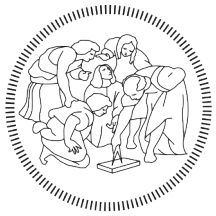
Politecnico di Milano
Dipartimento di Matematica
Ph.D. in Mathematical Models and Methods in Engineering



POLITECNICO
MILANO 1863

Non-conforming methods for the simulation of industrial polymer mixing processes

Ph.D. candidate:
Giorgio Negrini

Supervisor:
Prof. Nicola Parolini

Co-supervisor:
Prof. Marco Verani

The Chair of the Doctoral Program:
Prof. Michele Correggi

2023 – XXXV cycle

Abstract

This work focuses on the numerical modelling of polymer mixing processes. These processes are performed with the aid of devices (single- and twin- screw extruders, planetary extruders, Banbury mixers, etc.) characterized by complex geometries in which body-fitted simulations can hardly be performed. Non-conforming approaches, like diffuse interface, fictitious domain, immersed boundary or volume penalty methods, represent the best alternative to simulate this type of processes, that may also involve complex kinematics. In this work, we present an implementation of an Immersed Boundary method (IBM) to deal with this type of simulations and with realistic industrial screw geometries.

The flow of a polymer inside a mixing device is described by the incompressible Navier-Stokes equations. Moreover, the polymer is modelled as a non-Newtonian fluid with temperature dependent viscosity. Particular attention is paid to the description of rheological models of polymer viscosity, that depends on shear rate, temperature and filler fraction.

To solve numerically this problem we consider the Finite Volume method (FVM), widely adopted in industry because of its built-in conservation properties, its ability to deal with arbitrary mesh and its computational efficiency. In this context, a variational analysis of the FVM as a Box method (BM) applied to the Stokes problem is carried out. The BM is a piecewise linear Petrov-Galerkin formulation on the Voronoi dual mesh of a Delaunay triangulation. To recover the *inf-sup* stability of the Stokes problem discretized by the Box method, we resort to the Rhie-Chow stabilization, for which a convergence analysis is presented. Then, we consider the linear Stokes problem discretized using the Box method combined with a Diffuse Interface method (DIM). The application of DIM to approximate non-conforming boundaries leads to error convergence rates that are suboptimal with respect to the ones of the conforming solution.

To overcome the limited convergence properties of the Diffuse Interface Box method, we introduce an improved non-conforming approach based on the Immersed Boundary method. It consists in a discrete-forcing direct-imposition method where a mesh subset is selected and then the solution on this subset is computed with respect to the original boundary condition. The solution imposed is then corrected using a quadratic weighted least squares interpolation, that allows to significantly improve accuracy and recover optimal convergence rates. Moreover, due to the fact that the pressure-velocity coupling is solved with a projection algorithm (SIMPLE), a Neumann boundary condition for pressure that is consistent with the velocity profile has been developed.

Finally, several applications to real industrial devices complete this work, in particular we considered single- and twin-screw extruders and planetary roller extruders. In general,

extruders are made by an external shell, called barrel, and internal parts, called screws or rotors. While the barrel has often simple shape, screws present different geometric features and sharp edges. Moreover, the gaps between the barrel and the internal screws are narrow, four orders of magnitude smaller than the machine dimensions, which make the problem multi-scale. Here, the strategies that have been developed to deal with these complex objects are exposed, in particular for anisotropic grids or graded grids, multiple interacting and adjacent immersed geometries, moving objects, small gaps and the high non-linearity of the problem.

Sommario

Il seguente lavoro si concentra sulla modellazione numerica di processi di mescolamento di polimero fuso. Questi processi vengono effettuati grazie a macchinari come estrusori monovite e bivite, estrusori planetari, Banbury mixers, in genere caratterizzati da geometrie complesse. Per questo motivo, la loro simulazione non è praticabile tramite approcci body-fitted, dato l'alto costo computazionale. Gli approcci detti non-conformi, come diffuse interface, fictitious domain, immersed boundary oppure volume penalty methods, rappresentano la miglior alternativa per simulare questo tipo di processi, che spesso presentano anche cinematiche complesse. Quindi, al fine di simulare geometrie realistiche di estrusori industriali, presentiamo l'implementazione di un Immersed Boundary method (IBM).

Il flusso di un polimero all'interno di un mixer è descritto dalle equazioni di Navier-Stokes per fluidi incomprimibili. Inoltre, i polimeri si possono considerare fluidi non-Newtoniani con viscosità dipendente dalla temperatura. Vengono quindi descritti i modelli reologici per la viscosità dei polimeri, la quale dipende, oltre che dalla temperatura, dallo sforzo di taglio e dalla presenza di additivi, chiamati filler, misurati in frazione di volume.

Per la risoluzione numerica di questi problemi, abbiamo considerato il metodo ai volumi finiti (FVM), ampiamente utilizzato in campo industriale per la sua proprietà intrinseca di conservazione, per la sua capacità nel gestire mesh arbitrarie e per la sua efficienza computazionale. In questo contesto, è stata svolta un'analisi variazionale del FVM come Box method (BM) applicato al problema di Stokes lineare. Il BM è una formulazione Petrov-Galerkin lineare a tratti su una griglia Voronoi, duale di una triangolazione di Delaunay. Per recuperare l'*inf-sup* stabilità del problema di Stokes discretizzato con il BM ricorriamo alla stabilizzazione di Rhie-Chow, di cui presentiamo un'analisi di convergenza. Consideriamo poi il problema di Stokes lineare discretizzato con il Box method in combinazione con un metodo Diffuse Interface (DIM). L'utilizzo del DIM per approssimare contorni in maniera non-conforme porta a ordini di convergenza subottimali rispetto a quelli della soluzione conforme.

Per superare i limiti di accuratezza del Diffuse Interface Box method, introduciamo un approccio non-conforme migliorato, basato sull'Immersed Boundary method. Esso consiste in un metodo discreto a imposizione diretta in cui viene selezionato un sottoinsieme della mesh e, su questo sottoinsieme, la soluzione viene calcolata tenendo conto della condizione al contorno originale. La soluzione imposta viene poi corretta utilizzando un'interpolazione quadratica ai minimi quadrati pesati, che permette di migliorare significativamente láccuratezza e di ripristinare gli ordini di convergenza ottimali. Inoltre, dal momento che l'accoppiamento velocità-pressione viene risolto da un algoritmo di proiezione (SIMPLE),

per la pressione è stata sviluppata una condizione di Neumann consistente con il profilo di velocità.

A completamento di questo lavoro, presentiamo alcune applicazioni a macchinari industriali reali, in particolare abbiamo considerato estrusori monovite e bivite e un estrusore planetario. Gli estrusori sono di norma composti da un guscio esterno, chiamato cilindro, e internamente da viti o rotori. Il cilindro ha spesso una forma semplice, mentre le viti presentano spigoli e possono avere caratteristiche geometriche differenti. I traferri compresi tra le viti ed il cilindro sono molto stretti, quattro ordini di grandezza in meno rispetto alle dimensioni del macchinario: questo rende il problema multi-scala. A questo punto, descriviamo le strategie adottate per risolvere la forte non-linearità dei problemi di mescolamento dei polimeri fusi e per gestire la presenza delle viti e le complessità computazionali che introducono. Particolare attenzione è posta all'utilizzo di griglie con elementi anisotropi, alla gestione di più geometrie immerse o di geometrie in movimento e alla presenza di traferri stretti.

Ringraziamenti

Alla fine di lavori come una tesi di Dottorato, sono sempre numerose le persone da ringraziare. Queste parole vogliono sintetizzare una lunga e intensa esperienza in poche righe, senza fare un bilancio esaustivo di questi tre anni, ma piuttosto cercando di fissare nella mente e nel cuore quei momenti e quei rapporti che hanno reso ricco e vero questo percorso.

Parto da chi ha reso possibile questo lavoro, cioè i miei relatori Proff. Nicola Parolini e Marco Verani. Grazie perchè mi avete sempre guidato lasciandomi comunque libertà durante il lavoro. Grazie per l'immenso sostegno di questi anni e per il prezioso e costante confronto.

Ringrazio Fondazione Politecnico per aver finanziato la borsa di Dottorato e il Joint Lab PoliMi-Pirelli, in cui è inserito il progetto. Ringrazio inoltre Paolo Quinzani, Fabrizio Ricci, Daniele Cerroni e tutto il gruppo Pirelli. Avete sempre valorizzato questo progetto e il lavoro svolto. Un grazie particolare va a Umberto Visconti per la compagnia nel lavoro durante i primi mesi di Dottorato e, soprattutto, di pandemia.

Ora usciamo un attimo dall'università. Un grande grazie a mia moglie Benedetta. Mi hai sempre incoraggiato e voluto bene senza mai risparmiarti. Il nostro rapporto è stato fondamentale per affrontare tutte le gioie, le difficoltà e i traguardi raggiunti di questi anni.

Grazie a Caterina e Stefano, i miei genitori, vi siete sempre spesi per me durante questo percorso, dandomi fiducia e volendomi bene, accompagnandomi in ogni passo fatto in questi anni. Grazie anche a Lorenzo e Antonella, mi avete calorosamente accolto nella loro famiglia offrendomi sempre disponibilità e olive ascolane. Un grazie speciale va a Don Cesare e Don Alberto, siete stati una compagnia eccezionale nella vita e nel cammino al matrimonio.

Ringrazio gli amici della Fraternità, con cui condivido il percorso di fede, l'esperienza da "giovani lavoratori" e a breve, con alcuni, anche l'esperienza del matrimonio. Siete un grande aiuto e un punto di riferimento. Grazie al gruppo di Challand, una presenza costante in ogni situazione, dal liceo al Dottorato, di cui sono molto grato. Grazie Mate, Balordi e Automatici.

Grazie ai colleghi del Dottorato, Vero, Ste, Laro, Buch, Bea e Ale, per condividere con me un'amicizia oltre al lavoro. Grazie ai colleghi del Tender. Mi avete supportato e sopportato durante il lavoro di questi anni. Sono molto grato dei rapporti nati in questo luogo e penso siano la cosa più preziosa che mi porto dietro da questa esperienza. Grazie Enrico, Giulia e Giuseppe, avete iniziato con me questa avventura e con cui ho condiviso le gioie e i dolori del Dottorato. Grazie Alberto, per le chiacchierate nei caldi pomeriggi di Bovisa, per le discussioni su OpenFOAM e perchè sei stato un grande sostegno in questi anni.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and purpose of the research

Polymers are essential and ubiquitous in everyday life applications. They can be natural, for example silk, rubber, wool, keratin, or synthetic, for example PVC, polystyrene, silicone, nylon. In particular, synthetic polymers are used to make plastics, adhesives, and many common objects. Synthetic polymers can be divided into three main groups: thermoplastics, thermosets, and elastomers. Thermoplastic materials soften when they are heated and solidify when they are cooled; their chemical structure does not change significantly during the processing, thus they can generally be reground and recycled. Thermosets undergo a crosslinking reaction when the temperature is raised above a certain temperature, thus creating a three-dimensional network of chemical bonds. The main characteristic of elastomers and rubbers is that they can undergo very large deformations behaving in a largely elastic manner.

Polymers undergo a complex multistep processing, called compounding, before turning into common objects. Often, the compounding of a polymer consists in the following steps. At the beginning, a certain quantity of ingredients, under the form of chips or pellets, is melted. The melting can be performed in different ways, for instance by heating or by friction or using a solvent, and by different devices. Then, the solution passes under the mixing process in order to obtain a uniform polymeric molten phase, possibly adding other ingredients and additives to enhance the final product properties. Finally, the polymer melt is extruded and pushed into a die, an outlet designed to give a certain form to the extrudate, e.g. annular for wire coating or a thin slit for filming. Once the melt exits from a die, the process is concluded and it can be laid or transported elsewhere or it can be injected into a molder to be given a particular shape [70].

The main process used in polymer industry is the extrusion process. This process is performed by devices called extruders, that are the most important piece of machinery in the polymer processing industry. To extrude means to push or to force out, so when the material is extruded means that it is pushed through an opening. The part of the machine through which the material is forced is referred to as the die. Dies are used to give a shape to the extrudate, the extruded material. This is the case of wire coating or food applications, where the material has to assume a precise shape. There are two basic types of extruders: contin-

uous or batch type extruders. Continuous extruders can develop a steady and constant flow of material, whereas batch mixers have to be cyclically emptied and reloaded. Continuous extruders utilize rotating members for transport of the material. Batch extruders generally have a reciprocating member to cause transport of the material [118].

Beside the transport of material, extruders are useful devices also for mixing. Mixing may be used for blending of ingredients and incorporation of other additives, often particulates [40]. The embedding of solid particulates into polymer matrices is widely diffused in industrial contexts in order to reduce costs and to improve some desirable thermal, mechanical, electrical and magnetic properties. In this respect, besides transport and shaping, extruders have the task to perform the embedding of the particulate and the elimination of inhomogeneities. This can be achieved by maximizing the dispersive and distributive mixing. Dispersive mixing is defined as the ability to break a certain particulate, present in a mixture, into smaller and smaller pieces. Distributive mixing is defined as the ability to disperse a certain particulate, present in a mixture, uniformly throughout the mixture.

To analyse the ability of an extruder of mixing a polymer melt is fundamental to know what happens inside the device and, in particular, how the velocity, pressure and temperature fields develop. For this purpose, numerical simulation have become nowadays a fundamental tool to make prediction and estimates on how the polymer behaves during the extrusion process. However, this is not an easy task to achieve, often due to the several complexities present in this type of applications. All these procedures are complex and imply knowledge of chemistry, mechanics, fluid dynamics and thermodynamics. Hence, to optimize the processes maximizing the production rate of polymeric material, one has to know how the polymer behaves during all the processes it is subjected.

The focus of this research is on the part of polymer processing regarding the mixing of polymer melt, where the fluid is mixed in order to produce a uniform compound. Thus, among all the physical phenomena that occurs in the polymer processing workflow, we concentrate our analysis on the fluid dynamics of polymer melt inside mixing devices and, in particular continuous extruders, not taking into account multiphase flows or solid-liquid phase transitions. The analysis is carried out using numerical tools to simulate the flow of polymers, especially thermoplastics and elastomers at the molten state, that can be modelled as non-Newtonian fluids with temperature dependent viscosity.

Many mathematical models have been developed in order to understand the fluid dynamics and heat transfer of the flow inside mixers and molders, starting from known fluid flow models and trying to approximate the evolution of pressure, viscosity and temperature inside a polymer processing device. For instance, to model the flow inside capillary or slit dies, a Poiseuille flow can be employed or a cavity flow can be used to describe what happens inside a single-screw extruder. Moreover, processes like calendering and coating have been described by simple mathematical models [102]. Some modelling can also be applied to simple shapes of twin-screw extruders [118]. However, as technology develops, mixing devices become more and more complex and so our capacity of modelling what happens inside them has to increase.

First the difficulty in solving the problem itself: a nonlinear fluid dynamics problem where the viscosity depends on temperature. In a second place, the fact that the problem is a multiscale problem because the gaps between the various moving components of the extruders

can be extremely small and the physical scales that the computational grid should be able to capture vary from metres to tens of millimetres. The last, and maybe the most significant complexity present in polymer mixing, is the geometrical complexity.

The aim of the research is to develop a numerical tool able to simulate the fluid dynamics involved in this type of industrial processes, dealing with all the physical, geometrical and numerical complexities involved in the problem. We first describe the problem from the physical point of view, understanding which variables are responsible of what happens into extruders. Then we choose the Finite Volume method, as a numerical method, in order to deal with physical complexities and because it allows to efficiently solve large scale problems, of which industrial polymer processes are a part. Then, we consider, alongside to the Finite Volume method, an Immersed Boundary method to approximate the complex geometries that characterise extruders and mixers in general. The combination of the two methods has been analysed from the theoretical point of view and then it has been applied to several test cases to assess its numerical efficiency.

This work has been carried out in collaboration with the industrial partner Pirelli Tyre S.p.a., in the framework of the Pirelli-PoliMi Joint Lab and with the financial support of Fondazione Politecnico.

1.2 Research structure and state of the art

In continuum mechanics, polymers can be considered incompressible fluids with viscosity that may depend on shear rate and temperature. For this reason, in most of the cases, polymers are modelled as incompressible non-Newtonian fluids, with temperature dependent viscosity. Depending on the polymer type, thermoplastics or elastomers, viscosity can vary from thousands of Pa·s up to millions of Pa·s. Larger viscosity values cause the viscous friction of the fluid to dissipate a relevant amount of energy, that heats the fluid.

The equations that govern the motion of a polymer melt are the incompressible Navier-Stokes equations with a suitable nonlinear rheology. These equations are nonlinearly coupled by the presence of viscosity, that depends on the shear rate, i.e. velocity gradients, and temperature. Moreover, the viscous friction, or viscous heating, is a source term of energy equation that models how velocity gradients heat the fluid. The presence of this nonlinearity has implications on the difficulty in solving the problem numerically.

The scientific community has devoted a lot of effort in the theoretical and numerical investigation of non-Newtonian flows with temperature dependent viscosity. For power-law fluids with temperature dependent viscosity, a well-posedness result have been proved in [51]. Many numerical methods have been analysed and applied in this field. Spectral and finite elements methods [4, 5] have been developed for the solution of the Navier-Stokes equations coupled with the heat equation. A p-least-squares finite element method has been applied to non-isothermal flows of non-Newtonian fluids, investigating also the process of heat production by the viscous friction [16]. Then, in a more industrial context, the Finite Volume method has been applied to model the non-isothermal flow of a non-Newtonian fluid through an extruder die [50]. Moreover, for the decoupling of the Navier-Stokes equations [72] with variable viscosity, a shear rate based projection method has been proposed in

[46, 47] in order to compute a pressure field consistent with the flow of a non-Newtonian fluid. Finally, also a posteriori estimates have been proposed to perform grid adaptivity in problems involving non-Newtonian fluids [19].

In this work, to approximate this system of equations, we consider the Finite Volume method (FVM). The FVM is a popular numerical strategy for the spatial discretization of partial differential equations widely used for the solution of industrial flow problems. One crucial property of FVM is that, by construction, physical conservation laws governing in a given application are naturally discretized preserving global and local conservation properties [59]. This makes the method very attractive when dealing with problems where conservation plays an important role, such as fluid mechanics and heat and mass transfer.

This property is a consequence of the formulation of FVM. In fact, the core procedure of FVM is the imposition of the conservation law on each cell, or control volume, of the mesh [94, 106]. This is usually performed using Gauss theorem and then numerically reconstructing fluxes through each face of the control volume. The conservation properties of the Finite Volume method give raise to robust numerical schemes that work on arbitrarily complex geometries [59]. Moreover, the simplicity of FVM makes it easy to be implemented, giving raise to efficient codes also highly scalable on parallel infrastructures, due to the cheap matrix assembly and numerical integration costs.

The choice of employing the Finite Volume method to simulate mixing processes has been combined to the choice of using the C++ library OpenFOAM [71, 85], that is the one of the leading open-source +software for computational fluid dynamics and it is already widely employed in the industrial context. In the field of FVM, many different methods and numerical schemes have been developed in which vector variables are reconstructed using their face normal components [23]. From staggered schemes [109] to TPFA and MPFA [1, 2, 8], i.e. two- and multi- point flux approximation, respectively, and mixed finite volume schemes [53].

In the present work we consider a particular formulation of the FVM called Box method (BM), or Finite Volume Element method (FVEM) [33] or piecewise linear FVM. This method has been the object of an intense study in the literature. It was first introduced for scalar elliptic problems in [11, 73] and, more recently, in [57, 58, 61, 135]. Then the Box method was applied to Stokes system in [115]. Another version of the FVEM is reported in [35, 36], which extends the analysis to non-conforming piecewise linear finite elements. Moreover, as for the FVM, the FVEM shows to be suitable for several different applications, such as moving domain problems with Arbitrary Lagrangian-Eulerian formulations [68].

We consider the Box method to describe the classical FV method because of its simplicity and for its relationship with the Finite Element method (FEM) [56,69], indeed a relationship between BM and FEM solutions with respect to mesh discretization can be found [11]. This relationship derives from the fact that the Box method is the "dual method" of finite element, i.e. it consists in a piecewise linear Petrov-Galerkin formulation on the Voronoi dual mesh of a Delaunay triangulation.

We use the Box method to formulate the Stokes problem, discretized by OpenFOAM FVM, in a variational framework. This translates in discretizing Stokes problem using piecewise linear elements for both velocity and pressure also employing numerical discretization of fluxes, thus requiring a stabilization [7, 27, 55]. In this setting, we expose a convergence

analysis defining a stabilization equivalent to the Rhie-Chow interpolation [65,137], a common stabilization technique in Finite Volume applications.

As mentioned before, the Finite Volume method can be used to efficiently deal with domains with arbitrarily complex geometries. The capacity of describing complex geometries plays a fundamental role because of the core topic of this work: polymer mixing processes. In general, the screws and the rotors that are employed to perform transport and mixing of the polymer melt inside the extruders are characterised by complicated shapes. For this reason the generation of the computational grid is crucial.

There are two possible approaches to manage grid generation. The first is the body-fitted approach, where the mesh is constructed on a sufficiently accurate approximation of the exact physical domain. Improved approximation properties on complex domain discretized using body-fitted grids can be obtained resorting to isoparametric finite elements [38], isogeometric analysis [42] or spline-based parametrization [78]. Moreover, when dealing with moving domains, Arbitrary Lagrangian-Eulerian (ALE) formulations [42, 52, 79, 81] can be adopted. In particular, in the framework of ALE methods, it is worth mentioning the sliding mesh technique, first introduced in [104, 105]. It consists in building different grids for different parts of the domain and in mapping the solution across the so called sliding interfaces [14, 15, 76], simplifying the mesh construction avoiding the need of handling mesh deformations and remeshing.

The second is the non-conforming approach, where the physical domain is embedded into a simpler background mesh whose elements can intersect the boundary of the physical domain. The mesh generation process for the background mesh si extremely simplified with respect to the body-fitted approach, while a specific strategy for imposing of embedding the boundary conditions in a non-conforming way is required.

Non-conforming domain methods were first introduced by Peskin with its original version of the Immersed Boundary method [112], in which the immersed surface is represented with a set of Lagrangian points, while the flow field is computed on an Eulerian grid which is not required to conform the immersed body geometry. The immersed velocity boundary condition is then enforced by a source term in the momentum equation. The Immersed Boundary method has been the subject of intense studies fro many years [60, 82, 103]. In later works, this approach of enforcing boundary conditions into the equations governing the problem was interpreted as a penalty method in a variational context [10, 74]. A similar approach has also been developed in the context of finite element methods, with Fictitious domain and Embedding domain methods [22, 24] as well as with Diffuse Interface methods [29, 95]. These methods are often implemented with the aid of a characteristic function, or mask, that describes the region of the immersed surface. This mask can be approximated either by a discontinuous function or by a smoothed function using opportune signed distance functions. One advantage of these latter Diffuse Interface methods is that they are easy to be implemented, however, they are often subjected to suboptimal error convergence due to the fact that they “diffuse” the immersed interface over neighbouring mesh elements. The optimal convergence order can be recovered using, for example, mesh adaptivity [18, 124]. Another critical aspect of Diffuse Interface methods is mass conservation [123]. In particular in incompressible flows, mass and volume conservations coincide. This implies that the volume computed using the characteristic function of the immersed region has to be conserved

to avoid spurious pressure profiles near the interface.

To avoid the spreading of the embedded boundary across the computational domain, sharp interface methods can be employed. These methods manage to approximate non-conforming boundary as sharp interfaces. This can be done, for example, with cut element method, proposed in [31] together with its Hybrid High-Order version [30, 49]. This type of methods employ a Nitsche penalty method [108] to weakly enforce boundary conditions and by adopting consistent additional terms in the weak formulation. These methods typically guarantee good accuracy, however, a major drawback for cut methods comes from the "small cuts" that generate near the interface, when a small fraction of an element remains in the physical domain. When this happens, it reflects in a deterioration of the problem conditioning. To overcome this issue, many stabilization techniques have been developed, like the ghost penalty method introduced in [32]. Moreover, a significant computational effort has to be employed in order to find element cut points.

In the context of the Finite Volume method, many methods have been introduced in order to embed boundaries into a background mesh. A very popular method is the Ghost Cell method [129], which is an Immersed Boundary method that imposes the embedded boundary constraint on the mesh cells just inside the immersed surface, based on the surface values and the fluid cells values. Another widely adopted method is the Overset method [37, 125], in which two different meshes are generated for the fluid and the solid regions. Then the problem is solved on the fluid mesh using the geometrical information given by the position of the solid mesh. Finally, a sharp interface Cut-Cell method [91] has been implemented to approximate surfaces in a fluid domain that directly modifies the background mesh cells topology.

Another approach is the Shifted Boundary method [89, 99], in which the physical domain is substituted by a discrete approximation. In this approach a background mesh is built around the physical one and the mesh elements that cross the immersed boundary or that are outside it are discarded. Then, what remains is a discretized boundary on which boundary conditions have to be imposed accurately. For instance, "shifting" their value using a Taylor expansion. This approach does not imply cutting elements and shows the optimal convergence rates of conforming methods.

In industrial polymer processing, there exists a large variety of mixing devices, characterised by complex and diverse shapes, which mechanics can also involve complex kinematics. Therefore, we decided to adopt a non-conforming approach to describe these geometries in our simulations.

We first consider a simplified scenario in which the linear Stokes problem is discretized using the Box method combined with a Diffuse Interface method (DIM) [107]. The DIM consists in dividing the computational mesh into two subsets corresponding to the fluid and solid regions. The solid regions is the set of elements that are inside the immersed surface. The non-conforming imposition of the immersed boundary conditions is performed by constraining the solution values of the solid region to the boundary value. For this simplified problem, we performed a convergence analysis, employing the strategy described in [124]. As a result, we obtained a suboptimal convergence estimate for the DIM with respect to the conforming solution, due to the fact that we are imposing boundary conditions with a non-conforming strategy. We also showed that the original order of convergence can be

recovered by employing a deep local mesh refinement, with the measure of the mesh near the immersed object scaling as the square of the characteristic mesh length of the rest of the domain.

Thus, the former analysis suggests that, to be accurate, we have to adopt a local mesh refinement strategy. However, in real life applications, such refinement is most often unaffordable because the number of degrees of freedom raises rapidly. So, to overcome the convergence rate lost and to avoid local mesh refinement, we extend the concept of the Diffuse Interface method to the more general concept of the Immersed Boundary method (IBM). Our IBM implementation consists in a discrete-forcing direct-imposition method, where the mesh is divided into subsets with respect to the non-conforming surface and the constrained value of the solid region is corrected by the use of a least-squares interpolation strategy, that is eventually able to recover the conforming method accuracy. This approach is a middle way between classical immersed boundary methods and the Shifted Boundary method: we employ a direct forcing of the boundary conditions, but we correct the constrained value in such a way that the sharpness of the real boundary is preserved.

We describe the implementation that we made in the OpenFOAM library of the IBM, starting form the work described in [86], and we present a numerical assessment of the approximation properties of our implementation.

Another aspect in which non-conforming methods have an impact on the solution strategy is the usage of projection methods to solve fluid flow problems. A typical procedure in the context of Finite Volume method is the decoupling between velocity and pressure through splitting, or projection, methods. They commonly consists in iterating the following solving procedure: first solve the momentum equation, then a Poisson equation for the pressure and then project the velocity on the space of divergence-free functions. Projections methods are computationally efficient both in terms of CPU time and memory usage since the linear systems that generate are typically smaller and easier to solve than the monolithic one. Moreover, the iterative nature of these algorithms allows to robustly face the nonlinearities of the problem. In OpenFOAM, the common procedure is to apply the SIMPLE [110], PISO and PIMPLE [131] algorithms.

Combining these algorithms with IBM is not straightforward because we have to split, not only the equations, but also boundary conditions. If we have a rigidly moving body immersed in our computational domain we have to impose Dirichlet conditions on velocity and, consequently, Neumann conditions on pressure. Many strategies to impose non-conforming Neumann conditions have been proposed in the literature [6, 88, 121], but it has been proven that imposing homogeneous Neumann on pressure is not the best strategy when working with splitting methods. Indeed, consistent non-conforming conditions for pressure have been developed in [17, 83] for finite differences, in which the projection of velocity on divergence-free functional space is computed such that it respects its immersed boundary condition. In this work, we extend this concept to the Finite Volume method, experiencing improvements in simulation results, especially in the presence of anisotropic grids.

Ultimately, we present the results obtained employing the Immersed Boundary method to simulate the mixing of polymer melt inside representative geometries of industrial extruders. In particular, we consider a single-screw extruder, a twin-screw extruder and a planetary

roller extruder. We now present these three examples of mixing devices, which present increasing complexity.

In general, extruders are devices made by an external shell, called barrel, and one or more screws within it. Extruders can be divided into three regions: the feeding section, where the material enters the device, the metering section, where the mixing is actually performed, and the die, where the material is extruded and where it is given a shape (Figure 1.1). The geometrical complexity is obviously given by the presence of screws. The impact that

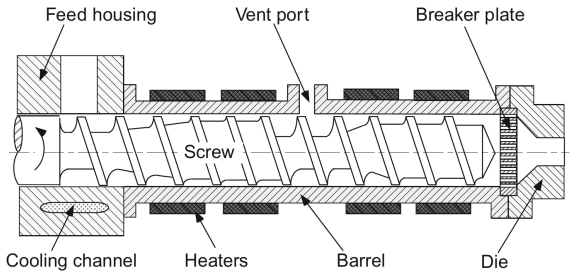


Figure 1.1: Schematics of a single-screw extruder [118].

screws presence has in numerical simulations resides in the grid generation. Despite screws are described by analytical shapes and can be often parametrized, the fact that they are in general characterized by sharp edges makes them hard to be meshed using body-fitted grids. Moreover, like in the case of the twin-screw extruder (Figure 1.2), the shafts are composed by more than one shape, implying the design of special strategies to generate the grid. It is

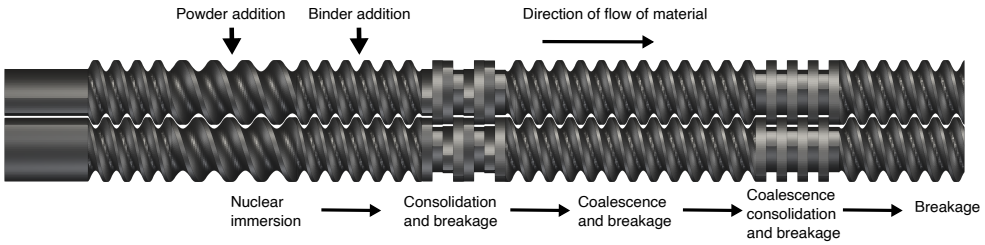


Figure 1.2: Schematics of a twin-screw extruder [48].

known that generating good computational meshes on complex shapes can be a very hard task, that can also result in unusable meshes. This task becomes even more difficult when dealing with moving objects, and so time dependent problems, that is the natural setting of mixing processes. This is mainly due to two reasons: the difficulty in describing the geometries in play and the computational effort in generating such grids.

There are many techniques to deal with single- and twin- screw extruders using body-fitted approaches. For instance the single-screw extruder can be considered a steady-state problem ,if simulated in a rotating non-inertial reference frame. Once the mesh for one SSE configuration is built, one can solve the Navier-Stokes equations adding Coriolis and centrifugal forces [39]. In this way, the observer sees the screw stationary and the barrel that rotates. Then, with a simple transformation, one can recover the absolute fields.

On the other hand, for twin-screw extruders some strategies have been developed, such as remeshing using spline-based interpolation [78] to build the grid that approximates the intermeshing region between the screw. An approach that is hard to be applied when considering different modules of the TSE (transport modules and kneading modules), requiring the addition of sliding interfaces between different modules [76]. However, these approaches are often customized for a precise application or a precise extruder and cannot be employed for arbitrary geometries and multiple types of devices. For instance, the approach based on a non-inertial frame of reference is applicable only when we have a single rotating object, like in single-screw extruder, while ad-hoc remeshing techniques that proved to be effective for twin-screw extruders (see [75,77]) cannot be extended to more recent mixing technologies that involve not only a significant increase of complexity in geometry and kinematics, namely the planetary roller extruder (PRE) (Figure 1.3).

The PRE is a multi-screw extruder composed by a central spindle and the barrel with variable number of smaller spindles, or planets, between them. The rotation of the central spindle drives the one of the planets thanks to their gear-like shapes.

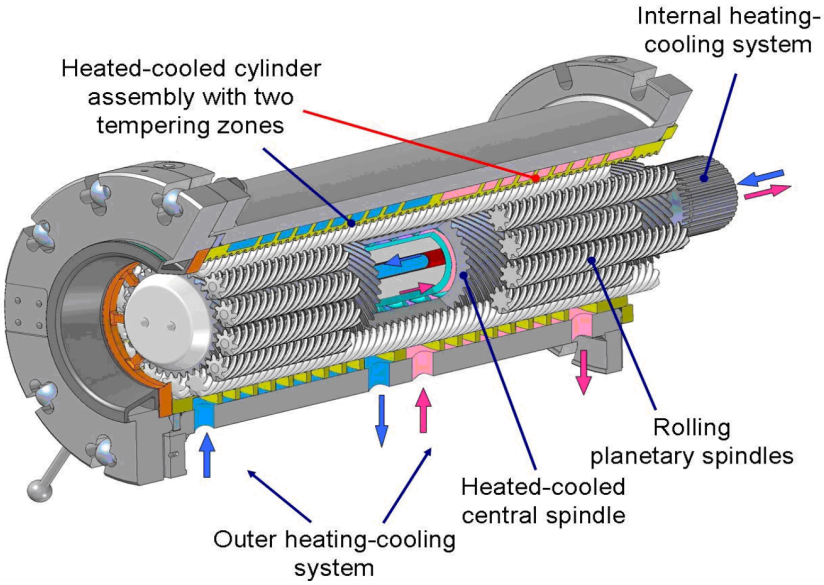


Figure 1.3: Schematics of a planetary roller extruder [13].

The results obtained on these geometries, were performed using an in-house developed C++ library, based on the official OpenFOAM release (openfoam.org). In this work we introduce the library and the major novelties that have been implemented with respect to the official release. A deeper look has been devoted to the implementation of the Immersed Boundary method, that is the major contribution of the project.

Then, we present numerical results for a complete single-screw extruder. For this benchmark we have also performed a scalability analysis of the code in order to measure its parallel performance and its ability to deal with large scale problems. As a second case, we present the simulation of a twin-screw extruder sector made by several transport and kneading modules. This allows us to exploit the modularity of our IBM and its ability in dealing with

multiple immersed boundaries. Finally we introduce the planetary roller extruder geometry and the meshing strategy that we employed to build a “semi-conforming” grid, in which the planets are the only geometries that are approximated with the IBM.

In conclusion the structure of the thesis is as follows. In Chapter 2 we introduce the governing equations and the rheological laws of non-Newtonian fluids with temperature dependent viscosity. In Chapter 3 we introduce the Box method formulation and we expose its convergence analysis for Poisson and Stokes problems. Then we add the Diffuse Interface method to the BM formulation, in order to include immersed surfaces in the formulation, and we extended the previous convergence analysis to this non-conforming case. In Chapter 4, we describe the implementation of the Immersed Boundary method, exploiting its characteristics when dealing with complex problems and geometries. We also numerically assess its accuracy properties, testing it on cases with analytical solutions and also on a case of interest, a simplified geometry of a single-screw extruder. In Chapter 5, we describe the family of SIMPLE algorithms for fluid flow problems. We also introduce the IBM in the SIMPLE formulation deriving a consistent immersed boundary condition for pressure. Finally, we introduce a new family of algorithms, inheriting the properties of the SIMPLE family, developed to deal with problems involving high-viscosity non-Newtonian flows with temperature dependent viscosity. We also introduce the implementation of the code developed and customized for the simulation of polymer mixing processes. Finally, in Chapter 6, we present the numerical results obtained with this code on realistic industrial extruders geometries: a single- and twin-screw extruders and a planetary roller extruder.

Chapter 2

Mathematical modelling of polymer mixing processes

In this chapter we presents an overview of the equations that govern polymer mixing processes, that involve motion of polymer melt. The equations involved are the incompressible Navier-Stokes equations and the temperature conservation equation. The equations are derived from mass, momentum and energy conservation laws through the use of Reynolds Transport Theorem (RTT). Then a discussion on the rheology of polymer melts is developed with a special focus on how the viscosity of the material is influenced by temperature and by the presence of additives in the compound.

In section 2.1 we derive [66] the momentum balance, mass and temperature conservation equations, with particular focus on the roles played by viscosity and viscous dissipation. In section 2.2 we introduce some viscosity law for generalized Newtonian fluids and we describe how temperature and volume filler fraction modify these laws.

2.1 Governing equations

Let Ω_0 be the reference configuration of a continuum at time 0 and let Ω_t be its configuration at time $t \in [0; +\infty)$. Its trajectory can be defined as the positions $\mathbf{x}$ and time instants t such that $\mathbf{x}$ belongs to the configuration of the body at time t Ω_t:

$$X = \{(\mathbf{x}, t) \in \mathbb{R}^3 \times [0; +\infty) : \mathbf{x} \in \Omega_t\} = \bigcup_{t \in [0;+\infty)} (\Omega_t \times \{t\}) . \tag{2.1}$$

Now we state the Reynolds Transport theorem (RTT), that is used to model the conservation of a certain intensive quantity of a material.

Theorem 2.1.1 (Reynolds Transport Theorem)**.** *Let ϕ be a sufficiently regular scalar field defined on X. Denote with $V_t \subset \Omega_t$ an arbitrary portion of the continuum at time t and let* $\mathbf{u}$ *be the Eulerian velocity field defined on X.*

Then, there holds

$$\frac{d}{dt} \int_{V_t} \phi \, dV = \int_{V_t} \frac{\partial \phi}{\partial t} \, dV + \int_{\partial V_t} \phi \mathbf{u} \cdot \mathbf{n} \, dS.$$

Remark 2.1. Theorem 2.1.1 holds also if the scalar field ϕ is replaced by a vectorial or tensorial field.

We now employ RTT to derive the conservation laws that we need to model mixing processes of polymer melts.

2.1.1 Mass conservation

The mass conservation principle states that the mass must remain constant over time. Let V_t be some portion of a continuum in its configuration Ω_t, at time t.

Let $\rho(\mathbf{x}, t)$ $[kg/m^3]$ be the density of the fluid, then the mass conservation principle reads as follows

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{V_t} \rho \,\mathrm{d}V = 0. \tag{2.2}$$

Making use of the RTT and the divergence theorem, equation (2.2) can be rewritten as

$$\int_{V_t} \frac{\partial \rho}{\partial t}\,\mathrm{d}V + \int_{\partial V_t} \rho\mathbf{u}\cdot\mathbf{n}\,\mathrm{d}S = \int_{V_t} \frac{\partial \rho}{\partial t}\,\mathrm{d}V + \int_{V_t} \nabla\cdot(\rho\mathbf{u})\,\mathrm{d}V = 0.$$

Since last equation must be satisfied independently of V_t, there holds

$$\frac{\partial \rho}{\partial t} + \nabla\cdot(\rho\mathbf{u}) = 0. \tag{2.3}$$

Moreover, we can further simplify the mass conservation equation by the incompressibility constraint. A flow is incompressible when the volume of any portion of the continuum does not change during its motion. This constraint is formulated as

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{V_t} \mathrm{d}V = 0. \tag{2.4}$$

Then, by RTT:

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{V_t} \mathrm{d}V = \int_{V_t} \frac{\partial 1}{\partial t}\,\mathrm{d}V + \int_{\partial V_t} \mathbf{u}\cdot\mathbf{n}\,\mathrm{d}S = \int_{V_t} \nabla\cdot\mathbf{u}\,\mathrm{d}V = 0.$$

Since the latter must be true for each portion of the body, the incompressibility constraint is satisfied if the following equation holds locally:

$$\nabla\cdot\mathbf{u} = 0. \tag{2.5}$$

Under incompressibility constraint the mass conservation equation becomes

$$\frac{\partial \rho}{\partial t} + \nabla\cdot(\rho\mathbf{u}) = \frac{\partial \rho}{\partial t} + \rho\nabla\cdot\mathbf{u} + \nabla\rho\cdot\mathbf{u} = \frac{\partial \rho}{\partial t} + \nabla\rho\cdot\mathbf{u} = \frac{\mathrm{d}\rho}{\mathrm{d}t} = 0, \tag{2.6}$$

that means density remain constant in time along the flow trajectory. Furthermore, if density is uniform in the reference configuration Ω_0, then each point of Ω_t has uniform density $\rho \in \mathbb{R}^+$, for any instant t.

2.1.2 Momentum balance

The linear momentum balance equation states that the rate of change of the linear momentum of a body equals the sum of surface forces and body forces applied to it:

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{V_t}\rho\mathbf{u}\,\mathrm{d}V = \int_{\partial V_t}\mathbf{s}\,\mathrm{d}S + \int_{V_t}\rho\mathbf{f}\,\mathrm{d}V, \tag{2.7}$$

where $\mathbf{s}$ and $\rho\mathbf{f}$ represent surface and volume forces, respectively.

By Cauchy theorem, we can rewrite $\mathbf{s}$ as $\mathbf{s} = \boldsymbol{\sigma}\mathbf{n}$, where $\mathbf{T}$ is the Cauchy stress tensor field and $\mathbf{n}$ the unit normal vector of V_t, directed outwards:

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{V_t}\rho\mathbf{u}\,\mathrm{d}V = \int_{\partial V_t}\boldsymbol{\sigma}\mathbf{n}\,\mathrm{d}S + \int_{V_t}\rho\mathbf{f}\,\mathrm{d}V.$$

We again use RTT with the divergence theorem:

$$\int_{V_t}\frac{\partial\rho\mathbf{u}}{\partial t}\,\mathrm{d}V + \int_{\partial V_t}(\rho\mathbf{u})(\mathbf{u}\cdot\mathbf{n})\,\mathrm{d}S = \int_{V_t}\nabla\cdot\boldsymbol{\sigma}\,\mathrm{d}V + \int_{V_t}\rho\mathbf{f}\,\mathrm{d}V,$$

that by the following vectorial identity

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

becomes

$$\int_{V_t}\frac{\partial\rho\mathbf{u}}{\partial t}\,\mathrm{d}V + \int_{\partial V_t}\rho(\mathbf{u}\otimes\mathbf{u})\mathbf{n}\,\mathrm{d}S = \int_{V_t}\nabla\cdot\boldsymbol{\sigma}\,\mathrm{d}V + \int_{V_t}\rho\mathbf{f}\,\mathrm{d}V. \tag{2.8}$$

Using divergence theorem:

$$\int_{V_t}\frac{\partial\rho\mathbf{u}}{\partial t}\,\mathrm{d}V + \int_{V_t}\nabla\cdot(\rho\mathbf{u}\otimes\mathbf{u})\,\mathrm{d}V = \int_{V_t}\nabla\cdot\boldsymbol{\sigma}\,\mathrm{d}V + \int_{V_t}\rho\mathbf{f}\,\mathrm{d}V. \tag{2.9}$$

The latter equation must hold for any portion V_t of the continuum, hence the momentum balance finally reads:

$$\frac{\partial\rho\mathbf{u}}{\partial t} + \nabla\cdot(\rho\mathbf{u}\otimes\mathbf{u}) = \nabla\cdot\boldsymbol{\sigma} + \rho\mathbf{f}. \tag{2.10}$$

Moreover, under the incompressibility assumption and with ρ uniform in the domain Ω_t, the momentum conservation reads:

$$\rho\frac{\partial\mathbf{u}}{\partial t} + \rho(\mathbf{u}\cdot\nabla)\mathbf{u} = \nabla\cdot\boldsymbol{\sigma} + \rho\mathbf{f}. \tag{2.11}$$

2.1.3 Energy conservation

The energy conservation is given by the first law of thermodynamics. It states that rate of change of energy in a system equals the sum of the rate of change of added heat to the system and the rate of work done on the system. Let Q, U, K be the incoming heat per unit

time, the internal energy and kinetic energy of the continuum, respectively. The first law of thermodynamics reads as follows:

$$Q = \frac{\mathrm{d}U}{\mathrm{d}t} + \frac{\mathrm{d}K}{\mathrm{d}t}. \tag{2.12}$$

In order to obtain the energy balance equation, we give a definition to these quantities on a control volume V_t at time t. The kinetic energy K is defined as

$$K = \frac{1}{2}\int_{V_t} \rho \mathbf{u}\cdot\mathbf{u}\,\mathrm{d}V, \tag{2.13}$$

and the internal energy as

$$U = \int_{V_t} \rho e\,\mathrm{d}V \tag{2.14}$$

where e represents the density of internal energy per unit mass. On the other hand, the rate of change of heat added to the system can be written as

$$Q = \int_{V_t} \rho r\,\mathrm{d}V - \int_{\partial V_t} h\,\mathrm{d}S$$

where h is the rate of outgoing heat per unit surface through the boundary of V_t and r is a source term. Taking advantage of Cauchy theorem, h may be written as $h = \mathbf{q}\cdot\mathbf{n}$ where $\mathbf{q}$ is the heat flux. So Q becomes such that

$$Q = \int_{V_t} \rho r\,\mathrm{d}V - \int_{\partial V_t} \mathbf{q}\cdot\mathbf{n}\,\mathrm{d}S. \tag{2.15}$$

We consider now the following result [66]:

Theorem 2.1.2 (Kinetic Energy Theorem)**.** *The rate of change of kinetic energy of V_t equals the sum of the power of external forces Π_{ext} and the power of internal forces Π_{int}.*

$$\frac{dK}{dt} = \Pi_{ext} + \Pi_{int},$$

with

$$\Pi_{ext} = \int_{\partial V_t} (\boldsymbol{\sigma}\mathbf{n})\cdot\mathbf{u}\,dS + \int_{V_t} \mathbf{u}\cdot\rho\mathbf{f}\,dV,$$
$$\Pi_{int} = -\int_{V_t} \boldsymbol{\sigma} : \mathbf{D}\,dV,$$

where **D** *is the symmetric component of velocity gradient:*

$$\mathbf{D} = \frac{1}{2}(\nabla\mathbf{u} + \nabla^{\intercal}\mathbf{u}). \tag{2.16}$$

Remark 2.2. The term Π_{int} represents the energy generated by the fluid itself, so the viscous heating, or viscous dissipation. In the context of polymer processing, the viscous heating plays a fundamental role in temperature variation of polymer melts due to the high viscosity values that they can assume.

