Two-Phase Compressible-Incompressible Flow Problems: Numerical Approximation and Optimal Control

Doctoral Dissertation of:
Mattia Tamellini

Advisors:
Prof. Nicola Parolini
Prof. Marco Verani

Coordinator of the Doctoral Program:
Prof. Irene Sabadini

XXX cycle
In this house we obey the laws of thermodynamics!

Homer Simpson
Acknowledgements

I would like to express my heartfelt gratitude towards my advisors, Prof. Nicola Parolini and Prof. Marco Verani, for being incredibly helpful, open and friendly, and for the many suggestions and long discussions.

I would also like to thank Prof. Elie Hachem and all the people at CEMEF - MinesParistech for welcoming me with open arms and helping me with any problem (scientific or otherwise) I faced during my stay.

Thanks to my friends, who kept my spirits up when I was feeling down. Nicholas, Stefano, Francesco, Alessandro, Desirée, Marco, Matteo, Giacomo, Carlo. Here’s to you.

Thanks to my family, for believing in me.
Abstract

The main topic of this thesis is two-phase compressible-incompressible flows and, in particular, whether it is possible to guide such flows towards a prescribed solution. To this end, two mathematical models, already established in the literature, for two-phase compressible-incompressible fluids are introduced and compared to one another in order to select the one that best fits the needs of this work, using both theoretical analysis and numerical results. Once this is done, an optimal control problem on the chosen model is derived, and the controllability of the system as a whole is discussed, again backing up the claims with numerical simulations to show the possible results that can be obtained.
# Contents

Acknowledgements v

List of Figures xiii

Introduction xv

1 Mathematical models for two-phase flows 1
   1.1 General model ............................................. 1
   1.2 A two-phase incompressible-incompressible model ............ 5
      1.2.1 Interface capturing technique ............................ 5
      1.2.2 The continuous model .................................... 7
      1.2.3 Space-time discretization ............................... 10
      1.2.4 Numerical results ....................................... 13
         1.2.4.1 Benchmark 1 ......................................... 16
         1.2.4.2 Benchmark 2 ......................................... 16
   1.3 Compressible-incompressible models .......................... 18
      1.3.1 Initial motivation ....................................... 20
      1.3.2 A single-pressure method ............................... 21
         1.3.2.1 The continuous model .............................. 21
         1.3.2.2 Space-time discretization .......................... 24
      1.3.3 A split-pressure method ............................... 26
         1.3.3.1 The continuous model .............................. 26
         1.3.3.2 Space-time discretization .......................... 29
      1.3.4 Numerical results ....................................... 33
         1.3.4.1 Test case 1: oscillating water column ............... 33
         1.3.4.2 Test case 2: non-isothermal water column .......... 34
         1.3.4.3 Test case 3: a 2D isothermal compression ........... 36
      1.3.5 Final comparison of the two models ...................... 42
      1.3.6 A compressible rising bubble ............................ 43
2 Abstract framework for optimal control problems 49
  2.1 Introductory definitions 49
    2.1.1 Basics of differential calculus in Banach spaces 49
    2.1.2 Optimization in Banach spaces 51
  2.2 General optimal control problems 52
    2.2.1 Structure of a control problem 52
    2.2.2 Lagrangian formulation of optimal control problems 53
    2.2.3 Solution of the control problem 55
      2.2.3.1 Minimization algorithm: the line search method 56
    2.2.4 A comment on the two possible discretization approaches 59

3 Optimal control for compressible-incompressible flows 61
  3.1 An introductory control problem 61
    3.1.1 Formulation of the control problem 61
    3.1.2 The optimality conditions 62
      3.1.2.1 The descent direction 64
    3.1.3 Space-time discretization and minimization algorithm 65
    3.1.4 Numerical results 66
  3.2 The optimal control problem of interest 68
    3.2.1 A compressible-incompressible control problem 70
      3.2.1.1 Compressible-incompressible optimality conditions 72
    3.2.2 The incompressible-incompressible case 75
      3.2.2.1 Incompressible-incompressible optimality conditions 76
    3.2.3 Derivation of the adjoint system for compressible-incompressible
    flows 77
      3.2.3.1 Derivatives of the Lagrangian functional 77
      3.2.3.2 The surface tension term 81
      3.2.3.3 The adjoint problem 85
    3.2.4 Space-time discretization and minimization algorithm 87

4 Numerical results of the optimal control problems 91
  4.1 Incompressible-incompressible results 91
    4.1.1 Test case 1: constant target control reconstruction 93
    4.1.2 Test case 2: reconstruction of a given non-constant control 95
  4.2 Compressible-incompressible results 98
    4.2.1 Test case 1: constant target control reconstruction 98
    4.2.2 Test case 2: Neumann boundary 100
    4.2.3 Test case 3: controlling the final height 104
    4.2.4 Test case 4: convergence analysis 107
    4.2.5 Test case 5: controlling the topology 107
## CONTENTS

5 A C++ library for optimal control problems  
  5.1 Main library classes  
  5.1.1 The problems hierarchy  
  5.1.2 The objective functional hierarchy  
  5.1.3 The optimizer hierarchy  

Conclusions  

Bibliography
**List of Figures**

1.1 Initial configuration for the incompressible-incompressible benchmarks 14
1.2 Bubble’s position for benchmark 1 .......................... 16
1.3 Benchmark quantities for benchmark 1 .......................... 17
1.4 Bubble’s position for benchmark 2 .......................... 18
1.5 Benchmark quantities for benchmark 2 .......................... 19
1.6 Initial configuration for the first compressible-incompressible test case 34
1.7 First compressible-incompressible test case: pressures coefficients at the domain boundary .......................... 35
1.8 Initial configuration for the second compressible-incompressible test case 35
1.9 Second compressible-incompressible test case: pressures at the domain boundary .......................... 37
1.10 Initial configuration for the third compressible-incompressible test case 38
1.11 Third compressible-incompressible test case: time convergence of the bubble’s position .......................... 39
1.12 Third compressible-incompressible test case: time convergence of the bubble’s volume .......................... 40
1.13 Third compressible-incompressible test case: comparison of the bubble’s positions .......................... 40
1.14 Third compressible-incompressible test case: comparison of the bubble’s volumes .......................... 41
1.15 Initial configuration for the compressible rising bubble 44
1.16 Bubble’s position for the compressible rising bubble 46
1.17 Benchmark quantities for the compressible rising bubble 47

3.1 Initial configuration and target solution for the introductory control problem .......................... 67
3.2 Introductory optimal control problem, history of convergence .......................... 68
3.3 Introductory optimal control problem, optimal value of the control function $C$ .......................... 69
3.4 Introductory optimal control problem results .......................... 69
3.5 Example configurations for the optimal control problem 3.2 .......................... 70
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Initial configuration for the control problem in the incompressible-incompressible case</td>
<td>92</td>
</tr>
<tr>
<td>4.2</td>
<td>Incompressible-incompressible test case 1, history of convergence</td>
<td>94</td>
</tr>
<tr>
<td>4.3</td>
<td>Incompressible-incompressible test case 1, optimal control problem results</td>
<td>94</td>
</tr>
<tr>
<td>4.4</td>
<td>Incompressible-incompressible test case 2, history of convergence</td>
<td>96</td>
</tr>
<tr>
<td>4.5</td>
<td>Incompressible-incompressible test case 2, optimal control problem results</td>
<td>97</td>
</tr>
<tr>
<td>4.6</td>
<td>Compressible-incompressible test case 1, initial position of the bubbles</td>
<td>99</td>
</tr>
<tr>
<td>4.7</td>
<td>Compressible-incompressible test case 1, history of convergence</td>
<td>100</td>
</tr>
<tr>
<td>4.8</td>
<td>Compressible-incompressible test case 1, optimal control problem results</td>
<td>100</td>
</tr>
<tr>
<td>4.9</td>
<td>Compressible-incompressible test case 2, initial position of the bubble</td>
<td>101</td>
</tr>
<tr>
<td>4.10</td>
<td>Compressible-incompressible test case 2, history of convergence</td>
<td>102</td>
</tr>
<tr>
<td>4.11</td>
<td>Compressible-incompressible test case 2, optimal control problem results</td>
<td>103</td>
</tr>
<tr>
<td>4.12</td>
<td>Compressible-incompressible test case 2, surface of the bubble under the action of the optimal control</td>
<td>103</td>
</tr>
<tr>
<td>4.13</td>
<td>Compressible-incompressible test case 3, initial position of the bubbles</td>
<td>105</td>
</tr>
<tr>
<td>4.14</td>
<td>Compressible-incompressible test case 3, history of convergence</td>
<td>105</td>
</tr>
<tr>
<td>4.15</td>
<td>Compressible-incompressible test case 3, optimal control problem results</td>
<td>106</td>
</tr>
<tr>
<td>4.16</td>
<td>Compressible-incompressible test case 4, evolution of the controlled solution at different time instants and with different refinement levels</td>
<td>107</td>
</tr>
<tr>
<td>4.17</td>
<td>Compressible-incompressible test case 5, initial position of the bubbles</td>
<td>108</td>
</tr>
<tr>
<td>4.18</td>
<td>Compressible-incompressible test case 5, history of convergence</td>
<td>109</td>
</tr>
<tr>
<td>4.19</td>
<td>Compressible-incompressible test case 5, optimal control problem results</td>
<td>110</td>
</tr>
<tr>
<td>5.1</td>
<td>UML diagram for the problems hierarchy of classes in the dcp library</td>
<td>114</td>
</tr>
<tr>
<td>5.2</td>
<td>UML diagram for the objective functional hierarchy of classes in the dcp library</td>
<td>116</td>
</tr>
<tr>
<td>5.3</td>
<td>UML diagram for the optimizer hierarchy of classes in the dcp library</td>
<td>118</td>
</tr>
</tbody>
</table>
Introduction

This work spawns from the industrial interest in studying the production process of a particular kind of gas-solid structures known as metal foams. These are metallic cellular structures which consist of gas cavities trapped within a metal framework, whose industrial relevance has been growing more and more in the past years, thanks to their particularly interesting properties of energy absorption, stiffness to weight ratio, vibration dampening and thermal conductivity, which make them suitable for many applications.

Several techniques for the production of metal foams have been proposed. We limit ourselves to briefly presenting those more relevant to this work and we refer to [Ban01], [RRV17], [Rep] and [KTS02] for further details on the subject.

The available production techniques can be divided into three main branches, depending on the state of matter the metal is in during the foaming process.

The first set of methods, which includes those we will consider in the rest of this work, is based on the use of liquid metal. Five different possibilities can be detailed within this framework:

1. **direct foaming**, which consists in adding gas bubbles to pre-molten metal, either by direct gas injection or by admixing with the liquefied metal a gas-releasing blowing agent, which yields gas bubbles under the action of heat;

2. **indirect foaming**, which consists in exploiting the ability of some liquid metals to combine with specific gases and undergo a eutectic transition that turns a homogeneous melt into a heterogeneous two-phase solid-gas system under the right physical conditions;

3. mixing metal powder with gas-releasing blowing agents and then have the mix melt by compression and heating, which, once again, activate the gas-releasing power of the blowing agents;

4. casting liquid metal in (usually polymeric) molds;

5. spraying atomized molten metal onto a substrate.

Other approaches, based on the use of metal in non-liquid state, include the generation of foams from solid powdered metal (through e.g. sintering, gas entrapment, slurries
INTRODUCTION

and molds), from a solution of metal ions in an electrolyte (through deposition onto polymeric foams) and even from gaseous or vaporized metal.

Methods 1 and 3 are the only ones, among the listed approaches, that allow the bubbles to move within the liquid matrix and change their size until the solidification phase starts, and, as such, they are the methods on which this work is based. As a matter of fact, our interest lies specifically in the investigation of the liquid-gas mixture that is part of the foaming process, and it extends no further than this particular production phase. In this sense, we are not interested in modelling the way the gas bubbles are generated, nor do we investigate the solidification process. Our goal is, instead, to control and optimize the position and size of the bubbles within the liquid phase at the final moment, just as solidification is about to begin, since these quantities determine the physical properties of the foam, once solidified.

In order to describe the motion of gas bubbles in a liquid domain a multi-phase flow model is needed. Many different approaches exist in the literature on the subject, each with its strengths and weaknesses.

On the one hand, a first issue to tackle when describing multi-phase flows is how to keep track of the position of the different fluids and of the interface between them. The most common techniques to address this matter are the Level Set Method ([OD13], [SS03]), the Volume Of Fluid Method ([HN81]) and the Ghost Fluid Method ([Fed+99]). Each of these approaches has been used in the literature of two-phase flows.

On the other hand, one must decide what the best model to describe each phase is. As a matter of fact, among the works that deal with immiscible multi-phase fluids flows, one can distinguish three main themes: incompressible-incompressible flows, compressible-compressible flows and compressible-incompressible flows. Each must then be combined with the interface tracking approach that best fits one’s needs.

Incompressible-incompressible flows are the easiest from the point of view of the mathematical models involved: the Navier-Stokes equation are valid in both fluids, and one only has to find a proper way to determine the values of the physical parameters (such as density, viscosity and maybe thermal conductivity, if an energy equation is considered) in the two phases, which is not always a trivial task, since high differences between the values of the physical parameters in the two phases may give rise to serious numerical issues. This is the case, for example, of [BM07] and [THB04], where the Level Set Method and the Volume of Fluid method (and even a combination of the two) are used to address the problem of incompressible-incompressible flows with high density ratios among the two phases. Similar numerical issues may arise for the computation, for example, of viscous terms at the interface, an issue that is dealt with in [Ste+08]. Other papers focus on the best way to track the interface: [ST10] and [Sus+07] propose a coupled Level Set - Volume Of Fluid method that takes the best of both worlds while trying to overcome the weaknesses of each of the two
approaches, while in [TB04] the Volume Of Fluid approach is combined with a mesh
refinement method to ensure sharpness of the interface. A zero-thickness interface
is proposed also in [Smo05], where Finite Elements are used in order to allow for
a sharp interface in a Level Set method framework. Finally, in [SYW13], a quasi-
incompressible model is used for both incompressible phases to ensure better volume
and mass conservation, while [Yee10] uses X-FEM to closer approximate the jumps
that lie in the elements along the interface.

In compressible-compressible flows, the same model can again be used for both
fluids, but the typical issues of compressibility (shocks and rarefaction, for example)
may arise as well. Some works that deal with this kind of problems are, just to name
a few, [AS03], [Kor+02], where high density ratios are considered and a coupled Level
Set - Ghost Fluid Method is employed for the interface tracking, and [Shy99].

The main problem that is specific only to compressible-incompressible flows is
the need for different models for the two fluids. Indeed, as already discussed above,
while an incompressible fluid behaves according to the incompressible Navier-Stokes
equations, a compressible fluid is modeled using a compressible model. Of course the
former is nothing but a simplified case of the latter, but it is exactly the simplifications
involved that make so that the Navier-Stokes equation can in no form be used for a
compressible flow. This means that, in principle, different sets of equations should be
solved in different subsets of the domain, and, moreover, these very subsets are part
of the set of unknowns of the problem. Some works ([APF13], [MPR99], [Cab05],
[Cab06], [CPR05] and [CCR12]) have proposed approximated approaches, meaning
that flow equations (and in particular, the Navier-Stokes equations) are solved in the
liquid phase only, while an equation of state is used to compute the state variables
in the gas, where, on the other hand, the flow is not solved. Other works, for exam-
ple [CFA01] and [LSR16], opt for the solution of the two different sets of equations in
the two phases, using interface boundary conditions as one would do in fluid-structure
interaction problems. This, of course, brings along all the issues of fluid-structure
interaction, and in particular the need to track (and mesh) a moving domain. The
approach we are interested in uses instead a unified formulation, in which the same set
of variables and equations is valid throughout the whole domain. This is achieved by
using some kind of approximation to make the compressible model in the gas phase be
as close as possible to the incompressible Navier-Stokes equations, without sacrificing
accuracy: in [BGN11] the weak-compressibility assumption is supposed to hold for the
gas while in [Dar+10] the Low-Mach approximation ([Pao82] and [Ala05]) is used.

This last class of models is of particular interest for those kinds of applications in
which the change of volume of the gaseous component of the multi-phase flow plays
a crucial role. This is the case, for instance, of the production process of metal foams
with the liquefied-metal approach discussed above.

Our goal in this thesis is to present a possible algorithm to solve a generic optimal
control problem in a two-phase compressible-incompressible framework. The selection of a suitable multi-phase model is then critical since it should combine the ability to capture the evolution of compressible bubbles in an incompressible fluid while keeping the overall formulation simple enough to be able to formulate and numerically solve an optimal control problem. The solution of optimal control problems for compressible-incompressible flows is, to the best of our knowledge, novel in the literature. In general, optimization in the framework of partial differential equations has been investigated for a long time in its various forms, may it be for example inverse problems (see e.g. [CER90]) or optimal control problems (see e.g. [Lio71] and [Trö10]). More specifically, numerical optimization in the field of computational fluid dynamics has been the object of intensive study, with different goals (e.g. drag reduction, energy minimization, heat dissipation and vorticity reduction). It is beyond the scope of this work to provide an exhaustive review on the topic. We limit ourselves to recalling, among others, the book [Gun03] and the references therein. In view of the discussion above, the goal of this work is to provide a first contribution towards the understanding of optimal control problems governed by compressible-incompressible flows. Here we will only consider control problems on isothermal models, for the sake of simplicity. However, notice that, in the production process of metal foams, the expansion phase happens at a temperature than can be considered constant, at least on a first order approximation, so our working hypothesis of discarding the temperature in the definition of the optimal control problem is justified.

The outline of this thesis is as follows.

In chapter 1 the two-phase models used in this work are presented. We begin by introducing the compressible Navier-Stokes equations, which describe a generic fluid flow, and we expand upon them until we reach the final models we will investigate in this work: after discussing a two-phase incompressible model, where the common elements for two-phase unified models are introduced without all the hassle related to compressibility, we move on to the compressible-incompressible models developed in [Dar+10] and [BGN11]. Both models present strengths and weaknesses, which will be discussed from a theoretical point of view as well as with numerical simulations. For all the models, space-time discretization is discussed.

In chapter 2 optimal control problems are introduced. The reader is provided with a brief overview of the subject in general, in order to present all the theoretical tools needed.

In chapter 3 we then delve into the specifics of a possible control problem in which we aim to drive the gas bubbles in a liquid domain towards a prescribed position. The problem is first introduced using a two-phase incompressible model to focus in on the specific challenges we have to face in the optimal control problem, and it is then generalized to a (more complex) compressible model once such difficulties have been discussed and overcome.
Chapter 4 presents some results obtained with the optimal control problem defined in chapter 3, which show the good controllability of the system.

In chapter 5 the library developed for this work, which was used for all the numerical experiments shown, is discussed.

Finally, in the last chapter, we draw some conclusions on what has been done, highlighting the most interesting results and the most engaging open questions.

The results obtained in this thesis have been published in [TPV18].
INTRODUCTION


Chapter 1

Mathematical models for two-phase flows

In this chapter we introduce the mathematical models used in this work to tackle two-phase flows, both for cases in which both fluids are incompressible and for cases where one of the two fluids is compressible. In particular, two different models will be investigated for the latter case: the first one is taken from [Dar+10], while the second comes from [BGN11]. In section 1.3 we will present these two models, highlighting similarities and differences and comparing the results they yield on a series of test cases. Notice that, while the physical models are taken directly from [Dar+10] and [BGN11], the space-time discretization proposed here for such models does not follow what was presented in the original works, but it is specific to this thesis.

1.1 General model

Before we dive into the specifics of each model, let us introduce some definitions that will be used throughout this chapter and present a general flow model which will be adopted as a starting point for all the models in the next sections.

Let $\Omega \subset \mathbb{R}^d$, $d = 1, 2, 3$, be a sufficiently regular bounded closed set and $[t_0, t_f]$ be the time interval in which the flow problem is to be solved. From here on after, $\mathbf{u} = \mathbf{u}(x, t)$ will denote the velocity, $p = p(x, t)$ the pressure $T = T(x, t)$ the temperature, $\rho = \rho(x, t)$ the density, $\sigma = \sigma(x, t)$ the Cauchy stress tensor, $\tau = \tau(x, t)$ the viscous stress tensor, $e_i = e_i(x, t)$ the specific internal energy, $\mu = \mu(x, t)$ the viscosity, $\lambda = \lambda(x, t)$ the second viscosity coefficient (which equals to $-2/3\mu$, unless stated otherwise), $k = k(x, t)$ the thermal conductivity, $c_p = c_p(x, t)$ the isobaric specific heat, $c_v = c_v(x, t)$ the isochoric specific heat, $r$ the specific gas constant, $\varsigma$ the surface tension coefficient, $\kappa = \kappa(x, t)$ the curvature, $s = s(x, t)$ the surface tension and $g$ the gravitational acceleration.

We start by considering the compressible Navier-Stokes equations in conservative...
form. Assuming that gravity is the only external forcing term and that the fluid is \textit{divariant} (i.e. its state can be fully described by two independent variables), the system reads:

\[
\begin{align*}
\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) &= 0, \\
\frac{\partial}{\partial t} (\rho \mathbf{u}) + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u} - \mathbf{\sigma}) &= \rho \mathbf{g}, \\
\frac{\partial}{\partial t} (\rho e) + \nabla \cdot (\rho e \mathbf{u} - \mathbf{u} \cdot \mathbf{\sigma} - k \nabla T) &= \rho \mathbf{u} \cdot \mathbf{g}, \\
p &= p(\rho, e), \\
T &= T(\rho, e),
\end{align*}
\]  

(1.1)

where \(\otimes\) denotes the outer product and \(e\) is the specific energy, given by the sum of the specific internal energy \(e_i\) and the specific kinetic energy \(e_k\):

\[e = e_i + e_k = e_i + \frac{1}{2} \|\mathbf{u}\|^2.\]

We refer to [Tor09] for further details on the derivation of such system of equations.

In this work, fluids are all supposed to be Newtonian, and gases are supposed to be perfect. This leads to the following constitutive equations:

\[
\begin{align*}
\mathbf{\sigma} &= -p \mathbf{I} + \mathbf{\tau}, \\
\mathbf{\tau} &= \lambda (\nabla \cdot \mathbf{u}) \mathbf{I} + 2\mu \mathbf{D}, \\
\mathbf{D} &= \frac{\nabla \mathbf{u} + (\nabla \mathbf{u})^t}{2}, \\
p &= \rho r T, \\
e_i &= c_v T.
\end{align*}
\]  

(1.2)

Using (1.2), system (1.1) can be restated as:

\[
\begin{align*}
\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) &= 0, \\
\frac{\partial}{\partial t} (\rho \mathbf{u}) + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u} + p \mathbf{I} - \mathbf{\tau}) &= \rho \mathbf{g}, \\
\frac{\partial}{\partial t} (\rho e) + \nabla \cdot (\rho e \mathbf{u} - \mathbf{u} \cdot \mathbf{\sigma} - k \nabla T) &= \rho \mathbf{u} \cdot \mathbf{g}, \\
e &= e_i + \frac{1}{2} \|\mathbf{u}\|^2, \\
p &= \rho r T, \\
e_i &= c_v T.
\end{align*}
\]  

(1.3)

Our goal is now to rewrite these equations in a more useful and familiar form and to simplify them by substituting the specific energy \(e\) out of the system.
1.1. GENERAL MODEL

First of all, let us perform some linear algebra on the momentum equation in system (1.3). Since:

$$\nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) = \nabla \cdot (\rho \mathbf{u}) \cdot \mathbf{u} + \rho \mathbf{u} \cdot \nabla \mathbf{u},$$

expanding the time derivative in the momentum equation we can write:

$$\frac{\partial}{\partial t} (\rho \mathbf{u}) + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) = \mathbf{u} \frac{\partial \rho}{\partial t} + \rho \frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u}) \cdot \mathbf{u} + \rho \mathbf{u} \cdot \nabla \mathbf{u}.$$

Using the continuity equation, we have that the momentum equation in the Navier-Stokes system (1.3) can be rewritten in advective form as follows:

$$\rho \frac{\partial \mathbf{u}}{\partial t} + \rho (\mathbf{u} \cdot \nabla) \mathbf{u} = \nabla \cdot \mathbf{\tau} + \nabla p = \rho \mathbf{g}.$$

(1.4)

Let us then consider the energy equation in (1.3), which can be recast as an equation for the internal energy $e_i$ only. Indeed, it can be shown that the kinetic energy behaves according to the following equation:

$$\frac{\partial}{\partial t} (\rho e_k) + \nabla \cdot (\rho e_k \mathbf{u} - \mathbf{u} \cdot \mathbf{\sigma}) + \mathbf{\sigma} : \mathbf{D} = \rho \mathbf{g} \cdot \mathbf{u}.$$

(1.5)

As a matter of fact, recalling that $e_k = \frac{1}{2} \rho \|\mathbf{u}\|^2$, by means of the Reynolds transport theorem we can write:

$$\frac{D}{Dt} \int_{V_t} \rho e_k \, d\mathbf{x} = \int_{V_t} \left( \frac{\partial}{\partial t} \left( \frac{1}{2} \rho \|\mathbf{u}\|^2 \right) + \nabla \cdot \left( \frac{1}{2} \rho \|\mathbf{u}\|^2 \mathbf{u} \right) \right) \, d\mathbf{x},$$

where $\frac{D}{Dt}$ denotes the material derivative and $V_t \subseteq \Omega$ is a generic control volume. Expanding the derivatives on the right hand side, we have:

$$\frac{D}{Dt} \int_{V_t} \rho e_k \, d\mathbf{x} = \int_{V_t} \left( \frac{1}{2} \frac{\partial \rho}{\partial t} \|\mathbf{u}\|^2 + \rho \frac{\partial \mathbf{u}}{\partial t} \cdot \mathbf{u} + \frac{1}{2} \|\mathbf{u}\|^2 \nabla \cdot (\rho \mathbf{u}) + \rho ((\mathbf{u} \cdot \nabla) \mathbf{u}) \cdot \mathbf{u} \right) \, d\mathbf{x}.$$

(1.6)

Using the continuity equation in (1.3), (1.6) reduces to:

$$\frac{D}{Dt} \int_{V_t} \rho e_k \, d\mathbf{x} = \int_{V_t} \left( \rho \frac{\partial \mathbf{u}}{\partial t} \cdot \mathbf{u} + \rho ((\mathbf{u} \cdot \nabla) \mathbf{u}) \cdot \mathbf{u} \right) \, d\mathbf{x},$$

which, thanks to the momentum equation in advective form (1.4), is equivalent to:

$$\frac{D}{Dt} \int_{V_t} \rho e_k \, d\mathbf{x} = \int_{V_t} \mathbf{u} \cdot (\rho \mathbf{g} + \nabla \cdot \mathbf{\sigma}) \, d\mathbf{x}.$$

(1.7)
CHAPTER 1. MATHEMATICAL MODELS FOR TWO-PHASE FLOWS

Now, let us notice that:

\[ \nabla \cdot (\mathbf{u} \cdot \sigma) = \sigma : \nabla \mathbf{u} + \mathbf{u} \cdot (\nabla \cdot \sigma) = \sigma : \mathbf{D} + \mathbf{u} \cdot (\nabla \cdot \sigma), \]  

(1.8)

the last equivalence being due to the the symmetry of \( \sigma \). Substituting (1.8) in (1.7), we finally have:

\[ \frac{D}{Dt} \int_{V_t} \rho e_k \, d\mathbf{x} = \int_{V_t} \left( \rho \mathbf{g} \cdot \mathbf{u} + \nabla \cdot (\mathbf{u} \cdot \sigma) - \sigma : \mathbf{D} \right) \, d\mathbf{x}. \]

which yields (1.5), since the control volume \( V_t \) is generic.

Subtracting (1.5) from the third equation in (1.3) and using the constitutive equations (1.2), we get to the following form of the energy equation:

\[ \frac{\partial}{\partial t} (\rho c_v T) + \nabla \cdot (\rho c_v T \mathbf{u} - k \nabla T) - \mathbf{\tau} : \mathbf{D} + p \nabla \cdot \mathbf{u} = 0. \]

The term \( \mathbf{\tau} : \mathbf{D} \) accounts for the increase of internal energy due to viscous dissipation. However, the velocities involved in the real-world cases we are interested in are small enough to neglect this contribution in the remainder of this work.

Writing out the time derivatives and the divergence terms explicitly and assuming \( c_v \) to be constant, we have:

\[ \rho c_v \frac{\partial T}{\partial t} + c_v T \frac{\partial \rho}{\partial t} + \rho c_v \mathbf{u} \cdot \nabla T + c_v T \nabla \cdot (\rho \mathbf{u}) - \nabla \cdot (k \nabla T) + p \nabla \cdot \mathbf{u} = 0, \]

which, using once again the continuity equation, reduces to

\[ \rho c_v \frac{\partial T}{\partial t} + \rho c_v \mathbf{u} \cdot \nabla T - \nabla \cdot (k \nabla T) + p \nabla \cdot \mathbf{u} = 0. \]  

(1.9)

Another common form for the energy equation makes use of a different thermodynamic variable called enthalpy, which will be denoted by \( h \). As seen in [Tor09], enthalpy satisfies the following relations:

\[ h = e_i + \frac{p}{\rho}, \]

\[ h = c_p T. \]

With the same argument used above, the specific energy \( e \) can be substituted out of the third equation of system (1.3) to yield the following form for the energy equation:

\[ \rho c_p \frac{\partial T}{\partial t} + \rho c_p \mathbf{u} \cdot \nabla T - \nabla \cdot (k \nabla T) = \frac{\partial p}{\partial T} + \mathbf{u} \cdot \nabla p, \]  

(1.10)
We can then write the final version of system (1.1):

\[
\begin{align*}
\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) &= 0, \\
\rho \frac{\partial \mathbf{u}}{\partial t} + \rho (\mathbf{u} \cdot \nabla) \mathbf{u} - \nabla \cdot \mathbf{\tau} + \nabla p &= \rho \mathbf{g}, \\
\rho c_p \frac{\partial T}{\partial t} + \rho c_p \mathbf{u} \cdot \nabla T - \nabla \cdot (k \nabla T) &= \frac{\partial p}{\partial t} + \mathbf{u} \cdot \nabla p, \\
p &= \rho r T.
\end{align*}
\]

(1.11)

The system must finally be closed by suitable initial and boundary conditions.

Notice that system (1.11) could have been written just as well using equation (1.9) instead of (1.10), since the two have been proven to be equivalent. The choice of deriving two different but equivalent energy equations for system (1.11) was made considering the models that will be introduced in section 1.3. Indeed, the two models for compressible-incompressible flows in sections 1.3.2 and 1.3.3 will make use of the equivalence between these two formulations for the energy equation.

1.2 A two-phase incompressible-incompressible model

In this section the governing equations for an incompressible-incompressible two-fluid flow will be introduced. Space-time discretization will be discussed, and a numerical assessment based on literature benchmarks will be shown. For the sake of completeness, a full non-isothermal model will be presented, even though the incompressible-incompressible test cases that will be considered in this work are actually all isothermal. Nevertheless, as discussed in section 1.1, the energy equation will come in handy in the next sections, when the compressible-incompressible models are presented.

1.2.1 Interface capturing technique

One of the key aspects that characterize two-phase flows models is the need for an interface capturing method. In this work a Level Set Method approach is used to identify the different phases in the domain and track the interface between them, which is represented by the zero isoline of a signed distance function \( \varphi (\mathbf{x}, t) \). This function is then used to compute a Heaviside function \( H (\mathbf{x}, t) \), which in turn is employed to define the physical properties (density, viscosity and so on) in the whole domain by mixing the specific properties of each single fluid. In particular, the signed distance function \( \varphi (\mathbf{x}, t) \) is supposed to be negative inside phase 1 and positive inside phase 2. The Heaviside function \( H (\mathbf{x}, t) \) is then defined at each point \( (\mathbf{x}, t) \) in the cylindrical
domain $\Omega \times [t_0, t_f]$ as:

$$H(x, t) = \begin{cases} 
1 & \text{if } \varphi(x, t) \leq 0 \\
0 & \text{if } \varphi(x, t) > 0,
\end{cases}$$

so that it is equal to 1 in phase 1 and to 0 in phase 2.

From here on after, subscripts 1 and 2 will denote the value of each quantity in the different liquids in the incompressible-incompressible case, while in the compressible-incompressible models we will use subscripts $l$ and $g$ to indicate “liquid” and “gas”. Furthermore, the gaseous domain will always be labeled as phase 1, so that $\varphi$ is negative inside the gas and positive in the liquid (and, therefore, $H(x, t) = 1$ inside the gas bubbles).

Notice that, in a three-dimensional setting, we would have to consider the zero isosurface of $\varphi(x, t)$ instead of the zero isoline. We will suppose $\Omega \subset \mathbb{R}^2$ throughout the rest of this section, for the sake of ease of presentation, but everything can be easily extended to a 3D case.

The Level Set Method is a standard approach for two-phase models in literature, and, as discussed in the introduction, a good overview of such methods can be found in [SS03]. The original idea comes from [SSO94], where one of the major drawbacks of this approach is also discussed. As a matter of fact, the advection of $\varphi$ causes it to lose its signed-distance-function shape. To prevent this from happening, a reinitialization step is necessary to bring $\varphi$ back to its theoretical shape. This process builds a new function which shares the zero isoline with the solution of the advection equation while, at the same time, making sure that the reinitialized signed distance is properly shaped. To perform the reinitialization, there is a plethora of methods to choose from. For example, in [SSO94] a Hamilton-Jacobi equation is used to this end. Such equation is then improved upon in [RS00]. Other later works, on the other hand, get rid of the reinitialization step all together, by using carefully crafted modified advection equations (see for example [Chu+05], [Li+10] and [GF00], just to name a few).

