In this study, a heterogeneous flow model is proposed based on a non–overlapping domain decomposition method. The model combines potential flow and incompressible viscous flow. Both flow domains contain a free surface boundary.

The heterogeneous domain decomposition method is formulated following the Dirichlet–Neumann method. Both an implicit scheme and an explicit scheme are proposed. The algebraic form of the implicit scheme is of the same form of the Dirichlet–Neumann method, whereas the explicit scheme can be interpreted as the classical staggered scheme using the splitting of the Dirichlet–Neumann method.

The explicit scheme is implemented based on two numerical solvers, a Boundary element method (BEM) solver for the potential flow model, and a finite element method (FEM) solver for the Navier–Stokes equations (NSE). The implementation based on the two solvers is validated using numerical examples.
A Heterogeneous Flow Numerical Model based on Domain Decomposition Methods

by

Yi Zhang

A THESIS
submitted to
Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented March 14 2013
Commencement June 2013

APPROVED:

________________________________________
Major Professor, representing Mathematics

________________________________________
Chair of the Department of Mathematics

________________________________________
Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

________________________________________
Yi Zhang, Author
I would like to express my deep gratitude to my advisor Professor Małgorzata Peszyńska, for her patient guidance, enthusiastic encouragement and thorough critiques of this study. I would also like to thank Professor Solomon C. Yim, for his great support and advice for my pursuing in mathematics. My grateful thanks are also extended to Professor Robert L. Higdon and Professor Harry Yeh, for their constructive discussions.

Finally, I wish to thank my wife and my parents for their support and encouragement throughout my study.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
</tr>
<tr>
<td>1.1</td>
<td>Notations and definitions</td>
</tr>
<tr>
<td>1.2</td>
<td>Physical model: viscous incompressible flow</td>
</tr>
<tr>
<td>1.3</td>
<td>Physical model: potential flow</td>
</tr>
<tr>
<td>1.4</td>
<td>Physical model: heterogeneous flow</td>
</tr>
<tr>
<td>2</td>
<td>Literature review</td>
</tr>
<tr>
<td>2.1</td>
<td>Numerical methods for NSE</td>
</tr>
<tr>
<td>2.2</td>
<td>Numerical methods for potential flows</td>
</tr>
<tr>
<td>2.3</td>
<td>Numerical methods for free surface tracking</td>
</tr>
<tr>
<td>2.4</td>
<td>Domain decomposition methods</td>
</tr>
<tr>
<td>3</td>
<td>Numerical methods</td>
</tr>
<tr>
<td>3.1</td>
<td>FEM for Navier-Stokes Equations</td>
</tr>
<tr>
<td>3.2</td>
<td>BEM solver for potential flow</td>
</tr>
<tr>
<td>3.3</td>
<td>Domain decomposition method for the heterogeneous model</td>
</tr>
<tr>
<td>4</td>
<td>Numerical examples</td>
</tr>
<tr>
<td>4.1</td>
<td>Initial stage of dam break wave</td>
</tr>
<tr>
<td>4.2</td>
<td>Solitary wave</td>
</tr>
<tr>
<td>5</td>
<td>Conclusions</td>
</tr>
<tr>
<td>Bibliography</td>
<td>76</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>1.1</td>
<td>Simplified decomposition of wave tank model.</td>
</tr>
<tr>
<td>1.2</td>
<td>Subdomains and the interface.</td>
</tr>
<tr>
<td>3.1</td>
<td>Using D–N method to solve BVP (3.34). The interface is located at x = 8.</td>
</tr>
<tr>
<td>3.2</td>
<td>Weakly–coupled (left) and strongly–coupled (right) time integration schemes</td>
</tr>
<tr>
<td>3.3</td>
<td>An explicit scheme for the heterogeneous flow model.</td>
</tr>
<tr>
<td>3.4</td>
<td>Meshed subdomains and the free surface near the interface.</td>
</tr>
<tr>
<td>4.1</td>
<td>Decomposition of the domain for the dam break wave.</td>
</tr>
<tr>
<td>4.2</td>
<td>Dambreak wave front evolution as predicted by ICFD solver in DD method</td>
</tr>
<tr>
<td>4.3</td>
<td>Surface elevation time history.</td>
</tr>
<tr>
<td>4.4</td>
<td>Error of free surface elevation at the interface.</td>
</tr>
<tr>
<td>4.5</td>
<td>Error of interface velocities.</td>
</tr>
<tr>
<td>4.6</td>
<td>Solitary wave generation in a wave tank domain.</td>
</tr>
<tr>
<td>4.7</td>
<td>Solitary wave profile at T=0.80—T=1.125</td>
</tr>
<tr>
<td>4.8</td>
<td>Solitary wave profile at T=1.190—T=1.515</td>
</tr>
<tr>
<td>4.9</td>
<td>Solitary wave profile at T=1.580—T=1.915</td>
</tr>
<tr>
<td>4.10</td>
<td>Time history of nodal velocity u.</td>
</tr>
<tr>
<td>4.11</td>
<td>Time history of nodal velocity v.</td>
</tr>
<tr>
<td>4.12</td>
<td>Time history of normalized pressure.</td>
</tr>
</tbody>
</table>
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>60</td>
</tr>
<tr>
<td>4.2</td>
<td>66</td>
</tr>
</tbody>
</table>
A HETEROGENEOUS FLOW NUMERICAL MODEL BASED ON
DOMAIN DECOMPOSITION METHODS

1 Introduction

Among many models that are applied to water flow phenomena, \textit{viscous} incompressible flow and potential flow (PF) are the most commonly used. The former describes Newtonian low-Mach-number flows in the most general way, with governing equations being the Navier-Stokes equations (NSE). The latter describes \textit{inviscid} incompressible flows, assuming additionally that they are irrotational.

The NSE is the most general but also poses one of the greatest challenges in mathematics, and it is difficult to solve numerically. In particular, when turbulence occurs, the multiscale nature of the flow requires an enormous amount of computing effort. This is the case for most engineering applications.

The potential flow model, due to its simplicity, does not apply to many complex applications, especially when Fluid-Structure Interaction (FSI) needs to be considered. However, its governing equation—Laplace’s equation—is simple to analyze and solve numerically. Therefore, the potential flow model is still actively adopted in applications such as ocean and coastal engineering.

Because of the complementary nature of the above two physical models, it is attractive to think of taking advantage of both, whenever it is compatible with the nature of the flow. In this study we examine a numerical method that combines the two models. In particular, we apply this method to problems involving free surface. The flow domain is decomposed into two subdomains, within which viscous incompressible flow and potential flow model are applied, respectively. In reality, the viscous flow subdomain would be the region “around” a structure, in order to recover “interesting” flow features.

At the \textit{far field} subdomain we use the PF model to preserve model accuracy
while reduce computing cost. In general, at the outer boundaries of a potential flow domain, the boundary condition should satisfy the open boundary condition (OBC). In this study, the outer boundary is considered to be the physical boundary of a wave tank. Thus the Neumann boundary condition is applied.

Ideally, to model a structure’s behavior in such a wave tank we can decompose the domain so that the inner subdomain, the subdomain “close” to the structure, is as small as an accurate simulation allows, such as shown in Figure 1.1. In such a decomposition, the wave tank is divided into three sections. In this study, we consider the case of only two subdomains separated by a vertical interface.

![Figure 1.1: Simplified decomposition of wave tank model.](image)

The methodology presented above bears many names. In aerodynamics, it is commonly referred to as the “zonal approach”. In the computational fluid dynamics (CFD) community, it is sometimes referred to as the “heterogeneous flow modeling”. In computational mechanics, it falls into the broader field of multiphysics modeling. In numerical mathematics, this approach belongs to the class of domain decomposition (DD) methods and specifically, it is a heterogeneous DD method. In our approach, the NSE is solved using the finite element method (FEM), as implemented in the solver ICFD LS–DYNA®, a multiphysics general-purpose FEM package [74]. The PF problem is solved using a boundary element method (BEM) solver NWT3D which is designed specifically for water wave problems [54]. We will demonstrate the soundness of our approach with numerical examples, and study the structure of the coupling.

This thesis is organized as follows. In Chapter 1 we introduce the notations and definitions related to the mathematical framework for the heterogeneous DD
method and numerical methods for fluid dynamics. Then formulations of the
NSE and the PF model are introduced, as well as the coupled model and other
formulations of the governing equations of the viscous incompressible flow. In
Chapter 2 we present the background on the numerical methods involved, namely,
the FEM for the NSE and the BEM for the PF, together with the DD methods.
In particular, the matching conditions in the DD method are defined. Chapter 3
is the kernel of this thesis, and it describes all the numerical methods. The FEM
scheme based on the orthogonal subscale stabilization method (OSS), and the BEM
scheme based on the collocation discretization domain are first introduced. Then
we formulate the coupled problem and its solution using a non-overlapping DD
method. In Chapter 4 we demonstrate the capability of the DD method through
two numerical examples. In each example, the flow domain is decomposed into
two subdomains. Convergence results as well as the features of each solver are
presented.

In the rest of this chapter, we will first introduce some common notations. Then
we will introduce the physical models used in this study.

1.1 Notations and definitions

Other than symbols and notations that are defined in the following chapters, here
we introduce notations and definitions that are commonly used in this thesis.

In this study, unless stated otherwise, the Eulerian representation of the flow is
used. Spatial vectors are denoted by $\mathbf{x}$, time $t$, velocity $\mathbf{u} = (u_1, u_2, u_3)$, pressure $p$, and flow potential $\varphi$. As commonly adopted in fluid dynamics literature, Cartesian
notations $(x, y, z)$ and $(u, v, w)$ are also used to describe the spatial position and
velocity, respectively. Correspondingly $\mathbf{e}_i$, $i = 1, 2, 3$, denotes the unit vectors for
the Cartesian system. In our flow model with free surface, free surface elevation is
denoted by $\eta$.

Tensors are denoted using either Einstein notation or the blackboard bold fonts.
In particular, strain rate tensor is either referred to as $e_{ij}$ or $\mathbf{E}$. In the former case,
we have

\[ e_{ij} \triangleq \frac{1}{2} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right). \]  

(1.1)

Similarly, stress tensor is denoted by \( \sigma_{ij} \) or \( S \). For Newtonian incompressible flow, \( \sigma_{ij} \) is given as

\[ \sigma_{ij} = -p \delta_{ij} + 2\mu e_{ij}. \]  

(1.2)

Here \( \delta_{ij} \) is the Kronecker symbol which is denoted in the tensor form as \( I \), and \( \mu \) is the dynamic viscosity. One can see that \( e_{ij} \) is the symmetric component of the velocity gradient by decomposing the latter as

\[ \frac{\partial u_i}{\partial x_j} = e_{ij} + \frac{1}{2} r_{ij}, \]  

(1.3)

\[ r_{ij} \triangleq \frac{\partial u_i}{\partial x_j} - \frac{\partial u_j}{\partial x_i}. \]  

(1.4)

The rotation tensor \( r_{ij} \), as the antisymmetric component of the decomposition, is related to vorticity \( \omega \triangleq \nabla \times u \) by \( r_{ij} = -\varepsilon_{ijk} \omega_k \), where \( \varepsilon_{ijk} \) is the Levi-Civita symbol. In the decomposition (1.3), \( e_{ij} \) describes the deformation of fluid particle around certain point, while \( r_{ij} \) describes the rotation around that point. To simplify the notation, we also denote the material derivative of a field quantity \( a \) in the flow domain as

\[ \frac{D a}{D t} \triangleq \frac{\partial a}{\partial t} + u \cdot \nabla a, \]

where the inner product term, usually referred to as the convection term, indicates the rate of change due to the spatial distribution of \( a \).

Let \( \Omega \in \mathbb{R}^n \), \( n = 2, 3 \), be an open bounded domain with boundary \( \partial \Omega \). This is the domain where our flow problem will be defined. Let \( L^1_{\text{loc}}(\Omega) \) be the set of locally integrable functions on \( \Omega \), and \( D(\Omega) \) be the space of \( C^\infty \) functions with compact support on \( \Omega \). We say that function \( u \in L^1_{\text{loc}}(\Omega) \) has \( \alpha \)th order weak derivative
$D^\alpha u$ if there exists $D^\alpha u \in L^1_{\text{loc}}(\Omega)$ such that [1]

$$\int_{\Omega} uD^\alpha \varphi \, d\Omega = (-1)^\alpha \int_{\Omega} D^\alpha u \varphi \, d\Omega.$$  

Here the multi–index notation has been used. Namely, $\alpha$ is an ordered $n$-tuple, $n \in \mathbb{Z}^+$, such that $\alpha \triangleq (\alpha_1, \alpha_2, \ldots, \alpha_n)$. We define $|\alpha| \triangleq \sum_{i=1}^{n} \alpha_i$, and the partial derivative

$$D^\alpha u \triangleq \frac{\partial^{|\alpha|} u}{\partial x_1^{\alpha_1} \partial x_2^{\alpha_2} \cdots \partial x_n^{\alpha_n}}.$$  

Let $W^{m,p}(\Omega)$ be the Sobolev spaces

$$W^{m,p}(\Omega) \triangleq \{ u \in L^p(\Omega) : D^\alpha u \in L^p(\Omega) \ \forall |\alpha| \leq m \}, \ m \geq 1, \ p \geq 1.$$  

We define the norm on the $W^{m,p}(\Omega)$ as a functional $\| \cdot \|_{m,p}$ such that

$$\|u\|_{m,p} = \left( \sum_{0 \leq |\alpha| \leq m} \|D^\alpha u\|_{L^p(\Omega)}^p \right)^{1/p}, \ p \geq 1, \ \forall u \in W^{m,p}(\Omega).$$  

A special type of $W^{m,p}(\Omega)$ is

$$H^m(\Omega) \triangleq W^{m,2}(\Omega).$$  

Note that $H^0(\Omega) = L^2(\Omega)$. We denote by $(\cdot, \cdot)$ the inner product on $H^m(\Omega)$, i.e.,

$$(u, v) \triangleq \int_{\Omega} \sum_{|\alpha| \leq m} D^\alpha u D^\alpha v \, d\Omega, \ \forall u, \ v \in H^m(\Omega).$$  

The closure of $D(\Omega)$ in $W^{m,p}(\Omega)$ is denoted by $W^{m,p}_0(\Omega)$, and the closure of $D(\Omega)$ in $H^m(\Omega)$ norm is denoted by $H^m_0(\Omega)$. We use bold face symbols to denote the $n$-dimensional version of corresponding space, e.g., $\mathbf{L}^2(\Omega) = (L^2(\Omega))^n$. Other than these standard Sobolev spaces, some spaces are frequently used in results related
to the NSE. We define

\[ \mathcal{V} \triangleq \{ u \in D(\Omega), \nabla \cdot u = 0 \}. \]

The basic spaces for studying the NSE are the closure of \( \mathcal{V} \) in \( L^2(\Omega_1) \) and \( H^1_0(\Omega_1) \), denoted as \( \mathcal{H} \) and \( \mathcal{V} \), respectively.

Now let us apply a decomposition to \( \Omega \), so that

\[ \overline{\Omega} = \overline{\Omega}_1 \cup \overline{\Omega}_2, \quad \Omega_1 \cap \Omega_2 = \emptyset, \quad \overline{\Omega}_1 \cap \overline{\Omega}_2 = \Gamma. \]

Such a decomposition is shown in Figure 1.2.

![Figure 1.2: Subdomains and the interface.](image)

In \( \Omega_1 \) and \( \Omega_2 \) we adopt different flow models. Specifically, the governing equation in \( \Omega_1 \) is the NSE, and the potential flow model is used in \( \Omega_2 \). In this study \( \Gamma \) is often referred to as the interface between \( \Omega_1 \) and \( \Omega_2 \). Unless specified otherwise, for certain quantity \( a \), the subscript \( i \) (\( i = 1, 2 \)) denotes the restriction of \( a \) in \( \Omega_i \), i.e., \( a_i \triangleq a|_{\Omega_i} \). Similarly, we have \( a_\Gamma \triangleq a|_{\Gamma} \). Across \( \Gamma \) the values from \( \Omega_1 \) and \( \Omega_2 \) could differ, and we use \( [\cdot] \) to denote the jump across of the interface, i.e.

\[ [a] = a_1 - a_2 \]

The superscript \( n \) denotes \( a \) at certain time step \( n \). In a numerical scheme with iterations, comma-separated superscripts, such as \( a^{n,(k)} \), describes \( a \) at its \( k \)th iteration within time step \( n \). Depending on the context, we use superscript \( \ast \) to denote the dual function space, the adjoint operator, and the Cauchy data for
boundary value problems. Subscript $h$ denotes a quantity’s discrete version in finite element space, with $h$ indicating the characteristic element size. In particular, $T_h$ is the set of elements from the triangulation of $\Omega$. The time stepping size is denoted as $\Delta t$.

1.2 Physical model: viscous incompressible flow

In this study, unless otherwise stated, we assume unit density, i.e., $\rho \equiv 1$. Thus the NSE for a viscous incompressible flow is

$$\frac{D\mathbf{u}}{Dt} = -\nabla p + \mu \nabla^2 \mathbf{u} + \mathbf{f}, \quad \forall (x, t) \in \Omega_1 \times (0, T], \quad (1.5a)$$

$$\nabla \cdot \mathbf{u} = 0, \quad \forall (x, t) \in \Omega_1 \times (0, T]. \quad (1.5b)$$

The equation (1.5a) describes the conservation of the momentum, and equation (1.5b) is derived from the conservation of mass. $\mathbf{f}$ denotes the body force in the field. In this study, the only body force in $\Omega$ is the gravitational force, i.e., $\mathbf{f} = -g\mathbf{e}_3$.