Employing equations (2.13), (2.14), (2.15) into equation (2.12) and exploiting the expression of K by Theorem 2.1.2, there holds

$$\begin{aligned}\int_{V_t} \rho r \,\mathrm{d}V - \int_{\partial V_t} \mathbf{q}\cdot\mathbf{n}\,\mathrm{d}S =& \frac{d}{dt}\int_{V_t}\rho e\,\mathrm{d}V + \frac{d}{dt}\frac{1}{2}\int_{V_t}\rho\mathbf{u}\cdot\mathbf{u}\,\mathrm{d}V \\ =& \frac{d}{dt}\int_{V_t}\rho e\,\mathrm{d}V - \int_{V_t}\boldsymbol{\sigma}:\mathbf{D}\,\mathrm{d}V + \int_{\partial V_t}(\boldsymbol{\sigma}\mathbf{n})\cdot\mathbf{u}\,\mathrm{d}S + \int_{V_t}\mathbf{u}\cdot\rho\mathbf{f}\,\mathrm{d}V.\end{aligned} \tag{2.17}$$

Let V_0 be the portion of continuum in the reference configuration corresponding to V_t at time t. By incompressibility, we perform the change of variable from Ω_0 to Ω_t:

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\int_{V_t}\rho e\,\mathrm{d}V =& \frac{\mathrm{d}}{\mathrm{d}t}\int_{V_0}\rho_m e_m J\,\mathrm{d}V_m \\ =& \int_{V_0}\frac{\mathrm{d}\rho_m}{\mathrm{d}t}e_m J\,\mathrm{d}V_m + \int_{V_0}\rho_m\frac{\mathrm{d}e_m}{\mathrm{d}t}J\,\mathrm{d}V_m + \int_{V_0}\rho_m e_m\frac{\mathrm{d}J}{\mathrm{d}t}\,\mathrm{d}V_m\end{aligned}$$

where the subscript m stays for the material description of a spatial field and J is the Jacobian of the transformation, i.e. the determinant of the deformation gradient tensor. J represents the variation of volume of the body, hence, by incompressibility

$$\frac{\mathrm{d}J}{\mathrm{d}t} = J(\nabla\cdot\mathbf{u})_m,$$

that implies

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{V_t}\rho e\,\mathrm{d}V_m = \int_{V_0}\frac{\mathrm{d}\rho_m}{\mathrm{d}t}e_m J\,\mathrm{d}V_m + \int_{V_0}\rho_m\frac{\mathrm{d}e_m}{\mathrm{d}t}J\,\mathrm{d}V_m + \int_{V_0}\rho_m e_m J\nabla\cdot\mathbf{u}_m\,\mathrm{d}V_m.$$

Performing again a change of variable, this time from Ω_t to Ω_0, and employing mass conservation equation (2.3), the following relationship is obtained:

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\int_{V_t}\rho e\,\mathrm{d}V &= \int_{V_t}\frac{\mathrm{d}\rho}{\mathrm{d}t}e\,\mathrm{d}V + \int_{V_t}\rho\frac{\mathrm{d}e}{\mathrm{d}t}\,\mathrm{d}V + \int_{V_t}\rho e\nabla\cdot\mathbf{u}\,\mathrm{d}V \\ &= \int_{V_t}\left(\frac{\mathrm{d}\rho}{\mathrm{d}t} + \rho\nabla\cdot\mathbf{u}\right)e\,\mathrm{d}V + \int_{V_t}\rho\frac{de}{dt}\,\mathrm{d}V \\ &= \int_{V_t}\rho\frac{de}{dt}\,\mathrm{d}V.\end{aligned}$$

Using the latter relationship into equation (2.17) and exploiting the material derivative of e, we obtain that

$$\begin{aligned}\int_{V_t}\rho r\,\mathrm{d}V - \int_{\partial V_t}\mathbf{q}\cdot\mathbf{n}\,\mathrm{d}S =& \int_{V_t}\rho\frac{\partial e}{\partial t} + \int_{V_t}\rho\mathbf{u}\cdot\nabla e\,\mathrm{d}V \\ &- \int_{V_t}\boldsymbol{\sigma}:\mathbf{D}\,\mathrm{d}V + \int_{\partial V_t}(\boldsymbol{\sigma}\mathbf{n})\cdot\mathbf{u}\,\mathrm{d}S + \int_{V_t}\mathbf{u}\cdot\rho\mathbf{f}\,\mathrm{d}V.\end{aligned} \tag{2.18}$$

We now consider the Fourier law for the heat flux $\mathbf{q}$,

$$\mathbf{q} = -k\nabla T, \tag{2.19}$$

and the expression of the internal energy for incompressible fluids e,

$$e = c_P T, \tag{2.20}$$

where T is the temperature, k is the heat conductivity and c_P is the heat capacity of the fluid. Employing equations (2.19) and (2.20) into equation (2.18), the integral balance for temperature is obtained:

$$\int_{V_t} \rho \frac{\partial c_p T}{\partial t} \,\mathrm{d}V + \int_{V_t} \rho \mathbf{u} \cdot \nabla (c_p T) \,\mathrm{d}V = \int_{V_t} \rho r \,\mathrm{d}V + \int_{\partial V_t} (k \nabla T) \cdot \mathbf{n} \,\mathrm{d}S + \int_{V_t} \boldsymbol{\sigma} : \mathbf{D} \,\mathrm{d}V - \int_{\partial V_t} (\boldsymbol{\sigma} \mathbf{n}) \cdot \mathbf{u} \,\mathrm{d}S - \int_{V_t} \mathbf{u} \cdot \rho \mathbf{f} \,\mathrm{d}V.$$

Observing that, by divergence theorem and incompressibility constraint,

$$- \int_{\partial V_t} (\boldsymbol{\sigma} \mathbf{n}) \cdot \mathbf{u} \,\mathrm{d}S = \int_{V_t} \boldsymbol{\sigma} \nabla \cdot \mathbf{u} \,\mathrm{d}V = 0$$

and then applying again divergence theorem, we obtain the energy equation for the temperature in integral form:

$$\int_{V_t} \rho \frac{\partial c_p T}{\partial t} \,\mathrm{d}V + \int_{V_t} \rho \mathbf{u} \cdot \nabla (c_p T) \,\mathrm{d}V = \int_{V_t} \rho r \,\mathrm{d}V + \int_{V_t} \nabla \cdot (k \nabla T) \,\mathrm{d}V + \int_{V_t} \boldsymbol{\sigma} : \mathbf{D} \,\mathrm{d}V - \int_{V_t} \mathbf{u} \cdot \rho \mathbf{f} \,\mathrm{d}V. \tag{2.21}$$

Ultimately, since equation (2.21) must hold for any V_t, there holds

$$\rho \frac{\partial c_p T}{\partial t} + \rho \mathbf{u} \cdot \nabla (c_p T) - \nabla \cdot (k \nabla T) = \boldsymbol{\sigma} : \mathbf{D} + \rho r - \rho \mathbf{u} \cdot \mathbf{f} \tag{2.22}$$

where the term $\boldsymbol{\sigma} : \mathbf{D}$ is a heat source modelling the viscous friction as the product between viscous stress and gradient tensor of deformation.

2.1.4 Motion of a generalized Newtonian fluid

We have now all the ingredients to set the mathematics modelling the motion of a polymeric material during its processing. As mentioned before, the viscosity of a polymer is not uniform but varies in space and time. The behaviour of generalized Newtonian fluids is characterised by the dependence of the viscosity on the shear rate and possibly temperature. Hence we need a more general definition to define the constitutive relation for the Cauchy stress tensor.

There is a multitude of constitutive equations proposed for polymer melts. However, only a few have been used to solve actual polymer processing problems. The generalized Newtonian fluid models are widely used in polymer processing flow analysis, since they are capable of describing well the strong dependence on shear rate of melts [70]. In particular, we consider the case in which the viscosity depends on the instantaneous value of the shear rate and not on the history experienced by the material, as for viscoelastic fluids.

In general, the Cauchy stress tensor of a *Stokesian* fluid can be written as:

$$\boldsymbol{\sigma} = -p\mathrm{I} + \boldsymbol{\tau}, \tag{2.23}$$

where p is the pressure and $\boldsymbol{\tau}$ is the viscous stress tensor. Morover, we also assume that $\boldsymbol{\tau}$ is a continuous function of $\nabla\mathbf{u}$ and that it is frame invariant, i.e. it does not change with respect to different reference frame, even non-inertial.

If we consider Newtonian fluids, we have that

$$\boldsymbol{\tau} = \lambda\nabla\cdot\mathbf{u}\mathrm{I} + 2\mu\mathbf{D}, \tag{2.24}$$

where μ is the dynamic viscosity of the fluid and λ is the second viscosity coefficient. Generalized Newtonian fluids present a viscosity $\mu = \mu(\dot\gamma, T)$ that depends from the shear rate $\dot\gamma$, defined as

$$\dot\gamma = \sqrt{2\mathbf{D}:\mathbf{D}}, \tag{2.25}$$

and possibly by temperature. We will discuss in next sections the explicit dependence of viscosity from shear rate and temperature.

Thus, in general we have that

$$\boldsymbol{\sigma} = \left(-p + \lambda\nabla\cdot\mathbf{u}\right)\mathrm{I} + 2\mu(\dot\gamma, T)\mathbf{D},$$

that by incompressibility constraint becomes

$$\boldsymbol{\sigma} = -p\mathrm{I} + 2\mu(\dot\gamma, T)\mathbf{D}. \tag{2.26}$$

Before writing the final system of partial differential equations that describes the motion of a generalized Newtonian fluid, we have to compute the divergence of Cauchy stress tensor and the viscous heating source term:

$$\begin{aligned}
\nabla\cdot\boldsymbol{\sigma} =& -\nabla\cdot(p\mathrm{I}) + \nabla\cdot(2\mu(\dot\gamma,T)\mathbf{D})\\
=& -\nabla p + \nabla\cdot\left(\mu(\dot\gamma,T)(\nabla\mathbf{u} + \nabla^\intercal\mathbf{u})\right),\\
\boldsymbol{\sigma}:\mathbf{D} =& \boldsymbol{\sigma}:\nabla\mathbf{u}\\
=& -p\nabla\cdot\mathbf{u} + \mu(\dot\gamma,T)(\nabla\mathbf{u} + \nabla^\intercal\mathbf{u}):\nabla\mathbf{u}\\
=& \mu(\dot\gamma,T)(\nabla\mathbf{u} + \nabla^\intercal\mathbf{u}):\nabla\mathbf{u}.
\end{aligned}$$

Taking now equations (2.11) divided by ρ, (2.6) and (2.22), the resulting governing system, that describes the motion of an incompressible generalized Newtonian fluid, reads: let ν be the kinematic viscosity,

$$\frac{\partial\mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} - \nabla\cdot(\nu(\dot\gamma,T)(\nabla\mathbf{u} + \nabla^\intercal\mathbf{u})) = -\nabla\frac{p}{\rho} + \mathbf{f}, \tag{2.27a}$$

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

$$\rho\frac{\partial c_pT}{\partial t} + \rho\mathbf{u}\cdot\nabla(c_pT) - \nabla\cdot(k\nabla T) = \mu(\dot\gamma,T)(\nabla\mathbf{u} + \nabla^\intercal\mathbf{u}):\nabla\mathbf{u} + \rho r - \rho\mathbf{u}\cdot\mathbf{f}. \tag{2.27c}$$

2.1.4.1 Non-inertial reference frames

The formulation of Navier-Stokes equations is in most cases formulated with respect to a reference frame that is still or moving with constant velocity. However, there exist applications in which it is convenient to set the reference frame, for instance, on points that are rotating.

Indeed, if considering for example the single-screw extruder (c.f. Chapters1 and 6), if one a flow driven by the central shaft, a time dependent simulation has to be implemented because of the presence of the teeth that moves in time. On the other hand, if the reference frame is placed on the screw, thus rotates with the screw. In this case, the problem can be considered steady-state because the screw will be still and the barrel will move with opposite velocity with respect to the original screw velocity. With this approach a lot of computational time can be saved.

When dealing with non-inertial reference frames, the Navier-Stokes equations have to be modified because we want to solve equations for the relative velocity of the new frame and not for the absolute velocity.

In the following, we rewrite the Navier-Stokes equations for the relative velocity computed on the absolute reference frame. For the derivation we considered some results from [66].

Let then $\mathbf{r} = \mathbf{x} - \mathbf{Q}$ be the position vector of a point $\mathbf{x}$ with respect to the axis passing by point $\mathbf{Q}$ directed as its rotational velocity $\boldsymbol{\omega}$. Let $\mathbf{u}$ be the absolute velocity and $\mathbf{u}_r$ be the velocity relative to the non-inertial reference frame. Then, the following law holds.

Theorem 2.1.3 (Galileo)**.** *The velocity of a point, computed with respect to an absolute frame of reference, is related with the velocity computed from a moving reference frame by*

$$\mathbf{u} = \mathbf{u}_r + \mathbf{u}_\tau, \tag{2.28}$$

where $\mathbf{u}_\tau = \mathbf{u}_Q + \boldsymbol{\omega} \wedge \mathbf{r}$ *is the drift velocity and* $\mathbf{u}_q$ *is the velocity of point* $\mathbf{Q}$*.*

A similar result holds for the acceleration.

Theorem 2.1.4 (Coriolis)**.** *The acceleration of a point, computed with respect to an absolute frame of reference, is related with the acceleration computed from a moving reference frame by*

$$\frac{d\mathbf{u}}{dt} = \frac{d\mathbf{u}_r}{dt} + \frac{d\mathbf{u}_Q}{dt} + \frac{d\mathbf{u}_C}{dt}, \tag{2.29}$$

where

$$\frac{d\mathbf{u}_Q}{dt} = \frac{d\mathbf{Q}}{dt} + \dot{\boldsymbol{\omega}} \wedge \mathbf{r} + \boldsymbol{\omega} \wedge (\boldsymbol{\omega} \wedge \mathbf{r})$$

is the drift acceleration and

$$\frac{d\mathbf{u}_C}{dt} = 2\boldsymbol{\omega} \wedge \mathbf{u}_r$$

is the Coriolis acceleration.

In the light of Theorems 2.1.3 and 2.1.4, we can rewrite Navier-Stokes equations in terms of the relative velocity $\mathbf{u}_r$.

First consider some operator of Navier-Stokes equations with respect to the relative velocity.

$$\begin{aligned}\frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t} =& \frac{\mathrm{D}\mathbf{u}_r}{\mathrm{D}t} + \frac{\mathrm{d}\mathbf{Q}}{\mathrm{d}t} + \dot{\boldsymbol{\omega}} \wedge \mathbf{r} + \boldsymbol{\omega} \wedge (\boldsymbol{\omega} \wedge \mathbf{r}) + 2\boldsymbol{\omega} \wedge \mathbf{u}_r,\\ \nabla \cdot \mathbf{u} =& \nabla \cdot \mathbf{u}_r + \nabla \cdot (\boldsymbol{\omega} \wedge \mathbf{r}) = \nabla \cdot \mathbf{u}_r,\\ \nabla \mathbf{u} =& \nabla \mathbf{u}_r + \nabla(\boldsymbol{\omega} \wedge \mathbf{r}) = \nabla \mathbf{u}_r + [\boldsymbol{\omega}]_\wedge,\end{aligned} \tag{2.30}$$

where

$$[\boldsymbol{\omega}]_\wedge = \begin{bmatrix} 0 & -\omega_k & \omega_j \\ \omega_k & 0 & -\omega_i \\ -\omega_j & \omega_i & 0 \end{bmatrix},$$

from which follows

$$\nabla \cdot \nabla \mathbf{u} = \nabla \cdot \nabla \mathbf{u}_r + \nabla \cdot \nabla(\boldsymbol{\omega} \wedge \mathbf{r}) = \nabla \cdot \nabla \mathbf{u}_r. \tag{2.31}$$

Then, setting

$$\mathbf{f}_r = \frac{\mathrm{d}\mathbf{Q}}{\mathrm{d}t} + \dot{\boldsymbol{\omega}} \wedge \mathbf{r} + \boldsymbol{\omega} \wedge (\boldsymbol{\omega} \wedge \mathbf{r}) + 2\boldsymbol{\omega} \wedge \mathbf{u}_r,$$

the Navier-Stokes equations, written with respect to relative velocity, read

$$\begin{aligned}\frac{\partial \mathbf{u}_r}{\partial t} + (\mathbf{u}_r \cdot \nabla)\mathbf{u}_r - \nabla \cdot (\nu(\dot{\gamma}, T)(\nabla \mathbf{u}_r + \nabla^\intercal \mathbf{u}_r)) =& -\nabla \frac{p}{\rho} + \mathbf{f} - \mathbf{f}_r,\\ \nabla \cdot \mathbf{u}_r =& 0,\\ \rho \frac{\partial c_p T}{\partial t} + \rho(\mathbf{u}_r + \boldsymbol{\omega} \wedge \mathbf{r}) \cdot \nabla(c_p T) - \nabla \cdot (k \nabla T) =& \mu(\dot{\gamma}, T)(\nabla \mathbf{u}_r + \nabla^\intercal \mathbf{u}_r) : (\nabla \mathbf{u}_r + [\boldsymbol{\omega}_\wedge])\\ &+ \rho r - \rho(\mathbf{u}_r + \boldsymbol{\omega} \wedge \mathbf{r}) \cdot \mathbf{f}.\end{aligned} \tag{2.32}$$

Another way of writing the Navier-Stokes equations with respect to a relative reference frame is the following. We set the system in the relative frame but we solve the equations with respect to the absolute velocity. In order to obtain this formulation, equations (2.32) have to be rewritten in order to have the absolute velocity as the convected velocity. First, rewrite the convective term in the following way:

$$\begin{aligned}(\mathbf{u}_r \cdot \nabla)\mathbf{u}_r =& (\mathbf{u}_r \cdot \nabla)(\mathbf{u} - \mathbf{u}_Q - \boldsymbol{\omega} \wedge \mathbf{r})\\ =& (\mathbf{u}_r \cdot \nabla)\mathbf{u} - (\mathbf{u}_r \cdot \nabla)\mathbf{u}_Q - (\mathbf{u}_r \cdot \nabla)\boldsymbol{\omega} \wedge \mathbf{r}\\ =& (\mathbf{u}_r \cdot \nabla)\mathbf{u} - [\boldsymbol{\omega}_\wedge]\mathbf{u}_r = (\mathbf{u}_r \cdot \nabla)\mathbf{u} - \boldsymbol{\omega} \wedge \mathbf{u}_r.\end{aligned} \tag{2.33}$$

Then, comparing the latter against the source term $\mathbf{f}_r$, it holds

$$\begin{aligned}(\mathbf{u}_r \cdot \nabla)\mathbf{u}_r + \mathbf{f}_r =& (\mathbf{u}_r \cdot \nabla)\mathbf{u}_r + \frac{\mathrm{d}\mathbf{Q}}{\mathrm{d}t} + \dot{\boldsymbol{\omega}} \wedge \mathbf{r} + \boldsymbol{\omega} \wedge (\boldsymbol{\omega} \wedge \mathbf{r}) + 2\boldsymbol{\omega} \wedge \mathbf{u}_r\\ =& (\mathbf{u}_r \cdot \nabla)\mathbf{u} - \boldsymbol{\omega} \wedge \mathbf{u}_r + \frac{\mathrm{d}\mathbf{Q}}{\mathrm{d}t} + \dot{\boldsymbol{\omega}} \wedge \mathbf{r} + \boldsymbol{\omega} \wedge (\boldsymbol{\omega} \wedge \mathbf{r}) + 2\boldsymbol{\omega} \wedge \mathbf{u}_r\\ =& (\mathbf{u}_r \cdot \nabla)\mathbf{u} + \frac{\mathrm{d}\mathbf{Q}}{\mathrm{d}t} + \dot{\boldsymbol{\omega}} \wedge \mathbf{r} + \boldsymbol{\omega} \wedge ((\boldsymbol{\omega} \wedge \mathbf{r}) + \mathbf{u}_r)\\ =& (\mathbf{u}_r \cdot \nabla)\mathbf{u} + \frac{\mathrm{d}\mathbf{Q}}{\mathrm{d}t} + \dot{\boldsymbol{\omega}} \wedge \mathbf{r} + \boldsymbol{\omega} \wedge \mathbf{u}.\end{aligned} \tag{2.34}$$

Finally, setting

$$\tilde{\mathbf{f}}_r = \frac{\mathrm{d}\mathbf{Q}}{\mathrm{d}t} + \dot{\boldsymbol{\omega}} \wedge \mathbf{r},$$

the Navier-Stokes equations formulated in the relative reference frame with respect to the absolute velocity read

$$\begin{aligned} \frac{\partial \mathbf{u}_r}{\partial t} + (\mathbf{u}_r \cdot \nabla)\mathbf{u} + \boldsymbol{\omega} \wedge \mathbf{u} - \nabla \cdot (\nu(\dot{\gamma}, T)(\nabla \mathbf{u} + \nabla^\intercal \mathbf{u})) &= -\nabla \frac{p}{\rho} + \mathbf{f} - \tilde{\mathbf{f}}_r, \\ \nabla \cdot \mathbf{u} &= 0, \\ \rho \frac{\partial c_p T}{\partial t} + \rho \mathbf{u} \cdot \nabla (c_p T) - \nabla \cdot (k \nabla T) &= \mu(\dot{\gamma}, T)(\nabla \mathbf{u} + \nabla^\intercal \mathbf{u}) : \nabla \mathbf{u} \\ &\quad + \rho r - \rho \mathbf{u} \cdot \mathbf{f}. \end{aligned} \tag{2.35}$$

Remark 2.3. The computation of the shear rate does not change in a non-inertial reference frame:

$$\begin{aligned} \dot{\gamma} &= \sqrt{2\mathbf{D} : \mathbf{D}} = \sqrt{\frac{1}{2}|\nabla \mathbf{u} + \nabla^\intercal \mathbf{u}|^2} \\ &= \sqrt{\frac{1}{2}|\nabla \mathbf{u}_r + [\boldsymbol{\omega}]_\wedge + \nabla^\intercal \mathbf{u}|^2 + [\boldsymbol{\omega}]_\wedge^\intercal} \\ &= \sqrt{\frac{1}{2}|\nabla \mathbf{u}_r + \nabla^\intercal \mathbf{u}|^2}, \end{aligned} \tag{2.36}$$

because of $[\boldsymbol{\omega}]_\wedge$ is skew-symmetric.

This formulation is convenient because it avoids to write new set of boundary conditions for $\mathbf{u}_r$. This allows also to apply relative reference frames just in some domain subsets. The latter approach is called multi-reference frame (MRF) [65].

2.1.4.2 Arbitrary Lagrangian-Eulerian formulation

Another popular formulation of the Navier-Stokes equations is the Arbitrary Lagrangian-Eulerian (ALE) formulation. This formulation is particularly useful when dealing with moving domains. For instance, in Finite Volume applications, when the computational grid moves or deforms, ad-hoc strategies have to be implemented.

To write the ALE formulation of the momentum balance equation, let $V(t)$ be a control volume changing in time, integrate equation (2.11) on $V(t)$ and integrate by parts the other terms:

$$\int_V (t) \frac{\partial \mathbf{u}}{\partial t} \mathrm{d}x + \int_{\partial V(t)} \mathbf{u}\mathbf{u} \cdot \mathbf{n} \mathrm{d}s = \int_{\partial V(t)} \frac{\boldsymbol{\sigma}}{\rho} \mathbf{n} \mathrm{d}s + \int_{V(t)} \mathbf{f} \mathrm{d}x.$$

Apply now the RTT to the temporal term:

$$\int_{V(t)} \frac{\partial \mathbf{u}}{\partial t} \mathrm{d}x = \frac{\mathrm{d}}{\mathrm{d}t} \int_{V(t)} \mathbf{u} \mathrm{d}x - \int_{\partial V(t)} \mathbf{u}\mathbf{v}_s \cdot \mathbf{n} \mathrm{d}s$$

where $\mathbf{v}_s$ is the velocity of the surface. Then the integral ALE formulation of the momentum balance equation becomes

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{V(t)}\mathbf{u}\mathrm{d}x+\int_{\partial V(t)}\mathbf{u}(\mathbf{u}-\mathbf{v}_s)\cdot\mathbf{n}\mathrm{d}s=\int_{\partial V(t)}\frac{\boldsymbol{\sigma}}{\rho}\mathbf{n}\mathrm{d}s+\int_{V(t)}\mathbf{f}\mathrm{d}x. \tag{2.37}$$

For what concerns the mass conservation and energy equations one can employ the same procedure, however, by incompressibility, mass conservation will remain the same. For the energy equation, by integrating on $V(t)$ and applying RTT, we obtain the following ALE formulation:

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\int_{V(t)}\rho c_pT\mathrm{d}x+\int_{\partial V(t)}\rho c_pT(\mathbf{u}-\mathbf{v}_s)\cdot\mathbf{n}\mathrm{d}s-\int_{\partial V(t)}k\nabla T\cdot\mathbf{n}\mathrm{d}s=\\ \int_{V(t)}\left[\mu(\dot\gamma,T)(\nabla\mathbf{u}+\nabla^{\intercal}\mathbf{u}):\nabla\mathbf{u}+\rho r-\rho\mathbf{u}\cdot\mathbf{f}\right]\mathrm{d}x.\end{aligned} \tag{2.38}$$

In conclusion, what changes from the original formulation of the Navier-Stokes equation is the fact that the convection velocity is replaces by the velocity relative to the domain motion, $\mathbf{u}-\mathbf{v}_s$, and the temporal derivative is now applied to the whole integral of the control volume. In temporal discretization schemes, this turns in the fact that both current and older volumes are involved in the discretization.

2.2 Rheological characterization of polymer melts

In previous section, we recalled the equations that govern the motion of a generalized Newtonian fluid. However, equations (2.27) are not closed until we define the constitutive relation for viscosity $\nu(\dot\gamma,T)$.

Viscosity is defined as the proportionality factor between stress and shear rate. Viscosity is the lack of slipperiness of a fluid, from which a resistance proportional to the velocity gradients arises. This is equivalent to take relation (2.24) between stress and shear rate, considering an incompressible fluid

$$\boldsymbol{\tau}=2\mu\mathbf{D}. \tag{2.39}$$

Thus, modelling μ is equivalent to model how a material deforms under a certain force. The science that studies this is called rheology.

Not all the materials show a linear behaviour between stress and shear like Newtonian fluids, modelling viscosity as a constant. Especially polymers have been found to obey non-linear laws where also the viscosity itself depends on the shear rate. The law describing this dependence is called *rheological law*. We have reported, for different values of rheological parameters, some examples of rheological laws in Figure 2.1 that we now present in the following.

The most widely used form of the general viscous constitutive relation is the power law model

$$\mu(\dot\gamma)=\mu_0\left(\lambda\dot\gamma\right)^{n-1}. \tag{2.40}$$

The parameters μ_0 [Pa·s], n and λ [s] represent the consistency factor, i.e. the zero-shear viscosity, the power law exponent and the relaxation time of the fluid, respectively. This is the simplest law that can be employed. To have an idea of its behaviour, consider $\mu_0 = 1$ and $\lambda = 1$, $\dot{\gamma} \in (0, 100)$ and $n \in (0, 2)$. As it can be seen in Figure 2.1a, if n is less than one, viscosity decreases with shear rate, so it shows a *shear-thinning* behaviour. On the contrary, if n is more than one, the fluid shows a *shear-thickening* behaviour. If $n = 1$, the fluid behaves as it is Newtonian.

When, for instance, $n < 1$, power law is suitable for the approximation of viscosity in regimes of high shear rate values. On the other hand, when shear rate goes to zero, the fluid assumes infinitely large viscosity values. However, what is usually observed experimentally is that at low shear rate viscosity goes to a constant and not to infinity, as power law describes [98].

Thus, Bird-Carreau-Yasuda law has been introduced:

$$\mu(\dot{\gamma}) = \mu_0 \left[1 + (\lambda\dot{\gamma})^a\right]^{\frac{n-1}{a}}, \tag{2.41}$$

which behaviour is reported in Figure 2.1b. Bird-Carreau-Yasuda law shows a power law behaviour in presence of high shear rates, indeed

$$\mu(\dot{\gamma}) \simeq \mu_0(\lambda\dot{\gamma})^{n-1},$$

as $\dot{\gamma} \to \infty$. On the other hand, when the shear rate goes to zero, the viscosity assumes a Newtonian behaviour

$$\mu(\dot{\gamma}) \simeq \mu_0,$$

as $\dot{\gamma} \to 0$.

The last law that we want to mention is the Cross law. Some types of dilute polymers and suspensions have been observed to present a Newtonian behaviour, as the shear rate goes to zero. Thus, Cross law is a power law link between two Newtonian behaviours, one at high shear rates and the other at low shear rates (Figure 2.1c). The law reads

$$\mu(\dot{\gamma}) = \mu_\infty + \frac{(\mu_0 - \mu_\infty)}{1 + (\lambda\dot{\gamma})^{1-n}}. \tag{2.42}$$

Cross law shows a power law behaviour in presence of high shear rates, up to μ_0, indeed

$$\mu(\dot{\gamma}) - \mu_\infty \simeq (\mu_0 - \mu_\infty)(\lambda\dot{\gamma})^{n-1},$$

as $\dot{\gamma} \to \infty$. On the other hand, when the shear rate goes to zero, we have

$$\mu(\dot{\gamma}) \simeq \mu_\infty.$$

Temperature dependence Polymer mixing applications can often involve temperature as a process parameter and temperature plays a fundamental role in changing the viscosity value. for non-isothermal processing problems, the temperature dependence of viscosity can be as important as shear rate dependence. In general, a temperature raise implies a

viscosity decrease for any liquid. To model this behaviour, let us consider the power law as an example and let $H(T)$ be the the temperature shift factor. Then the modified power law reads

$$\mu(\dot{\gamma}, T) = H(T)\mu_0 \left(\lambda\dot{\gamma}\right)^{n-1}. \tag{2.43}$$

$H(T)$ acts on the value of the consistency factor μ_0. Clearly, $H(T)$ has to model the behaviour described above, hence it will decrease as temperature increases and vice-versa.

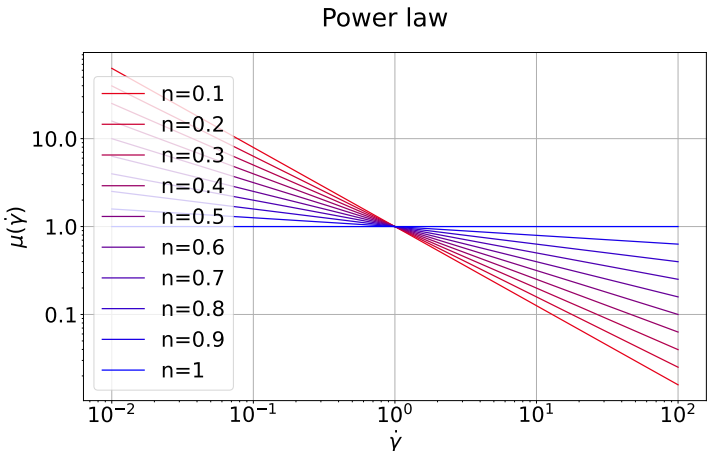
One simple model to define $H(T)$ is the Arrhenius model. Let α [K] be the activation energy, T_α [K] be the activation temperature and T_0 be the reference temperature.

$$H(T) = \exp\left[\alpha\left(\frac{1}{T - T_0} - \frac{1}{T_\alpha - T_0}\right)\right]. \tag{2.44}$$

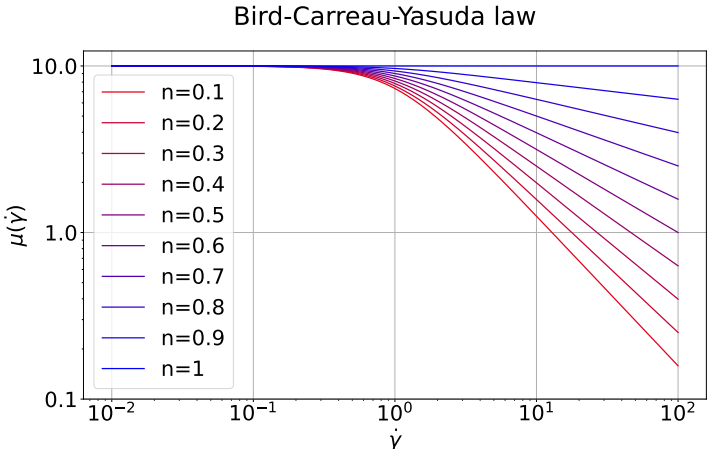
Another popular model to described the change of viscosity with respect to temperature is the Williams-Landel-Ferry (WLF) model [64]. The WLF model reads

$$\log(H(T)) = c_1\left[\frac{T_r - T_a}{c_2 + T_r - T_a} - \frac{T - T_a}{c_2 + T - T_a}\right], \tag{2.45}$$

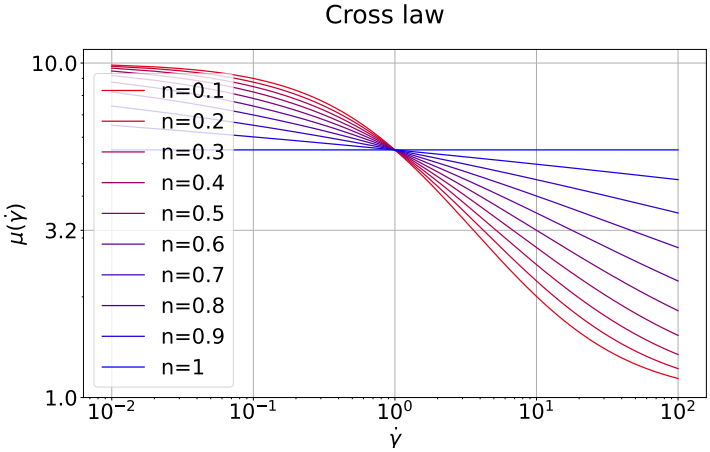
where T_r [K] is the reference temperature, T_a is the glass transition temperature and c_1, c_2 are two constant.



(a) Power law viscosity for various values of n in both shear thinning and shear thickening regimes.



(b) Bird-Carreau-Yasuda law viscosity for various values of n for a shear thinning fluid, where $\mu_0 = 100$ [Pa·s] and $a = 2$.



(c) Cross law viscosity for various values of n for a shear thinning fluid, where $\mu_0 = 100$ [Pa·s] and $\mu_\infty = 1$ [Pa·s].

Figure 2.1: Examples of rheological laws for shear thinning fluids.

Filler dependence The last quantity that can influence the viscosity of a polymer is the filler volume fraction. Usually, polymer melts are mixed with opportune particulates in order to enhance the mechanical properties of the final product. The addition of a solid compound in a polymeric matrix produces a solid-liquid suspension with different rheological properties with respect to those of the unfilled polymeric matrix.

Let us define the filler volume fraction of a polymer matrix. Let V be the volume of a pure polymer matrix. Let $V_{\text{dispersed}}$ be the volume of filler dispersed into polymeric matrix. The volume fraction ϕ_f of the filler is defined as

$$\phi_f = \frac{V_{\text{dispersed}}}{V + V_{\text{dispersed}}}. \tag{2.46}$$

The viscosity of the filled polymer in presence of a concentration of the suspended particulates, represented by the filler volume fraction ϕ_f, can be modelled similarly to what is done for temperature shift factor [114].

The modelling of the filler volume fraction dependence of viscosity is subordinated by the modelling of relative viscosity. The relative viscosity μ_r of a filled compound is defined as the ratio between the viscosity of the filled compound and the viscosity of the unfilled polymer matrix, at the same shear stress τ and temperature T [90].

$$\mu_r = \frac{\mu_{\text{filled}}(\dot{\gamma}_1, T)|_\tau}{\mu_{\text{pure}}(\dot{\gamma}_2, T)|_\tau}, \tag{2.47}$$

where $\dot{\gamma}_1$ and $\dot{\gamma}_2$ are such that $\tau = \mu_{\text{filled}}(\dot{\gamma}_1, T)\dot{\gamma}_1 = \mu_{\text{pure}}(\dot{\gamma}_2, T)\dot{\gamma}_2$.

Two considerations can be made on the presence of a filler in a polymer matrix. Since the dispersion of solid particulates increases internal friction and, consequently, energy dissipation, the viscosity of a filled compound must higher than the viscosity of the unfilled polymer matrix. The higher is the volume fraction of the filler, the higher is its concentration and so the higher is the relative viscosity of the filled compound. Thus, the following relation holds for the relative viscosity:

$$\frac{\partial \mu_r(\phi)}{\partial \phi} \geq 0. \tag{2.48}$$

On the other hand, the filled polymer viscosity must be consistent with the unfilled one, so the following must hold,

$$\mu_r(0) = 1, \tag{2.49}$$

since, if the filler volume fraction is null, the viscosity of the compound is the viscosity of the pure polymer.

Another aspect that has to be take into account is the maximum packing volume fraction. The maximum packing filler volume fraction ϕ_M is defined as the filler volume fraction corresponding to the maximum packing arrangement of filler particles, while still retaining a continuous material. This quantity depends on the shape and size distribution of filler particles.

Many models have been proposed in the literature to describe the behaviour of μ_r with respect to ϕ_f. For instance the Maron-Pierce model [100, 114] reads:

$$\mu_r(\phi_f) = \left(1 - \frac{\phi_f}{\phi_M}\right)^{-2}. \tag{2.50}$$

The expression of μ_r can be employed in the various rheological laws presented above. Consider first a Bird-Carreau viscosity law, that, for a filled polymer, reads:

$$\mu(\dot{\gamma}, T) = H(T)\mu_r(\phi_f)\mu_0 \left[1 + (\mu_r(\phi_f)\lambda\dot{\gamma})^2\right]^{\frac{n-1}{2}}. \tag{2.51}$$

Considering equation (2.51) for high shear rates, we can obtain the power law for filled compounds:

$$\mu(\dot{\gamma}, T) = H(T)\mu_r(\phi_f)^n\mu_0(\lambda\dot{\gamma})^{n-1}. \tag{2.52}$$

Finally, also Cross law can be modified in order to take into account the effect of the filler on the viscosity [114]:

$$\mu(\dot{\gamma}) = \frac{H(T)\mu_0\mu_r(\phi_f)}{1 + (\mu_r(\phi_f)\lambda\dot{\gamma})^{1-n}}. \tag{2.53}$$

2.3 Rheometry

The dependence of the rheology on the filler volume fraction has been modelled with the approach introduced in the previous section and it has been implemented within the software library that will be presented in details in Chapter 5. The analysis of filler dependence in polymer rheologies has been carried out in the context of this research, together with a comparison between numerical models and experimental data coming from experiments on a capillary rheometer. The results have been presented in [41]. This aspect has not been extensively analysed in the present work. However, for the sake of completeness, we recall here the analysis behind the experimental data fitting to find rheological parameters of a polymeric compound. Further details can be found in [41].

Rheometry is an experimental technique employed to quantitatively determine the parameters of rheological laws, i.e. the materials rheological properties. Rheometry implies the usage of apparati called rheometers. Rheometers are devices where a fluid is poured and undergoes extensional flows (extensional rheometers) rather than steady or oscillatory shear flows (shear rheometers). There exist many types of rheometers, such as Couette rheometer, plate-plate rheometer, cone-plate rheometer and capillary rheometer, depending on which range of shear rate one wnats to investigate.

In particular, shear rheometers operate by imposing a kinematic quantity, such as velocity or displacement, and then measuring a dynamic quantity, like pressure drop or shear stresses. Since rheometers reproduce flows which profile is known, we are able to estimate the viscosity, that is the proportionality factor between shear rate, generated by the kinematic quantity, and the stress estimated from the dynamic quantity. Moreover, rheometers are thermostatated in order to reproduce flows that can be approximated as isothermal flows.

In this section, we focus on the capillary rheometer.

2.3.1 Capillary Rheometer

Capillary rheometers are devices composed by a tank, or reservoir, where the fluid is collected, and then a piston pushes the fluid into a duct, called capillary. The reservoir and the capillary have very different cross sectional areas. The capillary can have both circular (capillary die) or rectangular (slit die) section. This setup is especially useful to study high viscous flows over a broad range of shear rate values.

The velocity at which the piston moves determines the shear rate, while the pressure drop between the tank and the capillary outflow is measured in order to estimate the shear stress. We can determine both shear rate and shear stress because, being the capillary a cylindrical duct, a Poiseuille flow, and so an analytical flow, develops in the duct.

Usually, the fluid is brought to a constant temperature before being pushed into the duct to obtain an isothermal flow. However, this is not obvious, especially for high viscous fluids, where viscous friction dissipates a lot of energy during the motion.

In the following section we introduce the so called Rabinowitz and Mooney analyses. These analyses are performed in order to fit experimental data with a rheological viscosity law in order to determine the law parameters. The first is employed in the case of no slip boundary conditions while the second for slip boundary conditions.

The aim of the analyses is to find a simplified representation of the relationship between viscous stress and shear rate, in which viscosity is the proportionality factor. In particular, the analyses focus on the evaluation of a representative value for the shear rate, computed from the flow rate imposed to the capillary inflow, and a value for the viscous stress, computed from the pressure drop measured between the inflow and the outflow of the capillary die. The resultant relation will be

$$\mu \simeq \frac{\tau_w}{\dot{\gamma}_w},$$

where the subscript w stays for wall, because τ_w and $\dot{\gamma}_w$ are the viscous stress and the shear rate, respectively, computed at the wall of the duct.

2.3.2 Isothermal Poiseuille flow in a cylindrical duct

Consider an incompressible power law fluid flowing inside a cylindrical channel of radius R. Assuming that the flow is isothermal, we discard the energy equation and the dependence of viscosity on temperature. Then, the equations governing the motion of the fluid read

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} = -\nabla\frac{p}{\rho} + \nabla\cdot(\nu(\dot{\gamma})(\nabla\mathbf{u} + \nabla^{\intercal}\mathbf{u})) + \mathbf{f}, \tag{2.54a}$$

$$\nabla\cdot\mathbf{u} = 0 \tag{2.54b}$$

$$\tag{2.54c}$$

where $\mathbf{u} = [u_x, u_y, u_z]^{\intercal}$.

Then, we consider a power law fluid, which constitutive equation reads

$$\nu(\dot{\gamma}) = \frac{\mu_0}{\rho}(\lambda\dot{\gamma})^{n-1} = \nu_0\dot{\gamma}^{n-1}. \tag{2.55}$$

Consider now the cylindrical coordinates r, θ, z, in such a way that the z axis is aligned with the axis of the capillary duct. With a slight abuse of notation, let $\mathbf{u} = [u_r, u_\theta, u_z]^\intercal$ be the velocity in cylindrical coordinates and $\tilde{p} = \frac{p}{\rho}$. Let us now write system (2.54) in cylindrical coordinates:

$$\frac{\partial u_r}{\partial t} + u_r\frac{\partial u_r}{\partial r} + \frac{u_\theta}{r}\frac{\partial u_r}{\partial \theta} - \frac{u_\theta^2}{r} + u_z\frac{\partial u_r}{\partial z} = -\frac{\partial \tilde{p}}{\partial r} + \left(\frac{1}{r}\frac{\partial (r\tau_{rr})}{\partial r} + \frac{1}{r}\frac{\partial \tau_{r\theta}}{\partial \theta} - \frac{\tau_{\theta\theta}}{r} + \frac{\partial \tau_{rz}}{\partial z}\right) + f_r, \tag{2.56a}$$

$$\frac{\partial u_\theta}{\partial t} + u_r\frac{\partial u_\theta}{\partial r} + \frac{u_\theta}{r}\frac{\partial u_\theta}{\partial \theta} - \frac{u_r u_\theta}{r} + u_z\frac{\partial u_\theta}{\partial z} = -\frac{\partial \tilde{p}}{\partial \theta} + \left(\frac{1}{r^2}\frac{\partial (r^2\tau_{r\theta})}{\partial r} + \frac{1}{r}\frac{\partial \tau_{\theta\theta}}{\partial \theta} + \frac{\partial \tau_{\theta z}}{\partial z}\right) + f_\theta, \tag{2.56b}$$

$$\frac{\partial u_z}{\partial t} + u_r\frac{\partial u_z}{\partial r} + \frac{u_\theta}{r}\frac{\partial u_z}{\partial \theta} + u_z\frac{\partial u_z}{\partial z} = -\frac{\partial \tilde{p}}{\partial z} + \left(\frac{1}{r}\frac{\partial (r\tau_{rz})}{\partial r} + \frac{1}{r}\frac{\partial \tau_{\theta z}}{\partial \theta} + \frac{\partial \tau_{zz}}{\partial z}\right) + f_z, \tag{2.56c}$$

$$\frac{1}{r}\frac{\partial (ru_r)}{\partial r} + \frac{1}{r}\frac{\partial u_\theta}{\partial \theta} + \frac{\partial u_z}{\partial z} = 0 \tag{2.56d}$$

where τ is the viscous stress tensor defined as $\tau_{ij} = \nu(\dot{\gamma})\mathbf{D}_{ij}$ where $\mathbf{D}_{ij}$ are the components of two times the symmetric part of the velocity gradient tensor,

$$\begin{aligned}
\mathbf{D}_{rr} &= 2\frac{\partial u_r}{\partial r}, & \mathbf{D}_{r\theta} = \mathbf{D}_{\theta r} &= r\frac{\partial}{\partial r}\left(\frac{u_\theta}{r}\right) + \frac{1}{r}\frac{\partial u_r}{\partial \theta},\\
\mathbf{D}_{\theta\theta} &= 2\left(\frac{1}{r}\frac{\partial u_\theta}{\partial \theta} + \frac{u_r}{r}\right), & \mathbf{D}_{\theta z} = \mathbf{D}_{z\theta} &= \frac{\partial u_\theta}{\partial z} + \frac{1}{r}\frac{\partial u_z}{\partial \theta},\\
\mathbf{D}_{zz} &= 2\frac{\partial u_z}{\partial z}, & \mathbf{D}_{rz} = \mathbf{D}_{zr} &= \frac{\partial u_r}{\partial z} + \frac{\partial u_z}{\partial r}.
\end{aligned} \tag{2.57}$$

It is known that the solution of a Poiseuille flow in cylindrical coordinates reads $\mathbf{u} = u_z(r)\mathbf{e}_z$. Neglecting body forces and assuming a steady-state flow, system (2.54) can be rewritten as

$$\begin{aligned}
0 &= -\frac{\partial \tilde{p}}{\partial r},\\
0 &= -\frac{1}{r}\frac{\partial \tilde{p}}{\partial \theta},\\
0 &= -\frac{\partial \tilde{p}}{\partial z} + \frac{1}{r}\frac{\partial}{\partial r}\left(\nu(\dot{\gamma})r\frac{\partial u_z}{\partial r}\right)
\end{aligned} \tag{2.58}$$

from which, by $\dot{\gamma} = \frac{\partial u_z}{\partial r}$, we derive the following relationships

$$\begin{aligned}
p &= p(z)\\
\frac{\partial p}{\partial z} &= \frac{1}{r}\frac{\partial}{\partial r}(\nu(\dot{\gamma})r\frac{\partial u_z}{\partial r})\\
\nu(\dot{\gamma}) &= \nu_0\dot{\gamma}^{n-1} = \nu_0\left|\frac{\partial u_z}{\partial r}\right|^{n-1}.
\end{aligned} \tag{2.59}$$

Let us consider system (2.59). From the first equation, we deduce that the pressure field is constant along cross flow sections, i.e. r, z. The second equation, since the right-hand-side depends only on r and the left-hand-side depends only on coordinate z, must be equal to a constant,

$$\frac{\partial p}{\partial z} = \frac{1}{r}\frac{\partial}{\partial r}(\nu(\dot{\gamma})r\frac{\partial u_z}{\partial r}) = A. \tag{2.60}$$

Then, employing the third equation in the right-hand-side of latter equation and integrating separately the differential problems, we obtain

$$\begin{aligned} \tilde{p} =& Az + B, \\ u_z =& -\left(\frac{|A|}{2\nu_0}\right)^{\frac{1}{n}} \frac{n}{n+1} r^{\frac{n+1}{n}} + C. \end{aligned} \tag{2.61}$$

Now employing boundary conditions into equation (2.61),

$$u_z(R) = 0,\ \tilde{p}(0) = -\Delta\tilde{p},\ \tilde{p}(L) = 0$$

where $\Delta\tilde{p} = \tilde{p}(L) - \tilde{p}(0)$, we obtain the expressions of constants A, B and C:

$$A = \frac{\Delta\tilde{p}}{L}, \quad B = -\Delta\tilde{p}, \quad C = \left(\frac{|\Delta\tilde{p}|}{2\nu_0 L}\right)^{\frac{1}{n}} \frac{n}{n+1} R^{\frac{n+1}{n}}. \tag{2.62}$$

Thus, the final expressions for velocity and pressure read

$$\begin{aligned} \tilde{p} =& \Delta\tilde{p}\left(\frac{z}{L} - 1\right), \\ u_z =& \frac{n}{n+1}\left(\frac{|\Delta\tilde{p}|}{2\nu_0 L}\right)^{\frac{1}{n}} (R^{\frac{n+1}{n}} - r^{\frac{n+1}{n}}). \end{aligned} \tag{2.63}$$

Equation (2.63) can be rewritten with respect to average velocity U_{avg}. Write

$$\begin{aligned} U_{\text{avg}} =& \frac{1}{\pi R^2}\int_0^R \int_0^{2\pi} u_z r \,\mathrm{d}\theta \,\mathrm{d}r \\ =& \left(\frac{|\Delta\tilde{p}|}{2\nu_0 L}\right)^{\frac{1}{n}} \frac{n}{3n+1} R^{\frac{n+1}{n}}, \end{aligned} \tag{2.64}$$

then

$$u_z(r) = U_{\text{avg}} \frac{3n+1}{n+1}\left(1 - \left(\frac{r}{R}\right)^{\frac{n+1}{n}}\right). \tag{2.65}$$

Slip boundary conditions If slip boundary conditions are imposed on velocity at wall the solution changes. Slip conditions are defined as

$$u_z(R) = f(\tau_t)$$

where τ_t is the tangential component of viscous stress tensor at the wall,

$$\tau_t = \boldsymbol{\sigma}\mathbf{n} - (\boldsymbol{\sigma}\mathbf{n}\cdot\mathbf{n})\mathbf{n}, \tag{2.66}$$

and f is some function. Since on the walls it holds

$$\boldsymbol{\sigma} = -p\mathrm{I} + \mu(\dot{\gamma})\frac{\partial u_z}{\partial r}\Big|_R(\mathbf{e}_r\otimes\mathbf{e}_z + \mathbf{e}_z\otimes\mathbf{e}_r), \tag{2.67}$$

if we set $\mathbf{n} = \mathbf{e}_r$, then

$$\tau_t = \mu(\dot{\gamma})\frac{\partial u_z}{\partial r}\Big|_R\mathbf{e}_z. \tag{2.68}$$

Now consider equation (2.61). Employing slip boundary conditions we have to recompute the value of C: let

$$u_z(R) = f(\tau_t) = f\left(\mu(\dot{\gamma})\frac{\partial u_z}{\partial r}\Big|_R\mathbf{e}_z\right),$$

then

$$u_z = \left(\frac{|\Delta\tilde{p}|}{2\nu_0 L}\right)^{\frac{1}{n}}\frac{n}{n+1}(R^{\frac{n+1}{n}} - r^{\frac{n+1}{n}}) + f\left(-\mu(\dot{\gamma})\left(\frac{|\Delta\tilde{p}|}{2L\nu_0}\right)^{\frac{1}{n}}R^{\frac{1}{n}}\mathbf{e}_z\right). \tag{2.69}$$

There exist many function that model the behaviour of a fluid flowing with slip conditions at wall. A model that is often employed is the one describe din [62]: the Navier slip condition,

$$f(\tau_t) = K_L\left\|\tau_t\right\|, \tag{2.70}$$

or its nonlinear version,

$$f(\tau_t) = K_{NL}\left\|\tau_t\right\|^m, \tag{2.71}$$

where the parameters K_L and K_{NL} are the linear and nonlinear slip coefficients, respectively, and m is the nonlinear exponent.

Other popular slip boundary conditions are the Hatzikiriakos and the asymptotic boundary conditions [63]:

$$f(\tau_t) = K_{H1}\sinh(K_{H2}\left\|\tau_t\right\|), \qquad f(\tau_t) = K_{A1}\log(1+K_{A2}\left\|\tau_t\right\|). \tag{2.72}$$

Let us computer the velocity average using the Navier slip conditions (2.70).

$$\begin{aligned} U_{\text{avg}} =& \frac{2}{R^2}\int_0^R u_z r\,\mathrm{d}r \\ =& \frac{2}{R^2}\int_0^R\left(\frac{|\Delta\tilde{p}|}{2\nu_0 L}\right)^{\frac{1}{n}}\frac{n}{n+1}(R^{\frac{n+1}{n}} - r^{\frac{n+1}{n}})r\,\mathrm{d}r + \frac{1}{2}K_L\frac{|\Delta P|R}{L} \\ =& \left(\frac{|\Delta\tilde{p}|}{2\nu_0 L}\right)^{\frac{1}{n}}\frac{n}{3n+1}R^{\frac{n+1}{n}} + \frac{1}{2}K_L\frac{|\Delta P|R}{L}. \end{aligned} \tag{2.73}$$

.