To keep things simple, in this work we chose the most straightforward (albeit not quite the most efficient) reinitialization method possible. First of all, the coordinates of the points where the zero-level isoline meets the triangulation on $\Omega$ are stored. If the intersection happens to be on an edge rather than on a mesh node, a weighted average of the values of the signed distance on the boundaries of such edge is used to compute the zero-level point. Note that this is a pretty accurate approximation, since the signed distance is supposed to be linear in the first place. Then the value of the signed distance function in the whole domain is determined by simply computing the distance of each point in the mesh to the closest zero-level point.

Furthermore, in order to preserve the correct mass of the bubbles, the signed
distance function is shifted upwards or downwards after it has been reinitialized. In practice, this amounts to building a new signed distance function \( \varphi(x, t; S) \), defined as:

\[
\varphi(x, t; S) = \varphi(x, t) + S,
\]

for a given constant \( S \in \mathbb{R} \). Notice that this does not change the signed distance function shape. However, by rigidly shifting the signed distance function, the zero isoline defining the interface between the two phases moves, therefore changing the computed values for the gas volume and mass. It is then possible to find the exact value for the shift \( S \) at each time step so that the gas mass computed using \( \varphi(x, t; S) \) remains equal to its initial value. In this work a simple bisection algorithm is used to this aim. Once this value is found, the signed distance function \( \varphi(x, t) \) is replaced by its shifted counterpart.

Finally, it is also worth pointing out that a surface tension term \( s(x, t) \) has to be added to the right hand side of the momentum equation in two-phase models. Indeed, surface tension is a physical quantity that makes no sense for a model with a single phase but that appears naturally once two immiscible fluids come into contact. If we denote by \( \Sigma \) the interface between the two phases, then surface tension is given by

\[
s(x, t) = \varsigma \kappa \mathbf{n} \quad \forall (x, t) \in \Sigma(t),
\]

where \( \mathbf{n} \) is the normal vector to the interface and \( \kappa \) is the curvature. They are respectively given by

\[
\mathbf{n} = \frac{\nabla \varphi}{||\nabla \varphi||}, \quad \kappa = -\nabla \cdot \mathbf{n} = -\nabla \cdot \frac{\nabla \varphi}{||\nabla \varphi||}.
\]

Note that \( s(x, t) \) can be equivalently defined on the whole domain \( \Omega \) as:

\[
s(x, t) = \varsigma \kappa \mathbf{n} \delta(\Sigma),
\]

where \( \delta(\Sigma) \) is the delta function (or Dirac’s function) on \( \Sigma \). See [Par04] for further details. Moreover, no mass transfer at the interface between the two phases will be considered, so surface tension is the only specific term to be added to the equations when we switch from single-phase to multiphase models.

### 1.2.2 The continuous model

The incompressible flow model for a single fluid with constant density can be derived from system (1.11). Under the incompressibility hypothesis, which states that
CHAPTER 1. MATHEMATICAL MODELS FOR TWO-PHASE FLOWS

\[ \nabla \cdot \mathbf{u} = 0, \]

the continuity equation reduces to

\[ \frac{\partial \rho}{\partial t} + \mathbf{u} \cdot \nabla \rho = 0, \]

which is equivalent to stating that the density remains constant along the trajectory of the particles. Thus, a fluid having a uniform density at initial time will have the same density at any time and at any point in space. The continuity equation can then be replaced by the incompressibility constraint:

\[ \nabla \cdot \mathbf{u} = 0. \quad (1.15) \]

Under this constraint, the viscous stress tensor assumes the following simplified form:

\[ \tau = \mu (\nabla \mathbf{u} + (\nabla \mathbf{u})^T). \quad (1.16) \]

As for the energy equation, the constraint given by equation (1.15) applied to (1.9) gives

\[ \rho c_p \frac{\partial T}{\partial t} + \rho c_p \mathbf{u} \cdot \nabla T - \nabla \cdot (k \nabla T) = 0. \]

Indeed, in this case, the specific heat constants coincide (i.e. \( c_p = c_v \)) and we will consider \( c_p \), for the sake of uniformity with the compressible case.

To sum up, in the incompressible case the flow of a single fluid is governed by the following system of partial differential equations:

\[
\begin{align*}
\nabla \cdot \mathbf{u} &= 0, \\
\rho \frac{\partial \mathbf{u}}{\partial t} + \rho (\mathbf{u} \cdot \nabla) \mathbf{u} - \nabla \cdot \tau + \nabla p &= \rho \mathbf{g}, \\
\rho c_p \frac{\partial T}{\partial t} + \rho c_p \mathbf{u} \cdot \nabla T - \nabla \cdot (k \nabla T) &= 0.
\end{align*}
\]

When considering multi-phase flows with different immiscible incompressible phases, each of them can be modeled in the same way, but the physical parameters characterizing each phase should be considered in the model. As seen in section 1.2.1, a Heaviside function is used to identify the phase in each point \((x,t)\) in the cylindrical domain \(\Omega \times [t_0, t_f]\):

\[ H = \begin{cases} 
1 & \text{in phase 1}, \\
0 & \text{in phase 2}.
\end{cases} \]

This function is built upon a signed distance function, negative inside phase 1 and positive inside phase 2, which is advected by the velocity using a standard advection equation:

\[ \frac{\partial \varphi}{\partial t} + \mathbf{u} \cdot \nabla \varphi = 0. \quad (1.17) \]

Putting it all back together, and recalling that the surface tension term \(s(x,t)\) has
1.2. A TWO-PHASE INCOMPRESSIBLE-INCOMPRESSIBLE MODEL

to be added to the right hand side of the momentum equation (see section 1.2.1 and in particular equations (1.13)-(1.14)), the following system describes the flow in both phases in a unified way, relying on a single set of variables valid throughout the whole domain:

\[
\begin{align*}
\nabla \cdot \mathbf{u} &= 0 \quad \text{in } \Omega \times (t_0, t_f], \\
\rho \frac{\partial \mathbf{u}}{\partial t} + \rho (\mathbf{u} \cdot \nabla) \mathbf{u} - \nabla \cdot \tau + \nabla p &= \rho g + \mathbf{s} \quad \text{in } \Omega \times (t_0, t_f], \\
\rho c_p \left( \frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T \right) - \nabla \cdot (k \nabla T) &= 0 \quad \text{in } \Omega \times (t_0, t_f], \\
\frac{\partial \varphi}{\partial t} + \mathbf{u} \cdot \nabla \varphi &= 0 \quad \text{in } \Omega \times (t_0, t_f], \\
H = H(\varphi) &= \begin{cases} 
1 & \text{if } \varphi \leq 0 \\
0 & \text{if } \varphi > 0 
\end{cases} \quad \text{in } \Omega \times (t_0, t_f], \quad (1.18) \\
\rho &= \rho_1 H + \rho_2 (1 - H) \quad \text{in } \Omega \times (t_0, t_f], \\
\mu &= \mu_1 H + \mu_2 (1 - H) \quad \text{in } \Omega \times (t_0, t_f], \\
c_p &= c_{p1} H + c_{p2} (1 - H) \quad \text{in } \Omega \times (t_0, t_f], \\
k &= k_1 H + k_2 (1 - H) \quad \text{in } \Omega \times (t_0, t_f], \\
\text{boundary conditions} & \quad \text{on } \partial \Omega \times (t_0, t_f], \\
\text{initial conditions} & \quad \text{in } \Omega.
\end{align*}
\]

It has been shown in the literature (see e.g. [DQQ13]) that the presence of discontinuous coefficients in boundary value problems may give rise to numerical issues. To reduce the onset of instabilities of the discrete solution at the interface, the standard approach in the level set literature relies on the introduction of a smoothed Heaviside function that will be substituted to (1.12) in system (1.18) (see [Par04] and the references therein). Such smooth approximation reads:

\[
H_\varepsilon(x) = \begin{cases} 
1 & \text{if } x < -\frac{\varepsilon}{2}, \\
\frac{1}{2} \left( 1 + \cos \left( \frac{\pi (2x + \varepsilon)}{2 \varepsilon} \right) \right) & \text{if } -\frac{\varepsilon}{2} \leq x \leq \frac{\varepsilon}{2}, \\
0 & \text{if } x > \frac{\varepsilon}{2},
\end{cases} \quad (1.19)
\]

for a given a constant \(\varepsilon \in \mathbb{R}^+\).

The same goes for the Dirac function in (1.14). A smooth approximation of the delta function can be obtained by computing the derivative of (1.19) with respect to
\( \delta_\epsilon (x) = \frac{dH_\epsilon}{dx} = \begin{cases} 
0 & \text{if } x < -\frac{\epsilon}{2}, \\
-\frac{\pi}{2}\epsilon \sin \left( \frac{\pi x + \epsilon}{2\epsilon} \right) & \text{if } -\frac{\epsilon}{2} \leq x \leq \frac{\epsilon}{2}, \\
0 & \text{if } x > \frac{\epsilon}{2}. 
\end{cases} \)  

(1.20)

### 1.2.3 Space-time discretization

System (1.18) was approximated using a Finite Element discretization in space, while a Finite Difference approach was used for the time discretization. To this end, the solution interval \( [t_0, t_f] \) is divided into \( N \) subintervals of length \( \Delta t = \frac{t_f - t_0}{N} \). In the following, a superscript \( (n) \) will denote the value of each quantity at time \( t_{p n} \).

In the space discretization at each time step \( t_{p n} \), for \( u_{p n} \) we use continuous \( P_2/P_1 \) finite elements, which are inf-sup stable (see e.g. [QV96]); for \( \varphi_{p n} \) we choose discontinuous \( P_1 \) finite elements, following the discretization for the level set equation adopted in [DLP06]; indeed, the use of discontinuous finite elements for the solution of the purely advective level set equation can yield improved stability and mass conservation properties (as shown in [DLP06]). Finally, continuous \( P_1 \) finite elements are used for all the remaining quantities. For each continuous quantity, a subscript \( h \) will denote its discrete counterpart.

The result of the space-time discretization is a nonlinear system of equations that needs to be solved at each time step. As a monolithic approach that yields the solution for all the variables simultaneously would be computationally unfeasible given the complexity of the system, we employ a staggered approach to define a sequence of simpler problems to be solved at each time step.

Let us introduce some preliminary notation that will be used throughout the remainder of this work. Let \( \mathcal{T}_h \) be a triangulation on the domain \( \Omega \) and let \( \{E_i\} \) be the elements of \( \mathcal{T}_h \). We consider the following discrete functional spaces:

\[
V_h = \left\{ v_h \in C^0 (\Omega) : v_h|_E \in \mathbb{P}^2 \quad \forall E \in \mathcal{T}_h \right\} , \\
Q_h = \left\{ q_h \in C^0 (\Omega) : q_h|_E \in \mathbb{P}^1 \quad \forall E \in \mathcal{T}_h \right\} , \\
W_h = \left\{ w_h \in L^2 (\Omega) : w_h|_E \in \mathbb{P}^1 \quad \forall E \in \mathcal{T}_h \right\} .
\]

Furthermore, we denote by \( V_{h, \Gamma} \) (respectively, \( Q_{h, \Gamma} \)) the space of all the functions in \( V_h \) (respectively, \( Q_h \)) which are zero-valued on \( \Gamma \subseteq \partial \Omega \), and the simpler notation \( V_{h,0} \) (respectively, \( Q_{h,0} \)) will stand for \( V_{h, \partial \Omega} \) (respectively, \( Q_{h, \partial \Omega} \)). Finally, we denote by \( Q_{h,0}^0 \) the space of all the functions in \( Q_h \) whose average on \( \Omega \) is zero.

Consider then an edge \( e \) of an element \( E_i \in \mathcal{T}_h \). Given generic scalar-valued and vector-valued functions \( f \) and \( \mathbf{F} \), let \( f_i \) and \( \mathbf{F}_i \) denote the traces of \( f \) and \( \mathbf{F} \) on \( e \) as taken in \( E_i \), respectively, and let \( \nu_i \) be the outward normal vector on \( e \) with respect
to $E_i$. If $e$ is an internal edge shared by elements $E_i$ and $E_j$, we define:

$$\llbracket f \rrbracket := f_i \nu_i + f_j \nu_j,$$

$$\{ f \} := \frac{1}{2} (f_i + f_j),$$

$$\llbracket F \rrbracket := F_i \cdot \nu_i + F_j \cdot \nu_j,$$

$$\{ F \} := \frac{1}{2} (F_i + F_j).$$

For a boundary edge $e$, on the other hand, we define:

$$\llbracket f \rrbracket := f_i \nu_i,$$

$$\{ f \} := f_i.$$

Finally, we denote by $E_i$ the set of all internal edges and $E_B$ the set of all boundary edges.

Let us now describe the staggered approach used to solve system (1.18). First of all, we remark that the time derivatives in equations (1.18)3 and (1.18)4 were discretized using a first-order implicit Euler scheme, while a semi-implicit method was used to treat the nonlinearity in the momentum equation (1.18)2.

As for the coefficients in each equation, the latest available solution is always used. This means that, as soon as the value of one of the unknowns at the current time step is computed, this same value is used each time this quantity appears in subsequent equations at the same time step. For example, on step 1T below the solution to the velocity-pressure problem at time $t^{(n)}$ is computed. From that moment on, until the end of the solution process for time step $n$ (i.e. step 6T), the new value for velocity and pressure is used each time these variables are needed.

Secondly, a set of boundary conditions must be enforced on the partial differential equations in system (1.18). For the sake of generality, let us suppose that the following boundary conditions hold:

$$\begin{cases}
\mathbf{u} = \mathbf{u}_D & \text{on } B_D \times (t_0, t_f], \\
-p \mathbf{v} + \mathbf{t} \cdot \mathbf{v} = \mathbf{u}_N & \text{on } B_N \times (t_0, t_f], \\
T = T_D & \text{on } \Gamma_D \times (t_0, t_f], \\
k \nabla T \cdot \mathbf{v} = T_N & \text{on } \Gamma_N \times (t_0, t_f], \\
\varphi = \varphi_D & \text{on } \Gamma_m \times (t_0, t_f],
\end{cases}$$

with $B_D \cup B_N = \Gamma_D \cup \Gamma_N = \partial \Omega$, $B_D \cap B_N = \Gamma_D \cap \Gamma_N = \emptyset$, and $\Gamma_m$ denoting the portion of $\partial \Omega$ where $\mathbf{u} \cdot \mathbf{v} > 0$.

Lastly, a set of initial condition is needed. Let the initial values $\varphi^{(0)}$, $\mathbf{u}_h^{(0)}$, $\rho_h^{(0)}$, $\mu_h^{(0)}$, $c_p^{(0)}$, and $k_h^{(0)}$ be given.

Then, for every $n = 1, 2, \ldots, N$, the solution process is as follows:

1T. Compute $\mathbf{u}_h^{(n)}$ and $\rho_h^{(n)}$, by solving the following discrete problem:

Find the pair $\left( \mathbf{u}_h^{(n)}, \rho_h^{(n)} \right)$ in $V_h^d \times Q_h$ such that $\mathbf{u}_h^{(n)}|_{B_D} = \mathbf{u}_{D,h}$ and for all
pairs \((v_h, q_h) \in [V_h, B_D]^d \times Q_h\)

\[
\int_\Omega \frac{\rho_h^{(n-1)}}{\Delta t} u_h^{(n)} \cdot v_h \, dx + \int_\Omega \frac{\rho_h^{(n-1)}}{\Delta t} (u_h^{(n-1)} \cdot \nabla) u_h^{(n)} \cdot v_h \, dx + \int_\Omega \tau_h^{(n)} \cdot \nabla v_h \, dx + \\
- \int_\Omega \rho_h^{(n)} \nabla \cdot v_h \, dx + \int_\Omega \nabla \cdot u_h^{(n)} q_h \, dx = \int_\Omega \frac{\rho_h^{(n-1)}}{\Delta t} u_h^{(n-1)} \cdot v_h \, dx + \\
+ \int_\Omega \rho_h^{(n-1)} g \cdot v_h \, dx + \int_\Omega s_h^{(n-1)} \cdot v_h \, dx + \int_{B_N} u_{N,h} \cdot v_h \, dx,
\]

where

\[
\tau_h^{(n)} = \mu_h^{(n-1)} \nabla u_h^{(n)} + \left( \nabla u_h^{(n)} \right)^T
\]

and

\[
s_h^{(n-1)} = \zeta \kappa_h^{(n-1)} n_h^{(n-1)} \delta \left( \nabla \phi_h^{(n-1)} \right),
\]

with \(\kappa_h^{(n-1)}\) and \(n_h^{(n-1)}\) computed using (1.13). Notice that if \(B_D = \partial \Omega\) the pressure \(p_h^{(n)}\) actually has to be sought for in \(Q_h^0\) instead of \(Q_h\).

2T. Compute the temperature \(T_h^{(n)}\), by solving the following discrete problem:

Find \(T_h^{(n)}\) in \(Q_h\) such that \(T_h^{(n)} \big|_{\Gamma_D} = T_{D,h}\) and for all \(q_h \in Q_h, \Gamma_D\)

\[
\int_\Omega \rho_h^{(n-1)} c_p^{(n-1)} \frac{T_h^{(n)}}{\Delta t} q_h \, dx + \int_\Omega \rho_h^{(n-1)} c_p^{(n-1)} u_h^{(n)} \cdot \nabla T_h^{(n)} \cdot q_h \, dx + \\
+ \int_\Omega k_h^{(n-1)} \nabla T_h^{(n)} \cdot \nabla q_h \, dx = \int_\Omega \rho_h^{(n-1)} c_p^{(n-1)} \frac{T_h^{(n-1)}}{\Delta t} q_h \, dx + \int_{\Gamma_N} T_{N,h} q_h \, dx.
\]

3T. Compute the signed distance function \(\varphi_h^{(n)}\), by solving the following discrete level-set equation:

Find \(\varphi_h^{(n)}\) in \(W_h\) such that for all \(w_h \in W_h\)

\[
\sum_{E \in \mathcal{T}_h} \int_E \frac{\varphi_h^{(n)}}{\Delta t} w_h \, dx - \sum_{E \in \mathcal{T}_h} \int_E \varphi_h^{(n)} u_h^{(n)} \cdot \nabla w_h \, dx - \sum_{E \in \mathcal{T}_h} \int_E \varphi_h^{(n)} \nabla \cdot u_h^{(n)} w_h \, dx + \\
+ \sum_{e \in \mathcal{E}_h} \int_e \left( \left\{ \varphi_h^{(n)} \right\} u_h^{(n)} \| w_h \| + \frac{1}{2} \| u_h^{(n)} \cdot \nu \| \varphi_h^{(n)} \| w_h \| \right) \, ds + \\
+ \sum_{e \in \mathcal{E}_h} \int_e u_h^{(n)} \cdot \| w_h \| \left\{ \varphi_h^{(n)} \right\} \, ds = \sum_{E \in \mathcal{T}_h} \int_E \frac{\varphi_h^{(n-1)}}{\Delta t} w_h \, dx,
\]

where

\[
\varphi_h^2 = \begin{cases} 
\varphi_{D,h} & \text{if } u_h^{(n)} \cdot \nu > 0, \\
\varphi_{h} & \text{if } u_h^{(n)} \cdot \nu \leq 0.
\end{cases}
\]

See [DLP06] for more details on how this discrete weak formulation was obtained. Then reinitialize it to preserve the proper shape and shift it rigidly to preserve
1.2. A TWO-PHASE INCOMPRESSIBLE-INCOMPRESSIBLE MODEL

<table>
<thead>
<tr>
<th>Test case</th>
<th>$\rho_1$</th>
<th>$\rho_2$</th>
<th>$\mu_1$</th>
<th>$\mu_2$</th>
<th>$g$</th>
<th>$\zeta$</th>
<th>$\rho_2/\rho_1$</th>
<th>$\mu_2/\mu_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td>1000</td>
<td>1</td>
<td>10</td>
<td>(0, -0.98)</td>
<td>24.5</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1000</td>
<td>0.1</td>
<td>10</td>
<td>(0, -0.98)</td>
<td>1.96</td>
<td>1000</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 1.1: Physical parameters for the two incompressible-incompressible test cases

the initial mass (see section 1.2.1).

4T. Compute $H_h^{(n)}$, using the smoothed approximation (1.19).

5T. Update the physical parameters $\rho_h^{(n)}$, $\mu_h^{(n)}$, $c_{ph}^{(n)}$, $k_h^{(n)}$ using equations (1.18)$_6$, (1.18)$_7$, (1.18)$_8$ and (1.18)$_9$, respectively.

6T. If $n < N$, set $n = n + 1$ and go back to 1T.

All the linear systems obtained from the discrete formulations in steps 1T-6T, as well as in the following sections, have been solved using the LU factorization. Moreover, notice that, for the sake of simplicity, we chose not to use an operator splitting method for the velocity-pressure problem, which is solved monolithically instead.

1.2.4 Numerical results

In this section two-phase incompressible-incompressible results are presented in order to validate the solver used in this work. The benchmark test cases are taken from [Hys+09] and investigate the behaviour of a bubble initially at rest in a denser liquid, as shown in figure 1.1. The domain has sides $L_1 = 1$ and $L_2 = 2$, while the initial bubble’s center is at $(0.5, 0.5)$ and has radius $r = 0.25$. On the domain boundary, no-slip conditions are enforced on the bottom and top walls, while slip conditions are used on the left and right sides. All quantities in this paragraph are to be intended as expressed in the International System of Units.

The results shown in this thesis, including both those in the current section and those in sections 1.3.4 and 3.1.4 and chapter 4, are all obtained using the deC library, a piece of software written in C++ completely developed during this thesis. This software is built upon the FEniCS/DOLFIN library ([FEni16] and [LMW12]) and its goal is to handle PDE-based optimal control problems, although it can also be used to solve plain systems of partial differential equations. See chapter 5 for further details on the implementation.

Two different test cases are considered, which only differ in the choice of the physical parameters. Table 1.1 sums up the values used in both cases. Of the two test cases, the second one is trickier, since the density and viscosity ratios are higher, and the lower value for the surface tension coefficient means that the bubble may break up during the evolution.
Figure 1.1: Initial configuration for the incompressible-incompressible benchmarks
1.2. A TWO-PHASE INCOMPRESSIBLE-INCOMPRESSIBLE MODEL

Besides the bubble’s shape and position, which are given by the Heaviside function at each time step and can only be object of a qualitative comparison, some specific quantities, identified as significant in [Hys+09], are computed during the simulation in order to have a precise quantitative comparison to the benchmark results. These are:

- the vertical position of the center of the bubble
  \[ y_c = \frac{\int_{\Omega_2} y \, dx}{\int_{\Omega_2} dx} ; \]

- the rise velocity
  \[ u_c = \frac{\int_{\Omega_2} u_y \, dx}{\int_{\Omega_2} dx} , \]
  where \( u_y \) represents the velocity along the vertical axis;

- the total mass, defined as the integral of the density on the whole domain (that should be preserved, since no fluid flows out of the domain through the boundaries). This is actually enforced by the signed distance shifting in step 3T in section 1.2.3, so, even though it is indeed used as an indicator in [Hys+09], it will not be considered here.

Furthermore, two choices, common in the literature, for the process of computing the global viscosity are investigated. Indeed, viscosity may be mixed by using either:

- an arithmetic average, as seen so far, e.g. in system (1.18) and step 5T in section 1.2.3;

- a harmonic average, which means that the viscosity at each point \((x, t)\) in \(\Omega \times (t_0, t_f)\) can be computed by solving the equation:
  \[ \frac{1}{\mu} = \frac{1}{\mu_1} H (x, t) + \frac{1}{\mu_2} (1 - H (x, t)) , \]
  where \( H (x, t) \) is the indicator function.

These two approaches are used for example (among others) in [Dar+10] and [BGN11] (respectively), which are the two works where the compressible-incompressible models which will be introduced in section 1.3 were first presented. In general, as discussed in [TSZ11] and the references therein, the harmonic average approach tends to have a stabilizing effect, thus preventing numerical instabilities around the interface that may ruin the discrete solution. It is then interesting to see how the two approaches compare when the numerical scheme considered in this work is used to solve system (1.18). To this end, the benchmarks were solved two times, first using the arithmetic average and then the harmonic average to compute the global viscosity.
In [Hys+09], three different groups participated in the experiment, each using their own source code for the implementation. The data obtained by the second group (called FreeLIFE in the original paper) are used in this work as a comparison. Because of this, the same values for the mesh size and the time step were used. This means that three different meshes are considered, with mesh size $h$ equal to $1/40$, $1/80$ and $1/160$ respectively, while the time step is set to $\Delta t = h/2$.

1.2.4.1 Benchmark 1

The bubble’s position at selected time steps for the first benchmark is shown in figure 1.2, while the plots in figure 1.3 show the evolution in time of the benchmark quantities (see section 1.2.4), obtained with each of the two approaches investigated. First of all, notice that for both approaches the solution tends to converge towards the benchmark results as the mesh grows finer (as seen in figure 1.3). Furthermore, at the finer mesh level, the solution obtained with the arithmetic average approach and the one where the harmonic average is used are close to each other to the point of being almost superimposed (figure 1.2). We may then conclude that, in this case, there is not much difference between the two approaches, as far as the results are concerned.

1.2.4.2 Benchmark 2

Figures 1.4 and 1.5 show the bubble’s position and the post-processing quantities for the second benchmark. Unlike section 1.2.4.1, the results here show that there is indeed a difference between the two approaches. In particular, as seen in figure 1.4,
1.2. A TWO-PHASE INCOMPRESSIBLE-INCOMPRESSIBLE MODEL

(a) Center of mass position, arithmetic average

(b) Rise velocity, arithmetic average

(c) Center of mass position, harmonic average

(d) Rise velocity, harmonic average for $\mu$

Figure 1.3: Benchmark quantities for benchmark 1
the bubble’s boundary presents numerical instabilities when the arithmetic approach is used, while remaining smoother in the harmonic average case. This is due to the stabilizing properties of the harmonic average approach, already mentioned in section 1.2.4 and discussed in [TSZ11].

Also the post-processing quantities (figure 1.5) hint at the fact that the harmonic average approach yields better results in this case. Indeed, the behaviour with this approach is much closer to the benchmark than the results obtained using an arithmetic average for the viscosity.

In spite of this, the control problems in section 3.2 will make use of the arithmetic average, in order to reduce the complexity of the derivation of the optimality conditions (see section 3.2.1.1). Because of the limitations of this approach, highlighted in this section, in the numerical tests in chapter 4 we limit ourselves to treat cases with low viscosity ratios, for which the arithmetic average has been shown to exhibit good behaviour.

1.3 Compressible-incompressible models

In this section the two different compressible-incompressible models used in this work are introduced and compared, both from a modellistic and from a numerical standpoint, highlighting similarities and differences among the two. Test cases taken from both original papers ([BGN11] and [Dar+10]) are run with both models, in order to achieve a satisfying numerical comparison.
1.3. COMPRESSIBLE-INCOMPRESSIBLE MODELS

(a) Center of mass position, arithmetic average

(b) Rise velocity, arithmetic average

(c) Center of mass position, harmonic average

(d) Rise velocity, harmonic average for $\mu$

Figure 1.5: Benchmark quantities for benchmark 2
1.3.1 Initial motivation

Before we properly introduce the compressible-incompressible models, object of this section, the choice of analyzing two different approaches in such depth needs explaining.

As stated early on in the introduction, the goal of this work is to put in place a way to control the evolution of gas bubbles within a liquid metal. To do so, the best trade-off between accuracy and simplicity of the model has to be sought. It is with this spirit that the two models presented in this section have been investigated.

These models share many similarities, as will become clear in the following sections. The main difference lies in the way pressure is handled. The basic idea for the split-pressure-term approach derived in [Dar+10] (and presented in section 1.3.3) first appeared in [Pao82], where the subject of acoustic waves in compressible flows is investigated. The key idea in [Pao82] is that it is possible to filter sound waves out of the compressible Navier-Stokes equations. Indeed, in general, a given problem may present several ways of transmitting information, each with its own propagation speed. In the case of free or forced convection of compressible fluids, information propagates through either acoustic or gravity-induced waves. If the Mach number is small enough, it can be proven, by expansion in series with respect to the Mach number of the terms in the compressible Navier-Stokes equations, that the acoustic waves in the fluid have negligible energy. This in turn means that the pressure can be considered constant in space, on a first order approximation, since the spatial variation of the pressure induced by the motion of the fluid is negligible as well. Once the terms with order 1 with respect to the Mach number are taken into account, a hydrodynamic space-dependent pressure is introduced as well, giving a model for compressible flows which the acoustic terms have been filtered out of. Such model was used as starting point in [Dar+10] to develop a similar splitting for the incompressible Navier-Stokes equations, leading to the final two-phase model that we will see in section 1.3.3.

All of this has a simple yet important consequence: acoustic waves can be filtered out of the compressible model (as their effect has been proven to be negligible), but, if they are not filtered out, they need to be solved, and this gives rise to numerical issues. Indeed, as they travel much faster than gravity-induced waves, if one needs to properly compute a solution involving acoustic waves, the time step needs to be much smaller, since one needs to make sure that the wave fronts do not travel further than a single mesh cell in a time step. This is true even for implicit time schemes, which, despite being unconditionally stable, may give inaccurate results if the acoustic waves are not solved with precision, even if they hardly carry any energy. On the other hand, neglecting the acoustic terms in the compressible model altogether allows us to do without computing the solution to the acoustic waves, meaning that a much larger time step can be used, without loss of accuracy.
1.3. COMPRESSIBLE-INCOMPRESSIBLE MODELS

This is where the comparison between the two models comes back into play: indeed, while the split-pressure model in section 1.3.3 does neglect the acoustic waves terms, the single-pressure model in section 1.3.2 does not. As we will see in the next sections, and as a consequence of the discussion carried out above, this means that the former has an advantage over the latter in terms of how big the time step can grow to be before loss of accuracy occurs in the numerical solution. However, the split-pressure approach is at a disadvantage with respect to the single-pressure method in terms of the complexity of the final system of equations.

Sections 1.3.2 and 1.3.3 will introduce the two models in details, while section 1.3.4 will present a comparison on the numerical results obtained with the two different approaches on some selected test cases. Finally, section 1.3.5 will draw a final conclusion on which model is best suited for our needs.

1.3.2 A single-pressure method

The compressible-incompressible model derived in [BGN11] will now be introduced. As discussed in section 1.3.1, the “single-pressure” label refers to the way the pressure term is handled in the model, which is in contrast with the approach used in the “split-pressure” compressible-incompressible model detailed in section 1.3.3. For further information on the model presented in this section the reader is referred to [CVC11] and [Hac+16], as well as the already cited [BGN11].

1.3.2.1 The continuous model

This model stems from the general compressible Navier-Stokes system (1.11), and it is obtained by rearranging the terms in the continuity equation. Indeed, the first equation in (1.11) can be written as:

\[
\frac{1}{\rho} \left( \frac{\partial \rho}{\partial t} + \mathbf{u} \cdot \nabla \rho \right) + \nabla \cdot \mathbf{u} = 0. \tag{1.21}
\]

We can then use the chain rule to rewrite equation (1.21) as:

\[
\frac{1}{\rho} \left( \frac{\partial \rho}{\partial t} + \mathbf{u} \cdot \nabla p \right) + \frac{1}{\rho} \left( \frac{\partial \rho}{\partial t} + \mathbf{u} \cdot \nabla T \right) + \nabla \cdot \mathbf{u} = 0.
\]

where the pedices in \( \left( \frac{\partial \rho}{\partial p} \right)_T \) and \( \left( \frac{\partial \rho}{\partial T} \right)_p \) denote derivatives at constant \( T \) and \( p \), respectively. For the sake of notation simplicity, let the following quantities be defined:
\[
\chi_p = -\frac{1}{\rho} \left( \frac{\partial \rho}{\partial T} \right)\bigg|_p, \quad \text{the volume expansivity,}
\]

and

\[
\chi_T = \frac{1}{\rho} \left( \frac{\partial \rho}{\partial p} \right)\bigg|_T, \quad \text{the isothermal compressibility,}
\]

(also called respectively \(\alpha\) and \(\beta\) in some literature on the subject). The continuity equation then becomes:

\[
\nabla \cdot \mathbf{u} + \chi_T \left( \frac{\partial p}{\partial t} + \mathbf{u} \cdot \nabla p \right) - \chi_p \left( \frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T \right) = 0.
\]

Notice that what has been done so far is completely general and can be applied to both gases and liquids. Indeed, for gases \(\chi_T\) and \(\chi_p\) will have an explicit expression depending on the constitutive equation, while for liquids, assuming that incompressibility holds, we will have \(\chi_T = \chi_p = 0\), thus reverting back to the classic continuity equation for incompressible flows. The complete system (1.11) can then be rewritten as

\[
\begin{aligned}
\nabla \cdot \mathbf{u} + \chi_T \left( \frac{\partial p}{\partial t} + \mathbf{u} \cdot \nabla p \right) - \chi_p \left( \frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T \right) &= 0, \\
\rho_g \frac{\partial \mathbf{u}}{\partial t} + \rho_g (\mathbf{u} \cdot \nabla) \mathbf{u} - \nabla \cdot \mathbf{\tau}_g + \nabla p &= \rho_g \mathbf{g}, \\
\rho_g c_{pg} \frac{\partial T}{\partial t} + \rho_g c_{pg} \mathbf{u} \cdot \nabla T - \nabla \cdot (k_g \nabla T) &= \frac{\partial p}{\partial t} + \mathbf{u} \cdot \nabla p, \\
p &= \rho_g rT,
\end{aligned}
\]

(1.22)

in the compressible phase \(^1\) and as

\[
\begin{aligned}
\nabla \cdot \mathbf{u} + \chi_l \left( \frac{\partial p}{\partial t} + \mathbf{u} \cdot \nabla p \right) - \chi_p \left( \frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T \right) &= 0, \\
\rho_l \frac{\partial \mathbf{u}}{\partial t} + \rho_l (\mathbf{u} \cdot \nabla) \mathbf{u} - \nabla \cdot \mathbf{\tau}_l + \nabla p &= \rho_l \mathbf{g}, \\
\rho_l c_{pl} \frac{\partial T}{\partial t} + \rho_l c_{pl} \mathbf{u} \cdot \nabla T - \nabla \cdot (k_l \nabla T) &= 0,
\end{aligned}
\]

in the incompressible phase, the viscous stress tensors being given by

\[
\mathbf{\tau}_g = \lambda_g (\nabla \cdot \mathbf{u}) \mathbf{I} + 2\mu_g \mathbf{D}
\]

and

\[
\mathbf{\tau}_l = \lambda_l (\nabla \cdot \mathbf{u}) \mathbf{I} + 2\mu_l \mathbf{D}
\]

\(^1\)Actually, the energy equation in [BGN11] is written in the form of (1.9), which we have proven to be equivalent to the equation used in system (1.22) anyway (see section 1.1). We decided to write out the system using the equation in the form of (1.10) for the sake of consistency among all the formulations presented in this chapter.
in the gas and liquid phases, respectively.

