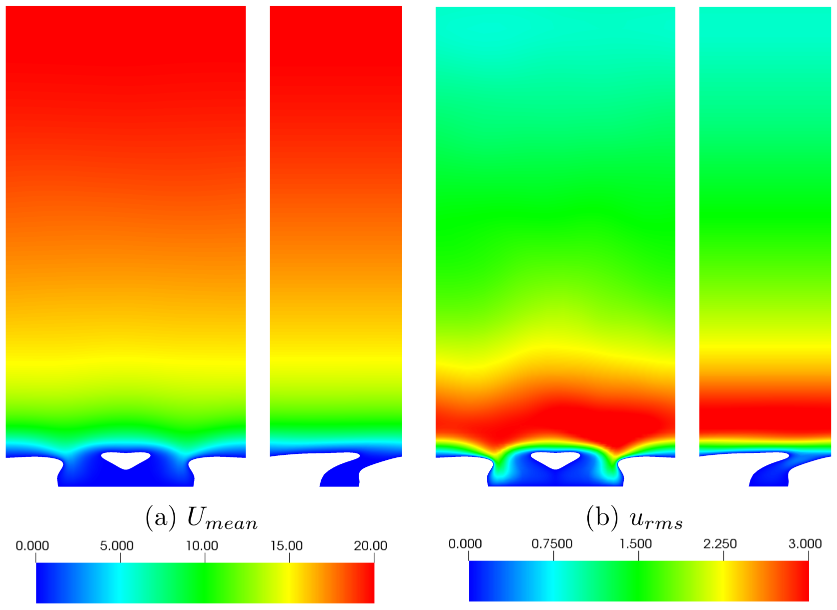
where $u_i^{'}$ is the ith component of the fluctuating velocity, and $\overline{u}$ indicates ensemble averaging for u. R_{ii} is a function of space and time, and is calculated at using the time-averaged and instantaneous velocity at each time-step. Figure 9.39 shows spatial correlations along six different lines in the domain for the streamwise (R_{11}) and spanwise velocity (R_{33}) components. Three lines are along the spanwise direction ($z = 0 - L_z$), taken at the mid-plane in the streamwise direction ($x = L_x/2$) at different heights ($y = 0.08h, 0.15h$ and $0.65h$). We have another three lines that are along the streamwise direction ($x = 0 - L_x$) at the mid-plane in the spanwise direction ($z = L_z/2$) at different heights ($y = 0.08h, 0.15h$ and $0.65h$). The height of the denticle is about $0.07h$, and the lines at $y = 0.08h$ help us determine the correlation in the velocity field right above the denticles.

The autocorrelation plots shown in Fig. 9.39(b), (c), (e) and (f) indicate that the streamwise and spanwise velocities are uncorrelated in the streamwise and spanwise directions at $y = 0.15h$ and $y = 0.65h$. Near the denticles, however, we notice in Fig. 9.39(a),(d) that R_{ii} for $y = 0.08h$ is not less than zero for a sustained length of the domain. These results indicate that L_x and L_z should be increased for accurately capturing all the turbulent structures in the flow.

Figure 9.40 shows the slice view of the time-averaged streamwise velocity U_{mean} and root-mean-square (RMS) velocity in the streamwise direction u_{rms}. We can see that U_{mean} is symmetric in the spanwise direction, but u_{rms} is slightly asymmetric. Similarly, the plots for v_{rms} and w_{rms} in 9.41 also show some asymmetry in the spanwise direction. One of the reasons for this asymmetry in the flow could be that the growth of turbulent structures (or disturbances) in the flow is limited due to the insufficient spanwise length of the domain. This hypothesis is based on a previous calculation that was done with a domain that was shorter by a factor of 2 in the spanwise direction and a factor of 4 in the streamwise direction. In that calculation, we had observed a more prominent asymmetry in the flow in the spanwise direction. These mean- and fluctuating-velocity plots are also consistent with the spatial correlations shown in Fig. 9.39, which indicates that the spanwise length of the domain is not sufficient.

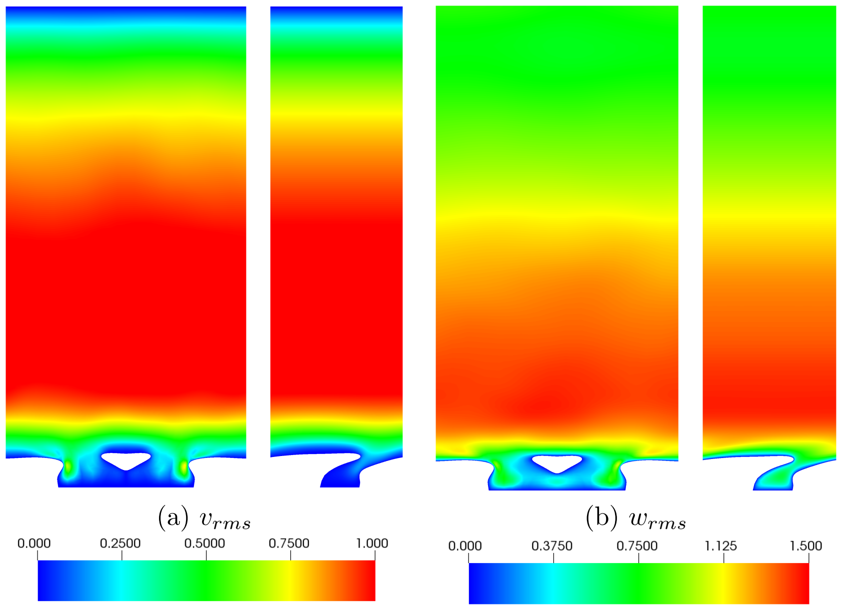
Despite the asymmetry in the temporally and spatially averaged flow field, which is due to insufficient streamwise and spanwise lengths, we observe that the Schwarz-SEM framework accurately captures the flow across overlapping grids. The time-averaged streamwise velocity U_{mean} and RMS velocities show good agreement between the overlapping grids.

In the ongoing production level calculations, we are analyzing flow over arrays of smoothed and riblettted denticles where we have set $L_x = 6.6h$ and $L_z = 3h$ to ensure that the streamwise and spanwise velocity components are uncorrelated.



(a) U_{mean} (b) u_{rms}

Figure 9.40: Slice view, along the (left) streamwise and (right) spanwise direction, for (a) U_{mean} and (b) u_{rms} , for flow over denticles.



(a) v_{rms} (b) w_{rms}

Figure 9.41: Slice view, along the (left) streamwise and (right) spanwise direction, for (a) v_{rms} (b) w_{rms} , for flow over denticles.

9.6 Oscillatory Boundary Layer Flow

This problem is being studied in collaboration with scientists in the Civil Engineering department at UIUC, who are analyzing the impact of oscillatory boundary layer flow on sediment transport. Dimitrios K. Fytanidis have done the monodomain calculations discussed here.

Oscillatory boundary layer (OBL) flows have a significant impact on sediment transport in environmental flows such as coastal and offshore environments [144, 145, 146]. An important aspect of OBLs is the phase angle at which the maximum wall shear stress (τ) peaks in comparison to the maximum velocity. Experimental and computational research in the past has shown that there is a phase lead (ϕ) between the maximum velocity with respect to the maximum wall shear stress [147,148,149,150], and Fig. 9.42 shows a plot of how the phase lead varies with Reynolds number. For OBL flow, the Reynolds number is typically defined as $Re_\delta = U_{max}\delta/\nu$, where U_{max} is the maximum oscillatory velocity, δ is the Stokes boundary layer length, and ν is the kinematic viscosity. The Stokes boundary layer length δ depends on ν and the angular frequency $\omega = 2\pi/T_p$ as $\delta = \sqrt{2\nu/\omega}$, where T_p is the period of the oscillating flow. For numerical calculations, the oscillatory flow is driven by a sinusoidal forcing term in the momentum equation (2.26), which is of the form $\mathbf{f} = -U_{max}\omega\sin(\omega t)\hat{e}_1$, where t is time, and $\hat{e}_1$ is the unit vector in the direction (x) of the flow.

Extensive experiments conducted by Mier in the Large Oscillatory Water-Sediment Tunnel (LOWST) at the Ven Te Chow Hydrosystems Laboratory (UIUC) have studied the transition between laminar and turbulent flow regimes with smooth bed and have found evidence that shear stress does not lag maximum velocity for all Reynolds number [152]. Mier has shown that there is a phase lead instead of a phase lag for maximum shear stress with respect to the maximum flow velocity, for a range of Reynolds number. This

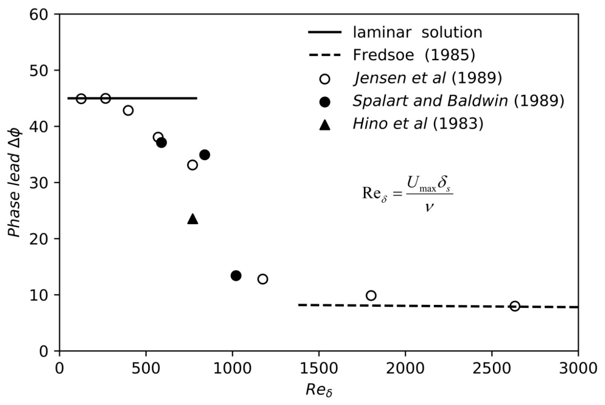


Figure 9.42: Phase difference between maximum shear stress and maximum velocity in OBL flow for a range of Reynolds number. Image taken from [151].

finding is an important stepping stone towards better understanding of OBL flows. Due to the limitation of the applied point-wise experimental technique (Laser Doppler Velocimetry), however, it is not possible to extend this study for investigating the relation of the phase-lag with the development of flow structures. The accurate prediction of sediment transport and complex flow structures in OBLs, thus, hinges on high fidelity calculations that can simulate this phenomenon.

9.6.1 Problem setup and monodomain SEM results

The ongoing computational calculations at UIUC are targeting a DNS study of OBLs in hydrodynamically smooth and rough channels. Due to the complexity and scale of the structures in OBL, DNS calculations require significant computational resources to analyze this problem. As a first step, preliminary calculations have been done using a conformal grid with $E = 353,400$ at $N = 9$, which has a total of roughly 353 million computational points. The computational domain is a channel that has periodic boundary conditions in the streamwise (x) and spanwise (z) direction, and a no-slip and stress-free boundary condition at the bottom and top surface, respectively, in the wall-normal direction (y).

The monodomain SEM calculation has been done at $Re_\delta = 763$ for a hydrodynamically smooth surface. Following standard nondimensional practice, $U_{max} = \delta = 1$, and $\nu = 1/Re_\delta$. The period of one oscillation is $T_p = \pi/\nu \approx 2396.45$ convective time units, the nondimensional forcing is $F = -2Re_\delta \sin(2Re_\delta t)\hat{e}_1$, and the size of the computational domain is $L_x \times L_y \times L_z = 160\delta \times 30\delta \times 40\delta$. Figure 9.43 shows the conforming mesh used for the monodomain SEM calculation, along with λ_2 vortices colored by velocity magnitude contours.

Due to the oscillatory nature of the flow, we analyze the results at different phase angles during an oscillation period. The phase angle $\phi = 2\pi t/T_p = 360^\circ t/T_p$ corresponding to maximum streamwise velocity magnitude is 180° and 360° (or 0°), and the flow reverses directions at phase angles 90° and 270°. Figure 9.44 shows contours of the phase-averaged mean streamwise velocity ($\overline{U}$) and velocity fluctuations ($\overline{u'u'}, \overline{v'v'}, \overline{u'v'}$) for the OBL for $0^\circ \leq \phi \leq 360^\circ$, and Fig. 9.45 shows $\overline{U}, \overline{u'u'}, \overline{v'v'}$, and $\overline{u'v'}$ at six different phases. $\overline{u}$ in Fig. 9.44 and Fig. 9.45 represents phase and spatial averaging of u. Mean streamwise velocity and velocity fluctuations have been obtained by first averaging the instantaneous flow based on the phase angle, and then spatially averaging the flow in the streamwise and spanwise direction.

The monodomain calculation showed that, as expected, the turbulence is primarily confined to near the wall of the channel, and does not grow away from the surface because of the oscillatory nature of the flow. These results suggest that we should be able to use overlapping grids with a dense mesh for the lower one-third of the domain and a coarser mesh away from the wall, to simulate this domain with many fewer elements for reduced computational cost. Additionally, the long term goal of this study is to understand

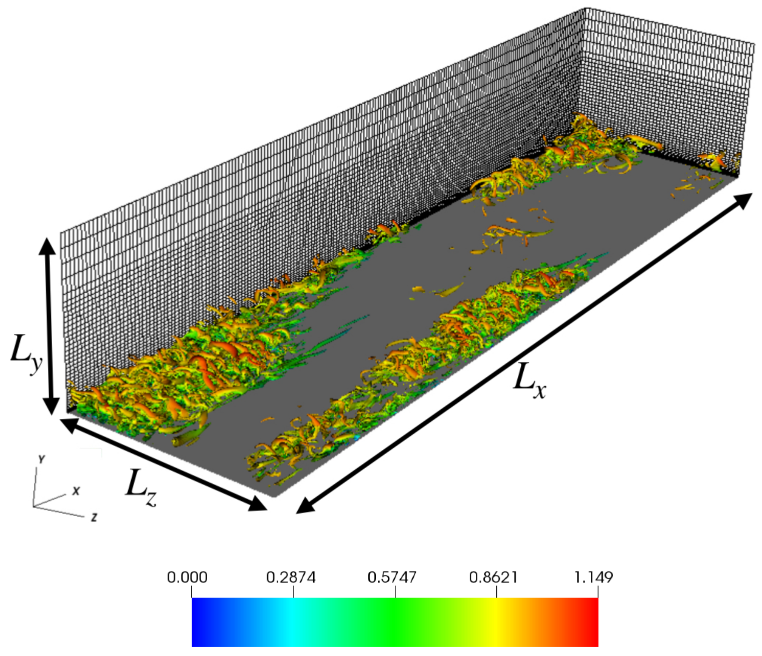


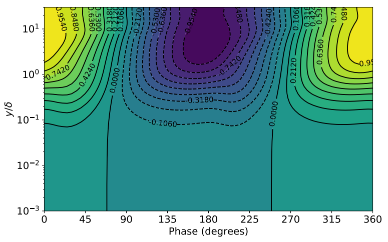
Figure 9.43: Conforming mesh used for the monodomain SEM calculation, along with λ_2 vortices colored by velocity magnitude contours.

OBL on hydrodynamically rough surfaces. Depending on roughness size, resolution for discretizing the channel-bed with a conforming mesh can be intractable due to the computational cost. Thus, benchmarking the Schwarz-SEM framework using the OBL flow in the smooth channel will give us valuable insight into modeling OBL flow in a rough channel.

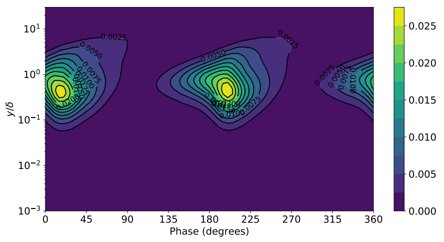
9.6.2 Schwarz-SEM results

Figure 9.46 shows the overlapping grids used for simulating the OBL. The lower mesh, with $E = 220,900$, has the same resolution as the corresponding section of the monodomain mesh. The upper mesh is coarser by a factor of roughly 50% in the streamwise and spanwise directions as compared to the original mesh, and has 38,000 elements. Using a coarse mesh for the upper portion of the domain reduces the total element count by about 26% in comparison to the monodomain mesh. The overlap between the upper and lower mesh is between $y = 9.75\delta$ and 10.5δ.

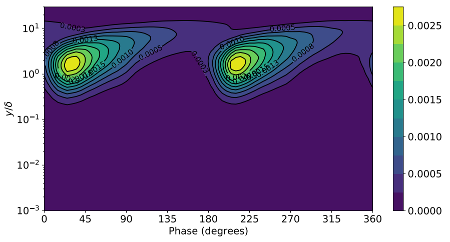
Figure 9.47 shows contours of the phase-averaged mean streamwise velocity ($\overline{U}$) and velocity fluctuations ($\overline{u'u'}, \overline{v'v'}, \overline{u'v'}$) for the OBL at $Re_\delta = 763$. These results show a good qualitative comparison with results from the monodomain SEM calculation. We also look at the mean streamwise velocity and velocity fluctuations at six different phase angles and compare them to the monodomain SEM results in Fig. 9.48. The



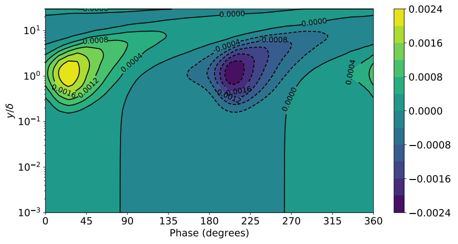
(a) $\overline{U}$



(b) $\overline{u'u'}$



(c) $\overline{v'v'}$



(d) $\overline{u'v'}$

Figure 9.44: Contours plots for the streamwise velocity and velocity fluctuations for the OBL at $Re_\delta = 763$.