2.3.3 Rabinowitz analysis for the capillary rheometer

The idea behind Rabinowitz analysis is to compare the dynamic and kinematic quantities measured by the rheometer in order to reduce their relationship to something similar to equation (2.39), where we consider the viscosity as proportionality factor between shear rate and stress.

A capillary rheometer, in particular, measures a pressure drop and the corresponding to an imposed flow rate, the kinematic quantity. To determine rheological parameters we employ the Rabinowitz analysis [102] the wall shear rates, relative to the magnitude of pressure drop, that corresponds to the flow rate that is imposed.

Let $\mathbf{e}_r, \mathbf{e}_\theta, \mathbf{e}_z$ be the unit vectors representing a cylindrical reference frame where $\mathbf{e}_z$ is aligned with the axis of the capillary die. Let C_V be a control volume of cylindrical shape, aligned with the axis of the capillary die, with radius $r < R$ and length L, with $C_S = \partial C_V$ (see Figure 2.2).

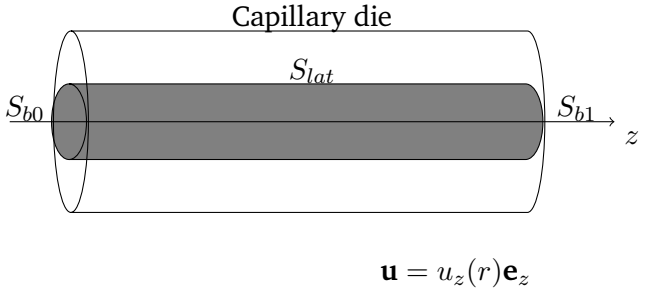


Figure 2.2: Cylindrical control volume inside capillary die.

Integrate then the momentum equation on C_V:

$$\int_{C_V} \rho \mathbf{f} \, \mathrm{d}V + \int_{C_V} \nabla \cdot (-p\mathrm{I} + 2\mu(\dot{\gamma})\mathbf{D}) \, \mathrm{d}V = \int_{C_V} \rho \frac{\partial \mathbf{u}}{\partial t} \, \mathrm{d}V + \int_{C_V} \rho(\mathbf{u} \cdot \nabla)\mathbf{u} \, \mathrm{d}V, \tag{2.74}$$

By Section 2.3.2, we know that

$$\dot{\gamma} = \left| \frac{\partial u_z}{\partial r} \right|,$$

then, by neglecting body forces and by assuming to be steady-state, equation (2.74) becomes

$$\int_{C_V} \nabla \cdot \left(-p\mathrm{I} + 2\rho\nu_0 \left| \frac{\partial u_z}{\partial r} \right| \mathbf{D} \right) \, \mathrm{d}V = 0. \tag{2.75}$$

Considering equation (2.65) and applying divergence theorem, the momentum equation reduces to

$$\int_{C_S} p\mathbf{n} dS = \int_{C_S} \mu(\dot{\gamma}) \nabla \cdot \mathbf{u} \, \mathbf{n} dS. \tag{2.76}$$

Projection the previous equation on $\mathbf{e}_z$ leads to

$$\int_{Sb1} p dS - \int_{Sb0} p dS = \int_{S_{lat}} \mu(\dot{\gamma}) \frac{\partial u_z}{\partial r} dS, \tag{2.77}$$

where S_{b0} and S_{b1} are the inflow and the outflow of C_V, respectively, and S_{lat} is the cylindrical surface of C_V (c.f. Figure 2.2).

Notice now that $\frac{\partial u_z}{\partial r}$ is constant on S_{lat}, and so $\dot{\gamma}$ and $\mu(\dot{\gamma})$. Assuming that p is constant on faces S_{b0} and S_{b1}, equation (2.77) can be simplified as

$$p_1 \pi r^2 - p_0 \pi r^2 = 2\pi r L \tau|_{S_{lat}} \tag{2.78}$$

where p_1 and p_0 are pressures on faces S_{b1} and S_{b0}, respectively. Then, τ reads

$$\tau = \frac{p_1 - p_0}{2L} r = -\frac{|\Delta p|}{2L} r. \tag{2.79}$$

Evaluating τ in $r = R$ we obtain the wall shear stress

$$\tau_w = \tau|_{r=R} = \frac{p_1 - p_0}{2L} R = \frac{\Delta p}{2L} R. \tag{2.80}$$

At the moment, what we have found is a relationship between the pressure drop and the stress tensor. Now we look for a relationship between flow rate and shear rate. Let Q be the flux across the section of the capillary die:

$$\begin{aligned} Q &= \int_0^R \int_0^{2\pi} u_z r \,\mathrm{d}\theta \,\mathrm{d}r = 2\pi \int_0^R u_z r \,\mathrm{d}r = 2\pi \left[\frac{1}{2} r^2 u_z\right]_0^R - 2\pi \int_0^R \frac{1}{2} r^2 \frac{\partial u_z}{\partial r} \,\mathrm{d}r \\ &= -\pi \int_0^R r^2 \frac{\partial u_z}{\partial r} \,\mathrm{d}r = \pi \int_0^R r^2 \dot{\gamma} \,\mathrm{d}r. \end{aligned} \tag{2.81}$$

We now employ equation (2.80) and the change of variables

$$\begin{aligned} r = -2\frac{L\tau}{|\Delta P|}, \qquad & \mathrm{d}r = -2\frac{L}{|\Delta P|} \,\mathrm{d}\tau, \\ r = R \Leftrightarrow \tau = \tau_w, \qquad & r = 0 \Leftrightarrow \tau = 0, \end{aligned} \tag{2.82}$$

in equation (2.81):

$$Q = \pi \int_0^{\tau_w} \left(-2\frac{L}{|\Delta P|}\right)^3 \tau^2 \dot{\gamma} \,\mathrm{d}\tau = \frac{\pi R^3}{\tau_w^3} \int_0^{\tau_w} \tau^2 \dot{\gamma} \,\mathrm{d}\tau. \tag{2.83}$$

Let the *apparent shear rate* be defined as

$$\dot{\gamma}_{app} = \frac{4Q}{\pi R^3},$$

then we have that

$$\dot{\gamma}_{app} = \frac{4Q}{\pi R^3} = 4\frac{1}{\tau_w^3} \int_0^{\tau_w} \tau^2 \dot{\gamma} \,\mathrm{d}\tau$$

and consequently

$$\tau_w^3 \dot{\gamma}_{app} = 4 \int_0^{\tau_w} \tau^2 \dot{\gamma} \,\mathrm{d}\tau. \tag{2.84}$$

Deriving now with respect to τ_w and dividing then by τ_w, the following equation is obtained:

$$\frac{1}{4}\frac{\partial\dot{\gamma}_{app}}{\partial\tau_w}\tau_w + \frac{3}{4}\dot{\gamma}_{app} = \dot{\gamma}_w \tag{2.85}$$

where $\dot{\gamma}_w$ is the shear rate at the wall.

Writing

$$\tau_w = \frac{\partial\tau_w}{\partial\log(\tau_w)} \qquad \dot{\gamma}_{app} = \frac{\partial\dot{\gamma}_{app}}{\partial\log(\dot{\gamma}_{app})},$$

we reformulate equation (2.85) as

$$\frac{1}{4}\frac{\partial\log(\dot{\gamma}_{app})}{\partial\log(\tau_w)}\dot{\gamma}_{app} + \frac{3}{4}\dot{\gamma}_{app} = \dot{\gamma}_w. \tag{2.86}$$

Finally, defining

$$\frac{1}{n} = \frac{\partial\log(\dot{\gamma}_{app})}{\partial\log(\tau_w)},$$

equation (2.86) becomes

$$\frac{1}{4n}\dot{\gamma}_{app} + \frac{3}{4}\dot{\gamma}_{app} = \dot{\gamma}_w \quad \Rightarrow \quad \dot{\gamma_w} = \frac{3n+1}{4n}\dot{\gamma}_{app}. \tag{2.87}$$

Computing $\dot{\gamma}_w$ through the latter equation and comparing it to the wall shear stress values computed from pressure drop of experimental acquisitions, the viscosity law of the fluid can be estimated interpolating the values thanks to relationship

$$\mu \simeq \frac{\tau_w}{\dot{\gamma}_w}. \tag{2.88}$$

2.3.4 Mooney analysis for cylindrical duct

The Mooney analysis is employed combined with the Rabinowitz one in order to take into account slip boundary conditions on the walls of the capillary duct.

Let $u_z(R) = v_{slip}$ be the slip velocity, unknown by now. By the symmetry of the cylindrical domain v_{slip} can be assumed to be independent of θ and z. We now recall equation (2.81), that under slip conditions becomes

$$Q = 2\pi\left[\frac{1}{2}r^2u_z\right]_0^R - 2\pi\int_0^R\frac{1}{2}r^2\frac{\partial u_z}{\partial r}\,\mathrm{d}r = \pi R^2 v_{slip} + \pi\int_0^R r^2\dot{\gamma}\,\mathrm{d}r. \tag{2.89}$$

Now, employing the change of variables (2.82), as for the Rabinowitz analysis, equation (2.89) becomes

$$Q = \pi R^2 v_{slip} + \frac{\pi R^3}{\tau_w^3}\int_0^{\tau_w}\tau^2\dot{\gamma}\,\mathrm{d}\tau \tag{2.90}$$

and then, rearranging terms,

$$\frac{4Q}{\pi R^3} = \frac{4v_{slip}}{R} + \frac{4}{R^3}\int_0^R r^2\dot{\gamma}dr. \tag{2.91}$$

In order to determine the value of v_{slip}, at least two rheometers with different duct lengths have to be employed. Let L_1, L_2, R_1, R_2 and Q_1, Q_2 be the lengths, the radii and the flow rates of the two rheometers, respectively. Then, it holds

$$\tau_{w1} = \tau_{w2} \quad \Longrightarrow \quad \frac{\Delta P_1}{2L_1} R_1 = \frac{\Delta P_2}{2L_2} R_2 = \tau_w \tag{2.92}$$

and consequently, considering equation (2.91),

$$\begin{aligned} \frac{4Q_1}{\pi R_1^3} =& \frac{4v_{slip}}{R_1} + \frac{4}{R_1^3} \int_0^{R_1} r_1^2 \left| \frac{\partial u_{z1}}{\partial r_1} \right| dr_1, \\ \frac{4Q_2}{\pi R_2^3} =& \frac{4v_{slip}}{R_2} + \frac{4}{R_2^3} \int_0^{R_2} r_2^2 \left| \frac{\partial u_{z2}}{\partial r_2} \right| dr_2, \end{aligned} \tag{2.93}$$

assuming v_{slip} depending only on $\tau_w = \tau_{w1} = \tau_{w2}$.

Since

$$\frac{\Delta P_1}{2L_1} R_1 = \frac{\Delta P_2}{2L_2} R_2 \tag{2.94}$$

it follows that

$$\frac{4}{R_1^3} \int_0^{R_1} r_1^2 \left| \frac{\partial u_{z1}}{\partial r_1} \right| \mathrm{d}r_1 = \frac{4}{R_2^3} \int_0^{R_2} r_2^2 \left| \frac{\partial u_{z2}}{\partial r_2} \right| \mathrm{d}r_2. \tag{2.95}$$

Considering equation (2.93), defining

$$\dot{\gamma}_{app1} = \frac{4Q_1}{\pi R_1^3}, \qquad \dot{\gamma}_{app2} = \frac{4Q_2}{\pi R_2^3},$$

the slippage velocity v_{slip} can be computed as the slope of the line connecting points $\left(\frac{4}{R_1}, \dot{\gamma}_{app1}\right)$ and $\left(\frac{4}{R_2}, \dot{\gamma}_{app2}\right)$.

Finally, the corrected apparent shear rate reads

$$\dot{\gamma}_{app,corr} = \frac{4Q_{corr}}{\pi R^3} = \frac{4(Q - \pi R^2 v_{slip})}{\pi R^3}. \tag{2.96}$$

Once the corrected apparent shear rate (2.96)), we can perform Rabinowitz analysis by substituting $\dot{\gamma}_{app}$ with $\dot{\gamma}_{app,corr}$ in equation (2.85).

2.3.5 Final remarks

The modelling of viscosity dependence on the volume fraction of fillers in polymer matrices has been inserted in this work through the implementation of these models in the numerical toolbox developed to simulate polymer mixing processes. As pointed out in [41], the presence of fillers into polymer matrices has been proven to significantly modify its rheological properties. The former work emphasised how the modelling of the rheology of a polymer compound is impactful, both in laboratory and numerical experiments.

The viscosity models presented in this chapter have been used in [41] to compare the results of numerical simulations of simple experimental flows with experimental data for different filler fraction levels, finding very good agreement in cases with low filler fraction. These models will also be tested in Chapter 6 in more complex industrial applications.

Chapter 3

Numerical discretization by the Diffuse Interface Box method

In this chapter we introduce the Box method formulation and its relation with the classical Finite Volume method and then we introduce the Diffuse Interface methods to deal with non-conforming boundaries.

In Section 3.1 we prove the well-posedness of Box method formulation for Poisson and Stokes problems (c.f. Sections 3.1.2 and 3.1.3) and we performed, for both problems, a convergence analysis finding a priori estimates of the numerical error in H^1 and L^2 norms. Theoretical results are then verified by a numerical assessments on cases with analytic solutions.

In Section 3.2, we combine the to the Box method formulation with a non-conforming method to approximate immersed boundaries. Among the many non-conforming methods mentioned in Chapter 1, we focus on the Diffuse Interface method, developed in [124] for elliptic problems. We extend the former analysis to the Box method applied to a Poisson problem, obtaining a priori H^1 and L^2 error estimates (c.f. Section 3.2.1). Then, in Sections 3.2.2 and 3.2.3, with the help of the theoretical results of Section 3.1.3, we verify numerically the convergence rates for the Diffuse Interface Box method applied to the Stokes problem. Finally, in Section 3.2.4, we propose a roadmap to prove theoretically the convergence rates obtained in numerical experiments.

3.1 Box method: a variational finite volume method

In this section we introduce the Box method applied to elliptic problems. More precisely, we first analyse the Box method for the Poisson problem and for the Stokes problem, obtaining a priori error estimates depending on the discretization parameter h, dictating the accuracy of the approximation of the PDE. We first study the Box method applied to a Poisson equation finding a priori H^1 and L^2 estimates for the numerical solution. Then, we consider the Newtonian Stokes problem exploiting the Rhie-Chow stabilization, a well known practice to stabilize finite volume approximations (see, e.g., [119, 137]), and finding numerical estimates for velocity and pressure.

We will provide some numerical tests to validate the theoretical results. The numerical results have been obtained using the open-source library OpenFOAM.

3.1.1 Preliminaries

We now introduce the geometrical setting. Let $\Omega \subset \mathbb{R}^d$, $d = 2, 3$ be a polyhedral bounded domain and let $\Gamma = \partial\Omega$ be its boundary. Let $\mathcal{T}_h = \{T\}$ be a conforming and shape regular triangulation of Ω. We denote by h_T the diameter of $T \in \mathcal{T}_h$ and we set $h = \max_T h_T$.

Let $\mathcal{V} = H^1(\Omega)$, $\mathcal{V}_g = H^1_\Gamma = \{v \in \mathcal{V} : v = g \text{ on } \Gamma\}$ and $\mathcal{Q} = L^2(\Omega)$ where g is a sufficiently regular function. With a slight abuse of notation, we will denote by a bold symbol, e.g. $\boldsymbol{\mathcal{V}}$, the d-dimensional counterparts of those spaces. Moreover, from now on, the standard norms for H^1 and L^2 spaces have to be intended on the whole domain Ω where the domain of integration is not specified.

On $\mathcal{T}_h$ we define the space of linear finite elements:

$$\mathcal{V}_h = \left\{v_h \in C^0(\overline{\Omega}) : v_h|_T \in \mathbb{P}^1(t)\, \forall T \in \mathcal{T}_h\right\} \subset \mathcal{V},$$
$$\mathcal{V}_{h,g_h} = \{v_h \in \mathcal{V}_h : v_h = g_h \text{ on } \partial\Omega\}$$

where g_h is a suitable piecewise linear approximation of g on $\partial\Omega$.

Then, we define the "box mesh" $\mathcal{B}_h$ (or dual mesh) associated to $\mathcal{T}_h$. We introduce the set $\mathcal{P}_h = \{p_i\}$ of vertices of $\mathcal{T}_h$ with $\mathcal{P}_h = \mathcal{P}_h^\partial \cup \mathcal{P}_h^o$, the set $\mathcal{P}_h^o$ containing the interior vertices of $\mathcal{T}_h$. We denote by $\mathcal{P}_{p_i}$ the set of triangles sharing vertex p_i. Let then $\mathcal{B}_h = \{B_i\}_{p_i \in \mathcal{P}_h^o}$ be the set of boxes B_i. Each box is a polyhedron with a boundary consisting of straight lines connecting the circumcentres of each triangle $T \in P_{p_i}$ (see Figure 3.1) and outer unit normal vector $\mathbf{n}_i$.

The results that are presented here will hold under the following assumption. The primal mesh $\mathcal{T}_h$ is assumed to be a Delaunay triangulation, i.e. no vertex of the triangulation is inside the circumcircle of any triangle of $\mathcal{T}_h$. Under this assumption, the dual mesh, defined as above, will be a Voronoi-type dual mesh and so it will be "orthogonal", i.e. segments connecting two barycentres of adjacent boxes are aligned with the unit normal vector of the face between them. In this way, the Box method and the classical finite volume method will present similar features.

On $\mathcal{B}_h$ we introduce the space of piecewise constant functions.

$$\mathcal{W}_h = \left\{w_h \in L^2(\Omega) : w_h|_{B_i} \in \mathbb{P}^0(B_i)\, \forall B_i \in \mathcal{B}_h\right\},$$

where the relation between the trial and test spaces is defined using a lumping map: let $v_h \in \mathcal{V}_h$,

$$\Pi_h : \mathcal{V}_h \to \mathcal{W}_h : \quad v_h = \sum_{B_i \in \mathcal{B}_h} v_h(p_i)\varphi_i \mapsto \Pi_h v_h = \sum_{B_i \in \mathcal{B}_h} v_h(p_i)\chi_i, \tag{3.1}$$

where φ_i and χ_i are the piecewise linear shape functions and the characteristic functions of the boxes, respectively.

These maps have also the following properties [115, Equations (2.11) and (2.12)]:

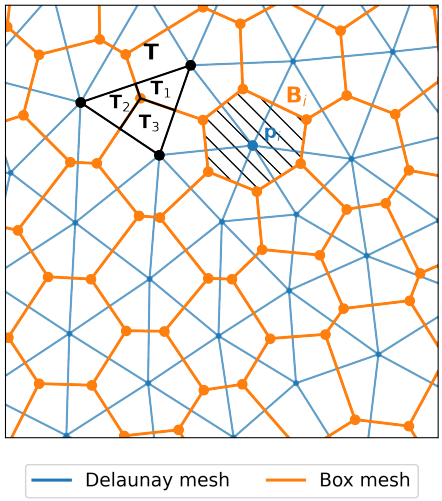


Figure 3.1: Example of a Delaunay triangulation and its Voronoi dual mesh.

Lemma 3.1.1. *Let $T \in \mathcal{T}_h$, $\forall v_h \in \mathcal{V}_h$,*

$$\begin{aligned} \int_T v_h - \Pi_h v_h \mathrm{d}x =& 0, \\ \|v_h - \Pi_h v_h\|_{L^2(T)} \leq& C h_T \, |v_h|_{H^1(T)}, \end{aligned}$$

In the next two sections we construct a convergence analysis of the BM Poisson and Stokes problems based on the above geometrical setting.

3.1.2 Poisson problem

We consider the following problem:

$$\begin{cases} -\Delta u = f & \text{in } \Omega, \\ u = g & \text{on } \Gamma = \partial\Omega \end{cases} \tag{3.2}$$

where $f \in L^2(\Omega)$ and $g \in H^{1/2}(\Gamma)$.

The Box method for the approximation of (3.2) is obtained by multiplying the equation by box-wise constant test functions and integrating by parts. The formulation reads as follows: find $u_B \in \mathcal{V}_{h,g_h}$ such that

$$a_B(u_B, w_h) = (f, w_h)_\Omega \quad \forall w_h \in \mathcal{W}_h, \tag{3.3}$$

where

$$a_B(u_B, w_h) = -\sum_{B_i \in \mathcal{B}_h} \int_{\partial B_i} \frac{\partial u_B}{\partial \mathbf{n}_i} w_h \mathrm{d}s, \tag{3.4}$$

being $\mathbf{n}_i$ the outer normal to B_i and $(\cdot,\cdot)_D$ is the usual L^2 scalar product on Ω.

Note that there holds (see [11, 73] for the two dimensional case and [115, 135] for the extension to any dimension)

$$\sum_{B_i \in \mathcal{B}_h} \int_{\partial B_i} \frac{\partial u_B}{\partial \mathbf{n}_i} \Pi_h v_h \mathrm{d}s = \int_\Omega \nabla u_B \cdot \nabla v_h \mathrm{d}x. \tag{3.5}$$

The relation (3.5) is crucial to show the following perturbation results (see [73, 135]):

$$\begin{aligned} \|\nabla(u_B - u_h)\|_{L^2(\Omega)} &\leq Ch \|f\|_{L^2(\Omega)}, \\ \|u_B - u_h\|_{L^2(\Omega)} &\leq Ch^2 \|f\|_{L^2(\Omega)}, \end{aligned} \tag{3.6}$$

where $u_h \in \mathcal{V}_{h,g_h}$ is the linear finite element approximation, obtained with the continuous Galerkin method, to the solution of problem (3.2). Equation (3.6) together with standard estimates for linear FEM and the triangular inequality yields a priori error estimates for the Box method.

Numerical assessment Consider Poisson problem (3.3) on $\Omega = [-1, 1]^2$ with $g = 0$, $u = \sin(\pi x)\sin(\pi y)$ and $f = -\Delta u$ is computed accordingly. Then, we solve the problem with Box method for various mesh sizes:

$$h = 0.1, 0.05, 0.025, 0.0125.$$

The convergence rates for the H^1 and L^2 errors are reported in Figure 3.2. As expected, the H^1 error converges with rate 1 and the L^2 error converges with rate 2.

3.1.3 Stokes problem

We consider the incompressible Newtonian Stokes problem:

$$\begin{aligned} -\nu\Delta\mathbf{u} + \nabla p &= \mathbf{f} \quad \text{in } \Omega, \\ \nabla \cdot \mathbf{u} &= 0 \quad \text{in } \Omega, \\ \mathbf{u} &= \mathbf{g} \quad \text{on } \Gamma = \partial\Omega \end{aligned} \tag{3.7}$$

where $\mathbf{f} \in [L^2(\Omega)]^d$ and $\mathbf{g} \in [H^{1/2}(\partial\Omega)]^d$.

Then, we define the following bilinear forms:

$$\begin{aligned} a : \boldsymbol{\mathcal{V}} \times \boldsymbol{\mathcal{V}} \to \mathbb{R} : \quad a(\mathbf{u}, \mathbf{v}) &= \int_\Omega \nu \nabla\mathbf{u} : \nabla\mathbf{v} \mathrm{d}x, \qquad \forall \mathbf{u}, \mathbf{v} \in \boldsymbol{\mathcal{V}}, \\ b : \boldsymbol{\mathcal{V}} \times \mathcal{Q} \to \mathbb{R} : \quad b(\mathbf{v}, p) &= -\int_\Omega \nabla \cdot \mathbf{v}\, p \mathrm{d}x, \qquad \forall \mathbf{v} \in \boldsymbol{\mathcal{V}}, \forall p \in \mathcal{Q}. \end{aligned} \tag{3.8}$$

The weak formulation of the problem reads: find $(\mathbf{u}, p) \in \boldsymbol{\mathcal{V}} \times \mathcal{Q}$, $\mathbf{u} = \mathbf{g}$ on Γ, such that

$$\begin{aligned} a(\mathbf{u}, \mathbf{v}) + b(\mathbf{v}, p) &= (\mathbf{f}, \mathbf{v})_\Omega, \quad \forall \mathbf{v} \in \boldsymbol{\mathcal{V}}_0, \\ b(\mathbf{u}, q) &= 0, \quad \forall q \in \mathcal{Q}, \end{aligned} \tag{3.9}$$

where $(\cdot, \cdot)_\Omega$ is the usual L^2 scalar product on Ω.

We also state the usual well-posedness result [9] for problem (3.9):

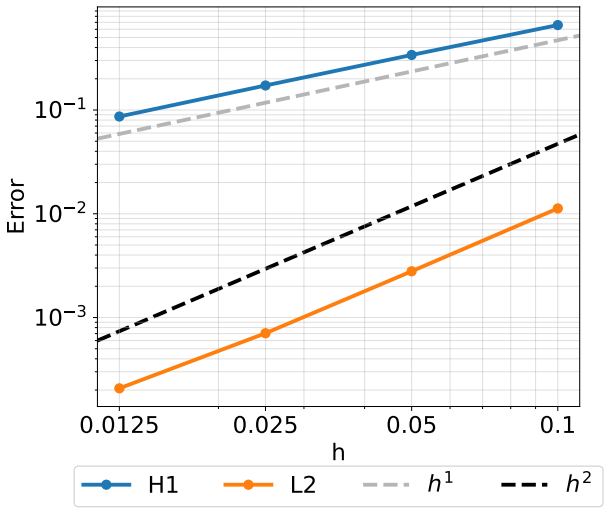


Figure 3.2: Convergence rates of errors obtained solving a Poisson problem employing the Box method.

Theorem 3.1.1 (Well-posedness)**.** *The saddle-point problem* (3.9) *is well-posed if*

1. *the bilinear form* a *is continuous and coercive;*
2. *the bilinear form* b *is continuous;*
3. *the inf-sup condition holds:* $\exists \beta > 0$ *s.t.*

$$\inf_{q \in \mathcal{Q}} \sup_{\mathbf{v} \in \boldsymbol{\mathcal{V}}} \frac{b(\mathbf{v}, q)}{\|\nabla \mathbf{v}\|_{L^2} \|q\|_{L^2}} \geq \beta > 0. \tag{3.10}$$

Moreover, the solution $(\mathbf{u}, p) \in \boldsymbol{\mathcal{V}} \times \mathcal{Q}$ *is unique and satisfies the following stability estimate:*

$$\|\nabla \mathbf{u}\|_{L^2(\Omega)} + \|p\|_{L^2(\Omega)} \leq C \left(\|\mathbf{f}\|_{L^2(\Omega)} + \|\mathbf{g}\|_{H^{1/2}(\partial\Omega)} \right). \tag{3.11}$$

In order to compare the continuous solution to the discrete one obtained using the Box method, c.f. Section 3.1.3.1, we need to introduce the stabilized $\mathbb{P}^1 - \mathbb{P}^1$ finite element discretization of problem (3.9), which read as follows: find $(\mathbf{u}_h, p_h) \in \boldsymbol{\mathcal{V}}_{h,\mathbf{g}_h} \times \mathcal{V}_h$ such that

$$\begin{aligned} a(\mathbf{u}_h, \mathbf{v}_h) + b(\mathbf{v}_h, p_h) &= (\mathbf{f}, \mathbf{v}_h)_\Omega, \qquad && \forall \mathbf{v}_h \in \boldsymbol{\mathcal{V}}_{h,0}, \\ -b(\mathbf{u}_h, q_h) + s(p_h, q_h) &= 0, && \forall q_h \in \mathcal{V}_h, \end{aligned} \tag{3.12}$$

where s is the stabilization term and

$$a : \boldsymbol{\mathcal{V}}_h \times \boldsymbol{\mathcal{V}}_h \to \mathbb{R} : \quad a(\mathbf{u}_h, \mathbf{v}_h) = \int_\Omega \nu \nabla \mathbf{u}_h : \nabla \mathbf{v}_h \mathrm{d}x,$$
$$b : \boldsymbol{\mathcal{V}}_h \times \mathcal{V}_h \to \mathbb{R} : \quad b(\mathbf{v}_h, q_h) = -\int_\Omega q_h \nabla \cdot \mathbf{v}_h \mathrm{d}x.$$

For what regards the operator s, we consider the Brezzi-Pitkaranta stabilization [27] term:

$$s(p_h, q_h) = -\delta \sum_{T \in \mathcal{T}_h} h_T^2 \int_T \nabla p_h \cdot \nabla q_h \mathrm{d}x.$$

The following convergence results hold [27]:

Theorem 3.1.2. *Let* $(\mathbf{u}, p) \in \boldsymbol{\mathcal{V}} \times \mathcal{Q}$ *be the solution of problem* (3.7) *and let* $(\mathbf{u}_h, p_h) \in \boldsymbol{\mathcal{V}}_h \times \mathcal{V}_h$ *be the solution of problem* (3.12)*. Then it holds*

$$\|\mathbf{u} - \mathbf{u}_h\|_{H^1(\Omega)} + \|p - p_h\|_{L^2(\Omega)} \leq \inf_{(\mathbf{v}_h, q_h) \in \boldsymbol{\mathcal{V}}_h \times \mathcal{V}_h} \left[\|\mathbf{v}_h - \mathbf{u}\|_{H^1} + \|q_h - p\|_{L^2} + \sup_{\substack{r_h \in \mathcal{V}_h \\ \|r_h\|_{L^2} = 1}} \sum_{T \in \mathcal{T}_h} h_T^2 \int_T \nabla q_h \cdot \nabla r_h \mathrm{d}x \right]. \tag{3.13}$$

Corollary 3.1.2.1. *Under hypothesis of Theorem 3.1.2 and suitable regularity properties of the solution* $(\mathbf{u}, p)$ *to problem* (3.7)*, it holds*

$$\|\mathbf{u} - \mathbf{u}_h\|_{H^1(\Omega)} + \|p - p_h\|_{L^2(\Omega)} \leq Ch \left(\|u\|_{H^2(\Omega)} + \|p\|_{H^1(\Omega)} \right).$$

3.1.3.1 Box method Stokes problem formulation

Employing piecewise constant functions on $\mathcal{B}_h$, the Box method formulation for problem (3.7) reads: find $(\mathbf{u}_B, p_B) \in \boldsymbol{\mathcal{V}}_{h,\mathbf{g}_h} \times \mathcal{V}_h$, such that

$$\begin{aligned} a_B(\mathbf{u}_B, \Pi_h \mathbf{v}_h) + b_B(\Pi_h \mathbf{v}_h, p_B) &= (\mathbf{f}, \Pi_h \mathbf{v}_h)_\Omega, && \forall \mathbf{v}_h \in \boldsymbol{\mathcal{V}}_h, \\ c_B(\mathbf{u}_B, \Pi_h q_h) + s_B(p_B, \Pi_h q_h) &= 0, && \forall q_h \in \mathcal{V}_h \end{aligned} \tag{3.14}$$

where

$$a_B : \boldsymbol{\mathcal{V}}_h \times \boldsymbol{\mathcal{W}}_h \to \mathbb{R} : \quad a_B(\mathbf{v}_h, \mathbf{w}_h) = -\sum_{B_i \in \mathcal{B}_h} \int_{\partial B_i} \nu \frac{\partial \mathbf{v}_h}{\partial \mathbf{n}_i} \cdot \mathbf{w}_h \mathrm{d}s,$$
$$b_B : \boldsymbol{\mathcal{W}}_h \times \mathcal{V}_h \to \mathbb{R} : \quad b_B(\mathbf{w}_h, q_h) = \sum_{B_i \in \mathcal{B}_h} \int_{\partial B_i} q_h \mathbf{n}_i \cdot \mathbf{w}_h \mathrm{d}s,$$
$$c_B : \boldsymbol{\mathcal{V}}_h \times \mathcal{W}_h \to \mathbb{R} : \quad c_B(\mathbf{v}_h, w_h) = \sum_{B_i \in \mathcal{B}_h} \int_{\partial B_i} \mathbf{n}_i \cdot \mathbf{v}_h w_h \mathrm{d}s,$$
$$s_B : \mathcal{V}_h \times \mathcal{W}_h \to \mathbb{R},$$

where s_B will be defined further in the text. In contrast to FEM formulation (3.12), here the bilinear forms b_B and c_B are different since the first operates on piecewise constant velocity and the second on piecewise linear velocity.

Notice that, by relationship (3.1), bilinear forms (3.1.3.1) satisfy the following lemma:

Lemma 3.1.2. *$\forall \mathbf{u}_h, \mathbf{v}_h \in \boldsymbol{\mathcal{V}}_h$, $p_h, q_h \in \mathcal{V}_h$, the following hold*

$$\begin{aligned} a_B(\mathbf{u}_h, \Pi_h \mathbf{v}_h) =& a(\mathbf{u}_h, \mathbf{v}_h), \\ b_B(\Pi_h \mathbf{v}_h, p_h) =& b(\mathbf{v}_h, p_h), \\ c_B(\mathbf{u}_h, \Pi_h q_h) =& -b(\mathbf{u}_h, q_h). \end{aligned} \tag{3.15}$$

Proof. By [115, Lemma 3.2], we just need to prove the last equality. We apply Lemma 3.1.1:

$$\begin{aligned} c_B(\mathbf{u}_h, \Pi_h q_h) &= \sum_{B \in \mathcal{B}_h} \Pi_h q_h \int_{\partial B} \mathbf{u}_h \cdot \mathbf{n}_b \mathrm{d}s \\ &= \sum_{B \in \mathcal{B}_h} \int_B \mathbf{u}_h \cdot \nabla \Pi_h q_h \mathrm{d}x + \sum_{B \in \mathcal{B}_h} \int_B \nabla \cdot \mathbf{u}_h \Pi_h q_h \mathrm{d}x \\ &= \sum_{T \in \mathcal{T}_h} \sum_{i=1}^{3} \int_{T_i} \nabla \cdot \mathbf{u}_h \Pi_h q_h \mathrm{d}x \\ &= \sum_{T \in \mathcal{T}_h} \sum_{i=1}^{3} \int_{T_i} \nabla \cdot \mathbf{u}_h \left(\Pi_h q_h - q_h\right) \mathrm{d}x + \sum_{T \in \mathcal{T}_h} \sum_{i=1}^{3} \int_{T_i} \nabla \cdot \mathbf{u}_h q_h \mathrm{d}x \\ &= \sum_{T \in \mathcal{T}_h} \int_T \nabla \cdot \mathbf{u}_h q_h \mathrm{d}x = -b(\mathbf{u}_h, q_h) \end{aligned} \tag{3.16}$$

where $T_i, i = 1, 2, 3$, are the partitions of the triangle T formed by the faces of intersecting boxes (see Figure 3.1). □

Proceeding we introduce some mesh quantities (Figure 3.3) that will be instrumental to introduce few discrete bilinear forms, (c.f. (3.17) below), that will be helpful for practical implementation on mesh quantities (Figure 3.3).

Each box B_i will be such that it has N_{B_i} faces. Considering a box B_i, we denote by B_j the box that shares face F_{ij} with B_i and $\mathcal{F}_h = \{F_{ij}\}$ be the set of faces. Let d_{ij} be the distance between the barycentres of boxes B_i and B_j, $\mathbf{n}_{ij}$ the unit normal vector directed outwards of B_i and w_{ij} be the interpolation weight (distance of B_j barycentre from face F_{ij} over d_{ij}, equal to $\frac{1}{2}$ when considering the Delaunay-Voronoi duality). Let D_{ij} be a "diamond", the polyhedron formed by the two box barycentres and by the face boundary; notice that $|D_{ij}| = |F_{ij}|\, d_{ij}/d$. Moreover, let $G_i = \{B_j : \exists B_i \cap B_j = F_{ij}$, for some $F_{ij} \in \mathcal{F}_h\}$ be the set of boxes that have a face in common with B_i. We also denote by $\phi_i = \phi|_{B_i}$ the restriction of a function $\phi_h \in \mathcal{V}_h$ evaluated on box B_i.

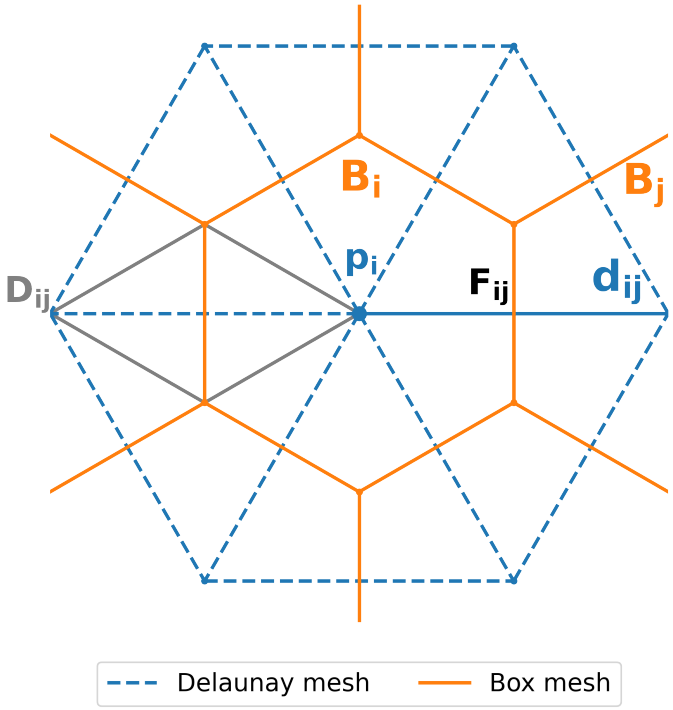


Figure 3.3: Scheme of dual mesh geometrical quantities.

Moreover, we introduce the following discrete bilinear forms:

$$a_B(\mathbf{v}_h, \mathbf{w}_h) = -\sum_{B_i \in \mathcal{B}_h} \sum_{B_j \in G_i} \nu \frac{|F_{ij}|}{d_{ij}} [\Pi_h \mathbf{v}_j - \Pi_h \mathbf{v}_i] \cdot \mathbf{w}_h, \tag{3.17a}$$

$$\tilde{b}_B(\mathbf{w}_h, q_h) = \sum_{B_i \in \mathcal{B}_h} \sum_{B_j \in G_i} |F_{ij}| \left[w_{ij} \Pi_h q_i + (1 - w_{ij}) \Pi_h q_j \right] \mathbf{n}_{ij} \cdot \mathbf{w}_h, \tag{3.17b}$$

$$\tilde{c}_B(\mathbf{v}_h, w_h) = \sum_{B_i \in \mathcal{B}_h} \sum_{B_j \in G_i} |F_{ij}| \left[w_{ij} \Pi_h \mathbf{v}_i + (1 - w_{ij}) \Pi_h \mathbf{v}_j \right] \cdot \mathbf{n}_{ij} w_h. \tag{3.17c}$$

Here, the Laplacian operator is discretized using a finite difference between barycentres of two adjacent boxes, while the gradient and the divergence are discretized using a linear weighted interpolation between the same two values using as weights the distances of the barycentres with respect to the face centre. By the fact that the discrete functions are piecewise linear on the primal mesh and given the orthogonality of the dual mesh, a_B is discretized exactly. On the other hand, the bilinear forms $\tilde{b}_B$ and $\tilde{c}_B$ are not exactly equal to b_B and c_B, respectively, and this will be taken into account in the analysis.

3.1.3.2 Stabilized problem

To find the expression of the stabilization functional s_B we have to consider the algebraic linear system. By the duality between FEM and BM, the problem matrices will share the same spectral properties, then the discrete *inf-sup* condition is not respected because we are not using *inf-sup* compatible elements.

First, let us define basis for the discrete spaces. Let N_p be the number of vertices p, $N_u = dN_p$ the number of d.o.f of velocity and $N_p = N_p$ the number of d.o.f. of pressure,

$$\begin{aligned}\boldsymbol{\mathcal{V}}_h : \{\boldsymbol{\phi}_j\}_{j=1}^{N_u} = \left\{ \begin{bmatrix}\phi_1\\0\\0\end{bmatrix}, \dots, \begin{bmatrix}\phi_{N_p}\\0\\0\end{bmatrix}, \begin{bmatrix}0\\\phi_{N_p+1}\\0\end{bmatrix}, \dots, \begin{bmatrix}0\\\phi_{2N_p}\\0\end{bmatrix}, \begin{bmatrix}0\\0\\\phi_{2N_p+1}\end{bmatrix}, \dots, \begin{bmatrix}0\\0\\\phi_{3N_p}\end{bmatrix} \right\},\\ \mathcal{V}_h : \{\phi_j\}_{j=1}^{N_p}\end{aligned} \tag{3.18}$$

For space $\boldsymbol{\mathcal{W}}_h, \mathcal{W}_h$ the basis are the projections on boxes of the ones of $\boldsymbol{\mathcal{V}}_h, \mathcal{V}_h$.

Then, we write $\mathbf{u}_B$ and p_B as linear combinations of basis functions:

$$\mathbf{u}_B(\mathbf{x}) = \sum_{j=1}^{N_u} u_j \boldsymbol{\phi}_j(\mathbf{x}), \qquad p_B(\mathbf{x}) = \sum_{j=1}^{N_p} p_j \phi_j(\mathbf{x}). \tag{3.19}$$

To find the expressions of the matrices of the problem, A, B and $\mathsf{B}^\intercal$, we plug the expressions for $\mathbf{u}_B$ and p_B into the bilinear forms setting, without loss of generality, $\mathbf{w}_h = \Pi_h \boldsymbol{\phi}_i = [\Pi_h \phi_i, 0, 0]^\intercal$, $w_h = \Pi_h \phi_i$:

$$\begin{aligned}(\mathsf{A}\mathbf{u})_i =& \sum_j \mathsf{A}_{ij}\mathbf{u}_j = a_B(\mathbf{u}_B, \mathbf{w}_h) = \sum_{j=1}^{N_u} u_j a_B(\boldsymbol{\phi}_j, \Pi_h \boldsymbol{\phi}_i) \qquad (3.20a)\\ =& -\sum_{j=1}^{N_u} u_j \sum_{k\in G_j} \nu \frac{|F_{jk}|}{d_{jk}} \left[\Pi_h \boldsymbol{\phi}_k - \Pi_h \boldsymbol{\phi}_j\right] \cdot \Pi_h \boldsymbol{\phi}_i\\ =& u_i \sum_{j\in G_i} \nu \frac{|F_{ij}|}{d_{ij}} - \sum_{j\in G_i} u_j \nu \frac{|F_{ij}|}{d_{ij}},\\ (\mathsf{B}^\intercal \mathbf{p})_i =& \sum_j \mathsf{B}^\intercal_{ij}\mathbf{p}_j = b_B(\mathbf{w}_h, p_B) = \sum_{j=1}^{N_p} p_j b_B(\Pi_h \boldsymbol{\phi}_i, \phi_j) \qquad (3.20b)\\ =& \sum_{j=1}^{N_p} p_j \sum_{k\in G_j} |F_{jk}| \left[w_{jk}\Pi_h\phi_j + (1-w_{jk})\Pi_h\phi_k\right] \mathbf{n}_{jk} \cdot \boldsymbol{\phi}_i\\ =& p_i \sum_{j\in G_i} |F_{ij}| \, w_{ij}\mathbf{n}_{ij} \cdot \mathbf{e}_i + \sum_{j\in G_i} p_j \, |F_{ij}| \, (1-w_{ij})\mathbf{n}_{ij} \cdot \mathbf{e}_i,\end{aligned}$$

$$
\begin{aligned}
(\mathrm{B}\mathbf{u})_i &= \sum_j \mathrm{B}_{ij}\mathbf{u}_j = c_B(\mathbf{u}_B, w_h) = \sum_{j=1}^{3N_p} u_j c_B(\boldsymbol{\phi}_j, \Pi_h\phi_i) \\
&= \sum_{j=1}^{3N_p} u_j \sum_{k\in G_j} |F_{jk}| \left[w_{jk}\Pi_h\boldsymbol{\phi}_j + (1-w_{jk})\Pi_h\boldsymbol{\phi}_k\right]\cdot\mathbf{n}_{jk}\Pi_h\phi_i \\
&= \sum_{\substack{i=1+(l-1)N_p,\\ l=1,\dots,d}}^{lN_p} u_i \sum_{j\in G_i} |F_{ij}|\, w_{ij}\mathbf{e}_l\cdot\mathbf{n}_{ij} + \sum_{j\in G_i} u_j\,|F_{ij}|\,(1-w_{ij})\mathbf{e}_l\cdot\mathbf{n}_{ij}.
\end{aligned}
\tag{3.20c}
$$

Now, using equations (3.20), let us write the algebraic linear system of equations (3.14), considering $s_B = 0$:

$$
\begin{bmatrix} \mathrm{A} & \mathrm{B}^\intercal \\ \mathrm{B} & 0 \end{bmatrix}
\begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix}
=
\begin{bmatrix} \mathbf{F} \\ \mathbf{0} \end{bmatrix}
\tag{3.21}
$$

where $\mathbf{F}$ is the discretization of the right-hand-side.

Let us now write A = D-H, where D is the diagonal of A and -H is the off-diagonal part. Inverting the system with respect to its diagonal we obtain the following expression for $\mathbf{u}$:

$$
\mathbf{u} = \mathrm{D}^{-1}\left[\mathrm{H}\mathbf{u} - \mathrm{B}^\intercal\mathbf{p} + \mathbf{F}\right] = \tilde{\mathbf{u}} - \mathrm{D}^{-1}\mathrm{B}^\intercal\mathbf{p},
$$

where $\tilde{\mathbf{u}} = \mathrm{D}^{-1}\left[\mathrm{H}\mathbf{u} + \mathbf{F}\right]$, and we substitute it in the second equation:

$$
\mathrm{B}\tilde{\mathbf{u}} - \mathrm{B}\mathrm{D}^{-1}\mathrm{B}^\intercal\mathbf{p} = \mathbf{0}.
$$

Roughly speaking, the term $\mathrm{B}\mathrm{D}^{-1}\mathrm{B}^\intercal\mathbf{p}$ is the algebraic counterpart of a Laplacian problem for the pressure where the diffusivity coefficient is D^{-1}, piecewise constant on boxes. This discretization procedure is know to generate spurious pressure modes [65] because the computational stencil of pressure Laplacian becomes too large. To stabilize the problem we employ the so called *Rhie-Chow interpolation* [65, 106, 137], that basically substitutes the term $\mathrm{B}\mathrm{D}^{-1}\mathrm{B}^\intercal\mathbf{p}$ with the discretization of a pressure Laplacian, as it is done for matrix A, in the scalar context, using D^{-1} as a viscosity. Following these considerations, we can define the Rhie-Chow stabilization operator.

Definition 3.1.1 (Rhie-Chow stabilization)**.** *The stabilization operator that represents the Rhie-Chow interpolation has the form:*

$$
s_B : H^1(\Omega)\times\mathcal{W}_h \to \mathbb{R} : \quad s_B(q, z_h) = s^1(q, z_h) - s^2(q, z_h),
$$

where s^1 and s^2 can be written explicitly as

$$
\begin{aligned}
s^1(q, z_h) &= \sum_{F_{ij}\in\mathcal{F}_h} \int_{F_{ij}} \left(w_{ij} D_i^{-1} \int_{B_i} \nabla q \,\mathrm{d}x + (1-w_{ij}) D_j^{-1} \int_{B_j} \nabla q \,\mathrm{d}x \right)\cdot\mathbf{n}_{ij}\,[\![z_h]\!]_{ij}\,\mathrm{d}s, \\
s^2(q, z_h) &= \sum_{F_{ij}\in\mathcal{F}_h} \int_{F_{ij}} \left(w_{ij}\,|B_i|\,D_i^{-1} + (1-w_{ij})\,|B_j|\,D_j^{-1} \right) \frac{\partial q}{\partial \mathbf{n}_{ij}}\,[\![z_h]\!]_{ij}\,\mathrm{d}s,
\end{aligned}
\tag{3.22}
$$

$\forall q \in H^1(\Omega)$, $z_h \in \mathcal{Z}_h$, where $|B_i|$ is the measure of box B_i.

Remark 3.1. s^1 and s^2 have a precise meaning, in particular, s^1 compensates $\mathrm{B}\mathrm{D}^{-1}\mathrm{B}^\intercal\mathbf{p}$ and s^2 is the scalar counterpart of bilinear form a_B.

The discrete form of Rhie-Chow stabilization reads.

$$\begin{aligned} s^1(p_B, z_h) = \sum_{B_i\in\mathcal{B}_h}\sum_{B_j\in G_i} |F_{ij}| \Bigg[& w_{ij}\mathrm{D}_i^{-1} \sum_{k\in G_i} |F_{ik}| \left(w_{ik}\Pi_h p_i + (1-w_{ik})\Pi_h p_k\right)\mathbf{n}_{ik} \\ & + (1-w_{ij})\mathrm{D}_j^{-1} \sum_{k\in G_j} |F_{jk}| \left(w_{jk}\Pi_h p_j + (1-w_{jk})\Pi_h p_k\right)\mathbf{n}_{jk} \Bigg] \cdot \mathbf{n}_{ij} z_h, \end{aligned} \tag{3.23}$$

$$s^2(p_B, z_h) = \sum_{B_i\in\mathcal{B}_h}\sum_{B_j\in G_i} \left[|B_i|\, w_{ij}\mathrm{D}_i^{-1} + |B_j|\,(1-w_{ij})\mathrm{D}_j^{-1} \right] |F_{ij}| \, \frac{\Pi_h p_j - \Pi_h p_i}{d_{ij}} z_h.$$

Finally, we can rewrite problem (3.14) in compact form: $(\mathbf{u}_B, p_B) \in \boldsymbol{\mathcal{V}}_h \times \mathcal{V}_h$ is the solution of the following discrete problem:

$$\begin{aligned} \widetilde{\mathcal{C}}_B((\mathbf{u}_B, p_B), (\Pi_h\mathbf{v}_h, \Pi_h q_h)) =& a_B(\mathbf{u}_B, \Pi_h\mathbf{v}_h) + \tilde{b}_B(\Pi_h\mathbf{v}_h, p_B) + \tilde{c}_B(\mathbf{u}_B, \Pi_h q_h) + s_B(p_B, \Pi_h q_h) \\ =& (\mathbf{f}, \Pi_h\mathbf{v}_h) \end{aligned} \tag{3.24}$$

$\forall (\mathbf{v}_h, q_h) \in \boldsymbol{\mathcal{V}}_h \times \mathcal{V}_h$.