As in section 1.2.2, a Heaviside function, defined as

$$H = \begin{cases} 
1 & \text{in the gas,} \\
0 & \text{in the liquid,}
\end{cases}$$

and built upon a signed distance function (positive in the liquid and negative in the gas), is used to distinguish the two phases. The signed distance function is advected using equation (1.17). Recalling once again that the surface tension term $s \mathbf{u}$ is to be added to the right hand side of the momentum equation, we are then able to write a unified system valid in both phases:

\begin{align*}
\nabla \cdot \mathbf{u} + \chi_T \left( \frac{\partial p}{\partial t} + \mathbf{u} \cdot \nabla p \right) - \chi_p \left( \frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T \right) &= 0 \quad \text{in } \Omega \times (t_0, t_f], \\
\rho \frac{\partial \mathbf{u}}{\partial t} + \rho (\mathbf{u} \cdot \nabla) \mathbf{u} - \nabla \cdot \mathbf{\tau} + \nabla p &= \rho g + s \quad \text{in } \Omega \times (t_0, t_f], \\
\rho c_p \left( \frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T \right) - \nabla \cdot (k \nabla T) &= H \left( \frac{\partial p}{\partial t} + \mathbf{u} \cdot \nabla p \right) \quad \text{in } \Omega \times (t_0, t_f], \\
\frac{\partial \varphi}{\partial t} + \mathbf{u} \cdot \nabla \varphi &= 0 \quad \text{in } \Omega \times (t_0, t_f], \\
H &= H (\varphi) = \begin{cases} 
1 & \text{if } \varphi \leq 0 \\
0 & \text{if } \varphi > 0
\end{cases} \quad \text{in } \Omega \times (t_0, t_f], \\
\rho &= \frac{p}{\gamma T} H + \rho_l (1 - H) \quad \text{in } \Omega \times (t_0, t_f], \\
\mu &= \mu_g H + \mu_l (1 - H) \quad \text{in } \Omega \times (t_0, t_f], \\
\lambda &= \lambda_g H + \lambda_l (1 - H) \quad \text{in } \Omega \times (t_0, t_f], \\
c_p &= c_{p_g} H + c_{p_l} (1 - H) \quad \text{in } \Omega \times (t_0, t_f], \\
k &= k_g H + k_l (1 - H) \quad \text{in } \Omega \times (t_0, t_f], \\
\chi_T &= \frac{1}{\gamma} H \quad \text{in } \Omega \times (t_0, t_f], \\
\chi_p &= \frac{1}{\gamma} H \quad \text{in } \Omega \times (t_0, t_f], \\
\text{boundary conditions} & \text{on } \partial \Omega \times (t_0, t_f], \\
\text{initial conditions} & \text{in } \Omega.
\end{align*}

Notice that the stress tensor $\mathbf{\tau}$ has not been defined as a piecewise function depending on $H$, unlike for example $\rho$ or $k$, since the constitutive equation given in (1.2) is self-consistent. Indeed, in the incompressible phase, $\nabla \cdot \mathbf{u}$ is constrained to be zero, so the term $\lambda (\nabla \cdot \mathbf{u}) \mathbf{I}$ will be turned off in the definition for $\mathbf{\tau}$, which will then automatically revert back to (1.16).
1.3.2.2 Space-time discretization

Let us now discuss the space-time discretization of system (1.23). The same notation and definitions first introduced in section 1.2.3 will be used. The staggering approach used to decouple the continuity and the momentum equations from the energy and the level set equations is similar to the one introduced in section 1.2.3 for the incompressible case, and so is the discrete weak formulation. However, the single weak problems have to be adapted to the changes that system (1.23) presents with respect to (1.18). As in section 1.2.3, the following boundary conditions are supposed to hold:

\[
\begin{cases}
    \mathbf{u} = \mathbf{u}_D & \text{on } B_D \times (t_0, t_f], \\
    -p \mathbf{v} + \tau \cdot \mathbf{v} = \mathbf{u}_N & \text{on } B_N \times (t_0, t_f], \\
    T = T_D & \text{on } \Gamma_D \times (t_0, t_f], \\
    k \nabla T \cdot \mathbf{n} = T_N & \text{on } \Gamma_N \times (t_0, t_f], \\
    \varphi = \varphi_D & \text{on } \Gamma_{in} \times (t_0, t_f], \\
\end{cases}
\]

with \( B_D \cup B_N = \Gamma_D \cup \Gamma_N = \partial \Omega, B_D \cap B_N = \Gamma_D \cap \Gamma_N = \emptyset, \) and \( \Gamma_{in} \) denoting the portion of \( \partial \Omega \) where \( \mathbf{u} \cdot \mathbf{n} > 0. \)

Let then the initial conditions \( \varphi_h^{(0)}, \mathbf{u}_h^{(0)}, p_h^{(0)}, T_h^{(-1)}, T_h^{(0)}, \rho_h^{(0)}, \rho_h^{(0)}, \lambda_h^{(0)}, c_p^{(0)}, k_h^{(0)} \) and \( \lambda_h^{(0)} \) and \( \lambda_h^{(0)} \) be given. Notice that we need two initial values for the temperature in order to compute its derivative in the continuity equation at the first time step. This is due to the staggered decoupling of the equations in system (1.23) and it has no physical meaning. The most sensible approach, that we will follow unless stated otherwise, is to assume \( T^{(-1)} = T^{(0)} \), i.e. supposing that \( \frac{\partial T}{\partial t} \bigg|_{t=t_0} = 0 \), the same holding for the discrete temperature \( T_h \).

Then for every \( n = 1, 2, \ldots, N \), the solution process is the following:

1B. Compute \( \mathbf{u}_h^{(n)} \) and \( p_h^{(n)} \), by solving the following discrete problem:

Find the pair \( (\mathbf{u}_h^{(n)}, p_h^{(n)}) \) in \( V_h^d \times Q_h \) such that \( \mathbf{u}_h^{(n)} \big|_{B_D} = \mathbf{u}_D,h \) and for all pairs \( (\mathbf{v}_h, q_h) \in [V_h,B_D]^d \times Q_h \)

\[
\begin{align*}
    &\int_{\Omega} \rho_h^{(n-1)} \frac{\partial}{\partial t} \mathbf{u}_h^{(n)} \cdot \mathbf{v}_h \, dx + \int_{\Omega} \rho_h^{(n-1)} \left( \mathbf{u}_h^{(n-1)} \cdot \nabla \right) \mathbf{u}_h^{(n)} \cdot \mathbf{v}_h \, dx + \int_{\Omega} \tau_h^{(n-1)} \cdot \nabla \mathbf{v}_h \, dx + \\
    &- \int_{\Omega} \rho_h^{(n)} \nabla \cdot \mathbf{v}_h \, dx + \int_{\Omega} \mathbf{v}_h \cdot q_h \, dx + \int_{\Omega} \chi_{T_h}^{(n-1)} \frac{\partial}{\partial t} p_h^{(n)} q_h \, dx + \\
    &+ \int_{\Omega} \lambda_{T_h}^{(n-1)} \mathbf{u}_h^{(n-1)} \cdot \nabla p_h^{(n)} q_h \, dx = \int_{\Omega} \rho_h^{(n-1)} \frac{\partial}{\partial t} \mathbf{u}_h^{(n-1)} \cdot \mathbf{v}_h \, dx + \int_{\Omega} \rho_h^{(n-1)} g \cdot \mathbf{v}_h \, dx + \\
    &+ \int_{\Omega} \lambda_{T_h}^{(n-1)} \frac{\partial}{\partial t} p_h^{(n-1)} q_h \, dx + \int_{\Omega} \lambda_{p_h}^{(n-1)} \frac{\partial}{\partial t} \left( T_h^{(n-1)} - T_h^{(n-2)} \right) q_h \, dx + \\
    &+ \int_{\Omega} \chi_{T_h}^{(n-1)} \mathbf{u}_h^{(n-1)} \cdot \nabla T_h^{(n-1)} q_h \, dx + \int_{\Omega} s_h^{(n-1)} \cdot \mathbf{v}_h \, dx + \int_{B_N} \mathbf{u}_{N,h} \cdot \mathbf{v}_h \, dx,
\end{align*}
\]
1.3. COMPRESSIBLE-INCOMPRESSIBLE MODELS

where

\[ \tau_h^{(n)} = \lambda_h^{(n-1)} \left( \nabla \cdot u_h^{(n)} \right) I + 2\mu_h^{(n-1)} \frac{\nabla u_h^{(n)}}{2} \]

and

\[ s_h^{(n-1)} = \zeta \kappa_h^{(n-1)} n_h^{(n-1)} \delta \left( \Sigma_h^{(n-1)} \right) , \]

with \( \kappa_h^{(n-1)} \) and \( n_h^{(n-1)} \) computed using (1.13).

2B. Compute the temperature \( T_h^{(n)} \), by solving the following discrete problem:

Find \( T_h^{(n)} \) in \( Q_h \) such that \( T_h^{(n)} \mid_{\Gamma_D} = T_{D,h} \) and for all \( q_h \in Q_h, \Gamma_D \)

\[
\int_{\Omega} \rho_h^{(n-1)} c_{ph}^{(n-1)} \frac{T_h^{(n)}}{\Delta t} q_h \, dx + \int_{\Omega} \rho_h^{(n-1)} c_{ph}^{(n-1)} u_h^{(n)} \cdot \nabla T_h^{(n)} q_h \, dx + \\
+ \int_{\Omega} k_h^{(n-1)} \nabla T_h^{(n)} \cdot \nabla q_h \, dx = \int_{\Omega} \rho_h^{(n-1)} c_{ph}^{(n-1)} \frac{T_h^{(n-1)}}{\Delta t} q_h \, dx + \\
+ \int_{\Omega} H_h^{(n-1)} \left( \frac{\partial p_h^{(n)}}{\partial t} + u_h^{(n)} \cdot \nabla p_h^{(n)} \right) q_h \, dx + \int_{\Gamma_N} T_{N,h} q_h \, dx .
\]

3B. Compute the signed distance function \( \varphi_h^{(n)} \), by solving the following discrete level-set equation:

Find \( \varphi_h^{(n)} \) in \( W_h \) such that for all \( w_h \in W_h \)

\[
\sum_{E \in \mathcal{T}_h} \int_{E} \frac{\varphi_h^{(n)}}{\Delta t} w_h \, dx - \sum_{E \in \mathcal{T}_h} \int_{E} \varphi_h^{(n)} u_h^{(n)} \cdot \nabla w_h \, dx - \sum_{E \in \mathcal{T}_h} \int_{E} \varphi_h^{(n)} \nabla \cdot u_h^{(n)} w_h \, dx + \\
+ \sum_{e \in \mathcal{E}_h} \int_{e} \left( \left\{ \varphi_h^{(n)} \right\} \left[ u_h^{(n)} \right] \left[ w_h \right] + \frac{1}{2} \left[ u_h^{(n)} \right] \left[ \varphi_h^{(n)} \right] \left[ \nabla w_h \right] \right) \, ds + \\
+ \sum_{e \in \mathcal{E}_h} \int_{e} u_h^{(n)} \left[ \nabla w_h \right] \left\{ \varphi_h^{(n)} \right\} \, ds = \sum_{E \in \mathcal{T}_h} \int_{E} \frac{\varphi_h^{(n-1)}}{\Delta t} w_h \, dx ,
\]

where

\[ \varphi_h^* = \left\{ \begin{array}{ll}
\varphi_{D,h} & \text{if } u_h^{(n)} \cdot \nu > 0 , \\
\varphi_h^{(n)} & \text{if } u_h^{(n)} \cdot \nu \leq 0 .
\end{array} \right. \]

Once again, see [DLP06] for more details. Then reinitialize it to preserve the proper shape and shift it rigidly to preserve the theoretical mass (see section 1.2.1).

4B. Compute \( H_h^{(n)} \), using the smoothed approximation (1.19).

5B. Update the physical parameters \( \rho_h^{(n)} , \mu_h^{(n)} , \lambda_h^{(n)} , c_{ph}^{(n)} , k_h^{(n)} , \lambda^{(n)} , \chi_{T,h}^{(n)} \) using equations (1.23) to (1.23)_{11}, respectively.

6B. If \( n < N \), set \( n = n + 1 \) and go back to 1B.
1.3.3 A split-pressure method

The split-pressure model for two-phase compressible-incompressible flows, already teased in section 1.3.1, will now be introduced. For more details, the reader is referred to [Dar+10].

1.3.3.1 The continuous model

Once again, the starting point is system (1.11). In the liquid phase, the usual model holds true:

\[
\begin{aligned}
\nabla \cdot \mathbf{u} &= 0, \\
\rho_l \frac{\partial \mathbf{u}}{\partial t} + \rho_l (\mathbf{u} \cdot \nabla) \mathbf{u} - \nabla \cdot \mathbf{\tau}_l + \nabla p &= \rho_l \mathbf{g}, \\
\rho_l c_p \frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T - \nabla \cdot (k_l \nabla T) &= 0,
\end{aligned}
\]

while in the gas phase we write:

\[
\begin{aligned}
\rho_g \frac{\partial \mathbf{u}}{\partial t} + \rho_g (\mathbf{u} \cdot \nabla) \mathbf{u} - \nabla \cdot \mathbf{\tau}_g + \nabla p &= \rho_g \mathbf{g}, \\
\rho_g c_p \frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T - \frac{1}{\rho_g c_p} \nabla \cdot (k \nabla T) - \frac{\gamma - 1}{\gamma} \frac{T}{p} \left( \frac{\partial p}{\partial t} + \mathbf{u} \cdot \nabla p \right) &= 0,
\end{aligned}
\]

where \( \gamma = c_p/c_v \) and, as in section 1.3.2, \( \mathbf{\tau}_g \) and \( \mathbf{\tau}_l \) are given by \( \lambda_g (\nabla \cdot \mathbf{u}) \mathbf{I} + 2\mu_g \mathbf{D} \) and \( \lambda_l (\nabla \cdot \mathbf{u}) \mathbf{I} + 2\mu_l \mathbf{D} \), respectively. Notice that the energy equation in the gas phase is exactly equivalent to its counterpart in system (1.22) and was just written in a different form for consistency to the original presentation found in [Dar+10]. Indeed, if we multiply the equation by \( \rho_g c_p \) and move the last term to the right hand side we have

\[
\rho_g c_p \frac{\partial T}{\partial t} + \rho_g c_p \mathbf{u} \cdot \nabla T - \nabla \cdot (k \nabla T) = \rho_g c_p \frac{\gamma - 1}{\gamma} \frac{T}{p} \left( \frac{\partial p}{\partial t} + \mathbf{u} \cdot \nabla p \right),
\]

which reverts back to the energy equation in (1.22) once we notice that

\[
\rho_g c_p \frac{\gamma - 1}{\gamma} \frac{T}{p} = \rho_g r_T \frac{T}{p} = 1.
\]

Let us then consider \( M \in \mathbb{N}_+ \) gaseous inclusion in \( \Omega \) (which will be denoted by \( \Omega_j \)). Unlike the models presented in the previous sections so far, in this case a distinct signed distance function (and thus a distinct Heaviside function) will be used for each
gas bubble:

\[ H_j(x, t) = \begin{cases} 
1 & \text{in the } j\text{-th bubble}, \\
0 & \text{otherwise}, 
\end{cases} \quad j = 1, 2, \ldots, M. \]

The global Heaviside function \( H(x, t) \) is then computed as

\[ H(x, t) = \sum_{j=1}^{M} H_j(x, t). \]

This is where this model parts ways with the one introduced in section 1.3.2. Indeed, as seen in section 1.3.1, supposing a sufficiently low Mach number (and this is the case in the real-world simulations we are interested in as well as in all of the test cases shown in this work), pressure in each bubble can be split into two separate terms (hence the name of this section): a hydrodynamic pressure \( q(x, t) \) and a thermodynamic pressure \( P(t) \), the latter being constant in space. See [Ala05], [Dar+10] and the references therein (in particular [LMP92] and [Pao82]) as well as section 1.3.1 for further details on the topic. This prompts us to split the pressure in the liquid phase in a similar fashion, in order to still be able to have a unified formulation in the end.

As a consequence, the splitting of the pressure in the whole domain is such that:

- \( P \) is uniform in space within each gas bubble and its value is denoted by \( P_j \), \( j = 1, 2, \ldots, M \) (but at the same time \( P_j \) might be different from \( P_i \), for \( i \neq j \));
- \( P \) is extended harmonically in the liquid;
- the total pressure is given by \( p(x, t) = q(x, t) + P(x, t) \), \( \forall (x, t) \in \Omega \times [t_0, t_f] \);
- the density in the gas phase only depends on the thermodynamic pressure - which is the same as saying that the constitutive equation in (1.2) should be changed to

\[ P = \rho rT. \quad (1.24) \]

Let us then rearrange the continuity equation to make use of this new variable that was just introduced. It is clear that

\[ \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \]

is equivalent to

\[ \nabla \cdot \mathbf{u} = -\frac{1}{\rho} \left( \frac{\partial \rho}{\partial t} + \mathbf{u} \cdot \nabla \rho \right). \quad (1.25) \]

Using (1.24), equation (1.25) in the gas phase becomes:

\[ \nabla \cdot \mathbf{u} = -\frac{rT}{P} \frac{1}{rT} \left( \frac{\partial P}{\partial t} + \mathbf{u} \cdot \nabla P \right) = -\frac{rT}{P} \frac{P}{r} \left( \frac{1}{T^2} \right) \left( \frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T \right). \]
Recalling that $P$ is uniform in space, we get to the final form for the continuity equation:

$$\nabla \cdot \mathbf{u} = H(\mathbf{x}, t) \frac{1}{T} \left( \frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T \right) - \sum_{j=1}^{M} H_j(\mathbf{x}, t) \frac{1}{P_j} \frac{dP_j}{dt}.$$ 

Notice that the presence of the Heaviside functions turns off the extra terms in the liquid phase, where the standard incompressibility constraint is enforced.

This formulation of the continuity equation also gives us a way to compute the term $\frac{1}{P_j} \frac{dP_j}{dt}, j = 1, \ldots, M$. Indeed, by taking the integral over $\Omega_j$ (which is equivalent to multiplying by $H_j$ and then taking the integral over the whole $\Omega$), the continuity equation gives:

$$\frac{1}{P_j} \frac{dP_j}{dt} = \frac{1}{\Omega_j} \int_{\Omega_j} H_j \frac{1}{T} \left( \frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T \right) d\mathbf{x} - \int_{\Omega_j} H_j \nabla \cdot \mathbf{u} d\mathbf{x},$$

which will then be used to compute $P_j(t)$ as:

$$P_j(t) = P_j(t_0) \exp \int_{t_0}^{t} \frac{1}{P_j} \frac{dP_j}{dt} dt.$$ 

For the sake of simplicity, from here on after, we will use the following notation:

$$D_{P_j} := \frac{1}{P_j} \frac{dP_j}{dt},$$

$$D_P := \sum_{j=1}^{M} H_j(\mathbf{x}, t) \frac{1}{P_j} \frac{dP_j}{dt}.$$ 

These two terms represent the relative variation of the thermodynamic pressure within the $j$-th bubble and globally, respectively, and, as such, they measure the rate of variation of $P$ with respect to its current value. Notice that, in an isothermal setting, they also represent the rate of change of the volume of the bubbles, since the divergence of the velocity is equal to $D_P$ in that case.

Finally, the thermodynamic pressure $P$ can be computed by solving the following differential problem:

$$\begin{cases} 
\frac{1}{\eta^2} P \cdot H(\mathbf{x}, t) + (1 - H(\mathbf{x}, t)) \Delta P = \sum_{j=1}^{M} H_j(\mathbf{x}, t) \frac{1}{\eta^2} P_j & \text{in } \Omega \times (t_0, t_f], \\
\nabla P \cdot \mathbf{\nu} = 0 & \text{on } \partial \Omega \times (t_0, t_f]. 
\end{cases}$$

Notice that, as mentioned above, the solution of this problem will yield a function that is equal to the given constant pressure $P_j$ in each bubble and that is then extended harmonically in the liquid.
Recalling, once again, that surface tension $s(x, t)$ must be added to the right hand side of the two-phase momentum equation, we are now ready to write the complete unified model:

$$
\begin{aligned}
\nabla \cdot \mathbf{u} &= H(x, t) \frac{1}{T} \left( \frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T \right) - \sum_{j=1}^{M} H_j \frac{1}{P_j} \frac{dP_j}{dt} \quad \text{in } \Omega \times (t_0, t_f],

\rho \frac{\partial \mathbf{u}}{\partial t} + \rho (\mathbf{u} \cdot \nabla) \mathbf{u} - \nabla \cdot \mathbf{\tau} + \nabla q + \nabla P &= \rho \mathbf{g} + \mathbf{s} \quad \text{in } \Omega \times (t_0, t_f],

\frac{1}{\eta^2} P \cdot H(x, t) + (1 - H(x, t)) \Delta P &= \frac{1}{\eta^2} \sum_{j=1}^{M} H_j \frac{1}{P_j} \frac{dP_j}{dt} \quad \text{in } \Omega \times (t_0, t_f],

\frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T - \frac{1}{\rho c_p} \nabla \cdot (k \nabla T) + \\
&\quad - \frac{\gamma - 1}{\gamma} \frac{1}{T} \sum_{j=1}^{M} H_j \frac{1}{P_j} \frac{dP_j}{dt} = 0 \quad \text{in } \Omega \times (t_0, t_f],

\frac{\partial \varphi}{\partial t} + \mathbf{u} \cdot \nabla \varphi &= 0 \quad \text{in } \Omega \times (t_0, t_f],

H_j &= H(\varphi_j) = \begin{cases} 
1 & \text{if } \varphi_j \leqslant 0 \\
0 & \text{if } \varphi_j > 0
\end{cases} \quad \text{in } \Omega \times (t_0, t_f],

H &= \sum_{j=1}^{M} H_j \quad \text{in } \Omega \times (t_0, t_f], \quad (1.27)

\rho &= \sum_{j=1}^{M} \frac{P_j}{rT} H_j + \rho_1 (1 - H) \quad \text{in } \Omega \times (t_0, t_f],

\mu &= \mu_g H + \mu_1 (1 - H) \quad \text{in } \Omega \times (t_0, t_f],

\lambda &= \lambda_g H + \lambda_1 (1 - H) \quad \text{in } \Omega \times (t_0, t_f],

c_p &= c_{p_g} H + c_{p_1} (1 - H) \quad \text{in } \Omega \times (t_0, t_f],

k &= k_g H + k_1 (1 - H) \quad \text{in } \Omega \times (t_0, t_f],

D_{P_j} &= \frac{1}{\sum_{j=1}^{M} H_j} \int_{\Omega} H_j \frac{1}{T} \left( \frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T \right) d\mathbf{x} + \\
&\quad - \int_{\Omega} H_j \nabla \cdot \mathbf{u} d\mathbf{x} \quad j = 1, \ldots, M \quad \text{in } (t_0, t_f],

P_j(t) &= P_j(t_0) \exp \int_{t_0}^{t} \frac{1}{P_j} \frac{dP_j}{dt} dt \quad j = 1, \ldots, M \quad \text{in } (t_0, t_f],

\text{boundary conditions} \quad \text{on } \partial \Omega \times (t_0, t_f],

\text{initial conditions} \quad \text{in } \Omega,
\end{aligned}
$$

### 1.3.3.2 Space-time discretization

Let us now briefly discuss the space-time discretization of system (1.27). As in section 1.3.2.2, the notation and definitions introduced in section 1.2.3 will be used,
and the following set of boundary conditions is supposed to hold:

\[
\begin{align*}
\mathbf{u} &= \mathbf{u}_D \quad \text{on } B_D \times (t_0, t_f], \\
-p\mathbf{v} + \mathbf{f} &= \mathbf{u}_N \quad \text{on } B_N \times (t_0, t_f], \\
T &= T_D \quad \text{on } \Gamma_D \times (t_0, t_f], \\
k\nabla T \cdot \mathbf{v} &= T_N \quad \text{on } \Gamma_N \times (t_0, t_f], \\
\varphi &= \varphi_D \quad \text{on } \Gamma_m \times (t_0, t_f].
\end{align*}
\]

In this case, though, the time discretization is trickier. Indeed, the heavy coupling of system (1.27) (in particular between the values of \(D_p, T\) and \(\nabla \cdot \mathbf{u}\)) requires a different approach from what we have seen so far. A staggered algorithm will still be used, but, instead of just taking the values at the previous time step as approximations for the quantities at the current time, subiterations will be used within each time step to ensure that the values are all coherent with each other. Furthermore, while in [Dar+10] Finite Differences are used for the discretization of spatial variables, the Finite Element framework is employed here. Because of this, some changes had to be made in the definition of the discrete problems to be solved with respect to what is discussed in [Dar+10].

Let us then present the sequence of problems to be solved at each time step. Given \(m_{\max} \in \mathbb{N}_+\), which denotes the maximum number of subiterations allowed at each time step, \(\varepsilon_t \in \mathbb{R}_+\) and initial conditions \(\varphi^{(0)}_{jh}, \mathbf{u}_h^{(0)}, q_h^{(0)}, P_h^{(0)}, \rho_h^{(0)}, \lambda_h^{(0)}, c_{ph}^{(0)}, \) and \(k_h^{(0)}\), for every \(n = 1, 2, \ldots, N\), the solution process is the following:

1D. Compute the signed distance functions \(\varphi^{(n)}_{jh}, j = 1, \ldots, M\), by solving the following discrete level-set equations:

Find \(\varphi^{(n)}_{jh}\) in \(W_h\) such that for all \(w_h \in W_h\)

\[
\frac{\varphi^{(n)}_{jh}}{\Delta t} w_h \bigg|_E - \sum_{\mathcal{E} \in \mathcal{T}_h} \int_{\mathcal{E}} \varphi^{(n)}_{jh} \mathbf{u}_h^{(n-1)} \cdot \nabla w_h \, dx +
\sum_{\mathcal{E} \in \mathcal{T}_h} \int_{\mathcal{E}} \varphi^{(n)}_{jh} \nabla \cdot \mathbf{u}_h^{(n-1)} w_h \, dx + \sum_{e \in \mathcal{E}_h} \int_e \mathbf{u}_h^{(n-1)} \cdot [w_h] \left\{ \varphi^*_h \right\} \, ds +
\sum_{e \in \mathcal{E}_h} \int_e \left\{ \varphi^{(n)}_{jh} \right\} \mathbf{u}_h^{(n-1)} [w_h] + \frac{1}{2} \left[ \mathbf{u}_h^{(n-1)} \cdot \mathbf{v} \right] \left[ \varphi^{(n)}_{jh} \right] \left[ \varphi^{(n)}_{jh} \right] \, ds =
\frac{\varphi^{(n-1)}_{jh}}{\Delta t} w_h \bigg|_E,
\]

where \(\varphi^*_h = \begin{cases} \varphi_{D,h} & \text{if } \mathbf{u}_h^{(n-1)} \cdot \mathbf{v} > 0, \\
\varphi^{(n)}_{jh} & \text{if } \mathbf{u}_h^{(n-1)} \cdot \mathbf{v} \leq 0. \end{cases}\)

Again, we refer to [DLP06] for more details. Then reinitialize it to preserve the
proper shape and shift it rigidly to preserve the initial mass (see section 1.2.1).

2D. Compute $H_{j_h}^{(n)}$, $j = 1, \ldots, M$, using the smoothed approach (1.19) and $H_{h}^{(n)}$ using equation (1.27)7.

3D. Update the physical parameters $\mu_{h}^{(n)}$, $\lambda_{h}^{(n)}$, $c_{p_h}^{(n)}$ and $k_{h}^{(n)}$, equations (1.27)9 to (1.27)12, respectively.

4D. Set $m = 1$ and let $P_{h}^{(n,m)} = P_{h}^{(n-1)}$, $\rho_{h}^{(n,m)} = \rho_{h}^{(n-1)}$, and $u_{h}^{(n,m)} = u_{h}^{(n-1)}$. Then:

(a) Compute $T^{(n,m)}$, by solving the following discrete problem:

Find $T_{h}^{(n,m)}$ in $Q_h$ such that $T_{h}^{(n,m)}|_{\Gamma_D} = T_{D,h}$ and for all $q_{h} \in Q_h,\Gamma_D$

$$
\int_{\Omega} \rho_{h}^{(n,m)} c_{p_h}^{(n)} T_{h}^{(n,m)} \frac{\Delta t}{\gamma - 1} q_{h} \, dx + \int_{\Omega} \nabla q_{h} \cdot \rho_{h}^{(n,m)} c_{p_h}^{(n)} u_{h}^{(n,m)} \, dx + \int_{\Omega} k_{h}^{(n)} \nabla T_{h}^{(n,m)} \cdot \nabla q_{h} \, dx - \int_{\partial \Omega} \rho_{h}^{(n,m)} c_{p_h}^{(n)} \gamma - 1 \frac{\Delta t}{\gamma} T_{h}^{(n,m)} D_{P_{h}}^{(n,m)} q_{h} \, dx = \int_{\Omega} \rho_{h}^{(n,m)} c_{p_h}^{(n)} T_{h}^{(n-1)} \frac{\Delta t}{\gamma - 1} q_{h} \, dx + \int_{\partial \Omega} T_{N,h} q_{h} \, dx.
$$

(b) Compute $D_{P_{h}}^{(n,m)}$, $j = 1, \ldots, M$ using the following algebraic relations:

(i)

$$
D_{P_{h}}^{(n,m)} = \frac{1}{J_{\Omega} \left( H_{j_h}^{(n)} \right)^2} \left( - \int_{\Omega} H_{j_h}^{(n)} \nabla \cdot u_{h}^{(n,m)} \, dx + \int_{\Omega} \left( H_{j_h}^{(n)} \right)^2 \frac{1}{T_{h}^{(n,m)}} \left( \frac{T_{h}^{(n,m)} - T_{h}^{(n-1)}}{\Delta t} + u_{h}^{(n,m)} \cdot \nabla T_{h}^{(n,m)} \right) \, dx \right). \tag{1.28}
$$

(ii)

$$
P_{j_h}^{(n,m)} = P_{j_h}^{(n-1)} \exp \left[ \frac{\Delta t}{2} \left( D_{P_{h}}^{(n,m)} + D_{P_{h}}^{(n-1)} \right) \right].
$$

(c) Compute $\rho_{h}^{(n,m)}$ using equation (1.27)8.