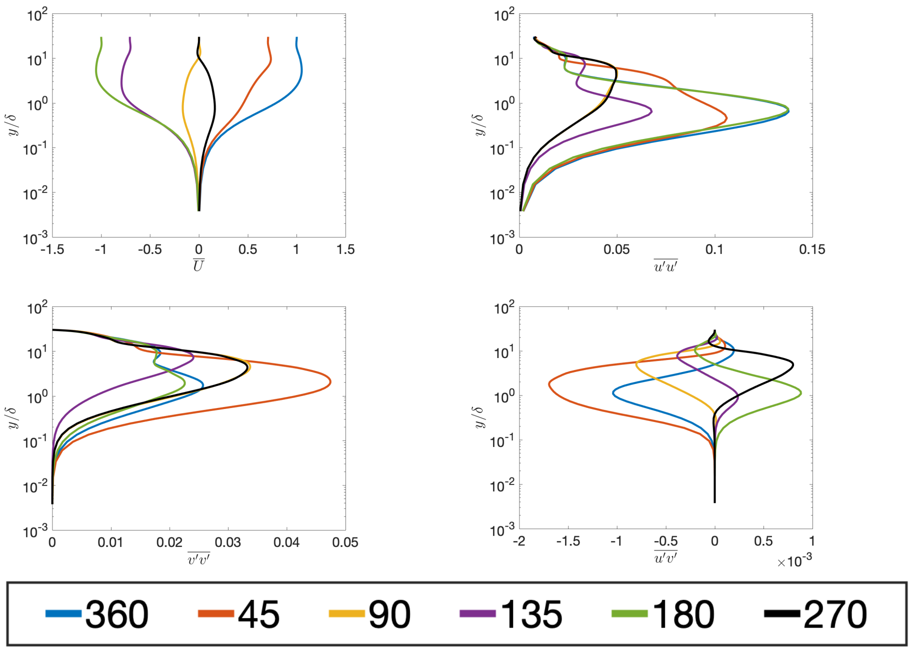


Figure 9.45: Streamwise velocity and velocity fluctuations plots for the OBL at $Re_\delta = 763$ at different phase angles of the flow.

solid lines in Fig. 9.48 correspond to the monodomain results, and the dashed lines correspond to the results from the overlapping grid calculation.

Figure 9.48(a) shows that the mean streamwise velocity from the overlapping grid calculation agrees

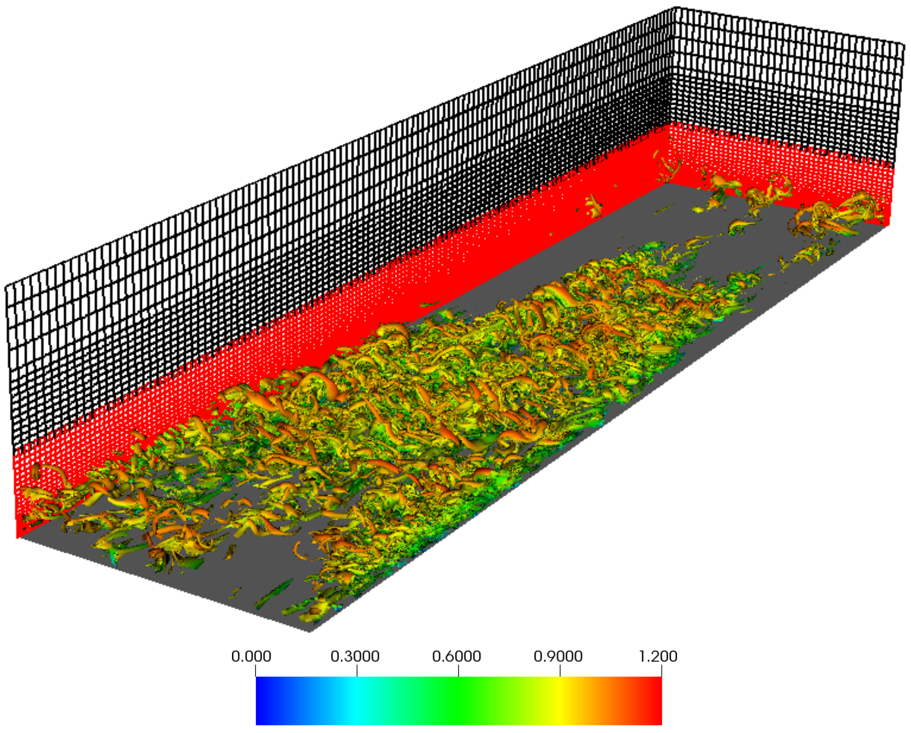
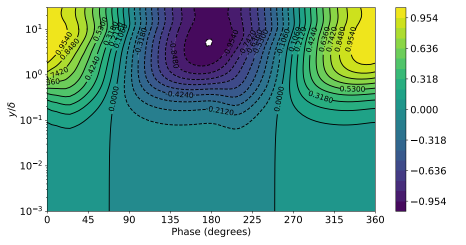
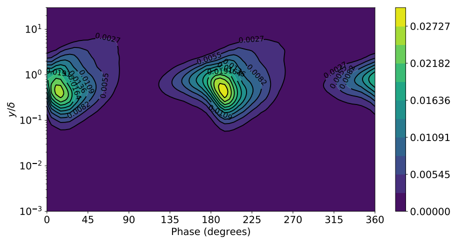


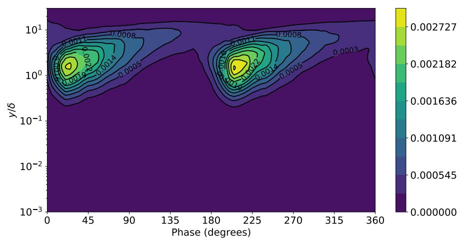
Figure 9.46: Overlapping meshes in the Schwarz-SEM framework, along with λ_2 vortices colored by velocity magnitude contours for the OBL at $Re_\delta = 763$.



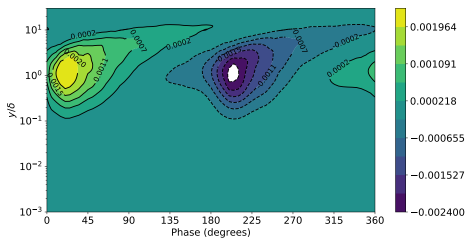
(a) $\overline{U}$



(b) $\overline{u'u'}$



(c) $\overline{v'v'}$



(d) $\overline{u'v'}$

Figure 9.47: Contours plots for the mean streamwise velocity and velocity fluctuations for the OBL at $Re_\delta = 763$.

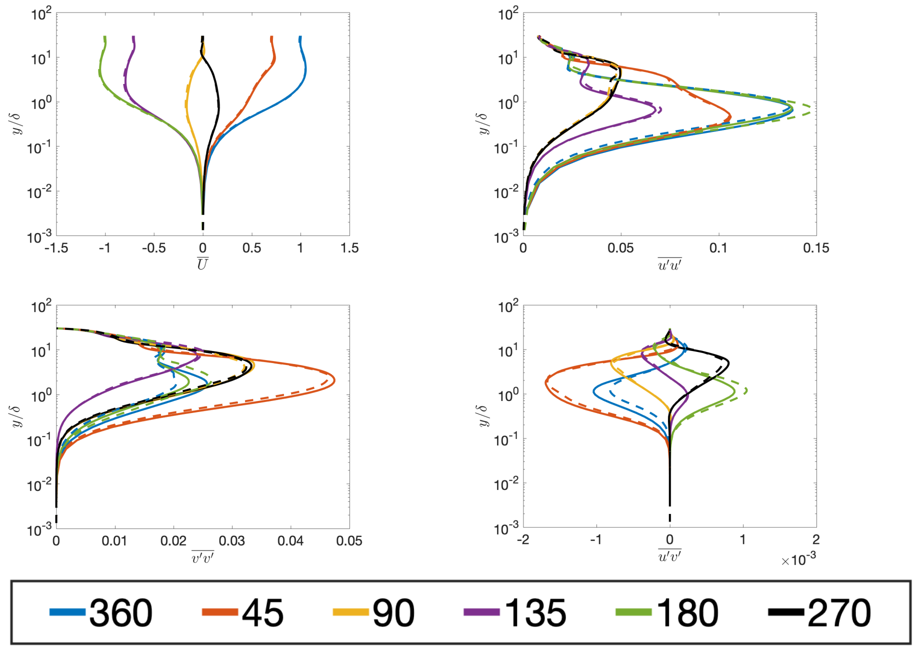


Figure 9.48: Streamwise velocity and velocity fluctuation profiles for the OBL at $Re_\delta = 763$ at different phase angles of the flow. Solid lines correspond to monodomain calculation and dashed lines correspond to overlapping grid calculation.

well with the results of the monodomain calculation. However, we observe that the results for velocity fluctuations ($\overline{u'u'}, \overline{v'v'}, \overline{u'v'}$) do not agree very well with the monodomain calculation. This difference is likely because the flow statistics were collected for over 5 oscillation periods for the monodomain calculation and for only 1 oscillation period for the overlapping grid calculation. In future work, we will continue these simulations to average the flow statistics over more oscillation periods and see how the velocity fluctuation results compare with the monodomain calculations. Nonetheless, the results shown here are promising and show yet another class of real-world problems where the Schwarz-SEM framework is an effective alternative to the monodomain SEM framework.

9.7 Multirate Time-Stepping for Modeling a Thermally-Buoyant Plume

This problem is being studied in collaboration with scientists at the City University of New York, who are analyzing buoyant plumes in oceanic environments. The monodomain calculations discussed here have been done by Dr. Som Dutta and Dr. Drew Poje.

Buoyant plumes are of interest in many industrial and environmental applications such as deepwater blowouts [153], volcanoes [154] and hydrothermal vents [155]. The Deep Water Horizon (DWH) incident in 2010, which leaked millions of barrels of oils into the ocean, is one of the incidents that has shed light on the importance of understanding the internal dynamics of buoyant plumes [156]. Buoyant plumes, such as the one resulting from the DWH oil spill, are driven by the buoyancy flux due to multiphase effluents (both, oil and gas) at elevated temperatures in comparison to a background fluid at a lower temperature. The background flow (oceanic, in this case) is highly irregular with a broad range of spatial and temporal scales and must be accurately modeled to properly account for the interaction between effluents and the background flow. Additionally, background stratification and the rotation of earth also impact the dynamics of the plume. The impact of these different factors on buoyant plumes has been the subject of various experimental and computational studies [157, 158, 159, 160, 161].

Oceanic environments are subject to stable stratification, and a singlephase plume grows in such a stably stratified environment because of the buoyancy flux associated with the temperature difference between the plume and the background fluid. As the plume rises, it entrains the surrounding fluid, which decreases the buoyancy flux due to the turbulent mixing of the plume with the relatively denser surrounding fluid. Eventually, the plume reaches a neutral buoyancy level, where the temperature difference vanishes. However, the plume keeps rising due to inertial effects until its density is significantly higher than the surrounding fluid. At this point, the plume is negatively buoyant, and the fluid falls back and flows *outwards.* This height at which the plume becomes neutrally stratified is known as the equilibrium height, and the height at which the neutrally buoyant plume accumulates and flows outwards is known as the trapping height. There is also a maximum height associated with the plume, which is the height at which the mean centerline velocity (in the vertical direction) vanishes. Figure 9.49(left) shows a schematic of a singlephase plume and indicates its maximum plume height (z_{max}), trapping height (z_{th}) and equilibrium height(z_{eq}) [160]. In Fig. 9.49, $< \hat{w} >$ indicates the centerline velocity in the direction of the plume (z), and $< \hat{g} >$ indicates the difference in the density of the plume with the background flow.

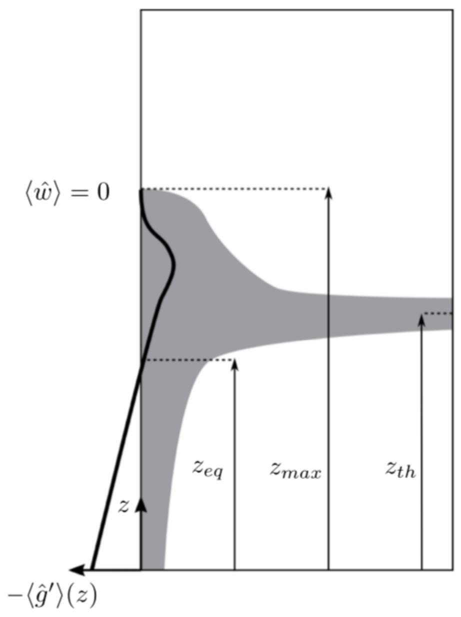


Figure 9.49: Schematic of a singlephase plume indicating the maximum plume height (z_{max}), trapping height (z_{th}) and equilibrium height(z_{eq}). Image taken from [160].

Multiphase plumes have a more complicated dynamic, but we do not discuss them here because they are beyond the scope of this dissertation. Computational analysis using SEM is currently underway through a collaboration between UIUC and the City University of New York to understand how rotation impacts singlephase and multiphase plumes in cross-flow. The domain size of this target problem is intractable with the monodomain framework due to its computational cost, and our goal is to demonstrate the effectiveness of the Schwarz-SEM framework for this class of problems.

9.7.1 Governing equations and problem setup

The governing equations for the incompressible fluid flow have been discussed in Chapter 2 (2.26,2.27,2.35), where we had assumed constant density in the domain. For stratified flows, however, the density is not constant in the domain, and we model the density variation using the Boussinesq approximation.

We assume that there is a reference density (ρ_r) and a reference temperature (T_r), with respect to which the density varies in the domain as $\rho = \rho_r(1 - \gamma(T - T_r))$, where γ is the thermal expansion coefficient of the fluid. We also assume that the temperature in the domain, which is a solution of (2.35), can be described as $T(\mathbf{x}, t) = \theta(\mathbf{x}, t) + T_r + \Gamma z$, where θ is the perturbation with respect to the unperturbed environment temperature, T_e, which varies linearly with a slope of Γ as $T_e = T_r + \Gamma z$. Here, z is the direction in which the fluid is stratified.

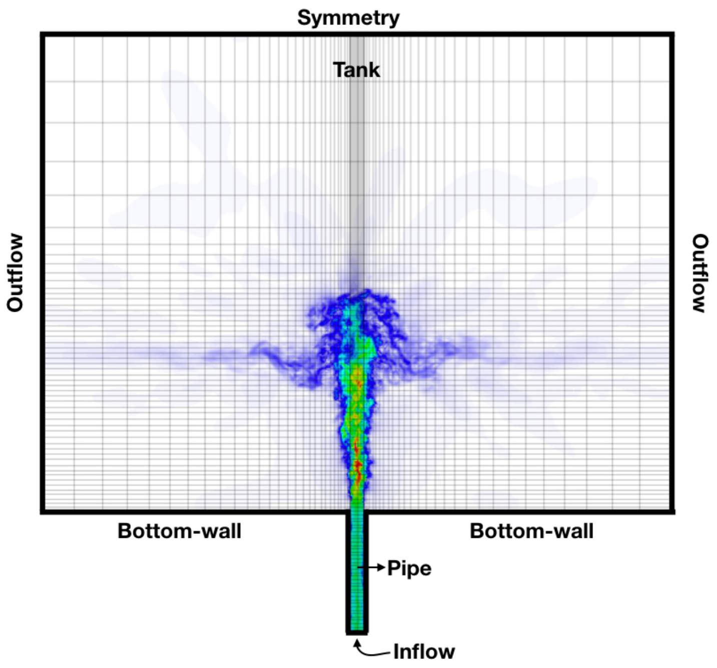


Figure 9.50: Schematic of the axisymmetric domain used to model the thermally buoyant plume, along with instantaneous velocity magnitude contours and the spectral element mesh.