Remark 1.2.1. Unlike in compressible flow, here energy conservation does not enter the governing equations, because the density $\rho$ and pressure $p$ are decoupled from thermodynamics equations, and the pressure wave travels at an infinite speed (sound speed $c = \infty$) while the density is kept constant.

As to the boundary conditions, note that they are required in both normal and tangential directions. Let $\Gamma_D$ be the Dirichlet boundary and $\Gamma_N$ be the Neumann boundary, such that $\partial \Omega = \Gamma_D \cup \Gamma_N$. Recall that superscript * denotes boundary data, then we have

$$\mathbf{u} = \mathbf{u}^*, \quad \text{on } \Gamma_D. \quad (1.6)$$

Recall the description of the stress tensor from §1.1, the corresponding Neumann boundary condition is

$$(-pI + \mu \nabla \mathbf{u}) \cdot \mathbf{n} = \mathbf{f}^*, \quad \text{on } \Gamma_N. \quad (1.7)$$
In the case of a planar boundary, the normal and tangential component of (1.7) are
\[
\begin{cases}
-p + \mu \frac{\partial u_n}{\partial n} = f_n^\ast, \\
\mu \frac{\partial u_\tau}{\partial n} = f_\tau^\ast.
\end{cases}
\] (1.8)

It is instructive to look at variant formulations and boundary conditions of the momentum equation of the NSE. Those formulations arise from different treatments of convection term and stress term(s).

First, we present some formulations of the stress term, the term that describes the contribution of the stress tensor to the momentum conservation.

1. Stress-divergence form: classical stress balance argument gives the stress term as \( \nabla \cdot (-p \mathbb{I} + 2\mu \mathbb{E}) \).

2. Laplace form of viscous term: in the stress-divergence form, applying the divergence operator to the strain rate tensor, and using the incompressibility condition \( \nabla \cdot \mathbf{u} = 0 \) to get \(-\nabla p + \mu \nabla^2 \mathbf{u}\).

3. Div-Curl form of viscous term: use identity \( \nabla^2 \mathbf{u} = \nabla (\nabla \cdot \mathbf{u}) - \nabla \times \nabla \times \mathbf{u} \), we have the stress terms as \(-\nabla p + \mu [\nabla (\nabla \cdot \mathbf{u}) - \nabla \times \omega] \).

4. Curl form of viscous term: bring \( \nabla \cdot \mathbf{u} = 0 \) into the previous form, then the stress term becomes \(-\nabla p + \mu \nabla \times \omega \).

Next, we collect some forms of the term that describes the convection.

1. Conventional form: the convection term is most often written as \( \mathbf{u} \cdot \nabla \mathbf{u} \).

2. Divergence form: introducing \( \nabla \cdot \mathbf{u} = 0 \) into the identity
\[
\frac{\partial}{\partial x_j} (u_i u_j) = u_j \frac{\partial u_i}{\partial x_j} + u_i \frac{\partial u_j}{\partial x_j},
\]
we have the convection term as \( \nabla \cdot (\mathbf{uu}) \).

3. Vorticity form: this is simply the previous form with identity
\[
\frac{1}{2} \nabla \cdot (\mathbf{uu}) = \mathbf{u} \times \omega + \mathbf{u} \cdot \nabla \mathbf{u}.
\]
Combing the stress and convection terms with other proper terms in the momentum conservation equation, we have several alternative formulations of (1.5a) with the corresponding boundary conditions. Here we only list a few that will be used in Chapter 3.

Let \((0, T]\) be the time interval during which the flow problem is defined, then with all flow properties being functions of location \(\mathbf{x} \in \Omega_1\) and time \(t \in (0, T]\), we have the following formulations.

1. NSE: the most frequently used form of the momentum conservation adopts the conventional form of the advection term and the Laplace form of the viscous term. This gives us (1.5a) with boundary conditions (1.6) and (1.7).

2. Divergence form: with the divergence form for the advection term and the viscous stress term, the momentum equation becomes

\[
\frac{\partial \mathbf{u}}{\partial t} = \nabla \cdot (-p\mathbb{I} + 2\mu\mathbf{E} - \mathbf{uu}) + \mathbf{f}, \quad \forall (\mathbf{x}, t) \in \Omega_1 \times (0, T].
\] (1.9)

This form leads easily to the global momentum balance, also the Neumann boundary condition imposing the total momentum flux

\[
(\mathbf{uu} - \mathbf{\sigma}) \cdot \mathbf{n} = \mathbf{f}^*, \quad \text{on} \ \Gamma_N.
\] (1.10)

3. Total stress form: use the conventional form for the advection term and the stress-divergence form, the momentum equation becomes

\[
\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = \nabla \cdot (-p + 2\mu\mathbf{E}) + \mathbf{f}, \quad \forall (\mathbf{x}, t) \in \Omega_1 \times (0, T].
\] (1.11)

The Neumann boundary condition related to this form is

\[
(-p + 2\mu\mathbf{E}) \cdot \mathbf{n} = \mathbf{f}^*, \quad \text{on} \ \Gamma_N.
\] (1.12)

Only this form can be used to impose a true physical traction boundary condition.
4. Vorticity and total pressure form: the curl form of the viscous term and the vorticity form of the convection term give

\[ \frac{\partial \mathbf{u}}{\partial t} + \mathbf{\omega} \times \mathbf{u} = -\nabla p_T - \mu \nabla \times \mathbf{\omega} + \mathbf{f}, \quad \forall (\mathbf{x}, t) \in \Omega_1 \times (0, T]. \quad (1.13) \]

Here \( p_T \) is the total pressure, i.e., the sum of static pressure and dynamic pressure

\[ p_T = p + \frac{1}{2} |\mathbf{u}|^2. \]

In this formulation, when the flow is close to be irrotational, even if the Reynolds number is high, nonlinearity is suppressed. This is usually the case at an outflow/open boundary. Thus the outflow/open boundary condition (OBC) can be imposed using the total pressure

\[ p + \frac{1}{2} |\mathbf{u}|^2 = p^*. \quad (1.14) \]

In addition, even if \( \omega \) is large, when it aligns with \( \mathbf{u} \), the nonlinear advection term would still vanish.

Remark 1.2.2. Different formulations of the momentum equation that are equivalent in the classical (continuum) sense are generally not equivalent when discretized. In particular, the incorporation of \( \nabla \cdot \mathbf{u} = 0 \) gives different forms of advection and viscous terms.

Remark 1.2.3. It is interesting to note that equation (1.13) gives an OBC. If one has \( \omega \approx 0 \) at the outlet, the corresponding homogeneous Neumann boundary condition is

\[ -p_T \mathbf{n} + \mu \nabla \mathbf{u} \cdot \mathbf{n} = 0. \quad (1.15) \]

This is similar to equation (1.7), except that the pressure is replaced with the total
pressure. Corresponding formulation of equation (1.8) is then

\[
\begin{align*}
-p_T + \mu \frac{\partial u_i}{\partial n} &= f_n^*, \\
\mu \frac{\partial u_i}{\partial n} &= f^*_\tau.
\end{align*}
\]

If the boundary data \( f^* \) is in the direction of normal of the boundary, the tangential component becomes

\[
\mu \frac{\partial u_i}{\partial x_j} n_j = f_i^* \tau_i = 0 \quad \Leftrightarrow \quad \frac{\partial u_i}{\partial n} \tau_i = 0 \quad \Leftrightarrow \quad \frac{\partial u_{\tau}}{\partial n} = 0
\]

In the context of our problem, this condition, if imposed at the interface, can be understood as a “soft” version of the matching of tangential velocity.

1.3 Physical model: potential flow

First, assume the flow is inviscid, then the momentum conservation equation becomes Euler’s equation, the “simplified” version of equation (1.5a) with the viscous term removed. Now assuming the flow is irrotational, i.e., \( \omega = 0 \), we have

\[
\mathbf{u} = \nabla \varphi,
\] (1.16)

where \( \varphi \) is referred to as the potential. Introducing (1.16) and \( \omega = 0 \) into (1.13), we have

\[
\nabla \left( \frac{\partial \varphi}{\partial t} \right) = -\nabla \left( p + \frac{1}{2} |\nabla \varphi|^2 \right) + \nabla (gz).
\]

Integration in space gives Bernoulli’s principle [4]

\[
\frac{\partial \varphi}{\partial t} + \frac{|\nabla \varphi|^2}{2} + p + gz = C(t).
\] (1.17)
Remark 1.3.1. Here \( C(t) \) can be incorporated in \( \varphi \) by redefining

\[
\varphi^* \triangleq \varphi + \int_{t_0}^t C(s) \, ds.
\]

However, in a steady inviscid (not necessarily irrotational) flow, \( C \) must remain in the equation since it varies among streamlines. On the other hand, in the case of steady potential flow, we can combine the *global* constant \( C \) with \( p \) by redefining the pressure. However, this is not applicable when the flow has physical reference pressure imposed, such as in water waves, otherwise the newly defined \( p^* \) containing \( C \) would not match gauge pressure \( p_0 \) at the free surface.

Since one equation suffices to solve for the unknown \( \varphi \), the problem of an *incompressible* potential flow is usually posed as Laplace’s equation based on the mass conservation equation. Namely, combining (1.16) and (1.5b), we have

\[
\nabla^2 \varphi = 0, \quad \forall (x, t) \in \Omega_2 \times (0, T].
\] (1.18)

Equation (1.18) must be complemented by proper boundary conditions. In a typical water wave tank model, other than the free surface, all boundaries are imposed with the Neumann boundary condition. At the free surface \( \Gamma_f \subset \partial \Omega_2 \), boundary conditions provide essential information regarding the temporal evolution. Specifically, a *kinematic boundary condition* describes the fact that the water particles on the free surface move with the free surface itself, i.e.,

\[
\frac{D\mathbf{x}}{Dt} = \mathbf{u} = \nabla \varphi, \quad \forall \mathbf{x} \in \Gamma_f.
\] (1.19)

In the Cartesian system, this is equivalent to

\[
w = \frac{\partial \eta}{\partial t} + u \frac{\partial \eta}{\partial x} + v \frac{\partial \eta}{\partial y}, \quad \forall \mathbf{x} \in \Gamma_f.
\] (1.20)

Since (1.20) introduces an extra unknown \( \eta \), another condition is required to close the system. For that, Bernoulli’s equation (1.17) on the free surface is used,
hence the *dynamical boundary condition*

\[ \frac{D\varphi}{Dt} = -gz + \frac{1}{2}|\nabla \varphi|^2 - p^*, \quad \forall \mathbf{x} \in \Gamma_f. \quad (1.21) \]

### 1.4 Physical model: heterogeneous flow

With the two flow models above, we have the heterogeneous model in Ω

\[
\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p + \mu \nabla^2 \mathbf{u} + \mathbf{f}, \quad \forall (\mathbf{x}, t) \in \Omega_1 \times (0, T],
\]

\[ \nabla \cdot \mathbf{u} = 0, \quad \forall (\mathbf{x}, t) \in \Omega_1 \times (0, T] \quad (1.22b) \]

\[ \nabla^2 \varphi = 0, \quad \forall (\mathbf{x}, t) \in \Omega_2 \times (0, T], \quad (1.22c) \]

\[ Q_1(\mathbf{u}, p, \varphi) = 0, \quad \forall (\mathbf{x}, t) \in \Gamma \times (0, T], \quad (1.22d) \]

\[ Q_2(\mathbf{u}, p, \varphi) = 0, \quad \forall (\mathbf{x}, t) \in \Gamma \times (0, T]. \quad (1.22e) \]

Let us emphasize here that the governing equations in Ω₁ and Ω₂ do not formally share any unknowns. Specifically, the velocity in the NSE is one order higher than the potential, and Laplace’s equation does not explicitly contain pressure as an unknown.

In this study, we solve equation (1.22), together with proper boundary conditions at ∂Ω, in a framework of DD methods. In particular, for this model we seek formulation and implementation of the matching conditions (1.22d) and (1.22e). In general, the formulation of \( Q_1 \) and \( Q_2 \) are derived from physical interpretation of the model or from mathematical analysis such as singular perturbations. See §3.3 for details.

**Remark 1.4.1.** Unlike Laplace’s equation that only requires normal flux for the Neumann boundary. For NSE, boundary conditions should provide information in all directions. The velocity boundary condition is the most straightforward option. Others include the combination of tangential velocity and pressure [44, 82]. This would be attractive during the iteration when both tangential velocity and pressure can be provided from the PF solver, which accepts normal velocity as the Neumann boundary condition. However, it is pointed out that this boundary
condition has seen “little or no laboratory testing/verification” [51]. To avoid further uncertainties, in this study the velocity boundary condition is imposed to the NSE at the interface.
2 Literature review

There exists a large number of references for fluid dynamics and the NSE, and a comprehensive review is out of our scope. For an introduction to the topic from a physical perspective, see the classical book by Batchelor [4], also Cohen and Kundu [28], Tritton [103] and Faber [37], among others. From mathematical perspective, Chorin and Marsden [21] is a great introduction, also see Ladyzhenskaya [69], Kreiss and Lorenz [68], Lions [72], and Majda and Bertozzi [75] for the theoretical expositions of incompressible flow and the NSE. The collection of boundary conditions in §1.2 is based on the review by Gresho and Sani [52, p. 362–373].

For water wave problems and potential flows, some of the above references contain dedicated chapters, such as Batchelor [4, Chap. 6] and Ladyzhenskaia [69, Chap. 3]. Dean and Dalrymple [30] is an excellent introduction to the topic from an application perspective.

2.1 Numerical methods for NSE

The research on numerical methods for solving the NSE is extensive. General introduction can be found in many computational fluid dynamics (CFD) books, such as Anderson [2], Date [29], Fletcher [42], and Zienkiewicz et al. [107]. The classical scheme by Chorin [19, 20] and Temam [99] introduces a projection method that has inspired many later methods. The theoretical background of this projection method is the Helmholtz–Hodge Decomposition [21, p. 37] or simply Hodge Decomposition [75, p. 32], also referred to as decomposition theorem of Ladyzhenskaya [69, p. 23]. To show the relation between Chorin-Temam’s scheme and the
decomposition, we rearrange equation (1.5a) (with body force term omitted) as

\[
\frac{\partial u}{\partial t} + \nabla p = \mu \nabla^2 u - u \cdot \nabla u.
\]

Recall that we assume unit density. This form shows that \(\partial u/\partial t\) and \(\nabla p\) can be viewed as the solenoidal and the irrotational part of the right hand side due to the Hodge Decomposition, respectively. Namely, \(\partial u/\partial t\) can be considered as the projection of right hand side to a solenoidal space. Inspired by this observation, the original Chorin-Temam’s projection method first calculates the velocity without meeting the incompressibility condition imposed through \(p\), then corrects the result by projection. Specifically, with \(u^n\) known from the previous time step, to find \(u^{n+1}\), we first solve for \(\tilde{u}\) in

\[
\frac{\tilde{u} - u^n}{\Delta t} = \mu \nabla^2 u^n - u^n \cdot \nabla u^n.
\]  

(2.1)

Then the solenoidal velocity is obtained in a projection step, by removing the irrotational component

\[
\begin{align*}
    u^{n+1} &= \tilde{u} - \Delta t \nabla p^{n+1}, \\
    \nabla \cdot u^{n+1} &= 0.
\end{align*}
\]  

(2.2)

This requires the knowledge of \(p^{n+1}\) which is calculated by taking divergence on both sides of the first equation of (2.2) and applying the incompressibility condition, i.e., \(p^{n+1}\) is obtained by solving Poisson’s equation

\[
\nabla^2 p^{n+1} = \frac{\nabla \cdot \tilde{u}}{\Delta t}.
\]  

(2.3)

The Chorin-Temam’s projection method, also referred to as \(L^2\)-projection method [45], has several features:

1. It is amenable to implementation.

2. It has the velocity and pressure decoupled.
3. It contains some intrinsic pressure stabilization capability (see later of this chapter).

On the other hand, a controversy around the projection methods is the boundary condition. Specifically, to solve for \( \tilde{\mathbf{u}} \) in (2.1) one has to impose the Dirichlet data on the Dirichlet boundary, and to solve for \( p^{n+1} \) from (2.3) one has to impose the homogeneous Neumann boundary condition. This artificial boundary condition induces a numerical boundary layer. Even though improved schemes such as that by Timmermans et al. [101] have addressed this problem, \( \mathbf{u}^{n+1} \) computed from the projection still does not meet the tangential boundary condition. In other words, with the projection methods, one needs to choose between \( \tilde{\mathbf{u}} \) and \( \mathbf{u}^{n+1} \) as the “final” velocity for the current time step. \( \tilde{\mathbf{u}} \) satisfies the Dirichlet boundary condition but is not solenoidal, whereas \( \mathbf{u}^{n+1} \) is divergence-free but does not fully meet the Dirichlet boundary condition. Even though it is common to adopt \( \mathbf{u}^{n+1} \) as the “final” velocity, the advantages of using \( \tilde{\mathbf{u}} \) have been pointed out by Guermond [55] and Guermond and Quartapelle [56]. In particular, they have shown that using \( \tilde{\mathbf{u}} \) as the final velocity does not spoil numerical results, while the numerical boundary layer introduced by using \( \mathbf{u}^{n+1} \) with incorrect boundary condition could contaminate the flow field. If \( \tilde{\mathbf{u}} \) is chosen to be the “final” velocity, by omitting the projection of \( \tilde{\mathbf{u}} \) onto \( \mathcal{V} \), one eliminates \( \mathbf{u}^{n+1} \) from equation (2.2). Namely, one solves for \( \tilde{\mathbf{u}} \) and \( p^{n+1} \) in

\[
\begin{align*}
\frac{\tilde{\mathbf{u}} - \mathbf{u}^n}{\Delta t} &= \mu \nabla^2 \mathbf{u}^n - \mathbf{u}^n \cdot \nabla \mathbf{u}^n, \\
\nabla^2 p^{n+1} &= \frac{\nabla \cdot \tilde{\mathbf{u}}}{\Delta t}.
\end{align*}
\] (2.4)

From (2.1)–(2.4), one can see that in this type of projection methods there is no explicit time integration for the pressure. It was first observed by Goda [47] that incremental–pressure projection improves model accuracy. Described using
the scheme (2.4), the standard incremental–pressure projection scheme gives

\[
\begin{align*}
\tilde{u} - u^n &= -p^n + \mu \nabla^2 u^n - u^n \cdot \nabla u^n, \\
\nabla^2 (p^{n+1} - p^n) &= \frac{\nabla \cdot \tilde{u}}{\Delta t}.
\end{align*}
\]

(2.5)

An overview of Chorin-Temam’s method, including its improved versions, can be found in Guermond [55] and Quarteroni [84, Chap. 15].