We also write the continuous counterpart of problem (3.24) $\mathcal{C}_B$. It is basically operator $\widetilde{\mathcal{C}}_B$ using continuous fluxes instead of their numerical approximations (c.f. equations (3.20)):

$$\begin{aligned} \mathcal{C}_B((\mathbf{u}_B, p_B), (\Pi_h\mathbf{v}_h, \Pi_h q_h)) =& a_B(\mathbf{u}_B, \Pi_h\mathbf{v}_h) + b_B(\Pi_h\mathbf{v}_h, p_B) + c_B(\mathbf{u}_B, \Pi_h q_h) + s_B(p_B, \Pi_h q_h) \\ =& (\mathbf{f}, \Pi_h\mathbf{v}_h) \end{aligned} \tag{3.25}$$

$\forall (\mathbf{v}_h, q_h) \in \boldsymbol{\mathcal{V}}_h \times \mathcal{V}_h$.

3.1.3.3 Well-posedness

Before proving the well-posedness of the discrete problem, we recall the following results, from [3, 20, 25, 26, 34, 126].

Lemma 3.1.3 (Poincarè inequality)**.** *Let $S \subset \mathbb{R}^d$ be a bounded convex domain and let $\phi \in H^1(S)$, then*

$$\left\| \phi - \frac{1}{|S|}\int_S \phi \mathrm{d}x \right\|_{L^2(S)} \leq C_d \mathit{diam}(S) \left\|\nabla\phi\right\|_{L^2(G)}.$$

Lemma 3.1.4 (Inverse trace inequality)**.** *Let T be a polyhedron and F be one of its faces and let $\phi_h \in \mathbb{P}^p(T)$, $\phi \in H^1(T)$, then*

$$\begin{aligned} \left\|\phi_h\right\|_{L^2(F)} \leq & C_{\mathit{inv}} \sqrt{\frac{|F|}{|T|}} \left\|\phi_h\right\|_{L^2(T)}, \\ \left\|\phi\right\|_{L^2(F)} \leq & C_{\mathit{inv}} \left(h_T^{-1} \left\|\phi\right\|_{L^2(T)} + h_T \left|\phi\right|_{H^1(T)} \right). \end{aligned}$$

Consider also the following property:

Proposition 3.1.1. *Let $\phi \in H^1(\Omega)$ and $q_h \in \mathcal{V}_h$, then there holds:*

$$\sum_{B_i \in \mathcal{B}_h} \sum_{j \in G_i} \int_{F_{ij}} \phi \mathbf{n}_{ij} \Pi_h q_i \mathrm{d}s = \sum_{F_{ij} \in \mathcal{F}_h} \int_{F_{ij}} \phi \mathbf{n}_{ij} \left[\!\left[\Pi_h q_h \right]\!\right]_{ij} \mathrm{d}s, \tag{3.26}$$

where $\left[\!\left[\Pi_h q_h \right]\!\right]_{ij} = \Pi_h q_i - \Pi_h q_j$.

Proof. For any face we have two contributions from two different values of the basis functions. Passing from box summation to face summation we obtain the proof:

$$\begin{aligned} \sum_{B_i \in \mathcal{B}_h} \sum_{j \in G_i} \int_{F_{ij}} \phi \mathbf{n}_{ij} \Pi_h q_i \mathrm{d}s &= \sum_{F_{ij} \in \mathcal{F}_h} \int_{F_{ij}} \phi \mathbf{n}_{ij} \Pi_h q_i + \phi \mathbf{n}_{ji} \Pi_h q_j \mathrm{d}s \\ &= \sum_{F_{ij} \in \mathcal{F}_h} \int_{F_{ij}} \phi \mathbf{n}_{ij} \Pi_h q_i - \phi \mathbf{n}_{ij} \Pi_h q_j \mathrm{d}s \\ &= \sum_{F_{ij} \in \mathcal{F}_h} \int_{F_{ij}} \phi \mathbf{n}_{ij} \left(\Pi_h q_i - \Pi_h q_j \right) \mathrm{d}s. \end{aligned} \tag{3.27}$$

□

We also define a discrete H^1-norm that uses the normal gradient to each face of the box mesh:

Proposition 3.1.2 ($*$-norm and its properties)**.** *The $*$-norm is defined in general as*

$$|q|_* = \left(\sum_{F_{ij} \in \mathcal{F}_h} d_{ij} \int_{F_{ij}} \left| \frac{\partial q}{\partial \mathbf{n}_{ij}} \right|^2 \mathrm{d}s \right)^{\frac{1}{2}},$$

where $q \in H^1(\Omega)$.

Moreover, the following properties hold: $\forall q_h \in \mathcal{V}_h$,

$$\begin{aligned} \left\| \Pi_h q_h \right\|_{L^2} &\leq C \left| q_h \right|_*, \\ \left| q_h \right|_* &\leq C \left| q_h \right|_{H^1}, \\ \left| q_h \right|_{H^1} &\leq C \left| q_h \right|_*, \\ \left\| q_h \right\|_{L^2} &\leq \left| q_h \right|_*, \\ \left| q_h \right|_* &\leq 2 h_m^{-1} \left\| \Pi_h q_h \right\|_{L^2}. \end{aligned}$$

Proof. 1. By Lemma 3.1.1, and by [43, Lemma 5.1], that states that $\forall q_h \in \mathcal{V}_h$, $\exists C > 0$:

$$\left\| \Pi_h q_h \right\|_{L^2} \leq C \left| q_h \right|_*. \tag{3.28}$$

2. $\forall q_h \in \mathcal{V}_h$, $\exists C > 0$:

$$|q_h|_* = \left(\sum_{F_{ij} \in \mathcal{F}_h} d_{ij} \int_{F_{ij}} \left| \frac{\Pi_h q_i - \Pi_h q_j}{d_{ij}} \right|^2 \mathrm{d}s \right)^{\frac{1}{2}}$$
$$= \left(\sum_{F_{ij} \in \mathcal{F}_h} d \int_{D_{ij}} \left| \frac{\partial q_h}{\partial \mathbf{n}_{ij}} \right|^2 \mathrm{d}s \right)^{\frac{1}{2}} \leq \sqrt{d}\, |q_h|_{H^1}$$

where d is the dimension of the space $\mathbb{R}^d$ and where we used the fact that

$$|D_{ij}| = \left| \frac{F_{ij} d_{ij}}{d} \right|.$$

This holds because, for piecewise linear functions, the face-centred finite difference between box centres values coincides with the face normal gradient of the function itself.

3. $\forall q_h \in \mathcal{V}_h$, by equivalence (3.5) and by Proposition 3.1.1, it holds:

$$\begin{aligned} |q_h|_{H^1}^2 &= \int_\Omega \nabla q_h \cdot \nabla q_h \mathrm{d}x \\ &= \sum_{B_i \in \mathcal{B}_h} \sum_{j \in G_i} \int_{F_{ij}} \nabla q_h \cdot \mathbf{n}_{ij} \Pi_h q_i \mathrm{d}s \\ &= \sum_{F_{ij} \in \mathcal{F}_h} \int_{F_{ij}} \nabla q_h \cdot \mathbf{n}_{ij} \left[\!\left[\Pi_h q_h \right]\!\right]_{ij} \mathrm{d}s. \end{aligned} \tag{3.29}$$

Now, by the Cauchy-Schwarz inequality

$$\begin{aligned} |q_h|_{H^1}^2 &= \sum_{F_{ij} \in \mathcal{F}_h} \int_{F_{ij}} \nabla q_h \cdot \mathbf{n}_{ij} \left[\!\left[\Pi_h q_h \right]\!\right]_{ij} \mathrm{d}s \\ &\leq \sum_{F_{ij} \in \mathcal{F}_h} \left(\int_{F_{ij}} |\nabla q_h \cdot \mathbf{n}_{ij}|^2 \mathrm{d}s \right)^{\frac{1}{2}} \left(\int_{F_{ij}} \left[\!\left[\Pi_h q_h \right]\!\right]_{ij}^2 \mathrm{d}s \right)^{\frac{1}{2}} \\ &= \sum_{F_{ij} \in \mathcal{F}_h} d_{ij} \left(\int_{F_{ij}} |\nabla q_h \cdot \mathbf{n}_{ij}|^2 \mathrm{d}s \right)^{\frac{1}{2}} \left(\int_{F_{ij}} d_{ij} \left| \frac{\left[\!\left[\Pi_h q_h \right]\!\right]_{ij}}{d_{ij}} \right|^2 \mathrm{d}s \right)^{\frac{1}{2}} \\ &\leq \sum_{F_{ij} \in \mathcal{F}_h} \left(\int_{D_{ij}} |\nabla q_h|^2 \mathrm{d}s \right)^{\frac{1}{2}} \left(d_{ij} \int_{F_{ij}} \left| \frac{\left[\!\left[\Pi_h q_h \right]\!\right]_{ij}}{d_{ij}} \right|^2 \mathrm{d}s \right)^{\frac{1}{2}} \\ &\leq \left(\sum_{F_{ij} \in \mathcal{F}_h} \int_{D_{ij}} |\nabla q_h|^2 \mathrm{d}s \right)^{\frac{1}{2}} \left(\sum_{F_{ij} \in \mathcal{F}_h} d_{ij} \int_{F_{ij}} \left| \frac{\left[\!\left[\Pi_h q_h \right]\!\right]_{ij}}{d_{ij}} \right|^2 \mathrm{d}s \right)^{\frac{1}{2}} \\ &= |q_h|_{H^1} |q_h|_* \end{aligned} \tag{3.30}$$

where we used the fact that $\nabla q_h \cdot \mathbf{n}_{ij}$ is constant on diamond D_{ij} and, in the last passage, we employed the Hölder inequality. To conclude the proof divide by $|q_h|_{H^1}$ on both sides of equation (3.30).

4. By Lemma 3.1.3,

$$\begin{aligned}
\|q_h\|_{L^2}^2 &= \sum_{T\in\mathcal{T}_h} \|q_h\|_{L^2(T)}^2 \\
&\lesssim \sum_{T\in\mathcal{T}_h} \left\| q_h - \frac{1}{|T|}\int_T q_h \mathrm{d}x \right\|_{L^2(T)}^2 + \left\| \frac{1}{|T|}\int_T q_h \mathrm{d}x \right\|_{L^2(T)}^2 \\
&\lesssim \sum_{T\in\mathcal{T}_h} h_T^2 \left|q_h\right|_{H^1(T)}^2 + \|\Pi_h q_h\|_{L^2(T)}^2 \\
&\leq h^2 \left|q_h\right|_{H^1}^2 + \|\Pi_h q_h\|_{L^2}^2 \\
&\leq (1+h^2)\left|q_h\right|_*^2 .
\end{aligned} \tag{3.31}$$

5. By triangular inequality,

$$\begin{aligned}
|q_h|_* &= \left(\sum_{F_{ij}\in\mathcal{F}_h} d_{ij} \int_{F_{ij}} \left| \frac{\Pi_h q_i - \Pi_h q_j}{d_{ij}} \right|^2 \mathrm{d}s \right)^{\frac{1}{2}} \\
&\leq \left(\sum_{F_{ij}\in\mathcal{F}_h} d_{ij} \int_{F_{ij}} \left| \frac{\Pi_h q_i}{d_{ij}} \right|^2 + \left| \frac{\Pi_h q_j}{d_{ij}} \right|^2 \mathrm{d}s \right)^{\frac{1}{2}} \\
&\leq \frac{1}{d_{ij}} \left(\sum_{F_{ij}\in\mathcal{F}_h} d_{ij} \int_{F_{ij}} |\Pi_h q_i|^2 + |\Pi_h q_j|^2 \mathrm{d}s \right)^{\frac{1}{2}} \\
&\leq \frac{1}{\min\limits_{T\in\mathcal{T}_h} h_T} \left(\sum_{F_{ij}\in\mathcal{F}_h} d_{ij} \int_{F_{ij}} |\Pi_h q_i|^2 + |\Pi_h q_j|^2 \mathrm{d}s \right)^{\frac{1}{2}} \\
&\leq 2h_m^{-1} \|\Pi_h q_h\|_{L^2}
\end{aligned}$$

where $h_m = \min_T h_T$.

□

Remark 3.2. $|\cdot|_*$ is a norm if, in general, $q \in H_0^1$. In $\mathcal{V}_h$ it is equivalent to the H^1-seminorm.

We also assume the following regularity hypothesis on the geometrical quantities defined in previous section:

Assumption 3.1.1 (Mesh regularity)**.** *Let $\mathcal{B}_h$ be the Voronoi-type dual mesh of a Delaunay triangulation. Moreover, $\mathcal{B}_h$ satisfies the following assumptions. Let $h_m = \min_{T\in\mathcal{T}_h} h_T$, then it is such that*

$$\exists \delta > 0 : h_m = \delta h.$$

Moreover, we assume that mesh size does not change too much between neighbouring boxes. Hence $\forall B_i \in \mathcal{B}_h$ *and* $\forall F_{ij} \in \mathcal{F}_h$*, for some* $T : T \cap B_i \neq \emptyset$*,*

$$d_{ij} \simeq h_T, \quad |F_{ij}| \simeq h_T^{d-1}, \quad |B_i| \simeq h_T^d, \quad w_{ij} = \frac{1}{2}.$$

Moreover, $\exists C_1, C_2 > 0 : \; C_1 h \leq h_T \leq C_2 h, \forall T \in \mathcal{T}_h$*.*

Now, before going on with the convergence analysis, we define the following mesh dependent norm:

$$|||\mathbf{v}_h, q_h|||_{box} = \left(|\mathbf{v}_h|_{H^1}^2 + \|\Pi_h q_h\|_{L^2}^2 + |q_h|_{T,*}^2 \right)^{\frac{1}{2}}, \tag{3.32}$$

$\forall (\mathbf{v}_h, q_h) \in \boldsymbol{\mathcal{V}}_h \times \mathcal{V}_h$, where, for $q \in H^1(\Omega)$,

$$|q|_{T,*} = \left(\sum_{F_{ij} \in \mathcal{F}_h} h_T^3 d_{ij} \int_{F_{ij}} \left| \frac{\partial q}{\partial \mathbf{n}_{ij}} \right|^2 \mathrm{d}s \right)^{\frac{1}{2}}, \tag{3.33}$$

is the mesh dependent version of norm defined in Proposition 3.1.2.

Remark 3.3. The $T, *$-norm has been defined in order to conform the coercivity estimate of Lemma 3.1.8 and the convergence theorem 3.1.4.

Before proving the well-posedness of the problem, we need $\widetilde{C}_B$ and $\mathcal{C}_B$ to satisfy some properties.

As a first step, we have to prove the problem consistency. We state the following results on the consistency of linear interpolation fluxes and of the compact form $\widetilde{\mathcal{C}}_B$.

Lemma 3.1.5 ($\widetilde{\mathcal{C}}_B$ consistency)**.** *Let* $(\mathbf{u}, p) \in \boldsymbol{\mathcal{V}} \times (\mathcal{Q} \cap H^2_{loc}(\Omega))$ *be the solution to problem* (3.9) *and* $(\mathbf{u}_B, p_B) \in \boldsymbol{\mathcal{V}}_h \times \mathcal{V}_h$ *be the solution to problem* (3.14)*. Then there holds:*

$$\begin{aligned} \mathcal{C}_B((\mathbf{u}, p), (\Pi_h \mathbf{v}_h, \Pi_h q_h)) - & \widetilde{\mathcal{C}}_B((\mathbf{u}_B, p_B), (\Pi_h \mathbf{v}_h, \Pi_h q_h)) \\ & \lesssim \left(h^2 \|\mathbf{f}\|_{L^2} + h \, |p_B|_{H^1} + h \, |\nabla p|_{h,1} \right) |||\mathbf{v}_h, q_h|||_{box} \end{aligned}$$

$\forall \mathbf{v}_h \in \boldsymbol{\mathcal{V}}_h$, $q_h \in \mathcal{V}_h$ *and* $|\cdot|_{h,1}$ *is the* H^1 *broken seminorm.*

Proof. By Cauchy-Schwarz inequality and Lemmas 3.1.2, 3.1.1, 3.1.3, we can write:

$$
\begin{aligned}
&\mathcal{C}_B((\mathbf{u},p),(\Pi_h\mathbf{v}_h,\Pi_h q_h)) - \widetilde{\mathcal{C}}_B((\mathbf{u}_B,p_B),(\Pi_h\mathbf{v}_h,\Pi_h q_h)) \\
&\quad = a_B(\mathbf{u}-\mathbf{u}_B,\Pi_h\mathbf{v}_h) + s_B(p-p_B,\Pi_h q_h) \\
&\qquad + b_B(\Pi_h\mathbf{v}_h,p) \pm b_B(\Pi_h\mathbf{v}_h,p_B) - \tilde{b}_B(\Pi_h\mathbf{v}_h,p_B) \\
&\qquad + c_B(\mathbf{u},\Pi_h q_h) \pm c_B(\Pi_h\mathbf{v}_h,p_B) - \tilde{c}_B(p_B,\Pi_h q_h) \\
&\quad = \sum_{T\in\mathcal{T}_h} [(\mathbf{f},\mathbf{v}_h)_T - (\mathbf{f},\Pi_h\mathbf{v}_h)_T] \\
&\qquad + (b_B-\tilde{b}_B)(\Pi_h\mathbf{v}_h,p_B) + (c_B-\tilde{c}_B)(\mathbf{u}_B,\Pi_h q_h) + s_B(p,\Pi_h q_h) \\
&\quad = \sum_{T\in\mathcal{T}_h} (\mathbf{f},\mathbf{v}-\Pi_h\mathbf{v}_h)_T \\
&\qquad + (b_B-\tilde{b}_B)(\Pi_h\mathbf{v}_h,p_B) + (c_B-\tilde{c}_B)(\mathbf{u}_B,\Pi_h q_h) + s_B(p,\Pi_h q_h) \\
&\quad = \sum_{T\in\mathcal{T}_h} \left(\mathbf{f} - \frac{1}{|T|}\int_T \mathbf{f}\mathrm{d}x, \mathbf{v}-\Pi_h\mathbf{v}_h\right)_T \\
&\qquad + (b_B-\tilde{b}_B)(\Pi_h\mathbf{v}_h,p_B) + (c_B-\tilde{c}_B)(\mathbf{u}_B,\Pi_h q_h) + s_B(p,\Pi_h q_h) \\
&\quad \leq C_d h^2 \|\mathbf{f}\|_{L^2} |\mathbf{v}_h|_{H^1} + (b_B-\tilde{b}_B)(\Pi_h\mathbf{v}_h,p_B) + (c_B-\tilde{c}_B)(\mathbf{u}_B,\Pi_h q_h) + s_B(p,\Pi_h q_h).
\end{aligned}
\tag{3.34}
$$

We have now to estimate the consistency of the bilinear forms $\tilde{b}_B$ and $\tilde{c}_B$ and of the stabilization term. We first use Proposition 3.1.1 to sum the integrals over mesh faces. Consider the face barycentres $\mathbf{f}_{ij}$ and notice that, being p_B and $\mathbf{u}_B$ piecewise linear, the following hold:

$$
\frac{\Pi_h p_i + \Pi_h p_j}{2} = p_B(\mathbf{f}_{ij}), \quad \text{and} \quad \frac{\Pi_h\mathbf{u}_i + \Pi_h\mathbf{u}_j}{2} = \mathbf{u}_B(\mathbf{f}_{ij}). \tag{3.35}
$$

Let now $\mathbf{x}\in F_{ij}$ be a point in space. Then, being p_B and $\mathbf{u}_B$ piecewise linear, we can rewrite them using a Taylor expansion in $\mathbf{f}_{ij}$:

$$
\begin{aligned}
p_B &= p_B(\mathbf{f}_{ij}) + \sum_{T\in\mathcal{T}_h : T\cap D_{ij}\neq\emptyset} (\mathbf{x}-\mathbf{f}_{ij})\cdot\nabla p_B \mathbb{1}_{T\cap D_{ij}}, \\
\mathbf{u}_B &= \mathbf{u}_B(\mathbf{f}_{ij}) + \sum_{T\in\mathcal{T}_h : T\cap D_{ij}\neq\emptyset} (\mathbf{x}-\mathbf{f}_{ij})^\intercal\nabla\mathbf{u}_B \mathbb{1}_{T\cap D_{ij}},
\end{aligned}
\tag{3.36}
$$

where $\mathbb{1}$ is the indicator function and ∇p_B is piecewise constant on each intersection between triangle T and diamond D_{ij}.

Employing now equations (3.35) and (3.36), we obtain that

$$
\begin{aligned}
(b_B - \tilde{b}_B)(\Pi_h \mathbf{v}_h, p_B) &= \sum_{B_i \in \mathcal{B}_h} \sum_{j \in G_i} \int_{F_{ij}} \left[\frac{\Pi_h p_i + \Pi_h p_j}{2} - p_B \right] \Pi_h \mathbf{v}_h \cdot \mathbf{n}_{ij} \mathrm{d}s \\
&= \sum_{F_{ij} \in \mathcal{F}_h} \frac{d_{ij}}{d_{ij}} \int_{F_{ij}} \left((\mathbf{x} - \mathbf{f}_{ij}) \cdot \nabla p_B \right) \left([\![\Pi_h \mathbf{v}_h]\!]_{ij} \cdot \mathbf{n}_{ij} \right) \mathrm{d}s \\
&= \sum_{F_{ij} \in \mathcal{F}_h} d_{ij} \int_{F_{ij}} \left((\mathbf{x} - \mathbf{f}_{ij}) \cdot \nabla p_B \right) \left(\frac{\partial \mathbf{v}_h}{\partial \mathbf{n}_{ij}} \cdot \mathbf{n}_{ij} \right) \mathrm{d}s \\
&\leq \sum_{F_{ij} \in \mathcal{F}_h} \left(d_{ij} \int_{F_{ij}} |\mathbf{x} - \mathbf{f}_{ij}|^2 \, |\nabla p_B|^2 \, \mathrm{d}s \right)^{\frac{1}{2}} \left(d_{ij} \int_{F_{ij}} \left| \frac{\partial \mathbf{v}_h}{\partial \mathbf{n}_{ij}} \right|^2 |\mathbf{n}_{ij}|^2 \, \mathrm{d}s \right)^{\frac{1}{2}} \\
&\leq \sum_{F_{ij} \in \mathcal{F}_h} \left(d_{ij}^2 d_{ij} \int_{F_{ij}} |\nabla p_B|^2 \, \mathrm{d}s \right)^{\frac{1}{2}} \left(d_{ij} \int_{F_{ij}} \left| \frac{\partial \mathbf{v}_h}{\partial \mathbf{n}_{ij}} \right|^2 \mathrm{d}s \right)^{\frac{1}{2}} \\
&\leq \left(\sum_{F_{ij} \in \mathcal{F}_h} d_{ij}^2 d_{ij} \int_{F_{ij}} |\nabla p_B|^2 \, \mathrm{d}s \right)^{\frac{1}{2}} \left(\sum_{F_{ij} \in \mathcal{F}_h} d_{ij} \int_{F_{ij}} \left| \frac{\partial \mathbf{v}_h}{\partial \mathbf{n}_{ij}} \right|^2 \mathrm{d}s \right)^{\frac{1}{2}} \\
&= h \, |p_B|_{H^1} \, |\mathbf{v}_h|_* \, . \\
(c_B - \tilde{c}_B)(\mathbf{u}_B, \Pi_h q_h) &= \sum_{B_i \in \mathcal{B}_h} \sum_{j \in G_i} \int_{F_{ij}} \left[\frac{\Pi_h \mathbf{u}_i + \Pi_h \mathbf{u}_j}{2} - \mathbf{u}_B \right] \cdot \mathbf{n}_{ij} \Pi_h q_h \mathrm{d}s \\
&= \sum_{F_{ij} \in \mathcal{F}_h} \int_{F_{ij}} (\mathbf{x} - \mathbf{f}_{ij})^\intercal \nabla \mathbf{u}_B \mathbf{n}_{ij} \, [\![\Pi_h q_h]\!]_{ij} \, \mathrm{d}s \\
&= \sum_{F_{ij} \in \mathcal{F}_h} \int_{F_{ij}} (\mathbf{x} - \mathbf{f}_{ij}) \cdot \frac{\partial \mathbf{u}_B}{\partial \mathbf{n}_{ij}} \, [\![\Pi_h q_h]\!]_{ij} \, \mathrm{d}s \\
&= \sum_{F_{ij} \in \mathcal{F}_h} [\![\Pi_h q_h]\!]_{ij} \, \frac{\partial \mathbf{u}_B}{\partial \mathbf{n}_{ij}} \cdot \int_{F_{ij}} (\mathbf{x} - \mathbf{f}_{ij}) \mathrm{d}s = 0,
\end{aligned}
\tag{3.37}
$$

Then, for the stabilization term, using the assumption that $p \in H^2_{\text{loc}}(\Omega)$ and using Defi-

nition 3.1.1:

$$
\begin{aligned}
s_B(p,\Pi_h q_h) &= \sum_{F_{ij}\in\mathcal{F}_h}\int_{F_{ij}}\frac{1}{2}\left(\mathrm{D}_i^{-1}\int_{B_i}\nabla p\mathrm{d}x+\mathrm{D}_j^{-1}\int_{B_j}\nabla p\mathrm{d}x\right)\cdot\mathbf{n}_{ij}\left[\!\left[\Pi_h q_h\right]\!\right]_{ij}\mathrm{d}s\\
&\quad-\sum_{F_{ij}\in\mathcal{F}_h}\int_{F_{ij}}\frac{1}{2}\left(|B_i|\,\mathrm{D}_i^{-1}+|B_j|\,\mathrm{D}_j^{-1}\right)\frac{\partial p}{\partial\mathbf{n}_{ij}}\left[\!\left[\Pi_h q_h\right]\!\right]_{ij}\mathrm{d}s\\
&=\sum_{F_{ij}\in\mathcal{F}_h}\int_{F_{ij}}\frac{1}{2}\mathrm{D}_i^{-1}\left(\int_{B_i}\nabla p\mathrm{d}x-|B_i|\,\nabla p\right)\cdot\mathbf{n}_{ij}\left[\!\left[\Pi_h q_h\right]\!\right]_{ij}\mathrm{d}s\\
&\quad+\sum_{F_{ij}\in\mathcal{F}_h}\int_{F_{ij}}\frac{1}{2}\mathrm{D}_j^{-1}\left(\int_{B_j}\nabla p\mathrm{d}x-|B_j|\,\nabla p\right)\cdot\mathbf{n}_{ij}\left[\!\left[\Pi_h q_h\right]\!\right]_{ij}\mathrm{d}s.
\end{aligned}
$$

Without loss of generalization, we treat only the case of the first addendum, the other terms being similar. Using the Cauchy-Schwarz inequality, Assumption 3.1.1, Lemmas 3.1.4, 3.1.3 and Hölder inequality:

$$
\begin{aligned}
&\sum_{F_{ij}\in\mathcal{F}_h}\int_{F_{ij}}\frac{1}{2}\mathrm{D}_i^{-1}\left(\int_{B_i}\nabla p\mathrm{d}x-|B_i|\,\nabla p\right)\cdot\mathbf{n}_{ij}\left[\!\left[\Pi_h q_h\right]\!\right]_{ij}\mathrm{d}s\\
&\qquad\le\sum_{F_{ij}\in\mathcal{F}_h}\left(\int_{F_{ij}}\frac{1}{4}\mathrm{D}_i^{-2}\left(\int_{B_i}\nabla p\mathrm{d}x-|B_i|\,\nabla p\right)^2\mathrm{d}s\right)^{\frac{1}{2}}\left(\int_{F_{ij}}d_{ij}^2\frac{\left[\!\left[\Pi_h q_h\right]\!\right]_{ij}^2}{d_{ij}^2}\mathrm{d}s\right)^{\frac{1}{2}}\\
&\qquad\le\sum_{F_{ij}\in\mathcal{F}_h}\mathrm{D}_i^{-1}\,|B_i|\left\|\frac{1}{|B_i|}\int_{B_i}\nabla p\mathrm{d}x-\nabla p\right\|_{L^2(F_{ij})}\sqrt{d_{ij}}\left(d_{ij}\int_{F_{ij}}\frac{\left[\!\left[\Pi_h q_h\right]\!\right]_{ij}^2}{d_{ij}^2}\mathrm{d}s\right)^{\frac{1}{2}}
\end{aligned}
$$

$$
\begin{aligned}
&\leq \sum_{F_{ij}\in\mathcal{F}_h} h_T^{\frac{1}{2}} \mathrm{D}_i^{-1} |B_i| \\
&\qquad \left[h_t^{-1} \left\| \frac{1}{|B_i|}\int_{B_i} \nabla p \mathrm{d}x - \nabla p \right\|_{L^2(B_i)} + h_t \left| \frac{1}{|B_i|}\int_{B_i} \nabla p \mathrm{d}x - \nabla p \right|_{H^1(B_i)} \right] \\
&\qquad \left(d_{ij} \int_{F_{ij}} \frac{[\![\Pi_h q_h]\!]_{ij}^2}{d_{ij}^2} \mathrm{d}s \right)^{\frac{1}{2}} \\
&\leq \sum_{F_{ij}\in\mathcal{F}_h} h_T^{\frac{1}{2}} \mathrm{D}_i^{-1} |B_i| \left[C_d h_T^{-1} h_T |\nabla p|_{H^1(B_i)} + h_T |\nabla p|_{H^1(B_i)} \right] \left(d_{ij} \int_{F_{ij}} \frac{[\![\Pi_h q_h]\!]_{ij}^2}{d_{ij}^2} \mathrm{d}s \right)^{\frac{1}{2}} \\
&\leq \sum_{F_{ij}\in\mathcal{F}_h} C_1 (C_d + h_T) h_T^{\frac{1}{2}} h_T^{2-d} h_T^d |\nabla p|_{H^1(B_i)} \left(d_{ij} \int_{F_{ij}} \frac{[\![\Pi_h q_h]\!]_{ij}^2}{d_{ij}^2} \mathrm{d}s \right)^{\frac{1}{2}} \\
&\leq C_1 C_d h_T^{\frac{3}{2}} \left(\sum_{F_{ij}\in\mathcal{F}_h} |\nabla p|_{H^1(B_i)}^2 \right)^{\frac{1}{2}} \left(\sum_{F_{ij}\in\mathcal{F}_h} h_T^2 d_{ij} \int_{F_{ij}} \frac{[\![\Pi_h q_h]\!]_{ij}^2}{d_{ij}^2} \mathrm{d}s \right)^{\frac{1}{2}} \\
&\leq C_1 C_d h |\nabla p|_{h,1} |q_h|_{T,*} .
\end{aligned}
\tag{3.38}
$$

where D_i^{-1} scales as h^{2-d}:

$$
\begin{aligned}
\mathrm{D}_i &= \sum_{j\in G_i} \frac{|F_{ij}|}{d_{ij}} \simeq \sum_{j\in G_i} h_T^{d-2}, \\
C_2 \frac{h_T^{2-d}}{N_{B_i}} &\leq \mathrm{D}_i^{-1} \leq C_1 \frac{h_T^{2-d}}{N_{B_i}}.
\end{aligned}
\tag{3.39}
$$

Thus, employing Proposition 3.1.2, the final estimate for the consistency reads:

$$
\begin{aligned}
&\mathcal{C}_B((\mathbf{u},p),(\Pi_h\mathbf{v}_h,\Pi_h q_h)) - \widetilde{\mathcal{C}}_B((\mathbf{u}_B,p_B),(\Pi_h\mathbf{v}_h,\Pi_h q_h)) \\
&\qquad \leq C_d h^2 \|\mathbf{f}\|_{L^2} |\mathbf{v}_h|_{H^1} + h |p_B|_{H^1} |\mathbf{v}_h|_* + 2C_1 C_d h^{\frac{3}{2}} |\nabla p|_{h,1} |q_h|_{T,*} \\
&\qquad \lesssim \left(h^2 \|\mathbf{f}\|_{L^2} + h |p_B|_{H^1} \right) |\mathbf{v}_h|_{H^1} + h |\nabla p|_{h,1} |q_h|_{T,*} .
\end{aligned}
$$

□

Lemma 3.1.6 (Continuity of $\mathcal{C}_B$)**.** *Let $(\mathbf{v},q) \in \boldsymbol{\mathcal{V}} \times (\mathcal{Q} \cap H^1(\Omega))$ and $(\mathbf{v}_h, q_h) \in \boldsymbol{\mathcal{V}}_h \times \mathcal{V}_h$, then there holds:*

$$
\mathcal{C}_B((\mathbf{v},q),(\Pi_h\mathbf{v}_h,\Pi_h q_h)) \lesssim \left(|\mathbf{v}|_{H^1} + \|q\|_{L^2} + h |\nabla q|_{1,h} \right) |\!|\!| \mathbf{v}_h, q_h |\!|\!|_{box}.
$$

Proof. Write the compact form $\mathcal{C}_B$:

$$
\mathcal{C}_B((\mathbf{v},q),(\Pi_h\mathbf{v}_h,\Pi_h q_h)) = a_B(\mathbf{v},\Pi_h\mathbf{v}_h) + b_B(\Pi_h\mathbf{v}_h,q) + c_B(\mathbf{v},\Pi_h q_h) + s_B(q,\Pi_h q_h). \tag{3.40}
$$

Consider now each term separately, employing Lemma 3.1.2 and Proposition 3.1.2:

1.
$$a_B(\mathbf{v}, \Pi_h \mathbf{v}_h) = a(\mathbf{v}, \mathbf{v}_h) \leq \nu \left|\mathbf{v}\right|_{H^1} \left|\mathbf{v}_h\right|_{H^1} \leq \nu \left|\mathbf{v}\right|_{H^1} \left|\mathbf{v}_h\right|_* .$$

2. knowing that $\left\|\nabla \cdot \mathbf{v}\right\|_{L^2} \leq \sqrt{d} \left\|\nabla \mathbf{v}\right\|_{L^2} \ \forall \mathbf{v} \in \boldsymbol{\mathcal{V}}$, by the Cauchy-Schwarz inequality,
$$\begin{aligned} b_B(\Pi_h \mathbf{v}_h, q) =& b(\mathbf{v}_h, q) \leq \sqrt{d} \left|\mathbf{v}_h\right|_{H^1} \left\|q\right\|_{L^2} \leq \sqrt{d} \left|\mathbf{v}_h\right|_* \left\|q\right\|_{L^2} , \\ c_B(\mathbf{v}, \Pi_h q_h) =& \sum_{B_i \in \mathcal{B}_h} \int_{B_i} \nabla \cdot \mathbf{v} \Pi_h q_h \mathrm{d}x \\ \leq& \sqrt{d} \sum_{B_i \in \mathcal{B}_h} \left|\mathbf{v}\right|_{H^1(B_i)} \left\|\Pi_h q_h\right\|_{L^2(B_i)} \leq \sqrt{d} \left|\mathbf{v}\right|_{H^1} \left\|\Pi_h q_h\right\|_{L^2} , \end{aligned}$$
where we used Hölder inequality in the last passage.

3. For the Rhie-Chow stabilization we recall inequality (3.38):
$$s_B(q, \Pi_h q_h) \lesssim h \left|\nabla q\right|_{h,1} \left|q_h\right|_{T,*} . \tag{3.41}$$

To conclude the proof we gather all above estimates:
$$\begin{aligned} \mathcal{C}_B((\mathbf{v}, q), (\Pi_h \mathbf{v}_h, \Pi_h q_h)) \lesssim& \left|\mathbf{v}\right|_{H^1} \left|\mathbf{v}_h\right|_* + \left|\mathbf{v}_h\right|_* \left\|q\right\|_{L^2} + \left|\mathbf{v}\right|_* \left\|\Pi_h q_h\right\|_{L^2} + h \left|q\right|_{H^1} \left|q_h\right|_{T,*} \\ \leq& \left(\left|\mathbf{v}\right|_{H^1} + \left\|q\right\|_{L^2} + h \left|\nabla q\right|_{1,h}\right) \left|\!\left|\!\left|\mathbf{v}_h, q_h\right|\!\right|\!\right|_{box}. \end{aligned}$$

□

Lemma 3.1.7 (Continuity of $\widetilde{\mathcal{C}}_B$)**.** *Let $\mathbf{v}_h, \mathbf{w}_h \in \boldsymbol{\mathcal{V}}_h$ and $q_h, z_h \in \mathcal{V}_h$, then there holds:*
$$\widetilde{\mathcal{C}}_B((\mathbf{v}_h, q_h), (\Pi_h \mathbf{w}_h, \Pi_h z_h)) \lesssim \left(\left|\mathbf{v}\right|_{H^1} + \left\|q_h\right\|_{L^2} + h^{\frac{1}{2}} \left|q_h\right|_{H^1}\right) \left|\!\left|\!\left|\mathbf{w}_h, z_h\right|\!\right|\!\right|_{box}.$$

Proof. Consider the compact form
$$\widetilde{\mathcal{C}}_B((\mathbf{v}_h, q_h), (\Pi_h \mathbf{w}_h, \Pi_h z_h)) = a_B(\mathbf{v}_h, \Pi_h \mathbf{w}_h) + \tilde{b}_B(\Pi_h \mathbf{w}_h, q_h) + \tilde{c}_B(\mathbf{v}_h, \Pi_h z_h) + s_B(q_h, \Pi_h z_h).$$
We can rely on proof of Lemma 3.1.6 for both a_B ans s_B. On the other hand, for $\tilde{b}_B$ and $\tilde{c}_B$ we employ equation (3.37), from the proof of Lemma 3.1.5, and equation (3.37):
$$\begin{aligned} a_B(\mathbf{v}_h, \Pi_h \mathbf{w}_h) \leq& \nu \left|\mathbf{v}_h\right|_{H^1} \left|\mathbf{w}_h\right|_* , \\ \tilde{b}_B(\Pi_h \mathbf{w}_h, q_h) \pm b_B(\Pi_h \mathbf{w}_h, q_h) =& b_B(\Pi_h \mathbf{w}_h, q_h) + (\tilde{b}_B - b_B)(\Pi_h \mathbf{w}_h, q_h) \\ \lesssim& \sqrt{d} \left|\mathbf{w}_h\right|_* \left\|q_h\right\|_{L^2} + h \left|q_h\right|_{H^1} \left|\mathbf{w}_h\right|_{T,*} , \\ \tilde{c}_B(\mathbf{v}_h, \Pi_h z_h) \pm c_B(\mathbf{v}_h, \Pi_h z_h) =& c_B(\mathbf{v}_h, \Pi_h z_h) + (\tilde{c}_B - c_B)(\mathbf{v}_h, \Pi_h z_h) \\ \lesssim& \sqrt{d} \left|\mathbf{v}_h\right|_* \left\|\Pi_h z_h\right\|_{L^2} . \end{aligned} \tag{3.42}$$

For the Rhie-Chow stabilization,
$$\begin{aligned} s_B(q_h, \Pi_h q_h) = & \sum_{F_{ij} \in \mathcal{F}_h} \int_{F_{ij}} \frac{1}{2} \mathrm{D}_i^{-1} \left(\int_{B_i} \nabla q_h \mathrm{d}x - \left|B_i\right| \nabla q_h\right) \cdot \mathbf{n}_{ij} \left[\!\left[\Pi_h z_h\right]\!\right]_{ij} \mathrm{d}s \\ & + \sum_{F_{ij} \in \mathcal{F}_h} \int_{F_{ij}} \frac{1}{2} \mathrm{D}_j^{-1} \left(\int_{B_j} \nabla q_h \mathrm{d}x - \left|B_j\right| \nabla q_h\right) \cdot \mathbf{n}_{ij} \left[\!\left[\Pi_h z_h\right]\!\right]_{ij} \mathrm{d}s \end{aligned} \tag{3.43}$$

Without loss of generalization, we treat only the case of the first addendum, the other terms being similar. Using the fact that q_h is piecewise linear and that its normal gradient across face F_{ij} is constant over D_{ij}, by the Cauchy-Schwarz inequality we obtain

$$\begin{aligned}
&\sum_{F_{ij}\in\mathcal{F}_h}\int_{F_{ij}}\frac{1}{2}\mathrm{D}_i^{-1}\left(\int_{B_i}\nabla q_h\mathrm{d}x-|B_i|\,\nabla q_h\right)\cdot\mathbf{n}_{ij}\,[\![\Pi_h z_h]\!]_{ij}\,\mathrm{d}s\\
&\quad=\sum_{F_{ij}\in\mathcal{F}_h}\int_{F_{ij}}\frac{1}{2}\mathrm{D}_i^{-1}\left(\int_{B_i\setminus D_{ij}}\nabla q_h\cdot\mathbf{n}_{ij}\mathrm{d}x\right)[\![\Pi_h z_h]\!]_{ij}\,\mathrm{d}s\\
&\quad\lesssim\sum_{F_{ij}\in\mathcal{F}_h}h_T^{2-d}\left(\int_{F_{ij}}\left|\int_{B_i\setminus D_{ij}}\nabla q_h\cdot\mathbf{n}_{ij}\mathrm{d}x\right|^2\mathrm{d}s\right)^{\frac12}\left(d_{ij}^2\int_{F_{ij}}\left|\frac{\partial z_h}{\partial\mathbf{n}_{ij}}\right|^2\mathrm{d}s\right)^{\frac12}\\
&\quad\le\sum_{F_{ij}\in\mathcal{F}_h}h_T^{2-d}\left(\int_{F_{ij}}\int_{B_i\setminus D_{ij}}|\nabla q_h|^2\,\mathrm{d}x\int_{B_i\setminus D_{ij}}|\mathbf{n}_{ij}|^2\,\mathrm{d}x\mathrm{d}s\right)^{\frac12}\left(d_{ij}^2\int_{F_{ij}}\left|\frac{\partial z_h}{\partial\mathbf{n}_{ij}}\right|^2\mathrm{d}s\right)^{\frac12}\\
&\quad\le\sum_{F_{ij}\in\mathcal{F}_h}h_T^{2-d}\left(h_T^d h_T^{d-3}\,|q_h|_{H^1}^2\right)^{\frac12}\left(h_T^3 d_{ij}\int_{F_{ij}}\left|\frac{\partial z_h}{\partial\mathbf{n}_{ij}}\right|^2\mathrm{d}s\right)^{\frac12}\\
&\quad\le h^{2-d}h^{d-\frac32}\left(\sum_{F_{ij}\in\mathcal{F}_h}|q_h|_{H^1}^2\right)^{\frac12}\left(\sum_{F_{ij}\in\mathcal{F}_h}h_T^3 d_{ij}\int_{F_{ij}}\left|\frac{\partial z_h}{\partial\mathbf{n}_{ij}}\right|^2\mathrm{d}s\right)^{\frac12}\\
&\quad=h^{\frac12}\,|q_h|_{H^1}\,|z_h|_{T,*}
\end{aligned}\tag{3.44}$$

where we used Hölder inequality.

Gathering all the above estimates concludes the proof. □

Lemma 3.1.8 ($\widetilde{\mathcal{C}}_B$ coercivity)**.** *Let $(\mathbf{v}_h,q_h)\in\boldsymbol{\mathcal{V}}_h\times\mathcal{V}_h$, then there holds:*

$$\widetilde{\mathcal{C}}_B((\mathbf{v}_h,q_h),(\Pi_h\mathbf{v}_h,\Pi_h q_h))\ge\nu\,|\mathbf{v}_h|_{H^1}^2+|q_h|_{T,*}^2\,.$$

Proof. Let us first notice that, using Proposition 3.1.1,

$$\begin{aligned}
&\tilde b_B(\Pi_h\mathbf{v}_h,q_h)+\tilde c_B(\mathbf{v}_h,\Pi_h q_h)=\\
&=\sum_{F_{ij}\in\mathcal{F}_h}\int_{F_{ij}}\frac{1}{2}(\Pi_h q_i+\Pi_h q_j)\mathbf{n}_{ij}\cdot[\![\Pi_h\mathbf{v}_h]\!]_{ij}\,\mathrm{d}s+\int_{F_{ij}}\frac{1}{2}(\Pi_h\mathbf{v}_i+\Pi_h\mathbf{v}_j)\cdot\mathbf{n}_{ij}\,[\![\Pi_h q_h]\!]_{ij}\,\mathrm{d}s\\
&=\sum_{F_{ij}\in\mathcal{F}_h}\sum_{j\in G_i}\int_{F_{ij}}[\![\Pi_h q_h\Pi_h\mathbf{v}_h]\!]_{ij}\cdot\mathbf{n}_{ij}\mathrm{d}s=\sum_{B_i\in\mathcal{B}_h}\Pi_h q_i\Pi_h\mathbf{v}_i\sum_{j\in G_i}\int_{F_{ij}}\mathbf{1}\cdot\mathbf{n}_{ij}\mathrm{d}s\\
&=\sum_{B_i\in\mathcal{B}_h}\Pi_h q_i\Pi_h\mathbf{v}_i\int_{B_i}\nabla\cdot\mathbf{1}\mathrm{d}x=0.
\end{aligned}\tag{3.45}$$

Using Lemma 3.1.2, let us first notice that

$$\begin{aligned}\widetilde{\mathcal{C}}_B((\mathbf{v}_h,q_h),(\Pi_h\mathbf{v}_h,\Pi_h q_h)) =& a_B(\mathbf{v}_h,\Pi_h\mathbf{v}_h)+\tilde{b}_B(\Pi_h\mathbf{v}_h,q_h)+\tilde{c}_B(\mathbf{v}_h,\Pi_h q_h)+s_B(q_h,\Pi_h q_h)\\ =& a(\mathbf{v}_h,\mathbf{v}_h)+s_B(q_h,\Pi_h q_h)\\ =& \nu\,|\mathbf{v}_h|^2_{H^1}+s_B(q_h,\Pi_h q_h),\end{aligned}$$

We have now to prove that the Rhie-Chow stabilization is coercive. Consider the following relation which is valid under the hypothesis that the mesh uniform:

$$s_B(q_h,\Pi_h q_h)\simeq\sum_{F_{ij}\in\mathcal{F}_h}\frac{1}{2}\mathrm{D}^{-1}\,|B|\int_{F_{ij}}\left(\frac{1}{|B_i|}\int_{B_i}\nabla q_h-2\nabla q_h+\frac{1}{|B_j|}\int_{B_j}\nabla q_h\right)\cdot\mathbf{n}_{ij}\,[\![\Pi_h q_h]\!]_{ij}\,\mathrm{d}s.$$

In view of

$$|q_h|^2_{T,*}\simeq h^3\,|q_h|^2_*\,, \tag{3.46}$$

to conclude the proof we need to prove

$$s_B(q_h,\Pi_h q_h)\simeq h^3\,|q_h|^2_*\,. \tag{3.47}$$

We numerically assess the validity of (3.47) by computing the minimum generalized eigenvalue of Rhie-Chow stabilization with respect to $*$-norm and showing that it is constant as h goes to zero. We consider the algebraic counterpart of s_B and $|\cdot|_{T,*}$ and we compute

$$R_*(s_B)=\min_{\mathbf{q}\in\mathbb{R}^{N_p}}\frac{\mathbf{q}^\intercal\mathrm{S}\mathbf{q}}{\mathbf{q}^\intercal\mathrm{Q}\mathbf{q}}, \tag{3.48}$$

where Q is the matrix that represents the $*$-norm of $\mathbf{q}$. □

Remark 3.4. [Rhie-Chow stabilization coercivity] To assess the result stated in Lemma 3.1.8, we have to verify numerically that the stabilization bilinear form is coercive. We have generated Voronoi dual grids on a squared domain $\Omega=[-1,1]^d$ where $d=2,3$, we then have written the algebraic systems and then we compute the minimum Rayleigh coefficient (3.48) of s_B and $|\cdot|_*$. The result is reported in Tables 3.1 and 3.2.

As expected, the minimum generalized eigenvalue diminishes with rate 2. This result confirms coercivity of s_B.

Assumption 3.1.2 (Rhie-Chow satisfies generalized *inf-sup*)**.** *Let* $(\mathbf{v}_h,q_h)\in\boldsymbol{\mathcal{V}}_h\times\mathcal{V}_h$*, then* $\exists\beta_h>0$*, independent of* h *s.t.*

$$\sup_{\mathbf{v}_h\in\boldsymbol{\mathcal{V}}_h}\frac{\tilde{c}_B(\mathbf{v}_h,\Pi_h q_h)}{|\mathbf{v}_h|_*}+s_B(q_h,\Pi_h q_h)^{\frac{1}{2}}\geq\beta_h\,\|\Pi_h q_h\|_{L^2}\,. \tag{3.49}$$

Numerical assessment of Assumption 3.1.2. To prove inequality (3.49), we employ a numerical assessment. Recall system 3.21. Let us consider now the Rhie-Chow stabilized monolithic algebraic system corresponding to the Box method formulation of Stokes system 3.14:

$$\begin{bmatrix}\mathrm{A} & \mathrm{B}^\intercal\\ \mathrm{B} & -\mathrm{C}\end{bmatrix}\begin{bmatrix}\mathbf{u}\\ \mathbf{p}\end{bmatrix}=\begin{bmatrix}\mathbf{F}\\ \mathbf{0}\end{bmatrix} \tag{3.50}$$

Coercivity constant of Rhie-Chow stabilization, 2D case						
h	0.025	0.013	0.0063	0.0031	0.0016	0.00078
$R_*(s_B)$	9.6e-07	1.3e-07	1.7e-08	2.2e-09	2.8e-10	3.3e-11
Diminishing rate	–	2.9	2.9	3	3	3

Table 3.1: Minimum generalized eigenvalue of Rhie-Chow matrix with respect to $*$-norm computed on a uniform polygonal mesh. It is also reported the diminishing rate of minimum eigenvalues, computed as $\log_2(R_*|_h/R_*|_{\frac{h}{2}})$, representing the coercivity lower bound of s_B.