Using these assumptions, and accounting for the buoyancy force, the governing equations for the flow are

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}.\nabla \mathbf{u} = -\nabla p + \frac{1}{Re}\nabla^2 \mathbf{u} + Ri\theta \hat{k}, \tag{9.4}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{9.5}$$

$$\frac{\partial \theta}{\partial t} + \mathbf{u}.\nabla \theta = \frac{1}{Pe}\nabla^2 \theta - \mathbf{u} \cdot \hat{k}, \tag{9.6}$$

where we use the Schwarz-SEM framework to solve for velocity $u(\mathbf{x}, t)$, pressure $p(\mathbf{x}, t)$ and the temperature perturbation $\theta(\mathbf{x}, t)$ in the buoyant plume. The actual temperature $T(\mathbf{x}, t)$ can be obtained by substituting $T(\mathbf{x}, t) = \theta(\mathbf{x}, t) + T_r + \Gamma z$. In (9.4), $Ri = g/(B_o^{1/4} N_b^{5/4})$ is the Richardson number that depends on the gravitational constant (g), inlet buoyancy flux (B_0) and the buoyancy frequency (N_b) (also known as the Brunt–Vaisala frequency). The velocity, time, length, pressure and temperature scales used for nondimensionalization are $U_0 = (B_0 N_b)^{1/4}$, $t_0 = 1/N_b$, $L_0 = (B_0/N_b^3)^{1/4}$, $p_0 = \rho_r U_0^2$, and $T_0 = \Gamma L_0 = (B_0 N_b^5)^{1/4}/(g\gamma)$, where g is the acceleration due to gravity. We assume that the reference density is $\rho_r = 1$ and the reference temperature is $T_r = 0$.

We follow [160] and specify $B_0 = 5 \times 10^{-6} m^4 s^{-3}$, $N_b = 0.1 s^{-1}$. The linear scaling for density is set to $\gamma = 2 \times 10^{-4} K^{-1}$, and for temperature to $\Gamma = 5.1 m^{-1}$. Double diffusion effects are ignored, and thus

$\nu = \alpha = 10^{-6} m^2 s^{-1}$, which leads to Prandtl number $Pr = 1$. The Reynolds number $(U_0 L_0/\nu)$ and Peclet number of the flow is about 7100, and the Richardson number is 3700.

Figure 9.50 shows a schematic of the axisymmetric domain used to model the thermally-buoyant plume, along with velocity magnitude contours of the flow and the corresponding monodomain mesh. The different boundary surfaces have been marked in the schematic. A stress-free boundary condition is used at the top of the tank, and outflow is used on the tank sides. For the bottom of the tank, a homogeneous Dirichlet boundary conditions are imposed for velocity $(u(\mathbf{x}, t))$ and temperature perturbation $(\theta(\mathbf{x}, t))$. Inhomogeneous Dirichlet boundary conditions are imposed at the inlet of the pipe, that is connected to the tank, for velocity and temperature perturbation. In nondimensional units, the source (pipe) diameter is $D_0 = 0.3$, the length of the pipe is 2.4, the radius of the tank is 6, and the height of the tank is 9.6.

We note that the results presented in [160] were obtained using the monodomain SEM framework. The only difference in the problem setup of [160] from the ongoing calculations is that [160] did not model the pipe attached to the bottom of the tank, and instead applied inhomogeneous Dirichlet conditions directly on the surface of the tank.

9.7.2 Motivation for multirate time-stepping

Due to the physics of the flow considered here, high-temperature fluid enters the tank through the pipe and is accelerated by buoyancy effects. As we can see in Fig. 9.50, due to the nature of buoyant plumes in a stably stratified environment, the critical regions of the plume are confined to a T-shaped region. This T-shaped region includes the region where the plume is driven by buoyancy and the region where the plume becomes neutrally buoyant and flows outwards. The flow is relatively less chaotic everywhere else in the domain.

Figure 9.51 shows the spectral element mesh used for the monodomain SEM calculation along with a slice view of the CFL distribution in the domain. We can observe that using a single conforming mesh leads to unnecessary resolution away from the region of interest, and overlapping grids can reduce the element count. The CFL distribution plot (right) shows that since the high-speed flow is confined to a specific portion of the domain, we can use the multirate time-stepping capability described in Chapter 6 to further reduce the computational cost of this calculation.

Here, we benchmark the Schwarz-SEM framework using a nonrotating singlephase plume and show that multirate time-stepping with overlapping grids is an effective way to solve this problem.

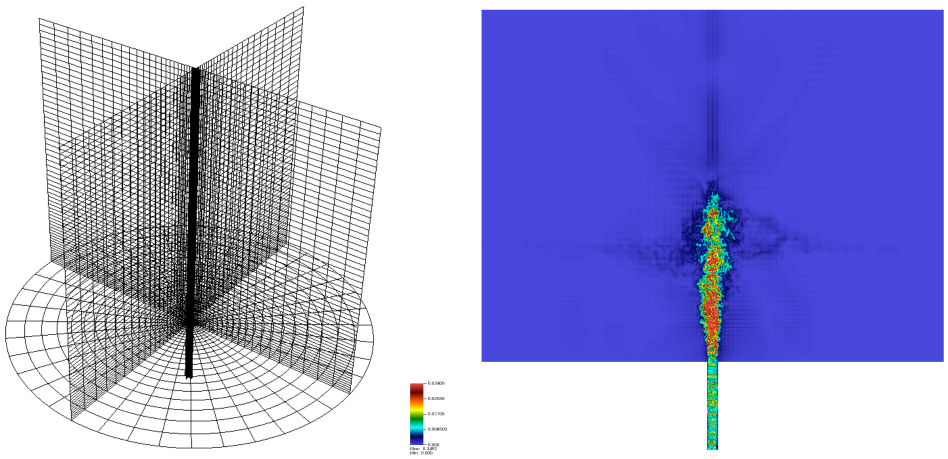


Figure 9.51: (left) Three-slice view of the monodomain mesh used to model the buoyant plume, and (right) slice view of the CFL distribution for the monodomain mesh.

9.7.3 Results

Figure 9.52 shows the spectral element meshes that were used for the monodomain and Schwarz-SEM calculations. The conforming mesh for monodomain SEM (left) has 70,400 elements and ongoing calculations have indicated that the resolution along the plume centerline is not sufficient. Consequently, the inner mesh (Fig. 9.52(right)) used in the Schwarz-SEM framework has about 35% more grid points in the plume region, in comparison to the monodomain mesh. The outer mesh has 15,560 elements and it is about 50% coarser in comparison to the monodomain mesh in that region. Thus, we see that overlapping grids allow us to increase the resolution in the region of interest while keeping the total element count similar to the monodomain mesh.

Using the multirate time-stepping method described in Chapter 6, two different time-step ratio (η) are used for the Schwarz-SEM framework; $\eta = 5$ and $\eta = 50$. Due to multirate time-stepping, the coarser grid has to integrate the INS for many fewer time-steps as compared to the finer inner grid. As a result, we can further reduce the computational cost of the calculation by using many fewer MPI ranks for multirate time-stepping in comparison to the MPI ranks needed for single-rate time-stepping for this problem (Section 8.2).

Following the monodomain calculation, the polynomial order was set to $N = 7$ for the overlapping grid calculation. Since the overlap region is away from the area of interest, we set $Q = 1$ with $m = 1$, and the flow statistics were temporally-averaged over more than 24 convective time units. Figure 9.53 shows the temporally and spatially averaged (azimuthally averaged) velocity magnitude contours from the overlapping grid ($\eta = 5$ and 50) and monodomain calculations. We observe that there is a difference in the velocity

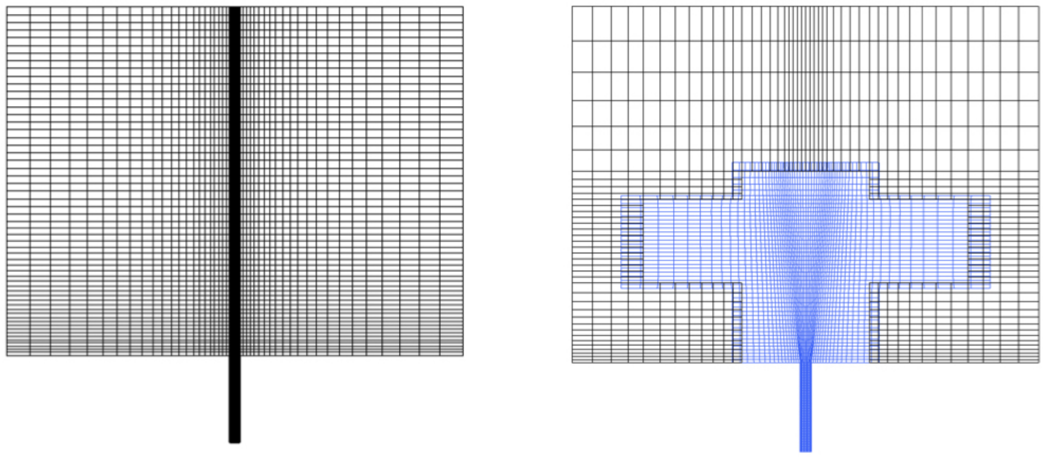


Figure 9.52: Slice view of the (left) single conforming and (right) overlapping spectral element meshes generated for the buoyant plume calculation.

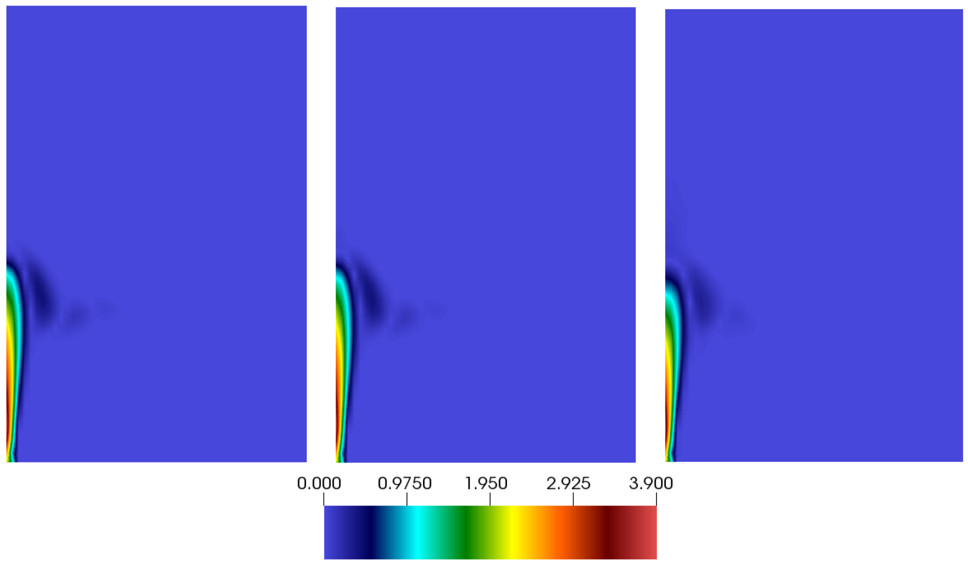


Figure 9.53: Temporally and spatially averaged velocity magnitude contours for the Schwarz-SEM calculations with overlapping grids using (left) $\eta = 5$ and (center) $\eta = 50$, and (right) the monodomain SEM calculation.

magnitude plots between the overlapping grids and monodomain calculation. Specifically, we notice that the maximum plume height, z_{max}, seems larger for the Schwarz-SEM framework calculations due to a relatively higher velocity magnitude along the plume centerline. This difference in the velocity magnitude is likely due to insufficient resolution in the spectral element mesh that was used for the monodomain calculation. We note that the results for the Schwarz-SEM calculations with different time-step ratios, shown in Fig. 9.53(a) and Fig. 9.53(b), agree well with each other.

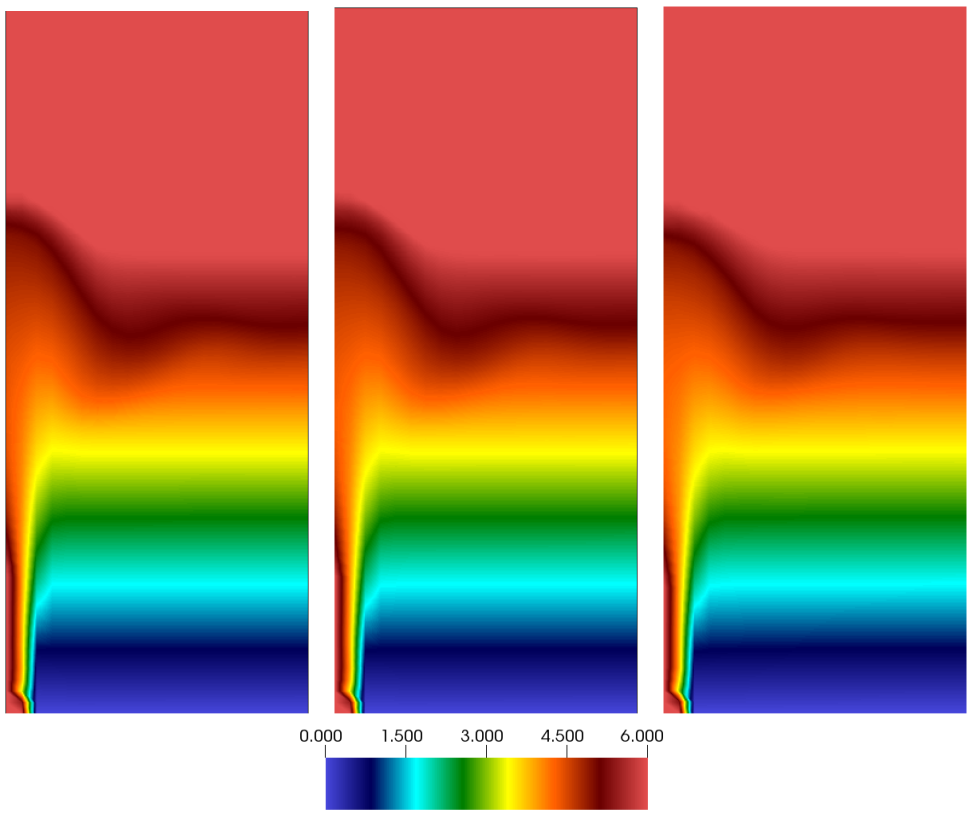


Figure 9.54: Temporally and spatially averaged temperature ($\overline{T} = T_r + \Gamma z + \overline{\theta}$) contours for the Schwarz-SEM calculations with overlapping grids using (left) $\eta = 5$ and (center) $\eta = 50$, and (right) monodomain SEM calculation.

Figure 9.54 shows the temporally and spatially averaged temperature perturbation contours from the overlapping grid and monodomain calculations. We can see that the temperature perturbation results agree well for the three cases considered here. Similarly, we also see a good comparison for these three cases for the turbulent kinetic energy (TKE) of the flow, which is shown in Fig. 9.55. We note that the monodomain results for the TKE are not as smooth as the results for the overlapping grid calculation, which is likely due to a lack of resolution in the plume region or due to insufficient duration over which the flow statistics were averaged (18 convective time units).