Remark 2.1.1. Note that we present the above projection methods using the forward Euler method. Using other suitable time stepping schemes follows the same procedure. Popular time stepping schemes in applications include \(\vartheta\)-methods and backward differencing (BDF) methods; see §3.1.

For general introduction of the application of finite element method to incompressible flow problems, see the books by Elma et al. [35], Glowinski [45], Gunzburger [57], Gresho and Sani [52], Quartapelle [83] and Temam [100]. Also, see the book chapter by Marion and Temam [76].

In general every numerical method solving incompressible viscous flow, in particular a method based on finite elements, is faced with three challenges. The first two are due to the convection term, and the third is the result of the incompressibility.

2.1.1 Instability due to convection

The examples of oscillation (wiggles) caused by Bubnov–Galerkin approximation for large Péclet number can be found in [33, 35]. To attenuate the effect of this instability, various stabilization approaches have been proposed. Overviews of such measures can be found in [23, 33, 89]. Among those measures, SUPG [12] and GLS [60] add a stabilization term of form

\[
\sum_e \int_{\Omega_e} P(v) \tau_e R(u) \, d\Omega
\]

(2.6)
into the weak formulation of the momentum equation. Here $\mathcal{R}(\mathbf{u})$ is the residual, $\mathcal{P}(\mathbf{v})$ is the stabilization operator on test function, and $\tau_e$ is the stabilization parameter (intrinsic time). Therefore SUPG and GLS belong to the family of Residual-Based stabilizations [89]. In particular, $\mathcal{P}(\mathbf{v}) = \mathbf{u}^n \cdot \nabla \mathbf{v}$ in SUPG, and $\mathcal{P}(\mathbf{v}) = -\mu \nabla^2 \mathbf{v} + \mathbf{u}^n \cdot \nabla \mathbf{v}$ in GLS.

*Remark 2.1.2.* In practice, it is common to simplify the $\mathcal{R}(\mathbf{u})$ in the SUPG by keeping only the convection term, i.e.,

$$\mathcal{R}(\mathbf{u}) \approx \mathbf{u}^n \cdot \nabla \mathbf{u}^n.$$ 

This is because that the convection stabilization is used when the effect of convection is much greater than that of the diffusion, therefore neglecting the latter does not affect the numerical results, even though this violates the consistency of the discrete formulation. Moreover, when $\mathbf{u} \in H^1(\Omega)$ the calculation of the second order diffusion term leads to awkward and artificial treatments across the elements, and usually the discrete version of $\mathcal{R}(\mathbf{u})$ would not make the weak form consistent even with the diffusion term.

Another type of stabilization methods is the Variational Multiscale (VMS) method [5, 48, 59]. In this case the instability is addressed from a perspective of subgrid scale effects. As a special case of VMS, bubble functions increases stability by providing subgrid degree of freedom. Also, with piecewise linear interpolation function space in FEM, using bubble functions is equivalent to SUPG [11], which also made it clear that why streamline diffusion method, traditionally derived from Petrov-Galerkin approach, has stabilizing capability. Both approaches to streamline diffusion can be found in [35, p. 126]. Similarly to the VMS methods, Codina proposed the orthogonal subscale (OSS) method [26, 27], which is the method that is implemented for the NSE solver in this study.

### 2.1.2 Nonlinearity

Because of the nonlinear convection term $\mathbf{u} \cdot \nabla \mathbf{u}$ in the NSE, iterative methods are in general necessary. The problem introduced by iterative methods are of two–folds:
first, a large amount of computation effort is usually required in applications, and second, the iterative methods may not converge toward the desired accuracy, or perform in a very slow rate.

The most popular iterative methods for solving the NSE are from Picard iteration and Newton’s method family. In Picard iteration, the convection velocity is “frozen” as the value from previous step, so the nonlinear term becomes $u^k \cdot \nabla u^{k+1}$. This converts the NSE into an Oseen flow problem, which has the general form

$$\begin{cases}
\frac{\partial u}{\partial t} + v \cdot \nabla u = -\nabla p + \mu \nabla^2 u + f, \\
\nabla \cdot u = 0,
\end{cases} \quad \forall (x, t) \in \Omega_1 \times (0, T].$$

(2.7)

Thus this type of linearization is also referred to as Oseen iteration. Though easy to implement, Oseen iteration can only be expected to have a linear rate of convergence. Note that to correctly solve (2.7), stabilization mentioned in §2.1.1 is still required to account for the effect of the term $u^k \cdot \nabla u^{k+1}$.

In fact, in an even simpler albeit slower version of the Picard iteration, the convection term can simply consist of velocity from previous iteration, namely $u^k \cdot \nabla u^k$. In this case, each iteration is equivalent to solving a Stokes problem

$$\begin{cases}
-\mu \nabla^2 u + \nabla p = f, \\
\nabla \cdot u = 0,
\end{cases} \quad \forall x \in \Omega_1.

(2.8)

To obtain superlinear or quadratic convergence, one can adopt the Newton–type method. Since in the classical Newton’s method one has to calculate the Jacobian at every time step, it is expensive to directly adopt this scheme. Instead, a quasi–Newton method such as the Broyden’s method [14] can be used to approximate the Jacobian. More recently, Jacobian–free Newton–Krylov (JFNK) methods have been proposed to combine the Newton–type methods with Krylov iterations [13, 17, 65]. A comprehensive review of JFNK methods is given by Knoll and Keyes [67]. In particular, they point out that Newton–type method are in fact not preferred among the CFD community in general. Instead, Picard iterations and operator
splitting methods such as the projection method mentioned earlier are widely used. In the latter, the operator splitting performs two tasks: the segregation of velocity and pressure, and the linearization of the NSE.

2.1.3 Instability due to violation of LBB condition

The other numerical challenge comes from the incompressibility. To see this, let us first point out that both the (stationary) Oseen flow (2.7) and Stokes flow (2.8) can be written as a saddle point problem. For example, we have the variational formulation of (2.8) as

\[
\begin{cases}
\mu(\nabla u, \nabla v) + (p, \nabla \cdot v) = \langle f, v \rangle, & \forall v \in H^1_0(\Omega_1), \\
-(\nabla \cdot u, q) = 0, & \forall q \in L^2(\Omega_1).
\end{cases}
\]

(2.9)

Now let us define Hilbert spaces \( \mathbb{V} \) and \( \mathbb{P} \)

\[
\mathbb{V} = H^1_0(\Omega_1), \quad \mathbb{P} = L^2(\Omega_1) \triangleq \left\{ q \in L^2(\Omega_1), \int_{\Omega_1} q = 0 \right\},
\]

and corresponding bilinear operators

\[
a(u, v) = \mu \int_{\Omega_1} \nabla u : \nabla v, \quad b(v, q) = -\int_{\Omega_1} q \nabla \cdot v,
\]

then equation (2.9) becomes

\[
\begin{cases}
a(u, v) + b(v, p) = \langle f, v \rangle, & \forall v \in \mathbb{V}, \\
b(u, q) = 0, & \forall q \in \mathbb{P}.
\end{cases}
\]

This is a special case of the saddle point problem

\[
\begin{cases}
a(u, v) + b(v, p) = \langle f, v \rangle, & \forall v \in \mathbb{V}, \\
b(u, q) = \langle g, q \rangle, & \forall q \in \mathbb{P}.
\end{cases}
\]

(2.10)
Here we solve for \((u, p) \in \mathbb{V} \times \mathbb{P}\), with \(a\) and \(b\) both being bilinear operators on Hilbert spaces \(\mathbb{V}\) and \(\mathbb{P}\), and \((f, g) \in \mathbb{V}^* \times \mathbb{P}^*\).

Since both (2.7) and (2.8), the linearized versions of the NSE, are of the form (2.10), we can examine some numerical problems of solving the NSE by studying (2.10). In fact, the Ladyzhenskaya–Babuška–Brezzi (LBB) condition [3, 10, 69] characterizes its well-posedness.

**Theorem** (Ladyzhenskaya–Babuška–Brezzi [3, 10, 69]). *With continuous bilinear operators \(a(\cdot, \cdot)\) and \(b(\cdot, \cdot)\), saddle point problem (2.10) is well-posed if and only if \(\exists \alpha \in \mathbb{R}\) and \(\beta \in \mathbb{R}\) such that

\[
\inf_{u \in \mathcal{Z}} \sup_{v \in \mathbb{V}} \frac{a(u, v)}{\|u\| \|v\|} = \inf_{v \in \mathcal{Z}} \sup_{u \in \mathbb{V}} \frac{a(u, v)}{\|u\| \|v\|} = \alpha > 0, \tag{2.11a}
\]

\[
\inf_{q \in \mathbb{P}} \sup_{v \in \mathbb{V}} \frac{b(v, q)}{\|v\| \|q\|} = \beta > 0. \tag{2.11b}
\]

Here \(\mathcal{Z}\) is the kernel of the linear operator induced by \(b(\cdot, \cdot)\), i.e.,

\[
\mathcal{Z} \triangleq \{ v \in \mathbb{V}, b(v, q) = 0, \forall q \in \mathbb{P} \}.
\]

When (2.11) holds, we have

\[
\|u\|_\mathbb{V} + \|p\|_\mathbb{P} \leq \|f\|_{\mathbb{V}^*} + \|p\|_{\mathbb{P}^*}. \tag{2.12}
\]

Equation (2.11) is usually referred to as the LBB stability condition.

In FEM solution of the NSE, the counterpart of (2.10) is the linear system

\[
\begin{pmatrix}
A & D^T \\
D & -C
\end{pmatrix}
\begin{pmatrix}
U \\
P
\end{pmatrix} = \begin{pmatrix}
f \\
g
\end{pmatrix}. \tag{2.13}
\]

Here \(A = \text{diag}(A_1, \ldots, A_n)\) is a block diagonal matrix, with each block corresponding to a discrete convection–diffusion operator, together with terms from time integral. \(D^T\) represents the discrete gradient operator, and \(D\) represents discrete divergence operator. Equation (2.13) forms a *generalized saddle point system* [6]. When \(C = 0\)
and \((V_h, Q_h)\) does not meet LBB condition, the ill-posedness of the problem is manifested as the coefficient matrix of (2.13) being singular, and the numerical solution being unstable.

In practice, due to implementation concern in many FEM schemes equal interpolation of \(u\) and \(p\) are used. Since those interpolation spaces do not satisfy the LBB condition, stabilization is necessary for those schemes. Such stabilization methods result in nonzero \(C\) in (2.13), hence sometimes they are referred to as “pseudo-compressibility methods”. By the fashion that the equation of continuity is perturbed, they can be categorized mainly into three methods [70, 86],

\[
\nabla \cdot u = \varepsilon \nabla^2 p, \quad \text{Petrov-Galerkin method, (2.14a)} \\
\nabla \cdot u = -\varepsilon p, \quad \text{Penalty method, (2.14b)} \\
\nabla \cdot u = -\varepsilon \frac{\partial p}{\partial t}, \quad \text{artificial compressibility method. (2.14c)}
\]

Recall that the projection methods (2.1)–(2.4) do not explicitly contain any pressure stabilization terms, however, the projection methods “work better than they supposed to” [45, p. 566]. This ambiguity of the nature of the projection method, especially about its stability, is largely clarified by Rannacher [86]. He has shown that the original projection method is equivalent to a monolithic scheme of format (2.13), with pressure stabilization enforced by adding a term \(\Delta t \nabla^2 p\) on the right hand side of the mass conservation equation. Therefore, the original projection method is equivalent to a monolithic scheme with perturbation of the incompressibility condition in the form of (2.14a). Based on this observation, a monolithic schemes is proposed by Soto et al. [96, 97] and Codina [25]. It is shown by Codina [25] that the OSS method’s pressure stabilization capability can be interpreted using the subscale decomposition argument, as well as the monolithic formulation. The FEM solver used in this study is based on the scheme proposed by Codina and Soto [96, 97]. For details, see §3.1.
2.2 Numerical methods for potential flows

Due to the nature of Laplace’s equation (3.14), the numerical solution of the governing equation of potential flow is less sophisticated than that of the NSE. Other than the FEM for elliptic equations [22], the BEM that takes advantage of the Green’s function is also widely adopted for this kind of problems [9]. For a problem with the same domain, since the BEM solution is based on discretizing only the boundary instead of the whole domain as in the FEM, the linear system produced by the BEM is of smaller size. However, coefficient matrices in BEM turn out to be in general fully populated, therefore both the formulation of the matrices and the matrix-vector multiplication as used in an iterative solution scheme require much more effort, compared with solving linear systems with a sparse coefficient matrix, such as those produced by a FEM discretization. The remedy to this problem, is to apply Fast Multipole Method (FMM) to the process of matrix formulation and multiplication. Proposed by Rokhlin [88] and later refined by Greengard and Rokhlin [49], the FMM was first used to speed up the calculation of long-ranged n-body forces, then later applied to accelerate the iterative solvers [50, 81]. It can improve the complexity of matrix-vector multiplication in an iterative solver from $O(N^2)$ to $O(N)$. The FMM acceleration of the BEM solver used in this study was implemented by Fochesato and Dias [43].

2.3 Numerical methods for free surface tracking

The numerical schemes of both viscous flow (FEM) and potential flow (BEM), when applied to water wave problem, need to capture or track the free surface. The most popular interface capturing/tracking methods are volume–of–fluid (VOF) techniques [79] and level set methods [80]. Unlike Lagrangian methods, these Eulerian methods contain a priori curvature regularization to handle merging characteristics. On the other hand, it is found that Lagrangian methods maintain filamentary interface structures better, while Eulerian techniques violates mass conservation or produce “blobs” during locally enforced mass conservation, when
the filaments are too thin to be resolved by the grid [87]. Between the above two Eulerian methods, VOF is amenable to implementation, but is relatively crude by approximating the front through volume fraction, and is problematic for complex flows [92, p. 41].

In order to combine the best properties of Eulerian schemes and Lagrangian methods, Enright et al. [36] proposed a hybrid particle level set method that brings the Lagrangian marker particles into the level set scheme. In this technique, randomly distributed marker particles near the interface act as a remedy when the level set scheme fails to capture the interface accurately, by providing sub-scale information to rebuild the level set function locally. This technique is adopted in the FEM solver used in this study.

The free surface tracking in the potential flow solver is realized in a different fashion by using the mixed-Eulerian-Lagrangian (MEL) approach [54, 73]. Being essentially a Lagrangian-type technique, within each time step one first solves the potential flow problem (3.14), then uses the solution to update the free surface boundary conditions (1.19) and (1.21). For details of the MEL method, see §3.2.

### 2.4 Domain decomposition methods

Domain Decomposition (DD) methods was first introduced by Schwarz [91] to study the classical Dirichlet boundary value problems associated with Laplace’s equation. The iterative method named after him have been extended since for many applications. Early such works include that of Bjørstad and Widlund [8], Cambier et al. [15], Dinh et al. [31], Dryja [34], Glowinski et al. [46], among others. Recent overviews of this topic include the review by Xu [105], Xu and Zou [106], the books by Khoromskij and Wittum [66], Mathew [77], Quarteroni and Valli [85], Toselli [102] and Wohlmuth [104].

It is natural to apply DD methods to multiphysics problems, and their ideas extend to fields such as Fluid–Structure Interaction (FSI) problems and heterogeneous flow problems. In this case, the presence of subdomains is intrinsic and the derivation of matching conditions at the interface sometimes proves to be intuitive.
In general, a DD method for unsteady problems is characterized by what information is exchanged during subdomains communication, and by the time stepping scheme defining when such a communication happens.

Heterogeneous flow models, like any other DD methods based models, can be categorized as either overlapping or non-overlapping. Since overlapping models provide heuristic basis for Schwarz-like methods, they are adopted by many applications. Usually the nodal velocity is used as the constraint of the subsolutions in the overlapping region [16, 18, 61, 93]. Such a method usually requires in-time iterations. The proper size of the overlapping region pertaining to the convergence rate of in–time iteration remains an open question [32, p. 50].

As to the temporal pattern of communication among subdomains, DD methods can be explicit or implicit. Explicit schemes such as the classical staggered scheme may suffer from instability and are usually first employed during prototyping stage, due to their easy implementation. For implicit time discretization and subdomain iterations, see [85, Chap. 8] and references therein. In this study a staggered scheme is adopted for the coupling between the two flow models.