(d) Compute $P_{h}^{(n,m)}$, by solving the following discrete problem:

Find $P_{h}^{(n,m)}$ in $Q_h$ such that for all $q_{h} \in Q_h$

$$
\int_{\Omega} \frac{1}{\eta^2} P_{h}^{(n,m)} H_{h}^{(n)} q_{h} \, dx + \int_{\Omega} \nabla P_{h}^{(n,m)} \nabla q_{h} \left( 1 - H_{j_h}^{(n)} \right) \, dx + \int_{\Omega} \nabla P_{h}^{(n,m)} \nabla (1 - H) q_{h} \, dx = \int_{\Omega} \sum_{j=1}^{M} H_{j_h}^{(n)} \frac{1}{\eta^2} P_{j_h}^{(n,m)} q_{h} \, dx.
$$

(e) Compute $u^{(n,m)}$ and $q^{(n,k+1)}$, by solving the following discrete problem:
CHAPTER 1. MATHEMATICAL MODELS FOR TWO-PHASE FLOWS

Find the pair \( \left( u_h^{(n,m)}, q_h^{(n,m)} \right) \) in \( V^d \times Q_h \) such that \( u_h^{(n,m)} |_{B_D} = u_{D,h} \) and for all pairs \((v_h, \pi_h) \in [V_h, B_D]^d \times Q_h\)

\[
\int_{\Omega} \frac{\rho_h^{(n,m)}}{\Delta t} u_h^{(n,m)} \cdot v_h \, dx + \int_{\Omega} \left( \rho_h^{(n,m)} \left( u_h^{(n,m)} \cdot \nabla \right) u_h^{(n-1)} \cdot v_h \, dx + 
\right.
\]

\[
+ \int_{\Omega} \tau_h^{(n,m)} \cdot \nabla v_h \, dx - \int_{\Omega} q_h^{(n,m)} \nabla \cdot v_h \, dx + \int_{\Omega} \nabla \cdot u_h^{(n,m)} \pi_h \, dx = 
\]

\[
= \int_{\Omega} \left( \frac{\rho_h^{(n,m)}}{\Delta t} u_h^{(n-1)} \cdot v_h \, dx - \int_{\Omega} \nabla P_h^{(n,m)} \cdot v_h \, dx + \int_{\Omega} \rho_h^{(n,m)} g \cdot v_h \, dx + 
\right.
\]

\[
- \int_{\Omega} D_{P_h}^{(n,m)} \pi_h \, dx + \int_{\Omega} \frac{1}{D_h^{(n,m)}} \left( \frac{\partial T_h^{(n,m)}}{\partial t} + u_h^{(n,m)} \cdot \nabla T_h^{(n,m)} \right) \pi_h \, dx + 
\]

\[
+ \int_{\Omega} s_h^{(n)} \cdot v_h \, dx + \int_{B_N} u_{N,h} \cdot v_h \, dx,
\]

with

\[
\tau_h^{(n,m)} = \lambda_h^{(n)} \nabla \cdot u_h^{(n,m)} \mathbf{I} + 2\mu_h^{(n)} \frac{\nabla u_h^{(n,m)} + \left( \nabla u_h^{(n,m)} \right)^t}{2}. \]

Notice that if \( B_D = \partial \Omega \) the hydrodynamic pressure \( q_h^{(n,m)} \) actually has to be sought for in \( Q_h^0 \) instead of \( Q_h \).

(f) Set \( m = m + 1 \)

(g) if the relative increment norm of any of the quantities involved in the subiterations is greater than \( \varepsilon_t \) and \( m < m_{\text{max}} \) go back to 4D(a). Otherwise, go to 5D.

5D. If \( n < N \), set \( n = n + 1 \) and go back to 1D.

Note that the computation of \( D_{P_h}^{(n,m)} \) is different from what one would find in [Dar+10]. This is due to the fact that, as mentioned above, we are employing a Finite Element framework for the space discretization. Indeed, the (local) incompressibility constraint on \( u_h^{(n,m)} \) is imposed in the Navier-Stokes system with the following weak equation:

\[
\int_{\Omega} \nabla \cdot u_h^{(n,m)} w \, dx = \int_{\Omega} H_h^{(n)} \frac{1}{T_h^{(n,m)}} \left( \frac{\partial T_h^{(n,m)}}{\partial t} + u_h^{(n,m)} \cdot \nabla T_h^{(n,m)} \right) w \, dx + 
\]

\[
- \int_{\Omega} D_{P_h}^{(n,m)} w \, dx \quad \forall w \in L^2(\Omega),
\]

which means that equation (1.26) no longer holds true. In other words, (1.26) is only valid in a strong sense. Let us then look for its weak sense counterpart. By taking
1.3. COMPRESSIBLE-INCOMPRESSIBLE MODELS

\[ w = H_j^{(n)} \], we get:

\[
\int_{\Omega} \nabla \cdot u^{(n,m)} H_j^{(n)} \, dx = \int_{\Omega} H_j^{(n)} \frac{1}{T^{(n,m)}} \left( \frac{\partial T^{(n,m)}}{\partial t} + u^{(n,m)} \cdot \nabla T^{(n,m)} \right) H_j^{(n)} \, dx + \int_{\Omega} D_{P_i}^{(n,m)} H_j^{(n)} \, dx,
\]

which, since

\[ H_j^{(n)} = H_j^{(n)} \]

and

\[ D_{P_i}^{(n,m)} H_j^{(n)} = \sum_i D_{P_i}^{(n,m)} H_i^{(n)} H_j^{(n)} = D_{P_j}^{(n,m)} \left( H_j^{(n)} \right)^2, \]

means:

\[
- \int_{\Omega} \nabla \cdot u^{(n,m)} H_j^{(n)} \, dx + \int_{\Omega} \frac{1}{T^{(n,m)}} \left( \frac{\partial T^{(n,m)}}{\partial t} + u^{(n,m)} \cdot \nabla T^{(n,m)} \right) H_j^{(n)} \, dx = \int_{\Omega} D_{P_j}^{(n,m)} \left( H_j^{(n)} \right)^2 \, dx.
\]

Recalling that \( D_{P_j}^{(n,m)} \) is uniform in space, the equation above, once discretized in space, yields (1.28).

1.3.4 Numerical results

An assessment of the compressible-incompressible solvers developed for this work is now presented, and the two compressible-incompressible models introduced in sections 1.3.2 and 1.3.3 are compared. To do so, test cases taken from both [BGN11] and [Dar+10] are solved with the two compressible-incompressible models previously introduced. In particular, we are interested in investigating whether the simpler system (1.23) can give solutions which are sufficiently close to what system (1.27) would compute (and to analytical solutions, for the cases for which one is available). In all of the test cases, we will use the specific gas constant of air, namely \( r = 278.058 \). Once again, all dimensional quantities are expressed in the International System of Units.

1.3.4.1 Test case 1: oscillating water column

This first test case is taken from [BGN11] and deals with the isothermal evolution of two 1D bubbles. The energy equation is therefore not considered. Figure 1.6 shows the set up for this case. A liquid slab is placed between two monodimensional gas bubbles in a closed domain of length \( L = 1 \). The liquid layer is initially situated between \( x = 0.1 \) and \( x = 0.9 \). Gravity is assumed to be \( g = -10 \). The initial time is \( t_0 = 0 \) and the final time is \( t_f = 1 \). The liquid has density \( \rho_l = 1000 \) and viscosity \( \mu_l = 1 \), while the gas has initial density \( \rho_g (t_0) = 1 \) and viscosity \( \mu_g = 1 \times 10^{-5} \). The
initial pressure is $1 \times 10^5$. Note that this is the initial value for the pressure $p(x, t)$ in system (1.23) and for the thermodynamic pressure $P(x, t)$ in system (1.27). In this latter case, the initial hydrodynamic pressure is set to zero. The domain is subdivided in $N = 800$ mesh elements, and time steps equal to $1 \times 10^{-4}$ and $1 \times 10^{-5}$ will be considered.

The quantities of interest for this case are the so-called pressure coefficients, that is the relative variation of the pressure at the domain boundary:

$$
\delta p(x^*, t) = \begin{cases} 
\frac{p(x^*, t) - p(x^*, 0)}{p(x^*, 0)} & \text{in the single-pressure method,} \\
\frac{P(x^*, t) - P(x^*, 0)}{P(x^*, 0)} & \text{in the split-pressure method.}
\end{cases}
$$

In particular, we will denote by $\delta p_1$ and $\delta p_2$ the quantity $\delta p$ computed in $x^* = 0$ and $x^* = 1$, respectively. They are represented in figure 1.7. The results in [BGN11] show that, because of the physical parameters considered, there should be no significant dampening of the oscillation in the time window considered. Comparing the top and bottom plots in figure 1.7, it appears quite clear that, while both models achieve the proper results with the finer time discretization, the single-pressure model becomes too diffusive with the coarser time step, leading to a dampened oscillation with a decrease in peak value for both pressure coefficients.

1.3.4.2 Test case 2: non-isothermal water column

The second test case is taken from [Dar+10]. The domain is still monodimensional, but, this time around, temperature is introduced. See figure 1.8 for a graphical representation of the case. Two gas bubbles enclose a liquid slab of length $1 \times 10^{-5}$, initially situated in the middle of the domain, which has length $L = 1 \times 10^{-4}$. The physical parameters are:

- $\rho_l = 1000$
- $\mu_l = 1 \times 10^{-3}$, $\mu_g = 1.82 \times 10^{-5}$,
- $k_l = 0.6$, $k_g = 0.0256$,
1.3. COMPRESSIBLE-INCOMPRESSIBLE MODELS

Figure 1.7: First compressible-incompressible test case: pressures coefficients at the domain boundary

Figure 1.8: Initial configuration for the second compressible-incompressible test case
\( c_p^l = 4184, \ c_p^g = 1004.5, \)
\[ \gamma = 1.4 \]

Initial pressure (that is, thermodynamic pressure for the split-pressure method) and temperature are \( p_0 = 101325 \) and \( T_0 = 293.15 \), respectively. The left wall is then heated to \( T_w = 373.15 \), while keeping the right wall insulated. The final theoretical state has uniform temperature \( T_w \) and uniform pressure \( p_f = p_0 \frac{T_w}{T_0} = 128976.37 \).

Final time is \( t_f = 0.4 \), while the time step is set to \( 1 \times 10^{-8} \). The domain is discretized using \( N = 100 \) mesh elements.

In figure 1.9 the pressure values \( p_1 = p(x = 0, t) \) and \( p_2 = p(x = 10^{-4}, t) \), computed using both models, are shown. The two plots make use of a logarithmic scale along the \( x \) axis in order to better show the fast transient dynamics that takes place in the very first few milliseconds. First of all, notice that the predicted theoretical pressure is indeed recovered once the new equilibrium is reached, independently of the model used. Secondly, let us turn our attention to the transient behaviour in the initial phase of the simulation. As explained in [Dar+10] and the references therein, the oscillatory behaviour is to be expected in this case, and it is due to gas compressibility. Both models do recover the same qualitative results, if we ignore the very first few time steps, where the effects of the non-consistent initial condition affect the single-pressure model more than they do the split-pressure. Nonetheless, as in the test case in section 1.3.4.1, the single-pressure method proves to be more diffusive than the split-pressure one, leading to a shorter transient in the former case with respect to the latter.

1.3.4.3 Test case 3: a 2D isothermal compression

We now move on to a 2D case taken from [BGN11] and represented in figure 1.10. We consider a \([0,1] \times [0,1]\) domain in which a gas bubble lies initially at rest, surrounded by liquid. The bubble has center in \((0.5,0.5)\) and radius \( r = 0.25 \). The initial pressure is \( p_0 = 1 \times 10^5 \) and the initial gas density is \( \rho_g = 1 \). The other physical parameters are: \( \rho_l = 1000 \), \( \mu_l = 1 \) and \( \mu_g = 0.01 \). The domain’s boundary is closed, except for the section \([y_l,y_u] = [0.25,0.75]\) on the left side. There, liquid is injected horizontally with a parabolic inflow, whose law is given by:

\[
    u_{in}(y,t) = 100t(y - 0.75)(0.25 - y) \quad \forall (y,t) \in [y_l,y_u] \times [t_0,t_f],
\]

with \( t_0 = 0 \) and \( t_f = 0.4 \). The total inflow volume can then be computed as:

\[
    V_{in}(t) = \int_{t_0}^{t} \int_{y_l}^{y_u} u_{in}(y,\bar{t}) \,dy\,d\bar{t} = \frac{100}{96} t^2,
\]
Figure 1.9: Second compressible-incompressible test case: pressures at the domain boundary.
hence the gas volume is given by:

$$V_g(t) = V_g(0) - V_i n(t) = \frac{\pi}{16} - \frac{100}{96} t^2.$$  

The computational grid has $N = 150$ intervals on each side of the domain. The time steps chosen are $\Delta t = 5 \times 10^{-3}$, $2 \times 10^{-3}$ and $1 \times 10^{-3}$.

For both models, the bubble’s position and volume, obtained with the different time steps used, are shown in figures 1.11 and figures 1.12 (respectively). Figures 1.13 and 1.14, on the other hand, show a comparison on such quantities between the two models when the finer time step is used.

Figures 1.11 and 1.12 show that indeed the results have converged as far as the time step used is concerned, while figures 1.13 and 1.14 indicate that there is good agreement between the results obtained with the two models being investigated.
Figure 1.11: Third compressible-incompressible test case: bubble’s position. Top row: results obtained with the single pressure method. Bottom row: results obtained using the split-pressure method. The time steps used are 0.005, 0.002 and 0.001 (from lightest to darkest line)
Figure 1.12: Third compressible-incompressible test case: time convergence of the bubble’s volume

Figure 1.13: Third compressible-incompressible test case: comparison between the bubble’s positions obtained using the single-pressure model (in blue) and the split-pressure model (in red)
Figure 1.14: Third compressible-incompressible test case: comparison between the bubble’s volumes obtained using the two compressible-incompressible methods introduced.
1.3.5 Final comparison of the two models

Finally, we present a closing comparison of the compressible-incompressible models introduced in the previous sections, highlighting both similarities and key differences between the two approaches, which led us to choose one over the other for the remainder of this work.

The main, and most obvious, similarity is that they both are single-domain formulations, which means that the same set of variables and equations is used throughout the whole domain. This is made possible by two significant choices taken upstream in the modelization process, namely the use of a signed distance function to identify the two phases within the domain and the use of the same set of unknowns for the incompressible and the compressible models. The latter, in turn, can be traced back to section 1.1, where the general model was introduced and tweaked in order to rearrange it in a fashion that would later prove suitable for both kinds of flows. While this is not a novelty of this work, it is nonetheless a decision that we had to make early on, since we could have adopted one of the other approaches that have been proposed in the literature, as discussed in the introduction (see e.g. [APF13], [CPR05], [CCR12], [CFA01] and [LSR16]). The use of a single-domain formulation is particularly important, since it simplifies both the derivation of the weak and discrete problems and the implementation of the code to solve the resulting algebraic systems. Furthermore, a formulation employing different variables and different equations in the two phases would have been much more complex to use in the framework of optimal control problems (see chapter 3).

The main difference clearly lies in the splitting of the pressure term. This has some interesting consequences, especially in terms of accuracy when the time step grows larger. As discussed in section 1.3.1, and as seen in the numerical experiments in section 1.3.4, the split-pressure method allows for bigger time steps, with respect to the single-pressure method, without loss of accuracy. However, the complexity of the split-pressure approach would have been a severe obstacle to overcome in the definition of all the mathematical entities needed to solve the optimal control problem we are interested in (see section 3.2). In particular, the fact that the split-pressure method employs multiple signed distance functions (one for each bubble) proves detrimental to the derivation and solution of the optimal control problem. Indeed, as we will see in section 3.2, our interest lies in having the final signed distance \( \varphi(x, t_f) \) match a given target configuration \( \bar{\varphi} \). This could be problematic in the case of splitting and merging of bubbles, if the split-pressure method is used. As a matter of fact, while the treatment of changes in the topology of the gas phase is straightforward in the single-pressure method, it requires special care when the split-pressure method is employed, as one need needs to change the number of signed distance functions considered, as well as manage the computation of the thermodynamic pressure in the
1.3. COMPRESSIBLE-INCOMPRESSIBLE MODELS

bubbles involved. This is why we chose to trade the possibility to have an accurate solution using a comparatively large time step for the chance to have a much simpler model to work with, and therefore the single-pressure model is used in the remainder of this work, and especially for the optimal control problem defined in section 3.2.

1.3.6 A compressible rising bubble

Lastly, let us consider the case of a compressible rising bubble. In order to allow the bubble to expand or contract, an open domain, with a Neumann condition imposed on the top boundary, is considered. Simulations have been performed with the single-pressure method from section 1.3.2, and in this way the chosen model has been tested with mixed boundary conditions for the velocity as well.

This test case was studied as a first extension of the results presented in section 1.2.4, in view of the construction of a future compressible rising bubble benchmark. Figure 1.15 shows the initial configuration for the case investigated in this section. Notice that it is quite close to what was used as initial configuration in section 1.2.4 (see figure 1.1), but in this case a Neumann condition is used for the top wall. Indeed, if this were not the case, the bubble would not be able to change its volume, as it would be confined in a closed domain filled with incompressible liquid, which has a constant volume. The value for the Neumann condition is $p_b = 101325$, i.e. atmospheric pressure.

On the one hand, in order for the test case to be as similar as possible to the incompressible-incompressible benchmark, we want the flow to be in the same regime as in section 1.2.4.1. On the other hand, real values for the viscosity and density of an air-water configuration need to be used, in order for the test to be of physical significance. In [Hys+09], two adimensional quantities are used to characterize the flow regime. These are the Reynolds number and the Eötvös number, which are given by

$$
Re = \frac{\rho_1 U_g L}{\mu_1}, \quad \text{Eo} = \frac{\rho_1 U_g^2 L}{\varsigma},
$$

respectively. In the definitions (1.29), $\rho_1$ and $\mu_1$ are the density and the viscosity of the surrounding fluid (in this case, water), $\varsigma$ is the surface tension coefficient and $L$ and $U_g$ are the characteristic length and velocity scales, defined as $L = 2r_0$, $U_g = \sqrt{2gr_0}$, $r_0$ and $g$ being the initial bubble radius and the gravitational acceleration, respectively. In the benchmark in section 1.2.4.1, the chosen physical parameters yield $Re = 35$ and $\text{Eo} = 10$.

In order to keep the same values for the adimensional quantities while using the physical values of $\rho_l = 1000$ and $\mu_l = 0.001$ for the liquid and $\mu_g = 1 \times 10^{-5}$ for the gas (whose density, on the other hand, is not constant, since it depends on temperature and pressure), the bubble size had to be rescaled with respect to the one used in section 1.2.4.1. In particular, the initial radius of the bubble is $r_0 = 2.5 \times 10^{-4}$,
Figure 1.15: Initial configuration for the compressible rising bubble
while the final time and the time step are $t_f = 0.05$ and $\Delta t = 6.25 \times 10^{-5}$, respectively. The surface tension coefficient is $\zeta = 2.45 \times 10^{-4}$ and the gravitational acceleration is $\mathbf{g} = (0, -9.8)$. Since the bubble is much smaller, we place its center at $(1 \times 10^{-3}, 5 \times 10^{-4})$ and we consider a domain with sides $L_1 = 0.002$, $L_2 = 0.003$, discretized with $N_x = 160$ elements along the $x$ direction and $N_y = 240$ elements along the $y$ direction, so the mesh size is $h = 1.25 \times 10^{-5}$. Notice that the domain sides ratio is different than that in section 1.2.4. This allows the bubble to expand without being constrained by the boundary. Still, the domain size does not affect the adimensional quantities value, so this has no effect on the flow regime. The temperature is set to $T = 470$ and the initial gas density is $\rho_0 = 1$, so that the initial pressure inside the bubble is $p_0 = \rho_0 r T \approx 135000 > p_b$.

The quantities of interest, in this case, are the position of the center of mass, the rise velocity and the surface of the bubble. The first two were used in the incompressible-incompressible benchmark as well, and are defined in section 1.2.4. The last one is specific to the compressible-incompressible test case.

Figures 1.16 and 1.17 show the position of the bubble at selected time steps and the evolution of the quantities of interest. Unlike figure 1.17a, the plots in figures 1.17b and 1.17c use a logarithmic scale on the $x$ axis, since most of the variation to the value of the quantities shown happens within the first time steps. Indeed, as seen in figure 1.17c, the bubble, which has higher pressure than its surroundings, grows to the proper size almost instantaneously, after a quick oscillatory transient due to the inertia of the expansion. The volume is basically constant after that. As a matter of fact, the domain is too small to have any significant difference between top and bottom in the value of the hydrostatic pressure, so the bubble does not expand much further while rising up.
Figure 1.16: Bubble’s position for the compressible rising bubble
Figure 1.17: Benchmark quantities for the compressible rising bubble
Chapter 2

Abstract framework for optimal control problems

In this chapter optimal control problems will be introduced from a theoretical point of view, providing the necessary definitions and the main results in a general framework. This introduction on the theoretical aspects of optimal control problems will just provide the basic notions needed in the remainder of this work (especially in chapter 3), as an in-depth presentation on the general theory of control problems goes beyond the scope of this thesis. The reader is referred to [Trö10], [Lio71] and [Gun03] (among others) for further details on the subject.

2.1 Introductory definitions

The basic notions needed to develop the theory on optimal control problems will now be quickly introduced, so that we may then build upon them in the following sections.

2.1.1 Basics of differential calculus in Banach spaces

Let us start by giving some preliminary definitions that will be used in the following sections. For greater details on the subject, we refer e.g. to [Bre10].

**Definition 2.1** (Dual space). *Let $E$ be a normed vector space over $\mathbb{R}$. The space of all continuous linear functionals $\phi : E \to \mathbb{R}$ is called dual space and it will be denoted by $E^*$. Furthermore, the dual space of $E^*$, $E^{**}$, is called bidual space.*

The canonical injection is now defined as follows.

**Definition 2.2** (Canonical injection). *Given $x \in E$, $f \in E^*$, the canonical injection*
I is the linear map from \( E \) to \( E^{**} \) defined as:
\[
I : E \to E^{**} \\
x \mapsto Ix,
\]
where \( Ix \) is the element of \( E^{**} \) such that
\[
\langle Ix, f \rangle_{E^{**}, E^*} = \langle f, x \rangle_{E^*, E}.
\]

The canonical injection \( I \) is linear, but it may not always be surjective. The following definition deals with the cases where \( I \) is indeed a bijection.

**Definition 2.3 (Reflexive Banach space).** Let \( E \) be a Banach space, i.e. a complete normed vector space and let \( I : E \to E^{**} \) be the canonical injection from \( E \) to \( E^{**} \). Then \( E \) is said to be reflexive if \( I \) is surjective, that is \( I(E) = E^{**} \). In this case, \( E^{**} \) is usually identified with \( E \).

Let us now consider the topic of differentiation in normed vector spaces. Let \( X \) and \( Y \) be two Banach spaces and \( F : U \subseteq X \to Y \) be a function from the open subset \( U \) of \( X \) into \( Y \).

**Definition 2.4 (Gâteaux differential).** We say that \( F \) is Gâteaux-differentiable in \( x_0 \in U \) if \( \exists A \in \mathcal{L}(X, Y) \) (i.e. the vector space of continuous linear functionals from \( X \) to \( Y \)) called Gâteaux differential of \( F \) in \( x_0 \), such that, for all \( h \in X \) such that \( x_0 + h \in U \), it holds that
\[
\lim_{t \to 0} \frac{F(x_0 + th) - F(x_0)}{t} = Ah,
\]
i.e.
\[
\left\| \frac{F(x_0 + th - F(x_0))}{t} - Ah \right\|_Y \to 0 \text{ when } t \to 0.
\]
The notation \( \mathcal{G} \)-differentiable will be used hereinafter for such functions.

If it exists, the Gâteaux differential is unique and it will be denoted by \( F'_G \). Moreover, if \( F \) is \( \mathcal{G} \)-differentiable in every point of \( U \), then
\[
F'_G : U \to \mathcal{L}(X, Y) \\
x \mapsto F'_G(x)
\]
is called Gâteaux derivative of \( F \).

Let now \( F : U \subseteq X \to \mathbb{R} \) be a \( \mathcal{G} \)-differentiable functional on \( U \). This means that \( F'_G(x) \in X^* \ \forall \ x \in U \) and \( F'_G : U \to X^* \).
2.1. INTRODUCTORY DEFINITIONS

Definition 2.5 (Convexity of a functional). Let $U$ be a convex set and $F : U \subseteq X \to \mathbb{R}$. We say that $F$ is convex on $U$ if

$$F(x + t(y - x)) \leq F(x) + t(F(x) - F(y)), \quad \forall x, y \in U, \forall t \in [0, 1].$$

If such inequality holds true with a strict less-than sign for $t \in (0, 1)$, then $F$ is said to be strictly convex.

2.1.2 Optimization in Banach spaces

Let us now consider the problem of minimizing a functional defined on a Banach space. The following theorem deals with the existence and uniqueness of the minimizing point.

Theorem 2.1. Let $U$ be a reflexive Banach space, $U \subseteq X$ be convex and closed and $J : U \to \mathbb{R}$. If:

- $U$ is bounded or $J(x) \to \infty$ for $\|x\|_X \to \infty$
- $J$ is weakly sequentially lower semicontinuous, i.e.:

$$x_j \rightharpoonup x \Rightarrow J(x) \leq \liminf_{j \to \infty} J(x_j)$$

then there exists $x^* \in X$ such that:

$$J(x^*) = \min_{x \in U} J(x).$$

If moreover $J$ is strictly convex, then $x^*$ is unique.

As for optimality conditions, the following extension of Fermat’s theorem holds true.

Theorem 2.2. Let $X$ be a reflexive Banach space and $J$ a $\mathcal{G}$-differentiable functional on $U \subseteq X$, with $U$ convex and closed. Then the following hold:

i) If $J(x^*) = \min_{x \in U} J(x)$, then

$$J'(x^*)(x - x^*) \geq 0, \quad \forall x \in U.$$  \hspace{2cm} (2.1)

ii) If $J$ is convex and equation (2.1) holds true, then $J(x^*) = \min_{x \in U} J(x)$.

A particular case is that of quadratic functionals in Hilbert spaces.
Chapter 2. Abstract Framework for Control Problems

**Theorem 2.3.** Let \( H \) be a Hilbert space with norm \( \| \cdot \| \) and scalar product \( (\cdot, \cdot) \). Let then \( \pi = \pi (u, v) \) be a continuous symmetric bilinear form on \( H \), that is a map such that:

\[ \exists M \in \mathbb{R}^+ \text{ such that } |\pi (u, v)| \leq M \| u \|_H \| v \|_H , \quad \forall \ u, v \in H . \]

Let then \( L \in H^* \). Let us consider a functional

\[ J (u) = \frac{1}{2} \pi (u, u) - Lu . \]

Let finally \( U \subseteq H \) be a convex closed set. Let us assume that one of the following properties holds true:

i) \( \pi \) is coercive, i.e.

\[ \exists \alpha \in \mathbb{R}^+ \text{ such that } \pi (u, u) \geq \alpha \| u \|_H^2 , \quad \forall \ u \in H ; \]

ii) \( \pi \) is positive, i.e.

\[ \pi (u, u) > 0 , \quad \forall \ u \in H, \ u \neq 0 . \]

Then there exist unique \( u^* \in U \) such that:

\[ J (u^*) = \min_{u \in U} J (u) \]

and the variational inequality

\[ \pi (u^*, u - u^*) \geq L (u - u^*) , \quad \forall \ u \in U . \]

holds true.

2.2 General optimal control problems

In this section the necessary concepts for the formulation and solution of a generic optimal control problem will be introduced. Such theoretical framework will be then applied to the case of interest for this work in section 3.2.1

2.2.1 Structure of a control problem

Let us consider a generic physical system that can be controlled through an input variable. The goal of a control problem is to determine the input value that results in a desired value for a given output variable. Obtaining precisely the desired output value is not always possible, though, so the above-mentioned goal is often relaxed, and we simply aim at minimizing the difference (in a suitable sense) between the solution and the desired output.
This problem can be accurately defined by introducing proper mathematical entities. Let us then consider:

- a control function \( g \) in a function space \( \mathcal{G}_{ad} \), called admissible controls space. In general, \( \mathcal{G}_{ad} \subseteq \mathcal{G} \), where \( \mathcal{G} \) is the most suitable function space for \( g \) to live in, based on the role that the control function plays in the equations. Possible additional constraints imposed on \( g \) determine the space \( \mathcal{G}_{ad} \) in which we seek such function;

- the state of the system \( u(g) \), which lies in a suitable function space \( \mathcal{U} \) and depends on \( g \) through the state equation. Such equation is a relation of the form:

\[
Au(g) = f + Bg ,
\]

where \( A: \mathcal{U} \rightarrow \mathcal{U}^* \) is a differential operator (linear or not) and \( B: \mathcal{G} \rightarrow \mathcal{U}^* \). Such problem defines the physical system being controlled;

- the observation function, denoted by \( z(g) \), which depends on the state \( u \) through a suitable operator \( C \):

\[
C: \mathcal{U} \rightarrow \mathcal{Z} \\
u \mapsto z(u) = Cu .
\]

The value of such function will be compared to the desired output value, which will be denoted by \( z_d \);

- a cost functional (or objective functional):

\[
J: \mathcal{G}_{ad} \rightarrow \mathbb{R} \\
g \mapsto J(g, z(g)) ,
\]

with \( J(g, z(g)) \geq 0 \forall g \in \mathcal{G} \).

The goal of the control problem may then be summed up as:

\[
\text{Find } g^* \in \mathcal{G}_{ad} : J(g^*) = \min_{g \in \mathcal{G}} J(g) .
\]

### 2.2.2 Lagrangian formulation of optimal control problems

In general, there exist two possible approaches to an optimal control problem: the Lions approach and the Lagrangian approach. In this work, the latter will be used. Such approach highlights the role of the adjoint variable within the control problem. Further details on the Lions approach can be found in [Lio71] and [Qua16].

The Lagrangian approach expands the Lagrangian multipliers method originally devised for constrained nonlinear optimization in \( \mathbb{R}^n \) (see [WN99]) to the infinite-dimension case. The optimal control problem presented in section 2.2.1 can in fact be
seen as a constrained optimization problem: we are looking for the minimizing point \( g^* \) of a function \( J(g) \), constrained by (2.2). Solving the optimal control problem is then equivalent to seeking the free critical points of the \textit{Lagrangian functional} defined as:

\[
\mathcal{L}(u, p, g) = J(g) + \langle p, f + Bg - Au \rangle_{\mathcal{G}},
\]

where \( p \in \mathcal{P} \) is the \textit{Lagrangian multiplier} (also known as \textit{adjoint variable}). Notice that, in this framework, the variables on which the Lagrangian functional depends must be considered as independent from each other. The problem can then be recast as:

Find \((u, p, g) : \nabla \mathcal{L}(u, p, g) [\delta u, \delta p, \delta g] = 0, \quad \forall (\delta u, \delta p, \delta g) \in \mathcal{U} \times \mathcal{P} \times \mathcal{G},\]

that is

\[
\begin{align*}
\frac{\partial \mathcal{L}}{\partial u}(u, p, g)[\delta u] &= 0, \quad \forall \delta u \in \mathcal{U}, \\
\frac{\partial \mathcal{L}}{\partial p}(u, p, g)[\delta p] &= 0, \quad \forall \delta p \in \mathcal{P}, \\
\frac{\partial \mathcal{L}}{\partial g}(u, p, g)[\delta g] &= 0, \quad \forall \delta g \in \mathcal{G}, 
\end{align*}
\]

where the partial derivatives are to be interpreted as Gâteaux derivatives (see section 2.1.2). The variables inside the square brackets denote the objects on which the partial derivatives act.

Notice that the second equation in system (2.5) corresponds to the state equation (2.2). Indeed, keeping in mind that \( u, p \in g \) are to be considered independent from each other, the derivative of (2.4) with respect to the adjoint variable \( p \) can be written as:

\[
\frac{\partial \mathcal{L}}{\partial p}(u, p, g)[\delta p] = \lim_{t \to 0} \frac{\mathcal{L}(u, p + t \delta p, g) - \mathcal{L}(u, p, g)}{t} = \lim_{t \to 0} \frac{J(g) + \langle p + t \delta p, f + Bg - Au \rangle - J(g) + \langle p, f + Bg - Au \rangle}{t} = \langle \delta p, f + Bg - Au \rangle.
\]

The second equation in (2.5) can then be recast as:

\[
\langle \delta p, f + Bg - Au \rangle = 0, \quad \forall \delta p \in \mathcal{P},
\]

which is clearly equivalent to (2.2).

On the other hand, by computing the derivative of (2.4) with respect to the primal
variable $u$ we have:

$$\frac{\partial \mathcal{L}}{\partial u} (u, p, g) [\delta u] = \lim_{t \to 0} \frac{\mathcal{L} (u + t \delta u, p, g) - \mathcal{L} (u, p, g)}{t} = \lim_{t \to 0} \frac{J (g) + \langle p, f + Bg - A (u + t \delta u) \rangle - J (g) + \langle p, f + Bg - Au \rangle}{t} = \langle p, - \frac{\partial A}{\partial u} [\delta u] \rangle.$$

Let us denote by $\tilde{A}$ the operator defined as:

$$\langle p, - \frac{\partial A}{\partial u} [\delta u] \rangle = \langle \tilde{A} p, \delta u \rangle, \quad \forall \delta u \in \mathcal{U},$$

for a given $p$. The first equation in (2.5) is then equivalent to:

$$\tilde{A} p = 0,$$

which is called adjoint equation.

Let us finally consider the third equation in system (2.5). As discussed in section 2.1.2, we have that $\frac{\partial \mathcal{L}}{\partial g} \in \mathcal{G}^*$. If $\mathcal{G}$ is a Hilbert space, thanks to Riesz representation theorem, we can recast such duality product as a scalar product:

$$\langle \nabla J, \delta g \rangle = 0, \quad \forall \delta g \in \mathcal{G}. \quad (2.6)$$

We then get the expression of $\nabla J$ as an element of $\mathcal{G}$.

The three equations in system (2.5) (or their recast counterparts, namely the primal equation, the adjoint equation and the variational equality (2.6)) are usually collectively known as optimality conditions associated to the control problem (2.3).

### 2.2.3 Solution of the control problem

Direct solution of system (2.5) is often numerically unfeasible, due to its complexity. To overcome this limit, one possible strategy is to resort to an iterative method to approximate the solution of the optimal control problem.

Algorithm 1 details the steps to be followed to iteratively solve a generic optimal control problem. Given an initial guess for the optimal control, the state equation is solved and its solution is used to compute the control increment iteratively, until convergence is reached.