Using the axial velocity and temperature perturbation results, we can obtain the maximum height of the plume (z_{max}) and equilibrium height (z_{eq}) by plotting the mean axial velocity (W_{mean}) and mean temperature perturbation ($\overline{\theta}$) at the plume centerline. We can also use the TKE plots to visually obtain the plume trapping height. The line plots comparing W_{mean} and $\overline{\theta}$ along the plume centerline are shown in Fig. 9.56. The plot for W_{mean} agrees well with our expectation that the plume centerline velocity is higher, based on the velocity magnitude plot shown in Fig. 9.53. The plot for $\overline{\theta}$ shows good comparison for the

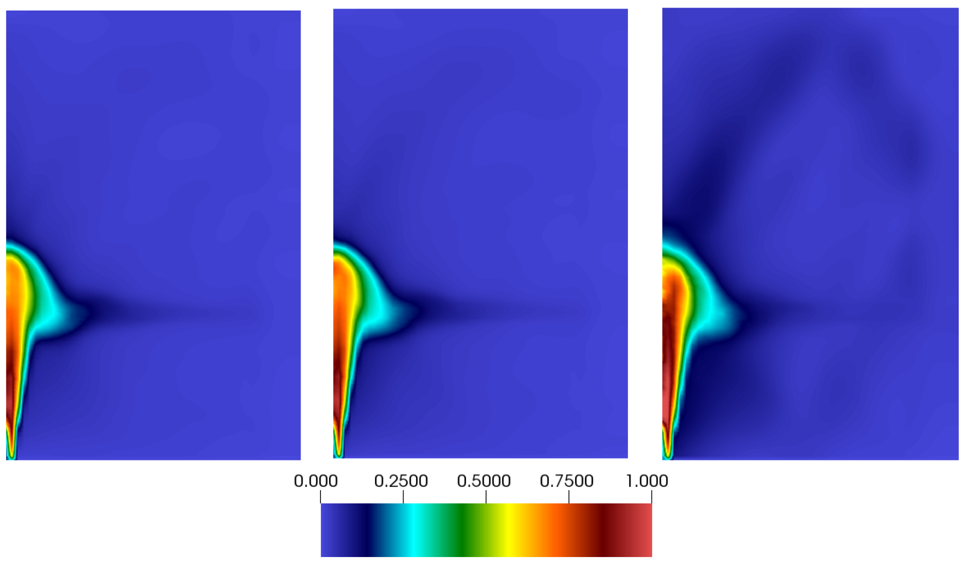


Figure 9.55: Temporally and spatially averaged TKE plots for the Schwarz-SEM calculations with overlapping grids using (left) $\eta = 5$, (center) $\eta = 50$ and (right) monodomain SEM calculation.

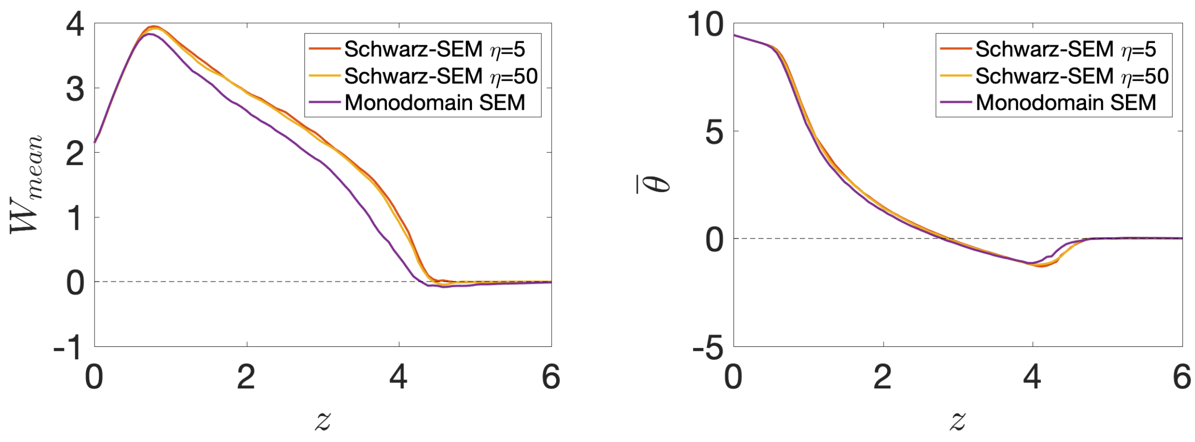


Figure 9.56: Time and spatially averaged mean (left) axial velocity (W_{mean}) and (right) temperature perturbation ($\overline{\theta}$) along the plume centerline.

three cases considered here.

Table 9.7 compares the maximum plume height (z_{max}), equilibrium height (z_{eq}), and trapping height (z_{th}) for the Schwarz-SEM results with the ongoing monodomain calculations and the monodomain SEM results by Fabregat et al. [160]. The maximum difference between the Schwarz-SEM calculations and monodomain calculation for the three parameters of interest that we consider here is 4.5% (for z_{max}). We reiterate that the difference in the monodomain calculation with the overlapping grid calculation is likely due to a lack of resolution in the monodomain mesh. We will be analyzing this difference in the future using monodomain

	z_{max}	z_{eq}	z_{th}
Fabregat et al.	4.5	2.85	3.11
Schwarz-SEM $\eta = 5$	4.51	2.88	3.1
Schwarz-SEM $\eta = 50$	4.45	2.85	3.1
Monodomain SEM	4.29	2.79	3.2

Table 9.7: z_{max}, z_{eq} and z_{th} obtained from the Schwarz-SEM framework, the current monodomain calculation, and the monodomain calculation by Fabregat [160].

and overlapping grid calculations at a higher resolution to ensure spatial convergence for each calculation.

9.7.4 Summary

In this section, we have used the multirate time-stepping scheme (Chapter 6) to model a singlephase thermally-buoyant nonrotating plume. Two different time-step size ratios, $\eta = 5$ and 50, were considered, and we see that the results from the Schwarz-SEM framework agree well with the monodomain calculation, in general. The buoyant plume example serves to show several advantages of the Schwarz-SEM framework. First, overlapping grids allow us to increase the resolution in the region of interest while keeping the total element count similar to the monodomain grid. Second, the multirate time-stepping scheme results for $\eta = 5$ and 50 agree well with each other. This independence of the results from η gives us confidence in using even higher time-step ratios. Third, multirate time-stepping requires fewer computational resources than the single-rate time-stepping scheme, which further increases the computational savings due to the Schwarz-SEM framework.

The results presented in this section show that the multirate time-stepping scheme that we have developed can accurately model turbulent flow in complex domains. This capability is crucial for analyzing rotating multiphase plumes in cross-flow, where the domain size is much larger than the domain that we consider in this section. Monodomain SEM calculations are intractable for this target problem due to their computational cost and the Schwarz-SEM framework provides an effective alternative.

9.8 Flow over a Wall-Mounted Cube

This problem is being studied in collaboration with scientists at the Argonne National Laboratory, who are analyzing flow over roughnesses. The monodomain calculations discussed here have been done by Dr. Ramesh Balakrishnan.

Flow over a roughness or multiple roughnesses is a problem of significant impact in the area of computational fluid dynamics (e.g., aerodynamic applications such as airplanes) and has been studied for decades. The literature in this area is abundant [162,163,164,165], and it is beyond the scope of the dissertation to discuss the different aspects of this class of problems. Instead, we focus on the effectiveness of the Schwarz-SEM framework in capturing the physics of the flow over a wall-mounted cube and compare our results with the monodomain DNS calculations that have been done at the Argonne National Lab.

9.8.1 Problem setup

Figure 9.57 shows the λ_2 vortices for flow over a cube roughness at $Re_H = 3900$. The Reynolds number, $Re_H = U_h H/\nu$, is based on the flow speed (U_H) at the roughness height (H) if the roughness was not present, and on the kinematic viscosity (ν). The domain is periodic in the spanwise direction (z), with inflow and outflow in the streamwise direction (x) and no-slip walls at the top and bottom in the wall-normal direction (y). The monodomain mesh (Fig. 9.57(center)) has 150,000 spectral elements, and the domain size is $L_x \times L_y \times L_z = 14H \times 3H \times 7H$. The center of the roughness is $3.5H$ from the inflow surface in x, and $3.5H$ from the spanwise periodic boundaries.

Fig. 9.57(bottom) shows a slice-view of the instantaneous velocity magnitude contours and as we can see, the laminar flow transitions to turbulence as the flow goes over the roughness, and the turbulent structures are mainly confined in the lower half of the domain in y. Using a monodomain mesh leads to unnecessary streamwise and spanwise resolution away from the roughness. Additionally, using a standard domain decomposition approach leads to a mesh with high aspect ratio elements, as shown in Fig. 9.57(center), that degrade the computational performance of the flow solver.

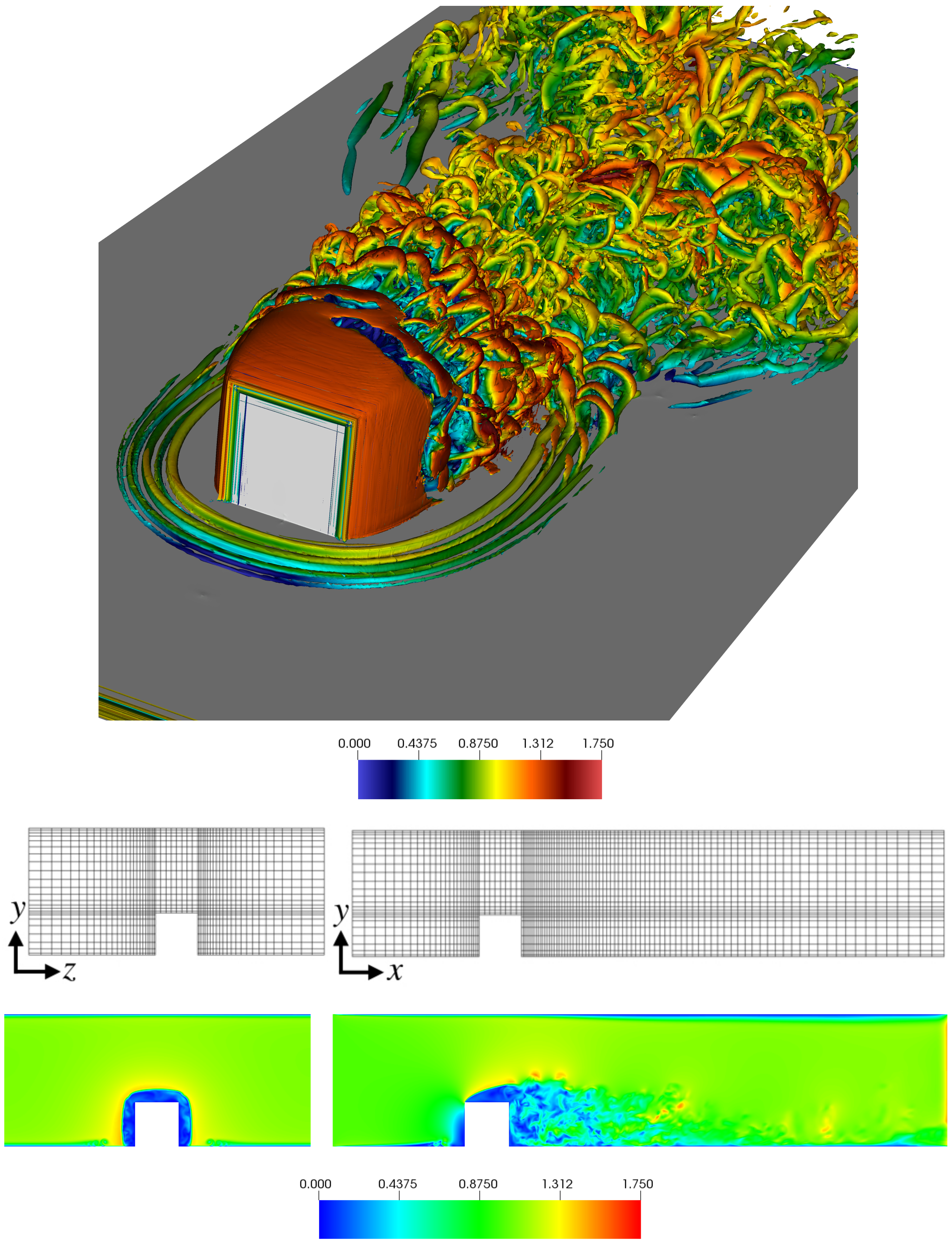


Figure 9.57: (top) λ_2 vortices colored by velocity magnitude contours for flow at $Re_H = 3900$, (center) slice view of the monodomain grid used for flow over a roughness, and (bottom) velocity magnitude contours.

In order to leverage that the physics of this problem confines the flow structures of interest in the lower half of the domain, we model the domain using two overlapping grids, which are shown in Fig. 9.58. The lower mesh discretizing the roughness has 90,000 elements and the upper mesh has only 30,000 elements in the far-field. Overlapping grids reduce the total element count by 20%. We note that we use a higher resolution near the roughness, in comparison to the monodomain grid, and smooth the mesh to get rid of the performance degrading elements. The overlap width between the two grids is $0.25H$, with $\Omega_y = [0, 1.75H]$ for the lower subdomain and $\Omega_y = [1.5H, 3H]$ for the upper subdomain.

Following the monodomain calculation, the polynomial order was set to $N = 7$, and flow statistics were averaged over 666 convective time units for both calculations. For the Schwarz-SEM framework, we set $Q = 0$ and $m = 1$. Figure 9.58(bottom) shows a snapshot of the instantaneous velocity magnitude plot for the calculation using overlapping grids.

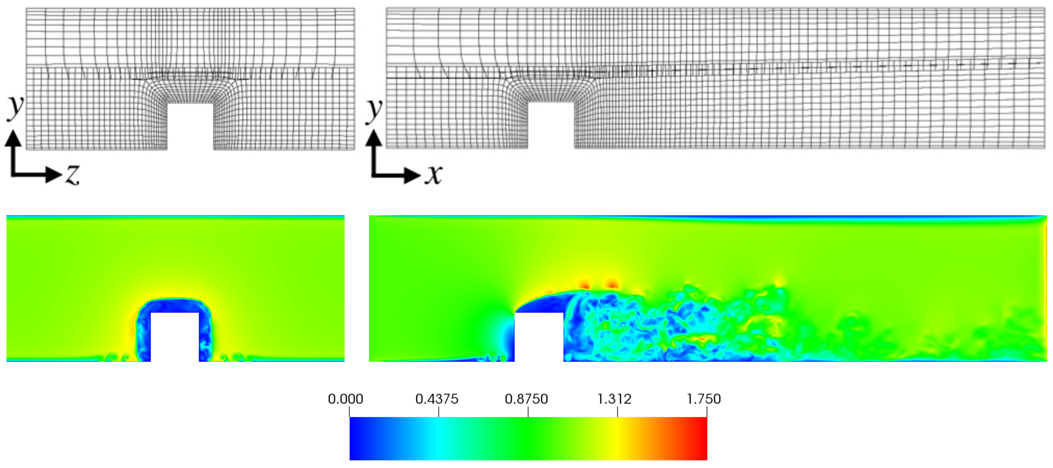


Figure 9.58: (center) Slice view of the overlapping grids used for flow over a roughness, and (bottom) velocity magnitude contours.

9.8.2 Results

As a first step towards validating the overlapping grid calculation with the monodomain calculation, we compare the mean streamwise velocity and RMS velocities for the two calculations. Figure 9.59 compares the time averaged velocity magnitude between the monodomain grid and overlapping grid calculation, and Fig.9.60-9.62 compares the RMS velocities in the streamwise (u_{rms}), wall-normal (v_{rms}), and spanwise (w_{rms}) direction, respectively, for the two cases considered here.

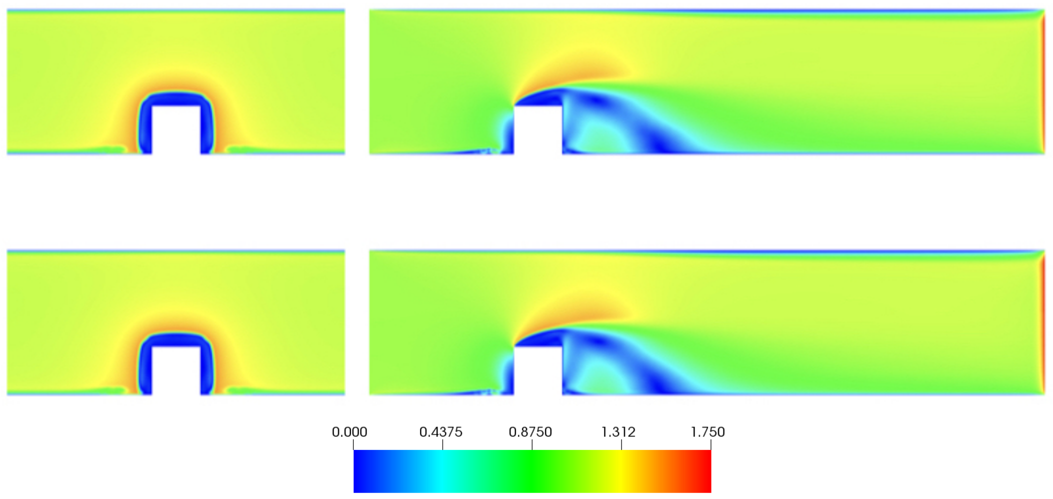


Figure 9.59: Ensemble averaged velocity magnitude for the (top) monodomain and (bottom) overlapping grid calculations.