DD flow models are studied, e.g., in [38, 39, 40, 41, 90]. In particular, problems of the NSE coupled with a Stokes flow and the NSE coupled with an Oseen flow are studied in [40, 41], respectively. In [40], the coupled stationary NSE and Stokes model

\[
\begin{cases}
  \mathbf{u}_1 \cdot \nabla \mathbf{u}_1 = -\nabla p_1 + \nu \nabla^2 \mathbf{u}_1 + \mathbf{f} & \text{in } \Omega_1, \\
  \nabla \cdot \mathbf{u}_1 = 0 & \text{in } \Omega_1, \\
  \mathbf{u}_1 = 0 & \text{in } \partial \Omega_1 \cap \partial \Omega, \\
  0 = -\nabla p_2 + \nu \nabla^2 \mathbf{u}_2 & \text{in } \Omega_2, \\
  \nabla \cdot \mathbf{u}_2 = 0 & \text{in } \Omega_2, \\
  \lim_{|x| \to \infty} \mathbf{u}_2 = \mathbf{u}_\infty & \text{in } \Omega_2,
\end{cases}
\]  

(2.15)

is studied, and the stress matching condition is established by looking for the
Poincaré-Steklov operator $\lambda$ on $\Gamma$ that matches dynamical stress from $\Omega_1$:  

$$
\lambda(u_2, p_2) = \left[-\left(p_1 + \frac{1}{2}|u_1|^2\right) \mathbb{I} + 2\nu \mathbb{E}\right] \cdot n.
$$

The tensor notation version of this condition is  

$$
\sigma_{ij} n_j - \frac{1}{2}|u_j u_j|^2 n_i = -\lambda(u_2, p_2), \quad (2.16)
$$

where $\sigma_{ij}$ is the stress tensor.

Through a singular perturbation analysis the formulation of $\lambda$ was found to be the stress at the normal direction $n$, i.e. $\lambda = -\sigma_n$ from $\Omega_2$. Then the matching conditions are  

$$
Q_1 = u_1 - u_2, \quad (2.17a)
$$

$$
Q_2 = \left[-(p_1 + \frac{1}{2}|u_1|^2) + 2\mu \mathbb{E}_1\right] \cdot n - \left[-p_2 + 2\mu \mathbb{E}_2\right] \cdot n. \quad (2.17b)
$$

We can see that the first condition matches the velocity, while the second one matches normal stress, with dynamic pressure $|u_1|^2/2$ considered in the NSE subdomain to include the effect of high Reynolds number. Also we can see that $Q_2 = 0$ is equivalent to a constraint on the jump of normal stress across $\Gamma$  

$$
||\sigma_n|| = \frac{1}{2}|u_1|^2. \quad (2.18)
$$

Remark 2.4.1. If the coupled problem (1.22) is used as an alternative of the NSE problem in the whole domain $\Omega$, then increasing the size of $\Omega_1$ with respect to $\Omega_2$ reduces the modeling error of the heterogeneous model.
3 Numerical methods

3.1 FEM for Navier-Stokes Equations

The method adopted in the incompressible flow solver in this study is based on the orthogonal sub-scales method (OSS) [24, 26]. It adopts the same philosophy as the Variational Multiscale (VMS) method [59] by modeling the subscale field. Recall that after linearization instead of solving the NSE one can solve the Oseen flow problem (2.7) during iterations, hence it is instructive to demonstrate the fundamental concept of the OSS method by looking at the stationary version of (2.7)

\[
\begin{aligned}
    a \cdot \nabla u &= -\nabla p + \mu \nabla^2 u + f, \\
    \nabla \cdot u &= 0,
\end{aligned}
\]

\( \forall (x, t) \in \Omega_1 \times (0, T]. \) (3.1)

Here \( a \in V \) represents the mean stream flow velocity. In numerical schemes for the NSE, \( a \) often is the velocity from previous time step or iteration.

Recall the Sobolev spaces defined in §1.1, let \( V \triangleq H^1(\Omega_1), Q \triangleq L^2(\Omega_1), V_0 \triangleq H^1_0(\Omega_1), W \triangleq V \oplus Q \) and \( W_0 \triangleq V_0 \oplus Q \). Then \( W \) and \( W_0 \) are Hilbert spaces, with inner product constructed from direct sum:

\[
(w_1, w_2) \triangleq (v_1, v_2) + (p_1, p_2), \quad \forall w_1 = (v_1, p_1) \in W, w_2 = (v_2, p_2) \in W.
\]

Henceforth we derive the FEM scheme for the stationary Oseen flow (3.1). Its variational formulation is

\[
\mu(\nabla u, \nabla v) + (a \cdot \nabla u, v) - (p, \nabla \cdot v) + (q, \nabla \cdot u) = (f, v), \quad \forall (v, q) \in W_0. \quad (3.2)
\]
With bilinear form
\[ B((u, p), (v, q)) \triangleq \mu(\nabla u, \nabla v) + (a \cdot \nabla u, v) - (p, \nabla \cdot v) + (q, \nabla \cdot u), \]
formulation (3.2) becomes
\[ B((u, p), (v, q)) = (f, v), \quad \forall (v, q) \in W_0. \]

We also define the linear map \( L : W \to W \) such that
\[ L(w) \triangleq (-\mu \Delta u + a \cdot \nabla u + \nabla p, \nabla \cdot u), \quad \forall w = (u, p) \in W. \]

As in all VMS–like methods, we decompose the function spaces into grid–scale and sub–scale. In FEM, we seek solution \((u_h, p_h) \in V_h \times Q_h\), where \(V_h \subset V\) and \(Q_h \subset Q\) are the finite element spaces. This implies a decomposition of the unknowns
\[ u = u_h + \tilde{u}, \quad p = p_h + \tilde{p}. \]

Here \(\tilde{u}\) and \(\tilde{p}\) denote the sub–grid component of \(u\) and \(p\), respectively. In fact, we have \(\tilde{u} \in \tilde{V}\) and \(\tilde{p} \in \tilde{Q}\), such that
\[ V = V_h \oplus \tilde{V}, \quad Q = Q_h \oplus \tilde{Q}. \]

Similarly, we have the decompositions
\[ Q_0 = Q_{h,0} \oplus \tilde{Q}_0, \quad V_0 = V_{h,0} \oplus \tilde{V}_0, \quad W = W_h \oplus \tilde{W}, \quad W_0 = W_{h,0} \oplus \tilde{W}_0. \]

Here \(W_h \triangleq V_h \times Q_h, \; W_{h,0} \triangleq V_{h,0} \times Q_h\).

In FEM we solve for \((u_h, p_h) \in W_h\)
\[ B((u_h, p_h), w_h) + B((\tilde{u}, \tilde{p}), w_h) = (f, v_h), \quad \forall w_h \in W_{h,0}, \quad (3.3a) \]
\[ B((u_h, p_h), \tilde{w}) + B((\tilde{u}, \tilde{p}), \tilde{w}) = (f, \tilde{v}), \quad \forall \tilde{w} \in \tilde{W}_0, \quad (3.3b) \]
and test functions are

\[ w_h \triangleq (v_h, q_h) \in W_{h,0}, \quad \tilde{w} \triangleq (\tilde{v}, \tilde{q}) \in \tilde{W}_0. \]

As in all VMS–like methods, now we need to model the subscale effect on the FEM solution, i.e., the second term of (3.3a). Modeling is necessary because the system (3.3a) is not closed with unknown subscale term.

We draw the clue of the modeling from the subscale equation (3.3b). It has the expansion

\[
\sum_e \int_{\partial e} \tilde{v} \cdot (p_h n + \mu n \cdot \nabla u_h) \, d\Gamma + \sum_e (\mathcal{L}(u_h, p_h) + \mathcal{L}(\tilde{u}, \tilde{p}), \tilde{w}) = \sum_e (f, \tilde{v}).
\]

Here we have the elementwise inner product \((\cdot, \cdot)\). When the exact traction is assumed to be continuous, the first sum is zero. Therefore the subscale equation (3.3b) is equivalent to

\[
(\mathcal{L}(\tilde{u}, \tilde{p}), \tilde{w}) = (F - \mathcal{L}(u_h, p_h), \tilde{w}), \quad \forall \tilde{w} \in \tilde{W}_0, \quad \forall e \in T_h,
\]

where \( F \triangleq (f, 0) \in W \), assuming enough regularity.

We hope to calculate \((\tilde{u}, \tilde{p})\) from (3.4) and introduce it to (3.3a) in order to formulate a modified variational form. To achieve this, in the OSS method we assume

\[
\tilde{W}_0 \subset \tilde{W} \subset W_h^\perp.
\]

Then equation (3.4) is equivalent to

\[
\mathcal{L}(\tilde{u}, \tilde{p}) = F - \mathcal{L}(u_h, p_h) + \tilde{w}_h, \quad \forall e \in T_h,
\]

for certain \( \tilde{w}_h \in W_h \). Introducing \( \mathcal{P} \) as the \( L^2 \)-projection onto \( W_h \), we further
assume

\[ \hat{w}_h = \mathcal{P}(F - \mathcal{L}(u_h, p_h)), \quad \forall e \in T_h. \]

Therefore

\[ \mathcal{L}(\tilde{u}, \tilde{p}) = \mathcal{P}^\perp(F - \mathcal{L}(u_h, p_h)), \quad \forall e \in T_h. \]

Here \( \mathcal{P}^\perp \triangleq \mathcal{I} - \mathcal{P} \), where \( \mathcal{I} \) is the identity map on \( W_h \). In theory, if we can find the inverse of \( \mathcal{L} \), then \( (\tilde{u}, \tilde{p}) \) becomes readily available. Instead, we simply assume the effect of \( \mathcal{L}^{-1} \) can be modeled by some linear operator \( S_e \), i.e.,

\[ (\tilde{u}, \tilde{p}) = S_e \mathcal{P}^\perp(F - \mathcal{L}(u_h, p_h)), \quad \forall e \in T_h. \]  \( (3.6) \)

Here subscript \( e \) is used to emphasize that \( S \) in general differs between elements.

How to design \( S \) is the cornerstone of VMS–like methods and the choice affects the stabilization significantly. Hence \( S \) is referred to as the stabilization parameter. A heuristic option for \( S_e \) is to decouple the two components as

\[ S_e(w) = \tau_{1,e}v + \tau_{2,e}q, \quad \forall w = (v, q) \in W. \]  \( (3.7) \)

Stabilization parameters \( \tau_{1,e} \) and \( \tau_{2,e} \) differ in general from scheme to scheme.

Combining (3.6)–(3.7), we have derived the modeled subscale solution as

\[ (\tilde{u}, \tilde{p}) = (\tau_{1,e} \mathcal{P}^\perp(f + \mu \nabla^2 u_h - a \cdot \nabla u_h - \nabla p_h), \tau_{2,e} \mathcal{P}^\perp(\nabla \cdot u)) , \forall e \in T_h. \]  \( (3.8) \)

Now we return to (3.3a), and assuming continuous traction across element
boundaries as in the expansion of (3.3b), we have

$$
\mathcal{B}((\tilde{\mathbf{u}}, \tilde{\mathbf{p}}), w_h) = \mathcal{B}((\tilde{\mathbf{u}}, \tilde{\mathbf{p}}), (\mathbf{v}_h, q_h)) \\
= \sum_e \left[ - (\tilde{\mathbf{u}}, \mu \nabla^2 \mathbf{v}) + (\mathbf{a}, \nabla(\tilde{\mathbf{u}}, \mathbf{v}_h)) \right] \\
- \sum_e [(\tilde{\mathbf{p}}, \nabla \cdot \mathbf{v}_h) + (\tilde{\mathbf{u}}, \nabla q_h + \mathbf{a} \cdot \nabla \mathbf{v}_h)].
$$

(3.9)

In the first sum of (3.9), we choose to neglect the second-order derivative term. We also know the second term vanishes due to the orthogonal subscale space assumption (3.5). Introducing (3.8) into equation (3.9), we have

$$
\mathcal{B}((\tilde{\mathbf{u}}, \tilde{\mathbf{p}}), w_h) = \sum_e \left[ \tau_2,e (\mathcal{P}^\perp (\nabla \cdot \mathbf{u}_h), \nabla \cdot \mathbf{v}_h) \\
+ \tau_1,e (\mathcal{P}^\perp (\mathbf{f} + \mu \nabla^2 \mathbf{u}_h - \mathbf{a} \cdot \nabla \mathbf{u}_h - \nabla p_h), \nabla q_h + \mathbf{a} \cdot \nabla \mathbf{v}_h) \right].
$$

We again neglect the second-order derivatives, and also assume \(\mathbf{f} \in \mathbb{V}_h\), hence \(\mathcal{P}^\perp (\mathbf{f}) = 0\). Thus

$$
\mathcal{B}((\tilde{\mathbf{u}}, \tilde{\mathbf{p}}), w_h) = - \sum_e \left[ \tau_2,e (\mathcal{P}^\perp (\nabla \cdot \mathbf{u}_h), \nabla \cdot \mathbf{v}_h) \\
+ \tau_1,e (\mathcal{P}^\perp (-\mathbf{a} \cdot \nabla \mathbf{u}_h - \nabla p_h), \nabla q_h + \mathbf{a} \cdot \nabla \mathbf{v}_h) \right].
$$

(3.10)

Introducing (3.10) into (3.3a), we finally have the stabilized FEM formulation in the OSS as

$$
\mathcal{B}((\mathbf{u}_h, p_h), w_h) + \sum_e \left[ \tau_2,e (\mathcal{P}^\perp (\nabla \cdot \mathbf{u}_h), \nabla \cdot \mathbf{v}_h) \\
+ \tau_1,e (\mathcal{P}^\perp (\mathbf{a} \cdot \nabla \mathbf{u}_h + \nabla p_h), \mathbf{a} \cdot \nabla \mathbf{v}_h + \nabla q_h) \right] = (\mathbf{f}, \mathbf{v}_h).
$$

(3.11)

A slightly different form of (3.11), in which the orthogonal projections of advection
term and pressure term are controlled separately, is

\[
\mathcal{B}( (u_h, p_h), w_h ) + \sum_e [ \tau_{2,e} ( P^\perp (\nabla \cdot u_h), \nabla \cdot v_h ) \\
+ \tau_{1,e} ( P^\perp (a \cdot \nabla u_h), a \cdot \nabla v_h ) + \tau_{1,e} ( P^\perp (\nabla p_h), \nabla q_h ) ] = ( f, v_h ).
\]

Notice that here the advection stabilization term

\[
\tau_{1,e} ( P^\perp (a \cdot \nabla u_h), a \cdot \nabla v_h )
\]

is similar to SUPG as in the stabilization expression (2.6). On the other hand, the pressure stabilization term

\[
\tau_{1,e} ( P^\perp (\nabla p_h), \nabla q_h )
\]

can be combined with the incompressibility constraint in \( \mathcal{B} ( \cdot, \cdot ) \) to provide a stabilization similar to the pseudo-compressibility method (2.14a), namely

\[
( \nabla \cdot u_h, q_h ) = -\tau_{1,e} ( P^\perp (\nabla p_h), \nabla q_h )
\]

In summary, we make two modifications to (3.12) [96, 97]

- Neglecting the \( \tau_{2,e} \) stabilization term.
- Adding the preconditioning term \( \Delta t ( \nabla p_h^{n+1,i} - \nabla p_h^{n+1,i-1}, \nabla q_h ) \) in the incompressibility constraint to enforce the convergence of the block Gauss-Seidel uncoupled solution; see below.

This modified version of (3.12), as shown in Scheme 3.1.1, is implemented in the NSE solver adopted in this study.

**Scheme 3.1.1.** Given \( u_h^n \), find \( ( u_h^{n+1}, p_h^{n+1}, \pi_h^{n+1}, \xi_h^{n+1} ) \) in \( V_h \times Q_h \times V_{h,0} \times V_{h,0} \).
so that \( \forall (v_h, q_h, \bar{v}_h, \bar{v}_h) \in V_h \times Q_h \times V_{h,0} \times V_{h,0} \)

\[
\begin{align*}
\frac{1}{\Delta t} & \left( u_h^{n+1,(i)} - u_h^n, v_h \right) + \left( u_h^{n+\vartheta,(i-1)} \cdot \nabla u_h^{n+\vartheta,(i)}, v_h \right) \\
\quad & + \left( \mu \nabla u_h^{n+\vartheta,(i)}, \nabla v_h \right) - \left( p_h^{n+1,(i-1)}, \nabla \cdot v_h \right) \\
\quad & + \left( \tau (u_h^{n+\vartheta,(i-1)} \cdot \nabla u_h^{n+\vartheta,(i)} - \pi_h^{n+\vartheta,(i-1)}), u_h^{n+\vartheta,(i-1)} \cdot \nabla v_h \right) \\
\quad & = (f^{n+\vartheta}, v_h) + \left( \sigma^{n+\vartheta,(i-1)} \cdot n, v_h \right)_{\Gamma_N}, \\
\Delta t \left( \nabla p_h^{n+1,(i)} - \nabla p_h^{n+1,(i-1)}, \nabla q_h \right) & + \left( \tau (\nabla p_h^{n+1,(i)} - \xi^{n+1,(i-1)}), \nabla q_h \right) \\
\quad & = - \left( \nabla \cdot u_h^{n+1,(i)}, q_h \right), \\
\left( \pi_h^{n+\vartheta,(i)}, \bar{v}_h \right) & = \left( u_h^{n+\vartheta,(i)} \cdot \nabla u_h^{n+\vartheta,(i)}, \bar{v}_h \right), \\
\left( \xi^{n+1,(i)}, \bar{v}_h \right) & = \left( \nabla p_h^{n+1,(i)}, \bar{v}_h \right).
\end{align*}
\] (3.13a)

In Scheme 3.1.1, for a variable \( x \), we define

\[
x^{n+\vartheta} \triangleq \vartheta x^{n+1} + (1 - \vartheta) u^n, \quad \vartheta \in [0, 1].
\]

Also \( n \) indicates the time step and superscript \( i \) stands for the in–time block Gauss-Seidel iteration number. This monolithic scheme treats the advection term in an implicit manner, therefore it allows large time step. Meanwhile, the in-time iteration decouples the solution of velocity and pressure, in order to avoid the expensive computing cost that usually accompanies monolithic schemes.