As a convergence criterion, a possible choice is to check the norm of the functional gradient:

$$\| \nabla J (g^{(k)}) \|_\mathcal{G} < \varepsilon_J,$$

where $\varepsilon_J \in \mathbb{R}^+$ is a given constant value. Indeed, the functional gradient theoretically
Algorithm 1 Optimal control solution algorithm

Given an initial guess $g^{(0)}$

$k \leftarrow 0$

while NOT CONVERGED do

Compute $u^{(k)} = u (g^{(k)})$ by solving the state equation

Compute the functional gradient $\nabla J (u^{(k)}, g^{(k)})$

Compute the control increment $\delta g^{(k)}$ based on $\nabla J (u^{(k)}, g^{(k)})$

$g^{(k+1)} \leftarrow g^{(k)} + \delta g^{(k)}$

$k \leftarrow k + 1$

end while

vanishes at a minimizing point. Nonetheless, this could lead to convergence problems in case the functional is quite flat in the neighbourhood of the minimizing point. The convergence test can then be made more robust if the relative increment of the control function is taken into account too, namely if the iterative method stops when:

$$\left\| \nabla J (g^{(k)}) \right\|_g < \varepsilon_J \quad \lor \quad \left\| \frac{g^{(k)} - g^{(k-1)}}{g^{(k-1)}} \right\|_g < \varepsilon_g,$$

with $\varepsilon_g \in \mathbb{R}^+$ given, that is when at least one of the two conditions is fulfilled.

2.2.3.1 Minimization algorithm: the line search method

Let us now detail one of the possible iterative minimization algorithms for nonlinear functionals. For further details on this and other alternative algorithms see [WN99].

One of the possibilities when dealing with unconstrained minimization is the so-called line search method, which reduces the complexity of the problem of seeking the minimum point by only considering a single variability direction at each iteration, albeit different from iteration to iteration.

Let than $g^{(k)}$ hold the value of the control variable at the $k$-th iteration. Given the function

$$\psi (\alpha) = J (g^{(k)} + \alpha d^{(k)}) ,$$

according to the line search method the new value $g^{(k+1)}$ is defined by:

$$g^{(k+1)} = g^{(k)} + \alpha^{(k)} d^{(k)} ,$$

where $d^{(k)}$ is the $k$-th descent direction and $\alpha^{(k)} = \arg\min_{\alpha \in \mathbb{R}^+} \psi (\alpha)$.

The choice of $d^{(k)}$

The search direction $d^{(k)}$ must obviously be a descent direction. In other words, there
Algorithm 2 Simple backtracking

For every $k$, let

- $g^{(k)}$
- $c_1 > 0$
- $c_2 \in (0, 1)$
- $J(g^{(k)})$
- $\nabla J(g^{(k)})$
- $d^{(k)}$

be given.

Set $m = 0$. Let us consider an initial guess $\alpha^{(m)}$

while $J(g^{(k)} + \alpha^{(m)}d^{(k)}) > J(g^{(k)}) + c_1 \alpha^{(m)}(\nabla J(g^{(k)}), d^{(k)})$ do

$\alpha^{(m+1)} = c_2 \alpha^{(m)}$

$m \leftarrow m + 1$

end while

must exist $\delta > 0$ such that

$$J(g^{(k)} + \alpha d^{(k)}) < J(g^{(k)}), \quad \forall \alpha \in (0, \delta).$$

The most obvious choice is taking

$$d^{(k)} = -\nabla J(g^{(k)}).$$

Nonetheless, every direction $d^{(k)}$ such that:

$$\left(d^{(k)}, \nabla J(g^{(k)})\right) < 0$$

(2.7)

(where $(\cdot, \cdot)$ denotes the scalar product) is a descent direction.

The choice of $\alpha^{(k)}$

The step length $\alpha^{(k)}$ is defined by

$$\alpha^{(k)} = \arg\min_{\alpha \in \mathbb{R}^+} \psi(\alpha).$$

However, in real applications it is hardly possible to solve such minimization problem exactly, due to its complexity. Instead, the value of $\alpha^{(k)}$ is computed using an approximated minimization approach. To do so, one possibility is to use a backtracking algorithm (see algorithm 2). Starting from a given initial guess $\alpha^{(0)}$, the final step length $\alpha$ is computed by iteratively multiplying its current value $\alpha^{(k)}$ by a constant $c_2 \in (0, 1)$, thus ever reducing its value, until the convergence condition in the while loop is satisfied. Typical values for the parameters used in algorithm 2 are $\alpha^{(0)} = 1$, $c_1 = 10^{-3}$ e $c_2 = 1/2$. The while loop condition is the negation of the Armijo condition (also known as the sufficient decrease condition), which requires $\alpha^{(m)}$ to be such
Algorithm 3 Optimal control search, \textit{line search} method with backtracking

Let an initial guess $g^{(0)}$ and two positive constants $\varepsilon_g$ and $\varepsilon_J$ be given. 

$k \leftarrow 0$

\[ \frac{\|g^{(k)} - g^{(k-1)}\|}{\|g^{(k-1)}\|} > \varepsilon_g \land \frac{\|\nabla J \left(g^{(k+1)}\right)\|}{\|g^{(k)}\|} > \varepsilon_J \] do

Compute search direction $d^{(k)}$ (e.g.: $d^{(k)} = -\nabla J \left(g^{(k)}\right)$)

Compute step length $\alpha^{(k)}$

\begin{algorithmic}
\State Let an initial guess $\alpha^{(0)}$ and two constants $c_2 \in (0, 1)$ and $c_1 > 0$ be given
\State $m \leftarrow 0$
\While{$J \left(g^{(k)} + \alpha^{(m)}d^{(k)}\right) > J \left(g^{(k)}\right) + c_1\alpha^{(m)} \left(\nabla J \left(g^{(k)}\right), d^{(k)}\right)$}
\State $\alpha^{(m+1)} = c_2\alpha^{(m)}$
\State $m \leftarrow m + 1$
\EndWhile
\State $\alpha^{(k)} = \alpha^{(m)}$
\End\end{algorithmic}

end

\[ \delta g^{(k)} \leftarrow \alpha^{(k)}d^{(k)} \]

\[ g^{(k+1)} \leftarrow g^{(k)} + \delta g^{(k)} \]

$k \leftarrow k + 1$

end while

that:

\[ \psi \left(\alpha^{(m)}\right) \leq \psi \left(0\right) + c_1\alpha^{(m)}\psi' \left(0\right). \]

Such condition can be recast in terms of the objective functional $J$ as:

\[ J \left(g^{(k)} + \alpha^{(m)}d^{(k)}\right) \leq J \left(g^{(k)}\right) + c_1\alpha^{(m)} \left(\nabla J \left(g^{(k)}\right), d^{(k)}\right). \]

Notice that $\psi' \left(0\right) = \left(\nabla J \left(g^{(k)}\right), d^{(k)}\right)$ is actually a negative quantity. This means that the Armijo condition requires the value of $\psi \left(\alpha\right)$ computed in $\alpha = \alpha^{(m)}$ to be below the line $y \left(\alpha\right) = \psi \left(0\right) + c_1\alpha\psi' \left(0\right)$, which has negative slope. For further detail, see [WN99].

The complete algorithm

We can now rewrite algorithm 1, adding the procedures to compute the search direction and the step length we just discussed. Algorithm 3 shows the whole procedure to be followed to find the optimal value for the control variable. Notice that the condition for the outermost \texttt{while} loop checks both the norm of the increment of the control variable and the norm of the functional gradient, but different conditions may be used, as discussed in section 2.2.3.
2.2.4 A comment on the two possible discretization approaches

Up to now, the discretization of the generic partial derivatives equations related to the optimal control problem has not been addressed, but it is clear that such equations cannot be solved exactly. One can then distinguish two different strategies for the solution of a control problem:

- the “differentiate-then-discretize” approach involves obtaining the optimality conditions (2.5) using the strong form of the equations that describe the control problem, and then discretizing such conditions;

- the “discretize-then-differentiate” method, on the other hand, consists of first discretizing the equation in the control problem and then later deriving the optimality conditions for the discretized problem; one then solves the resulting discrete version of system (2.5).

For this work, the former method was used. Nonetheless, it is important to notice that the two approaches may lead to different results, since the discretization step and the differentiation step do not commute. Both approaches have arguments for and against (see for example [Gun00] and [Gun03] for an in-depth analysis), but which one yields the best results is still a topic open for debate.
Chapter 3

Optimal control for compressible-incompressible flows

Using the tools discussed in chapter 2, the optimal control problem of interest in this work will now be introduced. As a preliminary step, a simpler control problem on an advection equation is considered, in order to be able to see the notions presented theoretically in chapter 2 applied to an easier case, before studying the (harder) complete problem.

3.1 An introductory control problem

In this section a first explanatory optimal control problem is discussed, as a preparatory introduction to the control problem of interest of section 3.2. In particular, the same goal as the control problem in section 3.2, namely driving the solution to an advection problem towards a given target, is considered. Let us now formally describe such problem.

3.1.1 Formulation of the control problem

For the sake of simplicity, a monodimensional domain is considered. Let then $\Omega = [a, b] \subset \mathbb{R}$, $[t_0, t_f] \subset \mathbb{R}$ be two closed bounded intervals. The aim of the optimal control problem is to identify the proper advection coefficient $C = C(x)$ in:

\[
\begin{align*}
\frac{\partial \varphi}{\partial t} + C \frac{\partial \varphi}{\partial x} &= 0 & \text{in } \Omega \times (t_0, t_f) \\
\varphi|_{t_0} &= \varphi_0 & \text{in } \Omega
\end{align*}
\] (3.1)

such that the final solution $\varphi(x, t_f)$ coincides with a target function $\bar{\varphi}$. 

CHAPTER 3. OPTIMAL CONTROL FOR COMPR.-INCOMPR. FLOWS

Notice that we do not impose any boundary conditions on system (3.1), as we assume the condition

\[ C = 0 \quad \text{on} \quad \partial \Omega \]  

(3.2)
to hold. See section 3.1.2 for further details on how this constraint is enforced.

Formally, the problem can be described as follows.

**Problem 3.1.** Let \( \mathcal{C} = H^1_0(\Omega) \), i.e. the set of functions in \( H^1(\Omega) \) that vanish on \( \partial \Omega \), be the space of admissible controls. Given three constants \( \sigma_1, \sigma_2, \sigma_3 \in \mathbb{R} \), a function \( \bar{\varphi} : \Omega \to \mathbb{R} \) and an objective functional \( J \) defined as:

\[
J : \mathcal{C} \to \mathbb{R}
\]

\[
C \to J(C) = \frac{\sigma_1}{2} \int_\Omega (\varphi(C)|_{t=t_f} - \bar{\varphi})^2 \, dx + \frac{\sigma_2}{2} \|C\|_{L^2(\Omega)}^2 + \frac{\sigma_3}{2} \|C'\|_{L^2(\Omega)}^2,
\]

(3.3)
fund \( C^* = \arg\min_{C \in \mathcal{C}} J(C) \) such that the equation system (3.1) holds.

The objective functional (3.3) does not only measure, through its first term, the discrepancy between the solution to problem (3.1) at the final time \( \varphi(t_f, x) \) and the target function \( \bar{\varphi} \), but it also penalizes the norm of the control and of its first derivative, via its second and third terms. This is a standard approach in the literature, used to prevent the control from becoming too wild. Furthermore, notice that the presence of the derivative of the control in the objective functional will affect the way the descent direction is computed. See section 3.1.2.1 for further details.

### 3.1.2 The optimality conditions

Let us introduce the Lagrangian functional \( \mathcal{L} \), which is defined as:

\[
\mathcal{L}(\varphi, \hat{\varphi}, C) = \frac{\sigma_1}{2} \int_\Omega (\varphi(C)|_{t=t_f} - \bar{\varphi})^2 \, dx + \frac{\sigma_2}{2} \|C\|_{L^2(\Omega)}^2 + \frac{\sigma_3}{2} \|C'\|_{L^2(\Omega)}^2 +
\]

\[
- \int_{t_0}^{t_f} \int_\Omega \left( \frac{\partial \varphi}{\partial t} + C \frac{\partial \varphi}{\partial x} \right) \hat{\varphi} \, dx \, dt - \int_\Omega (\varphi|_{t=t_0} - \varphi_0) \hat{\theta} \, dx,
\]

(3.4)

\( \hat{\varphi} \) and \( \hat{\theta} \) being the adjoint variable, which assume the role of Lagrangian multipliers. The first three terms on the right hand side of (3.4) are the cost functional \( J \), while the remaining terms are the inner products between the equations in the state system (3.1) and the adjoint variables.

As seen in section 2.2.2, the optimality conditions are obtained by looking for the stationary points of \( \mathcal{L} \), i.e.

\[ \nabla \mathcal{L} = 0. \]
3.1. AN INTRODUCTORY CONTROL PROBLEM

Explicitly, this means that the system:

\[
\begin{align*}
\sigma_1 & \int_0^t \left( \varphi|_{t=t_f} - \bar{\varphi} \right) \delta \varphi|_{t=t_f} \, dx - \int_{t_0}^{t_f} \int_\Omega \left( \frac{\partial}{\partial t} \delta \varphi + C \frac{\partial}{\partial x} \delta \varphi \right) \varphi \, dx \, dt + \\
& \quad - \int_\Omega \delta \varphi|_{t=t_0} \bar{\theta} \, dx = 0, \\
\int_{t_0}^{t_f} \int_\Omega \left( \frac{\partial}{\partial t} + C \frac{\partial}{\partial x} \right) \delta \varphi \, dx \, dt = 0, \\
\sigma_2 \left( C', \delta C \right)_{L^2(\Omega)} + \sigma_3 \left( C', (\delta C)' \right)_{L^2(\Omega)} - \int_{t_0}^{t_f} \int_\Omega \delta C \frac{\partial}{\partial x} \varphi \, dx \, dt = 0,
\end{align*}
\]

must hold \( \forall \delta \varphi \in \mathcal{F}, \forall \delta \hat{\varphi} \in \hat{\mathcal{F}} \) and \( \forall \delta C \in \mathcal{C} \), where \( \mathcal{F} \) and \( \hat{\mathcal{F}} \) denote the spaces where increments variables \( \delta \varphi \) and \( \delta \hat{\varphi} \) lie, respectively, and \( (\cdot, \cdot)_{L^2(\Omega)} \) denotes the usual scalar product in \( L^2(\Omega) \).

By integration by parts, equation (3.5) can be proven to be equivalent to

\[
\int_{t_0}^{t_f} \int_\Omega \left( \frac{\partial}{\partial t} + C \frac{\partial}{\partial x} \right) \delta \varphi \, dx \, dt + \int_\Omega \left( \sigma_1 \left( \varphi|_{t=t_f} - \bar{\varphi} \right) - \phi|_{t=t_f} \right) \delta \varphi|_{t=t_f} \, dx + \\
+ \int_\Omega \left( \bar{\varphi}|_{t=t_0} - \bar{\theta} \right) \delta \varphi|_{t=t_0} \, dx = 0.
\]

Section 3.2.3 shows this procedure in greater details.

Recalling that the variations \( \delta \varphi \) and \( \delta \hat{\varphi} \) are arbitrary, we obtain the optimality conditions. They are given by:

(A-i) the state system (3.1);

(A-ii) the adjoint system

\[
\begin{align*}
- \frac{\partial}{\partial t} \hat{\varphi} - \frac{\partial}{\partial x} (C \hat{\varphi}) &= 0 \quad \text{in } \Omega \times (t_0, t_f], \\
\hat{\varphi}|_{t=t_f} &= \sigma_1 \left( \varphi|_{t=t_f} - \bar{\varphi} \right) \quad \text{in } \Omega,
\end{align*}
\]

(A-iii) the variational equality

\[
\sigma_2 \left( C', \delta C \right)_{L^2(\Omega)} + \sigma_3 \left( C', (\delta C)' \right)_{L^2(\Omega)} + \\
- \int_{t_0}^{t_f} \int_\Omega \delta C \frac{\partial}{\partial x} \hat{\varphi} \, dx \, dt = 0 \quad \forall \delta C \in \mathcal{C}.
\]

In particular, equation (3.7) yields the gradient of the cost functional with respect to the control variable. Indeed, by integrating the second term by parts, (3.7) can be rewritten as

\[
\langle \nabla C J, \delta C \rangle_\mathcal{C} = 0 \quad \forall \delta C \in \mathcal{C},
\]
with
\[ \nabla C J = \sigma_2 C - \sigma_3 C'' - \int_{t_0}^{t_f} \frac{\partial \varphi}{\partial x} \dot{\varphi} \, dt. \] (3.8)

The gradient equation (3.8) will be employed in the gradient-type algorithm introduced in section 2.2.3.1, whose form in the specific case of problem 3.1 will be detailed in section 3.1.3.

3.1.2.1 The descent direction

As seen in section 2.2.3, a search direction \( d \) needs to be computed at each iteration in the minimization algorithm 3. While the most common choice would be using the negative gradient of the cost functional, this can not be done in this case, due to regularity issues that would arise. Indeed, since the computation of the gradient of the cost functional involves the second derivative of the control \( C \), setting
\[ d = -\nabla J \left( C^{(k)} \right) \]
and then using \( d \) to compute the new control value as:
\[ C^{(k+1)} = C^{(k)} + \alpha^{(k)} d \]
would cause \( C \) to lose two orders of regularity at each iteration.

To overcome this issue, let \( B \) be a coercive operator and let us consider the equation:
\[ B d = -\nabla J \left( C^{(k)} \right). \] (3.9)

Since \( B \) is coercive, it holds that:
\[ 0 < (B d, d) = \left( -\nabla J \left( C^{(k)} \right), d \right). \]

The solution of (3.9) then satisfies (2.7) and it is therefore a descent direction.

If we now choose \( B = -\Delta + I \), the solution of
\[ \begin{cases} -\Delta d + d = -\nabla J \left( C^{(k)} \right) & \text{in } \Omega, \\
 d = 0 & \text{on } \partial \Omega, \end{cases} \] (3.10)
can be used to update the value of the control function \( C^{(k)} \). Indeed, \( B \) is a coercive operator. Furthermore, thanks to the *elliptic regularity theorem* (see e.g. [Sal16]), the solution \( d \) to problem (3.10) has two orders of regularity more than \( \nabla J \left( C^{(k)} \right) \) (i.e. the same regularity as \( C^{(k)} \)). Finally, thanks to the boundary condition forcing it to be zero on \( \partial \Omega \), it enforces constraint (3.2), assuming the initial value \( C^{(0)} \) is zero-valued on \( \partial \Omega \) to begin with.
3.1.3 Space-time discretization and minimization algorithm

Let us now describe the way problems (3.1) and (3.6) were discretized. The same approach and the same notation as those used for the advection equation in section 1.2.3 are used. The solution interval \( (t_0, t_f) \) is divided into \( N \) subintervals of length \( \Delta t = \frac{t_f - t_0}{N} \) and discontinuous Finite Elements are used for the discrete counterparts of both \( \varphi \) and \( \bar{\varphi} \) (denoted by \( \varphi_h \) and \( \bar{\varphi}_h \), respectively) at each time step. The discrete control value \( C_h \), on the other hand, lies in \( Q_{h,0} \), the space of all functions in \( Q_h \) that vanish on \( \partial \Omega \).

Notice that the adjoint equation (3.6) is to be solved backwards (that is, from \( t = t_f \) to \( t = t_0 \) with negative time steps), as it is standard for time-dependent adjoint systems. However, in the following, we will always suppose \( \Delta t \) to be positive, so that a superscript \( (n) \) will still denote the value of each quantity at time \( t^{(n)} = t_0 + n\Delta t \).

The time discretization for the primal problem goes as follows. Let the initial value \( \varphi_h^{(0)} \) be given. Then, for every \( n = 1, 2, \ldots, N \), find \( \varphi_h^{(n)} \) in \( W_h \) such that for all \( w_h \in W_h \)

\[
\sum_{E \in \mathcal{T}_h} \int_E \frac{\varphi_h^{(n)}}{\Delta t} w_h \, dx - \sum_{E \in \mathcal{T}_h} \int_E \varphi_h^{(n)} C_h w'_h \, dx - \sum_{E \in \mathcal{T}_h} \int_E \varphi_h^{(n)} C_h w_h \, dx +
+ \sum_{e \in \mathcal{E}_h} \int_e \left( \left\{ \varphi_h^{(n)} \right\} C_h [w_h] + \frac{1}{2} C_h \nu [\varphi_h^{(n)}] \cdot [w_h] \right) \, ds
\]

\begin{equation}
(3.11)
\end{equation}

\[
+ \sum_{e \in \mathcal{E}_h} \int_e C_h [w_h] \{ \varphi_h^{(n)} \} \, ds = \sum_{E \in \mathcal{T}_h} \int_E \frac{\varphi_h^{(n-1)}}{\Delta t} w_h \, dx,
\]

where \( \nu \) is the outward normal, given by:

\[
\nu = \begin{cases} 
1 & \text{on the right boundary of } \Omega, \\
-1 & \text{on the left boundary of } \Omega,
\end{cases}
\]

\begin{equation}
(3.12)
\end{equation}

and \( \varphi_h^n = \varphi_h^{(n)} \), since \( C_h \) vanishes on \( \partial \Omega \) and therefore \( C_h \nu \leq 0 \) on the domain boundary (see section 1.2.3 and, in particular, step 3T).

As for the adjoint problem, given the initial condition \( \bar{\varphi}_h^{(N)} \), the discretized problem is solved by finding, for every \( n = N - 1, N - 2, \ldots, 0 \), \( \bar{\varphi}_h^{(n)} \) in \( W_h \) such that for all \( w_h \in W_h \)

\[
\sum_{E \in \mathcal{T}_h} \int_E \frac{\bar{\varphi}_h^{(n)}}{\Delta t} w_h \, dx - \sum_{E \in \mathcal{T}_h} \int_E \bar{\varphi}_h^{(n)} C_h w'_h \, dx + \sum_{e \in \mathcal{E}_h} \int_e C_h [w_h] \{ \varphi_h^n \} \, ds +
+ \sum_{e \in \mathcal{E}_h} \int_e \left( \left\{ \bar{\varphi}_h^{(n)} \right\} C_h [w_h] + \frac{1}{2} C_h \nu [\bar{\varphi}_h^{(n)}] \cdot [w_h] \right) \, ds
\]

\begin{equation}
(3.13)
\end{equation}

\[
= \sum_{E \in \mathcal{T}_h} \int_E \frac{\bar{\varphi}_h^{(n+1)}}{\Delta t} w_h \, dx,
\]
Algorithm 4 Optimal control problem solution method

Let an initial guess $C^{(0)}$ and two positive constants $\varepsilon_g \varepsilon_J$ be given.

Set \( k = 0 \)

\[
\text{while } \frac{\|C^{(k)} - C^{(k-1)}\|_{\mathcal{F}}}{\|C^{(k-1)}\|_{\mathcal{F}}} > \varepsilon_g \land \frac{\|\nabla J\left(C^{(k+1)}\right)\|_{\mathcal{F}}}{\|\nabla J\left(C^{(k)}\right)\|_{\mathcal{F}}} > \varepsilon_J \text{ do}
\]

Solve system (3.1) using equation (3.11) and the current control value $C^{(k)}$

Solve system (3.6) using equation (3.13) and newly computed solution of the state system

Solve equation (3.10) to compute the descent direction $d$

Compute step length:

\[
\begin{align*}
\text{begin} \\
&\text{Let an initial guess } \alpha^{(0)} \text{ and two constants } c_1 > 0 \text{ and } c_2 \in (0, 1) \text{ be given} \\
&\text{Solve system (3.1) using equation (3.11) and } C^{(k)} + \alpha^{(0)}d \text{ as advection coefficient} \\
&\text{Set } m = 0 \\
&\text{while } J\left(C^{(k)} + \alpha^{(m)}d\right) > J\left(C^{(k)}\right) + c_1\alpha^{(m)}\left(\nabla J\left(C^{(k)}\right), d\right) \text{ do} \\
&\quad \text{Set } \alpha^{(m+1)} = c_2 \alpha^{(m)} \\
&\quad \text{Solve system (3.1) using equation (3.11) and } C^{(k)} + \alpha^{(m+1)}d \text{ as advection coefficient} \\
&\quad m \leftarrow m + 1 \\
&\text{end while} \\
&\text{Set } C^{(k+1)} = C^{(k)} + \alpha^{(m)}d \\
&k \leftarrow k + 1 \\
\text{end while}
\end{align*}
\]

$\nu$ again being given by (3.12) and, using the same argument as above, $\hat{\varphi}_h^p = \hat{\varphi}_h^n$.

As in section 1.2.3, see [DLP06] for more details on how the discrete weak formulations (3.11) and (3.13) were obtained.

We can now rewrite algorithm 3 for the optimal control problems defined in section 3.1.1. Algorithm 4 sums up the complete solution procedure for problem 3.1.

3.1.4 Numerical results

To conclude this introduction, let us show some results on the introductory optimal control problem just presented. Notice that what has been discussed so far regarding problem 3.1 is completely general and it depends neither on the choice of the initial condition $\varphi_0$ for $\varphi$ nor on the target solution $\tilde{\varphi}$. However, since our interest lies in defining a building block for problems 3.2 and 3.3 in section 3.2, we want to investigate what happens when both the function being transported and the target solution are signed distance functions. Because of this, the discrete solution $\varphi_h^{(n)}$, computed using equation (3.11), is reinitialized at each time step when the discrete primal problem is solved, as seen in section 1.2.1 and in the discretized systems already discussed.
3.1. AN INTRODUCTORY CONTROL PROBLEM

e.g. steps 1T - 6T in section 1.2.3).

Let us then describe the test case being investigated. All the dimensional quantities are expressed in the International System of Units (SI). The domain is $\Omega = [-1, 1]$ and it is discretized using $N_h = 100$ mesh point. The initial time is $t_0 = 0$, while the final time is $t_f = 1$, with time step $\Delta t = 0.01$. The initial and target solution are given by:

$$\varphi_0(x) = \begin{cases} 
-x - 0.05 & x \in [-1, 0], \\
x - 0.05 & x \in [0, 1],
\end{cases} \quad \bar{\varphi}(x) = \begin{cases} 
-x + 0.15 & x \in [-1, 0.4], \\
x - 0.65 & x \in [0.4, 1],
\end{cases}$$

respectively. They are represented in figure 3.1.

As for the control parameters introduced in section 2.2.3, $\alpha$, $c_1$ and $c_2$ are set to 10, 0.001 and 0.5, respectively, while the thresholds for the convergence check are $\varepsilon_g = \varepsilon_J = 10^{-6}$. The penalization parameters in the objective functional (3.3) are $\sigma_1 = 100$ and $\sigma_2 = \sigma_3 = 1 \times 10^{-6}$. Finally, the initial guess for the control function is $C(0)(x) = 0$.

Figures 3.2, 3.3 and 3.4 show the history of convergence for the solution of problem 3.1, the optimal control $C^*$ and the solution to equation (3.1) computed with $C = C^*$. The target solution is well recovered (figure 3.4) and the objective functional (top left image in figure 3.2) shows a monotone decrease. Furthermore, the control terms in the objective functional can be computed exactly for $C^*$:

$$J_C(C^*) = \frac{\sigma_2}{2} \|C^*\|_{L^2(\Omega)}^2 + \frac{\sigma_3}{2} \|\nabla (C^*)\|_{L^2(\Omega)}^2 = 3.1958 \times 10^{-6},$$

which means that the real discrepancy between target and computed solution is ac-
3.2 The optimal control problem of interest

We can now derive the final optimal control problems that were used for the numerical results on the controllability of two-phase flows we are considering in this work. Our main interest lies in control problems where the compressible-incompressible model is used as state system, but intermediate results obtained in the incompressible-incompressible case will be shown as well.

Notice that, in this work, a distributed volumetric force, which may be thought of as the effect of a local temperature gradient, is used as the control term. Clearly,
3.2. THE OPTIMAL CONTROL PROBLEM OF INTEREST

Figure 3.3: Introductory optimal control problem, optimal value of the control function $C$.

Figure 3.4: Introductory optimal control problem results: controlled solution (orange) and target solution (dashed black).
other different choices are possible. For example, one may consider a boundary control, which could closer represent what is realized in practical industrial applications where, for instance, the mould wall can be suitably heated to drive the foaming process towards desired configurations.

### 3.2.1 A compressible-incompressible control problem

The optimal control problem we will deal with is the following. Let $\Omega$ represent a box containing a given number of gas bubbles moving within a liquid. Our aim is to formulate an optimal control problem whose solution yields the volumetric force to be applied to the system in order to drive the gas bubbles to a given final position at a given time. An example setting is shown in figure 3.5a.

Let us denote by $C$ the control variable acting on the right hand side of the momentum equation. Then the state system, which is based on the model detailed in
section 1.3.2, reads as follows:

\[
\begin{align*}
\nabla \cdot \mathbf{u} + \chi T \left( \frac{\partial p}{\partial t} + \mathbf{u} \cdot \nabla p \right) &= 0 \quad \text{in } \Omega \times (t_0, t_f], \\
\rho \frac{\partial \mathbf{u}}{\partial t} + \rho (\mathbf{u} \cdot \nabla) \mathbf{u} - \nabla \cdot \mathbf{\tau} + \nabla p &= \rho \mathbf{g} + \rho \mathbf{C} + \mathbf{s} \quad \text{in } \Omega \times (t_0, t_f], \\
\frac{\partial \varphi}{\partial t} + \mathbf{u} \cdot \nabla \varphi &= 0 \quad \text{in } \Omega \times (t_0, t_f], \\
H &= F(\varphi) = \begin{cases} 1 & \text{if } \varphi \leq 0 \\ 0 & \text{if } \varphi > 0 \end{cases} \quad \text{in } \Omega \times (t_0, t_f], \\
\rho &= \frac{p}{\rho T} H + \rho_l (1 - H) \quad \text{in } \Omega \times (t_0, t_f], \\
\mu &= \mu_\alpha H + \mu_l (1 - H) \quad \text{in } \Omega \times (t_0, t_f], \\
\lambda &= \lambda_\alpha H + \lambda_l (1 - H) \quad \text{in } \Omega \times (t_0, t_f], \\
\chi T &= \frac{1}{p} H \quad \text{in } \Omega \times (t_0, t_f], \\
\mathbf{u} &= \mathbf{u}_d \quad \text{on } \Gamma_D \times (t_0, t_f], \\
-p \mathbf{\nu} + \mathbf{\tau} \cdot \mathbf{\nu} &= p_b \mathbf{\nu} \quad \text{on } \Gamma_N \times (t_0, t_f], \\
\mathbf{u}|_{t=t_0} &= \mathbf{u}_0 \quad \text{in } \Omega, \\
p|_{t=t_0} &= p_0 \quad \text{in } \Omega, \\
\varphi|_{t=t_0} &= \varphi_0 \quad \text{in } \Omega,
\end{align*}
\]

where \( \mathbf{\nu} \) is the outward normal and \( \Gamma_D \) and \( \Gamma_N \) split up \( \partial \Omega \) in a way such that \( \Gamma_D \cup \Gamma_N = \partial \Omega \) and \( \Gamma_D \cap \Gamma_N = \emptyset \). Notice that, as already mentioned in the introduction, in this work we only consider isothermal flows, so the temperature \( T \) was discarded from (1.23) to build system (3.14).

We are interested in solving the following minimization problem:

**Problem 3.2.** Let \( \mathscr{C} = L^2 \left( (t_0, t_f); \left( L^2(\Omega) \right)^d \right) \) be the space of admissible controls. Given two constants \( \sigma_1, \sigma_2 \in \mathbb{R} \), a function \( \bar{\varphi}: \Omega \to \mathbb{R} \) and an objective functional \( J \) defined as:

\[
J: \mathscr{C} \to \mathbb{R} \\
C \to J(C) = \frac{\sigma_1}{2} \int_{\Omega} \left( \varphi(C)|_{t=t_f} - \bar{\varphi} \right)^2 \, dx + \frac{\sigma_2}{2} \|C\|^2_{\mathscr{C}},
\]

find \( C^* = \arg\min_{C \in \mathscr{C}} J(C) \) such that the equation system (3.14) holds.

Let us comment on the objective functional (3.15). Similarly to what was done for the cost functional (3.3), the first term ensures that the final solution for the signed distance equation (3.14) is close to the target solution \( \bar{\varphi} \). This amounts to prescribing a final distribution of the bubbles. The second term is the standard term penalising the norm of the control already discussed for (3.3).
Notice that we specifically chose to use the signed distance function instead of the Heaviside function to measure the discrepancy between the computed and the target bubbles’ position. The reason for this lies in the particular shape that Heaviside functions have. Indeed, if Heaviside functions were used, there would exist several distinct configurations that would be indistinguishable from each other, as far as the objective functional is concerned. Figure 3.5b shows a toy example of this, where the difference between the Heaviside functions associated with the target and computed position is exactly the same, independently of which of the blue bubbles we consider, since, by definition, each Heaviside function is non-zero only inside the bubble itself. But, of course, the final position tagged with the number “2” is much closer to the target. This means that, given a value for the control function, an increment for such function would not be seen as an improvement over the old control (even if it actually is), unless the overlapping area between the target position and the new final position increases. The minimization algorithm would then reject the new value for the optimal control in most cases, stalling in the current configuration as if it were the best it can achieve and failing to get to the real optimal control. All of these issues vanish if signed distance functions are used.