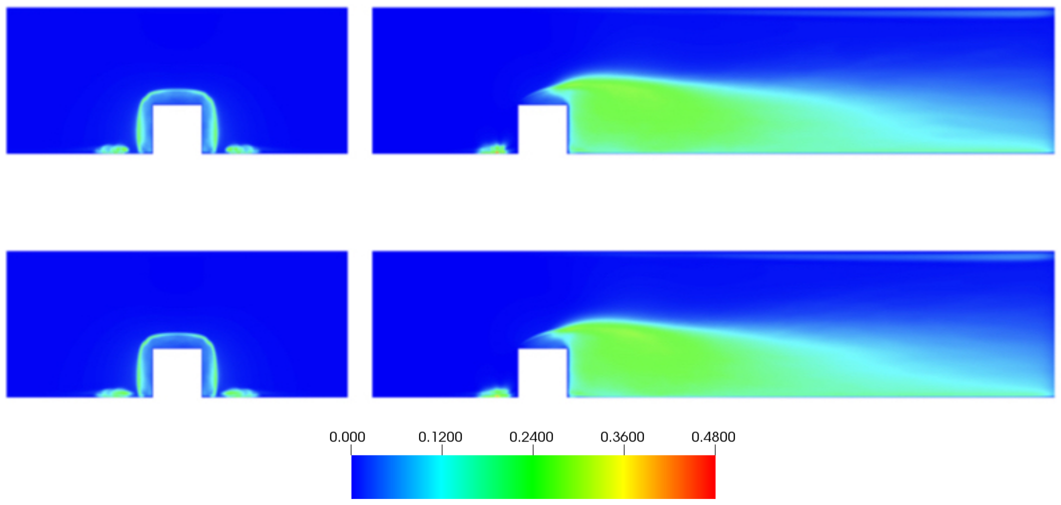


Figure 9.60: Streamwise RMS velocity u_{rms} for the (top) monodomain and (bottom) overlapping grid calculations.

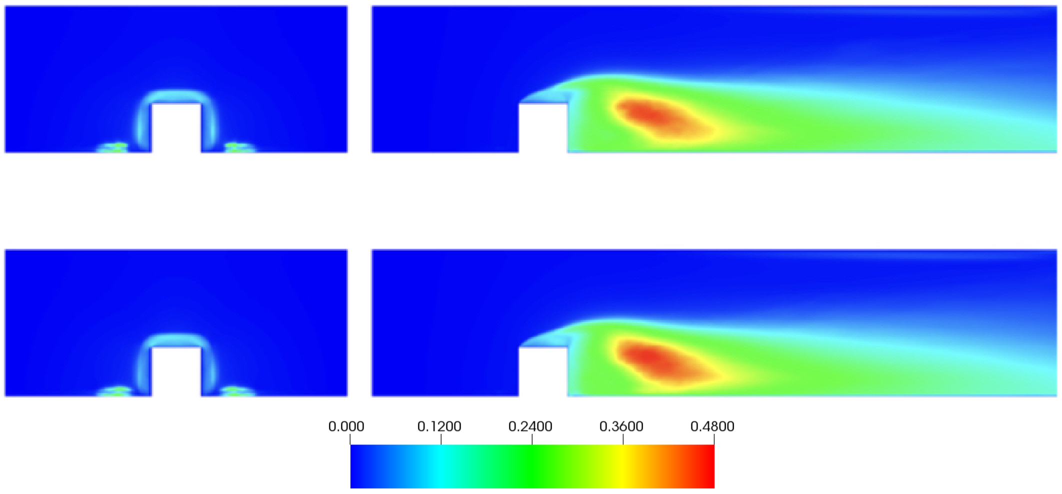


Figure 9.61: Wall-normal RMS velocity v_{rms} for the (top) monodomain and (bottom) overlapping grid calculations.

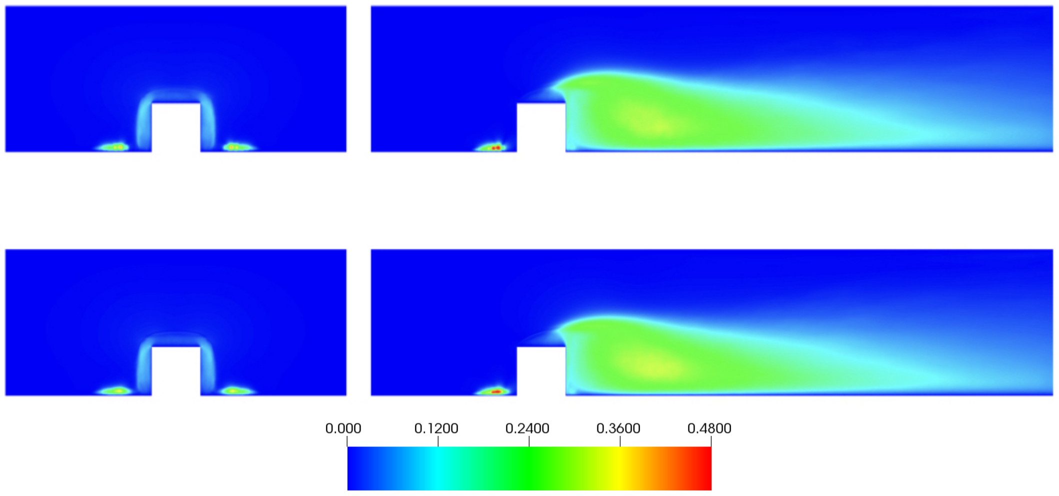


Figure 9.62: Spanwise RMS velocity w_{rms} for the (top) monodomain and (bottom) overlapping grid calculations.

As we can see in the results presented in Fig. 9.59-9.62, the Schwarz-SEM framework shows a good comparison for first- and second-order statistics with the monodomain calculation. As a part of the ongoing collaboration with researchers at ANL, our goal is to further quantify the comparison between these two calculations. In future work, we will be looking at quantities such as the skin friction distribution along the lower wall to compare the monodomain and overlapping grid calculations. Ongoing calculations at Argonne National Lab are already using the Schwarz-SEM framework with $S = 3$ overlapping subdomains to analyze laminar-to-turbulent flow transition at $Re_H = 20,000$, for which monodomain calculations are prohibitively expensive.

9.9 Mesh Smoothing

In this section, we demonstrate the effectiveness of the mesh smoother, described in Chapter 7, in improving the computation performance of spectral element meshes. The results presented in this section have been published in [73].

For each example that we present in this section, we show the spectral element mesh before and after smoothing. We also indicate the improvement in the iterations for the solution of the pressure Poisson equation (N_{iter}) and the iterative condition number (κ) of the pressure-solve system (Section 7.3). The N_{iter} and κ values reported in this section are based on data averaged over hundreds of time-steps for each example. In order to ensure a fair comparison between the original and smoothed meshes, each case was run with the same parameters, such as time-step size and tolerances for pressure and velocity solve. We summarize the results for all the cases considered here, at the end of this section.

9.9.1 Flow past a half-cylinder

Figure 9.63 shows the mesh constructed for flow past a cylinder in the half-domain, which has been used in the past for testing the performance of various preconditioners for the pressure Poisson solve [14]. This mesh contains a spectrum of element shapes and sizes, which make it effective for testing the convergence behavior of iterative solver. In this case, the calculation was run only for 1 time-step as the pressure solution is most expensive to compute for the first time-step.

In these tests, the base mesh of $E_s = 93$ elements (Fig. 9.63(left)) was quad-refined (each quad element split into 4 elements) to obtain mesh with $E = 372$ and 1488 elements. Smoothing was applied (as shown in Fig. 9.63, right) to yield improvements in condition number κ_{iter} of 76.2 , 67.9, and 72.7 % for respective element counts of E=93, 372, and 1488. The corresponding reductions in iteration count (N_{iter}) were 43.2, 43.2, and 46.6%.

9.9.2 Low pressure turbine blade

Figure 9.64 shows the 2D mesh constructed for flow past an LPT-106 turbine blade. The domain is periodic in the pitchwise direction, which requires that the elements on those boundaries have the same streamwise coordinate. This periodicity constraint leads to mesh skewness, which manifests itself the most near the

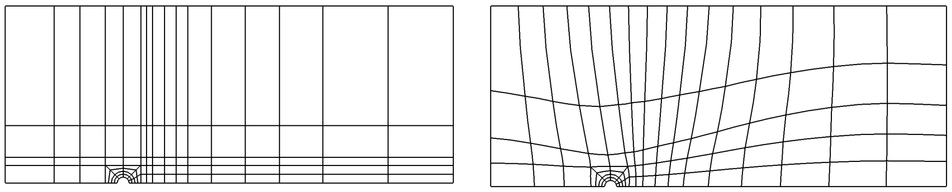


Figure 9.63: Half-cylinder mesh before and after smoothing

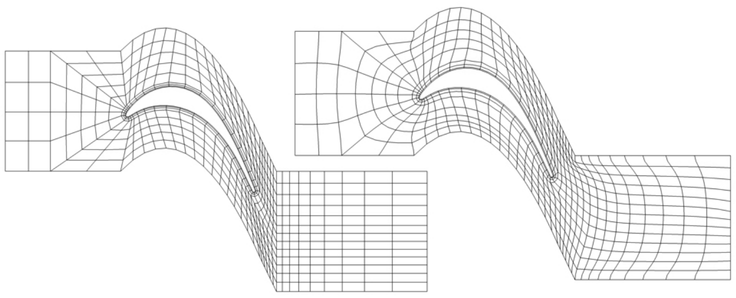


Figure 9.64: LPT blade mesh before and after smoothing

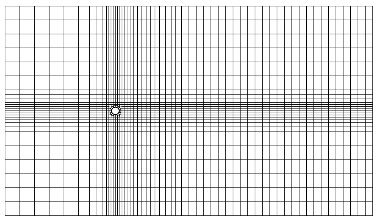
leading and trailing edge of the blade, and adversely impacts the CFL of the grid and performance of the pressure Poisson solver. As evident in Fig. 9.64, our mesh smoother makes the mesh homogeneous and preserves the boundary layer resolving elements of the mesh during the smoothing process. The global objective function, ϕ (7.4), is reduced by 66% by the mesh smoother.

Mesh smoothing leads to an overall decrease in κ_{iter} by 21.2% and reduces the pressure iteration count by 12.9%. Additionally, the CFL of the mesh is improved by 9.8% by the mesh smoother, which allows the use of a larger step size to time-advance the solution of the INSE.

9.9.3 Flow over a cylinder

Figure 9.65(a) shows the mesh constructed for analyzing the flow past a cylinder. There are 1472 elements in the mesh and, as is evident in Fig. 9.65(a), the element sizes are nonuniform with bands of thin elements extending from the cylinder to domain boundaries. The mesh anisotropy that is prominent around the cylinder, is significantly diminished through smoothing, as evident in Fig. 9.65(b). With use of surface smoothing on all exterior-boundaries of the domain, mesh smoothing reduces ϕ by 36%.

As a result of mesh smoothing, the relative magnitude of the maximum and minimum eigenvalue of the upper Hessenberg matrix decreases by 60.3%, which results in decreasing the mean pressure iteration count by 25.2%.


(a) Original Mesh

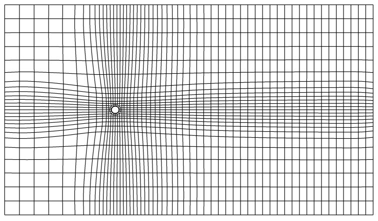

(b) Smoothed mesh

Figure 9.65: Comparison of original and smooth mesh for flow over a 2D cylinder

9.9.4 Boundary layer flow over a cylinder

Figure 9.66 shows the slice view of a 3D mesh (83,598 spectral elements) constructed for oscillatory flow over a cylinder. Figure 9.67 shows the mesh around the cylinder before and after smoothing. As we can see in Figure 9.67(a), the mesh contours are not smoothly changing at the boundaries of different geometry decomposition regions that were used to construct the mesh. Mesh smoothing makes the transition smoother and makes the elements uniformly sized, which leads to an improvement of 7.4% in κ_{iter} and 4% in N_{iter}.

9.9.5 Flow over a hemi-sphere

Figure 9.68(a) shows the mesh constructed for studying flow over a hemisphere at $Re = 2500$. There are a total of 2072 elements in this mesh. Figure 9.68(b) shows the improvement in mesh quality due to mesh smoothing, where ϕ is reduced by 53%. Consequently, mesh smoothing reduces κ_{iter} and N_{iter} by about 17.2% and 6.8%, respectively.

9.9.6 Piston cylinder

Figure 9.69(a) shows a slice view of the mesh for an internal combustion engine's (ICE) cylinder. As evident in Fig. 9.69(b), the original mesh has thin elements in the far-field because of the boundary layer resolving elements below the valve stem. Smoothing this mesh alleviates this issue and makes the mesh more uniform,

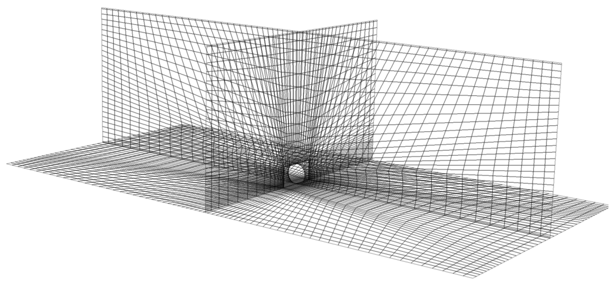
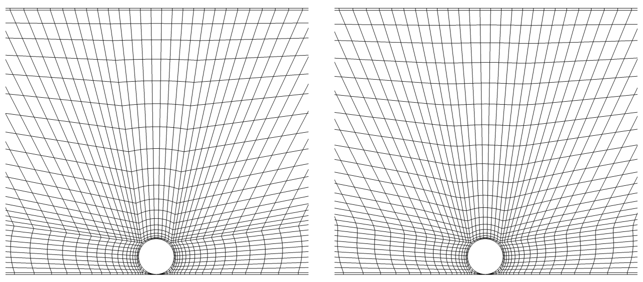
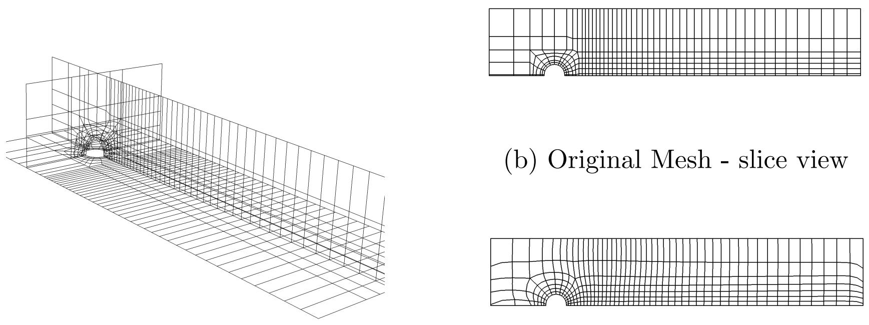


Figure 9.66: Three-slice view of a 3D Mesh for oscillatory flow over a cylinder



(a) Original mesh (b) Smoothed mesh

Figure 9.67: Slice view of the mesh before and after smoothing for oscillatory flow over a cylinder



(b) Original Mesh - slice view

(a) Original mesh -Three-slice view (c) Smoothed mesh -slice view

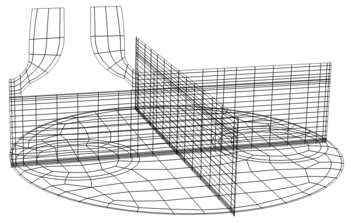
Figure 9.68: Comparison of original and smooth mesh for flow over a hemi-sphere

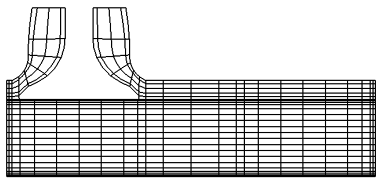
as shown in Fig. 9.69(c), which decrease ϕ by 71%. We note here that this calculation is an ALE application where the base of the intake valve and cylinder moved based on the relation between the crank angle and the piston height for a full operating-cycle of an engine.

As shown in Fig. 7.5, mesh smoothing preserves the boundary layer resolution of the mesh on the valve stem. Consequently, numerical experiments show a reduction of 39.8% and 15% in κ_{iter} and N_{iter}, respectively.