**Remark 3.1.2.** Scheme 3.1.1 can also be interpreted as an incremental–pressure projection scheme. Here the convection stabilization term is the orthogonal subscale component of the SUPG term; see equation (2.6) and Remark (2.1.2). Similarly, the pressure stabilization term is the orthogonal subscale component of the pseudo–compressibility term from (2.14a). In fact, other than the stabilization terms, (3.13) is the nothing but a block–iteration scheme for the incremental–pressure projection method (2.5).

In the ICFD implementation of the Scheme 3.1.1, a second-order backward differencing (BDF2) is used for time integral, and the \( P1-P1 \) triangle or tetrahedron
elements are used for triangulation. This type of element does not meet the LBB condition, and the OSS scheme provides stabilization toward pressure. The decision to use this element in implementation is based on several concerns. First, there is a large code base available. Second, the simple triangle and tetrahedron elements are consistent with the mesh generation module adopted. Finally, this simple element is amenable to be incorporated into greater software development projects.

For solving the linear systems, first the non-symmetric discrete momentum equation (3.13a) is solved using a standard GMRES method with diagonal preconditioning. Then, the pressure equation (3.13b) is computed by using a conjugate gradient with incomplete LU pre-conditioning (CG-ILU) solver for isotropic meshes, or a CG-Linelet solver for highly stretched grids [95]. Standard mass lumping is used in (3.13c) and (3.13d).

3.2 BEM solver for potential flow

In domain $\Omega_2$, we have the potential flow model

$$\nabla^2 \varphi = 0, \quad \forall (x, t) \in \Omega_2 \times (0, T],$$

$$\varphi(x, t) = \varphi^*(x, t), \quad \forall (x, t) \in \Gamma_f \cup \Gamma_D \times (0, T],$$

$$\frac{\partial \varphi}{\partial n} = g^*(x, t), \quad \forall (x, t) \in \Gamma_N \times (0, T].$$

Here $\Gamma_f$ denotes the free surface moving boundary, $\Gamma_D$ the Dirichlet boundary, and $\Gamma_N$ the Neumann boundary. In application we usually have $\Gamma_D = \emptyset$, and $\Gamma_N$ be the wall boundary and the wave–maker boundary. In our coupled model, $\Gamma_D = \Gamma$, see §3.3.1 for details.

Let $G_x(\bar{x})$ be the Green’s function. In $\mathbb{R}^3$ it is given as

$$G_x(\bar{x}) = \frac{1}{4\pi |\bar{x} - x|}, \quad \frac{\partial G_x(\bar{x})}{\partial n} = -\frac{1}{4\pi} \frac{(\bar{x} - x) \cdot n}{|\bar{x} - x|^3}.$$  

By Green’s third identity, the solution to (3.14) at the interior of $\Omega_2$ can be described
by the boundary integral [7]

$$
\varphi(x) = \int_{\partial \Omega_2(x)} \frac{\partial \varphi(\bar{x})}{\partial n} G_x(\bar{x}) - \varphi(\bar{x}) \frac{\partial G_x(\bar{x})}{\partial n} \, d\Gamma, \quad \forall x \in \Omega_2.
$$

(3.15)

Namely, the solution can be obtained through (3.15) when the boundary data \( \frac{\partial \varphi(\bar{x})}{\partial n} \) and \( \varphi(\bar{x}) \) are available on \( \partial \Omega_2 \).

This motivate us to solve the problem (3.14) on the domain boundary. For that we apply Green’s second identity to transform (3.14) into a boundary integral equation

$$
\alpha(x) \varphi(x) = \int_{\partial \Omega_2(x)} \frac{\partial \varphi(\bar{x})}{\partial n} G_x(\bar{x}) - \varphi(\bar{x}) \frac{\partial G_x(\bar{x})}{\partial n} \, d\Gamma, \quad \forall x \in \partial \Omega_2
$$

(3.16)

Here

$$
\alpha(x) = \frac{\vartheta(x)}{4\pi}, \quad \forall x \in \partial \Omega_2,
$$

(3.17)

where \( \vartheta \) is the exterior solid angle at the boundary point \( x \). For a smooth boundary, \( \vartheta = 2\pi \).

Equation (3.16) implies that \( \varphi \) can be obtained by evaluating the left hand side integral. Introducing the boundary data \( \varphi^* \) and \( g^* \) from (3.14) into (3.16), we have

$$
\alpha(x) \varphi(x) = \int_{\Gamma_D \cup \Gamma_f(x)} \frac{\partial \varphi(\bar{x})}{\partial n} G_x(\bar{x}) - \varphi(\bar{x}) \frac{\partial G_x(\bar{x})}{\partial n} \, d\Gamma + \int_{\Gamma_N(x)} g^*(x) G_x(x) - \varphi(x) \frac{\partial G_x(x)}{\partial n} \, d\Gamma, \quad \forall x \in \partial \Omega_2.
$$

(3.18)

Therefore we need to solve \( \varphi \) on \( \Gamma_N \) and \( \psi = \frac{\partial \varphi(\bar{x})}{\partial n} \) on \( \Gamma_D \cup \Gamma_f \). We denote these two unknowns by the pair \( (\varphi, \psi) \).

Note that in boundary integral equations (3.16) and (3.18) we have omitted the time \( t \) to simplify notation. Since the governing equation contains no time derivative explicitly, the time evolution is performed through the change of the domain and the boundary condition. At the free surface \( \Gamma_f(t) \), both \( \varphi(x,t)^* \) and \( x \) require
an update to satisfy the kinematic boundary condition (1.19) and the dynamical boundary condition (1.21).

An explicit second order time integration scheme for (1.19) and (1.21) is proposed in [53, 54], based on Taylor’s expansion:

\[
x(t + \Delta t) = x(t) + \Delta t \frac{Dx}{Dt} + \frac{\Delta t^2}{2} \frac{D^2 x}{Dt^2} + O(\Delta t^3),
\]

\[
\varphi(t + \Delta t) = \varphi(t) + \Delta t \frac{D\varphi}{Dt} + \frac{\Delta t^2}{2} \frac{D^2 \varphi}{Dt^2} + O(\Delta t^3).
\]

(3.19)

(3.20)

To calculate these updates, we need two pairs of material derivatives:

\[
\left( \frac{Dx}{Dt}, \frac{D\varphi}{Dt} \right), \quad \left( \frac{D^2 x}{Dt^2}, \frac{D^2 \varphi}{Dt^2} \right).
\]

The first order derivatives \( \left( \frac{Dx}{Dt}, \frac{D\varphi}{Dt} \right) \) are obtained from boundary conditions (1.19) and (1.21), which in turn require solving for \((\varphi, \psi)\) from (3.18). To get the second order derivatives \( \left( \frac{D^2 x}{Dt^2}, \frac{D^2 \varphi}{Dt^2} \right) \), we apply \( \frac{D}{Dt} \) to (1.19) and (1.21), thus on \( \Gamma_f \)

\[
\frac{D^2 x}{Dt^2} = \frac{D}{Dt} \left( -gz \left( \varphi, \psi \right) - p^* \rho \right) = -gu_3 + u \cdot \frac{\partial u}{\partial t} + (u \cdot \nabla u) \cdot u.
\]

Note that \( (u \cdot \nabla u) \cdot u = 0 \). To see this, let us introduce \( u = \nabla \varphi \). Thus using tensor notation, \( (u \cdot \nabla u) \cdot u = 0 \) becomes

\[
\frac{\partial u_i}{\partial x_j} u_i = u_j \frac{\partial^2 \varphi}{\partial x_i \partial x_j} \frac{\partial \varphi}{\partial x_i} = u_j \frac{\partial \varphi}{\partial x_j} \frac{\partial^2 \varphi}{\partial x_i \partial x_i} = 0,
\]

since \( \frac{\partial^2 \varphi}{\partial x_i \partial x_i} = 0 \). Therefore, to calculate the second order derivatives we need \( \frac{\partial u_i}{\partial x_j} \).

For this we solve a boundary integral equation similar to (3.16), only with \( (\varphi, \psi) \) replaced by \( (\varphi, \frac{\partial \varphi}{\partial n}) \).

In summary, to calculate \((\varphi, \psi)\) with third-order accuracy at the free surface, we solve two integral equations of in form of (3.16). At \( \Gamma_f \), boundary data is first obtained by updating \( \varphi \), then solving for \( (\varphi, \psi) \). Combined with the dynamical
boundary condition (1.21), this gives us the boundary data \( \frac{\partial \varphi}{\partial t} \) at \( \Gamma_f \), as required to solve the second boundary integral equation with unknowns \((\frac{\partial \varphi}{\partial t}, \frac{\partial^2 \varphi}{\partial t \partial n})\). After both the first-order and second-order terms are calculated, the geometry of \( \Gamma_f \) and the potential data on it are both updated using Taylor expansion (3.19) and (3.20).

In this study the boundary integral equations are solved using the classical collocation boundary element method [9]. Let the boundary \( \partial \Omega \) be discretized into elements \( \Gamma_i, i = 1, 2 \ldots, N \). The unknowns are values of the variables at the vertices of \( \Gamma_i \), so that (3.16) is satisfied pointwise. To demonstrate this method, here we use the pair \((\varphi, \psi)\). The case of \((\frac{\partial \varphi}{\partial t}, \frac{\partial \varphi}{\partial t \partial n})\) follows the exact same procedure.

Let \( \partial \Omega = \Gamma_D \cup \Gamma_N \). We discretize \( \Gamma_D \) into elements \( \Gamma_{D,j}, j = 1, \ldots, N_D \), and \( \Gamma_D \) into elements \( \Gamma_{N,j}, j = 1, \ldots, N_N \), so that \( N = N_D + N_N \). Let subscript \( i \) indicate the nodal value of certain quantity at \( x_i \in \Gamma \). In particular, let \( G_i \) be the Green’s function corresponding to \( x_i \in \Gamma \). Recall that superscript * indicates Cauchy data, the collocation version of (3.18) then is

\[
\alpha \varphi_i = \sum_{j=1}^{N_N} \psi_j^* \int_{\Gamma_{N,j}} G_i \, d\Gamma - \sum_{j=1}^{N_N} \varphi_j \int_{\Gamma_{N,j}} \frac{\partial G_i}{\partial n} \, d\Gamma \\
+ \sum_{j=1}^{N_D} \psi_j \int_{\Gamma_{D,j}} G_i \, d\Gamma - \sum_{j=1}^{N_D} \varphi_j^* \int_{\Gamma_{D,j}} \frac{\partial G_i}{\partial n} \, d\Gamma, \quad i = 1, 2, \ldots, N. \tag{3.21}
\]

To formulate the corresponding linear system of (3.21), we define

\[
\Phi \in \mathbb{R}^{N_N}, \quad \Phi^* \in \mathbb{R}^{N_D}, \quad \Psi \in \mathbb{R}^{N_D}, \quad \Psi^* \in \mathbb{R}^{N_N}, \\
J^N \in \mathbb{R}^{N_N \times N_D}, \quad J^N \in \mathbb{R}^{N_N \times N_N}, \quad K^N \in \mathbb{R}^{N_N \times N_N}, \quad K^{N^*} \in \mathbb{R}^{N_N \times N_D}, \\
J^D \in \mathbb{R}^{N_D \times N_D}, \quad J^D \in \mathbb{R}^{N_D \times N_N}, \quad K^D \in \mathbb{R}^{N_D \times N_N}, \quad K^{D^*} \in \mathbb{R}^{N_D \times N_D},
\]

such that

\[
\Phi = (\varphi_1, \ldots, \varphi_{N_N})^T, \quad \Phi^* = (\varphi_{1}^*, \ldots, \varphi_{N_D}^*)^T, \\
\Psi = (\psi_1, \ldots, \psi_{N_D})^T, \quad \Psi^* = (\psi_{1}^*, \ldots, \psi_{N_N}^*)^T.
\]
and

\[ J_{i,j}^N = \int_{\Gamma_{D,j}} G_i \, d\Gamma, \quad x_i \in \Gamma_N; \quad J_{i,j}^{N*} = \int_{\Gamma_{N,j}} G_i \, d\Gamma, \quad x_i \in \Gamma_N, \]

\[ K_{i,j}^{N*} = \int_{\Gamma_{D,j}} \frac{\partial G_i}{\partial n} \, d\Gamma, \quad x_i \in \Gamma_N; \quad J_{i,j}^D = \int_{\Gamma_{D,j}} G_i \, d\Gamma, \quad x_i \in \Gamma_D, \]

\[ J_{i,j}^{D*} = \int_{\Gamma_{N,j}} G_i \, d\Gamma, \quad x_i \in \Gamma_D; \quad K_{i,j}^D = \int_{\Gamma_{N,j}} \frac{\partial G_i}{\partial n} \, d\Gamma, \quad x_i \in \Gamma_D; \]

\[ K_{i,j}^N = \int_{\Gamma_{N,j}} \frac{\partial G_i}{\partial n} \, d\Gamma + \alpha \delta_{ij}, \quad x_i \in \Gamma_N; \]

\[ K_{i,j}^{D*} = \int_{\Gamma_{D,j}} \frac{\partial G_i}{\partial n} \, d\Gamma + \alpha \delta_{ij}, \quad x_i \in \Gamma_D. \]

Then (3.21) is equivalent to

\[
\begin{bmatrix}
  K^N & -J^N \\
  K^D & -J^D
\end{bmatrix}
\begin{bmatrix}
  \Phi \\
  \Psi
\end{bmatrix}
=
\begin{bmatrix}
  J^{N*} \Psi - K^{N*} \Phi^* \\
  J^{D*} \Psi - K^{D*} \Phi^*
\end{bmatrix}. \tag{3.22}
\]

To emphasize that \( \Phi^* \) could be our unknown at the interface, let us write the system (3.22) as

\[
\begin{bmatrix}
  K^N & -J^N & K^{N*} \\
  K^D & -J^D & K^{D*}
\end{bmatrix}
\begin{bmatrix}
  \Phi \\
  \Psi \\
  \Phi^*
\end{bmatrix}
=
\begin{bmatrix}
  J^{N*} \Psi^* \\
  J^{D*} \Psi^*
\end{bmatrix}. \tag{3.23}
\]

We will return to (3.23) when derive the algebraic formulation of our DD method in §3.3.2.

The BEM formulation (3.21)–(3.22) was applied to 3D water wave problems by Grilli et al. [54], with (3.22) being solved using a standard GMRES method with diagonal preconditioning. An improvement was suggested by Fochesato and Dias [43], where the matrix-vector product operation in GMRES together with the calculation of the matrix entries, are accelerated using the fast multipole method (FMM).
3.3 Domain decomposition method for the heterogeneous model

For the sake of exposition, we look at a problem that is similar to (1.22). Let us consider two-fluid flow with a movable interface, such as water-oil and water-air coupling problems. At the interface, the boundary conditions for two-fluid problem are the kinematic boundary condition (1.19) and dynamical boundary condition (1.21). However, the interface in the DD model is not moving and is described in a Eulerian representation. Though we can derive the matching conditions on physical grounds, there is no actual physical description of such an interface. Keeping this distinction in mind, we note two problems at the interface admit similar behavior in terms of mass and momentum conservation. In particular, It is shown in [62, 64] that the pressure jump at such interfaces is

\[ [p] = 2[\mu] \frac{\partial u_n}{\partial n} - \gamma \kappa. \]  

(3.24)

Here the first term on the right is the jump of the normal stress contributed by the deviatoric stress tensor, and the second term is the surface tension. Here \( \gamma \) is the surface tension coefficient and \( \kappa \) is the curvature of \( \Gamma \).
3.3.1 The non–overlapping domain decomposition method

Now let us complete the coupled formulation in §1.4. The problem, stated for classical solutions, is to solve for \((u, p, \varphi) \in C^2(\Omega_1) \times C^1(\Omega_1) \times C^2(\Omega_2)\) in

\[
\frac{\partial u}{\partial t} + u \cdot \nabla u = -\nabla p + \mu \nabla^2 u + f, \quad \forall (x, t) \in \Omega_1 \times (0, T],
\]

\[
\nabla \cdot u = 0, \quad \forall (x, t) \in \Omega_1 \times (0, T],
\]

\[
u = u^*, \quad \forall (x, t) \in (\partial \Omega_1 \cap \partial \Omega) \times (0, T],
\]

\[
\nabla^2 \varphi = 0, \quad \forall (x, t) \in \Omega_2 \times (0, T],
\]

\[
\varphi = \varphi^*, \quad \forall (x, t) \in (\partial \Omega_2 \cap \partial \Omega) \times (0, T],
\]

\[
Q_1(u, p, \varphi) = 0, \quad \forall (x, t) \in \Gamma \times (0, T],
\]

\[
Q_2(u, p, \varphi) = 0, \quad \forall (x, t) \in \Gamma \times (0, T].
\]

Recall that for a field quantity \(a\), we use \(a_i = a|_{\Omega_i}, \ i = 1,2, \) and \(a_\Gamma = a|_\Gamma\), then we have the matching conditions (3.25f) and (3.25g) in this study as

\[
Q_1 = u_1 - u_2,
\]

\[
Q_2 = \left[- \left(p_1 + \frac{1}{2}|u_1|^2\right) + 2\mu E_1\right] \cdot n - \left[-p_2 + \frac{1}{2}|
abla \varphi|^2\right] \cdot n,
\]

which are similar to those in (2.17). Introducing \(Q_1 = 0\) into the condition \(Q_2 = 0\), we have

\[
[p] = 2\mu n T E_1 n.
\]

This is similar to the jump condition (3.24), and we can have negligible \(\mu\) \textit{a priori}, therefore \(Q_2\) becomes the matching condition for the pressure. In summary, we use the matching conditions

\[
Q_1 = u_1 - u_2, \quad (3.27a)
\]

\[
Q_2 = p_1 - p_2. \quad (3.27b)
\]
For the problem (3.25), our scheme needs to update interface values at each time step. On one hand, the velocity matching (3.25f) requires the assignment of the velocity obtained through the solution of the PF problem in $\Omega_2$ to the NSE in $\Omega_1$, as the boundary condition. On the other hand, the pressure as the result of solution of the NSE affects the boundary value of $\varphi$ on $\Gamma$ directly through the matching condition (3.25g) and Bernoulli’s principle (1.17)

$$\frac{\partial \varphi}{\partial t} = -\frac{|\nabla \varphi|^2}{2} - p - gz, \quad \text{on } \Gamma.$$  

(3.28)

This formulation can be considered as an ordinary differential equation (ODE) of $\varphi|_\Gamma$, when the right hand side becomes available. In this perspective, we can solve $\varphi|_\Gamma$ using any suitable time integration scheme, and use it as the Dirichlet boundary condition at $\Gamma$. In fact, this is the approach adopted by Iafrati and Campana [61] where a Runge–Kutta scheme is applied to solve above ODE.