Coercivity constant of Rhie-Chow stabilization, 3D case					
h	0.1	0.05	0.025	0.013	0.0063
$R_*(s_B)$	9.3e-05	1.3e-05	1.9e-06	2.6e-07	3.6e-08
Diminishing rate	–	2.8	2.8	2.8	2.9

Table 3.2: Minimum generalized eigenvalue of Rhie-Chow matrix with respect to $*$-norm computed on a uniform polyhedral mesh. It is also reported the diminishing rate of minimum eigenvalues, computed as $\log_2(R_*|_h/R_*|_{\frac{h}{2}})$, representing the coercivity lower bound of s_B.

where -C is the matrix associated to Rhie-Chow stabilization. Equation (3.49) corresponds to the following algebraic inequality: $\forall \mathbf{q} \in \mathbb{R}^{N_p}$,

$$\sup_{\mathbf{w}\in\mathbb{R}^{N_u}} \frac{\mathbf{q}^\intercal \mathrm{B}^\intercal \mathbf{w}}{\sqrt{\mathbf{w}^\intercal \mathrm{H}\mathbf{w}}\sqrt{\mathbf{q}^\intercal \mathrm{V}\mathbf{q}}} + \frac{\sqrt{-\mathbf{q}^\intercal \mathrm{C}\mathbf{q}}}{\sqrt{\mathbf{q}^\intercal \mathrm{V}\mathbf{q}}} \geq \beta_h \tag{3.51}$$

where $\mathrm{V} \in \mathbb{R}^{N_p \times N_p}$ is the mass matrix, i.e. a diagonal matrix with box volumes on diagonal ($\mathrm{V}_{ii} = |B_i|$), representing the L^2-norm of box-wise constant functions, and $\mathrm{H} \in \mathbb{R}^{N_u \times N_u}$ is the matrix representing the $*$-norm in d dimensions. Notice also that by construction of A, it holds $\mathrm{A} = \nu\mathrm{H}$.

Choose now $\mathbf{w} = -\mathrm{A}^{-1}\mathrm{B}^\intercal\mathbf{q}$, then the supremum in equation (3.51) becomes

$$\begin{aligned}\sup_{\mathbf{w}\in\mathbb{R}^{N_u}} \frac{\mathbf{q}^\intercal \mathrm{B}^\intercal \mathbf{w}}{\sqrt{\mathbf{w}^\intercal \mathrm{H}\mathbf{w}}\sqrt{\mathbf{q}^\intercal \mathrm{V}\mathbf{q}}} \geq & \frac{-\mathbf{q}^\intercal \mathrm{B}\mathrm{A}^{-1}\mathrm{B}^\intercal\mathbf{q}}{\sqrt{\mathbf{q}^\intercal \mathrm{B}\mathrm{A}^{-1}\mathrm{H}\mathrm{A}^{-1}\mathrm{B}^\intercal\mathbf{q}}\sqrt{\mathbf{q}^\intercal \mathrm{V}\mathbf{q}}} \\ = & \nu\frac{-\mathbf{q}^\intercal \mathrm{B}\mathrm{A}^{-1}\mathrm{B}^\intercal\mathbf{q}}{\sqrt{\mathbf{q}^\intercal \mathrm{B}\mathrm{A}^{-1}\mathrm{A}\mathrm{A}^{-1}\mathrm{B}^\intercal\mathbf{q}}\sqrt{\mathbf{q}^\intercal \mathrm{V}\mathbf{q}}} \\ = & \nu\frac{\sqrt{-\mathbf{q}^\intercal \mathrm{B}\mathrm{A}^{-1}\mathrm{B}\mathbf{q}}}{\sqrt{\mathbf{q}^\intercal \mathrm{V}\mathbf{q}}}.\end{aligned} \tag{3.52}$$

Then, we have

$$\begin{aligned}\sup_{\mathbf{w}\in\mathbb{R}^{N_u}} \frac{-\mathbf{q}^\intercal \mathrm{B}^\intercal \mathbf{w}}{\sqrt{\mathbf{w}^\intercal \mathrm{H}\mathbf{w}}\sqrt{\mathbf{q}^\intercal \mathrm{V}\mathbf{q}}} + \frac{\sqrt{-\mathbf{q}^\intercal \mathrm{C}\mathbf{q}}}{\sqrt{\mathbf{q}^\intercal \mathrm{V}\mathbf{q}}} &\geq \nu \frac{\sqrt{-\mathbf{q}^\intercal \mathrm{B}\mathrm{A}^{-1}\mathrm{B}\mathbf{q}}}{\sqrt{\mathbf{q}^\intercal \mathrm{V}\mathbf{q}}} + \frac{\sqrt{-\mathbf{q}^\intercal \mathrm{C}\mathbf{q}}}{\sqrt{\mathbf{q}^\intercal \mathrm{V}\mathbf{q}}} \\ &\geq \nu \frac{\sqrt{\mathbf{q}^\intercal \left(-\mathrm{B}\mathrm{A}^{-1}\mathrm{B} - \mathrm{C}\right)\mathbf{q}}}{\sqrt{\mathbf{q}^\intercal \mathrm{V}\mathbf{q}}}.\end{aligned} \tag{3.53}$$

Hence, β_h must be equal to the square root of the generalized eigenvalue of the Schur complement $\mathrm{S}_{\text{box}} = -\nu \mathrm{B}\mathrm{A}^{-1}\mathrm{B}^\intercal$ with respect to matrix V:

$$\beta_h^2 = R_{\Pi_h}(\mathrm{S}_{\text{box}}) = \min_{\mathbf{q}\in\mathbb{R}^{N_p}} \frac{\mathbf{q}^\intercal(\mathrm{S}_{\text{box}} - \mathrm{C})\mathbf{q}}{\mathbf{q}^\intercal \mathrm{V}\mathbf{q}} \tag{3.54}$$

where the explicit expression of C is

$$\mathrm{C} = -\mathrm{B}\mathrm{D}^{-1}\mathrm{B}^\intercal + \mathrm{R}(\mathrm{D}^{-1}) \tag{3.55}$$

where $\mathrm{R}(\mathrm{D}^{-1})$ is a stiffness matrix computed with D^{-1} as diffusivity coefficient (c.f. Section 5.2, equation (5.8)).

To conclude the assessment of Assumption 3.1.2, we have to verify numerically that the algebraic *inf-sup* is satisfied by verifying that β_h does not decrease with h. We consider the same cases of Remark 3.4. For some values of h, we have generated Voronoi dual grids on a squared domain $\Omega = [-1, 1]^d$ where $d = 2, 3$, we then have estimated β_h by computing the minimum generalized eigenvalue of Schur complement with respect to mass matrix using the Rayleigh coefficient of equation 3.52.

As expected, the former eigenvalue does not diminishes with h, as it can be seen also in the diminishing rate that tends to zero. This result confirms that the *inf-sup* is satisfied. □

inf-sup constant, 2D case						
h	0.025	0.013	0.0063	0.0031	0.0016	0.00078
$R_*(S_{\text{box}} - \mathrm{C})$	0.13	0.13	0.14	0.14	0.15	0.15

Table 3.3: Minimum generalized eigenvalue of Schur complement with respect to L^2-norm of box-wise constant functions computed on a uniform polygonal mesh.

Theorem 3.1.3. *Problem* (3.24) *has a unique solution* $(\mathbf{u}_B, p_B) \in \boldsymbol{\mathcal{V}}_h \times \mathcal{V}_h$. *Moreover, the solution is stable with respect to problem data.*

Proof. The Stokes problem is a saddle-point problem, that is well-posed if (see [9, 116]):

1. a_B is continuous and coercive (c.f. Lemmas 3.1.6 and 3.1.8);
2. $\tilde{b}_B$ is continuous (c.f. Lemma 3.1.7);
3. $\tilde{c}_B$ and s_B satisfy the generalized *inf-sup* condition (c.f. Assumption 3.1.2).

□

inf-sup constant, 3D case					
h	0.1	0.05	0.025	0.013	0.0063
$R_*(S_{\text{box}} - \text{C})$	0.06	0.055	0.058	0.062	0.066

Table 3.4: Minimum generalized eigenvalue of Schur complement with respect to L^2-norm of box-wise constant functions computed on a uniform polyhedral mesh.

3.1.3.4 Convergence analysis

Theorem 3.1.4 (Convergence)**.** *Let Ω be of class C^2 and let $\mathbf{f} \in [H^1(\Omega)]^d$. Let $(\mathbf{u}, p) \in (\boldsymbol{\mathcal{V}} \cap [H^2(\Omega)]^d) \times (\mathcal{Q} \cap H^1(\Omega) \cap H^2_{loc}(\Omega))$ being the solution to problem* (3.9)*. Then*

$$\|\mathbf{u}_B - \mathbf{u}\|_{H^1} + \|p_B - p\|_{L^2} \lesssim h$$

where C is a positive constant only dependent on problem data.

Proof. Let us define $I\mathbf{u}, Ip \in \boldsymbol{\mathcal{V}}_h \times \mathcal{V}_h$ be the Lagrangian linear interpolations of exact solutions $\mathbf{u}, p$. Define also $\boldsymbol{\varepsilon} = \mathbf{u} - I\mathbf{u}$, $\boldsymbol{\varepsilon}_B = I\mathbf{u} - \mathbf{u}_B$ and $\eta = p - Ip$, $\eta_B = Ip - p_B$. Employing Lemma 3.1.8 we have

$$\begin{aligned}
|\eta_B|^2_{T,*} + \nu\,|\boldsymbol{\varepsilon}_B|^2_{H^1} \leq& \widetilde{\mathcal{C}}_B((\boldsymbol{\varepsilon}_B, \eta_B), (\Pi_h\boldsymbol{\varepsilon}_B, \Pi_h\eta_B)) \pm b_B(\Pi_h\boldsymbol{\varepsilon}_B, p) \pm c_B(\mathbf{u}, \Pi_h\eta_B) \\
=& \mathcal{C}_B((\boldsymbol{\varepsilon}, \eta), (\Pi_h\boldsymbol{\varepsilon}_B, \Pi_h\eta_B)) + (\tilde{b}_B - b_B)(\Pi_h\boldsymbol{\varepsilon}_B, Ip) + (\tilde{c}_B - c_B)(I\mathbf{u}, \Pi_h\eta_B) \\
&+ \mathcal{C}_B((\mathbf{u}, p), (\Pi_h\boldsymbol{\varepsilon}_B, \Pi_h\eta_B)) - \widetilde{\mathcal{C}}_B((\mathbf{u}_B, p_B), (\Pi_h\boldsymbol{\varepsilon}_B, \Pi_h\eta_B)).
\end{aligned} \tag{3.56}$$

Now, using continuity of $\mathcal{C}_B$ (Lemma 3.1.6) on the first term, consistency of face interpolation (equation (3.37), from the proof of Lemma 3.1.5,) on second and third terms and consistency (lemma 3.1.5) on the fourth and fifth terms, we get the following:

$$\begin{aligned}
|\eta_B|^2_{T,*} + \nu\,|\boldsymbol{\varepsilon}_B|^2_{H^1} \leq& \widetilde{\mathcal{C}}_B((\boldsymbol{\varepsilon}_B, \eta_B), (\Pi_h\boldsymbol{\varepsilon}_B, \Pi_h\eta_B)) \\
\lesssim& \left(|\boldsymbol{\varepsilon}|_{H^1} + \|\eta\|_{L^2} + h\,|\nabla\eta|_{1,h}\right) |||\boldsymbol{\varepsilon}_B, \eta_B|||_{box} \\
&+ h\,|Ip|_{H^1}\,|\boldsymbol{\varepsilon}_B|_{H^1} \\
&+ \left(h^2\,\|\mathbf{f}\|_{L^2} + h\,|p_B|_{H^1} + h\,|\nabla p|_{h,1}\right) |||\boldsymbol{\varepsilon}_B, \eta_B|||_{box} \\
\lesssim& \left(|\boldsymbol{\varepsilon}|_{H^1} + \|\eta\|_{L^2} + hC(\mathbf{f}, \mathbf{u}, p)\right) |||\boldsymbol{\varepsilon}_B, \eta_B|||_{box} \\
=:& \mathbb{S}|||\boldsymbol{\varepsilon}_B, \eta_B|||_{box}.
\end{aligned} \tag{3.57}$$

where we observed that

$$|\nabla\eta|^2_{1,h} = \sum_{B_i \in \mathcal{B}_h} \int_{B_i} D^2(p - Ip)\mathrm{d}x = \sum_{B_i \in \mathcal{B}_h} \int_{B_i} D^2 p\mathrm{d}x = |\nabla p|^2_{1,h}$$

and where, in the last passage, C comes from the stability of Lagrangian interpolator and from the continuity with respect to $\mathbf{f}$ of the continuous and box solutions (c.f. Theorems 3.1.1 and 3.1.3).

Now we consider the generalized *inf-sup*. By relationship (3.45),

$$\begin{aligned}
\beta_h \left\|\Pi_h \eta_B\right\|_{L^2} &\leq \sup_{\mathbf{v}_h \in \boldsymbol{\mathcal{V}}_h} \frac{\tilde{c}_B(\mathbf{v}_h, \Pi_h \eta_B)}{|\mathbf{v}_h|_*} + s_B(\eta_B, \Pi_h \eta_B)^{\frac{1}{2}} \\
&= \sup_{\mathbf{v}_h \in \boldsymbol{\mathcal{V}}_h} \frac{-\tilde{b}_B(\Pi_h \mathbf{v}_h, \eta_B)}{|\mathbf{v}_h|_*} + s_B(\eta_B, \Pi_h \eta_B)^{\frac{1}{2}} \\
&= \sup_{\mathbf{v}_h \in \boldsymbol{\mathcal{V}}_h} \frac{a_B(\mathbf{v}_h, \Pi_h \boldsymbol{\varepsilon}_B) - \widetilde{\mathcal{C}}_B((\mathbf{v}_h, \eta_B), (\Pi_h \mathbf{v}_h, 0))}{|\mathbf{v}_h|_*} + s_B(\eta_B, \Pi_h \eta_B)^{\frac{1}{2}} \\
&= \sup_{\mathbf{v}_h \in \boldsymbol{\mathcal{V}}_h} \frac{a_B(\mathbf{v}_h, \Pi_h \boldsymbol{\varepsilon}_B)}{|\mathbf{v}_h|_*} + s_B(\eta_B, \Pi_h \eta_B)^{\frac{1}{2}} - \sup_{\mathbf{v}_h \in \boldsymbol{\mathcal{V}}_h} \frac{\widetilde{\mathcal{C}}_B((\mathbf{v}_h, \eta_B), (\Pi_h \mathbf{v}_h, 0))}{|||\mathbf{v}_h, 0|||_{box}}
\end{aligned} \tag{3.58}$$

Now consider separately each term:

1. By continuity of a_B (c.f. proof of Lemma 3.1.6),

$$\sup_{\mathbf{v}_h \in \boldsymbol{\mathcal{V}}_h} \frac{a_B(\mathbf{v}_h, \Pi_h \boldsymbol{\varepsilon}_B)}{|\mathbf{v}_h|_*} \leq \frac{\nu \left|\mathbf{v}_h\right|_{H^1} \left|\boldsymbol{\varepsilon}_B\right|_*}{|\mathbf{v}_h|_*} \leq \nu \left|\boldsymbol{\varepsilon}_B\right|_*$$

2. By continuity of s_B (c.f. proof of Lemma 3.1.6, in particular equation (3.41)) and by Proposition 3.1.2,

$$s_B(\eta_B, \Pi_h \eta_B) \lesssim h \left|\eta_B\right|_{H^1} \left|\eta_B\right|_{T,*} \leq h^2 \left\|\eta_B\right\|_*^2 \leq h^2 h_m^{-2} \left\|\Pi_h \eta_B\right\|_{L^2}^2 \leq \delta \left\|\Pi_h \eta_B\right\|_{L^2}^2 .$$

   where we recall that $\delta = h/h_m$ (c.f. Assumption 3.1.1).

3. Using equation (3.57),

$$\sup_{\mathbf{v}_h \in \boldsymbol{\mathcal{V}}_h} \frac{\widetilde{\mathcal{C}}_B((\mathbf{v}_h, \eta_B), (\Pi_h \mathbf{v}_h, 0))}{|||\mathbf{v}_h, 0|||_{box}} \leq \mathbb{S} \frac{|||\mathbf{v}_h, 0|||_{box}}{|||\mathbf{v}_h, 0|||_{box}} = \mathbb{S}.$$

Putting together points (1-3) above, we obtain

$$\beta_h \left\|\Pi_h \eta_B\right\|_{L^2} \lesssim \nu \left|\boldsymbol{\varepsilon}_B\right|_* + \delta \left\|\eta_B\right\|_{L^2} + \mathbb{S}. \tag{3.59}$$

Following [49, Lemma 6.13], consider

$$\left\|\eta_B\right\|_{L^2}^2 \lesssim \left|\boldsymbol{\varepsilon}_B\right|_*^2 + \left\|\eta_B\right\|_{L^2}^2 + \mathbb{S}^2 \tag{3.60}$$

and

$$\left|\eta_B\right|_{T,*}^2 + \nu \left|\boldsymbol{\varepsilon}_B\right|_{H^1}^2 \lesssim \mathbb{S} |||\boldsymbol{\varepsilon}_B, \eta_B|||_{box}. \tag{3.61}$$

By Young inequality

$$\left|\eta_B\right|_{T,*}^2 + \left\|\eta_B\right\|_{L^2}^2 + \nu \left|\boldsymbol{\varepsilon}_B\right|_{H^1}^2 \lesssim \mathbb{S} |||\boldsymbol{\varepsilon}_B, \eta_B|||_{box} + \mathbb{S}^2 \leq \frac{1}{2}\mathbb{S}^2 + \frac{1}{2} |||\boldsymbol{\varepsilon}_B, \eta_B|||_{box}^2 + \mathbb{S}^2, \tag{3.62}$$

from which follows

$$\begin{aligned}\frac{1}{2}|||\boldsymbol{\varepsilon}_B,\eta_B|||^2_{box} \lesssim \mathbb{S}^2 &= \left(|\boldsymbol{\varepsilon}|_{H^1}+\|\eta\|_{L^2}+hC(\mathbf{f},\mathbf{u},p)\right)^2\\ &= \left(|\boldsymbol{\varepsilon}|_{H^1}+\|\eta\|_{L^2}+hC(\mathbf{f},\mathbf{u},p)\right)^2.\end{aligned} \tag{3.63}$$

To conclude the proof, use the triangular inequality and Proposition 3.1.2 and write

$$\begin{aligned}|\mathbf{u}_B-\mathbf{u}|_{H^1}+\|p_B-p\|_{L^2} &\le |\mathbf{u}_B-I\mathbf{u}|_{H^1}+\|p_B-Ip\|_{L^2}+|I\mathbf{u}-\mathbf{u}|_{H^1}+\|Ip-p\|_{L^2}\\ &\le |\boldsymbol{\varepsilon}_B|_{H^1}+\|\Pi_h\eta_B\|_{L^2}+|I\mathbf{u}-\mathbf{u}|_{H^1}+\|Ip-p\|_{L^2}\\ &\le |I\mathbf{u}-\mathbf{u}|_{H^1}+\|Ip-p\|_{L^2}+|||\boldsymbol{\varepsilon}_B,\eta_B|||_{box}\end{aligned} \tag{3.64}$$

where we used Lemma 3.1.1 in the following way:

$$\begin{aligned}\|p_B-Ip\|_{L^2} &\le \|p_B-\Pi_h p_B\|_{L^2}+\|\Pi_h p_B-\Pi_h Ip\|_{L^2}+\|Ip-\Pi_h Ip\|_{L^2}\\ &\lesssim h\,|p_B|_{H^1}+h\,|Ip|_{H^1}+\|\Pi_h\eta_B\|_{L^2}.\end{aligned} \tag{3.65}$$

Employing inequality (3.63) in the latter we obtain the following estimate depending on the interpolation errors of $I\mathbf{u}, Ip$:

$$|\mathbf{u}_B-\mathbf{u}|_{H^1}+\|p_B-p\|_{L^2} \lesssim \|I\mathbf{u}-\mathbf{u}\|_{H^1}+\|Ip-p\|_{L^2}+hC(\mathbf{f},\mathbf{u},p), \tag{3.66}$$

that, using the interpolation estimates for η and $\boldsymbol{\varepsilon}$ and neglecting higher order terms in h reads:

$$|\mathbf{u}_B-\mathbf{u}|_{H^1}+\|p_B-p\|_{L^2} \le hC(\mathbf{f},\mathbf{u},p). \tag{3.67}$$

□

Remark 3.5. In Theorem 3.1.4 we used quite strong regularity assumptions: domain Ω of class C^2 and the source term $\mathbf{f} \in [H^1(\Omega)]^d$. This is because we need pressure field to belong to $H^1(\Omega)$ and also to $H^2_{\text{loc}}(\Omega)$ [67, Theorems IV.4.1, IV.6.1] to satisfy assumptions of Lemma 3.1.5. In particular, these assumptions are needed when dealing with the Rhie-Chow stabilization, that directly involves pressure gradient, indeed the regularity assumptions are employed when proving consistency, continuity and coercivity of the stabilization term (c.f. Lemmas 3.1.5, 3.1.7 and 3.1.8).

Now, using triangular inequality and Theorems 3.1.2.1 and 3.1.4, we can find a relationship between FEM and BM solutions:

$$\|\mathbf{u}_B-\mathbf{u}_h\|_{H^1}+\|p_B-p_h\|_{L^2} \le Ch. \tag{3.68}$$

3.1.3.5 Numerical experiments

Here we present some numerical experiments to assess the theoretical analysis of Section 3.1.3.4. The objective is to validate estimate (3.68) proven in previous section. The method used to solve the Stokes equations is the SIMPLE splitting method, described in Section 5.1.1.

As mentioned in Section 3.1.1, all the computations have been performed employing a Voronoi dual mesh of a Delaunay triangulation (i.e. connecting circumcentres of triangles with straight lines). To generate the dual mesh we relied on a tool implemented in OpenFOAM called `polyDualMesh` modified in order to use circumcentres of triangles and nodes of dual mesh instead of barycentres.

The first is a 2D case. We consider the domain $\Omega = [-1/4, 1/4]^2$. We set the analytic solution to

$$\begin{aligned} \mathbf{u} &= \begin{bmatrix} -\sin(2\pi y)\cos(2\pi x) \\ \sin(2\pi x)\cos(2\pi y) \end{bmatrix}, \\ p &= -\frac{1}{4}(\cos(4\pi x) + \cos(4\pi y)) \end{aligned} \tag{3.69}$$

and we consider the following values of h:

$$h = 0.025, 0.0125, 0.00625, 0.003125.$$

The second is a 3D case. We consider the domain $\Omega = [-1/4, 1/4]^3$. We set the analytic solution to

$$\begin{aligned} \mathbf{u} &= \begin{bmatrix} \sin(2\pi y)\sin(2\pi z)\cos(2\pi x) \\ \sin(2\pi x)\sin(2\pi z)\cos(2\pi y) \\ -2\sin(2\pi x)\sin(2\pi y)\cos(2\pi z) \end{bmatrix}, \\ p &= -\frac{1}{8}(\cos(4\pi x) + \cos(4\pi y) + \cos(4\pi z)) \end{aligned} \tag{3.70}$$

and we consider the following values of h:

$$h = 0.05, 0.025, 0.0125, 0.00625.$$

For both cases we set the boundary conditions accordingly to analytic solutions and $\mathbf{f} = -\Delta\mathbf{u} + \nabla p$.

Solving the problems, we obtain the convergence rates of the H^1 error for the velocity and L^2 error for the pressure, represented in Figure 3.4. Recalling inequality (3.68), we should have a global convergence rate of 1. Indeed, for both velocity H^1 error and pressure L^2 error we observe a rate of convergence of order 1 in 2D and a rate of order slightly lower than 1 in 3D.

We impute the slightly lower convergence order in the pressure error of the 3D case to the fact that, in particular in 3D, is not easy to build a a triangulation that is perfectly Delaunay, and this reflects on the quality of Voronoi dual grid and consequently on the accuracy of the method.

3.2 The Diffuse Interface Box method

Let $\Omega \subset \mathbb{R}^d$, $d = 2, 3$ be a (non-polygonal) domain and let $\tilde{\Omega}$ be a hold-all domain such that $\Omega \subset \tilde{\Omega}$. In the sequel (c.f. Section 3.2.1), we will work under the hypothesis $\Gamma = \partial\Omega \in C^{1,1}$.

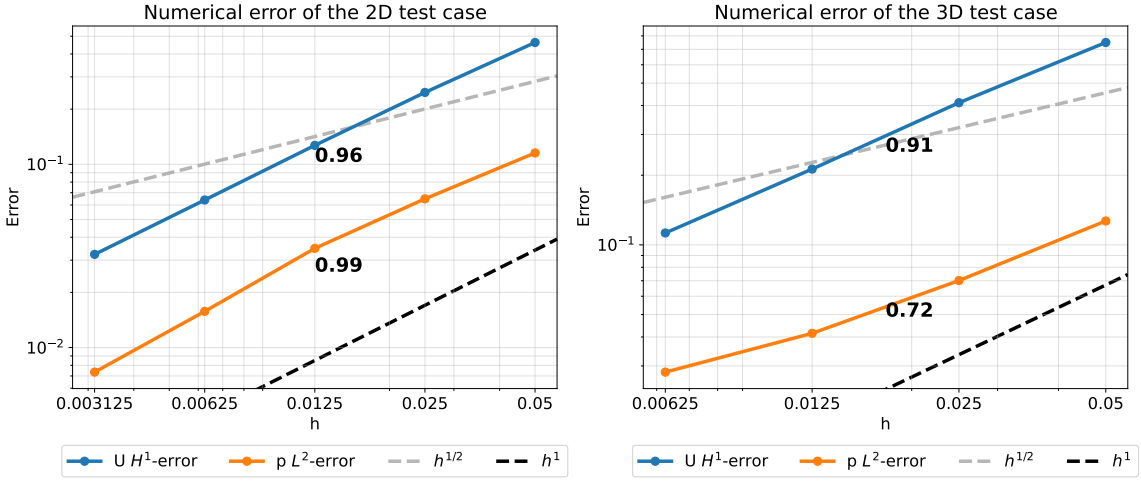


Figure 3.4: Convergence rates of the numerical error of BM solutions. On the left the 2D case errors and on the right the 3D case ones. Numbers are the rates computed using a Least Square approximation on the log-log plot values.

With a slight abuse of notation we denote by $\mathcal{T}_h$ a shape regular triangulation of Ω. It is worth noting that $\mathcal{T}_h$ is not conforming with Ω. Following [124] we first select a tubular neighbourhood S^ϵ of Γ, where ϵ denotes the width of S^ϵ (see Figures 3.5 and 3.6). Then we introduce the set S_h^ϵ which contains all the triangles of $\mathcal{T}_h$ having non-empty intersection with S^ϵ. Note that the width of the discrete tubular neighbourhood S_h^ϵ is $\delta + \epsilon$ where δ is the maximum diameter of triangles crossed by ∂S^ϵ (see Figures 3.5 and 3.7).

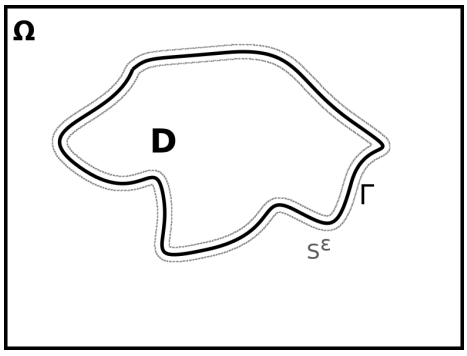


Figure 3.5: Diffuse interface representation: D is a surrogate domain of $\tilde{\Omega}$; Γ is the Dirichlet boundary and S^ϵ is its tubular neighbour.

3.2.1 Poisson problem

To proceed, we assume that there exists an extension $\tilde{g} \in H^2(\tilde{\Omega})$ of the boundary data g.

We set $\Omega_h^\epsilon = \Omega \backslash S_h^\epsilon$ and introduce the function $u^{\epsilon,h} \in H^1(\Omega_h^\epsilon)$ such that $u^{\epsilon,h} = g$ on $\partial\Omega_h^\epsilon$,

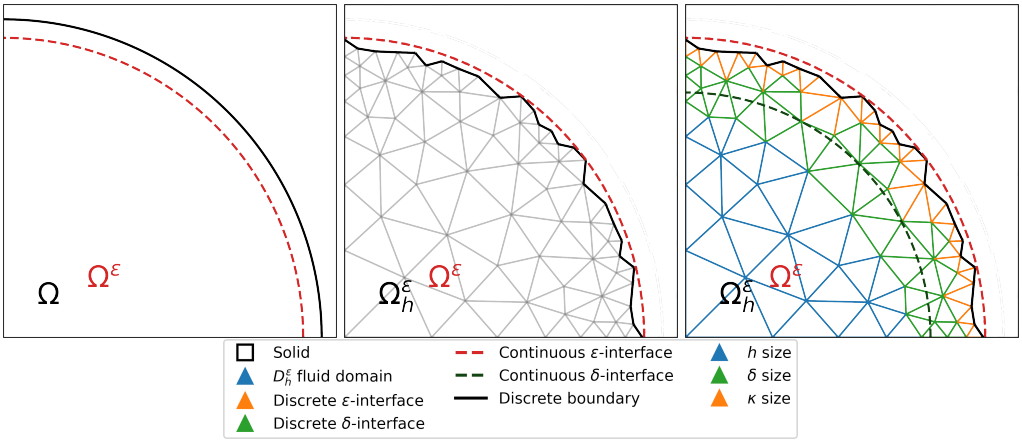


Figure 3.6: Left: continuous diffuse interface. Centre: Discrete diffuse interface. Right: mesh sizes subdivision.

which solves the following continuos problem:

$$\int_{\Omega_h^\epsilon} \nabla u^{\epsilon,h} \cdot \nabla v = \int_{\Omega_h^\epsilon} fv \quad \forall v \in H_0^1(\Omega_h^\epsilon). \tag{3.71}$$

The solution $u^{\epsilon,h}$ is then extended to S_h^ϵ by setting $u^{\epsilon,h} = \tilde{g}$ in S_h^ϵ.
The following results have been proved in [124, Thm 1.2]:

$$\frac{1}{\epsilon+\delta}\left\|u-u^{\epsilon,h}\right\|_{L^2(\Omega)} + \frac{1}{\sqrt{\epsilon+\delta}}\left\|\nabla u - \nabla u^{\epsilon,h}\right\|_{L^2(\Omega)} \leq C\left(\|f\|_{L^2(\Omega)} + \|g\|_{H^2(\Omega)}\right). \tag{3.72}$$

Let $\mathcal{V}^\epsilon_{h,\tilde{g}_h} = \left\{v_h|_{\Omega_h^\epsilon} : v_h \in \mathbb{P}^1(t) \forall t \in \mathcal{T}_h \text{ and } v_h = \tilde{g}_h \text{ on } \partial\Omega_h^\epsilon\right\}$, with $\tilde{g}_h$ the Lagrangian piecewise linear interpolant of $\tilde{g}$.

It has been proved (cf. [124, Thms 5.1 and 5.3]) that the linear finite element approximation $u^\epsilon_{G,h} \in \mathcal{V}^\epsilon_{h,\tilde{g}_h}$ of $u^{\epsilon,h}$ satisfies the following estimates:

$$\begin{aligned}\left\|\nabla(u^{\epsilon,h}-u_h^\epsilon)\right\|_{L^2(\Omega)} &\leq C(\sqrt{\delta}+\kappa^{\frac{2}{3}}+h)\left(\|f\|_{L^2(\Omega)}+\|\tilde{g}\|_{H^2(\Omega)}\right),\\ \left\|u^{\epsilon,h}-u_h^\epsilon\right\|_{L^2(\Omega)} &\leq C(\delta+\kappa^{\frac{4}{3}}+h^2)\left(\|f\|_{L^2(\Omega)}+\|\tilde{g}\|_{H^2(\Omega)}\right),\end{aligned} \tag{3.73}$$

where κ is the maximum diameter of the triangles intersecting $\partial S^{\epsilon+h}$ and $u^\epsilon_{G,h}$ has been extended to Ω_h^ϵ by setting $u^\epsilon_{G,h} = \tilde{g}_h$ on S_h^ϵ.

Let us now introduce the Box method with diffuse interface (DIBM). We denote by $u^\epsilon_{B,h} \in \mathcal{V}^\epsilon_{h,\tilde{g}_h}$, the approximation obtained from applying the Box method to (3.71) (cf. (3.3)). The solution $u^\epsilon_{B,h}$ is then extended to Ω by setting $u^\epsilon_{B,h} = \tilde{g}_h$ in S_h^ϵ. Then employing the triangle inequality in combination with (3.72), (3.73) and (3.6) we get the following estimates for DIBM:

$$\begin{aligned}\left\|\nabla(u-u^\epsilon_{B,h})\right\|_{L^2(\Omega)} &\lesssim \sqrt{\epsilon+\delta}+\sqrt{\delta}+k^{\frac{2}{3}}+h,\\ \left\|u-u^\epsilon_{B,h}\right\|_{L^2(\Omega)} &\lesssim \epsilon+\delta+k^{\frac{4}{3}}+h^2.\end{aligned} \tag{3.74}$$

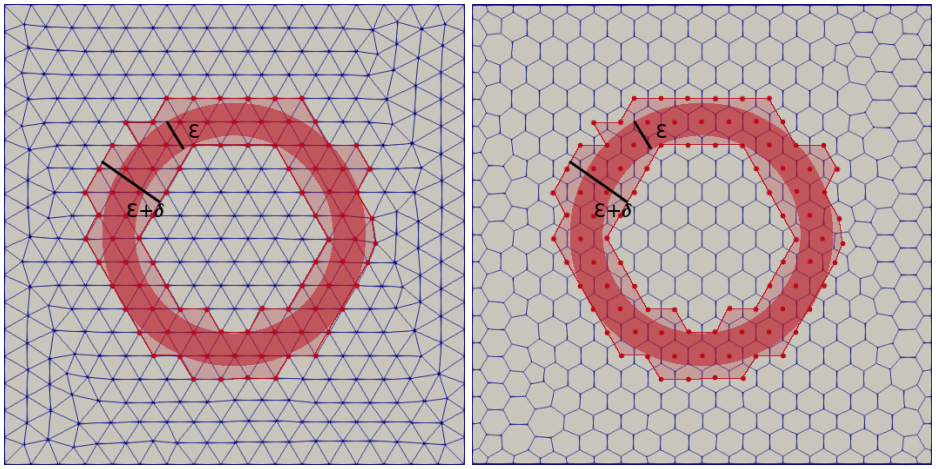


Figure 3.7: Discrete diffuse interface representation on triangulation (left) and on box mesh (right). Constrained cells are marked with red dots while the continuous and discrete diffuse interfaces are coloured by darker an lighter red respectively.

Remark 3.6. The existence of the H^2-extension $\tilde{g}$ of the boundary datum g is required in [124] to prove (3.73), where an equivalent problem with homogeneous boundary conditions and modified right hand side $\tilde{f} = f - \Delta\tilde{g}$ is considered. We note that in practice, the extension of g is only needed on S^ϵ and, depending on the regularity of $\partial\Omega$, a constant extension along the normal to the boundary may be sufficient.

3.2.1.1 Numerical experiments

In this section we numerically assess the theoretical estimates obtained in previous section. To this aim, we consider the test case originally introduced in [124, Section 6] that is briefly recalled in the sequel. Let $\tilde{\Omega} = (-1,1)^2$ and let Γ be the boundary of the circle $B_1(0)$ with centre $(0,0)$ and unitary radius. Thus, Γ splits the domain $\tilde{\Omega}$ into two subregions: $\Omega_1 = B_1(0)$ and $D_2 = \Omega \setminus \overline{\Omega}_1$. Let u be the solution of the following problem

$$-\Delta u = f \ \text{ in } \Omega, \qquad u = g \ \text{ on } \Gamma, \qquad u = 0 \ \text{ on } \partial\tilde{\Omega}, \tag{3.75}$$

where $g(x,y) = (4-x^2)(4-y^2)$ on Γ and extended to $\tilde{\Omega}$ as $\tilde{g}(x,y) = (4-x^2)(4-y^2)\cos(1-x^2-y^2)$.

Setting the solution equal to:

$$u(x,y) = (4-x^2)(4-y^2)\left(\chi_{D_2} + \exp(1-x^2-y^2)\chi_{\bar{D}_1}\right), \tag{3.76}$$

where χ_{D_i}, $i = 1,2$ are the characteristic functions of the two parts of Ω, the source term f is chosen as:

$$f = \begin{cases} -\Delta u & \text{in } \tilde{\Omega}\backslash\Gamma, \\ 0 & \text{on } \Gamma. \end{cases}$$

All the computations have been performed employing a Voronoi dual mesh of a Delaunay triangulation (i.e., the dual mesh is obtained by connecting the barycentres of the triangles with straight lines).

To validate the estimates (3.74) we consider in a separate way the influence of h and ϵ on the error. More precisely, we first explore the convergence with respect to h and then we study the convergence with respect to ϵ. In both cases we consider a uniform discretization of the domain $\tilde{\Omega}$ so to have $\kappa = \delta = h$.

Convergence w.r.t. h We set $\epsilon = 2^{-20} \ll h$ while we let h vary as

$$h = 0.056, 0.028, 0.0139, 0.00694.$$

From Figure 3.8 we observe that the L^2-norm of the error decreases with order 1 while the error decreases with order $1/2$ in the H^1-norm. These rates of convergence are in agreement with (3.74).

Remark 3.7. If a local refinement of the diffuse interface region is performed in such a way that $\delta \simeq \kappa \simeq h^2$ (Figure 3.10), then first and second order of convergence are recovered for H^1 and L^2 norms, respectively (cf. [124, Section 6]). Moreover, comparing the behaviour of the errors in terms of the number of degrees of freedom (cf. Figure 3.11) clearly shows the advantage of employing the adaptive strategy over the use of uniformly refined grids.

Convergence w.r.t. ϵ We employ a fine mesh ($h = 0.00694$) and let the value of ϵ vary as:

$$\epsilon = 2^i, \ i = -1, ..., -20.$$

The results are collected in Figure 3.9. The theoretical rates of convergence with respect to ϵ (cf. (3.74)) are obtained both in the L^2-norm (order 1) and in the H^1- norm (order $1/2$). It is worth noticing that when the value of ϵ becomes smaller than the chosen value of h, a plateau is observed as the (fixed) contribution from the discretization of the PDE (related to h) dominates over the contribution from the introduction of the diffuse interface (related to ϵ).

3.2.2 Stokes problem

In this section we deduce the Diffuse Interface method for the Stokes problem. Let $\Omega^\epsilon = \Omega \backslash S^\epsilon$ (Figure 3.6) and $\boldsymbol{\mathcal{V}}^\epsilon = [H^1(\Omega^\epsilon)]^d$, $\mathcal{Q}^\epsilon = L^2(\Omega^\epsilon)$. Let $\tilde{\mathbf{g}}$ be an extension of the Dirichlet boundary datum s.t. $\tilde{\mathbf{g}} \in [H^2(\tilde{\Omega})]^d$.

Consider then the problem written on Ω^ϵ: find $(\mathbf{u}^\epsilon, p^\epsilon) \in \boldsymbol{\mathcal{V}}^\epsilon \times \mathcal{Q}^\epsilon$ such that $\mathbf{u}^\epsilon = \tilde{\mathbf{g}}$ on $\partial\Omega^\epsilon$, which solves

$$\begin{aligned} -\nu\Delta\mathbf{u}^\epsilon + \nabla p^\epsilon &= \mathbf{f}, \quad \text{in } \Omega^\epsilon, \\ \nabla \cdot \mathbf{u}^\epsilon &= 0, \quad \text{in } \Omega^\epsilon, \end{aligned} \tag{3.77}$$

where it is worth noticing that $\mathbf{u} - \mathbf{u}^\epsilon = \mathbf{0}$, on $\partial\Omega$.

Moreover, we can state the following stability result:

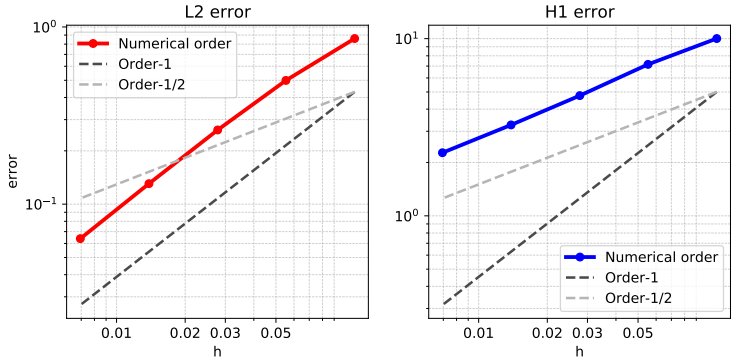


Figure 3.8: Poisson problem error behaviour with respect to h (fixed $\epsilon = 2^{-20}$): (left) L^2-norm error, (right) H^1-norm error. Dashed lines are theoretical convergence orders.

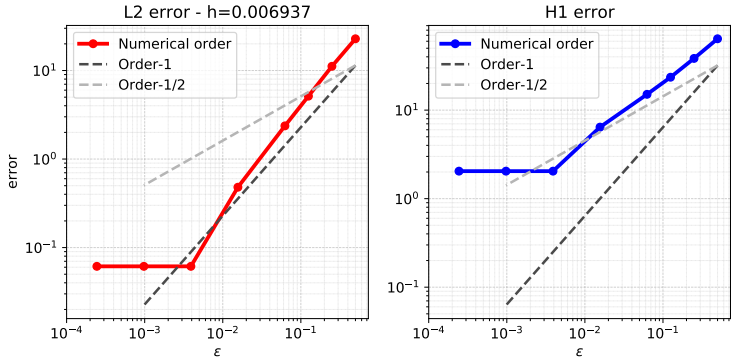


Figure 3.9: Poisson problem error behaviour with respect to ϵ (fixed $h = 0.00694$): (left) L^2-norm error, (right) H^1-norm error. Dotted lines are theoretical convergence orders.

Theorem 3.2.1. *Solution of problem* $(\mathbf{u}^\epsilon, p^\epsilon) \in \boldsymbol{\mathcal{V}}^\epsilon \cap [H^2(\Omega^\epsilon)]^d \times \mathcal{Q}^\epsilon \cap H^1(\Omega^\epsilon)$ *of problem* (3.77) *satisfy the following relationship:*

$$\|\mathbf{u}^\epsilon\|_{H^1(\Omega)} + \|p^\epsilon\|_{L^2(\Omega)} \leq C \left(\|\mathbf{f}\|_{L^2(\Omega)} + \|\tilde{\mathbf{g}}\|_{H^2(\Omega)} \right).$$

$C > 0$ *is independent of* ϵ.

We first recall notation defined at the beginning of Section 3.2 and in Figure 3.6. As before, define the following functional spaces on the discrete diffuse set $\Omega_h^\epsilon = \Omega \backslash S_h^\epsilon$: $\boldsymbol{\mathcal{V}}^{\epsilon,h} = \{\mathbf{v} \in [H^1(\Omega_h^\epsilon)]^d\}$, $\mathcal{Q}^{\epsilon,h} = \{q \in L^2(\Omega_h^\epsilon)\}$.

The problem on Ω_h^ϵ will read: find $(\mathbf{u}^{\epsilon,h}, p^{\epsilon,h}) \in \boldsymbol{\mathcal{V}}^{\epsilon,h} \times \mathcal{Q}^{\epsilon,h}$ such that $\mathbf{u}^{\epsilon,h} = \tilde{\mathbf{g}}$ on $\partial\Omega_h^\epsilon$, which solves

$$\begin{aligned} -\nu\Delta\mathbf{u}^{\epsilon,h} + \nabla p^{\epsilon,h} &= \mathbf{f}, \quad \text{in } \Omega_h^\epsilon, \\ \nabla \cdot \mathbf{u}^{\epsilon,h} &= 0, \quad \text{in } \Omega_h^\epsilon. \end{aligned} \tag{3.78}$$

Now we introduce the discrete formulation of the diffuse interface problem. Let

$$\boldsymbol{\mathcal{V}}_h^\epsilon = \left\{ v_h|_{\Omega_h^\epsilon} : v_h \in \mathbb{P}^1(T) \, \forall T \in \mathcal{T}_h \right\}$$

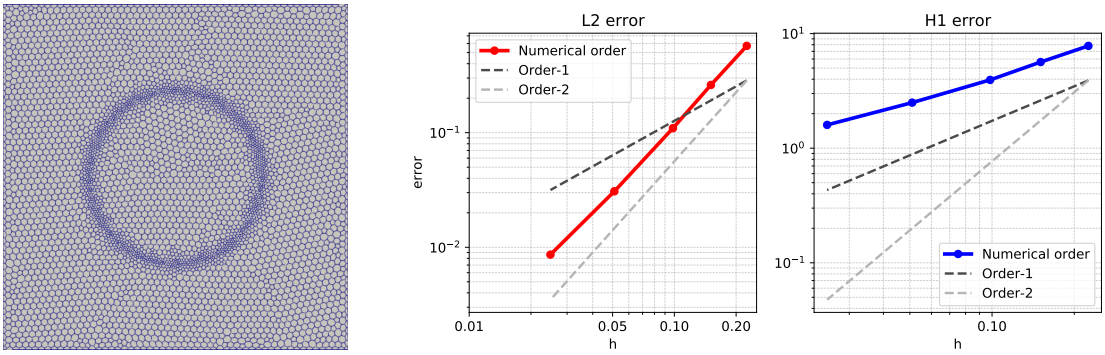


Figure 3.10: On the left: example of a dual mesh with local mesh refinement around surrogate boundary. On the right: error behaviour with respect to h with local mesh refinement around the interface (fixed $\epsilon = 2^{-20}$): (left) L^2-norm error, (right) H^1-norm error. Dashed lines are theoretical convergence orders.

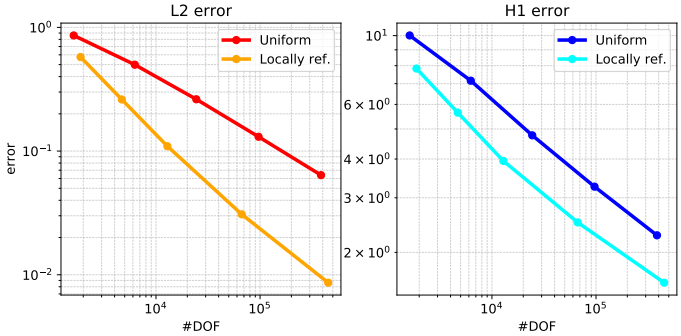


Figure 3.11: Plot of L^2-norm (left) and H^1-norm (right) error against the number of degrees of freedom for uniformly and locally refined meshes.

be the space of piecewise-linear functions and let

$$\boldsymbol{\mathcal{V}}^{\epsilon}_{h,\tilde{g}_h} = \left\{ \mathbf{v}_h|_{\Omega^{\epsilon}_h} : \mathbf{v}_h \in [\mathbb{P}^1(T)]^d \, \forall T \in \mathcal{T}_h \text{ and } \mathbf{v}_h = \tilde{\mathbf{g}}_h \text{ on } \partial\Omega^{\epsilon}_h \right\}$$

be the space of vectorial piecewise-linear functions on $\mathcal{T}_h$ that have value $\tilde{\mathbf{g}}_h$ on the discrete diffuse interface boundary, with $\tilde{\mathbf{g}}_h$ the Lagrangian piecewise linear interpolant of $\tilde{\mathbf{g}}$. We can then write the Galerkin formulation for the problem (3.78): find $(\mathbf{u}^{\epsilon}_h, p^{\epsilon}_h) \in \boldsymbol{\mathcal{V}}_{h,\mathbf{g}_h} \times \mathcal{V}^{\epsilon}_h$, such that

$$\begin{aligned} a(\mathbf{u}^{\epsilon}_h, \mathbf{v}_h) + b(\mathbf{v}_h, p^{\epsilon}_h) &= F(\mathbf{v}_h), \qquad && \forall \mathbf{v}_h \in \boldsymbol{\mathcal{V}}^{\epsilon}_{h,0}, \\ -b(\mathbf{u}^{\epsilon}_h, q_h) + s(p^{\epsilon}_h, q_h) &= 0, && \forall q_h \in \mathcal{V}^{\epsilon}_h. \end{aligned} \tag{3.79}$$

3.2.3 Numerical assessment of the DIBM applied to the Stokes problem

In this section we want to numerically expose the validity of the following inequality

$$\|\mathbf{u} - \mathbf{u}^{\epsilon}_B\|_{H^1(\Omega^{\epsilon}_h)} + \|p - p^{\epsilon}_B\|_{L^2(\Omega^{\epsilon}_h)} \lesssim \epsilon^{\frac{1}{2}} + \delta^{\frac{1}{2}} + \kappa^{\frac{2}{3}} + h. \tag{3.80}$$

To do this we act along two different directions. First, we consider two cases with analytic solution, one in 2D and one in 3D. The method used to solve the Stokes equations is the SIMPLE splitting method, described in Section 5.1.1. The usage of the SIMPLE method used along with non-conforming methods is exposed in Section 5.2.

In both experiments we considered the case of a uniform mesh, where $\delta \simeq \kappa \simeq h$, and the case where a local refinement of the diffuse interface region is performed, such that $\delta \simeq \kappa \simeq h^2$ (Figure 3.6, 3.12 and 3.14). In the second case, the standard order of convergence in H^1 and L^2 norms are recovered. The continuous interface width ϵ is set to be such that $\epsilon \simeq \delta$ in order to be neglected with respect to other terms.