As seen in section 3.1.2, the following optimality conditions will be crucial, in view of designing a numerical algorithm to solve problem 3.2.

3.2.1.1 Compressible-incompressible optimality conditions

The optimality conditions for the two-phase compressible-incompressible control problem 3.2 are given by:

\[(B-i) \text{ the state system (3.14);}\]
(B-ii) the adjoint system

\[
\begin{aligned}
- \frac{\partial}{\partial t} (\rho \hat{u}) + \rho (\nabla u)^T \cdot \hat{u} - \rho (u \cdot \nabla) \hat{u} + \\
- \nabla \cdot (\mu \nabla \hat{u} + \mu (\nabla \hat{u})^T) - \nabla (\lambda \nabla \cdot \hat{u}) + \\
- \nabla \hat{p} + \chi_T \hat{p} \nabla p + \nabla \varphi \hat{\varphi} = 0 & \quad \text{in } \Omega \times (t_0, t_f], \\
\nabla \cdot \hat{u} = - \frac{\partial}{\partial t} (\chi_T \hat{p}) - \nabla \cdot (\chi_T u \hat{p}) + \frac{H}{r_T^2} \hat{p} + \\
+ \chi_T \chi_T & \quad \text{in } \Omega \times (t_0, t_f], \\
- \frac{\partial \hat{\varphi}}{\partial t} - \nabla \cdot (\hat{\varphi} u) = - \delta (\varphi) \hat{H} + R (\varphi, \hat{u}) & \quad \text{in } \Omega \times (t_0, t_f], \\
\hat{H} = \left( \frac{p}{r_T^2} - p_1 \right) \hat{p} + (\mu_g - \mu_t) \hat{\mu} + \\
+ (\lambda_g - \lambda_t) \hat{\lambda} + \chi_T & \quad \text{in } \Omega \times (t_0, t_f], \\
\hat{p} = - \frac{\partial u}{\partial t} \cdot \hat{u} - (u \cdot \nabla) u \cdot \hat{u} + g \cdot \hat{u} + C \cdot \hat{u} & \quad \text{in } \Omega \times (t_0, t_f], \quad (3.16) \\
\hat{\mu} = - \nabla u : \nabla \hat{u} - (\nabla u)^T : \nabla \hat{u} & \quad \text{in } \Omega \times (t_0, t_f], \\
\hat{\lambda} = - (\nabla \cdot u) I : \nabla \hat{u} & \quad \text{in } \Omega \times (t_0, t_f], \\
p \chi_T = - \hat{p} \left( \frac{\partial p}{\partial t} + u \cdot \nabla p \right) & \quad \text{in } \Omega \times (t_0, t_f], \\
\hat{u} = 0 & \quad \text{on } \Gamma_D \times (t_0, t_f], \\
- (\mu \nabla \hat{u} + \mu (\nabla \hat{u})^T) \cdot \nu - (\lambda \nabla \cdot \hat{u}) \nu + \\
- \hat{p} \nu - \rho (u \cdot \nabla) \hat{u} = 0 & \quad \text{on } \Gamma_N \times (t_0, t_f], \\
\hat{u}|_{t=t_f} = 0 & \quad \text{in } \Omega, \\
\hat{p}|_{t=t_f} = 0 & \quad \text{in } \Omega, \\
\hat{\varphi}|_{t=t_f} = \sigma_1 \left( \varphi|_{t=t_f} - \hat{\varphi} \right) & \quad \text{in } \Omega,
\end{aligned}
\]

where \( \hat{u}, \hat{p}, \hat{\varphi}, \hat{H}, \hat{\mu}, \hat{\lambda} \) and \( \chi_T \) are the adjoint variables. The source term \( R (\varphi, \hat{u}) \) in equation \( (3.16)_3 \) is defined as:

\[
R (\varphi, \hat{u}) = \nabla \cdot \left\{ \left[ - \nabla \left( \varsigma \frac{\nabla \varphi}{\| \nabla \varphi \|} \delta_\varepsilon (\varphi) \hat{u} \right) + \\
+ \varsigma \nabla \cdot \left( \frac{\nabla \varphi}{\| \nabla \varphi \|} \right) \delta_\varepsilon (\varphi) \hat{u} \right\} \frac{1}{\| \nabla \varphi \|} \left( 1 - \left( \frac{\nabla \varphi}{\| \nabla \varphi \|} \right)^2 \right) \right\} + \\
- \varsigma \nabla \cdot \left( \frac{\nabla \varphi}{\| \nabla \varphi \|} \right) \frac{\nabla \varphi}{\| \nabla \varphi \|} \delta_\varepsilon' (\varphi) \hat{u},
\]

where \( \delta_\varepsilon' (\varphi) \) denotes the derivative of the smooth Dirac delta function \( (1.20) \);
(B-iii) the variational equality

\[
\sigma_2 (C, \delta C) + \int_{t_0}^{t_f} \int_\Omega \rho \delta C \cdot \dot{u} \, dx \, dt = 0 \quad \forall \delta C \in \mathcal{C},
\]

(3.17)

where \((\cdot, \cdot)_{\mathcal{C}}\) denotes the usual scalar product in \(L^2 \left((t_0, t_f); (L^2 (\Omega))^d\right)\).

The derivation of conditions (B-i)-(B-iii) follows the procedure described in section 2.2.2 for a generic optimal control problem, which was already applied to compute the optimality conditions (A-i)-(A-iii) for problem 3.1 (see section 3.1.2). Denoting the vectors of state and adjoint variables by \(y = (u, p, \varphi, H, \rho, \mu, \lambda, \chi_T)\) and \(\hat{y} = (\dot{u}, \dot{p}, \dot{\varphi}, \dot{H}, \dot{\rho}, \dot{\mu}, \dot{\lambda}, \dot{\chi_T}, \dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3, \dot{\theta}_4)\), respectively, we consider the following Lagrangian functional:

\[
\mathcal{L} (y, \dot{y}, C) = \frac{\sigma_1}{2} \int_\Omega \left( \phi (C)|_{t=t_f} - \dot{\varphi} \right)^2 \, dx + \frac{\sigma_2}{2} \|C\|^2_{\mathcal{C}} +
\]

\[
- \int_{t_0}^{t_f} \int_\Omega \left( \rho \frac{\partial u}{\partial t} + \rho (u \cdot \nabla) u - \nabla \cdot \tau + \nabla p - \rho g - \rho C - s \right) \cdot \dot{u} \, dx \, dt +
\]

\[
- \int_{t_0}^{t_f} \int_\Omega \left( \nabla \cdot u + \chi_T \left( \frac{\partial \rho}{\partial t} + u \cdot \nabla \rho \right) \right) \dot{p} \, dx \, dt +
\]

\[
- \int_{t_0}^{t_f} \int_\Omega \left( \frac{\partial \varphi}{\partial t} + u \cdot \nabla \varphi \right) \dot{\varphi} \, dx \, dt - \int_{t_0}^{t_f} \int_\Omega \left( H - F (\varphi) \right) \dot{H} \, dx \, dt +
\]

\[
- \int_{t_0}^{t_f} \int_\Omega \left( \rho - \frac{p}{rT} H - (1 - H) \rho I \right) \dot{p} \, dx \, dt +
\]

\[
- \int_{t_0}^{t_f} \int_\Omega \left( \mu - \mu_g H - (1 - H) \mu I \right) \dot{\mu} \, dx \, dt +
\]

\[
- \int_{t_0}^{t_f} \int_\Omega \left( \lambda - \lambda_g H - (1 - H) \lambda I \right) \dot{\lambda} \, dx \, dt - \int_{t_0}^{t_f} \int_\Omega \left( p \chi_T - H \right) \dot{\chi_T} \, dx \, dt +
\]

\[
- \int_{t_0}^{t_f} \int_{\Gamma_p} (u - u_d) \cdot \dot{\theta}_1 \, ds \, dt - \int_\Omega \left( u \big|_{t=t_0} - u_0 \right) \cdot \dot{\theta}_2 \, dx +
\]

\[
- \int_{t_0}^{t_f} \int_{\Gamma_N} \left( \rho \nu + \tau \cdot \nu - p_N \nu \right) \dot{\theta}_3 \, ds \, dt - \int_\Omega \left( p \big|_{t=t_0} - p_0 \right) \dot{\theta}_4 \, dx +
\]

\[
- \int_\Omega \left( \varphi \big|_{t=t_0} - \varphi_0 \right) \dot{\theta}_4 \, dx.
\]

The first term on the right hand side of (3.18) is the cost functional \(J\), while the remaining terms are the inner products between the equations in the state system and the adjoint variables. The optimality conditions are obtained by looking for a stationary point of \(\mathcal{L}\), i.e. by solving the system of equations:

\[
\nabla \mathcal{L} = 0.
\]

See section 3.2.3 for further details.

As shown in sections 2.2.2 and 3.1.2, the variational equality (3.17) yields the
3.2. THE OPTIMAL CONTROL PROBLEM OF INTEREST

The gradient of the cost functional with respect to the control variable, since (3.17) can be recast as:

\[(\nabla_C J, \delta C)_{\mathcal{C}} = 0 \quad \forall \delta C \in \mathcal{C},\]

with

\[\nabla_C J = \sigma_2 C + \rho \hat{u}. \quad (3.19)\]

Unlike section 3.1.2, here there are no regularity issues, and therefore the negative gradient of the cost functional can be directly used as search direction. Section 3.2.4 details the particularization of the minimization algorithm 3 (see section 2.2.3.1) to the case of problem 3.2.

3.2.2 The incompressible-incompressible case

The same approach as in section 3.2.1 can be used in the case of an incompressible-incompressible model. By using equation (1.18) as state system instead of (1.23), we can formulate a problem equivalent to 3.2 in the incompressible-incompressible case, which reads:

**Problem 3.3.** Let \( \mathcal{C} = L^2((t_0, t_f); (L^2(\Omega))^d) \) be the space of admissible controls. Given two constants \( \sigma_1, \sigma_2 \in \mathbb{R} \), a function \( \bar{\varphi} : \Omega \to \mathbb{R} \) and an objective functional \( J \) defined as:

\[
J : \mathcal{C} \to \mathbb{R}
C \to J(C) = \frac{\sigma_1}{2} \int_{\Omega} \left( \varphi(C)|_{t=t_f} - \bar{\varphi} \right)^2 \, dx + \frac{\sigma_2}{2} \|C\|_{\mathcal{C}}^2,
\]

find \( C^* = \arg\min_{C \in \mathcal{C}} J(C) \) such that the equation system

\[
\begin{aligned}
\rho \frac{\partial \hat{u}}{\partial t} + \rho (u \cdot \nabla) u - \nabla \cdot \tau + \nabla p &= \rho \mathbf{g} + \rho C + \mathbf{s} & \text{in } \Omega \times (t_0, t_f], \\
\nabla \cdot u &= 0 & \text{in } \Omega \times (t_0, t_f], \\
\frac{\partial \varphi}{\partial t} + u \cdot \nabla \varphi &= 0 & \text{in } \Omega \times (t_0, t_f], \\
H &= H(\varphi) = \begin{cases} 1 & \text{if } \varphi \leq 0 \\
0 & \text{if } \varphi > 0 \end{cases} & \text{in } \Omega \times (t_0, t_f], \\
\rho &= \rho_1 H + (1 - H) \rho_2 & \text{in } \Omega \times (t_0, t_f], \\
\mu &= \mu_1 H + (1 - H) \mu_2 & \text{in } \Omega \times (t_0, t_f], \\
u &= u_0 & \text{on } \Gamma_D \times (t_0, t_f], \\
-p \nu + \tau \cdot \nu &= p_b \nu & \text{on } \Gamma_N \times (t_0, t_f], \\
|_{t=t_0} &= u_0 & \text{in } \Omega, \\
|_{t=t_0} &= \varphi_0 & \text{in } \Omega.
\end{aligned}
\]

(3.21)
holds.

Using the same arguments as in section 3.2.1, the following optimality conditions can be computed.

### 3.2.2.1 Incompressible-incompressible optimality conditions

The optimality conditions for the incompressible-incompressible control problem 3.3 are given by:

**(T-i)** the state system (3.21);

**(T-ii)** the adjoint equation

\[
\begin{align*}
\frac{\partial}{\partial t} (p\hat{u}) + \rho (\nabla u)^T \cdot \hat{u} - \rho (u \cdot \nabla) \hat{u} + \\
- \nabla \cdot (\mu \nabla \hat{u} + \mu (\nabla \hat{u})^T) - \nabla \hat{p} + \nabla \varphi \hat{\varphi} &= 0 \\
\nabla \cdot \hat{u} &= 0 \\
\frac{\partial \hat{\varphi}}{\partial t} - \nabla \cdot (\hat{\varphi} \, u) &= -\hat{\varphi} (H + R (\varphi, \hat{u})) \\
\hat{H} &= (\rho_1 - \rho_2) \hat{\rho} + (\mu_1 - \mu_2) \hat{\mu} \\
\hat{\rho} &= -\frac{\partial u}{\partial t} \cdot \hat{u} - (u \cdot \nabla) u \cdot \hat{u} + g \cdot \hat{u} + C \cdot \hat{u} \\
\hat{\mu} &= -\nabla u : \nabla \hat{u} - (\nabla u)^T : \nabla \hat{u} \\
\hat{u} &= 0 \\
- (\mu \nabla \hat{u} + \mu (\nabla \hat{u})^T) \cdot \nu - \hat{\rho} \nu - \rho (u \cdot \nabla) \hat{u} &= 0 \\
\hat{u} |_{t=t_f} &= \frac{\sigma_1}{\rho} \left( \frac{u}{\rho} \right) |_{t=t_f} \\
\hat{\varphi} |_{t=t_f} &= \sigma_1 \left( \varphi |_{t=t_f} - \hat{\varphi} \right)
\end{align*}
\]

(3.22)

where \( \hat{u}, \hat{\rho}, \hat{\varphi}, \hat{H}, \hat{\rho} \) and \( \hat{\mu} \) are the adjoint variables and the source term \( R (\varphi, \hat{u}) \) in equation (3.22) is defined as in (3.16);

**(T-iii)** the variational equality

\[
\sigma_2 (C, \delta C)_{\delta \varphi} + \int_{t_0}^{t_f} \int_{\Omega} \rho \delta \nabla u \cdot \delta \hat{u} \, dx \, dt = 0 \quad \forall \delta C \in \mathcal{C},
\]

(3.23)

In particular, notice that the variational inequality has the same expression in both the compressible-incompressible case in section 3.2.1 and the incompressible-incompressible case above (compare equations (3.23) and (3.17)).
3.2.3 Derivation of the adjoint system for compressible-incompressible flows

In section 3.2.1.1 the final form of the adjoint system for problem 3.2 is acknowledged but never proved, for the sake of simplicity, as we wanted to keep the section as sleek as possible in order to better show the core argument. Let us now show all the computations needed to explicitly derive the adjoint problem (3.16).

Notice that most of the test cases in chapter 4 enforce Dirichlet boundary conditions on the whole boundary $\partial \Omega$. This can be seen as a particular case of (3.14) and (3.16) when $\Gamma_N$ is assumed to be equal to the empty set, so nothing formally changes in the computations shown in the next sections.

3.2.3.1 Derivatives of the Lagrangian functional

As seen in section 2.2.2 and 3.1.2, by computing the derivatives of the Lagrangian functional with respect to the primal variables and by setting them all equal to zero, we get to the adjoint system.

In the following, the function spaces $\mathcal{U}, \mathcal{P}, \mathcal{F}, \mathcal{H}, \mathcal{R}, \mathcal{N}, \mathcal{M}$ and $\mathcal{X}_T$ will denote the proper sets in which the increments variables $\delta u, \delta p, \delta \varphi, \delta H, \delta \rho, \delta \mu, \delta \lambda$ and $\delta \chi_T$ lie, respectively.

Let us then compute the derivatives of (3.18) explicitly and set them all equal to zero. For the time being, the contribution of the surface tension term will be neglected in the computation of the derivatives of the Lagrangian functional, and it will be tackled in section 3.2.3.2. Then the following holds true:

- Considering the primal variable $u$, we have:

$$
\frac{\partial \mathcal{L}}{\partial \delta u}[\delta u] = -\int_{t_0}^{t_f} \int_{\Omega} \left( \frac{\partial}{\partial t} \delta u + \rho (u \cdot \nabla) \delta u + \rho (u \cdot \nabla) u \right) \cdot \hat{u} \, dx \, dt +
\int_{t_0}^{t_f} \int_{\Omega} \left( \rho \nabla \cdot (\delta u + \mu (\nabla \delta u)^t + \lambda (\nabla \cdot \delta u) I) \right) \cdot \hat{u} \, dx \, dt +
\int_{t_0}^{t_f} \int_{\Omega} (\nabla \cdot \delta u + \chi_T \delta u \cdot \nabla p) \cdot \hat{p} \, dx \, dt +
\sum_{j=1}^M \int_{t_0}^{t_f} \int_{\Omega} \delta u \cdot \nabla \varphi \hat{\varphi} \, dx \, dt +
\int_{t_0}^{t_f} \int_{\Gamma_D} \delta u \cdot \hat{\theta}_1 \, ds \, dt +
\int_{t_0}^{t_f} \int_{\Gamma_N} (\mu \nabla \delta u + \mu (\nabla \delta u)^t + \lambda (\nabla \cdot \delta u) I) \cdot \nu \hat{\theta}_5 \, ds \, dt +
\int_{\Omega} \delta u |_{t=t_0} \cdot \hat{\theta}_2 \, dx = 0 \quad \forall \delta u \in \mathcal{U};
$$
• considering \( p \), we have:
\[
\frac{\partial \mathcal{L}}{\partial p} [\delta p] = - \int_{t_0}^{t_f} \int_{\Omega} \nabla \delta p \cdot \dot{u} \, dx \, dt + \int_{t_0}^{t_f} \int_{\Omega} \left( \chi T \frac{\partial \delta p}{\partial t} + \chi T u \cdot \nabla \delta p \right) \hat{\rho} \, dx \, dt + \int_{t_0}^{t_f} \int_{\Gamma_N} \delta p \hat{\nu} \, ds \, dt + \int_{t_0}^{t_f} \int_{\Omega} \delta \rho |_{t=t_0} \hat{\theta}_3 \, dx = 0 \quad \forall \delta p \in \mathcal{P};
\]

• considering \( \varphi \), we have:
\[
\frac{\partial \mathcal{L}}{\partial \varphi} [\delta \varphi] = \sigma_1 \int_{t_0}^{t_f} \int_{\Omega} \left( \dot{\varphi} |_{t=t_f} - \dot{\varphi} \right) \delta \varphi |_{t=t_f} \, dx + \int_{t_0}^{t_f} \int_{\Omega} \left( \frac{\partial \delta \varphi}{\partial t} + u \cdot \nabla \delta \varphi \right) \hat{\varphi} \, dx \, dt + \int_{t_0}^{t_f} \int_{\Omega} \delta \varphi \, \delta \varphi \hat{H} \, dx \, dt + \int_{t_0}^{t_f} \int_{\Omega} \delta \varphi |_{t=t_0} \hat{\theta}_4 \, dx = 0 \quad \forall \delta \varphi \in \mathcal{F};
\]

• considering \( H \), we have
\[
\frac{\partial \mathcal{L}}{\partial H} [\delta H] = - \int_{t_0}^{t_f} \int_{\Omega} \delta H \hat{H} \, dx \, dt + \int_{t_0}^{t_f} \int_{\Omega} \left( - \frac{\mu_H}{\rho} + \rho_t \right) \delta H \hat{\rho} \, dx \, dt + \int_{t_0}^{t_f} \int_{\Omega} (\mu_\lambda - \mu_g) \delta H \hat{\mu} \, dx \, dt + \int_{t_0}^{t_f} \int_{\Omega} (\lambda_t - \lambda_g) \delta H \hat{\lambda} \, dx \, dt + \int_{t_0}^{t_f} \int_{\Omega} -\delta H \chi_T \, dx \, dt = 0 \quad \forall \delta H \in \mathcal{H};
\]
3.2. THE OPTIMAL CONTROL PROBLEM OF INTEREST

• considering \( \rho \), we have:

\[
\frac{\partial \mathcal{L}}{\partial \rho} [\delta \rho] = - \int_{t_0}^{t_f} \int_{\Omega} \left( \frac{\partial p}{\partial t} + \delta \rho \left( \frac{\partial \mathbf{u}}{\partial t} + \delta \rho \mathbf{g} - \delta \rho \mathbf{C} \right) \right) \cdot \mathbf{u} \, dx \, dt +
- \int_{t_0}^{t_f} \int_{\Omega} \delta \rho \hat{\rho} \, dx \, dt = 0 \quad \forall \delta \rho \in \mathcal{R} ;
\]

• considering \( \mu \), we have:

\[
\frac{\partial \mathcal{L}}{\partial \mu} [\delta \mu] = - \int_{t_0}^{t_f} \int_{\Omega} \left( - \nabla \cdot (\delta \mu \nabla \mathbf{u}) - \nabla \cdot (\delta \mu (\nabla \mathbf{u})^\dagger) \right) \cdot \mathbf{u} \, dx \, dt +
- \int_{t_0}^{t_f} \int_{\Gamma_N} (\delta \mu \nabla \mathbf{u} + \delta \mu (\nabla \mathbf{u})^\dagger) \cdot \mathbf{\nu} \hat{\theta}_5 \, ds \, dt +
- \int_{t_0}^{t_f} \int_{\Omega} \delta \mu \hat{\mu} \, dx \, dt = 0 \quad \forall \delta \mu \in \mathcal{M} ;
\]

• considering \( \lambda \), we have:

\[
\frac{\partial \mathcal{L}}{\partial \lambda} [\delta \lambda] = - \int_{t_0}^{t_f} \int_{\Omega} - \nabla \cdot (\delta \lambda (\nabla \cdot \mathbf{u}) \mathbf{I}) \cdot \mathbf{u} \, dx \, dt +
- \int_{t_0}^{t_f} \int_{\Gamma_N} \delta \lambda (\nabla \cdot \mathbf{u}) \mathbf{I} \cdot \mathbf{\nu} \hat{\theta}_5 \, ds \, dt +
- \int_{t_0}^{t_f} \int_{\Omega} \delta \lambda \hat{\lambda} \, dx \, dt = 0 \quad \forall \delta \lambda \in \mathcal{N} ;
\]

• considering \( \chi_T \), we have:

\[
\frac{\partial \mathcal{L}}{\partial \chi_T} [\delta \chi_T] = - \int_{t_0}^{t_f} \int_{\Omega} \delta \chi_T \left( \frac{\partial p}{\partial t} + \mathbf{u} \cdot \nabla p \right) \hat{p} \, dx \, dt +
- \int_{t_0}^{t_f} \int_{\Omega} p \delta \chi_T \hat{\chi}_T \, dx \, dt = 0 \quad \forall \delta \chi_T \in \mathcal{X}_T .
\]

Let us now integrate by parts all the derivatives in order to free all the increment variables:

i) in \( \frac{\partial \mathcal{L}}{\partial \mathbf{u}} \) we have:

\[
- \int_{t_0}^{t_f} \int_{\Omega} p \frac{\partial \mathbf{u}}{\partial t} \hat{\mathbf{u}} \, dx \, dt = \int_{t_0}^{t_f} \int_{\Omega} \mathbf{u} \frac{\partial}{\partial t} (p \hat{\mathbf{u}}) \, dx \, dt +
- \int_{\Omega} \left[ (p \hat{\mathbf{u}} \cdot \mathbf{u})_{t=t_f} - (p \hat{\mathbf{u}} \cdot \mathbf{u})_{t=t_0} \right] \, dx ;
\]
• \(- \int_{t_0}^{t_f} \int_{\Omega} \rho (\mathbf{u} \cdot \nabla) \delta \mathbf{u} \cdot \hat{\mathbf{u}} + \rho (\delta \mathbf{u} \cdot \nabla) \mathbf{u} \cdot \hat{\mathbf{u}} \, dx \, dt = \) \\
= \int_{t_0}^{t_f} \int_{\Omega} -\rho (\nabla \mathbf{u})' \hat{\mathbf{u}} \cdot \delta \mathbf{u} \, dx \, dt + \int_{t_0}^{t_f} \int_{\Omega} \rho (\mathbf{u} \cdot \nabla) \hat{\mathbf{u}} \cdot \delta \mathbf{u} \, dx \, dt + \int_{t_0}^{t_f} \int_{\partial \Omega} \rho (\mathbf{u} \cdot \mathbf{\nu}) (\hat{\mathbf{u}} \cdot \delta \mathbf{u}) \, ds \, dt ; \\

• \(- \int_{t_0}^{t_f} \int_{\Omega} -\nabla \cdot (\mu \nabla \delta \mathbf{u}) \cdot \hat{\mathbf{u}} \, dx \, dt = \int_{t_0}^{t_f} \int_{\Omega} \nabla \cdot (\mu \nabla \hat{\mathbf{u}}) \cdot \delta \mathbf{u} \, dx \, dt + \int_{t_0}^{t_f} \int_{\partial \Omega} \mu (\nabla \hat{\mathbf{u}}) \cdot \delta \mathbf{u} \, ds \, dt ; \\

• \(- \int_{t_0}^{t_f} \int_{\Omega} -\nabla \cdot (\mu (\nabla \delta \mathbf{u})') \cdot \hat{\mathbf{u}} \, dx \, dt = \int_{t_0}^{t_f} \int_{\Omega} \nabla \cdot (\mu (\nabla \hat{\mathbf{u}})') \cdot \delta \mathbf{u} \, dx \, dt + \int_{t_0}^{t_f} \int_{\partial \Omega} \mu (\nabla \hat{\mathbf{u}})' \cdot \delta \mathbf{u} \, ds \, dt ; \\

• \(- \int_{t_0}^{t_f} \int_{\Omega} -\nabla \cdot (\lambda (\nabla \cdot \mathbf{u}) \mathbf{I}) \cdot \hat{\mathbf{u}} \, dx \, dt = \int_{t_0}^{t_f} \int_{\Omega} \nabla (\lambda (\nabla \cdot \hat{\mathbf{u}})) \cdot \delta \mathbf{u} \, dx \, dt + \int_{t_0}^{t_f} \int_{\partial \Omega} \lambda (\nabla \cdot \hat{\mathbf{u}}) \delta \mathbf{u} \cdot \mathbf{\nu} \, ds \, dt ; \\

• \(- \int_{t_0}^{t_f} \int_{\Omega} -\nabla \cdot \mathbf{\delta u} \hat{p} \, dx \, dt = \int_{t_0}^{t_f} \int_{\Omega} \delta \mathbf{u} \cdot \nabla \hat{p} \, dx \, dt + \int_{t_0}^{t_f} \int_{\partial \Omega} \hat{p} \delta \mathbf{u} \cdot \mathbf{\nu} \, ds \, dt ; \\

\text{ii) in } \frac{\partial \mathcal{L}}{\partial \hat{p}}, \text{ we have:} \\

• \(- \int_{t_0}^{t_f} \int_{\Omega} \nabla \delta \hat{p} \cdot \hat{\mathbf{u}} \, dx \, dt = \int_{t_0}^{t_f} \int_{\Omega} \delta \hat{p} \nabla \cdot \hat{\mathbf{u}} \, dx \, dt - \int_{t_0}^{t_f} \int_{\partial \Omega} \delta \hat{p} \hat{\mathbf{u}} \cdot \mathbf{\nu} \, ds \, dt ; \\

• \(- \int_{t_0}^{t_f} \int_{\Omega} \chi_T \frac{\partial \delta \hat{p}}{\partial t} \hat{p} \, dx \, dt = \int_{t_0}^{t_f} \int_{\Omega} \delta \hat{p} \frac{\partial}{\partial t} (\chi_T \hat{p}) \, dx \, dt + \int_{t_0}^{t_f} \int_{\partial \Omega} (\chi_T \delta \hat{p})|_{t=t_f} - (\chi_T \delta \hat{p})|_{t=t_0} \, dx ; \\

• \(- \int_{t_0}^{t_f} \int_{\Omega} \chi_T \mathbf{u} \cdot \nabla \delta \hat{p} \, dx \, dt = \int_{t_0}^{t_f} \int_{\Omega} \nabla \cdot (\chi_T \hat{p} \mathbf{u}) \delta \mathbf{p} \, dx \, dt + \int_{t_0}^{t_f} \int_{\partial \Omega} \chi_T \hat{p} \delta \mathbf{u} \cdot \mathbf{\nu} \, ds \, dt ; \\

\text{iii) in } \frac{\partial \mathcal{L}}{\partial \hat{\varphi}}, \text{ we have:} \\

• \(- \int_{t_0}^{t_f} \int_{\Omega} \frac{\partial \delta \hat{\varphi}}{\partial t} \hat{\varphi} \, dx \, dt = \int_{t_0}^{t_f} \int_{\Omega} \delta \hat{\varphi} \frac{\partial \hat{\varphi}}{\partial t} \, dx \, dt + \int_{\Omega} \left[ (\delta \hat{\varphi})|_{t=t_f} - (\delta \hat{\varphi})|_{t=t_0} \right] \, dx ; \)
3.2. THE OPTIMAL CONTROL PROBLEM OF INTEREST

\[ - \int_{t_0}^{t_f} \int_{\Omega} \mathbf{u} \cdot \nabla \delta \varphi \, d\mathbf{x} \, dt = \int_{t_0}^{t_f} \int_{\Omega} \nabla \cdot (\delta \mathbf{u}) \delta \varphi \, d\mathbf{x} \, dt + \]
\[ - \int_{t_0}^{t_f} \int_{\partial \Omega} \varphi \delta \varphi \mathbf{u} \cdot \mathbf{\nu} \, ds \, dt ; \]

iv) in \( \frac{\partial \mathcal{L}}{\partial \mu} \) we have:

\[ - \int_{t_0}^{t_f} \int_{\Omega} -\nabla \cdot (\delta \mu \nabla \mathbf{u}) \cdot \mathbf{\hat{u}} \, d\mathbf{x} \, dt = - \int_{t_0}^{t_f} \int_{\Omega} \delta \mu \nabla \mathbf{u} : \nabla \mathbf{\hat{u}} \, d\mathbf{x} \, dt + \]
\[ + \int_{t_0}^{t_f} \int_{\partial \Omega} \delta \mu (\nabla \mathbf{u} \cdot \mathbf{\nu}) \cdot \mathbf{\hat{u}} \, ds \, dt ; \]

\[ - \int_{t_0}^{t_f} \int_{\Omega} -\nabla \cdot (\delta \mu (\nabla \mathbf{u})^t) \cdot \mathbf{\hat{u}} \, d\mathbf{x} \, dt = - \int_{t_0}^{t_f} \int_{\Omega} \delta \mu (\nabla \mathbf{u})^t : \nabla \mathbf{\hat{u}} \, d\mathbf{x} \, dt + \]
\[ + \int_{t_0}^{t_f} \int_{\partial \Omega} \delta \mu ((\nabla \mathbf{u})^t \cdot \mathbf{\nu}) \cdot \mathbf{\hat{u}} \, ds \, dt ; \]

v) in \( \frac{\partial \mathcal{L}}{\partial \lambda} \) we have:

\[ - \int_{t_0}^{t_f} \int_{\Omega} -\nabla \cdot (\delta \lambda (\nabla \cdot \mathbf{u}) \mathbf{I}) \cdot \mathbf{\hat{u}} \, d\mathbf{x} \, dt = - \int_{t_0}^{t_f} \int_{\Omega} \delta \lambda ((\nabla \cdot \mathbf{u}) \mathbf{I}) : \nabla \mathbf{\hat{u}} \, d\mathbf{x} \, dt + \]
\[ + \int_{t_0}^{t_f} \int_{\partial \Omega} \delta \lambda (((\nabla \cdot \mathbf{u}) \mathbf{I}) \cdot \mathbf{\nu}) \cdot \mathbf{\hat{u}} \, ds \, dt ; \]

3.2.3.2 The surface tension term

Let us now tackle the derivative of the surface tension term, which in the lagrangian functional reads:

\[ - \int_{t_0}^{t_f} \int_{\Omega} -s \cdot \mathbf{\hat{u}} \, d\mathbf{x} \, dt . \quad (3.25) \]

Using (1.13) and (1.14), expression (3.25) can be recast as

\[ - \int_{t_0}^{t_f} \int_{\Omega} -\left( \zeta \left( -\nabla \cdot \left( \frac{\nabla \varphi}{\|\nabla \varphi\|} \right) \frac{\nabla \varphi}{\|\nabla \varphi\|} \delta \varepsilon (\varphi) \right) \right) \cdot \mathbf{\hat{u}} \, d\mathbf{x} \, dt , \quad (3.26) \]

where the Dirac function in (1.14) was replaced with its smooth approximation (1.20) in order to be able to differentiate it. As this extra term depends only on \( \varphi \) and \( \mathbf{\hat{u}} \), its derivative, which will be computed in this section, will have to be added to the right hand side of equation (3.24).
The derivative of (3.26) reads:

\[
\frac{\partial}{\partial u} \left( - \int_0^T \int_{\Omega} -s \cdot \hat{u} \, dx \, dt \right) [\delta \varphi] =
\]

\[
= - \int_0^T \int_{\Omega} -\varsigma \frac{\partial}{\partial \varphi} \left( -\nabla \cdot \frac{\nabla \varphi}{\| \nabla \varphi \|} \right) \left[ \frac{\nabla \varphi}{\| \nabla \varphi \|} \delta_{\varepsilon} (\varphi) \cdot \hat{u} \, dx \, dt + \right.
\]

\[
\int_0^T \int_{\Omega} -\varsigma \frac{\partial}{\partial \varphi} \left( -\nabla \cdot \frac{\nabla \varphi}{\| \nabla \varphi \|} \right) \cdot \frac{\partial}{\partial \varphi} \left( \frac{\nabla \varphi}{\| \nabla \varphi \|} \right) \left[ \delta \varphi \right] \delta_{\varepsilon} (\varphi) \cdot \hat{u} \, dx \, dt +
\]

\[
\left. \int_0^T \int_{\Omega} -\varsigma \left( -\nabla \cdot \frac{\nabla \varphi}{\| \nabla \varphi \|} \right) \cdot \frac{\nabla \varphi}{\| \nabla \varphi \|} \frac{\partial}{\partial \varphi} \left( \delta_{\varepsilon} (\varphi) \right) \left[ \delta \varphi \right] \cdot \hat{u} \, dx \, dt \right) \tag{3.27}
\]

Before we get into the details of the expansion of each single part of (3.27), let us compute some common derivatives that will be used extensively in the rest of this section. First of all, it holds that

\[
\frac{\partial}{\partial \varphi} \left( \frac{\nabla \varphi}{\| \nabla \varphi \|} \right) [\delta \varphi] = \frac{\nabla \delta \varphi}{\| \nabla \varphi \|} - \left( \frac{\nabla \delta \varphi \cdot \nabla \varphi}{\| \nabla \varphi \| \cdot \| \nabla \varphi \|} \right) \frac{\nabla \varphi}{\| \nabla \varphi \|}.
\]

Secondly, by direct derivation of (1.20),

\[
\frac{\partial}{\partial \varphi} \delta_{\varepsilon} (\varphi) [\delta \varphi] = \begin{cases} 
0 & \text{if } |\varphi| > \varepsilon \\
\frac{\pi^2}{2\varepsilon^2} \cos \left( \frac{\pi \varepsilon}{2} \right) & \text{otherwise.}
\end{cases}
\]

We will denote this quantity by \( \delta_{\varepsilon}' (\varphi) \).