9.9.7 Summary

The mesh smoother presented in Chapter 7 has been designed to maximize the computational efficiency of SEM-based calculations. By using a combination of Laplacian smoothing and optimization-based approach, the mesh smoother gets rid of performance degrading elements and decreases the overall run-time of the calculation. Table 9.8 summarizes the results from various examples that we have presented for mesh smoothing. For each case, we present the improvement in iterative condition number ($\Delta\kappa$), the number of


(a) Original mesh - Three-slice view


(b) Original Mesh - slice view

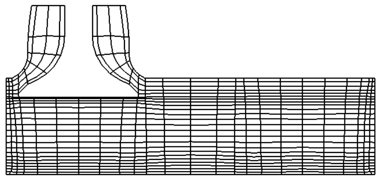

(c) Smoothed mesh -slice view

Figure 9.69: Comparison of original and smooth mesh for flow in an ICE

pressure iterations per time-step (ΔN_{iter}), and the overall run-time (ΔT) of the calculation. The results reported here are obtained by averaging data from hundreds of time-steps for each example.

Case	E	N_{orig}	N_{smooth}	$\Delta N_{iter}\%$	κ_{orig}	κ_{smooth}	$\Delta\kappa\%$	$\Delta T\%$
Half-cylinder	93	44	25	43.2	35.1	8.4	76.2	20.9
Half-cylinder	372	37	21	43.2	64.1	20.6	67.9	25.4
Half-cylinder	1488	73	39	46.6	115.5	31.5	72.7	33.3
LPT	532	6.4	5.6	12.9	110.7	87	21.2	6.3
Cylinder	1472	7.1	5.3	21.2	9.3	3.7	60.3	8.7
Oscillating flow	83598	21.3	20.4	3.9	1094.8	1014.3	7.4	2.2
Hemi-sphere	2072	4.19	3.9	6.8	14.6	12.1	17.2	1.5
Piston cylinder	6784	22.7	19.3	15.0	97.5	58.7	39.8	8.3

Table 9.8: Comparison of number of iterations (N_{iter}), iterative condition number (κ) and overall run-time (T) for original and smoothed meshes with the number of spectral elements (E) for each case.

The mesh smoothing method that is presented here has been made available open-source through the incompressible flow solver, Nek5000. Researchers have reported as much as 40% improvement in the computational efficiency of their SEM-based calculations through the use of our mesh smoothing method.

Chapter 10

Conclusion & Future Work

In this chapter, we present a summary of the scientific contributions made by this dissertation and describe possible avenues that we plan to explore in the future.

10.1 Summary

In this dissertation, we have presented the latest developments to the Schwarz-SEM framework. The starting point of this work is the framework presented in [24], which is based on a predictor-corrector scheme for time-advancing the solution of the incompressible Navier-Stokes equations in overlapping grids.

In Chapter 3, we have extended the Schwarz-SEM framework to support an arbitrary number of overlapping subdomains. This extension is made possible by introducing discriminators in a high-order interpolation library, *findptslib*, thich was originally developed for the monodomain SEM. These discriminators enable *findpts* to discriminate between elements of different overlapping subdomains and choose computational coordinates for each interdomain boundary grid point to maximize the convergence of the solution during the Schwarz iterations. We have also improved the computational performance of the interpolation routine in *findptslib*, *findpts_eval*, by reducing redundant sorting and communication cost for interpolating interdomain boundary data at each time-step. Using these latest improvements, we showed that the improved Schwarz-SEM framework maintains the spatial- and temporal-convergence of the monodomain SEM solver.

Chapter 4 presented a novel velocity correction technique that ensures that use of interpolation for the interdomain boundary data does not violate the divergence-free constraint of the INSE. Numerical experiments show that this mass flux based correction reduces the number of corrector iterations that are required at each time-step, thus improving the computational efficiency of the Schwarz-SEM framework. Here, we also describe our methodology for maintaining fixed flow-rate through overlapping subdomains, a capability that is crucial for modeling internal flow in periodic domains. A novel global integration method is presented, which generates partition-of-unity functions in subdomains with an arbitrary overlap.

We have used matrix stability analysis in Chapter 5 to analyze the stability properties of the predictor-correct scheme for time-advancing the solution of the unsteady heat equation in overlapping grids. Using a simple 1D example, we have showed that odd number of corrector iterations (Q) are more stable than even-Q, when high-order extrapolation is used for the interdomain boundary data at the predictor step. This stability analysis has allowed us to qualitatively capture the stability behavior that we had observed in the Schwarz-SEM framework for solving the INSE, and has also led to the development of a novel PC Scheme with superior stability properties.

Overlapping grids enable the use of grids of varying resolution to efficiently capture flow with a wide range of spatial and temporal scales. In order to leverage the difference in the scales of the flow in different subdomains, we have developed a novel multirate time-stepping scheme for the incompressible Navier-Stokes equations. In Chapter 6, we present this predictor-corrector-based multirate time-stepping scheme, which simultaneously time-advances the solution to the INSE in two overlapping subdomains using different time step sizes. Here, we also extend the stability analysis framework for the single-rate scheme from Chapter 5 to analyze the stability properties of the multirate scheme. This stability analysis gives us a novel insight into the impact of increasing the time-step ratio, η, to as high as 10. We find that the general stability behavior of the multirate scheme becomes independent of η for $\eta \geq 3$. Finally, we demonstrated that the multirate time-stepping scheme preserves the spatial- and temporal-convergence of the monodomain SEM solver.

Chapter 7 presented our mesh smoothing method, which is based on a combination of Laplacian smoothing and constrained optimization. Since the quality of a mesh is constrained by the surface mesh, the mesh smoother uses a novel approach for surface smoothing while making sure that the geometrical approximation errors are minimized during the smoother iterations. Chapter 7 also describes a method to monitor the conditioning of the pressure Poisson system to understand how mesh smoothing improves the computational efficiency of calculations. The novel element-by-element parallel mesh smoothing approach presented in this chapter has proven to be highly effective for large-scale spectral element meshes, and it is available open-source through the SEM-based solver, Nek5000.

In Chapter 8, we presented timing results that show how different parameters of the Schwarz-SEM framework impact the time to solution for turbulent flow problems. The results presented in this chapter indicate that the cost of each corrector iteration is significant, which emphasizes the importance of the improved predictor-corrector scheme that we have developed in Chapter 5. Here, we have also demonstrated the effectiveness of multirate time-stepping in reducing the computational cost of a calculation in comparison to the single-rate time-stepping scheme. With improvements to the interpolation routine, *findpts_eval* (Chapter 3), a 3X speed-up is realized in high-order interpolation for the interdomain boundary data. Finally, we show strong scaling results for the Schwarz-SEM framework on up to 40,000 MPI ranks. These scaling results indicate that donor element search and interpolation account for only 10% and 1%, respectively, of the total time to solution per time-step, for most turbulent flow calculations. Since the donor element search is required only once for nonmoving grids, its cost is negligible. We also notice that the scaling for *findpts* and *findpts_eval* is not ideal, but that is expected since not all elements on interdomain boundaries are mapped to separate MPI ranks.

The scientific contributions made through this dissertation are already having a real-world impact, as demonstrated in Chapter 9. The Schwarz-SEM framework has been validated against several challenging fluid-thermal applications, ranging from turbulent flow in a smooth channel to heat transer enhancement in noncircular pipes with wire-coil inserts. The overlapping grid method is being used in collaboration with researchers at different organizations to solve problems that are intractable with the monodomain SEM. Some of the problems that we have presented in this chapter are (i) flow over rotating ellipsoids (Section 9.4), (ii) impact of denticles on the drag experienced by a shark (Section 9.5), (iii) thermally buoyant plume in a stably stratified environment, which was modeled using the multirate time-stepping scheme with a time-step ratio of $\eta = 5$ and 50 (Section 9.7), and (iv) laminar-to-turbulent transition in flow over a wall-mounted cube (Section 9.8). In Section 9.9, we demonstrated the effectiveness of our mesh smoothing method in improving the computational efficiency of spectral element meshes. This mesh smoother is available open-source through Nek5000, and has reduced the computational cost of calculations by as much as 40%.

10.2 Future Work

The work done in this dissertation has helped advance the state-of-the-art of overlapping Schwarz-based methods for the incompressible Navier-Stokes equations. In addition to the scientific contributions that we have made through this dissertation, we have identified several avenues that we plan to explore in the future.

The results from stability analysis in Chapter 5 have shown that odd-Q are more stable than even-Q for the PC-based time-stepping methods in overlapping grids (e.g., Fig. 5.4). We have found that a high-order PC scheme exhibits this odd-even pattern even for ODEs, and preliminary results indicate that this difference in stability between odd- and even-Q might be a property of all high-order PC schemes. A survey of the literature shows that this property of PC schemes has not been analyzed, and we will look to determine the fundamental reasoning behind this difference in stability of odd- and even-Q. Based on the timing results that we have presented in Section 8.1, we will also look at PC schemes that can help us reduce the number of corrector iterations needed to ensure stability for high-order temporal accuracy in the Schwarz-SEM framework.

In Chapter 6, we have presented a novel parallel approach for a predictor-corrector based multirate time-stepping scheme for the INSE. In the current work, we restrict multirate time-stepping to only two overlapping subdomains ($S = 2$). In future work, we plan to extend this multirate time-stepping approach to $S > 2$. We also plan to develop an adaptive multirate time-stepping method, which can automatically switch between single-rate and multirate time-stepping scheme (with variable η) during a calculation, based on the CFL number of each subdomain.

In order to further expand the scope of the Schwarz-SEM framework, we plan on coupling LES/DNS calculations with unsteady Reynolds-Averaged Navier Stokes (RANS) simulations for modeling large-scale systems such as nuclear reactors. This capability will also require us to develop a more accurate method for determining the distance function because RANS based calculations rely on the determination of each grid point's distance from the nearest wall for *wall models.* We plan to address the issue of distance function by solving a time-dependent Eikonal equation, as discussed in Section 3.4. We also plan to couple the Schwarz-SEM framework with the discontinuous Galerkin based SEM (DGSEM) solver that is being developed at the Spectral Element Analysis Laboratory (SEAL) at UIUC. Coupling the Schwarz-SEM framework with the DGSEM solver will allow us to tackle problems featuring subsonic or supersonic flow, which are intractable in the Schwarz-SEM framework.

Appendix A

Impact of Grid Stretching on the Spatial Convergence in SEM

In order to understand how grid distortion impacts the spatial convergence in SEM, we consider the exact Navier-Stokes eigenfunctions in a periodic domain $\Omega = [0, 2\pi]^2$, derived by Walsh [94].

In Section 2.3, we had used the Navier-Stokes eigenfunctions to demonstrate the exponential convergence of the solution with polynomial order N, in SEM. A spectral element mesh with $E_x \times E_y$ equally-sized elements was used for the numerical experiments, where $E_x = E_y = 16$. The 16×16 mesh is shown in Fig. A.1(a).

Here, we take this spectral element mesh with 256 elements and apply a stretching of the form

$$\tilde{x} = x + \sigma y, \tag{A.1}$$

$$\tilde{y} = y, \tag{A.2}$$

where $\tilde{\mathbf{x}} = [\tilde{x}, \tilde{y}]$ and $\mathbf{x} = [x, y]$ are the modified and original mesh coordinates, respectively. In (A.1), σ is the grid-skewness factor. Figures A.1(b) and A.1(c) show the spectral element mesh with $\sigma = 1$ and 2, respectively.

The advantage of using this approach for grid stretching is that as σ is increased, spectral element edges that were originally vertical are stretched by a factor $\sqrt{1+\sigma^2}$, and the resolution is effectively decreased. As a result, we can use these modified meshes to understand how grid stretching impacts the error and spatial convergence for the SEM.

The initial conditions, boundary conditions, and parameters such as time-step size (Δt), polynomial

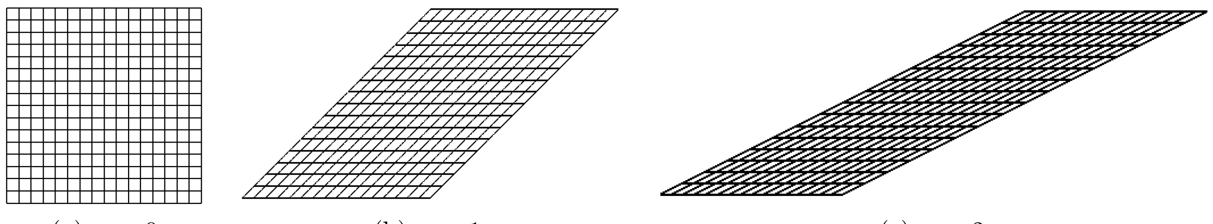
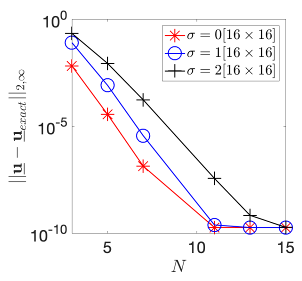
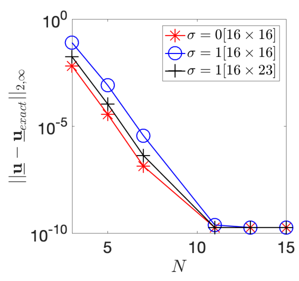

(a) $\sigma = 0$ (b) $\sigma = 1$ (c) $\sigma = 2$

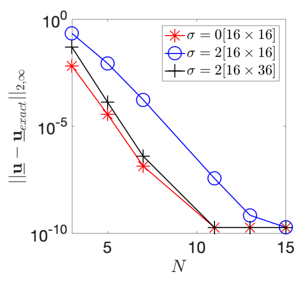
Figure A.1: Spectral element meshes used to understand how grid stretching impacts the spatial convergence and error in the SEM.



(a) Original meshes



(b) $\sigma = 0$ and 1



(c) $\sigma = 0$ and 2

Figure A.2: Spatial convergence of the solution with polynomial order N for different grids considered here.

order (N) and Reynolds number (Re) are the same that were used for the numerical experiment in Section 2.3. Figure A.2(a) shows the spatial convergence with N for the original grid ($\sigma = 0$) and the sheared grids ($\sigma = 1$ and 2). As we can see, the decrease in the effective resolution of the grid leads to an increase in the error. We note, however, that the exponential convergence of the solution is maintained despite the grid stretching.

To account for the grid stretching, we increase the resolution in y-direction by increasing the number of elements from E_y to $(\sqrt{1+\sigma^2}E_y)$ in the y-direction. This increase in resolution leads to a 16×23 mesh for $\sigma = 1$ and a 16×36 mesh for $\sigma = 2$. Figure A.2(b) shows the spatial convergence plot comparing the original mesh with $\sigma = 0$ and $\sigma = 1$, and the mesh for $\sigma = 1$ with increased resolution. As evident, increasing the grid resolution allows us to account for the grid stretching and recover the solution with the same accuracy that we get on the original grid with $\sigma = 0$. Similarly, Fig. A.2(c) shows that increasing the grid resolution in y-direction to roughly $\sqrt{5}$ times the resolution in the original mesh compensates for the decrease in resolution due to grid stretching.

References

[1] M. J. Berger and J. Oliger, "Adaptive mesh refinement for hyperbolic partial differential equations," *Journal of computational Physics*, vol. 53, no. 3, pp. 484–512, 1984.

[2] M. J. Berger and P. Colella, "Local adaptive mesh refinement for shock hydrodynamics," *Journal of computational Physics*, vol. 82, no. 1, pp. 64–84, 1989.

[3] J. Červenỳ, V. Dobrev, and T. Kolev, "Non-conforming mesh refinement for high-order finite elements," *arXiv preprint arXiv:1905.04033*, 2019.

[4] D. Rosenberg, A. Fournier, P. Fischer, and A. Pouquet, "Geophysical–astrophysical spectral-element adaptive refinement (gaspar): Object-oriented h-adaptive fluid dynamics simulation," *Journal of Computational Physics*, vol. 215, no. 1, pp. 59–80, 2006.

[5] C. Bernardi, Y. Maday, and A. T. Patera, "Domain decomposition by the mortar element method," in *Asymptotic and numerical methods for partial differential equations with critical parameters*, pp. 269–286, Springer, 1993.

[6] F. Ben Belgacem and Y. Maday, "The mortar element method for three dimensional finite elements," *ESAIM: Mathematical Modelling and Numerical Analysis-Modélisation Mathématique et Analyse Numérique*, vol. 31, no. 2, pp. 289–302, 1997.