As mentioned in Section 3.1, all the computations have been performed employing a Voronoi dual mesh of a Delaunay triangulation (i.e. connecting circumcentres of triangles with straight lines). To generate the dual mesh we relied on a tool implemented in OpenFOAM called `polyDualMesh` modified in order to use circumcentres of triangles and nodes of dual mesh instead of barycentres.

3.2.3.1 A 2D case-study: Taylor-Couette flow

For the first case we considered a Taylor-Couette flow. Let $\tilde{\Omega} = B_0(1)$, where $B_0(1)$ is the circle of radius 1 centred in (0,0), and $\Gamma = \partial B_0(\frac{1}{2})$.

Let $(\mathbf{u}, p)$ be the solution to problem

$$\begin{aligned} -\Delta\mathbf{u} + \nabla p &= \mathbf{f}, && \text{in } \tilde{\Omega}\backslash\overline{\Gamma}, \\ \nabla \cdot \mathbf{u} &= 0, && \text{in } \tilde{\Omega}\backslash\overline{\Gamma}, \\ \mathbf{u} &= \mathbf{g}, && \text{on } \Gamma, \\ \mathbf{u} &= [-y, x]^\intercal, && \text{on } \partial B_0(1), \end{aligned}$$

where $\mathbf{g} = \mathbf{0}$ on Γ and extended to $\tilde{\Omega}$ as $\tilde{\mathbf{g}} = \mathbf{0}$.

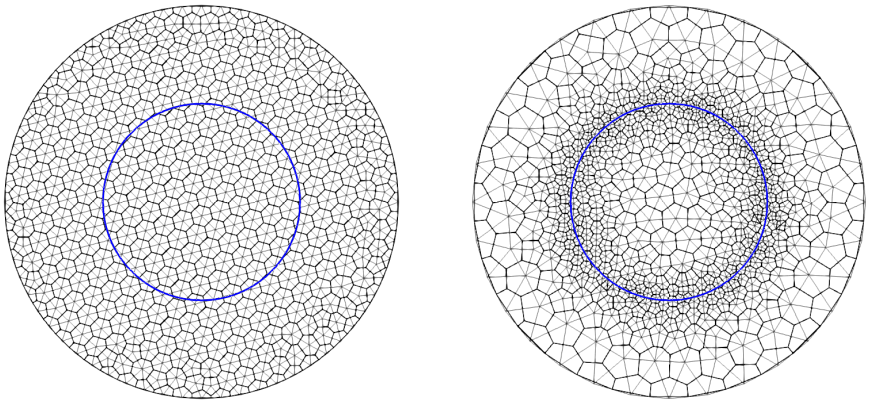


Figure 3.12: Embedded cylinder, primal and dual meshes in both uniform and refined cases.

Setting the solution equal to:

$$\begin{aligned}\mathbf{u} =& \begin{bmatrix} -\dfrac{y\left(4x^2+4y^2-1\right)}{3(x^2+y^2)} \\ \dfrac{x\left(4x^2+4y^2-1\right)}{3(x^2+y^2)} \end{bmatrix}, \\ p =& e^{-4x^2-4y^2},\end{aligned} \tag{3.81}$$

the source term is chosen accordingly:

$$\mathbf{f} = -\Delta\mathbf{u} + \nabla p, \text{ in } \tilde{\Omega}\backslash\Gamma.$$

We first let $\epsilon \to 0$ and we let vary h as

$$h = 0.075, 0.05, 0.025, 0.0075.$$

From Figure 3.13 we observe that both the L^2-norm of the pressure error and H^1-norm of velocity error decrease with order 1/2, while we recover order 1 for both quantities when performing the local mesh refinement. These rates of convergence are in agreement with (3.80).

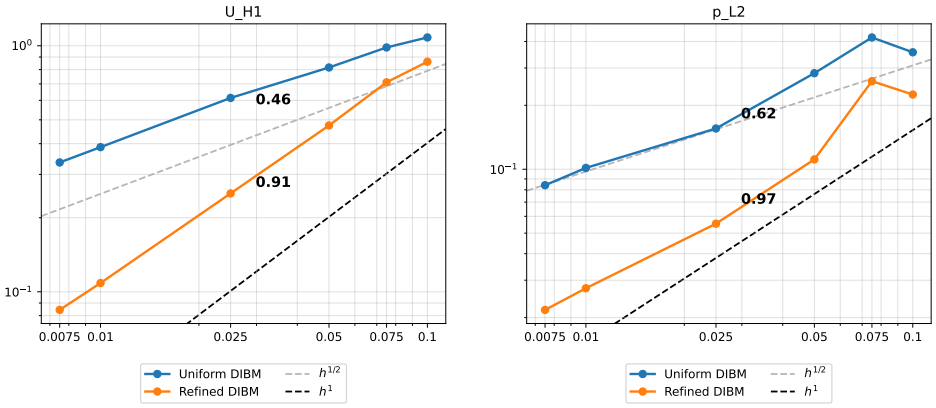


Figure 3.13: Error behaviour with respect to h (as $\epsilon \to 0$): (left) H^1-norm velocity error, (right) L^2-norm pressure error. Dash and dotted lines are theoretical convergence orders.

3.2.3.2 A 3D case-study: Stokes flow around a sphere

For the second case we considered a Stokes flow around a sphere. Let $L = \frac{3}{2}$ and $R_s = \frac{1}{2}$, $\tilde{\Omega} = (-L, L)^3$, $\Gamma = \partial B_0(R_s)$ and $\Gamma_{\text{out}} = (-L, L)^2 \times \{z = L\}$. We let also $\mathbf{U}_0 = [0, 0, U_0]^\intercal$, where $U_0 = 1$ m/s, be a uniform velocity coming towards the sphere.

Consider the following problem:

$$\begin{aligned} -\Delta \mathbf{u} + \nabla p &= \mathbf{0}, && \text{in } \tilde{\Omega}\backslash\overline{\Gamma}, \\ \nabla \cdot \mathbf{u} &= 0, && \text{in } \tilde{\Omega}\backslash\overline{\Gamma}, \\ \mathbf{u} &= \mathbf{g}, && \text{on } \Gamma, \\ \mathbf{u} &= \mathbf{u}_{\text{ex}}, && \text{on } \partial\tilde{\Omega}\backslash\Gamma_{\text{out}}, \\ -p\mathbf{n} + \nabla \mathbf{u} \cdot \mathbf{n} &= \mathbf{0}, && \text{on } \Gamma_{\text{out}}, \end{aligned}$$

where $\mathbf{g} = \mathbf{0}$ on Γ and extended to $\tilde{\Omega}$ as $\tilde{\mathbf{g}} = \mathbf{0}$.

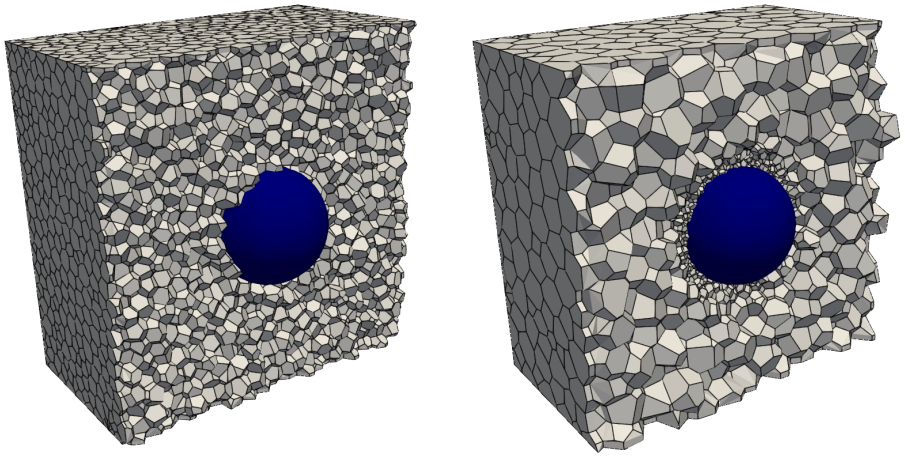


Figure 3.14: Embedded sphere and dual meshes in both uniform and refined cases.

Then, its solution $(\mathbf{u}, p)$ reads:

$$\mathbf{u} = \mathbf{u}_{\text{ex}} = \begin{bmatrix} \frac{U_0 xz\left(\frac{{R_s}^3}{2\left(x^2+y^2+z^2\right)^{3/2}} - \frac{3R_s}{2\sqrt{x^2+y^2+z^2}} + 1\right)}{x^2+y^2+z^2} + \frac{U_0 xz\left(\frac{{R_s}^3}{4\left(x^2+y^2+z^2\right)^{3/2}} + \frac{3R_s}{4\sqrt{x^2+y^2+z^2}} - 1\right)}{x^2+y^2+z^2} \\ \frac{U_0 yz\left(\frac{{R_s}^3}{2\left(x^2+y^2+z^2\right)^{3/2}} - \frac{3R_s}{2\sqrt{x^2+y^2+z^2}} + 1\right)}{x^2+y^2+z^2} + \frac{U_0 yz\left(\frac{{R_s}^3}{4\left(x^2+y^2+z^2\right)^{3/2}} + \frac{3R_s}{4\sqrt{x^2+y^2+z^2}} - 1\right)}{x^2+y^2+z^2} \\ \frac{U_0 z^2\left(\frac{{R_s}^3}{2\left(x^2+y^2+z^2\right)^{3/2}} - \frac{3R_s}{2\sqrt{x^2+y^2+z^2}} + 1\right)}{x^2+y^2+z^2} - \frac{U_0\left(x^2+y^2\right)\left(\frac{{R_s}^3}{4\left(x^2+y^2+z^2\right)^{3/2}} + \frac{3R_s}{4\sqrt{x^2+y^2+z^2}} - 1\right)}{x^2+y^2+z^2} \end{bmatrix},$$

$$p = -\frac{3R_s^3 z}{\left(x^2+y^2+z^2\right)^{\frac{3}{2}}}.$$

We first let $\epsilon \to 0$ and we let vary h as

$$h = 0.09, 0.075, 0.05, 0.025.$$

From Figure 3.15 we observe that the H^1-norm of velocity error converges with order 1/2 when using a uniform mesh and with order 1 when employing the local refinement. On the other hand, we observe convergence rates slightly higher than 1/2 and 1 for the L^2-norm of the pressure in uniform and locally refined cases, respectively. These rates of convergence are in agreement with (3.80).

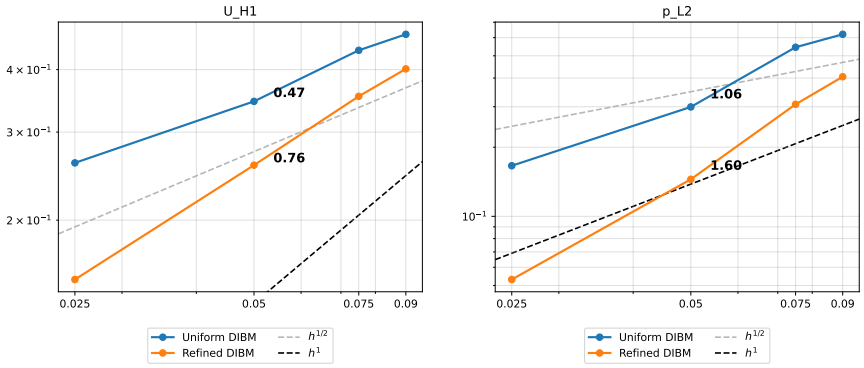


Figure 3.15: Error behaviour with respect to h (as $\epsilon \to 0$): (left) H^1-norm velocity error, (right) L^2-norm pressure error. Dash and dotted lines are theoretical convergence orders.

3.2.4 A roadmap to prove a priori error estimate for DIBM

In this section we propose a roadmap to prove theoretically the convergence rates obtained in numerical experiments of Section 3.2.3. As in Section 3.2.1, the idea of the proof is based on proving perturbation results between the solutions on different perturbation of the domain (Ω, Ω^ϵ and Ω_h^ϵ), in terms of mesh discretization and diffuse interface parameters.

Ansatz The solution $(\mathbf{u}, p) \in \boldsymbol{\mathcal{V}} \times \mathcal{Q}$ of problem (3.9) and the solution $(\mathbf{u}_h^\epsilon, p_h^\epsilon) \in \boldsymbol{\mathcal{V}}_{h,\mathbf{g}_h}^\epsilon \times \mathcal{Q}_h$ of problem (3.79) satisfy the following relationship:

$$\|\mathbf{u} - \mathbf{u}_B^\epsilon\|_{H^1(\Omega_h^\epsilon)} + \|p - p_B^\epsilon\|_{L^2(\Omega_h^\epsilon)} \lesssim \epsilon^{\frac{1}{2}} + \delta^{\frac{1}{2}} + \kappa^{\frac{2}{3}} + h.$$

where $h = \max_T\{h_T\}$, $\kappa = \max\{h_T,\, T \in \mathcal{T}_h \cap \partial S^{\epsilon+\delta}\}$, $\delta = \max\{h_T, T \in \mathcal{T}_h \cap \partial \mathcal{S}_h^\epsilon\}$.

Step 1 (FEM approximation). First prove the following convergence result for FEM solution on the discrete interface domain Ω_h^ϵ. The solution $(\mathbf{u}^{\epsilon,h}, p^{\epsilon,h}) \in \boldsymbol{\mathcal{V}} \times \mathcal{Q}$ of problem (3.78) and the solution $(\mathbf{u}_h^\epsilon, p_h^\epsilon) \in \boldsymbol{\mathcal{V}}_{h,\mathbf{g}_h}^\epsilon \times \mathcal{Q}_h$ of problem (3.79) satisfy the following relationship:

$$\left\|\mathbf{u}^{\epsilon,h} - \mathbf{u}_h^\epsilon\right\|_{H^1(\Omega_h^\epsilon)} + \left\|p^{\epsilon,h} - p_h^\epsilon\right\|_{L^2(\Omega_h^\epsilon)} \lesssim \sqrt{\delta} + \kappa^{\frac{2}{3}} + h.$$

Step 2 (Perturbation result). Prove then the following perturbation result between the continuous solution on the discrete interface domain Ω_h^ϵ and the continuous solution of the original problem (3.9).

$$\left\|\mathbf{u} - \mathbf{u}^{\epsilon,h}\right\|_{H^1(\Omega_h^\epsilon)} + \left\|p - p^{\epsilon,h}\right\|_{L^2(\Omega_h^\epsilon)} \lesssim \sqrt{\delta} + \sqrt{\epsilon} \tag{3.82}$$

where $\delta = \max\{h_T, T \in \mathcal{T}_h \cap \partial \mathcal{S}_h^\epsilon\}$.

This can be done by proving separately

$$\|\mathbf{u}-\mathbf{u}^{\epsilon}\|_{H^1(\Omega)}+\|p-p^{\epsilon}\|_{L^2(\Omega)}\leq\sqrt{\epsilon} \tag{3.83}$$

and

$$\left\|\mathbf{u}^{\epsilon}-\mathbf{u}^{\epsilon,h}\right\|_{H^1(\Omega)}+\left\|p^{\epsilon}-p^{\epsilon,h}\right\|_{L^2(\Omega)}\leq\sqrt{\delta} \tag{3.84}$$

where $(\mathbf{u}^{\epsilon},p^{\epsilon})$ is the solution of problem (3.77), and finally employing triangular inequality in the following way:

$$\begin{aligned}\left\|\mathbf{u}-\mathbf{u}^{\epsilon,h}\right\|_{H^1(\Omega_h^{\epsilon})}+\left\|p-p^{\epsilon,h}\right\|_{L^2(\Omega_h^{\epsilon})}&\leq\left\|\mathbf{u}-\mathbf{u}^{\epsilon,h}\right\|_{H^1(\Omega)}+\left\|p-p^{\epsilon,h}\right\|_{L^2(\Omega)}\\&\leq\|\mathbf{u}-\mathbf{u}^{\epsilon}\|_{H^1(\Omega)}+\left\|\mathbf{u}^{\epsilon}-\mathbf{u}^{\epsilon,h}\right\|_{H^1(\Omega)}\\&\quad+\|p-p^{\epsilon}\|_{L^2(\Omega)}+\left\|p^{\epsilon}-p^{\epsilon,h}\right\|_{L^2(\Omega)}.\end{aligned} \tag{3.85}$$

Step 3 (BM approximation). Introduce the Box method with diffuse interface (DIBM) denoting by $(\mathbf{u}_B^{\epsilon},p_B^{\epsilon})\in\boldsymbol{\mathcal{V}}_{h,\tilde{g}_h}^{\epsilon}$, the approximation obtained from applying the Box method to (3.78) (cf. (3.14)).

Recall then inequality (3.67) applied to the DIM case:

$$\|\mathbf{u}_B^{\epsilon}-\mathbf{u}_h^{\epsilon}\|_{H^1(\Omega_h^{\epsilon})}+\|p_B^{\epsilon}-p_h^{\epsilon}\|_{L^2(\Omega_h^{\epsilon})}\lesssim h^{\frac{1}{2}} \tag{3.86}$$

and apply triangular inequality

$$\|\mathbf{u}_B^{\epsilon}-\mathbf{u}\|_{H^1(\Omega_h^{\epsilon})}+\|p_B^{\epsilon}-p\|_{L^2(\Omega_h^{\epsilon})}\lesssim\epsilon^{\frac{1}{2}}+\delta^{\frac{1}{2}}+\kappa^{\frac{2}{3}}+h^{\frac{1}{2}}. \tag{3.87}$$

As for the Poisson case and as it is evident from numerical experiments (see Section 3.2.3), if a local refinement of the diffuse interface region is performed in such a way that $\delta\simeq\kappa\simeq h^2$, then the first order of convergence is recovered for pressure and velocity error.

Remark 3.8. By what was mentioned in Section 3.1.3.5, we have shown numerically that the error estimate 3.67 is suboptimal. By this fact, also the estimate of Step 3 is suboptimal. However, the numerical experiments of Sections 3.2.3.1 and 3.2.3.2 suggest that we have order 1 convergence.

Chapter 4

Extension to the Immersed Boundary method

The aim of this chapter is to introduce an Immersed Boundary method (IBM) in the context of the Finite Volume method. The IBM is here intended as an improved non-conforming approach with respect to the Diffuse Interface method, introduces in Chapter 3. Due to DIM limitations in terms of accuracy, we revisited the IBM proposed in [86] in order to develop a more accurate non-conforming method able to deal with complex geometries.

In section 4.1, we first introduce some concept and notation on generic polyhedral meshes. Then, in section 4.2, we generalize the concept of Diffuse Interface method to the one of the Immersed Boundary method, describing in details its way of imposing non-conforming boundary conditions. Finally, in 4.7 we report several test cases in which we compare accuracy of conforming, DIM and IBM approaches, showing the numerical advantages of IBM against DIM.

4.1 Preliminaries

We first define Finite Volume mesh as a generic polyhedral tessellation of the physical domain.

Definition 4.1.1 (Finite volume mesh)**.** *Let $\Omega \subset \mathbb{R}^d$, where $d = 3$, be a Lipschitz bounded domain. Let $\mathcal{T}_h$ be a polyhedral tessellation of Ω.*

1. *Each polyhedric cell of $\mathcal{T}_h$ is denoted by K_i and is such that $\Omega = \bigcup K_i$ and $K_i \cap K_j = \emptyset$, $\forall K_i, K_j \in \mathcal{T}_h$. Let $h_i = diam(K_i)$ be the diameter of K_i and $h = \max_{K_i \in \mathcal{T}_h} \{h_i\}$. Let $\mathbf{k}_i$ and $|K_i|$ be the barycentre and the volume of cell K_i, respectively.*

2. *Let $\mathcal{F}_h$ be the set of faces F_i of $\mathcal{T}_h$. Let $\mathcal{F}_h = \mathcal{F}_h^o \cup \mathcal{F}_h^\partial$, where $\mathcal{F}_h^o$ is the set of internal faces and $\mathcal{F}_h^\partial$ is the set of boundary faces. Let $\mathbf{f}_i$ be the barycentre of face F_i and $|F_i|$ be its area.*

3. *Let $\mathcal{E}_h$ be the set of edges E_i of $\mathcal{T}_h$. Let $\mathbf{e}_i$ be the midpoint of edge E_i and $|E|_i$ be its length.*

4. Let $\mathcal{P}_h$ be the set of vertices $\mathbf{p}_i$ of $\mathcal{T}_h$.

Then, we describe the notation for the mesh elements connectivity.

Definition 4.1.2 (Mesh connectivity)**.** *Consider a cell K_i. Let $\mathcal{F}_i = \{F \in \mathcal{F}_h : K_i \cap F \neq \emptyset\}$ be the set of faces of cell K_i. Moreover, denote by F_{ij} the face shared by cell K_i and K_j and denote by F_i^∂ each face of K_i living on $\partial\Omega$. Let also $\mathcal{K}_i$ be the set of cells adjacent to cell K_i and let, analogously, $\mathcal{E}_i$ and $\mathcal{P}_i$ as the sets of edges and vertices of K_i, respectively.*

Then, for each face F_{ij} we associate a unit normal vector $\mathbf{n}_{ij}$, directed from cell i to cell j, a distance h_{ij} between centres $\mathbf{k}_i$ and $\mathbf{k}_j$, and a *diamond* D_{ij} as the polyhedron formed by the face vertices and the correspondent cell barycentres $\mathbf{k}_i$ and $\mathbf{k}_j$. Moreover, let D_{ij}^i, D_{ij}^j be the parts of the diamond D_{ij} belonging to cells K_i, K_j, respectively.

4.2 The weighted least-squares IBM (WLS-IBM)

Although local refinement allows us to recover the original convergence rate of a conforming method using the DIM, performing such a deep local mesh adaptation can be a very expensive procedure, especially in complex applications.

Hence, we consider another non-conforming approach, based on the Immersed Boundary method (IBM), that can recover the conforming convergence rate avoiding the local refinement strategy.

In particular, we considered the IBM described in [86] as a starting point. The IBM has been implemented and tested as an alternative to the Diffuse Interface method (DIM), since it should be able to guarantee a higher order of accuracy with respect to the DIM.

Let $\tilde{\Omega}$ be a hold-all domain such that $\Omega \subset \tilde{\Omega}$. Let Σ be the surface triangulation of a closed manifold, such that $\Sigma \cap \tilde{\Omega} \neq \emptyset$, representing the immersed boundary . Denote with T_i a triangle of Σ and with $|T_i|$, $\mathbf{t}_i$ the area and the barycentre of the i-th triangle, respectively. We denote with $\overline{\Sigma}$ the volume enclosed by the surface Σ.

For each cell K_i, we define the *cell-to-cell stencil* set, namely the set of cells that share at least a face with K_i. With a recursive expression, the extended stencil of level c, i.e. the set of cells connected to K_i through c connectivity steps, can be defined as:

$$\begin{aligned}
&\mathcal{K}_i^0 = \{K_i\},\\
&\mathcal{K}_i^1 = \{K_j \in \mathcal{T}_h : K_i \cap \mathcal{K}_i^0 = F_{ij}, \text{ for some } F_{ij} \in \mathcal{F}_i\} \cup \mathcal{K}_i^0,\\
&\quad\vdots\\
&\mathcal{K}_i^c = \{K_j \in \mathcal{T}_h : K_i \cap \mathcal{K}_i^{c-1} = F_{ij}, \text{ for some } F_{ij} \in \mathcal{F}_i\} \cup \bigcup_{n=0}^{c-1} \mathcal{K}_i^n.
\end{aligned} \tag{4.1}$$

Notice that $\mathcal{K}_i^1$ corresponds to the set defined in Definition 4.1.2. On the other hand, the *point-to-cell stencil* of level c can be defined with a similar recursive relation. Here the con-

nectivity is given by having a common point between two cells:

$$\begin{aligned}
&\mathcal{K}_i^0 = \{K_i\},\\
&\mathcal{K}_i^1 = \{K_j \in \mathcal{T}_h : K_i \cap \mathcal{K}_i^0 \neq \emptyset\} \cup \mathcal{K}_i^0,\\
&\quad\vdots\\
&\mathcal{K}_i^c = \{K_j \in \mathcal{T}_h : K_i \cap \mathcal{K}_i^{c-1} \neq \emptyset\} \cup \bigcup_{n=0}^{c-1} \mathcal{K}_i^n.
\end{aligned} \tag{4.2}$$

In general, one can choose at will either the cell-to-cell or the point-to-cell stencil. Unless specified, for now on we will consider point-to-cell stencil. An example for both choices is represented in Figure 4.1.

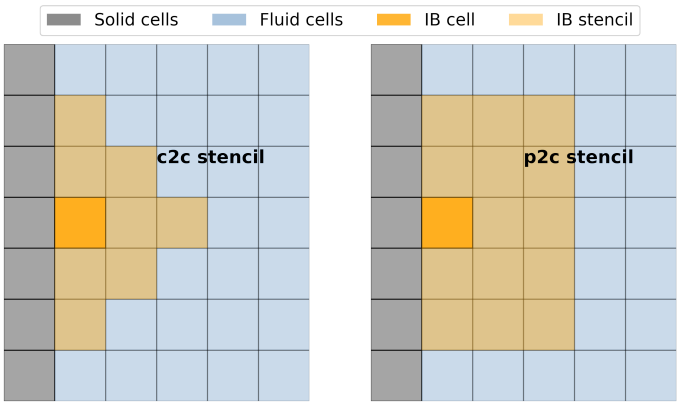


Figure 4.1: Representation of cell-to-cell (c2c) and point-to-cell (p2c) stencils of level 2.

Without loss of generality, suppose the flow to be external to the immersed surface. The procedure of a generic Immersed Boundary method (IBM) is resumed by the following steps:

1. generate the triangulated surface Σ (typically a STL file) representing the immersed object;
2. use the surface to divide the mesh in three regions:
   - *solid set*: cells with barycentre inside the surface, representing the immersed object;
   - *IB set*: immersed boundary (IB) cells just outside of solid set, that is non-solid cells which have at least one solid cell as a neighbour;
   - *fluid set*: the remaining cells, representing the fluid domain;
3. generate the IB cell stencil $\mathcal{S}$, for each IB cell, that is a set of mesh cells which are selected within certain criteria;
4. generate the IB approximator on each stencil (i.e. interpolation matrices);

5. at each time step and for each field, using the values of the actual solution evaluated on the stencil of an IB cell, evaluate the approximator function in the IB cell barycentre and impose the value on the solution.

Mesh subdivision First, the mesh is subdivided in three regions based on the position of the mesh elements with respect to the triangulated surface Σ. We denote the three regions Γ_S, Γ_{IB} and Γ_F as the solid, immersed boundary and fluid (cells), respectively. They are defined as follows:

$$\begin{aligned}\Gamma_S &= \{K_i \in \mathcal{T}_h : \mathbf{k}_i \in \overline{\Sigma}\},\\ \Gamma_{\text{IB}} &= \{K_i \in \mathcal{T}_h \backslash \Gamma_S : K_i \in \mathcal{K}_j^1, \text{ for some } K_j \in \Gamma_S\},\\ \Gamma_F &= \{K_i \in \mathcal{T}_h \backslash (\Gamma_S \cup \Gamma_{\text{IB}})\}.\end{aligned} \tag{4.3}$$

We also define the projections on Σ of the cell centres of Γ_{IB} and we denote them as *IB points* $\mathcal{P}_{\text{IB}} = \{\mathbf{p}_{\text{IB},i}\}$, where i is the index of the associated IB cells. Moreover, we denote with $\mathbf{n}_{\text{IB}}$ the normal unit vectors in the IB points directed inwards the solid set, as shown in Figure 4.2.

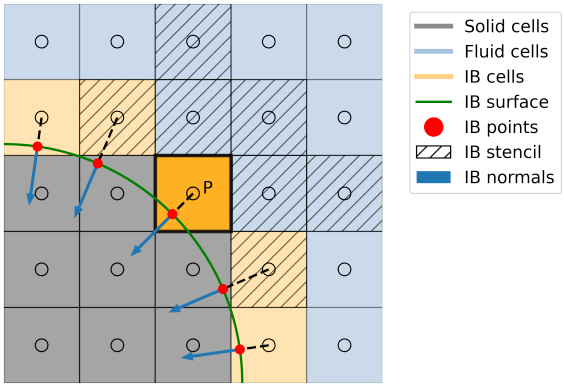


Figure 4.2: Schematics of IBM mesh elements.

Finally, we define the solid, IB and fluid face sets and we denote them with the symbols γ_S, γ_{IB} and γ_F:

$$\begin{aligned}\gamma_S &= \{F_{ij} \in \mathcal{F} : K_i \in \Gamma_S \wedge K_j \in \Gamma_S\},\\ \gamma_{\text{IB}} &= \{F_{ij} \in \mathcal{F} : K_i \in \Gamma_{\text{IB}} \nleftrightarrow K_j \in \Gamma_{\text{IB}}\},\\ \gamma_F &= \{F_{ij} \in \mathcal{F} : K_i \in \Gamma_F \wedge K_j \in \Gamma_F\}\end{aligned} \tag{4.4}$$

where $\nleftrightarrow$ is the logical operator representing the exclusive or.

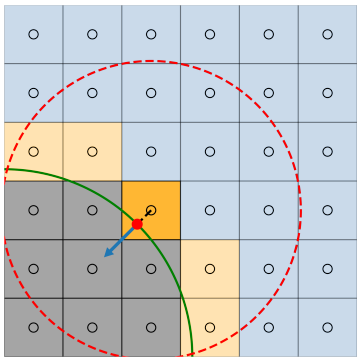
Extended stencil Consider an IB cell $K_i \in \Gamma_{\text{IB}}$. We construct its extended stencil $\mathcal{S}_i$ opportunely collecting some neighbouring cells centres among IB and fluid sets. In particular, we choose cells whose barycentres simultaneously satisfy the following three criteria:

- *spatial distance*: if centre-to-centre distance with respect to the IB cell is within a certain bound (Figure 4.3a);

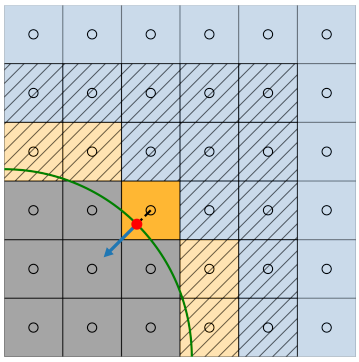
- *connectivity distance* c: if the cell belongs to the stencil of level c $\mathcal{K}_i^c$ (definitions (4.1) and (4.2), Figure 4.3b);

- *field of view*: if cell centre is within a certain angle with respect the the IB normal unit vector (Figure 4.3c).

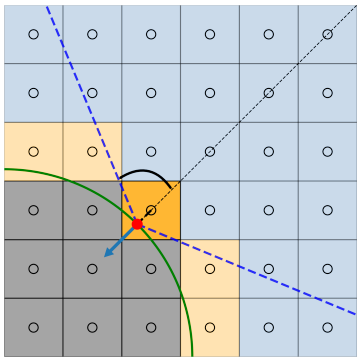
Examples of level 2 stencils can be seen in Figure 4.1 and an example of the application of the three criteria can be seen in Figure 4.3d.



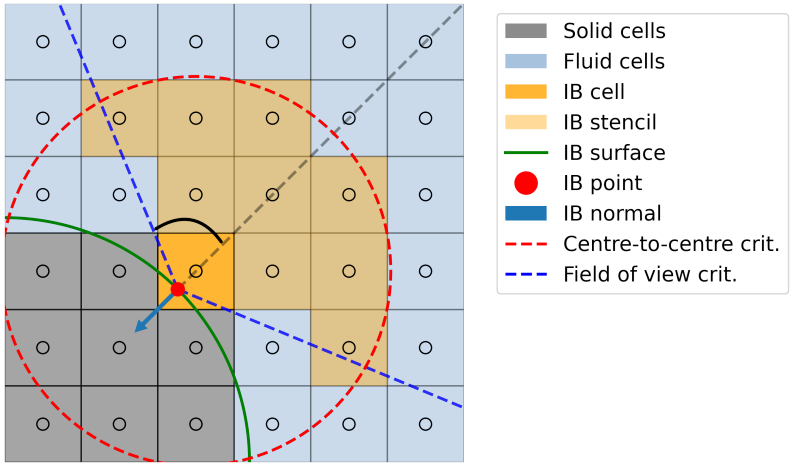
(a) Spatial distance criterion, around two times the IB cell diameter.



(b) Connectivity distance criterion: an example of a level 2 point-to-cell stencil.



(c) Field of view criterion with an angle of 80°.



(d) Intersection of all the criteria to select the final stencil.

Figure 4.3: Example of filtering criteria to select the extended stencil of an IB cell. Orange cells represent the final stencil for the least squares interpolation. Stencil is chosen within the intersection of the red circle and the two blue lines and at a maximum connectivity distance of two cells.

We define the extended stencil $\mathcal{S}_i$ of an IB cell $K_i \in \Gamma_{\text{IB}}$ as the set of cells satisfying the

three criteria:

$$\mathcal{S}_i = \Bigg\{ K_j \notin \Gamma_S : \mathbf{k}_j \in \mathbb{R}^3 : j \neq i, K_j \in \mathcal{K}_i^{\bar{c}} \cap (\Gamma_{\text{IB}} \cup \Gamma_F), \text{ (connectivity)}, \\ \|\mathbf{k}_i - \mathbf{k}_j\|_2 \leq \bar{r}, \text{ (spatial distance)}, \\ -\mathbf{n}_i \cdot \frac{(\mathbf{k}_i - \mathbf{k}_j)}{\|\mathbf{k}_i - \mathbf{p}_i\|_2} \leq \cos(\bar{\theta}), \text{ (field of view)} \Bigg\}, \tag{4.5}$$

where $\bar{c}$ is the maximum connectivity level, $\bar{r}$ is the limit distance and $\bar{\theta}$ is the field of view angle. These constraints are defined a priori.

IB value approximator First define the *IB value* $\mathbf{g}_{\text{IB}} = [g_i]$ as the vector of values of the immersed boundary condition evaluated at IB points associated to IB cell $K_i \in \Gamma_{\text{IB}}$.

We define the WLS approximant as a polynomial of degree p. Given the polynomial basis of degree p

$$\mathbb{P}_{\mathbf{x}} = [1, x, y, z, \ldots, x^p, y^p, z^p]^\intercal \tag{4.6}$$

and a vector of N_{coeffs} unknown coefficients

$$\boldsymbol{\beta} = [\beta_0, \ldots, \beta_{N_{coeffs}}]^\intercal, \tag{4.7}$$

we define the approximation function f_{IB} as

$$f_{\text{IB}}(\mathbf{x}) = \mathbb{P}_{\mathbf{x}}^\intercal \boldsymbol{\beta},$$

where $\mathbf{x} = [x, y, z]^\intercal$.

The WLS approximation is performed determining a set of observations of the solution and associating the correspondent weights. In particular, the observations are the solution values in the cells of the extended stencil plus the value of the immersed boundary condition evaluated on the correspondent IB point.

Now, consider the stencil $\mathcal{S}_i$ of cardinality n. Let u_j denote the FVM solution on the j-th cell of stencil $\mathcal{S}_i$. For each observation define also the weights $w_j \in [0, 1], j = 0, \ldots, n$ associated to the observations points, where 0 is the IB point as a convention. We want to determine $\boldsymbol{\beta}^*$ such that:

$$\boldsymbol{\beta}^* = \underset{\boldsymbol{\beta}}{\text{argmin}} \left(w_0 \left| g_{\text{IB},i} - \mathbb{P}_{\mathbf{p}_{\text{IB},i}}^\intercal \boldsymbol{\beta} \right|^2 + \sum_{j \in \mathcal{S}_i} w_j \left| u_j - \mathbb{P}_{\mathbf{k}_j}^\intercal \boldsymbol{\beta} \right|^2 \right) = \underset{\boldsymbol{\beta}}{\text{argmin}} \left\| \mathrm{W}^{\frac{1}{2}} \left(\mathbf{U}_{\text{IB}} - \mathrm{A}\boldsymbol{\beta} \right) \right\|^2 \tag{4.8}$$

where W is a square diagonal matrix with $\mathrm{W}_{jj} = w_j$. $\mathbf{U}_{\text{IB}}$ is the vector collecting all the of observations, and A is the matrix that has on the j-th row equation (4.6) evaluated in the j-th observation point.

When Dirichlet condition on the immersed boundary are considered, the matrix A will be of the following form

$$\mathrm{A} = \begin{bmatrix} 1 & x_0 & y_0 & z_0 & \dots & x_0^p & y_0^p & z_0^p \\ 1 & x_1 & y_1 & z_1 & \dots & x_1^p & y_1^p & z_1^p \\ \vdots & & & & & & & \vdots \\ 1 & x_n & y_n & z_n & \dots & x_n^p & y_n^p & z_n^p \end{bmatrix}. \tag{4.9}$$

In the case of a Neumann condition on the immersed boundary ($\nabla u \cdot \mathbf{n} = g_N$) we could just consider the approximation function

$$f_{\mathrm{IB}} = \beta_0 + \beta_1 x + \cdots + \beta_{N_{coeffs}} z^p$$

and, recalling that the 0-th point is associated to the IB condition, compute the normal component of its gradient in $\mathbf{p}_{\mathrm{IB},i}$ as:

$$\nabla f_{\mathrm{IB}}(\mathbf{x}) \cdot \mathbf{n} = \beta_1 n_x + \cdots + p\beta_{N_{coeffs}} z^{p-1} n_z.$$

Then, the first row of the matrix A becomes

$$\mathrm{A} = \begin{bmatrix} 0 & n_x & n_y & n_z & \dots & p x_0^{p-1} n_x & p y_0^{p-1} n_y & p z_0^{p-1} n_z \\ \vdots & & & & & & & \vdots \end{bmatrix}, \tag{4.10}$$

while in the $\mathbf{U}_{\mathrm{IB}}$ vector we set the first element value to the condition g_N. In this way we can impose easily different types of boundary conditions, for instance the incompressibility of the field in the IB cell ($\nabla \cdot f_{\mathrm{IB}}(\mathbf{x})|_{K_i} = 0$), see [101].

Finally, let S_{IB} be the interpolation operator that corrects the solution field on the IB and solid sets. We define S_{IB} as a linear combination:

$$\mathbf{U}^{\mathrm{corr}} = S_{\mathrm{IB}}(\mathbf{g}, \mathbf{U}) = \begin{cases} s_{\mathrm{IB}} g_{\mathrm{IB},i} + \sum\limits_{K_j \in \mathcal{S}_i} s_j \mathbf{U}_j, & \text{if } K_i \in \Gamma_{\mathrm{IB}}, \\ \mathbf{U}_i, & \text{if } K_i \in \Gamma_F, \\ \mathbf{g}_i, & \text{if } K_i \in \Gamma_S, \end{cases} \tag{4.11}$$

where $\mathbf{U}^{\mathrm{corr}}$ is the corrected solution vector, $\mathbf{U}$ is the current solution vector, $\mathbf{s}_i = [s_{i,k}], k = 1, \dots, \mathrm{card}(\mathcal{S}_i)$ are the linear combination coefficients computed by $\mathbf{s}_i = \mathbb{P}_{\mathbf{k}_i}^{\intercal} \boldsymbol{\beta}^*$ and $\mathbf{g}$ is such that

$$\mathbf{g} = \begin{cases} \mathbf{g}_{\mathrm{IB},i} & \text{if } K_i \in \Gamma_{\mathrm{IB}}, \\ 0, & \text{if } K_i \in \Gamma_F, \\ \mathbf{g}_{S,i}, & \text{if } K_i \in \Gamma_S, \end{cases}$$

where $\mathbf{g}_S$ is the vector of velocity values on Γ_S.

It is worth noticing that, being S_{IB} al linear combination, it can be represented by two matrices S_g and S, such that:

$$\mathbf{U}^{\mathrm{corr}} = \mathrm{S}_g \mathbf{g} + \mathrm{S}\mathbf{U}. \tag{4.12}$$

Remark 4.1. For all the derived fields, such as gradients, shear rate, viscosity, etc., we impose the following condition on the IB point:

$$\mathbf{n} \cdot \nabla\nabla f_{\text{IB}} \cdot \mathbf{n} = 0.$$

This conditions comes form the fact that any derived quantity is assumed to vary linearly inside a mesh cell, thus their Hessian must be zero.

Least-squares weights Different weighting functions are used for Dirichlet and Neumann conditions even if they both depend on the distance from IB point [86].

$$\begin{aligned} \text{Dirichlet}: \quad & w_i = \frac{1}{2}\left[1 + \cos\left(\frac{\pi}{c}\frac{r_i}{r_{max}}\right)\right], \\ \text{Neumann}: \quad & w_i = 1 - \frac{r_i}{c r_{max}}, \end{aligned} \tag{4.13}$$

where r_i is the distance of the i-th stencil point form the IB cell centre and $c > 1$ is a factor to avoid null weights.

4.3 Time dependent IBM

As mentioned before in this work, mixing processes are naturally time dependent because of the kinematics involved. Hence, we need to adopt a strategy to move the IB surface and consequently compute solution based on the motion dynamics. The idea is not different from the one exposed in previous section. The only difference resides in the velocity value that we impose on the IB surface Σ.

In the steady-state case the value of $\mathbf{g}_{\text{IB}}$ is prescribed a priori. Equivalently, if the body kinematics is known and the velocity can be described analytically, at each time t we impose exactly on the surfaces the boundary value $\mathbf{g}_{\text{IB}}(t)$.

On the other hand, when we do not know a priori the motion of a body, we are able to estimate its velocity by empoying a BDF2 differencing scheme. Set time at instant t^n. Let $\mathcal{P}_{\text{IB}}^n$ be the set of points of triangulation S_{IB}. Let $\mathcal{U}_{\text{IB}}^n$ be the set of velocity values evaluated on each point $\mathbf{p} \in \mathcal{P}_{\text{IB}}$. Then, we compute $\mathcal{U}_{\text{IB}}^{n+1}$ element-wise as:

$$\mathcal{U}_{\text{IB}}^{n+1} = \frac{\mathcal{P}_{\text{IB}}^n - 4\mathcal{P}_{\text{IB}}^{n+\frac{1}{2}} + 3\mathcal{P}_{\text{IB}}^{n+1}}{\Delta t}. \tag{4.14}$$

This formula ensures enough accuracy, given a reasonable time step. If the motion is prescribed, the accuracy of $\mathcal{U}_{\text{IB}}^{n+1}$ computation can be increased at will substituting formula (4.14) with a more accurate one.

In both cases, according to the experience developed on numerical experiments, an acceptable constraint on time step when moving the IB surface is that, between two time instants, Γ_S does not change of more that one cell, so

$$\left\|\mathcal{P}_{\text{IB}}^{n+1} - \mathcal{P}_{\text{IB}}^n\right\|_\infty \lesssim h.$$

4.4 Extension to multiple IB surfaces

There exist problems that need the application of the Immersed Boundary method to more than one geometry because multiple elements of the device move passing one close to the others, or because the geometries in object are modular and composed parts that present different shapes. This is the case of the the twin-screw extruder (c.f. Section 6.2 and Figures 1.2 and 6.14) where there are two co-rotating elements, both composed by multiple modules of different shapes (see transport and kneading elements).

Therefore, to handle these type of problems we have to be able to consider the interaction between multiple immersed boundaries. Here, we describe the strategy that we developed to consider every IB surface as an independent geometrical element. This feature allows to simulate potentially any combination of screw elements, that makes our tool powerful in dealing with complex geometries typically present in industrial mixing processes.

Let $\{\Sigma_i\}, i = 1, \dots, N_{IB}$ be a set of IB surfaces. If $N_{IB} > 1$, then some quantities defined in Section 4.2 have to be redefined in order to describe the multiple IB framework. First, the global mesh subsets, that now depend on the ensemble of surfaces:

$$
\begin{aligned}
\Gamma_S &= \bigcup_{i=0}^{N_{IB}} \Gamma_{S,i}, \\
\Gamma_{\text{IB}} &= \{K_i \in \mathcal{T}_h \backslash \Gamma_S : K_i \in \Gamma_{\text{IB},j} \wedge (K_i \notin \Gamma_{\text{IB},k}, \forall k \neq j), j = 1, \dots, N_{IB}\}, \\
\Gamma_{\text{SIB}} &= \{K_i \in \mathcal{T}_h \backslash \Gamma_S : K_i \in \Gamma_{\text{IB},j} \wedge (K_i \in \Gamma_{\text{IB},k}, \text{ for at least one } k \neq j), j = 1, \dots, N_{IB}\}, \\
\Gamma_F &= \{K_i \in \mathcal{T}_h \backslash (\Gamma_S \cup \Gamma_{\text{IB}} \cup \Gamma_{\text{SIB}})\},
\end{aligned}
\tag{4.15}
$$

where Γ_{SIB} is the set of "shared" IB cells, i.e. cells that are IB cells for more than one immersed surface.

Moreover, the IB cells stencils and the WLS interpolation are the most delicate parts of the IB condition imposition when dealing with multiple geometries. We have to specialize treatments for stencils of cells that are shared between or near two or more IB surfaces.

When an IB cell is between many IB surfaces, it can happen that a stencil has solid cells relative to other IB surfaces. If this is the case the WLS interpolation would not be reliable because it would depend on prescribed values of solid cells, that are not part of the physical domain. Hence, we eliminate the solid cells from the stencil

$$
\tilde{\mathcal{S}}_i = \mathcal{S}_i \backslash \{K_i \in \Gamma_{S,j}, j \neq i\},
$$

ensuring that enough points remain to perform the WLS interpolation. Otherwise, we add to the stencil the IB points of the other IB surfaces, looking for these IB points among the IB cells of other surfaces. This mostly happens when dealing with adjacent IB surfaces, i.e. at the interface between two screw modules, or near IB surfaces. An example is sketched in Figure 4.4.

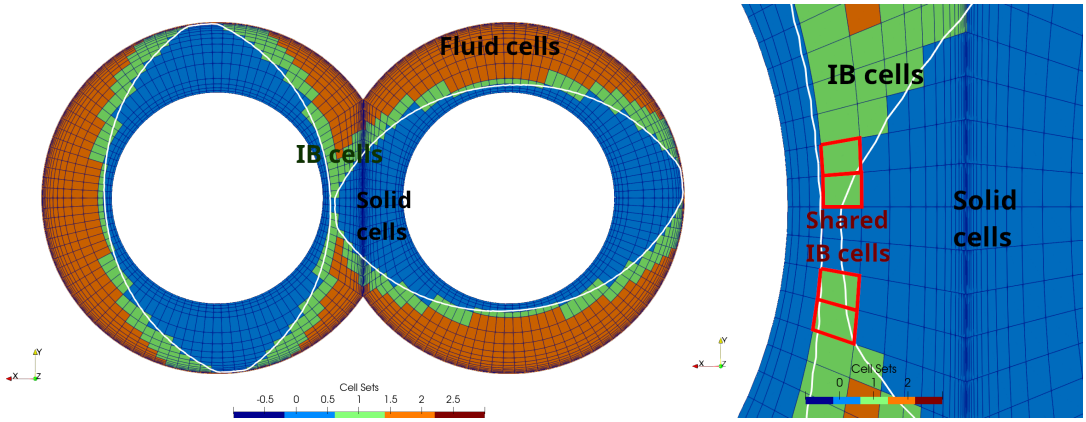


Figure 4.4: Demonstrative example of a grid presenting shared IB cells between the screws of a TSE. Shared IB cells are highlighted.

4.5 Issues on anisotropic mesh

The first testing on realistic geometries with the original IBM implementation [86], was performed on the single-screw extruder problem (see Section 6.1 and Figures 1.1 and 6.1). In order to simulate the flow in the SSE we had to adopt anisotropic computational grids (Figure 4.5), to avoid an excessive number of degrees of freedom to explode and to capture the flow inside the gap between the screw and the external barrel.

However, the original implementation of the IBM suffered the presence of such stretched elements and, in general, with anisotropic grids. In our experience, numerical instabilities arouse because of the combination between the usage of IBM, anisotropic grids and strong pressure gradients, where there are sharp edges or the flow is perpendicular to the wall. These instabilities were due to the imposition of the Neumann pressure condition in the SIMPLE algorithm (c.f. 5).

The original Neumann condition for pressure The idea behind the original implementation of a Neumann boundary condition was to impose a "conforming" condition on a staircase boundary. The IB cells are initially removed from the computational domain and the Laplacian pressure equation with Neumann boundary conditions is solved on the domain subset Γ_F (blue region in Figure 4.6). Finally, the IB cells pressure values are corrected using the WLS approximation imposing $\nabla p \cdot \mathbf{n} = 0$ on the immersed surface (equation (4.10)).