Let us now compute the derivatives of all the terms in (3.27), starting with (\( \Delta 1 \)). Let

\[
R_{(\Delta 1)} := -\varsigma \frac{\nabla \varphi}{\| \nabla \varphi \|} \delta_{\varepsilon} (\varphi) \cdot \hat{u}.
\]
3.2. THE OPTIMAL CONTROL PROBLEM OF INTEREST

Then the following chain of equivalences holds:

\[
\begin{align*}
(\Delta 1) &= -\int_{t_0}^{t_f} \int_{\Omega} R_{(\Delta 1)} \frac{\partial}{\partial \varphi} \left( -\nabla \cdot \frac{\nabla \varphi}{\|\nabla \varphi\|} \right) [\delta \varphi] \, dx \, dt = \\
&= \int_{t_0}^{t_f} \int_{\Omega} R_{(\Delta 1)} \nabla \cdot \left( \frac{\partial}{\partial \varphi} \left( \frac{\nabla \varphi}{\|\nabla \varphi\|} \right) [\delta \varphi] \right) \, dx \, dt = \\
&\quad \text{by parts} - \int_{t_0}^{t_f} \int_{\Omega} \nabla R_{(\Delta 1)} \frac{\partial}{\partial \varphi} \left( \frac{\nabla \varphi}{\|\nabla \varphi\|} \right) [\delta \varphi] \, dx \, dt = \\
&\quad = -\int_{t_0}^{t_f} \int_{\Omega} \nabla R_{(\Delta 1)} \left( \nabla \delta \varphi \left( \frac{\nabla \varphi}{\|\nabla \varphi\|} - \left( \frac{\nabla \varphi}{\|\nabla \varphi\|} \right) \right) \right) \, dx \, dt = \\
&\quad \quad \text{by parts} - \int_{t_0}^{t_f} \int_{\Omega} \nabla \cdot \left( \frac{\nabla R_{(\Delta 1)}}{\|\nabla \varphi\|^2} \right) \delta \varphi \, dx \, dt - \int_{t_0}^{t_f} \int_{\Omega} \delta \varphi \frac{\nabla R_{(\Delta 1)}}{\|\nabla \varphi\|^2} \cdot \nu \, ds \, dt + \\
&\quad \quad \quad - \int_{t_0}^{t_f} \int_{\Omega} \nabla \cdot \left( \frac{\nabla R_{(\Delta 1)}}{\|\nabla \varphi\|^3} \right) \delta \varphi \, dx \, dt + \\
&\quad \quad \quad \quad + \int_{t_0}^{t_f} \int_{\delta \Omega} \delta \varphi \frac{\nabla R_{(\Delta 1)}}{\|\nabla \varphi\|^2} \cdot \nu \, ds \, dt
\end{align*}
\]

Notice that the boundary term is equal to zero in the first integration by parts, since \( \hat{u} \big|_{\delta \Omega} = 0 \), which also implies

\[
\nabla R_{(\Delta 1)} \cdot \nu = -\frac{\nabla \varphi}{\|\nabla \varphi\|} \delta_{\epsilon} (\varphi) \nabla \hat{u} \cdot \nu.
\]

Finally, we have that \((\Delta 1)\) yields

\[
\begin{align*}
\int_{t_0}^{t_f} \int_{\Omega} \nabla \cdot \left( \nabla \left( -\frac{\nabla \varphi}{\|\nabla \varphi\|} \delta_{\epsilon} (\varphi) \cdot \hat{u} \right) \frac{1}{\|\nabla \varphi\|} \left( 1 - \frac{(\nabla \varphi)^2}{\|\nabla \varphi\|^2} \right) \right) \delta \varphi \, dx \, dt + \\
- \int_{t_0}^{t_f} \int_{\delta \Omega} \frac{\nabla \varphi}{\|\nabla \varphi\|} \delta_{\epsilon} (\varphi) \nabla \hat{u} \cdot \nu \left( 1 - \frac{(\nabla \varphi)^2}{\|\nabla \varphi\|^2} \right) \, ds \, dt,
\end{align*}
\]

which is equal to

\[
\begin{align*}
\int_{t_0}^{t_f} \int_{\Omega} \nabla \cdot \left( \nabla \left( -\frac{\nabla \varphi}{\|\nabla \varphi\|} \delta_{\epsilon} (\varphi) \cdot \hat{u} \right) \frac{1}{\|\nabla \varphi\|} \left( 1 - \frac{(\nabla \varphi)^2}{\|\nabla \varphi\|^2} \right) \right) \delta \varphi \, dx \, dt,
\end{align*}
\]

since \( \delta_{\epsilon} (\varphi) \big|_{\delta \Omega} = 0 \).

Secondly, let us consider \((\Delta 2)\) and let

\[
R_{(\Delta 2)} := -\nabla \cdot \left( -\frac{\nabla \varphi}{\|\nabla \varphi\|} \right) \delta_{\epsilon} (\varphi) \cdot \hat{u}.
\]
Similarly to what was done for \( \triangle 1 \), we have:

\[
(\triangle 2) = -\int_{t_0}^{t_f} \int_{\Omega} R(\triangle 2) \frac{\partial}{\partial \varphi} \left( \frac{\nabla \varphi}{\| \nabla \varphi \|} \right) [\delta \varphi] \, dx \, dt = \\
= -\int_{t_0}^{t_f} \int_{\Omega} R(\triangle 2) \left( \nabla \delta \varphi - \left( \frac{\nabla \delta \varphi}{\| \nabla \varphi \|} \right) \nabla \varphi \right) \, dx \, dt = \\
\text{by parts} \int_{t_0}^{t_f} \int_{\Omega} \nabla \cdot \left( \nabla R(\triangle 2) \frac{\nabla \varphi}{\| \nabla \varphi \|} \right) \delta \varphi \, dx \, dt - \int_{t_0}^{t_f} \int_{\Omega} \nabla \cdot \left( \frac{\nabla R(\triangle 2) (\nabla \varphi)^2}{\| \nabla \varphi \|^3} \right) \delta \varphi \, dx \, dt,
\]

where, once again, the boundary term in the integration by parts, \( \hat{u} |_{\partial \Omega} = 0 \). We then have that \( \triangle 2 \) yields

\[
\int_{t_0}^{t_f} \int_{\Omega} \nabla \cdot \left( -\varsigma \left( -\nabla \cdot \left( \frac{\nabla \varphi}{\| \nabla \varphi \|} \right) \right) \frac{\nabla \varphi}{\| \nabla \varphi \|} \right) \delta \varphi \, dx \, dt.
\]

Lastly,

\[
(\triangle 3) = -\int_{t_0}^{t_f} \int_{\Omega} -\varsigma \left( -\nabla \cdot \left( \frac{\nabla \varphi}{\| \nabla \varphi \|} \right) \right) \frac{\nabla \varphi}{\| \nabla \varphi \|} \delta \varphi \, dx \, dt.
\]

We can now reassemble all the pieces to get:

\[
\frac{\partial}{\partial \theta} \left( -\int_{t_0}^{t_f} \int_{\Omega} -\varsigma \cdot \hat{u} \, dx \, dt \right) [\delta \varphi] = \int_{t_0}^{t_f} \int_{\Omega} R(\varphi, \hat{u}) \delta \varphi \, dx \, dt,
\]

with

\[
R(\varphi, \hat{u}) = \nabla \cdot \left( \left[ -\nabla \left( \varsigma \frac{\nabla \varphi}{\| \nabla \varphi \|} \delta \varepsilon(\varphi) \hat{u} \right) + \\
+ \varsigma \nabla \cdot \left( \frac{\nabla \varphi}{\| \nabla \varphi \|} \right) \delta \varepsilon(\varphi) \hat{u} \right) \frac{1}{\| \nabla \varphi \|} \left( 1 - \left( \frac{\nabla \varphi}{\| \nabla \varphi \|} \right)^2 \right) \right) + \\
- \varsigma \nabla \cdot \left( \frac{\nabla \varphi}{\| \nabla \varphi \|} \right) \left( \frac{\nabla \varphi}{\| \nabla \varphi \|} \right) \delta \varepsilon(\varphi) \hat{u},
\]

which is exactly the definition used in section 3.2.1.1. Equation (3.24) then reads:

\[
\frac{\partial \mathcal{L}}{\partial \delta \varphi} [\delta \varphi] = \sigma_1 \int_{t_0}^{t_f} \left( \varphi |_{t=t_f} - \bar{\varphi} \right) \delta \varphi |_{t=t_f} \, dx + \\
- \int_{t_0}^{t_f} \int_{\Omega} \left( \frac{\partial \delta \varphi}{\partial t} + \mathbf{u} \cdot \nabla \delta \varphi \right) \hat{\varphi} \, dx \, dt + \\
- \int_{t_0}^{t_f} \int_{\Omega} \delta(\varphi) \delta \varphi \delta \hat{H} \, dx \, dt + \\
+ \int_{t_0}^{t_f} \int_{\Omega} R(\varphi, \hat{u}) \delta \varphi \, dx \, dt + \\
- \int_{\Omega} \left. \delta \varphi |_{t=t_0} \right| \delta \lambda_4 \, dx = 0 \quad \forall \delta \varphi \in \mathcal{F},
\]
and it will be used with this form in the next section.

### 3.2.3.3 The adjoint problem

Putting it all back together, we have that the system composed of the following equations must hold:

- $\frac{\partial \mathcal{L}}{\partial u} [\delta u] = \int_{t_0}^{t_f} \int_{\Omega} \left( \frac{\partial}{\partial t} (\rho \hat{u}) - \rho (\nabla u)^T \cdot \hat{u} + \rho (u \cdot \nabla) \hat{u} + \nabla \cdot (\mu \nabla \hat{u} + \mu (\nabla u)^T) + \nabla \cdot \left( \lambda \nabla \cdot \hat{u} \right) + \nabla \hat{p} - \chi T \hat{p} \nabla p - \nabla \varphi \hat{\phi} \right) \cdot \delta u \, dx \, dt +$

- $+ \int_{t_0}^{t_f} \int_{\Gamma_D} \left( -\theta_1 - \rho (u \cdot \nu) \hat{u} - \mu (\nabla \hat{u} \cdot \nu) - \mu ((\nabla \hat{u})^T \nu) - \hat{p} \nu - \lambda (\nabla \cdot \hat{u}) \nu \right) \cdot \delta u \, ds \, dt +$

- $+ \int_{t_0}^{t_f} \int_{\Gamma_N} -\rho (u \cdot \nu) \hat{u} - \mu (\nabla \hat{u} \cdot \nu) - \mu ((\nabla \hat{u})^T \nu) - \hat{p} \nu +$

- $- \lambda (\nabla \cdot \hat{u}) \nu \right) \cdot \delta u \, ds \, dt +$

- $+ \int_{t_0}^{t_f} \int_{\Gamma_D} \mu (\nabla \delta u \cdot \nu + (\nabla \delta u)^T \nu) \cdot \hat{u} \, ds \, dt$

- $+ \int_{t_0}^{t_f} \int_{\Gamma_N} -\mu (\nabla \delta u \cdot \nu + (\nabla \delta u)^T \nu) \cdot \left( \hat{u} + \hat{\theta}_5 \right) \, ds \, dt$

- $+ \int_{t_0}^{t_f} \int_{\Gamma_D} \lambda (\nabla \cdot \delta u) \hat{u} \cdot \nu \, ds \, dt +$

- $+ \int_{t_0}^{t_f} \int_{\Gamma_N} -\lambda (\nabla \cdot \delta u) \left( \hat{u} + \hat{\theta}_5 \right) \cdot \nu \, ds \, dt +$

- $+ \int_{\Omega} \left( (\rho \hat{u})_{t=t_0} \right) \cdot \delta u |_{t=t_0} \, dx +$

- $+ \int_{\Omega} - (\rho \hat{u})_{t=t_f} \cdot \delta u |_{t=t_f} \, dx = 0 \quad \forall \delta u \in \mathcal{W}$

- $\frac{\partial \mathcal{L}}{\partial p} [\delta p] =$

- $\int_{t_0}^{t_f} \int_{\Omega} \left( \nabla \cdot \hat{u} + \frac{\partial}{\partial t} \left( \chi T \hat{p} \right) + \nabla \cdot \left( \chi T u \hat{p} \right) - \frac{H}{rT} \hat{p} - \chi T \hat{T} \right) \delta p \, dx \, dt +$

- $- \int_{t_0}^{t_f} \int_{\partial \Omega} (\hat{u} + \chi T \hat{p} u) \cdot \nu \, \delta p \, ds \, dt +$

- $+ \int_{t_0}^{t_f} \int_{\Gamma_N} \delta p \cdot \nu \hat{\theta}_5 \, ds \, dt +$

- $+ \int_{\Omega} \left( (\chi T \hat{p})_{t=t_0} - \hat{\theta}_3 \right) \cdot \delta p |_{t=t_0} \, dx +$

- $- \int_{\Omega} (\chi T \hat{p})_{t=t_f} \delta p |_{t=t_f} \, dx = 0 \quad \forall \delta p \in \mathcal{P}.$
choosing such variations properly.

For admissible variation, we can derive the continuous form of the adjoint system by

$$
\frac{\partial L}{\partial \varphi} [\delta \varphi] = \int_{t_0}^{t_f} \int_{\Omega} \left( \frac{\partial \hat{\varphi}}{\partial t} + \nabla \cdot (\hat{\varphi} \mathbf{u}) - \delta (\varphi) \hat{H} + R(\varphi, \hat{\mathbf{u}}) \right) \delta \varphi \, dx \, dt +
+ \int_{t_0}^{t_f} \int_{\partial \Omega} -\hat{\varphi} \mathbf{u} \cdot \mathbf{n} \, \delta \varphi \, ds \, dt +
+ \int_{\Omega} \left( \sigma_1 \left( \varphi|_{t=t_f} - \hat{\varphi} \right) - \delta \varphi|_{t=t_f} \right) \delta \varphi|_{t=t_f} \, dx +
+ \int_{\Omega} \left( \hat{\varphi}|_{t=t_0} - \hat{\vartheta}_l \right) \delta \varphi|_{t=t_0} \, dx = 0 \quad \forall \delta \varphi \in \mathscr{F};
$$

$$
\frac{\partial L}{\partial H} [\delta H] = \int_{t_0}^{t_f} \int_{\Omega} \left( -\hat{H} - \left( \hat{\rho} - \frac{p}{\rho T} \right) \hat{\rho} +
- (\mu_l - \mu_g) \hat{\rho} - (\lambda_l - \lambda_g) \hat{\lambda} + \hat{\chi}_T \right) \delta H \, dx \, dt = 0 \quad \forall \delta H \in \mathcal{H};
$$

$$
\frac{\partial L}{\partial \rho} [\delta \rho] = \int_{t_0}^{t_f} \int_{\Omega} \left( -\hat{\rho} \mathbf{u} \cdot \mathbf{n} - \left( \mathbf{u} \cdot \nabla \right) \mathbf{u} \cdot \hat{\mathbf{u}} + \mathbf{g} \cdot \hat{\mathbf{u}} +
+ \mathbf{C} \cdot \hat{\mathbf{u}} - \hat{\rho} \right) \delta \rho \, dx \, dt = 0 \quad \forall \delta \rho \in \mathcal{R};
$$

$$
\frac{\partial L}{\partial \mu} [\delta \mu] = \int_{t_0}^{t_f} \int_{\Omega} \left( -\nabla \mathbf{u} : \nabla \hat{\mathbf{u}} - \left( \nabla \mathbf{u} \right)^T : \nabla \hat{\mathbf{u}} - \hat{\mu} \right) \delta \mu \, dx \, dt +
+ \int_{t_0}^{t_f} \int_{\partial \Omega} \delta \mu \left( \nabla \mathbf{u} \cdot \mathbf{n} + \left( \nabla \mathbf{u} \right)^T \cdot \mathbf{n} \right) \hat{\mathbf{u}} \, ds \, dt +
- \int_{t_0}^{t_f} \int_{\Gamma_N} \delta \mu \left( \nabla \mathbf{u} \cdot \mathbf{n} + \left( \nabla \mathbf{u} \right)^T \cdot \mathbf{n} \right) \hat{\mathbf{g}} \, ds \, dt = 0 \quad \forall \delta \mu \in \mathcal{M};
$$

$$
\frac{\partial L}{\partial \lambda} [\delta \lambda] = \int_{t_0}^{t_f} \int_{\Omega} \left( -\left( \left( \nabla \cdot \mathbf{u} \right) I : \nabla \hat{\mathbf{u}} - \hat{\lambda} \right) \right) \delta \lambda \, dx \, dt +
+ \int_{t_0}^{t_f} \int_{\partial \Omega} \delta \lambda \left( \left( \nabla \cdot \mathbf{u} \right) I \cdot \mathbf{n} \right) \hat{\mathbf{u}} \, ds \, dt +
- \int_{t_0}^{t_f} \int_{\Gamma_N} \delta \lambda \left( \left( \nabla \cdot \mathbf{u} \right) I \right. \cdot \mathbf{n} \, \hat{\mathbf{g}} \, ds \, dt = 0 \quad \forall \delta \lambda \in \mathcal{N};
$$

$$
\frac{\partial L}{\partial \chi_T} [\delta \chi_T] = \int_{t_0}^{t_f} \int_{\Omega} \left( -p \hat{\chi}_T - \hat{p} \left( \frac{\partial \mathbf{p}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{p} \right) \right) \delta \chi_T \, dx \, dt = 0 \quad \forall \delta \chi_T \in \mathcal{X}_T.
$$

Now, since the variations are arbitrary and the equations must hold for every admissible variation, we can derive the continuous form of the adjoint system by choosing such variations properly.
3.2. THE OPTIMAL CONTROL PROBLEM OF INTEREST

The adjoint variables in which we are interested solve the following set of equations:

\[
\begin{cases}
- \frac{\partial}{\partial t} (\rho \dot{u}) + \rho (\nabla u)^T \cdot \dot{u} - \rho (u \cdot \nabla) \dot{u} - \nabla \cdot (\mu \nabla \mu + \mu (\nabla \mu)^T) + \\
\nabla \cdot \dot{u} = - \frac{\partial}{\partial t} (\chi_T \ddot{p}) - \nabla \cdot (\chi_T u \dot{p}) + \frac{H}{rT} \ddot{p} + \chi_T \dot{\chi_T} \\
- \frac{\partial}{\partial t} \ddot{\varphi} - \nabla \cdot (\dot{\varphi} u) = - \delta (\varphi) \ddot{H} + R (\varphi, \dot{u}) \\
\dot{H} = \left( \frac{p}{rT} - \rho \right) \dot{\rho} + (\mu_g - \mu_u) \dot{\mu} + \\
\quad + (\lambda_g - \lambda_u) \ddot{\lambda} + \dddot{\chi_T} \\
\dddot{\rho} = - \frac{\partial}{\partial u} \ddot{u} - (u \cdot \nabla) u \cdot \ddot{u} + g \cdot \ddot{u} + C \cdot \ddot{u} \\
\dddot{\lambda} = - (\nabla \cdot u) I : \nabla \ddot{u} \\
p \dddot{\chi_T} = - \dddot{\rho} \left( \frac{\partial}{\partial p} + u \cdot \nabla p \right)
\end{cases}
\]

in \( \Omega \times [t_0, t_f] \) with boundary conditions:

\[
\begin{cases}
\dot{u} = 0 & \text{on } \Gamma_D, \\
- (\mu \nabla \dot{u} + \mu (\nabla \dot{u})^T) \cdot \nu - (\lambda \nabla \cdot \dot{u}) \nu - \ddot{p} \nu - \rho (u \cdot \nabla) \dot{u} = 0 & \text{on } \Gamma_N
\end{cases}
\]

and final conditions:

\[
\begin{cases}
\dot{u} |_{t=t_f} = 0 & \text{in } \Omega, \\
\dddot{\rho} |_{t=t_f} = 0 & \text{in } \Omega, \\
\dddot{\varphi} |_{t=t_f} = \sigma_1 (\varphi |_{t=t_f} - \dddot{\varphi}) & \text{in } \Omega,
\end{cases}
\]

which is exactly system (3.16).

3.2.4 Space-time discretization and minimization algorithm

The discretization for the state problems (3.21) and (3.14) has already been discussed in sections 1.2.3 and 1.3.3.2, respectively. Let us briefly show how the adjoint problems have been solved. We will use the (more complex) compressible-incompressible model as an example, but the same approach can be applied to the incompressible-incompressible adjoint system just as well. The same notation and definitions introduced in section 1.2.3 will be used.

Notice that, as it is the case for all adjoint problems, the structure of system (3.16) closely resembles that of its primal. As in section 1.3.2.2, a staggered method is then needed to solve system (3.16). Given the similarity between systems (3.14) and (3.16),
we will use the same approach as in steps 1B-6B in section 1.3.2.2 to decouple the equations in (3.16). Notice also that, as seen in section 3.1.3, the adjoint equations are to be solved backwards in time, i.e. from $t = t_f$ to $t = t_0$ with negative time steps. Again, as in section 3.1.3, we will still always suppose $\Delta t$ to be positive.

The time discretization goes as follows. Given the final conditions $\hat{\varphi}_h^{(N)}$, $\hat{\boldsymbol{u}}_h^{(N)}$, $\hat{\rho}_h^{(N)}$, $\hat{\mu}_h^{(N)}$, $\hat{\chi}_h^{(N)}$, and $\hat{\chi}_h^{(N)}$, for every $n = N - 1, N - 2, \ldots 0$, the solution process is:

1A. Compute $\hat{\boldsymbol{u}}_h^{(n)}$ and $\hat{\rho}_h^{(n)}$, solving the following discrete problem:

Find the pair $(\hat{\boldsymbol{u}}_h^{(n)}, \hat{\rho}_h^{(n)})$ in $\mathcal{V}_h^d \times Q_h$ such that $\boldsymbol{u}_h^{(n)} \big|_{\Gamma_D} = \mathbf{0}$ and for all pairs $(\mathbf{v}_h, q_h) \in [\mathcal{V}_h, \Gamma_D]^d \times Q_h$

\[
\int_{\Gamma_D} \frac{\hat{\rho}_h^{(n)}}{\Delta t} \hat{\boldsymbol{u}}_h^{(n)} \cdot \mathbf{v}_h \, \mathrm{d}x + \int_{\Omega} \rho_h^{(n)} \left( \nabla \hat{\boldsymbol{u}}_h^{(n)} \right)^T \hat{\boldsymbol{u}}_h^{(n)} \cdot \mathbf{v}_h \, \mathrm{d}x + \int_{\Omega} \mu_h^{(n)} \nabla \hat{\boldsymbol{u}}_h^{(n)} : \nabla \mathbf{v}_h \, \mathrm{d}x + \int_{\Omega} \lambda_h^{(n)} \nabla \cdot \hat{\boldsymbol{u}}_h^{(n)} \nabla \cdot \mathbf{v}_h \, \mathrm{d}x + \int_{\Omega} \nabla \cdot \hat{\boldsymbol{u}}_h^{(n)} q_h \, \mathrm{d}x + \int_{\Omega} \chi_h^{(n)} \nabla \cdot \hat{\rho}_h^{(n)} \cdot \mathbf{v}_h \, \mathrm{d}x + \int_{\Omega} \nabla \cdot \hat{\rho}_h^{(n)} q_h \, \mathrm{d}x - \int_{\Omega} \nabla \left( \chi_h^{(n)} \hat{\boldsymbol{u}}_h^{(n)} \hat{\varphi}_h^{(n)} \right) q_h \, \mathrm{d}x = -\frac{\hat{\rho}_h^{(n+1)}}{\Delta t} \hat{\boldsymbol{u}}_h^{(n+1)} \cdot \mathbf{v}_h \, \mathrm{d}x \quad \text{for all } \mathbf{v}_h \in \mathcal{V}_h^d, q_h \in Q_h.
\]

2A. Compute the adjoint signed distance function $\hat{\varphi}_h^{(n)}$, solving the following discrete problem:

Find $\hat{\varphi}_h^{(n)}$ in $W_h$ such that for all $w_h \in W_h$

\[
\sum_{E \in \mathcal{E}_h} \int_E \frac{\hat{\varphi}_h^{(n)}}{\Delta t} w_h \, \mathrm{d}x - \sum_{E \in \mathcal{E}_h} \int_E \hat{\varphi}_h^{(n)} \hat{\boldsymbol{u}}_h^{(n)} \cdot \nabla w_h \, \mathrm{d}x + \sum_{e \in \mathcal{E}_e} \int_{e} u_h^{(n)} \cdot \left\| w_h \right\| \{ \hat{\varphi}_h^{*} \} \, \mathrm{d}s + \sum_{e \in \mathcal{E}_0} \int_{e} \left( \left\{ \hat{\varphi}_h^{(n)} \right\} u_h^{(n)} \left\| w_h \right\| + \frac{1}{2} \left\| u_h^{(n)} \cdot \nu \right\| \left\| \hat{\varphi}_h^{(n)} \right\| \left\| w_h \right\| \right) \, \mathrm{d}s = -\sum_{E \in \mathcal{E}_h} \int_E \frac{\hat{\varphi}_h^{(n+1)}}{\Delta t} w_h \, \mathrm{d}x + \sum_{E \in \mathcal{E}_h} \int_{E} \left( \delta_{\nu} \left( \hat{\varphi}_h^{(n)} \right) + R \left( \hat{\varphi}_h^{(n)} , \hat{\boldsymbol{u}}_h^{(n)} \right) \right) w_h \, \mathrm{d}x,
\]

where

\[
\hat{\varphi}_h^{*} = \begin{cases} 
\hat{\varphi}_{D,h} & \text{if } u_h^{(n)} \cdot \nu > 0, \\
\hat{\varphi}_h^{(n)} & \text{if } u_h^{(n)} \cdot \nu \leq 0.
\end{cases}
\]

See [DLP06] and section 1.2.3 (in particular, step 3T) for further details.
3.2. THE OPTIMAL CONTROL PROBLEM OF INTEREST

Algorithm 5 Optimal control problem solution method

Let an initial guess $C^{(0)}$ and two positive constants $\varepsilon_g$ and $\varepsilon_J$ be given.

Set $k = 0$

while $\left\|C^{(k)} - C^{(k-1)}\right\|_{\varepsilon_g} > \varepsilon_g \land \left\|\nabla J \left(C^{(k+1)}\right)\right\|_{\varepsilon_g} > \varepsilon_J$ do

Solve system (3.14) using steps 1B-6B and the current control value $C^{(k)}$

Solve system (3.16) using steps 1A-5A and the newly computed solution of the state system

Set $d = -\nabla J \left(C^{(k)}\right)$

Compute step length:

begin

Let an initial guess $\alpha^{(0)}$ and two constants $c_1 > 0$ and $c_2 \in (0, 1)$ be given.

Solve system (3.14) using steps 1B-6B and $C^{(k)} + \alpha^{(0)}d$ as control value

Set $m = 0$

while $J \left(C^{(k)} + \alpha^{(m)}d\right) > J \left(C^{(k)}\right) + c_1 \alpha^{(m)} \left(\nabla J \left(C^{(k)}\right), d\right)$ do

Set $\alpha^{(m+1)} = c_2 \alpha^{(m)}$

Solve system (3.14) using steps 1B-6B and $C^{(k)} + \alpha^{(m+1)}d$ as control value

$m \leftarrow m + 1$

end while

Set $C^{(k+1)} = C^{(k)} + \alpha^{(m)}d$

$k \leftarrow k + 1$

end while

3A. Compute $\hat{H}^{(n)}_h$ using equation (3.16)4.

4A. Update the adjoint counterpart of the physical parameters $\hat{\rho}^{(n)}_h$, $\hat{\mu}^{(n)}_h$, $\hat{\lambda}^{(n)}_h$, $\hat{\chi}^{(n)}_{Th}$ using equations (3.16)5, (3.16)6, (3.16)7 and (3.16)8, respectively.

5A. If $n < N$, set $n = n + 1$ and go back to 1A.

Notice that the number $N$ of time steps used for the discretization of the primal and the adjoint systems is supposed to be the same. This entails that the primal solution at time $t^{(n)}$ can be directly used when the adjoint solution at that exact same time is computed (see steps 1A-5A).

Finally, as done in section 3.1.3, let us explicitly rewrite algorithm 3 for the optimal control problems defined in sections 3.2.1 and 3.2.2. Algorithm 5 shows the solution method for problem 3.2. The solution procedure for problem 3.3 is akin.
Chapter 4

Numerical results of the optimal control problems

Let us now show some numerical results for the control problems detailed in sections 3.2.1 and 3.2.2. We will begin by investigating the incompressible-incompressible case and then move on to the compressible-incompressible control problem. Indeed, the similarity between the incompressible-incompressible and the compressible-incompressible problems means that it is possible to use the former (simpler) formulation as an intermediate step towards the final goal. As in sections 1.2.4, 1.3.4 and 3.1.4, all dimensional quantities are expressed in the International System of Units.

With the aim of reducing the computational burden, all the numerical results in the present section have been obtained by employing an inexact version of the gradient method presented in section 2.2.3. More specifically, the adjoint systems (3.16) and (3.22) have been solved by setting to zero the term $R(\varphi, \hat{u})$. This choice clearly yields an approximated value for the gradient $\nabla \mathcal{C}J$. Moreover, in the numerical tests, mild viscosity and density ratios are considered in order to limit the numerical issues associated with large discontinuities in the coefficients (see section 1.2.4). Still, the numerical results shown in this section are paradigmatic and represent a first step towards the application of optimal control problems to more physically-relevant two-phase compressible-incompressible flows.

4.1 Incompressible-incompressible results

In this section the results for various optimal control problem on the two-phase incompressible-incompressible model (see section 3.2.2) are presented.

The same configuration (represented in figure 4.1) is used for all the tests shown in this section. The domain is a square with sides $L_1 = L_2 = 2$ and, in all of the test cases, it is discretized using a structured triangular grid with $N_h = 80$ nodes on each side. The bubble is initially at rest in $(x_c, y_c) = (1, 0.5)$ and has radius $r_0 = 0.3$. 
CHAPTER 4. NUMERICAL RESULTS OF THE CONTROL PROBLEMS

Figure 4.1: Initial configuration for the control problem in the incompressible-incompressible case

The final time is \( t_f = 2 \) and the time step is \( \Delta t = 0.05 \), which gives a number of time steps \( N = 40 \). The two fluids have density \( \rho_1 = 0.1 \) and \( \rho_2 = 1 \) respectively. Viscosities are set to \( \mu_1 = 0.1 \) and \( \mu_2 = 1 \). Gravity is \( \mathbf{g} = (0,0) \). The smoothing parameter \( \varepsilon \) in equations (1.19) and (1.20) is set to \( \varepsilon = 4h \), where \( h \) is the mesh size, so that the transition zone between the two fluids is four-cells thick.

As for the control parameters, the same set of values is used throughout all the cases shown. In particular, the chosen values for the penalization parameters in the cost functional (3.20) are \( \sigma_1 = 100 \) and \( \sigma_2 = 10^{-6} \).