[7] F. B. Belgacem, "The mortar finite element method with Lagrange multipliers," *Numerische Mathematik*, vol. 84, no. 2, pp. 173–197, 1999.

[8] A. Fluent *et al.*, "Ansys Fluent 12.0 user's guide," *Ansys Inc*, vol. 15317, pp. 1–2498, 2009.

[9] A. Bakker, R. D. LaRoche, M.-H. Wang, and R. V. Calabrese, "Sliding mesh simulation of laminar flow in stirred reactors," *Chemical Engineering Research and Design*, vol. 75, no. 1, pp. 42–44, 1997.

[10] A. Hansbo, P. Hansbo, and M. G. Larson, "A finite element method on composite grids based on Nitsche's method," *ESAIM: Mathematical Modelling and Numerical Analysis*, vol. 37, no. 3, pp. 495–514, 2003.

[11] A. Massing, M. G. Larson, and A. Logg, "Efficient implementation of finite element methods on nonmatching and overlapping meshes in three dimensions," *SIAM Journal on Scientific Computing*, vol. 35, no. 1, pp. C23–C47, 2013.

[12] B. Smith, P. Bjorstad, and W. Gropp, *Domain decomposition: parallel multilevel methods for elliptic partial differential equations*. Cambridge university press, 2004.

[13] M. Dryja, B. F. Smith, and O. B. Widlund, "Schwarz analysis of iterative substructuring algorithms for elliptic problems in three dimensions," *SIAM journal on numerical analysis*, vol. 31, no. 6, pp. 1662–1694, 1994.

[14] P. F. Fischer, "An overlapping Schwarz method for spectral element solution of the incompressible Navier–Stokes equations," *Journal of Computational Physics*, vol. 133, no. 1, pp. 84–101, 1997.

[15] H. A. Schwarz, *Ueber einen Grenzübergang durch alternirendes Verfahren.* Zürcher u. Furrer, 1870.

[16] K. Miller, "Numerical analogs to the Schwarz alternating procedure," *Numerische Mathematik*, vol. 7, no. 2, pp. 91–103, 1965.

[17] E. A. Volkov, "The method of composite meshes for finite and infinite regions with piecewise smooth boundary," *Trudy Matematicheskogo Instituta imeni VA Steklova*, vol. 96, pp. 117–148, 1968.

[18] D. R. Stoutemyer, "Numerical implementation of the Schwarz alternating procedure for elliptic partial differential equations," *SIAM Journal on Numerical Analysis*, vol. 10, no. 2, pp. 308–326, 1973.

[19] G. Starius, "Composite mesh difference methods for elliptic boundary value problems," *Numerische Mathematik*, vol. 28, no. 2, pp. 243–258, 1977.

[20] G. Starius, "On composite mesh difference methods for hyperbolic differential equations," *Numerische Mathematik*, vol. 35, no. 3, pp. 241–255, 1980.

[21] J. Benek, P. Buning, and J. Steger, "A 3D Chimera grid embedding technique," in *7th Computational Physics Conference*, p. 1523, 1985.

[22] J. Benek, T. Donegan, and N. Suhs, "Extended Chimera grid embedding scheme with application to viscous flows," in *8th Computational Fluid Dynamics Conference*, p. 1126, 1987.

[23] W. D. Henshaw, "A fourth-order accurate method for the incompressible Navier-Stokes equations on overlapping grids," *Journal of computational physics*, vol. 113, no. 1, pp. 13–25, 1994.

[24] B. E. Merrill, Y. T. Peet, P. F. Fischer, and J. W. Lottes, "A spectrally accurate method for overlapping grid solution of incompressible Navier–Stokes equations," *Journal of Computational Physics*, vol. 307, pp. 60–93, 2016.

[25] S. E. Rogers, D. Kwak, and C. Kiris, "Steady and unsteady solutions of the incompressible Navier-Stokes equations," *AIAA journal*, vol. 29, no. 4, pp. 603–610, 1991.

[26] S.-C. CD-adapco, "V7. 02.008," *User Manual*, 2012.

[27] J. Ahmad and E. P. Duque, "Helicopter rotor blade computation in unsteady flows using moving overset grids," *Journal of Aircraft*, vol. 33, no. 1, pp. 54–60, 1996.

[28] L. Cambier, S. Heib, and S. Plot, "The Onera elsA CFD software: input from research and feedback from industry," *Mechanics & Industry*, vol. 14, no. 3, pp. 159–174, 2013.

[29] S. Eberhardt and D. Baganoff, "Overset grids in compressible flow," in *7th Computational Physics Conference*, p. 1524, 1985.

[30] R. H. Nichols and P. G. Buning, "User's manual for OVERFLOW 2.1," *University of Alabama and NASA Langley Research Center*, 2008.

[31] O. Saunier, C. Benoit, G. Jeanfaivre, and A. Lerat, "Third-order Cartesian overset mesh adaptation method for solving steady compressible flows," *International journal for numerical methods in fluids*, vol. 57, no. 7, pp. 811–838, 2008.

[32] D. Blake and T. Buter, "Overset grid methods applied to a finite-volume time-domain Maxwell equation solver," in *27th Plasma Dynamics and Lasers Conference*, p. 2338, 1996.

[33] J. B. Angel, J. W. Banks, and W. D. Henshaw, "A high-order accurate FDTD scheme for Maxwell's equations on overset grids," in *Applied Computational Electromagnetics Society Symposium (ACES), 2018 International*, pp. 1–2, IEEE, 2018.

[34] F. Meng, J. W. Banks, W. D. Henshaw, and D. W. Schwendeman, "A stable and accurate partitioned algorithm for conjugate heat transfer," *Journal of Computational Physics*, vol. 344, pp. 51–85, 2017.

[35] K.-H. Kao and M.-S. Liou, "Application of chimera/unstructured hybrid grids for conjugate heat transfer," *AIAA journal*, vol. 35, no. 9, pp. 1472–1478, 1997.

[36] W. D. Henshaw and K. K. Chand, "A composite grid solver for conjugate heat transfer in fluid-structure systems," *Journal of Computational Physics*, vol. 228, no. 10, pp. 3708–3741, 2009.

[37] A. Koblitz, S. Lovett, N. Nikiforakis, and W. Henshaw, "Direct numerical simulation of particulate flows with an overset grid method," *Journal of Computational Physics*, vol. 343, pp. 414–431, 2017.

[38] C. J. Greenshields, "OpenFOAM user guide," *OpenFOAM Foundation Ltd, version*, vol. 3, no. 1, p. 47, 2015.

[39] S. B. Lee, "A study on temporal accuracy of OpenFOAM," *International Journal of Naval Architecture and Ocean Engineering*, vol. 9, no. 4, pp. 429–438, 2017.

[40] B. Courbet, C. Benoit, V. Couaillier, F. Haider, M. Le Pape, and S. Péron, "Space discretization methods," *AerospaceLab*, no. 2, pp. p–1, 2011.

[41] S. Domino, "Sierra low mach module: Nalu theory manual 1.0," *SAND2015-3107W, Sandia National Laboratories Unclassified Unlimited Release (UUR)*, 2015.

[42] N. Foster and R. Noack, "High-order overset interpolation within an OVERFLOW solution," in *50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, p. 728, 2012.

[43] F. Bassetti, D. Brown, K. Davis, W. Henshaw, and D. Quinlan, "Overture: an object-oriented framework for high performance scientific computing," in *Proceedings of the 1998 ACM/IEEE conference on Supercomputing*, pp. 1–9, IEEE Computer Society, 1998.

[44] A. Noorani, A. Peplinski, and P. Schlatter, "Informal introduction to program structure of spectral interpolation in nek5000," 2015.

[45] P. Fishcer, "gslib/gslib," 2017.

[46] M. J. Brazell, J. Sitaraman, and D. J. Mavriplis, "An overset mesh approach for 3D mixed element high-order discretizations," *Journal of Computational Physics*, vol. 322, pp. 33–51, 2016.

[47] R. Noack, "SUGGAR: a general capability for moving body overset grid assembly," in *17th AIAA computational fluid dynamics conference*, p. 5117, 2005.

[48] J. Sitaraman, M. Floros, A. Wissink, and M. Potsdam, "Parallel domain connectivity algorithm for unsteady flow computations using overlapping and adaptive grids," *Journal of Computational Physics*, vol. 229, no. 12, pp. 4703–4723, 2010.

[49] S. E. Rogers, N. E. Suhs, and W. E. Dietz, "PEGASUS 5: an automated preprocessor for overset-grid computational fluid dynamics," *AIAA journal*, vol. 41, no. 6, pp. 1037–1045, 2003.

[50] J. G. Coder, D. Hue, G. Kenway, T. H. Pulliam, A. J. Sclafani, L. Serrano, and J. C. Vassberg, "Contributions to the sixth drag prediction workshop using structured, overset grid methods," *Journal of Aircraft*, pp. 1–14, 2017.

[51] J. A. Crabill, J. Sitaraman, and A. Jameson, "A high-order overset method on moving and deforming grids," in *AIAA Modeling and Simulation Technologies Conference*, p. 3225, 2016.

[52] J. Aarnes, N. Haugen, and H. Andersson, "High-order overset grid method for detecting particle impaction on a cylinder in a cross flow," *arXiv preprint arXiv:1805.10039*, 2018.

[53] D. D. Chandar, "Assessment of interpolation strategies and conservative discretizations on unstructured overset grids in OpenFOAM," in *2018 AIAA Aerospace Sciences Meeting*, p. 0828, 2018.

[54] G. Chesshire and W. D. Henshaw, "A scheme for conservative interpolation on overlapping grids," *SIAM Journal on Scientific Computing*, vol. 15, no. 4, pp. 819–845, 1994.

[55] Y. T. Peet and P. F. Fischer, "Stability analysis of interface temporal discretization in grid overlapping methods," *SIAM Journal on Numerical Analysis*, vol. 50, no. 6, pp. 3375–3401, 2012.

[56] W. M. Chan, "Development of numerical methods for overset grids with applications for the integrated space shuttle vehicle," 1995.

[57] J. R. Rice, "Split runge-kutta methods for simultaneous equations," *Journal of Research of the National Institute of Standards and Technology*, vol. 60, 1960.

[58] E. M. Constantinescu and A. Sandu, "Multirate timestepping methods for hyperbolic conservation laws," *Journal of Scientific Computing*, vol. 33, no. 3, pp. 239–278, 2007.

[59] B. Seny, J. Lambrechts, R. Comblen, V. Legat, and J.-F. Remacle, "Multirate time stepping for accelerating explicit discontinuous Galerkin computations with application to geophysical flows," *International Journal for Numerical Methods in Fluids*, vol. 71, no. 1, pp. 41–64, 2013.

[60] S. A. Canann, M. B. Stephenson, and T. Blacker, "Optismoothing: An optimization-driven approach to mesh smoothing," *Finite Elements in analysis and Design*, vol. 13, no. 2-3, pp. 185–190, 1993.

[61] L. A. Freitag, "On combining Laplacian and optimization-based mesh smoothing techniques," *ASME APPLIED MECHANICS DIVISION-PUBLICATIONS-AMD*, vol. 220, pp. 37–44, 1997.

[62] G. Taubin, "Linear anisotropic mesh filtering," *Res. Rep. RC2213 IBM*, vol. 1, no. 4, 2001.

[63] S. A. Canann, J. R. Tristano, M. L. Staten, and T. Drive, "An Approach to Combined Laplacian and Optimization-Based Smoothing for Triangular, Quadrilateral, and Quad-Dominant Meshes.," in *IMR*, pp. 479–494, Citeseer, 1998.

[64] P. Knupp, "Introducing the target-matrix paradigm for mesh optimization via node-movement," *Engineering with Computers*, vol. 28, no. 4, pp. 419–429, 2012.

[65] P. M. Knupp, "A method for hexahedral mesh shape optimization ‡," *International journal for numerical methods in engineering*, vol. 332, no. November 2002, pp. 319–332, 2003.

[66] L. A. Freitag, P. Plassmann, *et al.*, "Local optimization-based simplicial mesh untangling and improvement," *International Journal for Numerical Methods in Engineering*, vol. 49, no. 1, pp. 109–125, 2000.

[67] P. M. Knupp, "Hexahedral Mesh Untangling & Algebraic Mesh Quality Metrics.," in *IMR*, pp. 173–183, Citeseer, 2000.

[68] L. Branets and G. F. Carey, "Extension of a mesh quality metric for elements with a curved boundary edge or surface," *Journal of Computing and Information Science in Engineering*, vol. 5, no. 4, pp. 302–308, 2005.

[69] A. Gargallo-Peiró, X. Roca, J. Peraire, and J. Sarrate, "Defining quality measures for validation and generation of high-order tetrahedral meshes," in *Proceedings of the 22nd International Meshing Roundtable*, pp. 109–126, Springer, 2014.

[70] A. Salem, S. Saigal, and S. A. Canann, "Mid-node admissible space for 3D quadratic tetrahedral finite elements," *Engineering with Computers*, vol. 17, no. 1, pp. 39–54, 2001.

[71] E. Ruiz-Gironés, X. Roca, and J. Sarrate, "High-order mesh curving by distortion minimization with boundary nodes free to slide on a 3D CAD representation," *Computer-Aided Design*, vol. 72, pp. 52–64, 2016.

[72] V. Dobrev, P. Knupp, T. Kolev, K. Mittal, and V. Tomov, "The target-matrix optimization paradigm for high-order meshes," *SIAM Journal on Scientific Computing*, vol. 41, no. 1, pp. B50–B68, 2019.

[73] K. Mittal and P. Fischer, "Mesh smoothing for the spectral element method," *Journal of Scientific Computing*, vol. 78, no. 2, pp. 1152–1173, 2019.

[74] T. Taniguchi and H. Katagiri, "New concept of hexahedral mesh generation for arbitrary 3d domain-block degeneration method," tech. rep., Mississippi State Univ., Mississippi State, MS (United States), 1996.

[75] S. Dey, R. M. O'bara, and M. S. Shephard, "Curvilinear mesh generation in 3D.," in *IMR*, pp. 407–417, 1999.

[76] X. Luo, M. S. Shephard, and J.-F. Remacle, "The influence of geometric approximation on the accuracy of high order methods," *Rensselaer SCOREC report*, vol. 1, 2001.

[77] X.-J. Luo, M. S. Shephard, R. M. O'bara, R. Nastasia, and M. W. Beall, "Automatic p-version mesh generation for curved domains," *Engineering with Computers*, vol. 20, no. 3, pp. 273–285, 2004.

[78] S. Sherwin and J. Peiró, "Mesh generation in curvilinear domains using high-order elements," *International Journal for Numerical Methods in Engineering*, vol. 53, no. 1, pp. 207–223, 2002.

[79] K. Mittal, S. Dutta, and P. Fischer, "Nonconforming Schwarz-spectral element methods for incompressible flow," *Computers & Fluids*, p. 104237, 2019.

[80] A. T. Patera, "A spectral element method for fluid dynamics: laminar flow in a channel expansion," *Journal of computational Physics*, vol. 54, no. 3, pp. 468–488, 1984.

[81] E. Merzari, A. Obabko, and P. Fischer, "Spectral element methods for liquid metal reactors applications," *arXiv preprint arXiv:1711.09307*, 2017.

[82] S. Dutta, D. Wang, P. Tassi, and M. H. Garcia, "Three-dimensional numerical modeling of the Bulle effect: the nonlinear distribution of near-bed sediment at fluvial diversions," *Earth Surface Processes and Landforms*, vol. 42, no. 14, pp. 2322–2337, 2017.

[83] R. Vinuesa, P. S. Negi, M. Atzori, A. Hanifi, D. S. Henningson, and P. Schlatter, "Turbulent boundary layers around wing sections up to Re_c= 1,000,000," *International Journal of Heat and Fluid Flow*, vol. 72, pp. 86–99, 2018.

[84] M. O. Deville, P. F. Fischer, and E. H. Mund, *High-order methods for incompressible fluid flow*, vol. 9. Cambridge University Press, 2002.

[85] B. A. Finlayson, *The method of weighted residuals and variational principles*, vol. 73. SIAM, 2013.

[86] E. Merzari, K. Heisey, and P. Fischer, "Nek deep dive,"

[87] Y. Saad, *Iterative methods for sparse linear systems*, vol. 82. siam, 2003.

[88] G. Fox and W. Furmanski, "Hypercube algorithms for neural network simulation: the Crystal_Accumulator and the Crystal Router," in *Proceedings of the third conference on Hypercube concurrent computers and applications: Architecture, software, computer systems, and general issues-Volume 1*, pp. 714–724, ACM, 1988.