The deletion of IB cells from the computational domain is performed in the following way. Consider the i-th row of the pressure Laplacian algebraic problem: let $K_i \in \mathcal{T}_h$, $a_j \in \mathbb{R}$, $j \in \mathcal{K}_i \cup \{K_i\}$,

$$-a_i p_i + \sum_{j \in \mathcal{K}_i} a_j p_j = f_i,$$

where f_i is the source term. Now, for all the faces that are between an IB cell and a fluid or solid cell $F_{ij} \in \mathcal{F}_i \cap \gamma_{\text{IB}}$, we impose the constraint that $p_i = p_j$. In this way, we are imposing

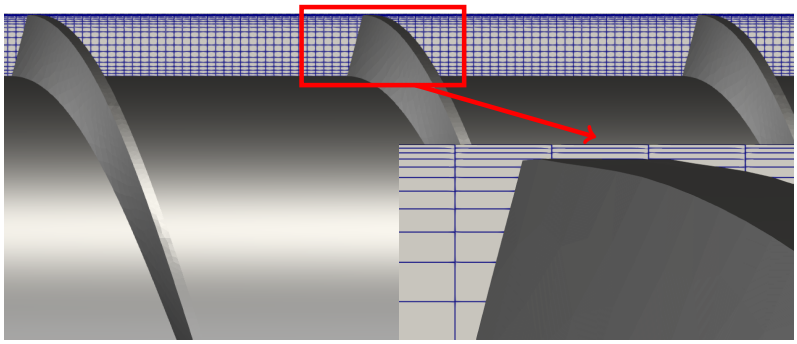


Figure 4.5: Example of a SSE coarse computational mesh. Here the external barrel and the shaft are approximated with a conforming boundary, while the screw teeth are approximated using immersed boundaries. The figure represents the metering section of the grid of a SSE with a zoom lens on the grid cells between the gap. In this example the grading has been computed to have three elements within the gap.

a homogeneous Neumann condition of face F_{ij} because

$$p_j = p_i + h_{ij}\nabla^{\perp}_{h,ij}p_h,$$

where $\nabla^{\perp}_{h,ij}$ is defined in equation (A.5), meaning

$$p_i = p_j \quad \Longrightarrow \quad \nabla^{\perp}_{h,ij}p_h = 0.$$

Making this on both sides of an IB cell, decouples the IB region from fluid and solid regions. Then, let $K_i \in \Gamma_{\text{IB}}$, $F_{ij} \in \mathcal{F}_i \cap \gamma_{\text{IB}}$,

$$\begin{aligned}(a_i - a_j)p_j + \sum_{k\in\mathcal{K}_i, k\neq j} a_k p_k &= f_i,\\ p_i &= \bar{p}_i,\end{aligned} \tag{4.16}$$

where $\bar{p}_i$ is some value decided a priori. Following the above procedure coincided in solving a Neumann homogeneous problem on a staircase domain: looking at Figure 4.6, it is easy to notice that, in this way, we are committing a consistency error because we are computing a pressure field that satisfies a zero normal gradient in two different points: the set of mesh faces $\gamma_{\text{IB}} \cap \gamma_S$ and the IB points. Moreover, applying this procedure we are also performing a poor approximation of pressure gradient near immersed walls because we are smoothing normal gradients. Of course, situation gets worse when dealing with very anisotropic meshes and with sharp edges, where pressure singularities are expected (Figure 4.6).

As mentioned, when trying to apply this approach to the SSE case-study we could not reach the end of the simulation, because we were forced to use anisotropic meshes to limit

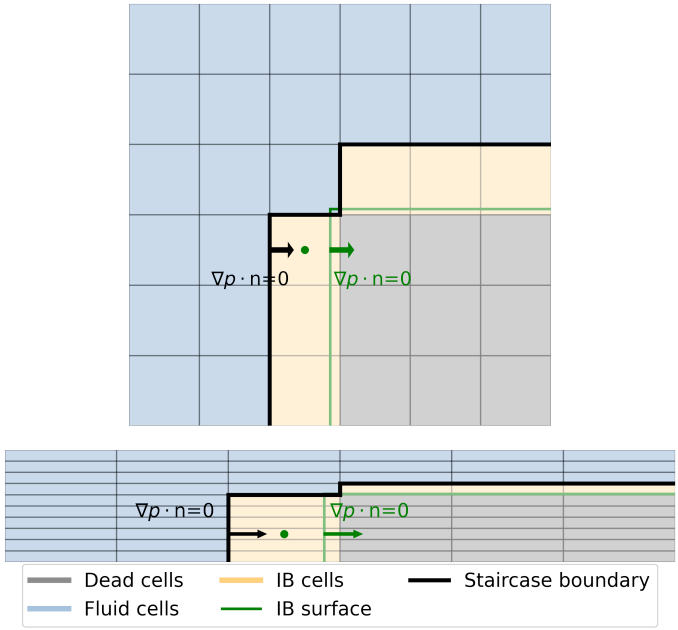


Figure 4.6: Up: scheme of where the Neumann boundary condition is imposed using the original implementation of IBM. Down: scheme of how the inconsistency grows when dealing with anisotropic grids

the number of cells. Unfortunately, it is unaffordable to use a uniform grid for this kind of test cases.

To overcome this issue, we have to compute a pressure field that is consistent with the fact that we are imposing velocity in a non-conforming way. For this reason we employ the strategy that will be described in details in Section 5.2, to include the IBM in a splitting algorithm like the SIMPLE. In this way, we do overcome the stability issue that arise when dealing with anisotropic meshes.

4.6 Parallelization of IBM for large scale problems

The IBM procedure explained in the above sections can be applied also to problems that require a great number of degrees of freedom without significantly loosing computational performance.

In general, when dealing with large scale problems in FVM, we employ a domain decomposition and then each core, or processor, solves the problem and has direct access to data only on a specific part of the decomposed domain. The difficulty of this approach is to impose opportune boundary conditions between each domain and to perform in an efficient way the intra-core communication of geometrical and solution data.

In the framework of IBM, we have to deal with cases where the embedded surface crosses a processor boundary. This is a critical aspect, as there exist an infinite number of configu-

rations of IB cells, stencils and processor boundaries, depending on problem complexity. In general, what is critical, is that, when an IB cell has a processor-local stencil with a number of points less than the number of coefficients, the WLS interpolation is impossible to be performed.

This imply the usage of specific strategies to deal with this situation. What is done is to establish whether an IB cell is near to a processor boundary: either one of its face or some face of cells in its stencil is on the boundary. Then, we communicate these IB cells and their cell centres between processors and look for the stencil on neighbouring processors. Once the stencil is built on other processors, addressing and connectivity information are communicated and stored. Then, during the computations, when the stencil values are required, they are retrieved and communicated between the various processors.

In this work, the above procedure has been optimized in order to minimize intra-processor communication of data, relative to IB information, by limiting the parallel communications. The idea is to build a communication map: each processor knows at which processors it has to send data and from which processor it has to receive data.

We set in processor n_0. For each processor $n \neq n_0$, we check for each IB cell in n_0 what processors have to be interrogated in order to retrieve data relative to the IB cell stencil. We gather in a map all the ranks of these processors. This map will indicate to which processor n_0 have to send data. The is then communicated to other processors to tell them from which rank they have to receive data.

This procedure prevents far processors domains to exchange data, avoiding the communication of useless information. However, despite the implementation of the above procedure, from this application, we could not expect to have the same parallel performance of a conforming method. This will be verified by numerical examples in the results presented in Chapter 6.

4.7 Numerical estimation of IBM accuracy

In this section we report the simulation campaign that has been carried out to estimate the convergence rate of non-conforming methods, DIM and IBM, on several test cases using FVM. All the results are then compared with the ones of conforming FVM.

The convergence assessment of the former methods is performed on several test cases: a Poisson problem, an advection-diffusion problem, a Stokes problem with a Newtonian fluid, a Navier-Stokes problem with a non-Newtonian fluid and finally a case of interest, i.e. a Navier-Stokes problem with a non-Newtonian fluid with temperature dependent viscosity. While for the first cases we consider simple geometries, for the latter we consider a more complex geometry of a single-screw extruder [92].

Fluid problems in this section will be solved with PIMPLE and SIMPLE family of algorithms. The description and the analysis of these algorithms will be postponed to Chapter 5.

For each problem we will report the error computed for different mesh sizes and the convergence rate obtained by employing four different methods: the conforming method, the IBM and the DIM. For the DIM we generated two meshes: one uniform and the other with

a local mesh refinement, in order to verify the validity of the results presented in Section 3.2.

For the error computation, we refer to Appendix A, in which we have reported a brief review of the classical Finite Volume method for advection-diffusion problems. In this review, we also reported the correspondent convergence analysis, where we exploit the influence on the global error of consistency errors of FVM numerical schemes that are employed to approximate face fluxes. We recall the error estimate for scalar problems discretized by the FVM (c.f. Theorem A.4.1).

For the sake of clarity, let u be the continuous solution, $\Pi^0 u$ be its L^2 piecewise constant projection on mesh cells and u_h be the piecewise constant FVM solution. The error estimate for the FVM discretization of a generic scalar problem reads

$$\begin{aligned} \|u - u_h\|_{L^2} + \left|\Pi^0 u - u_h\right|_* \lesssim & h, \\ \left|\Pi^0 u - u_h\right|_* \lesssim & h^p, \end{aligned} \tag{4.17}$$

where p is the global consistency order of the FVM (c.f. A.4.2) and

$$|v_h|_* = \left(\sum_{F_{ij} \in \mathcal{F}_h} h_{ij} \int_{F_{ij}} \left| \frac{v_i - v_j}{h_{ij}} \right|^2 \mathrm{d}s \right)^{\frac{1}{2}}.$$

Estimate (4.17) splits the error between the FVM solution and the continuous one in two contributions. The first one is relative to the interpolation error, in which we compare the continuous solution to the piecewise constant FV solution. The second one is a discrete H^1-norm in which the L^2-projection of the continuous solution and the FV solution face fluxes are compared.

The value of p in the above estimate is typically, i.e. using standard linear FV schemes, equal to one [43–45, 59, 87], hence leading to a global error estimate of order 1 with respect to h.

In general, if a continuous reconstruction is not performed on u_h, using the classical L^2-norm to measure the numerical error, the maximum convergence rate that we reach is of order one. This is because we are comparing a piecewise constant function against a continuous one. Moreover, a deterioration of the L^2-norm convergence rate is observable only when dealing with particularly irregular meshes, characterized by strong non-orthogonality and skewness (c.f. Appendix A).

Thus, in the L^2-norm, the error is dominated by the fact that the FVM solution is piecewise constant. This does not always allow to appropriately compare methods accuracy because, if the mesh is regular enough, the convergence rate is up to order one. For this reason, in the next analysis we use the $|\cdot|_*$ to compute the numerical error of the FVM solution, computed with the various methods mentioned above, in order to enlighten their numerical accuracy properties.

Now we present some test cases with analytical solution. We consider two scalar problems and two fluid flow problems. For the solution u_h of scalar problems we compute $|u - u_h|_*$ and $\|u - u_h\|_{L^2}$, for which we expect to have a global convergence of order one. For the solution $(\mathbf{u}_h, p_h)$ of fluid flow problems we compute $|\mathbf{u} - \mathbf{u}_h|_*$ and $\|\mathbf{u} - \mathbf{u}_h\|_{L^2}$ for velocity

and only $\|p - p_h\|_{L^2}$ for pressure, that, based on the results of Chapter 3, we expect to converge with order 1.

Remark 4.2. Our experience with this type of error computation suggests that the optimal order of convergence with respect to $*$-norm is $\frac{3}{2}$. This result is observable when the computational grid is regular enough, in terms of orthogonality and skewness and the adopted discretization schemes are accurate enough. In other cases, applying special numerical schemes that are able to compensate mesh regularity lacking, the observed order of convergence is 1 and it can reach a maximum rate of $\frac{3}{2}$, depending on the mesh regularity and on the correction schemes employed. In worse cases, for particularly distorted meshes, the order of convergence can deteriorate to $\frac{1}{2}$ or even to zero.

Introducing the first test case, we want to mention the meshing strategy, that will be analogous for all the next cases. For each case, we consider three mesh types: the conforming mesh, a non-conforming uniform mesh and a non-conforming mesh with local refinement. In the spirit of what we observed for the Diffuse Interface Box method (c.f. 3.2), we performed the refinement in such a way that the mesh size is h^2, where h is its measure far from the IB surface. The local mesh refinement is performed using the refinement library of OpenFOAM. In particular, it is performed splitting a hexahedron along the three main directions creating hanging nodes with adjacent hexahedra, hence from one hexahedra eight are generated. The three mesh type are represented in Figure 4.7.

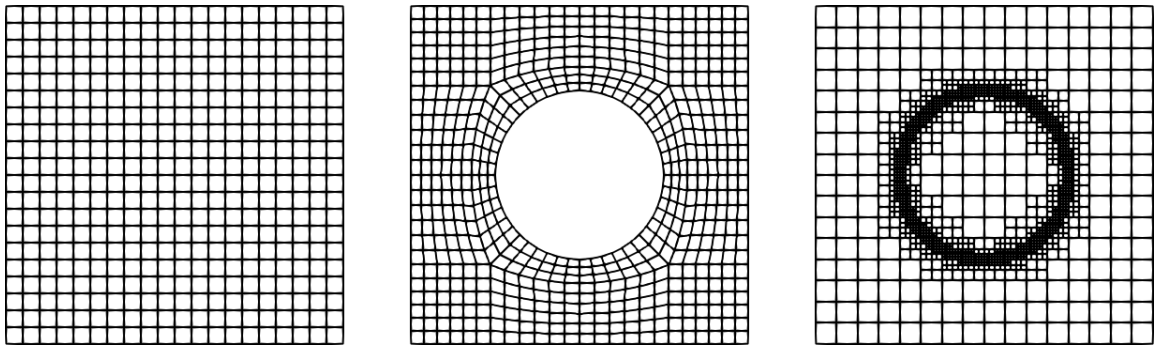


Figure 4.7: Example of the three mesh types employed to perform the numerical assessment of non-conforming methods convergence. Left: uniform non-conforming mesh. Center: conforming mesh. Right: locally refined non-conforming mesh.

4.7.1 Poisson problem

The first case is the simplest one: a Poisson problem on a squared domain with a circular hole: $\tilde{\Omega} = [-1, 1]^2$ and $\Omega = \tilde{\Omega} \backslash B_{\frac{1}{2}}(0)$ (Figure 4.7).

The differential problem reads

$$\begin{cases} -\Delta u = f, & \text{in } \tilde{\Omega}, \\ u = 0, & \text{on } \partial\tilde{\Omega} \end{cases} \tag{4.18}$$

where we set the analytical solution to

$$u = (x^2 - 1)(y^2 - 1)\left(x^2 + y^2 - \frac{1}{4}\right).$$

and compute $f = -\Delta u$, accordingly. We set $h = 0.1, 0.05, 0.025, 0.0125$.

We computed the $*$-norm and the L^2-error of the FVM solution and reported the convergence rates in Figure 4.8. While the L^2 errors show a first order of convergence for each method, the $*$ errors show some difference. In particular, while conforming, IB and refined DIM show the same rates of convergence ($\frac{3}{2}$), we can see for the DIM a lost of half an order, from $\frac{3}{2}$ to 1.

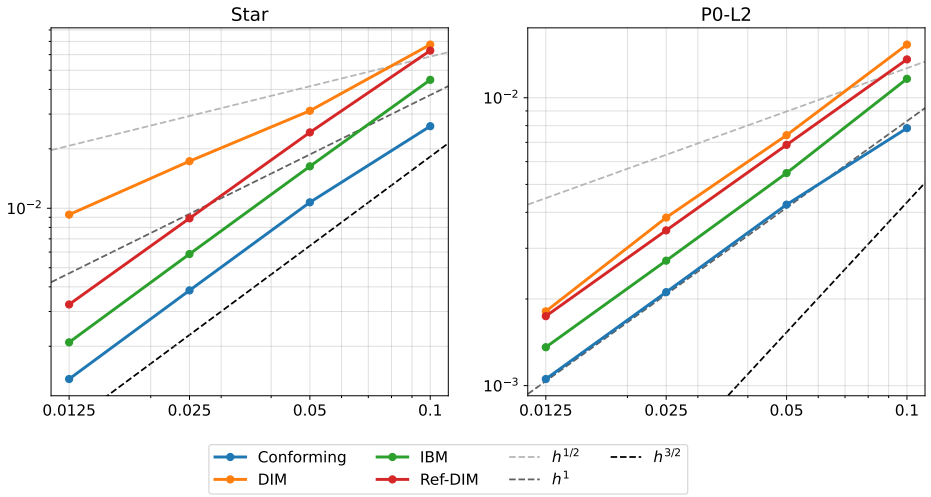


Figure 4.8: Convergence rates for non-conforming methods on Poisson test case in (left) $*$-norm, (right) L^2-norm.

4.7.2 Advection-diffusion problem

The second test case is an advection-diffusion problem. The physical domain is a one by one square $\Omega = [0, 1]^2$.

The differential problem [80] reads

$$\begin{cases} -k\Delta u + \mathbf{b} \cdot \nabla u = f, & \text{in } \tilde{\Omega}, \\ u = g, & \text{on } \partial\tilde{\Omega} \end{cases} \tag{4.19}$$

where k is the diffusivity, $\mathbf{b} = [1, 1]^\intercal$ is the transport velocity. We set the analytical solution to

$$u = x - y\,(x - 1) - \frac{e^{-\frac{(x-1)(y-1)}{k}} - e^{-\frac{1}{k}}}{1 - e^{-\frac{1}{k}}}.$$

and compute g and $f = -k\Delta u + \mathbf{b}\cdot\nabla u$, accordingly. As mesh sizes we set $h = 0.05, 0.025, 0.0125, 0.00625$.

By the fact that the problem does not have an immersed object in its formulation, for the non-conforming methods we set as a hold-all domain $\tilde{\Omega} = [\frac{3}{2}, \frac{3}{2}]^2 \cap B_0(\frac{3}{2})$, where $B_0(\frac{3}{2})$ is the ball of radius $\frac{3}{2}$ centred in 0.

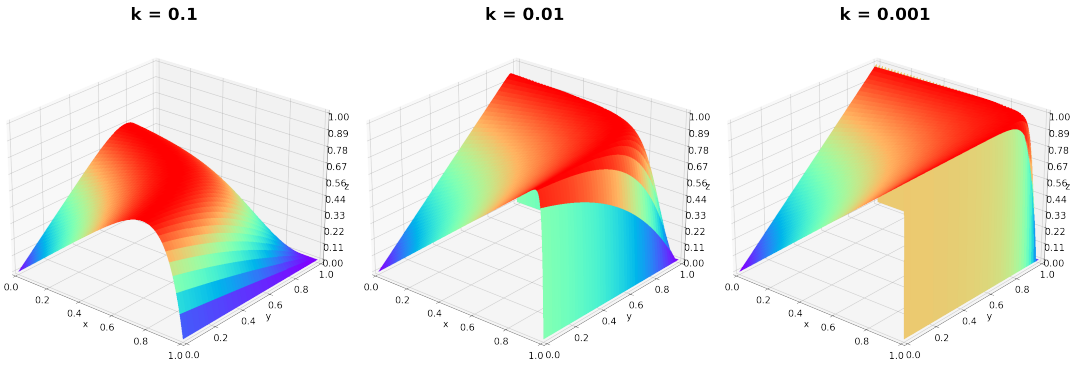


Figure 4.9: Solution representation varying the value of diffusivity k and so increasing the Péclet number.

We select three values of $k = 0.1, 0.01, 0.001$ increasing the importance of the advection and consequently the Péclet number $\mathbb{P}\text{e} = \frac{|\mathbf{b}L|}{k} \simeq 10, 100, 1000$. To solve each problem accurately we adopt the so called *linear upwind* discretization scheme to approximate the advection term. The scheme is as follows:

$$\begin{aligned}\int_{K_i} \mathbf{b}\cdot\nabla u \mathrm{d}x &= \sum_{F_{ij}\in\mathcal{F}_i} \int_{F_{ij}} \mathbf{b}\cdot\mathbf{n}_{ij} u \mathrm{d}s \\ &\simeq \int_{F_{ij}} \mathbf{b}\cdot\mathbf{n}_{ij}\left(w_{ij}u_i + (1-w_{ij})u_j + (\mathbf{f}_{ij} - \mathbf{k}_{\text{up}})\cdot(\nabla_h u_h)_{\text{up}}\right)\mathrm{d}s,\end{aligned}$$

$$\text{where } w_{ij} = \begin{cases} 1 \text{ if } \mathbf{b}\cdot\mathbf{n}_{ij} \geq 0, \\ 0 \text{ if } \mathbf{b}\cdot\mathbf{n}_{ij} < 0 \end{cases}$$

$$\text{and up} = \begin{cases} i \text{ if } \mathbf{b}\cdot\mathbf{n}_{ij} \geq 0, \\ j \text{ if } \mathbf{b}\cdot\mathbf{n}_{ij} < 0 \end{cases}$$

where $(\nabla_h u_h)_{\text{up}}$ is an approximation of the cell gradient in the upwind cell and it is considered explicit in the formulation. It is basically an upwind scheme with an explicit correction: in the upwind scheme $(\nabla_h u_h)_{\text{up}}$ would be zero.

Given this setup, for each simulation, we computed the $*$-norm and the L^2-error of the FVM solution and reported the convergence rates in Figure 4.10. It is visible how the numerical error deteriorates for all methods with the decrease of parameter k. Moreover, for the non-conforming methods, the deterioration is both in error magnitude and convergence rate, while the conforming approach keeps the convergence rate constant, at $\frac{3}{2}$ in the $*$-norm and 1 in the L^2 norm.

For $k = 0.1$ we observe the same convergence rates of the Poisson problems for both conforming and IB methods, while for the DIM we observe a loss of half an order between the refined and the uniform meshes cases.

What is interesting to notice is the behaviour of the IBM. Despite a slight deterioration of the convergence rate with the decrease of diffusivity, the IBM always keeps the error magnitude under the one of the conforming method. This is probably due to the fact that it can happen to let some cell outside of the error count.

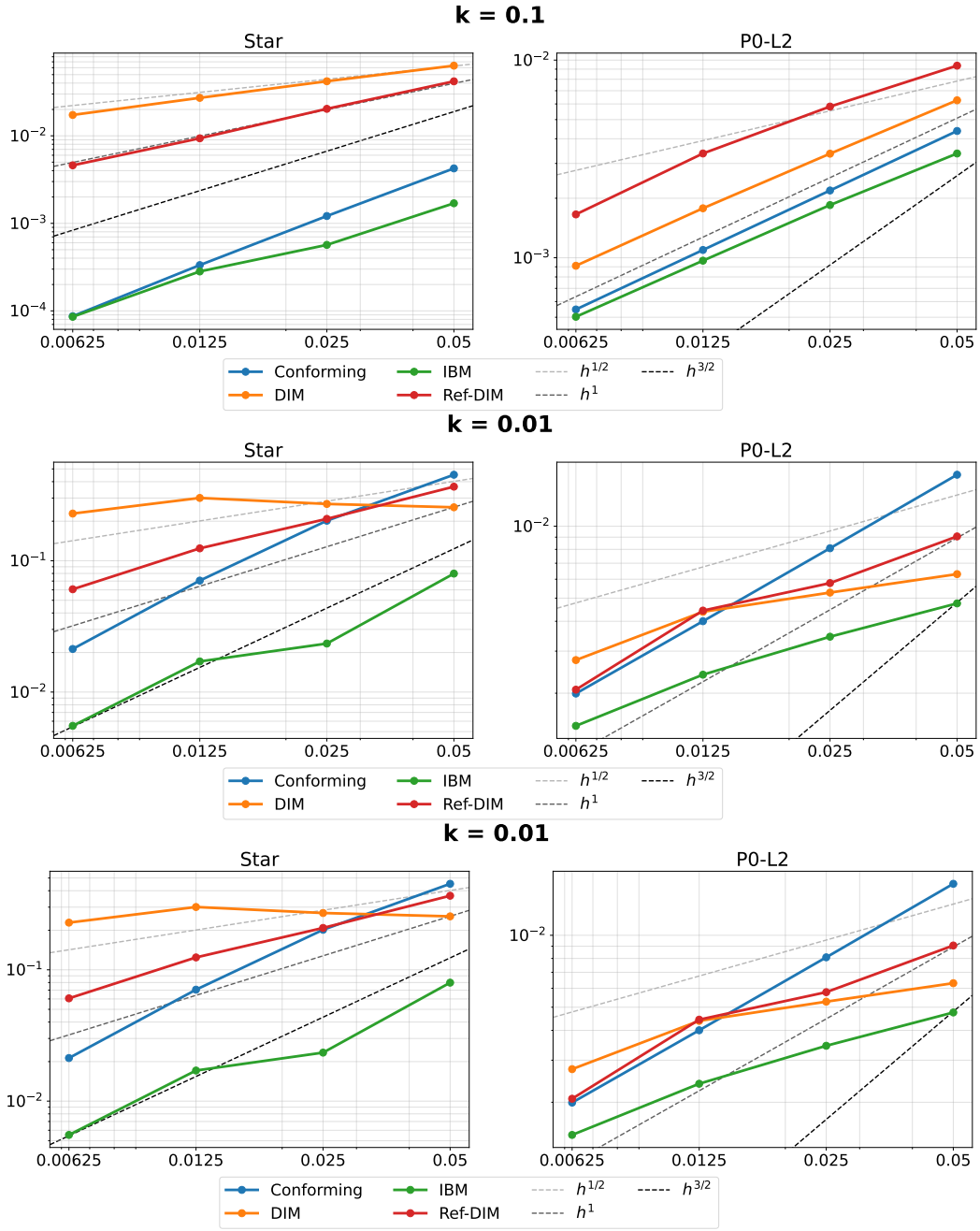


Figure 4.10: Convergence rates for non-conforming methods on the advection diffusion problem, varying the Péclet number. For each chart, we represent the $*$ norm on left and the L^2-norm on right.

4.7.3 Stokes problem for a Newtonian fluid

The third test case is made by a Stokes system describing the flow around a sphere. The domain is rectangular with a spheric obstacle of radius R_s in its centre: $\tilde{\Omega} = [-2R_s, 2R_s] \times [-2R_s, 2R_s] \times [-2R_s, 2R_s]$ and $\Omega = \tilde{\Omega} \backslash B_{R_s}(0)$. The problem is the following:

$$\begin{cases} -\Delta \mathbf{u} = -\nabla p & \text{in } \tilde{\Omega}, \\ \nabla \cdot \mathbf{u} = 0 & \text{in } \tilde{\Omega} \\ \mathbf{u} = \mathbf{0}, & \text{in } B_{R_s}(0), \\ \mathbf{u} = \bar{\mathbf{u}} & \text{on } \partial\tilde{\Omega}, \end{cases} \tag{4.20}$$

with the following analytical solution:

$$\mathbf{u} = \begin{bmatrix} \dfrac{U_0 xz \left(\frac{{R_s}^3}{2(x^2+y^2+z^2)^{\frac{3}{2}}} - \frac{3R_s}{2\sqrt{x^2+y^2+z^2}} + 1 \right)}{x^2+y^2+z^2} + \dfrac{U_0 xz \left(\frac{{R_s}^3}{4(x^2+y^2+z^2)^{\frac{3}{2}}} + \frac{3R_s}{4\sqrt{x^2+y^2+z^2}} - 1 \right)}{x^2+y^2+z^2} \\ \dfrac{U_0 yz \left(\frac{{R_s}^3}{2(x^2+y^2+z^2)^{\frac{3}{2}}} - \frac{3R_s}{2\sqrt{x^2+y^2+z^2}} + 1 \right)}{x^2+y^2+z^2} + \dfrac{U_0 yz \left(\frac{{R_s}^3}{4(x^2+y^2+z^2)^{\frac{3}{2}}} + \frac{3R_s}{4\sqrt{x^2+y^2+z^2}} - 1 \right)}{x^2+y^2+z^2} \\ \dfrac{U_0 z^2 \left(\frac{{R_s}^3}{2(x^2+y^2+z^2)^{\frac{3}{2}}} - \frac{3R_s}{2\sqrt{x^2+y^2+z^2}} + 1 \right)}{x^2+y^2+z^2} - \dfrac{U_0 \left(x^2+y^2\right) \left(\frac{{R_s}^3}{4(x^2+y^2+z^2)^{\frac{3}{2}}} + \frac{3R_s}{4\sqrt{x^2+y^2+z^2}} - 1 \right)}{x^2+y^2+z^2} \end{bmatrix}$$

where U_0 is the characteristic velocity of the flow, that in this precise case is 1 m/s.

The convergence rates for each method are reported in Figure 4.11. While conforming method, as expected, shows the most accurate results, we can observe the same difference resulting from the analysis of the Box method described in Section 3.2.3. It is clearly visible the difference in accuracy and convergence rate of the DIM on a uniform mesh with respect to the same method on a locally refined mesh. This difference is visible for the $*$-norm velocity error and for the L^2-norm pressure error.

Furthermore, the IB method is comparable in accuracy with the conforming one. The only difference between the two is for the velocity $*$-norm error, where the convergence rate is of order 1, aligned with the FVM theory but loosing half an order with respect to the conforming result.

4.7.4 Non-Newtonian Navier-Stokes problem

The fourth test case is made by a Navier-Stokes system describing the non-Newtonian flow past between two cylinders: a Taylor-Couette flow. The domain is made by two concentric cylinders: $\tilde{\Omega} = B_{R_1}(0)$ and $\Omega = \tilde{\Omega} \backslash B_{R_0}(0)$. The problem is the following:

$$\begin{cases} -\nabla \cdot (\nu(\dot{\gamma})(\nabla \mathbf{u} + \nabla^{\intercal} \mathbf{u})) + (\mathbf{u}\nabla) \cdot \mathbf{u} = -\nabla p & \\ \nabla \cdot \mathbf{u} = 0 & \text{in } \tilde{\Omega}, \\ \mathbf{u} = \mathbf{0} & \text{on } \partial B_{R_1}(0), \\ \mathbf{u} = R_0 \boldsymbol{\omega}_0 & \text{on } \partial B_{R_0}(0) \end{cases} \tag{4.21}$$

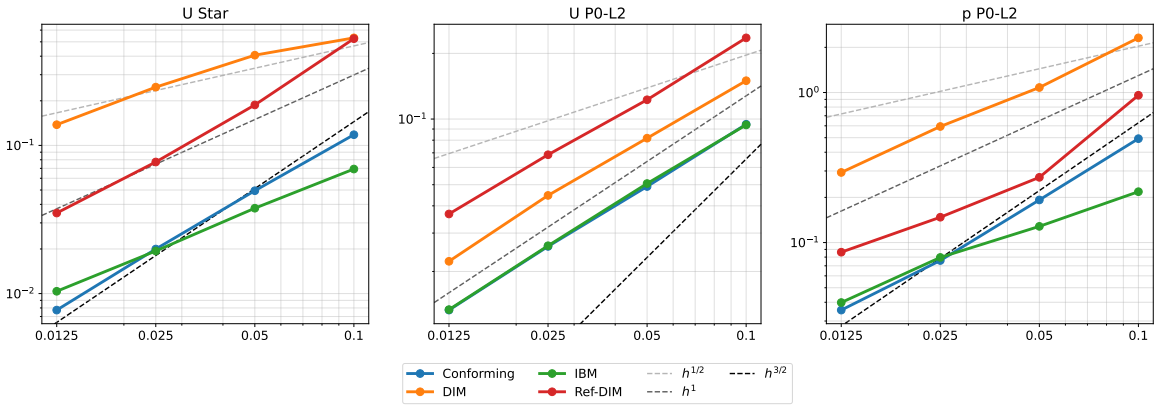


Figure 4.11: Convergence rates for non-conforming methods applied on the Stokes flow past a sphere test case. Left: $*$-norm of velocity error. Centre: L^2-norm of the velocity error. Right: L^2-norm of pressure error.

where

$$\dot{\gamma} = \sqrt{2}\,\|\nabla \mathbf{u} + \nabla^{\intercal}\mathbf{u}\|_2\,, \qquad \nu(\dot{\gamma}) = \dot{\gamma}^{n-1},$$

$\boldsymbol{\omega}_0$ is the angular velocity of the inner cylinder and n is the exponent of the power law. The analytical solution reads:

$$\mathbf{u} = \begin{bmatrix} -\dfrac{\omega_0 y\left(\left(\dfrac{R_1}{\sqrt{x^2+y^2}}\right)^{2/n} - 1\right)}{\left(\dfrac{R_0}{R_1}\right)^{-2/n} - 1} \\ \dfrac{\omega_0 x\left(\left(\dfrac{R_1}{\sqrt{x^2+y^2}}\right)^{2/n} - 1\right)}{\left(\dfrac{R_0}{R_1}\right)^{-2/n} - 1} \end{bmatrix}.$$

The convergence rates for each method are reported in Figure 4.11. The results are similar to the ones of the Stokes problem. The conforming method is the most accurate, the DIM the less accurate and the IBM shows the same convergence features of conforming method apart from the error computed with the $*$-norm, where the convergence rate deteriorates. A similar deterioration is also observed on the L^2 error of pressure. Whilst, the L^2 error of velocity is the same for IB and conforming methods.

For this specific case, we do not have any Dirichlet condition on pressure, hence the associated Poisson problem has only homogeneous Neumann boundary conditions. This is dealt in OpenFOAM by weakly penalizing (adding an artificial value in the algebraic system to the diagonal term of a row) the pressure value of a degree of freedom to be zero. This can imply some slight translation of the pressure field, that can reflect on the error computation.

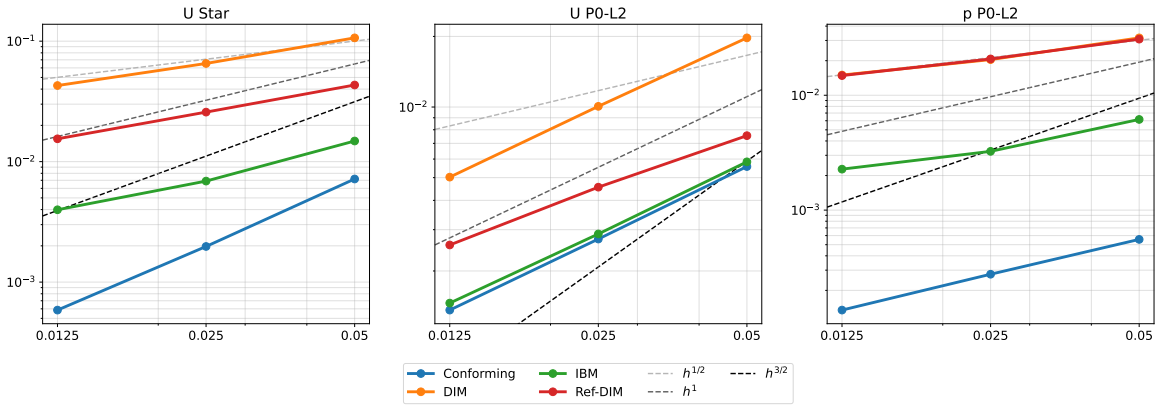


Figure 4.12: Convergence rates for non-conforming methods applied to the non-Newtonian Taylor-Couette flow test case. Left: $*$-norm of velocity error. Centre: L^2-norm of the velocity error. Right: L^2-norm of pressure error.

4.7.5 A case of interest: incompressible non-Newtonian Navier-Stokes with temperature-dependent viscosity

In this section we consider a single screw extruder geometry of the test case introduced in [92], represented in Figure 4.13. The geometry is built by twisting, i.e. extruding and rotating, the xy section along z direction. The fluid domain is between the screw and a barrel, that is represented by a cylinder.

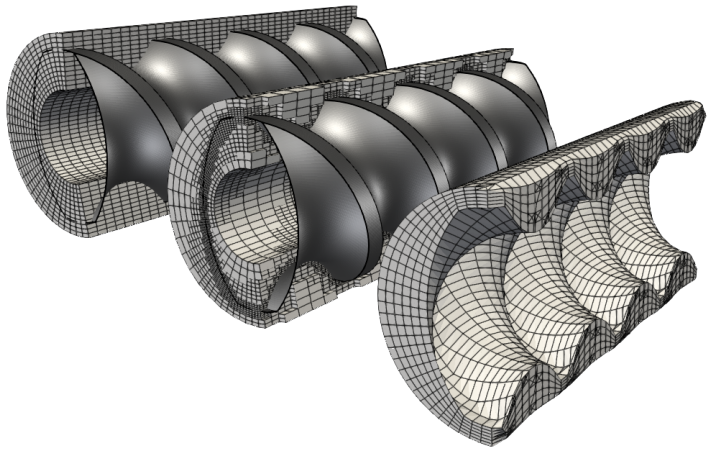


Figure 4.13: Representation of the meshes employed to perform simulations. Left: uniform non-conforming mesh. Center: refined non-conforming mesh. Right: conforming mesh. We represented also how the single-screw geometry intersects the non-conforming meshes.

We now recall steady-state problem (2.27), that we are going to solve on this geometry:

$$\begin{aligned}(\mathbf{u}\cdot\nabla)\mathbf{u} =& -\nabla p + \nabla\cdot(\nu(\dot\gamma,T)(\nabla\mathbf{u}+\nabla^{\intercal}\mathbf{u}))\,,\\ \nabla\cdot\mathbf{u} =& 0,\\ \rho\mathbf{u}\cdot\nabla(c_pT) =& \nabla\cdot(k\nabla T)+\mu(\dot\gamma,T)(\nabla\mathbf{u}+\nabla^{\intercal}\mathbf{u}):\nabla\mathbf{u}.\end{aligned}$$

For the rheological characterization of the material we take values from [92] (Table 4.1). We consider Arrhenius law for the shift factor and Carreau Yasuda law for the shear dependence:

$$\begin{aligned}H(T) &= \exp\left(\alpha\left(\frac{1}{T}-\frac{1}{T_\alpha}\right)\right),\\ \nu(\dot\gamma,T) &= \max\left\{\nu_{min},\,\min\left\{\nu_{max},\,H(T)K\left(1+H(T)\dot\gamma\right)^{n-1}\right\}\right\}.\end{aligned} \tag{4.22}$$

K	n	ν_{min}	ν_{max}	λ	α	T_α	ρ	C_p	k
21.18	0.334	0.0001	15.28	0.6247	5530	420	700	500	1.5

Table 4.1: Thermal and rheological parameters.

Dirichlet conditions are imposed for velocity on the walls: a rotational velocity of 90 RPM is imposed on the screw and no-slip condition on the barrel. For temperature we imposed 473K on the barrel while the inflow temperature is set to a uniform distribution at 463K. Neumann boundary conditions on pressure are imposed on walls and a zero mean distribution at inflow and outflow. For the screw temperature we considered two different conditions: an homogeneous Neumann condition and a Dirichlet conditions, set to 463K.

In the following numerical experiments we considered three methods to approximate the geometry. First the geometry-conforming method, second the DIM and finally the IBM. Clearly, for this test case we do not have the expression of the analytical solution, hence, to assess the convergence of the above methods we measure some global quantities of interest, in particular surface integrals. To build the conforming grid we have first generated an hollow cylinder and then we have deformed the internal cylinder face points to the screw surface, moving the internal points accordingly.

For five decreasing mesh sizes,

$$h = 10^{-3}, 8\cdot10^{-4}, 6\cdot10^{-4}, 4\cdot10^{-4}, 2\cdot10^{-4},$$

we measured the following quantities: the average temperature, the viscous heating, the viscosity, the pressure and the flow rate. For each of these quantities we computed a global integral, or average, on the screw surface. In particular, to evaluate quantities on the screw in non-conforming methods, we used an extrapolation derived from the way in which that method imposes boundary conditions: a constant extrapolation for the DIM and a quadratic extrapolation for the IBM. The results of the analysis for this 3D cases are reported in Figures 4.14 and 4.16. We also reported the surface distribution of these quantities on the screw

surface for each method in Figures 4.15 and 4.17, for the Dirichlet and Neumann conditions cases, respectively.

Here, the reference solution is set to the conforming solution, that is considered the most accurate method, in the light of considerations made in previous sections.

In both cases, viscosity is far better approximated by the IBM than by the DIM. This is because, in the IBM, computed gradients are corrected as well as the velocity by the quadratic least-square approximation. The same observation can be done for the viscous heating, even if here we are farther from the conforming solution.

For what concerns the flow rate estimation, it is well known that using non-conforming like IBM or DIM can produce mass leakage due to the presence of non-standard strategies to impose boundary conditions. In these results we can observe how this quantity is well preserved in all methods, due to the consistent pressure condition that we implemented for our methods (c.f. Chapter 5).

Finally, from Figures 4.15 and 4.17, it can be seen how the extrapolation of screw boundary values behaves for the different methods. In all the cases, the IBM profile is smoother than DIM profiles, also in the refined mesh case, and it is much similar to the one of conforming method.

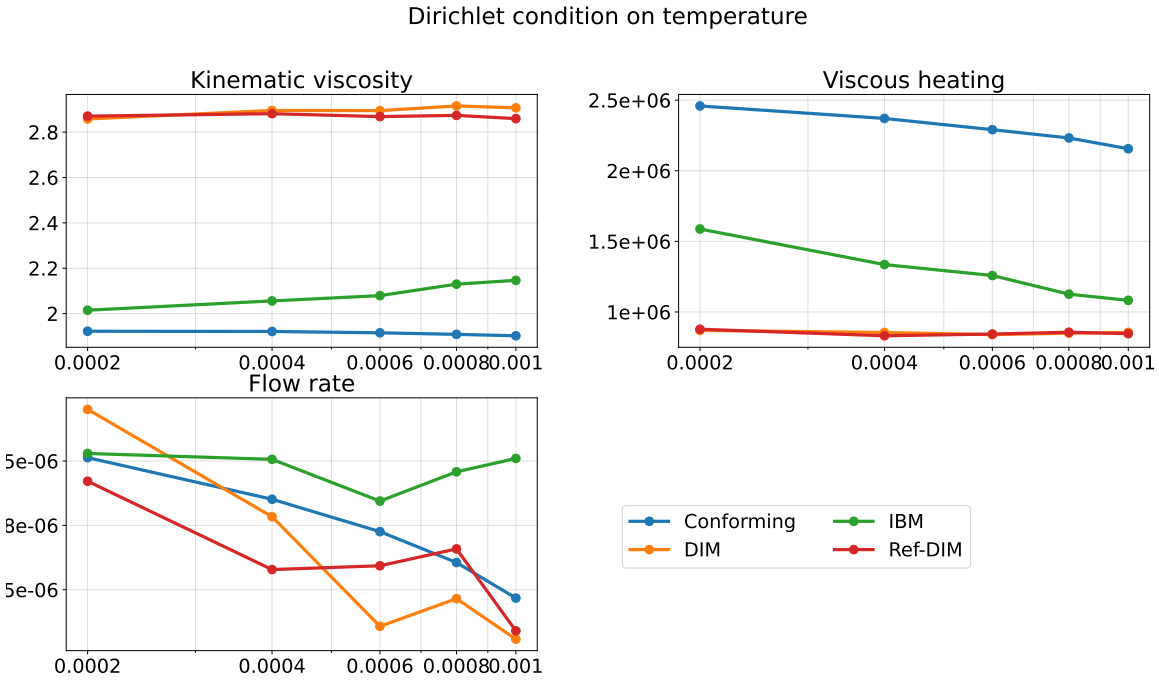


Figure 4.14: Convergence of quantities integrated on the screw surface quantities on the 3D single-screw extruder test case.

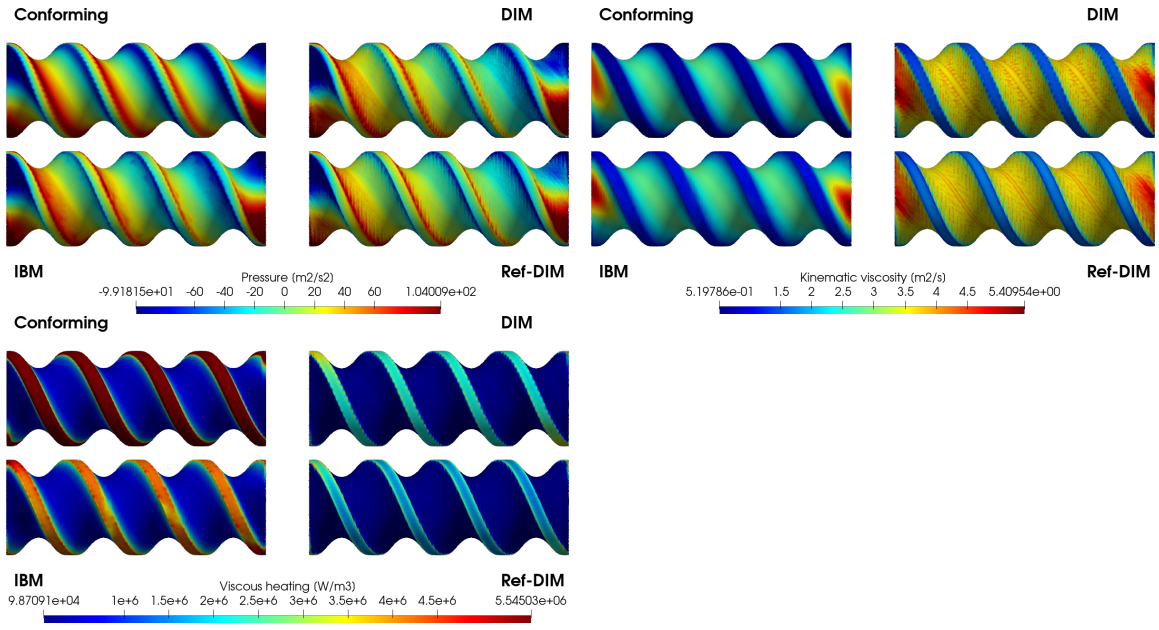


Figure 4.15: Evaluation of various quantities on the screw surface for each method in the Dirichlet temperature condition case.

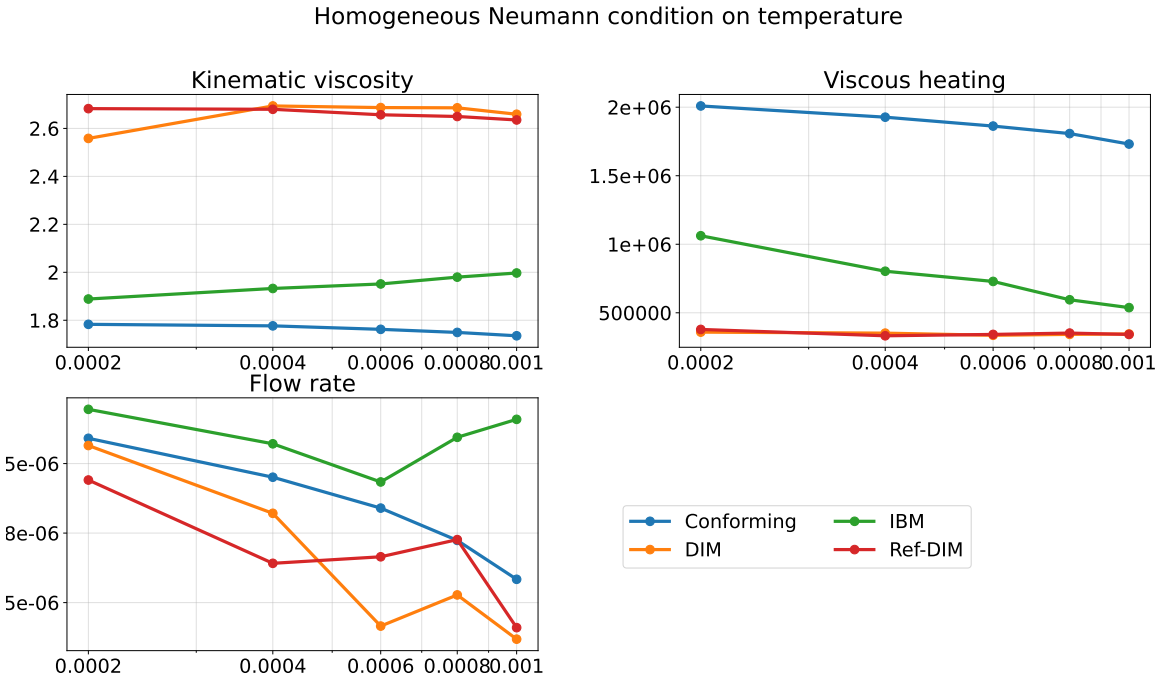


Figure 4.16: Convergence of quantities integrated on the screw surface quantities on the 3D single-screw extruder test case.

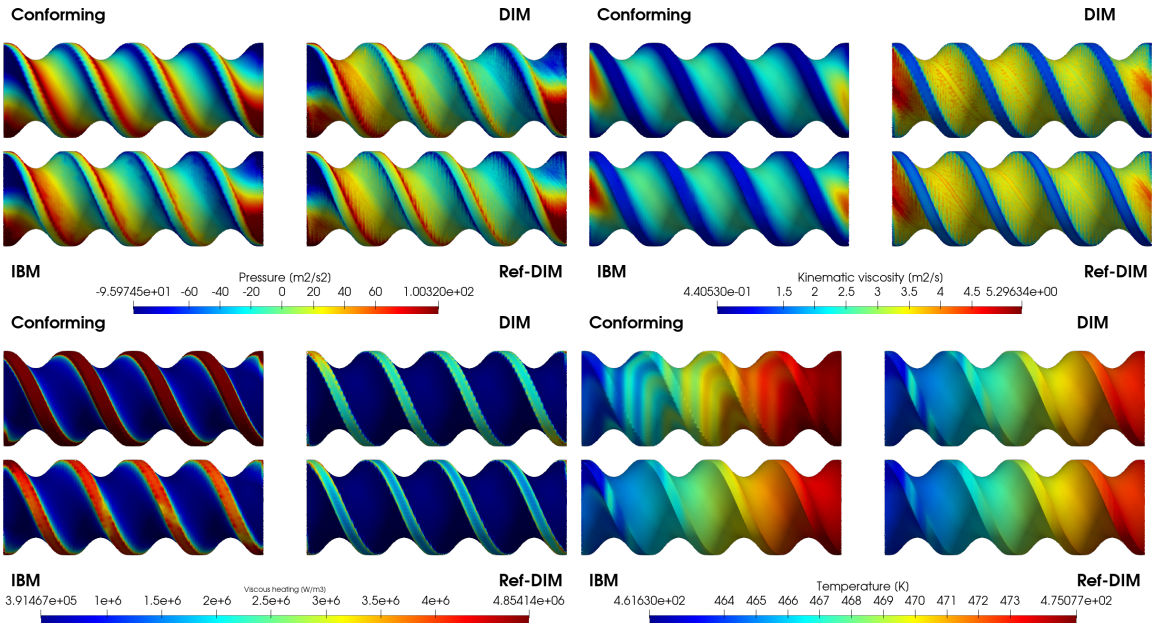


Figure 4.17: Evaluation of various quantities on the screw surface for each method in the Neumann temperature condition case.