To examine the nature of Bernoulli’s principle (1.17) and its effect on the matching conditions (3.27), let us apply them to the decomposition to the potential flow water wave problem. Namely, we decompose the flow domain $\Omega$ and solve the non–stationary potential flow water wave problem (3.14) in both $\Omega_1$ and $\Omega_2$, together with the matching conditions

$$Q_1 = \frac{\partial \varphi_1}{\partial n} - \frac{\partial \varphi_2}{\partial n},$$

(3.29a)

$$Q_2 = p_1 - p_2.$$  

(3.29b)

Note conditions (3.29) are the counterpart of (3.27) for our new problem. We assume $\partial \Omega = \Gamma_f \cup \Gamma_N$, and impose the homogeneous Neumann boundary condition on $\Gamma_N$. With unit density and combining (3.29) and Bernoulli’s principle (1.17), we have an iterative scheme to solve the non–stationary potential flow problem using DD method as follows.

**Scheme 3.3.1.** At time step $n + 1$, given $\lambda^k$ with $\lambda^0 = \varphi^p_\Gamma$, from iteration $k$ to $k + 1$, we do
1. Solve for $\varphi_{1}^{n+1,(k+1)}$ from
\[
\begin{cases}
\nabla^{2} \varphi_{1}^{n+1,(k+1)} = 0, & \forall x \in \Omega_{1}, \\
\frac{\partial \varphi_{1}^{n+1,(k+1)}}{\partial n} = 0, & \forall x \in \Gamma_{N}, \\
\varphi_{1}^{n+1,(k+1)} = \lambda^{k}, & \forall x \in \Gamma.
\end{cases}
\] (3.30)

2. Solve for $\varphi_{2}^{n+1,(k+1)}$ in
\[
\begin{cases}
\nabla^{2} \varphi_{2}^{n+1,(k+1)} = 0, & \forall x \in \Omega_{2}, \\
\frac{\partial \varphi_{2}^{n+1,(k+1)}}{\partial n} = 0, & \forall x \in \Gamma_{N}, \\
\frac{\partial \varphi_{2}^{n+1,(k+1)}}{\partial n} = \frac{\partial \varphi_{1}^{n+1,(k+1)}}{\partial n}, & \forall x \in \Gamma.
\end{cases}
\] (3.31)

3. Update $p^{n+1,(k+1)}$ and $\lambda^{k+1}$ in
\[
\begin{cases}
p^{n+1,(k+1)} = -\frac{1}{\Delta t}(\varphi_{2}^{n+1,(k+1)} - \lambda^{k}) - \frac{1}{2}|\nabla \varphi_{2}^{n+1,(k+1)}|^{2} - gz, & \forall x \in \Gamma. \\
\lambda^{k+1} = \lambda^{k} - \Delta t(p^{n+1,(k+1)} + \frac{1}{2}|\nabla \varphi_{2}^{n+1,(k+1)}|^{2} + gz), & \forall x \in \Gamma.
\end{cases}
\] (3.32)

**Remark 3.3.2.** Note that in (3.30) and (3.31) we have neglected the free surface boundary conditions (1.19) and (1.21). They must be imposed to close the system.

By eliminating $p^{n+1,(k+1)}$ in (3.32) one can see that the two equations are equivalent to
\[
\lambda^{k+1} = \varphi_{2}^{n+1,(k+1)}. \tag{3.33}
\]

Therefore Scheme 3.3.1 is nothing but the Dirichlet–Neumann (D–N) method [85, p. 11] for the non–stationary potential flow problem. In fact, from D–N method one would directly arrive at (3.33), without using (3.32). We insert (3.32) to illustrate the nature of Bernoulli’s principle (1.17) in our coupling scheme. Namely,
it “translates” the pressure into the potential that can be imposed as the Dirichlet data. During this translation, a proper time integration scheme must be applied. In (3.32), the backward Euler method is used.

In summary, we conclude: the matching conditions (3.29), combined with Bernoulli’s principle, are equivalent to the D–N method for the non–stationary potential flow problem, as a direct extension of the classical D–N method for the Laplace’s equation. Pressure \( p_i \) in (3.29) serves as the “translator” of the potential \( \varphi_i \), since in the heterogeneous flow model \( \varphi_1 \) is not readily available in the NSE subdomain.

**Example 3.3.3.** To further illustrate the behavior of the D–N method, let us applying it to a 1–D boundary value problem (BVP). As shown in Figure 3.1, we solve for \( y(x) \) in

\[
\frac{\partial^2 y}{\partial x^2} = 2 \cos x - x \sin x, \quad \text{on } [0, 13],
\]

with boundary data \( y|_{x=0} = 0 \) and \( y|_{x=13} = 0 \). Equation (3.34) has exact solution

\[
y(x) = x \sin(13) - x \sin(x).
\]

Here domain \( \Omega = (0, 13) \), and it is decomposed into \( \Omega_1 = (0, 8) \) and \( \Omega_2 = (8, 13) \). The iterations follow equations (3.30), (3.31) and the accelerated form of (3.33) as

\[
\lambda^{k+1} = \vartheta \lambda^k + (1 - \vartheta) \lambda^k,
\]

with \( \vartheta = 0.5 \) and \( \lambda^0 = 0 \). The exact solution is also plotted for comparison.

Now we are ready to apply the D–N like method to the problem (3.25). Let us first introduce two mappings. With boundary data \( \varphi \) on \( \Gamma \), we can solve the Laplace’s equation in \( \Omega_2 \), and compute \( u = \nabla \varphi \) on \( \Gamma \). We use the mapping \( \mathcal{M} \) to denote this “conversion” of the boundary data, i.e.,

\[
\mathcal{M}(\varphi_\Gamma) \triangleq u_\Gamma, \quad \text{such that}
\]

\[
u_\Gamma = \nabla \psi, \quad \text{where } \psi \text{ is the solution of } \begin{cases} 
\Delta \psi = 0, & \text{on } \Omega_2, \\
\psi = \varphi_\Gamma, & \text{on } \Gamma.
\end{cases}
\]
Figure 3.1: Using D–N method to solve BVP (3.34). The interface is located at $x = 8$. 
Operator $\mathcal{M}$ extends the Poincaré-Steklov operator [85, p. 3], in which only the normal derivative of $\varphi$ is obtained. Here both normal and tangential components of $\nabla \varphi$ are included.

Similarly, we define a mapping $\mathcal{N}$ on $\Gamma$ such that

$$\mathcal{N}(u_\Gamma) \triangleq p_\Gamma, \text{ on } \Gamma,$$ such that

$$\begin{cases} (v, p) \text{ solves the NSE on } \Omega_1, \\ v = u_\Gamma \text{ on } \Gamma. \end{cases}$$

Of course proper boundary conditions must be provided.

Introduce $\mathcal{M}$ and $\mathcal{N}$ in (3.28), we have

$$\frac{\partial \varphi}{\partial t} = -\frac{1}{2} |\mathcal{M}(\varphi)|^2 - \mathcal{N}[\mathcal{M}(\varphi)] - g z.$$ \hspace{0.5cm} (3.35)

Thus we have reduced the problem (3.25) with (3.27) to the interface. The reduction results in an ODE on $\Gamma$, and we can apply any suitable time integration scheme on it.

For example, let us we apply the forward Euler method to (3.35). On $\Gamma$ we solve for $\varphi^{n+1}$ in

$$\frac{\varphi^{n+1} - \varphi^n}{\Delta t} = -\frac{1}{2} |\mathcal{M}(\varphi^n)|^2 - \mathcal{N}[\mathcal{M}(\varphi^n)] - g z.$$ \hspace{0.5cm} (3.36)

This is equivalent to

$$\frac{\varphi^{n+1} - \varphi^n}{\Delta t} = -\frac{1}{2} |u^n|^2 - p^n - g z^n.$$ \hspace{0.5cm} (3.37)

Here $u^n$ and $p^n$ are from solving subproblems in each subdomain.

On the other hand, to improve numerical stability, we can also solve for $\varphi^{n+1}$ with the backward Euler method

$$\frac{\varphi^{n+1} - \varphi^n}{\Delta t} = -\frac{1}{2} |\mathcal{M}(\varphi^{n+1})|^2 - \mathcal{N}[\mathcal{M}(\varphi^{n+1})] - g z.$$ \hspace{0.5cm} (3.38)
This is equivalent to
\[
\frac{\varphi^{n+1} - \varphi^n}{\Delta t} = -\frac{1}{2}|u^{n+1}|^2 - p^{n+1} - gz. \tag{3.39}
\]

An iterative method could be used to solve (3.39). For instance, using a predictor–corrector method, we have the following scheme.

**Scheme 3.3.4.** From $\varphi^{n+1,(k-1)}$, solve for $\varphi^{n+1,(k)}$ by

1. Solve the PF problem in $\Omega_2$ to get $M(\varphi^{n+1,(k-1)})$.
2. Solve the NSE problem in $\Omega_1$ to get $N[M(\varphi^{n+1,(k-1)})]$.
3. On $\Gamma$, solve $\varphi^{n+1,(k)}$ in

   \[
   \varphi^{n+1,(k)} = \varphi^{n+1,(k-1)} + \lambda \left[ \varphi^n - \Delta t \left( \frac{1}{2}|M(\varphi^{n+1,(k-1)})|^2 + N[M(\varphi^{n+1,(k-1)})] + gz \right) \right].
   \]

Here $\lambda$ is the iteration parameter chosen to ensure the convergence.

In our sequential implementation, we use the forward Euler method (3.36). This results in the staggered time update shown in Figure 3.2 and 3.3. The scheme is as follows.

**Scheme 3.3.5.** From time step $n$ to $n+1$, with $\varphi^n|_\Gamma$ we do:

1. Solve the PF problem in $\Omega_2$ to get the flow velocity at $\Gamma$ as $u^n_\Gamma = \nabla \varphi^n$.
2. Solve the NSE problem in $\Omega_1$ to get $p^n|_\Gamma$.
3. Calculate $\varphi^{n+1}$ using (3.37) and $(u^n,p^n)|_\Gamma$.

**Remark 3.3.6.** Methods similar to Scheme 3.3.5 are implemented in [58] and [61].

**Remark 3.3.7.** In the above scheme, we have

\[
Q_1(u_1,p_1,\varphi) = u_1 - \nabla \varphi,
\]
\[
Q_2(u_1,p_1,\varphi) = \frac{\partial \varphi}{\partial t} + \frac{1}{2}|
abla \varphi|^2 + p_1 + gz.
\]
The matching condition (3.25g) for $Q_2$ then comes from Bernoulli’s equation (1.17) with $C(t) \equiv 0$.

Remark 3.3.8. As shown in Figure 3.3, the coupling between the two solvers is weak, i.e., there is no iteration within each time step to ensure match of the data at $\Gamma$. A predictor-corrector strong coupling can be derived using the time update plan above as the predictor step. Specifically, after the explicit update of $\varphi|_\Gamma$ to time $t + \Delta t$, $\varphi^k(t + \Delta t)$ and $(u^k(t + \Delta t), p^k_1(t + \Delta t))$ can be solved for in each subdomain, then a corrector step can be taken as

$$\varphi^*(t + \Delta t) - \varphi(t) = \frac{1}{\Delta t} + \frac{1}{2} |\nabla \varphi^k(t + \Delta t)|^2 + p^k_1 + gz = 0,$$

$$\varphi^{k+1}(t + \Delta t) = (1 - \lambda)\varphi^k(t + \Delta t) + \lambda \varphi^*(t + \Delta t).$$

The Relaxation parameter $\lambda$ can be stationary or non-stationary. Similar predictor-corrector coupling schemes are widely used for FSI simulations, see, e.g., [71]. Here the potential flow subdomain $\Omega_2$ plays the part of structure in FSI problems, and compatibility conditions are for velocity and stress (pressure), instead of velocity and displacement.

3.3.2 Algebraic formulation

In this section we examine the structure of the Schemes 3.3.4 and 3.3.5.

Since the FEM scheme (3.13) is based on a monolithic formulation, we can write the linear system of the approximated NSE similar to the saddle point problem.
Figure 3.3: An explicit scheme for the heterogeneous flow model.
formulation (2.13). Let $U$ and $P$ be the vectors for $u$ and $p$, respectively, after using certain time integration scheme, we have the linear system corresponding to the NSE (1.5) as

$$
\begin{bmatrix}
  F & D^T \\
  D & -C
\end{bmatrix}
\begin{bmatrix}
  U \\
  P
\end{bmatrix}
= \begin{bmatrix}
  f \\
  0
\end{bmatrix}.
$$

(3.40)

Recall that in general $F$ depends on $U$ because of the nonlinear advection term, i.e.,

$$
F = F(U),
$$

before the linearization is applied to the NSE.

As explained in §2.1.3, $C$ is not trivial since stabilization is needed.

Now we derive the system for the coupled problem. First, let us write equation (3.40) using the components from $\Omega_1$ and $\Gamma$. Let subscript 1 and $\Gamma$ denote the variables on $\Omega_1$ and $\Gamma$, respectively, we have

$$
\begin{bmatrix}
  F_{11} & F_{1\Gamma} & D_{11} & D_{1\Gamma} \\
  F_{\Gamma1} & F_{\Gamma\Gamma} & D_{\Gamma1} & D_{\Gamma\Gamma} \\
  D_{11} & D_{1\Gamma} & -C_{11} & -C_{1\Gamma} \\
  D_{\Gamma1} & D_{\Gamma\Gamma} & -C_{\Gamma1} & -C_{\Gamma\Gamma}
\end{bmatrix}
\begin{bmatrix}
  U_1 \\
  U_{\Gamma} \\
  P_1 \\
  P_{\Gamma}
\end{bmatrix}
= \begin{bmatrix}
  f_{1U} \\
  f_{\Gamma U} \\
  f_{1U} \\
  f_{\Gamma U}
\end{bmatrix}.
$$

(3.41)

Recall that $U_{\Gamma}$ will serve as the boundary data, therefore the momentum equation for the nodes on $\Gamma$ is redundant. Removing the momentum equation for the interface nodes, with a rearrangement we have

$$
\begin{bmatrix}
  F_{11} & D_{11}^T & D_{1\Gamma}^T & F_{1\Gamma} \\
  D_{11} & -C_{11} & -C_{1\Gamma} & D_{1\Gamma} \\
  D_{\Gamma1} & -C_{\Gamma1} & -C_{\Gamma\Gamma} & D_{\Gamma\Gamma}
\end{bmatrix}
\begin{bmatrix}
  U_1 \\
  P_1 \\
  P_{\Gamma}
\end{bmatrix}
= \begin{bmatrix}
  f_{1U} \\
  0 \\
  0
\end{bmatrix}.
$$

(3.41)

Now we want to connect $\Phi|_{\Gamma}$ to $P|_{\Gamma}$, with $\Phi$ being the vector for $\varphi$. Apply the
Euler method to (3.28), we have
\[
P_Γ = \frac{Φ_Γ - Φ_Γ^*}{Δt} - \frac{1}{2} E^T U_Γ - f_C. \tag{3.42}
\]

Here \( f_C = g Z \), with \( Z \) being the vector for the nodal coordinate in vertical direction. \( Φ_Γ^* \) is the potential on the interface from previous time step, and \( E^T \) is a diagonal matrix with entries \( E^T_{i,i} = U_{Γ,i} \).