[89] A. Tomboulides, J. Lee, and S. Orszag, "Numerical simulation of low Mach number reactive flows," *Journal of Scientific Computing*, vol. 12, no. 2, pp. 139–167, 1997.

[90] P. Fischer, M. Schmitt, and A. Tomboulides, "Recent developments in spectral element simulations of moving-domain problems," in *Recent Progress and Modern Challenges in Applied Mathematics, Modeling and Computational Science*, pp. 213–244, Springer, 2017.

[91] J. Malm, P. Schlatter, P. F. Fischer, and D. S. Henningson, "Stabilization of the spectral element method in convection dominated flows by recovery of skew-symmetry," *Journal of Scientific Computing*, vol. 57, no. 2, pp. 254–277, 2013.

[92] J. W. Lottes and P. F. Fischer, "Hybrid multigrid/Schwarz algorithms for the spectral element method," *J. Sci. Comput.*, vol. 24, pp. 45–78, 2005.

[93] P. Fischer and J. Lottes, "Hybrid Schwarz-multigrid methods for the spectral element method: Extensions to Navier-Stokes," in *Domain Decomposition Methods in Science and Engineering Series* (R. Kornhuber, R. Hoppe, J. Périaux, O. Pironneau, O. Widlund, and J. Xu, eds.), pp. 35–49, Springer, Berlin, 2004.

[94] O. Walsh, "Eddy solutions of the Navier–Stokes equations," in *The Navier-Stokes Equations II—Theory and Numerical Methods*, pp. 306–309, Springer, 1992.

[95] S. Dutta, K. Mittal, J. W. Lottes, and P. Fischer, "Efficient lagrangian particle tracking for high-order cfd in complex geometries," *Letter from Lead-Chairs*.

[96] V. Dobrev, F. Grogan, T. Kolev, R. Rieben, and V. Tomov, "Level set methods for detonation shock dynamics using high-order finite elements," Tech. Rep. LLNL-TR-732038, Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States), 2017.

[97] J. A. Sethian, *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*, vol. 3. Cambridge university press, 1999.

[98] H. Zhao, "A fast sweeping method for eikonal equations," *Mathematics of computation*, vol. 74, no. 250, pp. 603–627, 2005.

[99] P. Fishcer, "Nek5000/nek5000," 2017.

[100] G. K. El Khoury, P. Schlatter, A. Noorani, P. F. Fischer, G. Brethouwer, and A. V. Johansson, "Direct numerical simulation of turbulent pipe flow at moderately high Reynolds numbers," *Flow, turbulence and combustion*, vol. 91, no. 3, pp. 475–495, 2013.

[101] C. Kelley, "Iterative methods for linear and nonlinear equations," *Frontiers in applied mathematics*, vol. 16, pp. 575–601, 1995.

[102] H. J. Stetter, "Improved absolute stability of predictor-corrector schemes," *Computing*, vol. 3, no. 4, pp. 286–296, 1968.

[103] C. Gear, "Multirate methods for ordinary differential equations," Tech. Rep. COO-2383-0014, Illinois Univ., Urbana (USA). Dept. of Computer Science, 1974.

[104] A. Verhoeven, E. J. W. Ter Maten, R. M. Mattheij, and B. Tasić, "Stability analysis of the BDF slowest-first multirate methods," *International Journal of Computer Mathematics*, vol. 84, no. 6, pp. 895–923, 2007.

[105] C. W. Gear and D. Wells, "Multirate linear multistep methods," *BIT Numerical Mathematics*, vol. 24, no. 4, pp. 484–502, 1984.

[106] A. Klöckner, *High-performance high-order simulation of wave and plasma phenomena.* PhD thesis, Brown University, 2010.

[107] C. Mikida, A. Klöckner, and D. Bodony, "Multi-rate time integration on overset meshes," *arXiv preprint arXiv:1805.06607*, 2018.

[108] C. Engstler and C. Lubich, "Multirate extrapolation methods for differential equations with different time scales," *Computing*, vol. 58, no. 2, pp. 173–185, 1997.

[109] M. Emmett, W. Zhang, and J. B. Bell, "High-order algorithms for compressible reacting flow with complex chemistry," *Combustion Theory and Modelling*, vol. 18, no. 3, pp. 361–387, 2014.

[110] I. Rybak, J. Magiera, R. Helmig, and C. Rohde, "Multirate time integration for coupled saturated/unsaturated porous medium and free flow systems," *Computational Geosciences*, vol. 19, no. 2, pp. 299–309, 2015.

[111] C. J. Trahan and C. Dawson, "Local time-stepping in Runge–Kutta discontinuous Galerkin finite element methods applied to the shallow-water equations," *Computer Methods in Applied Mechanics and Engineering*, vol. 217, pp. 139–152, 2012.

[112] B. Fornberg, *A practical guide to pseudospectral methods*, vol. 1. Cambridge university press, 1998.

[113] V. A. Dobrev, T. V. Kolev, and R. N. Rieben, "High-order curvilinear finite element methods for Lagrangian hydrodynamics," *SIAM Journal on Scientific Computing*, vol. 34, no. 5, pp. B606–B641, 2012.

[114] W. J. Gordon and C. A. Hall, "Transfinite element methods: blending-function interpolation over arbitrary curved element domains," *Numerische Mathematik*, vol. 21, no. 2, pp. 109–129, 1973.

[115] P. Fischer, M. Schmitt, and A. Tomboulides, "Recent developments in spectral element simulations of moving-domain problems," in *Recent Progress and Modern Challenges in Applied Mathematics, Modeling and Computational Science*, pp. 213–244, Springer, New York, NY, 2017.

[116] Y. Maday and A. T. Patera, "Spectral element methods for the incompressible Navier-Stokes equations," in *IN: State-of-the-art surveys on computational mechanics (A90-47176 21-64). New York, American Society of Mechanical Engineers, 1989, p. 71-143. Research supported by DARPA.*, vol. 1, pp. 71–143, 1989.

[117] J. B. Perot, "An analysis of the fractional step method," *Journal of Computational Physics*, vol. 108, no. 1, pp. 51–58, 1993.

[118] Y. Maday, A. T. Patera, and E. M. Rønquist, "An operator-integration-factor splitting method for time-dependent problems: application to incompressible fluid flow," *Journal of Scientific Computing*, vol. 5, no. 4, pp. 263–292, 1990.

[119] J. W. Lottes and P. F. Fischer, "Hybrid multigrid/Schwarz algorithms for the spectral element method," *Journal of Scientific Computing*, vol. 24, no. 1, pp. 45–78, 2005.

[120] R. E. Lynch, J. R. Rice, and D. H. Thomas, "Direct solution of partial difference equations by tensor product methods," *Numerische Mathematik*, vol. 6, no. 1, pp. 185–199, 1964.

[121] P. F. Fischer, "Scaling limits for PDE-based simulation," in *22nd AIAA Computational Fluid Dynamics Conference*, 2015.

[122] C. Canuto, P. Gervasio, and A. Quarteroni, "Finite-element preconditioning of G-NI spectral methods," *SIAM Journal on Scientific Computing*, vol. 31, no. 6, pp. 4422–4451, 2010.

[123] M. T. Heath, *Scientific computing.* McGraw-Hill New York, 2002.

[124] L. A. Freitag and P. M. Knupp, "Tetrahedral mesh improvement via optimization of the element condition number," *International Journal for Numerical Methods in Engineering*, vol. 53, no. 6, pp. 1377–1391, 2002.

[125] M. Escudier, "Observations of the flow produced in a cylindrical container by a rotating endwall," *Experiments in fluids*, vol. 2, no. 4, pp. 189–196, 1984.

[126] R. D. Moser, J. Kim, and N. N. Mansour, "Direct numerical simulation of turbulent channel flow up to $\mathrm{Re}_\tau = 590$," *Physics of fluids*, vol. 11, no. 4, pp. 943–945, 1999.

[127] A. Vreman and J. G. Kuerten, "Comparison of direct numerical simulation databases of turbulent channel flow at $\mathrm{Re}_\tau = 180$," *Physics of Fluids*, vol. 26, no. (1), 015102, 2014.

[128] J. T. Collins, C. M. Conley, J. N. Attig, and M. M. Baehl, "Enhanced heat transfer using wire-coil inserts for high-heat-load applications.," Tech. Rep. ANL/XFD/CP-107826, Argonne National Lab., IL (US), 2002.

[129] A. Y. Goering, "Numerical investigation of wire coil heat transfer augmentation," Master's thesis, 2016.

[130] J. Best, "The influence of particle rotation on wake stability at particle Reynolds numbers, $\mathrm{Re}_p <$ 300—implications for turbulence modulation in two-phase flows," *International Journal of Multiphase Flow*, vol. 24, no. 5, pp. 693–720, 1998.

[131] J. Dobson, A. Ooi, and E. Poon, "The flow structures of a transversely rotating sphere at high rotation rates," *Computers & Fluids*, vol. 102, pp. 170–181, 2014.

[132] D. Kim, "Laminar flow past a sphere rotating in the transverse direction," *Journal of mechanical science and technology*, vol. 23, no. 2, pp. 578–589, 2009.

[133] E. K. Poon, A. S. Ooi, M. Giacobello, and R. C. Cohen, "Laminar flow structures from a rotating sphere: Effect of rotating axis angle," *International Journal of Heat and Fluid Flow*, vol. 31, no. 5, pp. 961–972, 2010.

[134] J. Jeong and F. Hussain, "On the identification of a vortex," *Journal of fluid mechanics*, vol. 285, pp. 69–94, 1995.

[135] J. D. Schwarzkopf, M. Sommerfeld, C. T. Crowe, and Y. Tsuji, *Multiphase flows with droplets and particles.* CRC press, 2011.

[136] D. Bechert, M. Bruse, W. v. Hage, J. T. Van der Hoeven, and G. Hoppe, "Experiments on drag-reducing surfaces and their optimization with an adjustable geometry," *Journal of fluid mechanics*, vol. 338, pp. 59–87, 1997.

[137] M. J. Walsh, "Riblets as a viscous drag reduction technique," *AIAA journal*, vol. 21, no. 4, pp. 485–486, 1983.

[138] S.-R. Park and J. M. Wallace, "Flow alteration and drag reduction by riblets in a turbulent boundary layer," *AIAA journal*, vol. 32, no. 1, pp. 31–38, 1994.

[139] S.-J. Lee and S.-H. Lee, "Flow field analysis of a turbulent boundary layer over a riblet surface," *Experiments in fluids*, vol. 30, no. 2, pp. 153–166, 2001.

[140] H. Choi, P. Moin, and J. Kim, "Direct numerical simulation of turbulent flow over riblets," *Journal of fluid mechanics*, vol. 255, pp. 503–539, 1993.

[141] D. Goldstein, R. Handler, and L. Sirovich, "Direct numerical simulation of turbulent flow over a modeled riblet covered surface," *Journal of Fluid Mechanics*, vol. 302, pp. 333–376, 1995.

[142] A. Boomsma and F. Sotiropoulos, "Direct numerical simulation of sharkskin denticles in turbulent channel flow," *Physics of Fluids*, vol. 28, no. (3), 035106, 2016.

[143] T. M. Fletcher, *The Evolution of Speed: an empirical and comparative analysis of drag-reducing scales in early fishes.* PhD thesis, University of Leeds, 2015.

[144] P. Nielsen, *Coastal bottom boundary layers and sediment transport*, vol. 4. World scientific, 1992.

[145] R. Deigaard *et al.*, *Mechanics of coastal sediment transport*, vol. 3. World scientific publishing company, 1992.

[146] M. H. Garcia, *Sedimentation engineering: processes, measurements, modeling and practice*. 2008.

[147] B. Jensen, B. Sumer, and J. Fredsøe, "Turbulent oscillatory boundary layers at high Reynolds numbers," *Journal of Fluid Mechanics*, vol. 206, pp. 265–297, 1989.

[148] J. Fredsøe, "Turbulent boundary layer in wave-current motion," *Journal of Hydraulic Engineering*, vol. 110, no. 8, pp. 1103–1120, 1984.

[149] P. R. Spalart and B. S. Baldwin, "Direct simulation of a turbulent oscillating boundary layer," in *Turbulent shear flows 6*, pp. 417–440, Springer, 1989.

[150] M. Hino, M. Kashiwayanagi, A. Nakayama, and T. Hara, "Experiments on the turbulence statistics and the structure of a reciprocating oscillatory flow," *Journal of Fluid Mechanics*, vol. 131, pp. 363–400, 1983.

[151] D. Fytanidis, J. Mier, M. Garcia, and P. Fischer, "Direct numerical simulation of oscillatory boundary layers in the intermittent-turbulent/transitional regime: coherent structures and turbulent statistics," in *AGU Fall Meeting Abstracts*, 2018.

[152] J. M. Mier Lopez, *Experimental analysis of flow and turbulence characteristics in oscillatory boundary layers from LDV measurements*. PhD thesis, University of Illinois at Urbana-Champaign, 2015.

[153] C. M. Reddy, J. S. Arey, J. S. Seewald, S. P. Sylva, K. L. Lemkau, R. K. Nelson, C. A. Carmichael, C. P. McIntyre, J. Fenwick, G. T. Ventura, *et al.*, "Composition and fate of gas and oil released to the water column during the deepwater horizon oil spill," *Proceedings of the National Academy of Sciences*, vol. 109, no. 50, pp. 20229–20234, 2012.

[154] E. Kaminski, S. Tait, and G. Carazzo, "Turbulent entrainment in jets with arbitrary buoyancy," *Journal of Fluid Mechanics*, vol. 526, pp. 361–376, 2005.

[155] M. Walter, C. Mertens, U. Stöber, C. R. German, D. R. Yoerger, J. Sültenfuß, M. Rhein, B. Melchert, and E. T. Baker, "Rapid dispersal of a hydrothermal plume by turbulent mixing," *Deep Sea Research Part I: Oceanographic Research Papers*, vol. 57, no. 8, pp. 931–945, 2010.

[156] L. C. Smith, M. Smith, and P. Ashcroft, "Analysis of environmental and economic damages from british petroleum's deepwater horizon oil spill," *Albany Law Review*, vol. 74, no. 1, pp. 563–585, 2011.

[157] W. Schmidt, "Turbulent propagation of a stream of heated air," *Z. Angew. Math. Mech*, vol. 21, pp. 265–278, 1941.

[158] J. Lavelle, "Buoyancy-driven plumes in rotating, stratified cross flows: Plume dependence on rotation, turbulent mixing, and cross-flow strength," *Journal of Geophysical Research: Oceans*, vol. 102, no. C2, pp. 3405–3420, 1997.

[159] A. F. Tomàs, A. C. Poje, T. M. Özgökmen, and W. K. Dewar, "Effects of rotation on turbulent buoyant plumes in stratified environments," *Journal of Geophysical Research: Oceans*, vol. 121, no. 8, pp. 5397–5417, 2016.

[160] A. Fabregat Tomàs, A. C. Poje, T. M. Özgökmen, and W. K. Dewar, "Dynamics of multiphase turbulent plumes with hybrid buoyancy sources in stratified environments," *Physics of Fluids*, vol. 28, no. (9), 095109, 2016.

[161] A. F. Tomàs, A. C. Poje, T. M. Özgökmen, and W. K. Dewar, "Numerical simulations of rotating bubble plumes in stratified environments," *Journal of Geophysical Research: Oceans*, vol. 122, no. 8, pp. 6795–6813, 2017.

[162] Q. Ye, F. F. Schrijer, and F. Scarano, "Geometry effect of isolated roughness on boundary layer transition investigated by tomographic piv," *International Journal of Heat and Fluid Flow*, vol. 61, pp. 31–44, 2016.

[163] S. P. Schneider, “Effects of roughness on hypersonic boundary-layer transition,” *Journal of Spacecraft and Rockets*, vol. 45, no. 2, pp. 193–209, 2008.

[164] P. Fischer and M. Choudhari, “Numerical simulation of roughness-induced transient growth in a laminar boundary layer,” in *34th AIAA Fluid Dynamics Conference and Exhibit*, p. 2539, 2004.

[165] M. S. Genc, I. Karasu, H. H. Acikel, M. T. Akpolat, and M. Genc, “Low Reynolds number flows and transition,” *Low Reynolds Number Aerodynamics and Transition, Intech-Sciyo Publishing*, 2012.