Finally, the backtracking parameters \( \alpha, c_1 \) and \( c_2 \) (see section 2.2.3.1) have been set to 1, 0.001 and 0.5 respectively, which are standard values in the literature, while the thresholds for the convergence check are \( \varepsilon_g = \varepsilon_f = 10^{-6} \). The initial guess for the control function \( C(x,t) \) is

\[
C^{(0)}(x,t) = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \forall (x,t) \in \Omega \times [t_0, t_f].
\]

All the test cases considered enforce Dirichlet conditions on the whole domain boundary \( \partial \Omega \), and the optimality conditions used are given by a slightly modified version of (T-i)-(T-iii) (from section 3.2.2.1), where \( \Gamma_D = \partial \Omega \) and \( \Gamma_N = \emptyset \).
4.1. INCOMPRESSIBLE-INCOMPRESSIBLE RESULTS

4.1.1 Test case 1: constant target control reconstruction

In this first test case, we will try to recover a given target control. That is, a given known value $C_t$ is preliminarily used as the control variable in equation (3.21) to compute the target position of the bubbles. More precisely, the state system is solved once beforehand setting $C = C_t$ and the computed signed distance function $\varphi(x, t)$ at the final time is used as $\bar{\varphi}$. Then the optimal control problem is setup to drive the system towards such a desired configuration.

The chosen target control is:

$$C_t = \begin{pmatrix} 0 \\ -5 \end{pmatrix},$$

which makes the bubble move upward, as shown in figure 4.3. In this case, the optimal control problem 3.3 is solved in the set of admissible control containing constant 2D vectors. Since the control is constant in space and time, the gradient of the cost functional is given by:

$$\nabla_{C} J = \sigma_2 C + \int_{t_0}^{t_f} \int_{\Omega} \rho \tilde{u} d\Omega dt,$$

instead of (3.19).

The history of convergence obtained by employing algorithm 5 is plotted in figure 4.2, while figure 4.3 shows the time evolution of the bubble’s position once the obtained control is employed for the solution of the primal system. Such computed optimal control has value:

$$C^* = \begin{pmatrix} -1.07106 \times 10^{-5} \\ -4.9998 \end{pmatrix}.$$

First of all, let us notice that the targets are well recovered by the optimal solution, in terms of both the bubble’s position (figure 4.3) and the value of the control function $C^*$. This will not always be the case (see section 4.1.2). As for the history of convergence (figure 4.2), the plots for the the functional value and the gradient norm show a decrease in both quantities at each minimization iteration. Finally, the figure presenting the number of backtracking iterations shows that the adopted choice for the backtracking parameters is enough to satisfy Armijo’s condition.
CHAPTER 4. NUMERICAL RESULTS OF THE CONTROL PROBLEMS

Figure 4.2: Incompressible-incompressible test case 1, history of convergence

Figure 4.3: Incompressible-incompressible test case 1, optimal control problem results: controlled solution (orange) and target position (dashed black)
4.1.2 Test case 2: reconstruction of a given non-constant control

Expanding upon what we obtained in section 4.1.1, in this test case we will try to recover a target control which is not constant.

The chosen target control is:

\[
C_t(x, t) = \begin{pmatrix} \frac{5}{t_f} (t_f - t) x(L_1 - x)y(L_2 - y) \\ -\frac{5}{t_f} (t_f - t)^2 x(L_1 - x)y(L_2 - y) \end{pmatrix}, \quad \forall ((x, y), t) \in \Omega \times [t_0, t_f],
\]

where all the quantities are considered adimensional in this context. This expression was not picked completely at random. Indeed, some (hidden) constrains, consequence of the particular choices for the boundary conditions imposed on the adjoint velocity \( \hat{u} \) and for the initial guess for the optimal control, are to be valid in order for \( C_t \) to be an admissible control (that is, a control that can be reached using the framework detailed above). First of all, the control must vanish on the boundary of \( \Omega \). Indeed, the initial guess \( C^{(0)} \) is zero-valued on the boundary and so is the adjoint velocity, due to the boundary conditions enforced on it (see equation (3.22) and recall that \( \Gamma_N = \emptyset \)). Since the gradient of the cost functional, which will be used to compute the new guess for the optimal control at each minimization step, only takes into account the adjoint velocity and the current value of the control function (see equation (3.19)), \( C \) will always be equal to zero on the domain boundary. Secondly, using the same argument and recalling the final condition for the adjoint problem (see again equation (3.22)), we notice that the control \( C \) must vanish in \( \Omega \times \{t_f\} \) as well. The expression of the target control \( C_t \) above satisfies both these constraint, and therefore (4.1) does indeed define an admissible control. We can then finally substitute expression (4.1) for \( C \) in system (3.21) and obtain the evolution of the bubble’s position in time.

The history of convergence of the minimization algorithm is plotted in figure 4.4, while figure 4.5 shows the results of the optimal control problem.

First of all, let us consider figure 4.5. As a matter of fact, here lies the main difference between this test case and the one detailed in section 4.1.1. Indeed, figure 4.5 shows that the final position of the bubble is well recovered (as was in section 4.1.1), but, unlike the constant case, the computed control is far from equal to the target control. This is due to the fact that we are only forcing the bubble to be in a given position at the final time \( t_f \), while we enforce nothing on the position of the bubble in the time interval \( (t_0, t_f) \) itself. This, in combination with the use of a larger space than before for the admissible controls, results in the fact that the control found by the minimization algorithm is but one of the possible controls that drive the bubble to the given position (albeit following a different path). In other words, the solution is no longer unique, and we can simply find one of the many possible minimizers for problem 3.3.
Nonetheless, the optimal control problem does manage to find a solution. Indeed, the plots representing the history of convergence (figure 4.4) show a decrease in both the functional value and the norm of its gradient, and the target is recovered to satisfaction.
4.1. INCOMPRESSIBLE-INCOMPRESSIBLE RESULTS

Figure 4.5: Incompressible-incompressible test case 2, optimal control problem results. Top row: controlled solution (orange) and target position (dashed black). Middle row: computed optimal control. Bottom row: difference between the computed optimal control and the target optimal control $C_t$. 
4.2 Compressible-incompressible results

Let us now consider test cases based on the compressible-incompressible optimal control problem 3.2.

The same set of optimal control parameters as in section 4.1 (which are reported here for ease of reference) is used in all the cases shown in this section. In particular, we set the penalization parameters in equation (3.20) to $\sigma_1 = 100$ and $\sigma_2 = 10^{-6}$, while the backtracking parameters $\alpha$, $c_1$ and $c_2$ (see section 2.2.3.1) are equal to 1, 0.001 and 0.5, respectively, and the thresholds for the convergence check are $\varepsilon_g = \varepsilon_f = 10^{-6}$.

Finally, the initial guess for the control function $C(p, t)$ is

$$C^{(0)}(p, t) = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \forall (p, t) \in \Omega \times [t_0, t_f].$$

As for the physical parameters, the values chosen for the viscosity are $\mu_l = 1$ and $\mu_g = 0.1$, while the density is equal to $\rho_l = 10$ in the liquid and will have different values in the gas, depending on the test case. The specific gas constant is set to 300. The temperature is 300, unless stated otherwise. All the dimensional quantities are expressed in the International System of Units (SI). Lastly, the smoothing parameter $\varepsilon$ is set to $\varepsilon = 4h$, $h$ being the mesh size.

Notice that all the test cases shown, with the exception of section 4.2.2, enforce Dirichlet conditions on the whole domain boundary $\partial \Omega$. The optimality conditions employed in this case are given by a modified version of (B-i)-(B-iii) (see section 3.2.1.1), where $\Gamma_D = \partial \Omega$ and $\Gamma_N = \emptyset$.

4.2.1 Test case 1: constant target control reconstruction

This first compressible test case resembles that in section 4.1.1. Once again, a given known value is preliminarily used for $C$ in equation (3.14) to compute the target position of the bubbles. Then the optimal control problem is setup to drive the system towards such a desired configuration. The difference lies in the fact that here a two-bubble configuration is used and compressibility is considered.

The initial configuration is represented in figure 4.6. The domain is a square with sides $L_1 = L_2 = 2$ and it was discretized using a structured triangular grid with $N_h = 80$ nodes on each side. The two bubbles are initially at rest in the domain, with centers in (0.5,0.5) and (1.5,1.5), radiuses $r_1 = r_2 = 0.2$ and densities $\rho_1 = 1$ and $\rho_2 = 2$, corresponding to pressures $p_1 = 90 000$ and $p_2 = 180 000$, respectively. The final time is $t_f = 1$ and the time step is $\Delta t = 0.05$, which gives a number of time steps $N = 20$. No gravity is considered.
4.2. COMPRESSIBLE-INCOMPRESSIBLE RESULTS

The chosen target control is:

\[ C_t = \begin{pmatrix} 0 \\ -5 \end{pmatrix}. \]

As in section 4.1.1, the optimal control problem 3.2 is solved in the set of admissible control containing constant 2D vectors, so that the gradient of the cost functional is given by:

\[ \nabla C J = \sigma_2 C + \int_{t_0}^{t_f} \int_\Omega \rho \hat{u} \hat{d}x \, dt, \]

instead of (3.19).

The history of convergence for this test case is plotted in figure 4.7, while figure 4.8 shows the time evolution of the bubbles’ position once the obtained optimal control, which has value

\[ C^* = \begin{pmatrix} 5.858 \times 10^{-5} \\ -4.9998 \end{pmatrix}, \]

is used.

Both the bubbles’ position (figure 4.8) and the value of the control function \( C \) are well recovered by the optimal solution. The history of convergence (figure 4.7), shows a decrease in both the functional value and its gradient norm at each minimization iteration. Finally, the figure presenting the number of backtracking iterations shows that the adopted choice for the backtracking parameters is enough to satisfy Armijo’s
4.2.2 Test case 2: Neumann boundary

The initial setting for the second compressible test case is shown in figure 4.9. The domain has sides $L_1 = L_2 = 2$ and it is discretized using a structured triangular grid with $N_h = 80$ nodes on each side. Initially, the bubble has center in $(1, 0.5)$ and radius $r_0 = 0.25$. The initial density is $\rho_0 = 2.25$, corresponding to an initial pressure $p_0 = 202500$, and gravity is $g = (0, -9.81)$, while the boundary pressure is $p_b = 101325$. The final time is $t_f = 1$ and the time step is $\Delta t = 0.05$, which gives a number of time steps $N = 20$.

As already mentioned in the introduction to this section, this is the only test case
4.2. COMPRESSIBLE-INCOMPRESSIBLE RESULTS

Figure 4.9: Compressible-incompressible test case 2, initial position of the bubble shown here which enforces mixed boundary conditions on $\partial\Omega$.

Since the bubble has higher pressure and lower density than the surrounding liquid, it rises up vertically under the action of gravity, while increasing its volume in order to balance inner and outer pressure, as already seen in section 1.3.6. The goal of the optimal control problem in this section is to force the bubble to follow a bent path while rising. To this aim, the target position for the bubble has center in $(x_{C,t}, y_{C,t}) = (1.5, 1)$ and a radius that allows its pressure to be at equilibrium with the outer pressure at the final height. Such value is given by:

$$r_f = \sqrt{\frac{\rho_0 r_0^2 r_T}{p_b + \rho_l g_y (L_2 - y_{C,t})}} \approx 0.353252,$$

Figures 4.10 and 4.11 show the history of convergence and the results of the optimal control problem, respectively, while figure 4.12 shows the behaviour in time of the surface of the bubble once the optimal control (figure 4.11, bottom row) is used to solve the problem.

The target is well recovered by the optimal control (figure 4.11, middle row). Figure 4.10 shows that, by the end of the optimization loop, both the cost functional and the norm of its gradient are much smaller than their initial values. Finally, the bubble does grow in size, as expected, reaching almost double its initial volume (figure 4.12).
CHAPTER 4. NUMERICAL RESULTS OF THE CONTROL PROBLEMS

Figure 4.10: Compressible-incompressible test case 2, history of convergence
4.2. COMpressible-Incompressible Results

Figure 4.11: Compressible-incompressible test case 2, optimal control problem results. Top row: uncontrolled solution. Middle row: controlled solution (orange) and target position (dashed black). Bottom row: computed optimal control.

Figure 4.12: Compressible-incompressible test case 2, surface of the bubble under the action of the optimal control.
4.2.3 Test case 3: controlling the final height

Figure 4.13 shows the initial settings for the third compressible test case. Once again, we consider a square domain with sides $L_1 = L_2 = 2$, which is discretized using a structured triangular grid with $N_h = 80$ nodes on each side. The two bubbles, initially at rest, are placed in the liquid domain, with centers in $(0.5, 0.5)$ and $(1.5, 0.5)$, radiuses $r_1 = r_2 = 0.25$ and initial densities $\rho_1 = 1$ and $\rho_2 = 3$, corresponding to pressures $p_1 = 90000$ and $p_2 = 270000$, respectively. The final time is $t_f = 1$ and the time step is $\Delta t = 0.05$, which gives a number of time steps $N = 20$. Gravity is $g = (0, -9.81)$.

Having the bubbles different pressures but equal volume at the initial state, they would be pushed upwards at different speed by the buoyancy force. The aim of the solution of the optimal control problem is to prevent this phenomenon from happening, driving the bubbles to the same height instead. To do so, the target position of the two bubbles is chosen to be centered in $(0.5, 1)$ and $(1.5, 1)$, with radiuses given by

$$
r_{1,f} = \sqrt{\frac{\rho_1}{\rho_1 + \rho_2} \left( r_1^2 + r_2^2 \right)} \approx 0.176776, $$
$$
r_{2,f} = \sqrt{\frac{\rho_2}{\rho_1 + \rho_2} \left( r_1^2 + r_2^2 \right)} \approx 0.306186. $$

(4.2)

so that total volume is conserved and the pressures in the bubbles are at equilibrium.

Figures 4.14 and 4.15 show the history of convergence and the results of the optimal control problem, respectively.

As can be seen from the plots in the middle row of figure 4.15, the target is well recovered by the optimal solution. The target and computed bubbles’ positions are not exactly superimposed, but are clearly much closer if compared to the respective positions of the target and the uncontrolled bubbles at final time (figure 4.15, top row). Furthermore, and more importantly, the obtained optimal control does indeed drive the bubbles to a position where both centers are at the same height, as desired.

From figure 4.14, which shows the history of convergence, we appreciate the efficacy of the gradient-type algorithm in reducing the cost functional. Notice that, despite the good results in term of reconstruction of the target position, the norm of the gradient of the cost functional is not quite as small as in section 4.2.1, because of the failure (at iteration 26) of the backtracking algorithm in finding a proper value for the step length $\alpha$. This is probably due to the use of an inexact gradient method.
4.2. COMPRESSIBLE-INCOMPRESSIBLE RESULTS

Figure 4.13: Compressible-incompressible test case 3, initial position of the bubbles

Figure 4.14: Compressible-incompressible test case 3, history of convergence
Figure 4.15: Compressible-incompressible test case 3, optimal control problem results. Top row: uncontrolled solution. Middle row: controlled solution (orange) and target position (dashed black). Bottom row: computed optimal control
4.2.4 Test case 4: convergence analysis

The role of the mesh size and of the number of time steps is now analyzed. In particular, simulations on the test case of section 4.2.3 were carried out for three different levels of refinement:

(R1) $N_h = 40$ and $\Delta t = 0.1$;
(R2) $N_h = 80$ and $\Delta t = 0.05$;
(R3) $N_h = 160$ and $\Delta t = 0.025$.

Figure 4.16 investigates the behaviour of the solution for cases (R1) through (R3), showing that the position of the bubbles at the final time is quite consistent among the refinement levels examined. In particular, a convergence behaviour is observed, as the positions of the bubbles at the two finest resolutions are very close, virtually indistinguishable at the final time $t_f = 1$.

![Figure 4.16: Compressible-incompressible test case 4, evolution of the controlled solution at different time instants and with different refinement levels (R1) (solid line), (R2) (dashed line) and (R3) (dash-dotted line).](image)

4.2.5 Test case 5: controlling the topology

The initial configuration for the last test case is shown in figure 4.17. For this test case, we move to a rectangular domain with sides $L_1 = 2$ and $L_2 = 3$, which has been discretized using a structured triangular grid with $N_x = 80$ nodes on $L_1$ and $N_y = 120$ nodes along $L_2$, so that the grid size is the same on both sides. The two bubbles have initial centers in $(1, 0.5)$ and $(1, 1.25)$ and radiiuses $r_1 = r_2 = 0.25$, while the densities are set to $\rho_1 = 1$ and $\rho_2 = 3$, corresponding to pressures $p_1 = 90000$ and $p_2 = 270000$, respectively. The final time is $t_f = 1.5$ and the time step is $\Delta t = 0.05$, which gives a number of time steps $N = 30$. Gravity is $g = (0, -9.81)$.

The goal of this test case is to control the topology of the bubbles. In the uncontrolled case, the higher-pressure bubble placed initially at the bottom would rise
Figure 4.17: Compressible-incompressible test case 5, initial position of the bubbles quicker than the top one, ending in a merging of the two, thus giving rise to a larger single bubble. The goal of the optimal control problem is to keep the two bubbles separated up until the end of the simulation. To this end, the target position chosen for the two bubbles has centers in \((1, 1.75)\) and \((1, 2.5)\), so that they maintain the same relative distance as in the initial state, and radiuses given by equation (4.2).

The history of convergence and the results of the optimal control problem are shown in figures 4.18 and 4.19, respectively.

In particular, figure 4.18 shows that the cost functional decreases monotonically along the iterations. As in section 4.2.3, the final value for the norm of the gradient is not as small as in section 4.2.1, due probably to the use of an inexact gradient method. However, this does not prevent the solution from being quite satisfactory, as the goal of having two separate bubbles at the end of the simulation is accomplished.
4.2. COMPRESSIBLE-INCOMPRESSIBLE RESULTS

Figure 4.18: Compressible-incompressible test case 5, history of convergence
Figure 4.19: Compressible-incompressible test case 5, optimal control problem results. Top row: uncontrolled solution. Middle row: controlled solution (orange) and target position (dashed black). Bottom row: computed optimal control.
Chapter 5

A C++ library for optimal control problems

In order to handle the solution of optimal control problems, a new library was developed. Written in C++, it is based on DOLFIN/FEniCS version 1.6.0, and it is called dcp (DOLFIN Control Problem). For further details on DOLFIN/FEniCS the reader is referred to the online site [FEn16] and the user manual [LMW12], which contains introductory tutorials and several examples.

The dcp library was created with generality in mind and can be used to solve not only problems 3.2 and 3.3 but basically any optimal control problem based on partial differential equations, as well as plain systems of partial differential equations.

The source code is available via a public Git repository hosted on Bitbucket (https://bitbucket.org/mattia_tamellini/dcp) and it is released under GNU GPL license.

As discussed in section 2.2.1, a PDE-based optimal control problem is composed of three parts:

1. one or more partial differential problems;
2. an objective functional;
3. a minimization algorithm.

The dcp library follows this structure exactly, by defining specific classes for each of these three components, as well as other auxiliary classes, not all of which will be discussed here.

5.1 Main library classes

All of the code in the dcp library was developed using an object-oriented paradigm. In the next sections the main features of the most important C++ hierarchies of classes
in the *dcp* library will be discussed. In particular, the hierarchies used to describe
differential problems, objective functionals and minimization algorithms will be pre-
 presented. However, notice that the key implementation aspects are only discussed in
broad strokes. As a matter of fact, in order to avoid cluttering the UML diagrams for
the classes shown below, only the main members and methods of each class are shown,
and all the pointers and references used are omitted, even though the code makes abund-
 dant use of polymorphism whenever possible. This section only aims at providing a
 brief introduction to the library. For an extensive and accurate representation of the
classes and their relations, please see the Doxygen-generated documentation (see the
file `README` within the repository for details on how to build it). A test directory
containing some examples of use of the different classes, which can be employed as
guidelines, is also available in the repository.

### 5.1.1 The problems hierarchy

Figure 5.1 shows the UML diagram for the problems hierarchy of classes in the
*dcp* library.

The base class is called *GenericProblem* and it represents the basic interface for
a problem, in the sense of an entity whose solution is a Finite Element function which
is not known a priori (unlike, for example, an algebraic expression, for which the class
dolfin::Expression exists). It is an abstract class, which defines the basic public
interface and protected members shared by the classes in its hierarchy.

In particular, the protected member *solution* represents the solution of the prob-
lem itself and it is declared as a *TimeDependentFunction*, which is an auxiliary
class defined in the *dcp* library. Such a class is little more than a wrapper around
a `vector<pair<double, dolfin::Function>>`, with the second element in the pair
holding the Finite Element function representing the solution to the problem and the
first element holding the timestamp associated to that function. In the case of a
time-independent problem, *solution* is simply used as a single-element vector, with
a meaningless value used as a placeholder for the first element in the pair.

The `solve` method in *GenericProblem* is a pure virtual function that must be
overridden in derived classes, since there is no common behaviour that can be shared
throughout the hierarchy.

The `setCoefficient` method is purely virtual as well, and, in the derived classes,
it simply wraps a call to the `set_coefficient` function in `dolfin::Form`. Many other
methods, not shown in the UML diagram, share the same behaviour. For example,
the base class *GenericProblem* provides pure virtual methods to set the integration
subdomains (used for Neumann boundary conditions) or the location and value of
Dirichlet boundary conditions, which must all be overridden in the concrete classes.

The two branches that descend from the classes *GenericNonlinearProblem* and
*GenericLinearProblem* share the same idea. The leaf classes, *NonlinearProblem*
and `LinearProblem`, represent nonlinear and linear differential problems, respectively, and the parent classes are defined simply to enable polymorphism. Indeed, the children classes are templatized over the `dolfin::Form`-s representing the residual and jacobian forms (for the nonlinear problem) and the bilinear and linear forms (for the linear problem), which should be defined in a `ufl` file (see [LMW12]). The classes `GenericNonlinearProblem` and `GenericLinearProblem` make writing methods that take any nonlinear or linear problem as input arguments possible. While the parent classes contain the complete interface for the derived ones, it is only in `NonlinearProblem` and `LinearProblem` that the `dolfin::Form`-s are stored and that the pure virtual `solve` method is overridden.

The `AlgebraicProblem` class, on the other hand, defines problems which are not differential in nature, but whose solution can be computed as an algebraic expression. To do so, it stores a `GenericExpression` (see section 5.1.2) that holds the way the solution is computed. The key difference between an `AlgebraicProblem` and a regular expression is twofold. First, the solution to an `AlgebraicProblem` may depend on the solution of other problems. Second, it may be used within an object in the `GenericEquationSystem` hierarchy, which is discussed below.

Finally, `TimeDependentProblem` is a decorator for the `GenericProblem` hierarchy, in that it derives from `GenericProblem` and it stores a pointer to a `GenericProblem` at the same time. Such an object is used by the class to step through time by calling the `solve()` method on it at each time step during the simulation. The protected member `solution`, inherited from `GenericProblem`, is used here as a real vector, with each element storing the solution of the problem at a different time step. The `TimeDependentProblem` class also contains the control elements for the time loop, i.e. the initial and final time, the time step and a `Time` object. This is another one of the auxiliary classes defined in the `dep` library and it is not discussed in depth here. Suffice it to say that it helps keep track of the physical time throughout the whole main program during the simulation. See the library documentation for more details.

The second hierarchy in this module spawns from `GenericEquationSystem` and it represent systems of equations. In particular, it stores objects of type `GenericProblem` (each identified by a name, given by the user) and links between those objects (represented by a `map<LinkKey, LinkValue>`, where `LinkKey` and `LinkValue` are typedef-s defined in the `GenericEquationSystem` class itself). The `solve` method takes care of solving the problems in the given order, using the solutions of those already solved as coefficients in the problems yet to solve (upon request by the user), in fact using a staggered approach for the solution of the system at each time step. Like in the other classes in this module, the `solve` method is a pure virtual function in the base class and it needs to be overridden in `EquationSystem` and `TimeDependentEquationSystem`.
Problem Hierarchy

GenericProblem

# functionSpace : dolfin::FunctionSpace
# solution : dcp::TimeDependentFunction

+ solve (solveType : string) = 0
+ setCoefficient (coefficientType : string, coefficientValue : dolfin::GenericFunction) = 0

GenericNonlinearProblem

# residualForm : ResidualForm
# jacobianForm : JacobianForm

NonlinearProblem

# residualForm : ResidualForm
# jacobianForm : JacobianForm
+ solve (solveType : string)

GenericLinearProblem

# bilinearForm : BilinearForm
# linearForm : LinearForm

LinearProblem

# bilinearForm : BilinearForm
# linearForm : LinearForm
+ solve (solveType : string)

AlgebraicProblem

# expression : GenericExpression

+ solve (solveType : string)

ResidualForm, JacobianForm

TimeDependentProblem

# t : double
# dt : double
# startTime : double
# endTime : double
# timeSteppingProblem : GenericProblem

+ solve (solveType : string)
+ isFinished ()

GenericEquationSystem

# storedProblems : map<string, GenericProblem>
# solveOrder : vector<string>
# problemsLinks : map<LinkKey, LinkValue>

+ solve (solveType : string) = 0
+ addProblem (name : string, problem : GenericProblem)
+ operator[] (name : string)
+ linkProblems (link : Link)

EquationSystem

+ solve ()

TimeDependentEquationSystem

+ solve ()
+ isFinished ()

Figure 5.1: UML diagram for the problems hierarchy of classes in the dcp library
5.1. MAIN LIBRARY CLASSES

5.1.2 The objective functional hierarchy

The second main aspect of an optimal control problem is the definition of an objective functional. Figure 5.2 shows the UML diagram for the classes involved. Two hierarchy of classes are shown, one having GenericObjectiveFunctional as root and the other GenericExpression. They are both shown here since the former uses the latter extensively.

The GenericExpression class actually inherits from the Expression class in the DOLFIN library. This accomplishes three goals. First, the basic interface of the two classes is the same, so that the users who already know the DOLFIN library will be familiar with it. Second, an object declared using one of the types in the GenericExpression hierarchy can be used in lieu of a dolfin::Expression anywhere in the code. Third, it allows to expand the functionalities in dolfin::Expression beyond simply overriding the eval() method. Indeed, the use of expressions in the DOLFIN library is pretty basic, and it only allows the definition of simple expression depending solely on the domain coordinates \((x, y, z) \in \Omega\). The GenericExpression hierarchy expands upon that, by allowing the expression to depend on other expressions or Finite Element functions. This is accomplished by means of the protected member variables, which stores a map where a name (stored as a string) identifies each dolfin::GenericFunction the expression being defined depends on.

The AlgebraicProblem class discussed in section 5.1.1 does exactly this: it stores a GenericExpression defining the way the solution to the problem should be computed as a protected member. If the problem is part of a GenericEquationSystem, this latter object takes care of setting the solution of other problems in the system as coefficients in the AlgebraicProblem, by calling the setCoefficient method of the GenericExpression stored within the problem itself.

The derived classes specialize the behaviour of GenericExpression (which is purely virtual, so it can not be instantiated). Each of these has a different behaviour. Indeed, unlike in the DOLFIN library, the GenericExpression hierarchy can be employed in two ways. First, as in DOLFIN, the eval() method can be overridden in a derived class. Second, a functor can be passed to the constructor of the class, as the default behaviour of the derived classes in the hierarchy is to have the eval() function call the functor stored in the protected member evaluator. The type of this object changes from class to class, and it is defined through a typedef definition in the classes themselves. This is why many children of GenericExpression are provided, since each uses a different type for evaluator, based on what type of expression it defines. In particular, Expression defines a regular expression (so its type overlaps with the type provided by dolfin::Expression), VariableExpression defines an expression depending on other expressions and functions, TimeDependentExpression defines an expression depending on the spatial coordinates and on the time and TimeDependentVariableExpression defines an expression depending on spatial co-
Figure 5.2: UML diagram for the objective functional hierarchy of classes in the *dcp* library.
ordinates, time and other expressions and functions. The last two make use of the \texttt{Time} class discussed in section 5.1.1.

Finally, the \texttt{GenericObjectiveFunctional} hierarchy defines classes to represent an objective functional. Like \texttt{GenericLinearProblem} and \texttt{GenericNonlinearProblem} in section 5.1.1, the only use of \texttt{GenericObjectiveFunctional} is to allow polymorphism to be employed, since the concrete objective functional class is templated on the objective functional form, defined in a \texttt{ufi} file. The objective functional then has methods to return its value and the value of its gradient in a given point. The gradient of the objective functional is represented by a \texttt{GenericExpression} that must be supplied to the constructor of the class.

### 5.1.3 The optimizer hierarchy

Lastly, figure 5.3 shows the UML diagram for the optimizer hierarchy of classes. The class \texttt{GenericImplementer} and its children serve as auxiliary classes to the hierarchy spawning from \texttt{GenericDescentMethod}. As a matter of fact, the latter is completely general and only defines the operations needed to iteratively get to the minimum of an objective functional depending on a system of partial differential equations, assuming nothing on how such system is solved or how the descent direction is computed. This is the role of the \texttt{GenericImplementer} hierarchy. The parent class simply defines the interface for all \texttt{Implementer}s, which are the classes that take care of:

- updating the equation system when the control variable changes, by using the \texttt{update()} method and the protected member \texttt{updater}, a functor whose type is defined within the class itself. Some predefined classes that can be used as an updater are provided with the library and implement basic behaviours such as updating a system for which the control is a Dirichlet or Neumann boundary condition, or a distributed right-hand-side coefficient;

- computing the search direction for the minimizing algorithm, by using a second functor, \texttt{searchDirectionComputer}. The most basic search direction computer, the one that uses the negative gradient, is provided with the library and is used as the default value when the class is created;

- solving the systems of partial differential equations involved.

\texttt{BacktrackingImplementer} and \texttt{TimeDependentBacktrackingImplementer} implement the standard default behaviour for the backtracking algorithm (see section 2.2.3.1 and algorithms 3 and 5). The whole hierarchy is templated over the type of the object representing the optimal control itself.

The \texttt{GenericDescentMethod} hierarchy, on the other hand, implements the minimization algorithm. The base class is purely virtual, and once again is only defined
Figure 5.3: UML diagram for the optimizer hierarchy of classes in the *dcp* library
to be able to use polymorphism. Its child, \texttt{BacktrackingOptimizer} exposes, through the \texttt{apply()} method, the concrete use of the backtracking algorithm (again, see section 2.2.3.1 and algorithms 3 and 5), and it makes use of the \texttt{GenericImplementer} hierarchy discussed above. The final optimal control which minimizes the objective functional is made available outside of the class by storing it in the input parameter \texttt{initialGuess}, after the function returns.
Conclusions

In this work two-phase compressible-incompressible flows were considered, motivated by the need to investigate the production process of metal foams.

The first key aspect that was studied is the modelization of such kind of flows, which includes both the choice of a proper way to keep track of the position of each phase and the selection of the best model to describe the flow in each phase. Such model should be able to provide an accurate description of the flow variables while at the same time avoid introducing too much complexity, so that it can be employed in an optimal control problem framework.

The literature on the subject is vast and many different choices are possible. After an initial screening, in chapter 1 the two models that seemed the most promising, as far as our goal was concerned, were studied, presented and compared to one another, in order to show the similarities and the differences among them. In order to do so, some selected test cases, chosen to be paradigmatic for the application that is the final goal of this work, have been investigated using both models. The numerical results show that, for a small enough time step, the two models behave similarly. The single-pressure model introduced in [BGN11], which has a simpler structure than the split-pressure model in [Dar+10], was then chosen, so that the adjoint problem could be derived, leading to the definition of an optimal control problem.

The second aspect which was investigated is the controllability of compressible-incompressible flows when the optimal control problem built upon the single-pressure model is employed. Starting from the general definitions, in chapter 3 the specific control problem, core of this thesis, was detailed. The optimality conditions spawning from it and the complete optimization algorithm used were made explicit. The former give rise to three coupled systems of equations, which have been numerically discretized to obtain the latter.

Several numerical experiments were conducted, albeit in a controlled environment where some of the issues concerning real-world compressible-incompressible flows were smoothed out. The results show the efficacy of the proposed methodology and pave the way for more realistic simulations related to the production of metal foams. These test cases show that the physical system is indeed controllable, both when a specific final position of the gas bubbles and when a specific topology (or a specific number
of bubbles) is required. This means that by choosing a proper forcing term in the momentum equation the system can be driven to a configuration that would not be the natural solution of the equations system, if left uncontrolled.

A possible topic for future work may be to extend this approach to use a type of control that is closer to what one would employ in an industrial framework. Indeed, a distributed volumetric force can hardly be applied consistently and precisely during the foaming process. A more realistic choice for the control variable should be the boundary temperature on the wall of the domain. This would entail the extensions to the non-isothermal case of the optimality conditions in the optimal control problem. Also, simulations with more realistic physical parameters should be carried out. On the one hand, this would require the use of more sophisticated techniques to handle the various numerical pitfalls that may arise in this case. Moreover, if the formulation of the primal problem is modified to face these additional difficulties (for example, by using a harmonic average for the computation of the global viscosity, as discussed in section 1.2.4, or by using a turbulence model), the adjoint system and the optimality conditions may have to be derived from scratch. On the other hand, finer space-time grids would have to be used, and therefore more efficient solution methods (e.g. preconditioned iterative methods and operator splitting methods) would have to be employed for the linear systems obtained from the space-time discretization in sections 1.2.3, 1.3.2.2 and 1.3.3.2, in order to handle the greater computational effort.

A second topic that might be worth investigating is the use of quicker and better minimization algorithms, such as, for example, Newton or Quasi-Newton methods. As a matter of fact, the gradient-like algorithm employed in this thesis, although relatively easy to define and implement, leaves margins of improvements on the subject.

Finally, a last area of interest is the optimization and parallelization of the source code used throughout this thesis. Indeed, since the solution of the optimal control problem requires solving the primal and adjoint systems multiple times, the total computational time would hugely benefit even from a slight reduction of the time needed to solve the systems from initial to final time once.
Bibliography