Combing equations (3.41) and (3.42), we can replace \( P_Γ \) in the linear system (3.41) with \( Φ_Γ \), thus
\[
\begin{bmatrix}
F_{11} & D^T_{11} & \frac{1}{Δt} D^T_{Γ1} & F_{1Γ} - \frac{1}{2} D^T_{Γ1} E^Γ \\
D_{11} & -C_{11} & -\frac{1}{Δt} C_{Γ1} & D_{1Γ} + \frac{1}{2} C_{Γ1} E^Γ \\
D_{Γ1} & -C_{Γ1} & -\frac{1}{Δt} C_{ΓΓ} & D_{ΓΓ} + \frac{1}{2} C_{ΓΓ} E^Γ
\end{bmatrix}
\begin{bmatrix}
U_1 \\
P_1 \\
Φ_Γ \\
U_Γ
\end{bmatrix}
=
\begin{bmatrix}
f^U_1 \\
f^P_1 \\
f^P_Γ
\end{bmatrix}. \tag{3.43}
\]

Here
\[
\begin{bmatrix}
f^U_1 \\
f^P_1 \\
f^P_Γ
\end{bmatrix}
=
\begin{bmatrix}
f_{1U} \\
0 \\
0
\end{bmatrix} + (Φ_Γ^* + f_C)
\begin{bmatrix}
D^T_{Γ1} \\
-C_{ΓΓ}
\end{bmatrix}.
\]

Given boundary data \( U_Γ \), solving equation (3.43) for \( (U_1, P_1, Φ_Γ) \) amounts to solving the NSE in \( Ω_1 \) and use \( (U_Γ, P_Γ) \) to calculate \( Φ_Γ \). To simplify the notations, let us combine the nodal variables within \( Ω_1 \) as \( V \triangleq (U_1, P_1)^T \), and \( f_1 \triangleq (f^U_1, f^P_1)^T \), then rewrite (3.43) as
\[
\begin{bmatrix}
A_{11} & A_{1Γ} & A^V_{Γ1} \\
A^P_{Γ1} & A^P_{ΓΓ} & A^P_{ΓΓ}
\end{bmatrix}
\begin{bmatrix}
V \\
Φ_Γ \\
U_Γ
\end{bmatrix}
=
\begin{bmatrix}
f_1 \\
f^P_Γ
\end{bmatrix}. \tag{3.43*}
\]
Here

\[ A_{11} = \begin{bmatrix} F_{11} & DT_{11} \\ D_{11} & -C_{11} \end{bmatrix}, \quad A_{1\Phi} = \begin{bmatrix} \frac{1}{\Delta t}DT_{1\Phi} \\ -\frac{1}{\Delta t}C_{1\Phi} \end{bmatrix}, \quad A_{1\Gamma}^{\Gamma} = \begin{bmatrix} F_{1\Gamma} - \frac{1}{2}DT_{1\Gamma}E_{1\Gamma}^{\Gamma} \\ D_{1\Gamma} + \frac{1}{2}C_{1\Gamma}E_{1\Gamma}^{\Gamma} \end{bmatrix}, \]

\[ A_{P1} = \begin{bmatrix} D_{P1} & -C_{P1} \end{bmatrix}, \quad A_{P\Phi} = -\frac{1}{\Delta t}C_{P\Phi}, \quad A_{P\Gamma} = D_{P\Gamma} + \frac{1}{2}C_{P\Gamma}E_{P\Gamma}. \]

**Remark 3.3.9.** Instead of the backward Euler method, if the forward Euler method is used, modifications are only needed for the fourth column of the matrix in (3.43) and its right hand side, thus the corresponding definitions for the block matrices and right–hand–side vectors in (3.43*), but the scheme of (3.43*) remains the same.

On the other hand, recall that following (3.23) the problem in \( \Omega_2 \) can be formulated as

\[
\begin{bmatrix}
K^N & -J^N & K^{N*} \\
K^D & -J^D & K^{D*} \\
G_2 & G_{\Gamma} & -I_{\Gamma}^t
\end{bmatrix}
\begin{bmatrix}
\Phi \\
\Psi_{\Gamma} \\
\Phi_{\Gamma}
\end{bmatrix} = \begin{bmatrix}
J^{N*}\Psi^* \\
J^{D*}\Psi^* \\
0
\end{bmatrix}.
\]

Here we assume the boundary \( \partial\Omega_2 \setminus \Gamma \) is imposed with the Neumann boundary data \( \Psi^* \). Note that \( U_{\Gamma} \) has components \( U_{\Gamma,n} = \Psi_{\Gamma} \) and \( U_{\Gamma,t} \), with respect to the local normal and tangential direction at \( \Gamma \). We know \( u_{\Gamma,t} = \frac{\partial u}{\partial t} \), so we can formally describe the fact that \( U_{\Gamma,t} \) can be acquired from \( \Phi \) and \( \Phi_{\Gamma} \) as

\[ U_{\Gamma,t} = G_2\Phi + G_{\Gamma}\Phi_{\Gamma}. \]

Now the system (3.44) can be written as

\[
\begin{bmatrix}
K^N & K^{N*} & -J^N \\
K^D & K^{D*} & -J^D \\
G_2 & G_{\Gamma} & -I_{\Gamma}^t
\end{bmatrix}
\begin{bmatrix}
\Phi \\
\Phi_{\Gamma} \\
U_{\Gamma,n}
\end{bmatrix} = \begin{bmatrix}
J^{N*}\Psi^* \\
J^{D*}\Psi^* \\
0
\end{bmatrix}.
\]
Let $I^r_Γ$ be an identity matrix, and $R_n$ and $R_t$ be the restriction operators such that

$$R_nU_Γ = U_Γ, n, \quad R_tU_Γ = U_Γ, t.$$  

We can define the following matrices and vectors to simplify notations:

$$A_{22} = K^N, \quad A_{2Φ} = K^N^*,$$
$$A_{2U} = R_n^T K^D + R_t^T G_2, \quad A_{UΦ} = R_n^T K^D^* + R_t^T G_Γ,$$
$$A_{2U} = -J^N R_n, \quad A_{UU} = -R_n^T J^D R_n - R_t^T R_t,$$
$$f_2 = J^N^* Ψ^*, \quad f_2^U = R_n^T J^D^* Ψ^*.$$

Then (3.45) can be written as

$$\begin{bmatrix} A_{22} & A_{2Φ} & A_{2U} \\ A_{U2} & A_{UΦ} & A_{UU} \end{bmatrix} \begin{bmatrix} Φ \\ Φ_Γ \end{bmatrix} = \begin{bmatrix} f_2 \\ f_2^U \end{bmatrix}. \quad (3.45*)$$

Combining (3.43*) and (3.45*), we have the linear system for the DD formulation (3.38)

$$\begin{bmatrix} A_{22} & A_{2Φ} & A_{2U} \\ A_{U2} & A_{UΦ} & A_{UU} \\ A_{ΓΦ} & A_{ΓΦ} & A_{ΓU} \\ A_{1Φ} & A_{1Γ} & A_{11} \end{bmatrix} \begin{bmatrix} Φ \\ Φ_Γ \\ U_Γ \\ V \end{bmatrix} = \begin{bmatrix} f_2 \\ f_2^U \\ f_Γ \\ f_Γ \end{bmatrix}. \quad (3.46)$$

**Remark 3.3.10.** Formulation (3.46) has a format similar to that of a FSI problem; see, e.g., [63, eq. 4.53].

Let us simplify the notations in equation (3.46), thus

$$\begin{bmatrix} A_{11} & 0 & A_{1Γ} \\ 0 & A_{22} & A_{2Γ} \\ A_{Γ1} & A_{Γ2} & A_{ΓΓ} \end{bmatrix} \begin{bmatrix} V \\ Φ \\ V_Γ \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ f_Γ \end{bmatrix}. \quad (3.46*)$$
with

\[
V_\Gamma = \left( \Phi_\Gamma \ U_\Gamma \right)^T, \quad f_\Gamma = \left( f_U^\Gamma \ f_f^\Gamma \right)^T, \quad A_{1\Gamma} = \left( A_{1\Phi} \ A_{1\Gamma} \right), \quad A_{2\Gamma} = \left( A_{2\Phi} \ A_{2\Gamma} \right), \quad A_{1\Gamma} = \left( 0 \ A_{1\Gamma} \right)^T, \quad A_{2\Gamma} = \left( A_{2\Gamma} \ 0 \right)^T,
\]

and

\[
A_{1\Gamma} = A_{1\Gamma}^{(1)} + A_{1\Gamma}^{(2)}, \quad A_{1\Gamma}^{(1)} = \begin{bmatrix} 0 & 0 \\ A_{1\Phi}^P & A_{1\Gamma}^P \end{bmatrix}, \quad A_{1\Gamma}^{(2)} = \begin{bmatrix} A_{U\Phi} & A_{UU} \\ 0 & 0 \end{bmatrix}.
\] (3.47)

For equation (3.46*), we have its Schur complement system

\[
\Sigma_\Gamma V_\Gamma = \chi_\Gamma, \quad \Sigma_\Gamma = \Sigma_1 + \Sigma_2.
\] (3.48)

with

\[
\Sigma_i \triangleq A_{i\Gamma}^{(i)} - A_{i\Gamma} A_{ii}^{-1} A_{i\Gamma}, \quad i = 1, 2, \quad \chi_\Gamma \triangleq f_\Gamma - A_{1\Gamma} A_{11}^{-1} f_1 - A_{2\Gamma} A_{22}^{-1} f_2.
\]

Equation (3.46*), together with the splitting (3.47), can be solved using the schemes proposed in §3.3.

**Implicit schemes** First, let us consider the implicit schemes. Namely, in equation (3.42) we have \( \Phi_\Gamma = \Phi^n_\Gamma \), \( P_\Gamma = P^n_\Gamma \) and \( U_\Gamma = U^n_\Gamma \). In this case the term \( \frac{1}{2} E_U^\Gamma U_\Gamma \) in (3.42) can be linearized by assuming \( E_i^\Gamma = U_i^n \Gamma \). Thus in (3.46*) we have

\[
\left( V \ \Phi \ V_\Gamma \right)^T = \left( V^n \ \Phi^n \ V^n_\Gamma \right)^T, \quad V^n_\Gamma = \left( \Phi^n_\Gamma \ U^n_\Gamma \right)^T.
\]

In this setting, we have following two schemes. The difference between them is the way application of the splitting (3.47), and the manner in which the two components of \( V_\Gamma \) are updated during an iteration.

**Scheme 3.3.11.** Let \( V_\Gamma^{(k-1)} = \left( \Phi_{(k-1)}^\Gamma \ U_{(k-1)}^\Gamma \right)^T \) be from the previous iteration. From \( k - 1 \) to \( k \) do
1. Solve for the unknowns on the Neumann boundary in the PF problem

\[ A_{22} \Phi^{(k)} = f_2 - A_{2\Gamma} V^{(k-1)}_{\Gamma}. \]  

(3.49)

2. Solve the NSE problem

\[
\begin{bmatrix}
A_{11} & A_{1\Gamma} \\
A_{\Gamma1} & A_{\Gamma\Gamma}
\end{bmatrix}
\begin{bmatrix}
V^{(k)} \\
V^{(k-1/2)}_{\Gamma}
\end{bmatrix} =
\begin{bmatrix}
f_1 \\
f_{\Gamma} - A_{\Gamma2} \Phi^{(k)}
\end{bmatrix}.
\]  

(3.50)

3. Update using the relaxation

\[ V^{(k)}_{\Gamma} = \lambda V^{(k-1/2)}_{\Gamma} + (1 - \lambda) V^{(k-1)}_{\Gamma}, \]  

(3.51)

with a suitable relaxation parameter \( \lambda \).

Scheme 3.3.11 follows naturally the formulation (3.46*). Similar to the D–N method, at each iteration, \( V_{\Gamma} \) is updated in (3.50). However, unlike the D–N method, equation (3.50) does not use the splitting (3.47). In fact, by eliminating \( \Phi^{(k)} \) and \( V^{(k)} \), we can see that Scheme 3.3.11 is equivalent to a successive under–relaxation method applied to the Schur system (3.48), with preconditioner \( \Sigma + A_{2\Gamma}^{(2)} \).

Scheme 3.3.12. Let \( V^{(k-1)}_{\Gamma} \) be from the previous iteration. From \( k - 1 \) to \( k \) do

1. Solve for \( \Phi^{(k)} \) and \( \tilde{V}^{(k-1/2)}_{\Gamma} = \left( \Phi^{(k-1)}_{\Gamma} \hspace{1em} U^{(k)}_{\Gamma} \right)^T \) in the PF problem

\[
\begin{bmatrix}
A_{22} & A_{2\Gamma} \\
A_{\Gamma2} & A_{\Gamma\Gamma}^{(2)}
\end{bmatrix}
\begin{bmatrix}
\Phi^{(k)} \\
\tilde{V}^{(k-1/2)}_{\Gamma}
\end{bmatrix} =
\begin{bmatrix}
f_2 \\
f_{\Gamma}^{(2)}
\end{bmatrix}.
\]  

(3.52)

Here \( f_{\Gamma}^{(2)} = \left( f_{U}^{(2)} \hspace{1em} 0 \right)^T \).

2. Solve for \( V^{(k)} \) and \( \tilde{V}^{(k)}_{\Gamma} = \left( \tilde{\Phi}^{(k)}_{\Gamma} \hspace{1em} U^{(k)}_{\Gamma} \right)^T \) in the NSE problem

\[
\begin{bmatrix}
A_{11} & A_{1\Gamma} \\
A_{\Gamma1} & A_{\Gamma\Gamma}^{(1)}
\end{bmatrix}
\begin{bmatrix}
V^{(k)} \\
\tilde{V}^{(k)}_{\Gamma}
\end{bmatrix} =
\begin{bmatrix}
f_1 \\
f_{\Gamma}^{(1)}
\end{bmatrix}.
\]  

(3.53)
Here \( f^{(1)}_{\Gamma} = (0 \quad f^{U}_{\Gamma})^T \).

3. Update using the relaxation

\[
V^{(k)}_{\Gamma} = \lambda \tilde{V}^{(k)}_{\Gamma} + (1 - \lambda)V^{(k-1)}_{\Gamma},
\]

with a suitable relaxation parameter \( \lambda \).

Scheme 3.3.12 is the counterpart of Scheme 3.3.4. It can also be interpreted as a block Gauss–Seidel iteration method for equation (3.46*), with \( V_{\Gamma} \) as the overlapping component. Therefore, Scheme 3.3.12 is equivalent to the Schwarz alternating method, with \( V_{\Gamma} \) as the overlapping part between the two sub–problems (even though the subdomains do not overlap).

**Explicit scheme**  Following Scheme 3.3.11 and Scheme 3.3.12, we have similar explicit methods. Here we only introduce the counterpart of Scheme 3.3.12. The explicit version of Scheme 3.3.11 can be derived in the same way. The following scheme is the algebraic version of Scheme 3.3.5.

**Scheme 3.3.13.** Let \( \Phi^n_{\Gamma} \) be from the previous time step. From \( n \) to \( n + 1 \) do

1. Solve for \( \Phi^n_{\Gamma} \) and \( \tilde{V}^n_{\Gamma} = \left( \begin{array}{c} \Phi^n_{\Gamma} \\ U^n_{\Gamma} \end{array} \right) \) in the PF problem

\[
\begin{bmatrix}
A_{22} & A_{2\Gamma} \\
A_{\Gamma2} & A_{\Gamma\Gamma}^{(2)}
\end{bmatrix}
\begin{bmatrix}
\Phi^n \\
\tilde{V}^{n-1/2}_{\Gamma}
\end{bmatrix} =
\begin{bmatrix}
f_2 \\
\tilde{f}^{(2)}_{\Gamma}
\end{bmatrix}.
\]

2. Solve for \( V^n_{\Gamma} \) and \( \tilde{V}^{n+1}_{\Gamma} = \left( \begin{array}{c} \tilde{\Phi}^{n+1}_{\Gamma} \\ U^n_{\Gamma} \end{array} \right) \) in the NSE problem

\[
\begin{bmatrix}
A_{11} & A_{1\Gamma} \\
A_{\Gamma1} & A_{\Gamma\Gamma}^{(1)}
\end{bmatrix}
\begin{bmatrix}
V^n \\
\tilde{V}^{n+1}_{\Gamma}
\end{bmatrix} =
\begin{bmatrix}
f_1 \\
\tilde{f}^{(1)}_{\Gamma}
\end{bmatrix}.
\]

3. Update using the relaxation

\[
V^{n+1}_{\Gamma} = \lambda \tilde{V}^{n+1}_{\Gamma} + (1 - \lambda)V^n_{\Gamma},
\]
3.3.3 Matching of free surface

At the free surface $\Gamma_f$, the BEM and FEM results should match, i.e., $\eta_2 = \eta_1$. Here $\eta_2$ is acquired in BEM by updating the free surface using MEL method, whereas $\eta_1$ is based on the level set tracking technique. The latter is based on the solution of an advection equation near $\Gamma_f$, hence has the resolution of the FEM mesh. Therefore, the free surface matching at $\Gamma$ is of the same accuracy as that of the mesh size of $\Omega_1$, as shown in Figure 3.4.

![Figure 3.4: Meshed subdomains and the free surface near the interface.](image)

When an in-time iteration is used, $\eta_1$ and $\eta_2$ can be updated at every iteration. In our implementation of the staggered scheme, we must choose which side imposes the height of the free surface. Imposing $\eta_1$ on the boundary nodes in general introduces discontinuity of the level set function and numerical instability of the advection equation. Thus for the boundary element node at $\Gamma \cap \Gamma_f$, we impose

$$\eta_2^n = \eta_1^{n-1}. \tag{3.58}$$

For a general water wave problem, the simple matching (3.58) proves to be
insufficient. It introduces discontinuity of the surface slope at $\Gamma$, which in turn contaminates the wave profile. To preserve smoothness of the free surface, we use an interpolation function $\bar{\eta}$ to assign surface height values as

$$\eta^n_2(x) = \bar{\eta}^n(x), \quad x \in \Gamma_f \cap \Gamma.$$  \hfill (3.59)

Here $\bar{\eta}^n$ is the curve-fitting function based on the surface height of nearby nodes. In our implementation we fit $\bar{\eta}$ a quadratic function

$$\bar{\eta}^n(x) = ax^2 + bx + c.$$  

To obtain $a$, $b$ and $c$, nearby nodal values $x^n_1 \in \partial \Omega_2$, $x^{n-1}_2 \in \partial \Omega_1$ and $x^{n-1}_3 \in \partial \Omega_1$ are used, thus

$$\bar{\eta}^n(x^n_1) = y^n_1, \quad \bar{\eta}^n(x^{n-1}_2) = y^{n-1}_2, \quad \bar{\eta}^n(x^{n-1}_3) = y^{n-1}_3.$$  

**Smoothing through a moving average filter**  In our numerical tests, it was noticed that near the interface, spurious oscillations arise at the free surface $\Gamma_f$ of the PF subdomain. A simple moving average filter is applied to suppress these oscillations. Specifically, at the end of each update of the free surface position in the PF subdomain, for nodes near $\Gamma$ we do

$$\tilde{z}_i = \left( \sum_{j=1-N_{MA}}^{i+N_{MA}} z_j \right) / (2N_{MA} + 1), \quad i = 1, \ldots, N_F.$$  

Here $N_F$ is the number of nodes on which the smoothing is performed. For such a node, on one side of it there are $N_{MA}$ nodes are used to provide the averaging data. This low-pass filter allows us to preserve the free surface wave and remove the high frequency oscillations, while preserve the sharpest step response. In that sense, moving average filter is considered *optimal* [94, Chap. 15]. In our implementation, $N_{MA} = 3$, $N_F = 5$.

Among the numerical examples in Chapter 4, in the solitary wave example in 4.2 we will use the matching based on equation (3.59) and the moving-average filter.
In the dambreak example the simple nodal match (3.58) will be adopted.

3.3.4 Implementation

The scheme in Figure 3.3 is implemented on a workstation of Linux platform. The workstation is equipped with 12 Intel® Xeon® X5660 (2.80GHz) CPUs, and 190 GB RAM. Since both the BEM and FEM solvers are implemented in Fortran, the DD method routines are also in Fortran. For the data transfer the access to ICFD solution is through an Application Programming Interface (API) from LS-DYNA®.

The BEM solver NWT3D is intrinsically designed for 3D problems. In our 2D examples, at the end of each time step we ensure the boundary data are strictly in 2D. Then those data are used for the coupling.
4 Numerical examples

We present two numerical examples to demonstrate the implementation of the DD method described in the previous chapter. The first example is the initial stage of a dam–break wave. The second example shows a solitary wave traveling in a numerical wave tank.

4.1 Initial stage of dam break wave

In this first example we demonstrate the convergence of the DD method and its basic capability by looking at the initial stage of a dam–break wave. Here the reservoir is modeled using the potential flow model, and downstream is modeled by the NSE. Since the reservoir provides the potential energy that is converted to kinetic energy of wave traveling, in the absence of a large enough reservoir we can not capture the water elevation change correctly (see Figure 4.3). In this setting, the downstream flow domain belongs to $\Omega_1$, and the reservoir is occupied by $\Omega_2$ and part of $\Omega_1$, as shown in Figure 4.1 and table 4.1. We use the shallow water wave celerity $C = \sqrt{gd_2}$ as the characteristic velocity scale. In a nondimensional

<table>
<thead>
<tr>
<th>Reservoir water depth $d_2$</th>
<th>0.360 m</th>
</tr>
</thead>
<tbody>
<tr>
<td>Downstream water depth $d_1$</td>
<td>0.036 m</td>
</tr>
<tr>
<td>$l_1$</td>
<td>0.600 m</td>
</tr>
<tr>
<td>$l_2$</td>
<td>9.000 m</td>
</tr>
<tr>
<td>$l_3$</td>
<td>5.500 m</td>
</tr>
</tbody>
</table>

Table 4.1: Geometric configurations of the dam break test.
form, coordinates \((x, y)\) are normalized as

\[
X = \frac{x}{d_2}, \quad Y = \frac{y}{d_2},
\]

and the time \(t\) is normalized as

\[
T = \frac{t}{d_2/\sqrt{gd_2}},
\]

with \(T = 0\) indicating the beginning of the simulation, when the dam is removed. In this and the following examples, we adopt that \(X < 0\) is the region of \(\Omega_2\), whereas \(X > 0\) indicates subdomain \(\Omega_1\). Then \(X = 0\) is where the interface \(\Gamma\) lies.

Figure 4.2 shows snapshots of the wave front from \(T = 2\) to \(T = 12\). The level set free surface tracking successfully captures the complex water front.

The water height \(Y\) at \(X = 0\) decreases after dam is released. So let us first use \(Y(X = 0, T)\) as a physics indicator in the mesh convergence test.

Let \(n_{1,h}\) and \(n_{2,h}\) be the number of elements at the interface on the \(\Omega_1\) and \(\Omega_2\) side, respectively. In the convergence tests \(n_{2,h} = 6\), and \(n_{1,h}\) is increased from 8 to 128. In what follows the numeric subscript indicates corresponding \(n_{1,h}\). Two ICFD FEM simulations are also performed with the reservoir length equal to \(l_1 + l_2\) and \(l_1\), respectively. The first case is used for our convergence tests, and its results are denoted with subscript 0. The second FEM simulation is used to demonstrate the outcome when no mass flux at the interface is provided. The boundary condition
Figure 4.2: Dambreak wave front evolution as predicted by ICFD solver in DD method simulation at $T = 2, 4, 6, 8, 10, 12$ (top to bottom).
at $X = 0$ is instead a free-slip boundary condition in that case.

Figure 4.3 shows the time history of the interface water depth. A snapshot of the wave profile at $T = 5$ near $X = 0$ is also provided. One can see that the water depth from the DD simulation matches with FEM results closely. The match of the water depth time history and the free surface profiles indicates the conservation of mass across the interface.

![Graph showing the time history of the interface water depth.](image)

Figure 4.3: Surface elevation time history.

Figure 4.4 shows the convergence results of the water depth at $X = 0$. Both converge to $Y_0$, the ICFD FEM results, and to $Y_{128}$, the DD method solution, are shown.

As described in previous chapters, in our coupled problem $u$ can be considered
as the unknown in a formulation that is similar to using a Poincaré-Steklov-like operator. This motivates us to use $u_{\Gamma}$ to study convergence properties empirically. Similar to previous convergence results, we define two $L_2$ errors for two types of convergence. Let $u_{0,h}$ be the velocity at the interface from ICFD FEM solution, and $u_{n_{1,h}}$ be the velocity from DD simulations. We define

$$e_0 \triangleq \frac{\|u_{0,h} - u_{n_{1,h}}\|_2}{\|u_{0,h}\|_2}$$

and

$$e_1 \triangleq \frac{\|u_{128} - u_{n_{1,h}}\|_2}{\|u_{128}\|_2}.$$ 

The convergence results based on $e_0$ and $e_1$ are shown in Figure 4.5.

Both Figure 4.4 and Figure 4.5 indicate that the error of DD solutions is of $O(h)$, and the convergence rate is in general small. On one hand, we notice that the mesh refinement in the FEM subdomain improves the accuracy. This is consistent with that with the increased ratio $n_{1,h}/n_{2,h}$ the DD method models the wave front better, even though the element size of the BEM subdomain does not change. On the other hand, since the BEM mesh is not refined, the model accuracy between the potential
Figure 4.5: Error of interface velocities.
flow and the NSE persists in subdomain $\Omega_2$. Therefore the convergence rate to the FEM solution ($Y_0$ and $u_{0,h}$) is less than that to the refined DD model ($Y_{128}$ and $u_{128}$).

### 4.2 Solitary wave

In this example we apply the DD method to the phenomenon of a traveling solitary wave over a constant depth. The wave parameters are shown in table 4.2. The decomposed domain of the wave tank is shown in Figure 4.6.

![Figure 4.6: Solitary wave generation in a wave tank domain.](image)

**Table 4.2: Solitary wave parameters.**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wave height $H$</td>
<td>0.470 m</td>
</tr>
<tr>
<td>Water depth $d$</td>
<td>0.780 m</td>
</tr>
<tr>
<td>$l_1$</td>
<td>10.634 m</td>
</tr>
<tr>
<td>$l_2$</td>
<td>10.634 m</td>
</tr>
</tbody>
</table>

We apply above settings to a solitary wave given by the equation

$$
\eta(x, t) = H \cosh^{-2}(k(x - Ct)).
$$

We have

$$
k = \sqrt{\frac{3H}{4d^3}} = 0.591 \text{ m}^{-1},
$$

$$
C = \sqrt{g(d + H)} = 3.503 \text{ m}^{-1}\text{s}.
$$
The length of a solitary wave is theoretically infinite, but since the water elevation $\eta$ decays to zero reasonably fast with the increase of $x - Ct$, we can define the wave length $\lambda$ as

$$\lambda = \frac{2\pi}{k} = 10.634 \text{ m}.$$ 

At a distance $\lambda/2$ away from the wave crest, $\eta$ is reduced to 74% of its original value.

In this example, the computation domain is divided in half. For the right-running wave, the left half domain is $\Omega_2$, and the right half domain is $\Omega_1$. $x = 0$ denotes the location of our interface $\Gamma$. The wave is generated using NWT3D. The wave generation capability of the BEM solver is discussed in [53]. At the interface $\Gamma$, the BEM discretization gives 6 boundary elements, with each element size being approximately 0.100 m; whereas the FEM side has mesh size 0.013 m. At the initial state, the free surface boundary of $\Omega_2$ is uniformly meshed with element size 0.180 m, and the free surface in $\Omega_1$ has mesh size 0.060 m.

After the solitary wave is generated in subdomain $\Omega_2$, it runs toward $\Omega_1$, and the information is exchanged at $x = 0$. A series of snapshots are shown in Fig 4.7–4.9. In the plots, the $x$ and $y$ coordinates are normalized using the water depth $d$, and time $t$ is normalized using the derived wave period. Namely,

$$X = \frac{x}{d}, \quad Y = \frac{y}{d}, \quad T = \frac{t}{\lambda/C}.$$ 

A NWT3D BEM simulation for the whole domain $\Omega = \bar{\Omega}_1 \cup \bar{\Omega}_2$ was also run for comparison, as the results of NWT3D has been verified by a series of work [53, 54, 78, 98].

In the snapshots, one notices the general consistency of the results of DD method and NWT3D. In particular, at $\Gamma$ both the location and steepness are matched from the two solvers. It is obvious that the input of NWT3D is incorporated as the boundary condition for ICFD, hence the traveling the soliton. On the other hand, the feedback—the potential boundary condition updated by a pressure input—also
presents a trough and trailing oscillations, which is consistent with the NWT3D results. Those trailing trough and oscillations are predicted by the soliton theory for highly nonlinear waves like the one used here. A related phenomenon is the decrease of the wave height, as shown in the NWT3D results. Such a decay is consistent with the inverse scattering theory prediction, and is common for highly nonlinear waves that are generated by the wavemaker movement based on the Boussinesq (or KdV) equation. This is pointed out in many researches. For example, see the study by Grilli and Svendsen for numerical simulation of such a decay [98].

One can also note that, at the initial states, the wave height matches very well. As the soliton travels into $\Omega_2$, the wave crest is below the NWT3D results when the crest sits on the interface $\Gamma$ (Figure 4.8), indicating the increase of error. This error is soon mitigated by the increase of the wave height in DD solution. The difference of wave height in the later stage of the simulation (Figure 4.9), however, is due to the accumulated error of our explicit coupling scheme, after a long-time integration.

The efficacy of the DD method can also be examined by looking at the nodal values at the interface. Let nodes A and B be two nodes on the interface, i.e., $x_A \in \Gamma$, $x_B \in \Gamma$. In fact, we select them so that they are at 1/3 and 2/3 water depth, respectively, and their normalized positions are

$$X_A = X_B = 0, \quad Y_A = 0.33, \quad Y_B = 0.67.$$ 

Figure 4.10 and 4.11 show the (normalized) time history of the normalized velocities $U \triangleq \frac{u}{C}$, $V \triangleq \frac{v}{C}$ at A and B. Also shown are the corresponding values of two other nodes C and D. They are at the same vertical location as A and B, respectively, but in the middle of $\Omega_1$ in horizontal direction: $X_C = X_D = 4.00$. Velocities in different location are shown to be consistent by having same history profile. One can also notice a slight increase of the velocity magnitude, contributing to the non-decaying wave height at the later stage of the simulation, as shown in Figure 4.9. On the other hand,
Figure 4.7: Solitary wave profile at $T=0.80$—$T=1.125$
Figure 4.8: Solitary wave profile at $T=1.190$—$T=1.515$
Figure 4.9: Solitary wave profile at $T=1.580$—$T=1.915$
Figure 4.12 shows $P$ as the value normalized by corresponding hydrostatic pressure, namely,

$$P(z) \triangleq \frac{p(z)}{\rho g z},$$

and nodal values at both depths show oscillations in time. We believe those oscillations are the results of the weak coupling. First, the velocity input at $\Gamma_f$ to $\Omega_1$ is inaccurate due to interpolation. Such inaccuracy is equivalent to imposing a perturbed value of the proper boundary data. This further induces oscillations in pressure. The other cause is the error of free surface elevation. Since free surface is imposed to $\Omega_2$ using the value from $\Omega_1$, there is no immediate regulation of the free surface on $\Omega_1$ side but only the mass and momentum conservation at $\Gamma$. This error is manifested as hydrostatic pressure. Both problems can be solved with a strong coupling scheme with techniques developed for non-conforming meshes, such as mortar methods. In that case, mass conservation is achieved in weak sense, and two-way free surface matching can mitigate the interface oscillations.

![Figure 4.10: Time history of nodal velocity $u$.](image)

Figure 4.10: Time history of nodal velocity $u$. 
Figure 4.11: Time history of nodal velocity $v$.

Figure 4.12: Time history of normalized pressure.
5 Conclusions

In this study, a non–overlapping domain decomposition (DD) method was adopted to solve a heterogeneous flow model. In this model, a viscous flow solver and a potential flow solver are coupled in a non-overlapping decomposition framework, and the velocity and pressure are used to define proper boundary conditions. The matching conditions at the subdomain interface are supplemented by the time integral from Bernoulli’s equation in order to translate the pressure from the solution of the Naver–Stokes Equation (NSE) into the potential, and to provide the boundary condition to the potential flow model. This is needed the NSE do not contain explicitly the potential as the variable. The velocity calculated from the potential flow model is imposed at the interface as the boundary condition for the NSE. At the interface, this mapping from the potential data to the velocity data is an extension of the Poincaré-Steklov operator. Therefore the coupling scheme is similar to the Dirichlet–Neumann (D–N) method.

In the discrete form, the solution strategy depends on the time integral method of Bernoulli’s equation at the interface. When an implicit scheme is used, the scheme follows exactly the D–N method, and the iterations within each time step can be interpreted as preconditioned Richardson iterations of the Schur complement system of the original problem. The preconditioner is the Schur complement of the sub–problem from the subdomain. On the other hand, in an explicit scheme, the D–N scheme is performed once per time step, equivalent to the staggered scheme for the monolithic system.

A sequential explicit scheme for the proposed DD method is implemented, based on two numerical solvers: NWT3D, an boundary element method (BEM) solver based on the potential flow model, and the ICFD LS–DYNA®, a finite element
method (FEM) solver for incompressible viscous flows. Both solvers contain free surface tracking capability.

Two numerical examples are presented for model validation. The first one models the initial stage of a dam–break wave. Convergence results are shown for the mesh refinement at the interface. The second example demonstrates the travel of a highly nonlinear solitary wave.

The future work includes:

- Extend the implementation to 3D.
- Improve the accuracy and stability of the method by adopting higher order and implicit time integration schemes.
- Parallelize the implementation to take advantage of the DD method.
- Study the theoretical accuracy and convergence of the method.
Bibliography


between stabilized finite element methods and the Galerkin method with

[12] A.N. Brooks and T.J.R. Hughes. Streamline upwind/Petrov-Galerkin for-
mulations for convection dominated flows with particular emphasis on the
incompressible Navier-Stokes equations. *Computer Methods in Applied Me-

481, May 1990.


approach for the computation of viscous transonic flows by unsteady type

coupling in free surface ship flows. *International Journal for Numerical

methods for discrete newton algorithms. *SIAM Journal on Scientific and

[18] H.-C. Chen and S.-K. Lee. RANS/Laplace calculations of nonlinear waves

[19] A.J. Chorin. The numerical solution of the Navier-Stokes equations for an
incompressible fluid. *Bulletin (New Series) of the American Mathematical


chanics*, volume 4 of *Texts in Applied Mathematics*. Springer, 3rd edition,
May 1993.