Chapter 5

Solution algorithms for simulation of polymer mixing

In this chapter, we report the description of the tolls that remain for us to describe, after the Immersed Boundary method (see 4), to solve numerical problems related to the mixing of non-Newtonian fluids with temperature-dependent viscosity.

The chapter is divided into two parts. in the first part the procedure to solve the Navier-Stokes equations decoupling velocity and pressure using a projection method is first recalled. In particular, the SIMPLE algorithm is described in depth. Then, the same decoupling algorithm is extended to problem involving immersed boundaries (Chapter 4), discussing how the usage of a generic non-conforming method can be integrated in the SIMPLE algorithm.

Finally, the strategy developed to solve the strong nonlinearity associated to the non-Newtonian rheology of polymer melts is described. The coupling between velocity and temperature through the viscosity can be tough to solve and it might require a special treatment. A sensitivity analysis is carried out to study how the problem parameters can influence the robustness of this family of algorithms.

In the second part we report the description of the C++ tools that were implemented in order to solve the problems in object. First we introduce the implementations regarding rheological models of non-Newtonian fluids with temperature dependent viscosity. Therefore, we briefly describe the implementation of the SIMPLEX and PIMPLEX algorithms. Ultimately, we report a more detailed description of the library part that implements the Immersed Boundary method.

Hence, in section 5.1, we introduce the SIMPLE and PIMPLE projection methods for the decoupling of Navier-Stokes equations in the FVM framework. Then, in section 5.2, we exploit how IBM is integrated into SIMPLE and PIMPLE projection algorithms. In section 5.3 we describe the new methodology implemented to solve robustly the polymer mixing processes and the nonlinearity they involve. Finally, in section 5.4 we introduce the POLIMIX toolbox, that we used to produce all the numerical results in this work, and all the novelties that we have implemented with respect to the official OpenFOAM distribution.

5.1 Projection methods for isothermal incompressible flows

The SIMPLE (Semi-Implicit method for Pressure Linked Equations) algorithm is an iterative method first developed by Patankar [110] for solving steady-state flow problems. This algorithm is intensively used in the framework of the open source code OpenFOAM. The advantage in using this type of solving strategy lays in the fact that it is very efficient when dealing with Navier-Stokes system, both in terms of memory usage and of ability in solving nonlinearities, thanks to its iterative nature. Moreover, at the algebraic level, the SIMPLE algorithm can be seen as an inexact factorization algorithm [54].

On the other hand, to deal with time dependent problems, we consider the PIMPLE algorithm, that is a combination of the PISO (Pressure Implicit with Splitting of Operator) algorithm [131] and the SIMPLE algorithm. PISO algorithm decoupling is the same of the SIMPLE, where the advancing in time is solved with just one SIMPLE iteration between time steps. Then, one can see the PIMPLE algorithm as a PISO where each time step is solved by a SIMPLE iteration, thus for each time step some sub-iterations are performed until convergence is reached. The choice of using PIMPLE instead of PISO has been made for robustness reasons because of the nonlinearity present in mixing processes.

5.1.1 SIMPLE algorithm

To introduce the SIMPLE algorithm, we consider the steady-state incompressible Navier-Stokes equations for a Newtonian fluid:

$$\begin{aligned} (\mathbf{u}\cdot\nabla)\,\mathbf{u} - \nabla\cdot(\nu(\nabla\mathbf{u}+\nabla^{\intercal}\mathbf{u})) &= -\nabla p, \quad \text{in } \Omega, \\ \nabla\cdot\mathbf{u} &= 0, \quad \text{in } \Omega, \end{aligned} \tag{5.1}$$

complemented with suitable boundary conditions.

The main idea behind SIMPLE algorithm is a splitting of the Navier-Stokes equations (5.1) leading to a sequence of simpler problems involving either the velocity or the pressure, namely at each iteration:

$$\begin{aligned} &\text{Velocity prediction:} && \mathbf{u}^{n}\cdot\nabla\mathbf{u}^{*} - \nabla\cdot(\nu(\nabla\mathbf{u}^{*}+\nabla^{\intercal}\mathbf{u}^{n})) = -\nabla p^{n}, \\ &\text{Pressure equation:} && \nabla\cdot\left(\mathrm{D}^{-1}\nabla p^{n+1}\right) = \nabla\cdot\mathbf{u}^{*}, \\ &\text{Velocity correction:} && \mathbf{u}^{n+1} = \mathbf{u}^{*} - \mathrm{D}^{-1}\nabla p^{n+1}, \end{aligned} \tag{5.2}$$

where $\mathbf{u}^{*}$ is the velocity prediction, p is the pressure and D is the diagonal of momentum discretization matrix.

The idea is to solve first the momentum equation using the pressure field at step n, then to solve a pressure Laplacian equation derived from the incompressibility constraint and finally to project velocity on divergence free functions space applying a pressure-dependent correction.

Relaxation Iterative methods, like Jacobi of Preconditioned Conjugate Gradient methods, are often employed to solve large linear systems. The convergence of these methods may be not guaranteed when the matrix is not diagonal dominant. However, working with general

grids requires the usage of non-orthogonality and skewness corrections (c.f. Appendix A) that include the dependence of other mesh cells apart from the adjacent cells [85]. This disrupts the diagonal dominance and often prevents the solver to converge. In time dependent problems, this issue is overcome by the presence of the diagonal mass matrix associated to the time derivative. In steady-state under-relaxation is usually adopted to guarantee the convergence.

The relaxation of velocity and pressure problems is performed in order to keep the solution stable. Let $\mathrm{A}\mathbf{x}^{n+1} = \mathbf{b}$ be the algebraic system of momentum equation at step $n+1$ and let D be the diagonal of matrix A. The relaxation with factors $\alpha_{\mathbf{u}}, \alpha_p \in (0, 1]$ is performed in the following way:

$$(A + \frac{1-\alpha_{\mathbf{u}}}{\alpha_{\mathbf{u}}} D)\mathbf{x}^{n+1} = \mathbf{b} + \frac{1-\alpha_{\mathbf{u}}}{\alpha_{\mathbf{u}}} D\mathbf{x}^{n}. \tag{5.3}$$

On the other hand, pressure is relaxed after having solved the linear system associated to the pressure Poisson problem:

$$p^{n+1} = (1-\alpha_p)p^n + \alpha_p p^{n+1}. \tag{5.4}$$

Pressure relaxation is needed to take into account the error, that resides in pressure equation source term, performed when solving momentum prediction. The recommended values of under-relaxation factors are such that $\alpha_p = 1 - \alpha_{\mathbf{u}}$ [65]. A typical choice is $\alpha_{\mathbf{u}} = 0.7 \sim 0.8$ and $\alpha_p = 0.2 \sim 0.3$.

5.1.2 PIMPLE algorithm

Before introducing the PIMPLE algorithm, we introduce the PISO one. We now consider the time dependent incompressible Navier-Stokes equations for a Newtonian fluid:

$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} - \nabla\cdot(\nu(\nabla\mathbf{u} + \nabla^{\intercal}\mathbf{u})) &= -\nabla p, \quad \text{in } \Omega, \\ \nabla\cdot\mathbf{u} &= 0, \quad \text{in } \Omega, \end{aligned}$$

complemented with suitable initial and boundary conditions.

Consider a backward Euler discretization for the time derivative, with time step Δt. Analogously to what is done in the SIMPLE, the PISO algorithm splits the Navier-Stokes equations as follows: at each time step t^{n+1},

$$\begin{aligned} &\text{Velocity prediction:} && \frac{1}{\Delta t}\mathbf{u}^* + \mathbf{u}^n\cdot\nabla\mathbf{u}^* - \nabla\cdot(\nu(\nabla\mathbf{u}^* + \nabla^{\intercal}\mathbf{u}^n)) = -\nabla p^n + \frac{1}{\Delta t}\mathbf{u}^n, \\ &\text{Pressure equation:} && \nabla\cdot\left(\mathrm{D}^{-1}\nabla p^{n+1}\right) = \nabla\cdot\mathbf{u}^*, \\ &\text{Velocity correction:} && \mathbf{u}^{n+1} = \mathbf{u}^* - \mathrm{D}^{-1}\nabla p^{n+1}, \end{aligned} \tag{5.5}$$

where $\mathbf{u}^*$ is the velocity prediction, p is the pressure and D is the diagonal of momentum discretization matrix.

Again, as for SIMPLE, we solve first the momentum equation using the pressure field at step n, then we solve a pressure Poisson equation derived from the incompressibility constraint and finally to project velocity on divergence free functions space applying a pressure-dependent correction.

The PIMPLE algorithm is a combination of PISO and SIMPLE in which, for each PISO iteration, we perform SIMPLE sub-iterations in order to reach convergence at each time step up to a described tolerance. Employing PIMPLE is useful when using large time steps or when the nonlinearity is particularly relevant, like in our applications.

5.2 The SIMPLE-IBM algorithm for isothermal incompressible flows

In this section we discuss how the IBM is integrated into the SIMPLE algorithm, in particular with respect to the imposition of a motion and its effect on the pressure equation. The idea is to impose a consistent condition on pressure as it is done in [83] for finite differences. Here we extend this procedure to the FVM framework.

Let N be the number of degrees of freedom of the tessellation $\mathcal{T}_h$ of Ω. Let $\mathbf{U} \in \mathbb{R}^{3N}$ and $\mathbf{P} \in \mathbb{R}^N$ be the vectors of velocity and pressure values, respectively. Let $\mathbf{f} \in \mathbb{R}^{3N}$ be the source term. Let $\mathrm{A} \in \mathbb{R}^{3N\times 3N}$ be the matrix representing the transport and diffusion operators and $\mathrm{B} \in \mathbb{R}^{3N\times N}$ be the matrix representing the divergence operator. Notice that $\mathrm{B}^\intercal$ represents the gradient operator. Let also $\mathrm{C} \in \mathbb{R}^{N\times N}$ be the matrix that represents Rhie-Chow stabilization term (c.f. Section 3.1.3). Then the algebraic counterpart of system (5.1) reads:

$$\begin{bmatrix} \mathrm{A} & \mathrm{B}^\intercal \\ \mathrm{B} & -\mathrm{C} \end{bmatrix} \begin{bmatrix} \mathbf{U} \\ \mathbf{P} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \mathbf{0} \end{bmatrix}, \tag{5.6}$$

that can be rewritten as

$$\begin{aligned} \mathrm{A}\mathbf{U} + \mathrm{B}^\intercal\mathbf{P} &= \mathbf{f}, \\ \mathrm{B}\mathbf{U} - \mathrm{C}\mathbf{P} &= \mathbf{0} \end{aligned}$$

Let χ be the indicator function of the discrete fluid region Γ_F (see equation (4.3)). As we showed in Section 4.2, when dealing with non-conforming methods the velocity imposed in the IB region can be also dependent on neighbouring values of the solution. Consider the IBM interpolation operator (4.11): $\mathbf{U}^* = S_{\mathrm{IB}}(\mathbf{g}, \mathbf{U}^*)$, where $*$ represents the algorithm step and $\mathbf{g} \in \mathbb{R}^N$ is the immersed boundary and solid region datum.

Assume we are at time t^n and we want to compute solution at time t^{n+1}. The algebraic stabilized monolithic system of problem (5.1) reads:

$$\begin{aligned} \mathrm{A}\mathbf{U}^{n+1} + \mathrm{B}^\intercal\mathbf{P}^{n+1} &= \mathbf{f} + \mathbf{b}, \\ \mathrm{B}\mathbf{U}^{n+1} - \mathrm{C}\mathbf{P}^{n+1} &= \mathbf{0} \end{aligned} \tag{5.7}$$

where $\mathbf{b}$ is used to impose the non-conforming condition and it is defined as

$$\mathbf{b} = (1-\chi)\left(-\mathrm{A}\mathbf{U} + \mathrm{B}^\intercal\mathbf{P} + \mathbf{U} - S_{\mathrm{IB}}(\mathbf{g}, \mathbf{U})\right).$$

The vector $\mathbf{b}$ is built such that it equals the matricial terms and sets the IB and solid values where $\chi = 0$.

We now split the matrix A in its diagonal part D and its off-diagonal part -H, such that A=D-H. We also define matrix C as follows

$$\mathrm{C} = -\mathrm{B}\mathrm{D}^{-1}\mathrm{B}^{\intercal} + \mathrm{R}(\mathrm{D}^{-1}) \tag{5.8}$$

where $\mathrm{R}(\mathrm{D}^{-1})$ is the stiffness matrix associated to the second term of Rhie-Chow stabilization (see Proposition 3.1.1), representing a Laplacian of pressure where the diffusivity coefficient is D^{-1}.

We divide the algorithm in five steps.

1. **Momentum predictor:** we make a prediction of the velocity field solving the first equation of the system imposing the rigid body motion and using pressure at step n, obtaining $\mathbf{U}^{n+\frac{1}{3}}$:

$$\mathrm{A}\mathbf{U}^{n+\frac{1}{3}} + \mathrm{B}^{\intercal}\mathbf{P}^{n} = \mathbf{f} + \mathbf{b} \quad \Longrightarrow \quad \mathbf{U}^{n+\frac{1}{3}} = \mathrm{A}^{-1}[-\mathrm{B}^{\intercal}\mathbf{P}^{n} + \mathbf{f} + \mathbf{b}] \tag{5.9}$$

2. **Velocity correction:** we compute a new velocity field as

$$\mathbf{U}^{n+\frac{2}{3}} = \mathrm{D}^{-1}(\mathrm{H}\mathbf{U}^{n+\frac{1}{3}} + \mathbf{f}).$$

   At this point, the solution vector should satisfy:

$$\mathbf{U}^{n+\frac{2}{3}} = S_{\mathrm{IB}}(\mathbf{g}, \mathbf{U}^{n+\frac{2}{3}})$$

   where $\chi = 0$. Then velocity field at step $n+1$ would read:

$$\mathbf{U}^{n+1} = \mathbf{U}^{n+\frac{2}{3}} - \mathrm{D}^{-1}\mathrm{B}^{\intercal}\mathbf{P}^{n+1}. \tag{5.10}$$

   However $\mathbf{P}^{n+1}$ is unknown at this step.

3. **Pressure equation:** by multiplying equation (5.10) by divergence matrix B, by definition we obtain $\mathrm{B}\mathbf{U}^{n+\frac{2}{3}} - \mathrm{C}\mathbf{P}^{n+\frac{2}{3}} = 0$, hence the pressure equation reads:

$$\mathrm{B}\mathbf{U}^{n+\frac{2}{3}} = \mathrm{B}\mathrm{D}^{-1}\mathrm{B}^{\intercal}\mathbf{P}^{n+\frac{2}{3}} + \mathrm{C}\mathbf{P}^{n+\frac{2}{3}} = \mathrm{R}(\mathrm{D}^{-1})\mathbf{P}^{n+\frac{2}{3}}, \tag{5.11}$$

   that is the algebraic counterpart of a Laplacian problem:

$$\nabla \cdot \mathbf{u}^{n+\frac{2}{3}} = \nabla \cdot \left(\mathrm{D}^{-1}\nabla p^{n+\frac{2}{3}}\right).$$

   IB interpolation: in general, when applying the IB interpolator S_{IB} we are performing a matrix-vector multiplication (see equation (4.12)). If it is the case in which the IBM performs a constant extrapolation of the immersed boundary datum in the IB cell centre (c.f. DIM), the matrix that represents S_{IB} is an identity and the SIMPLE algorithm is not modified. In any other case, i.e. when the IB cells values are corrected, the pressure equation requires a modification.

Requiring that $\mathbf{U}^{n+1}$, as defined in equation (5.10), is consistent with the non-conforming condition, we can derive a correction for the pressure field $\mathbf{P}^{n+\frac{2}{3}}$. The constraint reads: let S_g, S be the matrices that represent the IB interpolator S_{IB},

$$\begin{aligned}\mathbf{U}^{n+1} =& S_{\mathrm{IB}}(\mathbf{g}, \mathbf{U}^{n+1}) \\ =& \mathrm{S}_g\mathbf{g} + \mathrm{S}\mathbf{U}^{n+\frac{2}{3}} - \mathrm{S}\mathrm{D}^{-1}\mathrm{B}^\intercal\mathbf{P}^{n+\frac{2}{3}} \\ =& \mathbf{U}^{n+\frac{2}{3}} - \mathrm{S}\mathrm{D}^{-1}\mathrm{B}^\intercal\mathbf{P}^{n+\frac{2}{3}}\end{aligned}$$

By applying the latter correction, by linearity of the operators and by choosing opportunely the Rhie-Chow stabilization matrix (in order to vanish the Schur complement see Remark 3.1), the pressure equation will read:

$$\begin{aligned}\mathrm{B}\mathbf{U}^{n+\frac{2}{3}} =& \mathrm{B}\mathrm{S}\mathrm{D}^{-1}\mathrm{B}^\intercal\mathbf{P}^{n+\frac{2}{3}} + \mathrm{C}\mathbf{P}^{n+\frac{2}{3}} \\ =& \mathrm{B}\mathrm{S}\mathrm{D}^{-1}\mathrm{B}^\intercal\mathbf{P}^{n+\frac{2}{3}} + (-\mathrm{B}\mathrm{S}\mathrm{D}^{-1}\mathrm{B}^\intercal + \mathrm{R}(\mathrm{S}\mathrm{D}^{-1}))\mathbf{P}^{n+\frac{2}{3}} \\ =& \mathrm{R}(\mathrm{S}\mathrm{D}^{-1})\mathbf{P}^{n+\frac{2}{3}}.\end{aligned} \tag{5.12}$$

As above, it is the algebraic counterpart of a Laplacian problem, with a modified diffusivity coefficient:

$$\nabla \cdot \mathbf{u}^{n+\frac{2}{3}} = \nabla \cdot \left(\mathrm{S}\mathrm{D}^{-1}\nabla p^{n+\frac{2}{3}}\right).$$

4. Pressure relaxation:

   $\mathbf{P}^{n+1}$ is then relaxed by a factor α:

$$\mathbf{P}^{n+1} = \mathbf{P}^n + \alpha_p\left(\mathbf{P}^{n+\frac{2}{3}} - \mathbf{P}^n\right).$$

5. Final correction:

   Compute $\mathbf{U}^{n+1}$ using equation (5.10).

Remark 5.1. The algorithm can be extended to the PIMPLE-IBM case adding the temporal discretization to the algebraic system 5.6. We add $\frac{1}{\Delta t}\mathrm{M}$ to matrix A and $\frac{1}{\Delta t}\mathrm{M}\mathbf{U}^n$ to source term **f**. The rest of the algorithm remains the same.

5.3 Solution algorithms for energy coupled systems

When dealing with polymer mixing applications, we have to take into account the role that temperature and shear have in the development of the flow field. We recall the governing equations (2.27):

$$\begin{aligned}\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} =& -\nabla p + \nabla\cdot(\nu(\dot\gamma, T)(\nabla\mathbf{u} + \nabla^\intercal\mathbf{u})) + \mathbf{f}, \\ \nabla\cdot\mathbf{u} =& 0, \\ \rho\frac{\partial c_pT}{\partial t} + \rho\mathbf{u}\cdot\nabla(c_pT) =& \rho r + \nabla\cdot(k\nabla T) + \mu(\dot\gamma, T)(\nabla\mathbf{u} + \nabla^\intercal\mathbf{u}) : \nabla\mathbf{u}.\end{aligned}$$

Momentum and temperature equations are strongly coupled by both the non-Newtonian rheology and by the viscous heating source term. This makes the problem highly nonlinear, which requires suitable strategies in the solution procedure to achieve convergence, in particular way where a segregated approach is employed.

The behaviour of the solver when dealing with this system of equations depends especially on the viscosity, that represents the strongest link between momentum and energy. In particular, the problem is tougher to solve as viscosity magnitude and shear thinning (or thickening) increase. To have an idea of this behaviour, consider a power law fluid (2.40) in which $K = 1$, $\dot{\gamma} \in (0, 100)$ and $n \in (0, 1)$. As it can be seen in Figure 2.1a, if n decreases, viscosity reaches higher values faster when shear rate decreases and vice-versa.

If we just solve first the velocity, then pressure and then temperature and we loop this process until convergence, we run into numerical instabilities, especially when there is a significant shear thinning effect (n values lower than one). This because in the first iterations, velocity has not a fully developed profile, due to relaxation, and high gradient values can occur especially near boundaries. This implies high viscous heating and high temperature and high temperature implies low viscosity and so high velocity gradients and so on and so for.

Here we proposed a modified SIMPLE algorithm that we called SIMPLEX (i.e. SIMPLE-ECS, SIMPLE for Energy Coupled Systems), that has proved to be stable and robust when considering viscosities presenting real industrial rheologies. The flow chart is represented in Figure 5.1.

The concept is to start solving a Newtonian problem with $\nu = K$, so as if $n = 1$. We solve, up to a certain tolerance, momentum and pressure equations, just like in the original SIMPLE. Then we update viscosity value, we solve temperature and we again update the viscosity value. We call this inner step X-loop.

The tolerances set to stop X-loops are higher than the one of the global problem. In this way, the solver spends more iterations in solving the initial Newtonian problem, that is linear and easier to solve, and in the end it spends fewer iteration to solve the non-Newtonian problem with non-uniform viscosity problem, giving more importance to the viscosity update.

This procedure has given good results in terms of robustness and also in terms of computational time. At the beginning, more iterations are performed in the inner loop, but the effort is posed on a simple and linear problem. At the end, less iterations are spent on more difficult and nonlinear problem, but with guaranteed stability.

For time dependent simulations we developed the PIMPLEX algorithm, which flow chart is represented in Figure 5.2. The concept of the PIMPLEX algorithm is the same of the PIMPLE one. At each time step we solve iteratively the problems employing the SIMPLEX algorithm to reach the solution at time t^{n+1}. This algorithm is particularly useful to robustly deal with IBM because when the immersed surface moves, it can happen that solid cells become fluid cells and so the solution is undetermined at the beginning of the time step. Using the PIMPLEX algorithm avoid the raise of possible numerical instabilities.

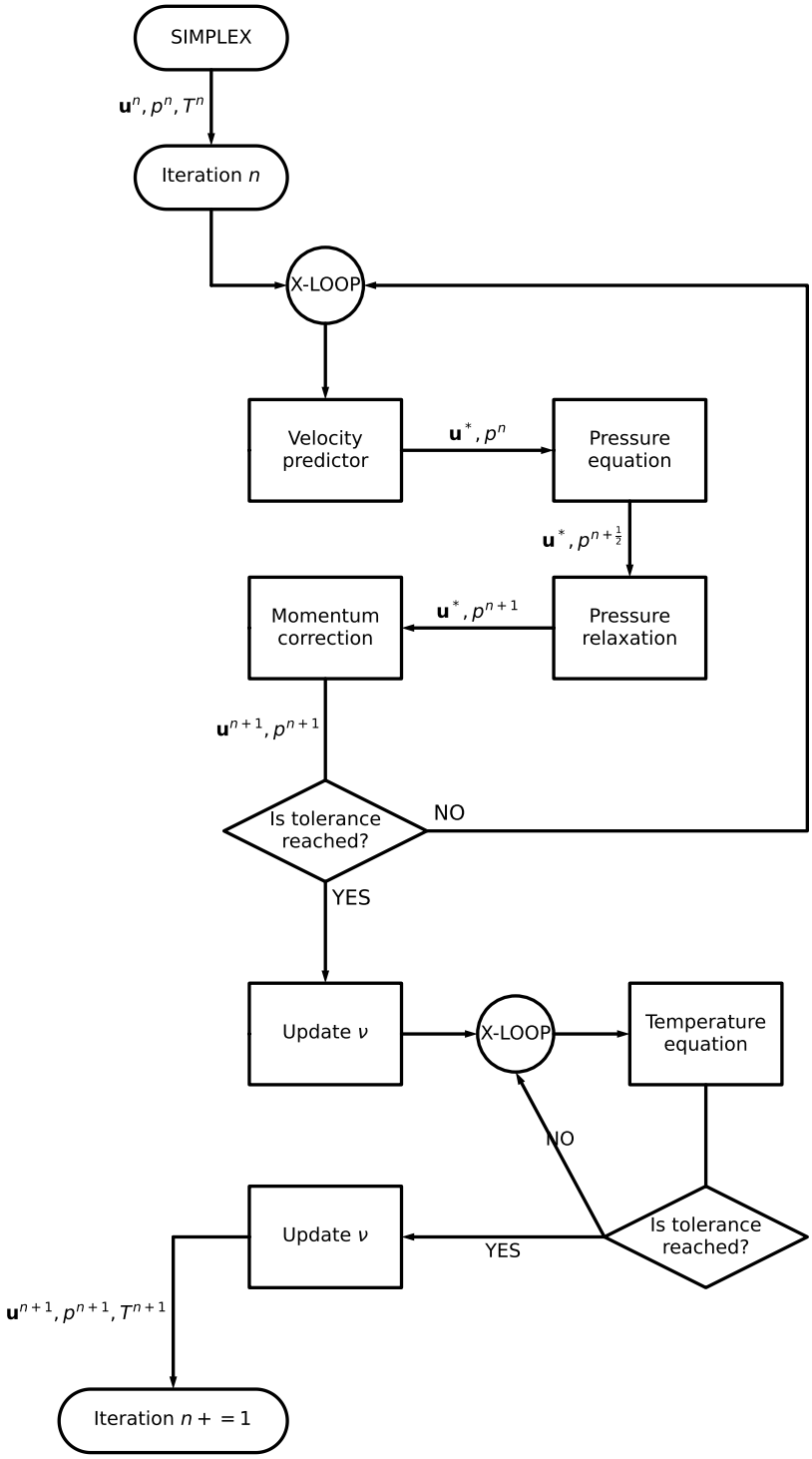


Figure 5.1: SIMPLEX diagram for the solution of iteration $n+1$.

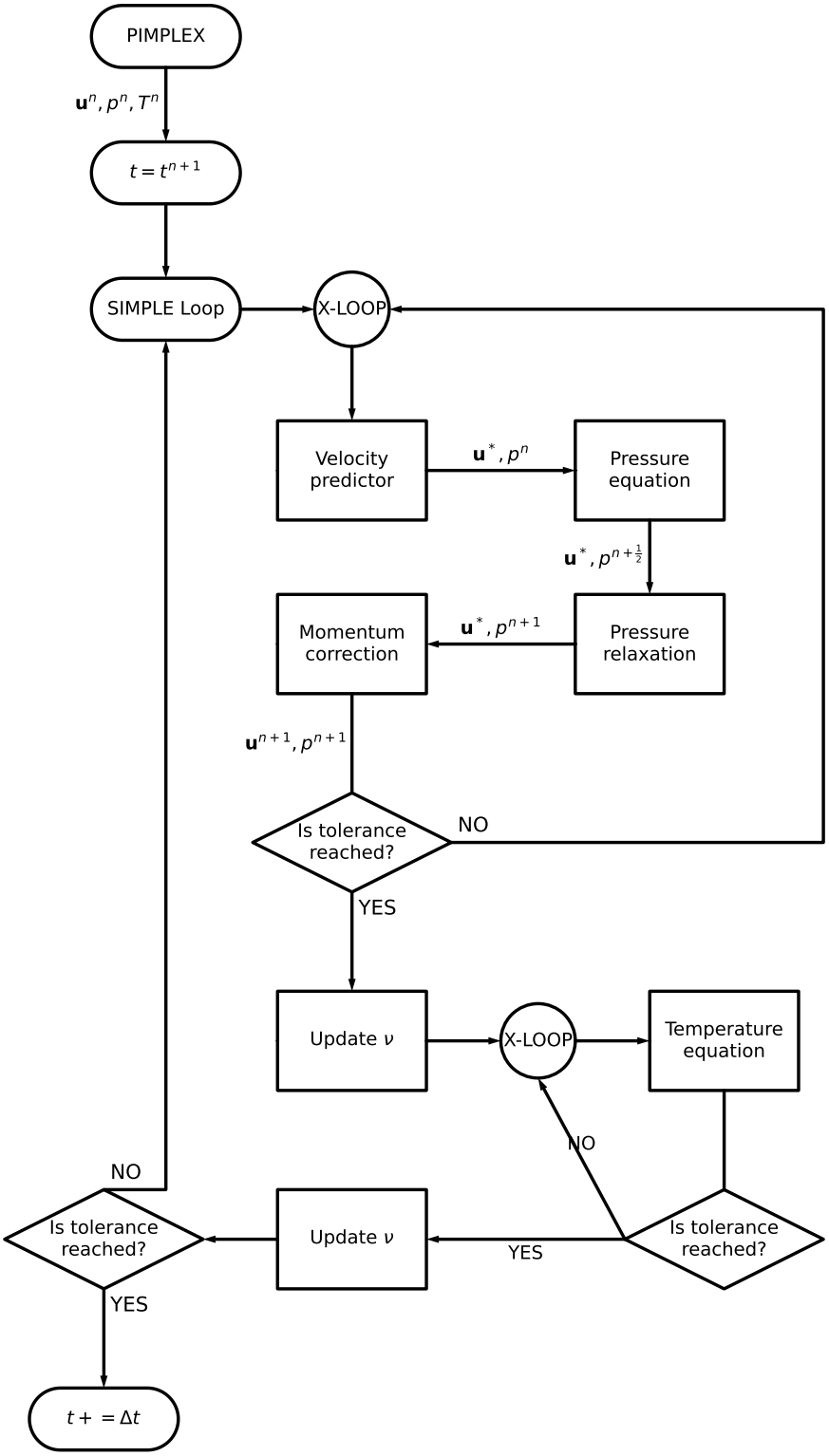


Figure 5.2: PIMPLEX diagram for the solution of time step t^{n+1}.

Material			Power law			Arrhenius law	
ρ	c_p	k	K	n	λ	α	T_α
1200	2000	0.2	100-400	0.2-0.5	1	3000	350

Table 5.1: Model material rheology parameters.

5.3.1 Robustness assessment

To assess the robustness of this algorithm we performed a sensitivity analysis. We considered a benchmark test case, in particular the Simplified SSE described in Section 4.7. We selected some parameters of this test case and we varied them in order to test the robustness of the SIMPLEX algorithm during the first iterations.

We set the pitch of the screw at 30mm and considered a single geometrical period. We set a flow rate at the inflow and no slip velocity on screw and barrel. Then we set zero mean pressure at the outflow. For the temperature we imposed a uniform value at the inflow of 300K, the same value for the barrel and 293K on the screw.

We take also a model material representing a high viscosity power law polymer with Arrhenius law for temperature dependence. The values are reported in Table 5.1.

Then we choose some parameters to vary. The clearance between the screw crest and the barrel, the consistency factor and the exponent of the power law, the rotation velocity of the screw and the number of sub-iterations of the X-loop.

Maximum temperature for each parameter combination is reported in Figure 5.3 while the boxes that are blanked out correspond to divergent simulations. At a first glance, problems solved with one X-loop sub-iteration (the first three rows starting from the top) are more sensible with respect to the nonlinearity. Then, with 10 sub-iterations we recover the stability in the cases where the rotational velocity is set to -50 RPM, meaning lower shear rate. Finally, employing 100 sub-iterations we are solving robustly all the problems in the range of parameters that has been considered.

In conclusion, employing the SIMPLEX algorithm for such problems makes the results stable.

Remark 5.2. The last column cases with clearance set to 0.0001 m are the most delicate because the mesh significantly deforms and there is a complex interplay between mesh quality (c.f. Section A.1), shear rate and non-Newtonian degree.

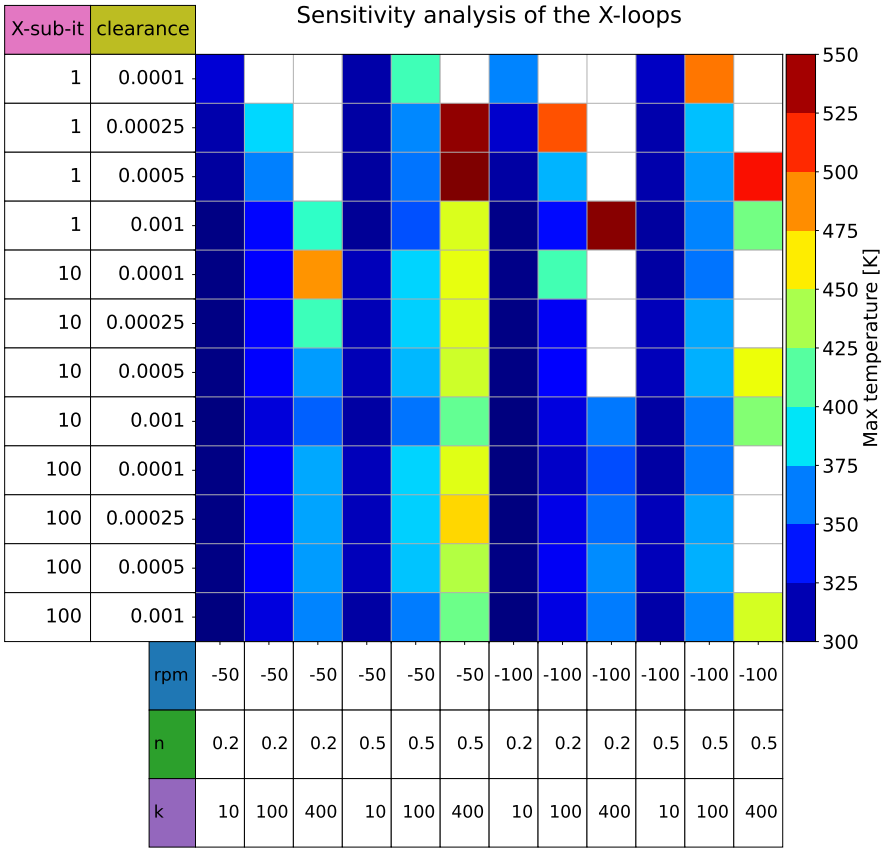


Figure 5.3: Maximum temperature values for each combination of parameters after 20 SIMPLEX iterations. Boxes correspondent to divergent simulations are blanked out.

5.4 POLIMIX code structure

Here we give an overview of the code structure and of all the novel features with respect to the base OpenFOAM code. The toolbox is built along with the OpenFOAM-7 version and can be compiled without modifying any line of the base source code. We adhered as closely as possible to coding style rules and base code structure. Here we review the main structure of the base code and then we enlighten the most important features of our implementation.

The new features that we implemented regard the resolution of polymer processes, CFD problems with incompressible non-Newtonian fluid with temperature dependent viscosity, and in particular the Immersed Boundary method.

We started from considering built-in non-Newtonian laws. They are implemented as `transportModels`, a library that models to change the viscosity value according to a rheological law such as power law, Bird-Carreau or Cross laws. Considering this framework, we added the possibility to include temperature dependence and also filler dependence to the viscosity law. Additionally, we implemented also temperature dependence for other physical parameters like density, thermal conductivity and heat capacity. This framework has been implemented under the name of `mixingTransportModels`, whose relation with other classes is represented in Figure 5.4. Here, `transportModel` and `viscosityModel` classes are native of OpenFOAM, while `thermoDependentViscosityModel`, representing the viscosity laws with filler, temperature and shear rate dependences, and `thermalModel`, the material properties (density, diffusivity, heat capacity), are new implementations. Moreover, `thermo DependentViscosityModel` is linked to two other classes: `thermoDependenceModel`, that computes the temperature shift factor, and `fillerFractionModel`, that includes the dependence from filler volume fraction of the compound.

Our implementation relies on the OpenFOAM code structure and also to the user-interface structure, the so called `runTimeSelectionTable`, to allow the user to arbitrarily choose every possible combination of shear, temperature and filler dependence. Moreover, the viscous heating source term was included through the `fvOptions` framework, that is the framework that allows the user to choose source terms to be applied to problem equations.

The second important implementation is the one regarding the X-loops framework (c.f. Figure 5.2). We considered the library that controls the convergence tolerance and time discretization, i.e. `solutionControl`, and we integrate the sub-iterations in the `simple Control` and `pimpleControl`, in order to let the user choose convergence parameters using the same interface of the built-in version. Class dependences are represented in Figure 5.5.

Finally, to describe the implementation of the IBM library, a brief discussion on how boundary conditions are dealt in OpenFOAM is needed. Boundary conditions in OpenFOAM are defined by two objects: a geometrical object, the patch, and a physical object, the boundary field. The patch, defined by `polyPatch` and `fvPatch` classes, contains all the geometrical information about face areas, normal and interpolation weights and it is unique for each field. The class of the boundary field, `fvPatchField`, is instantiated for each field (velocity, pressure, temperature, gradients, viscosity, etc.) and contains information about the type of boundary condition (Dirichlet or Neumann or Robin) for the specific field. The boundary field has the task to modify the FVM matrix including the specific constraints, associated to the field boundary condition.

To implement our IBM, we attained to this exact structure to make the code flexible and readable and editable by any OpenFOAM developer. This implementation is also very intuitive because it works as any other conforming boundary condition also from the point of view of the user-interface. Moreover, all the tuning parameters can be chosen by the user in a dedicated dictionary file in the case folder.

To describe an immersed boundary condition, as the official OpenFOAM release, we have a geometrical patch, defined by `ibPolyPatch` and `ibFvPatch` classes. They read the external STL files, divide the computational mesh in solid, fluid and IB regions and set all the necessary to perform the WLS interpolation. In particular, it finds the IB cells, computes the IB points and IB normals, generates the cell stencils, initializes the WLS interpolators (c.f. Section 4.2). Stencil search is the most important function because stencils have to be built in the neighbouring of the IB cells but also in other processors, when running in parallel. Another feature resides in the IB interpolator framework, that consists in the abstract class `ibInterpolator`. this framework is very flexible: the structure of the code allows to implement custom interpolators besides the WLS and the type of interpolation can be selected by the user in the already mentioned IB dictionary (Figures 5.6 and 5.7).

On the other hand, the IB boundary fields, implemented in the template class `ibFvPatch Field`, make all the work of the imposition of the embedded boundary condition to the solution. They register the original boundary condition, defined on the STL surface, and then they use the interpolators on the patch to make the IB correction on the field at hand (Figure 5.8).

The last feature of the IBM library is the handling of time dependent problems. The library, called `immersedBoundaryDynamicFvMesh`, is able to move STL surfaces just as the built-in library, called `dynamicFvMesh`. Moreover, it is able to perform dynamic local mesh refinement based on the position of the immersed surface. It is fully integrated with the built-in library and can handle both IB surface motion and built-in mesh motion together with the same user-interface framework. In particular, we considered the native class `dynamic MotionSolverFvMesh` that handles rigid body motions and we integrated the IBM through the class `movingIb`, that manages to move the surface triangulation and to compute the surface velocity, as described in Section 4.3 (Figure 5.9).

All these features have been included in new solvers, deriving from the original `simple Foam` and `pimpleFoam` solvers: `simplexIbFoam` and `pimplexIbFoam`, where the `x` indicates the inclusion of X-loops and temperature equation.

In conclusion, the tool that has been implemented inherits the flexibility in usage and code development of the original OpenFOAM distribution. This makes the POLIMIX library a tool that can keep pace with the development of the main platform.

In the following sections, we present some applications of our toolbox to real industrial problems, demonstrating its efficiency and the wide variety of problems to which it has been successfully applied.

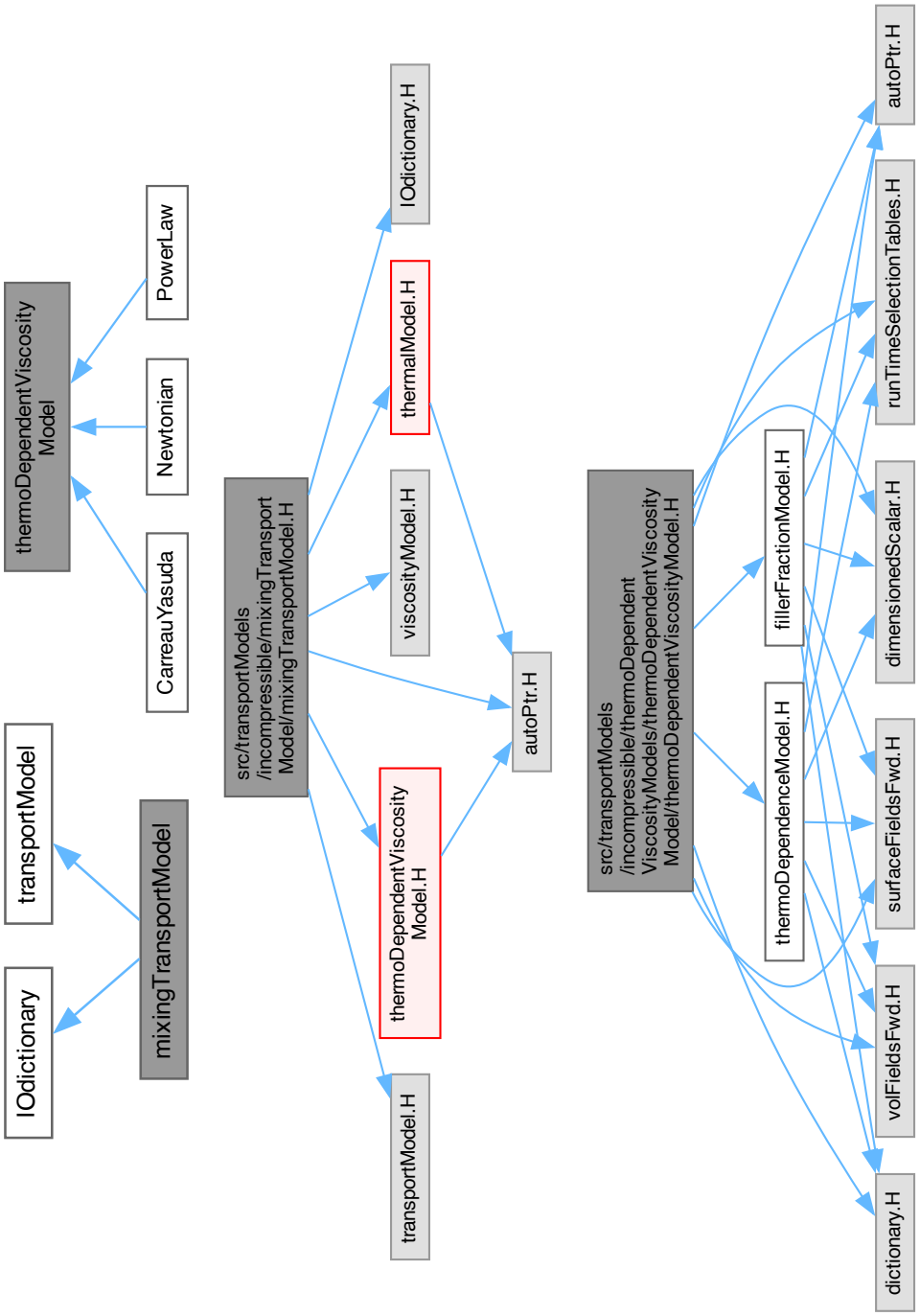


Figure 5.4: `mixingTransportModel` class inclusion.

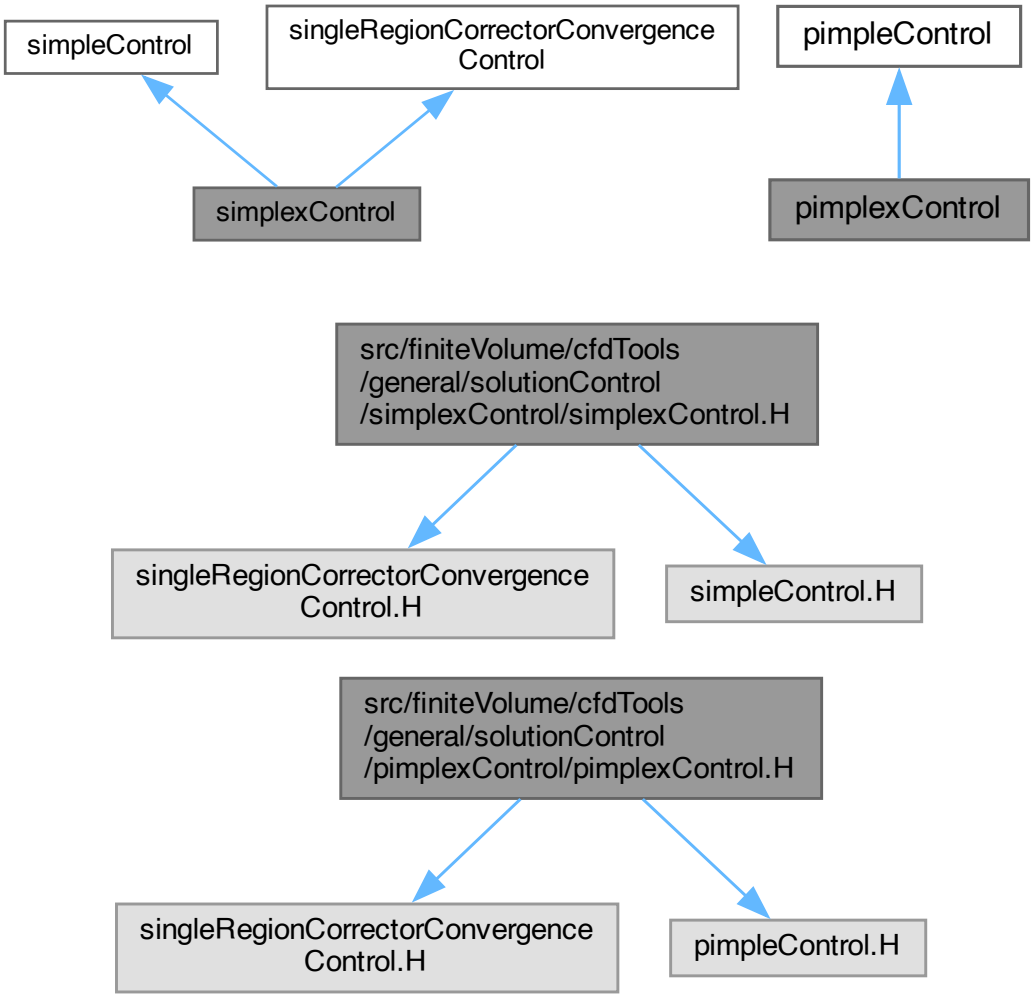


Figure 5.5: `simplexControl` and `pimplexControl` classes inclusion.

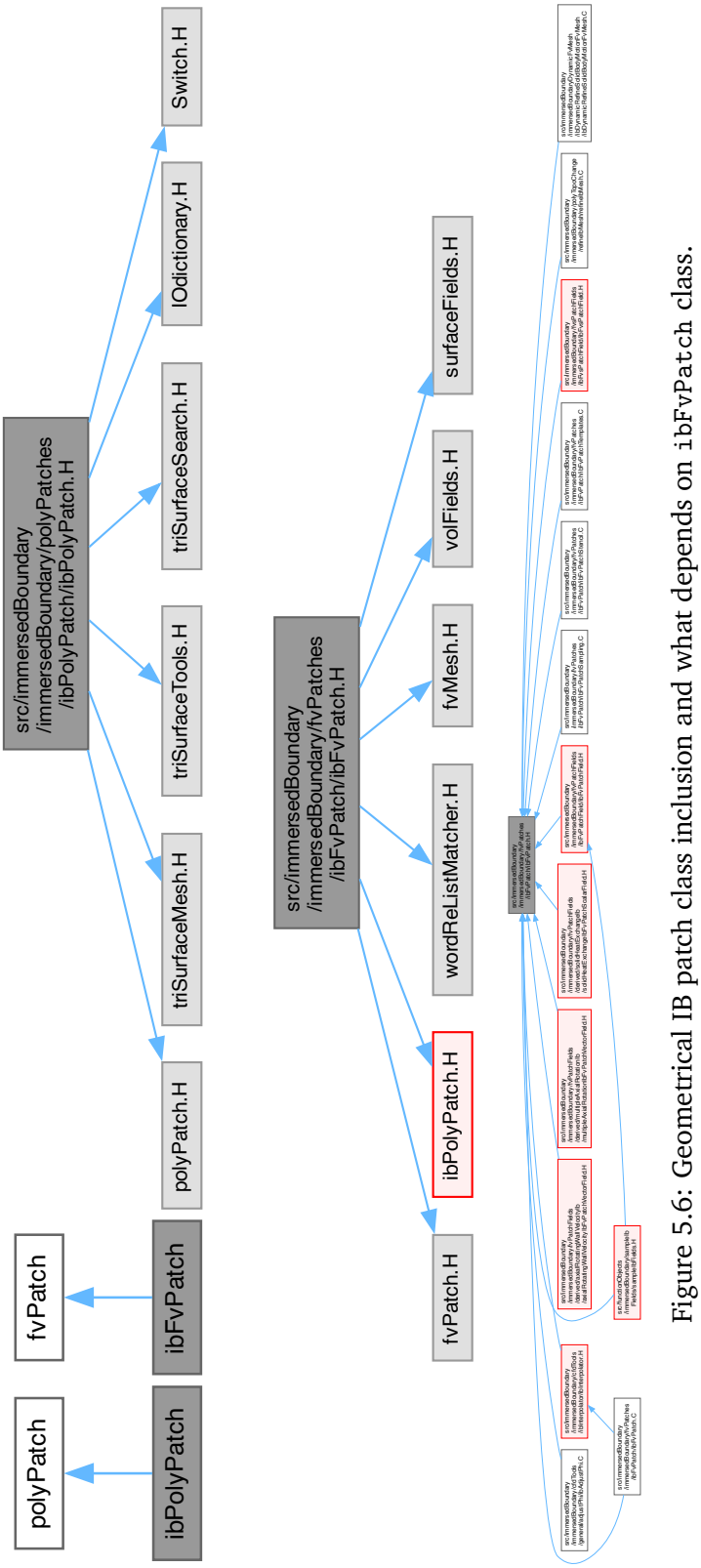


Figure 5.6: Geometrical IB patch class inclusion and what depends on `ibFvPatch` class.

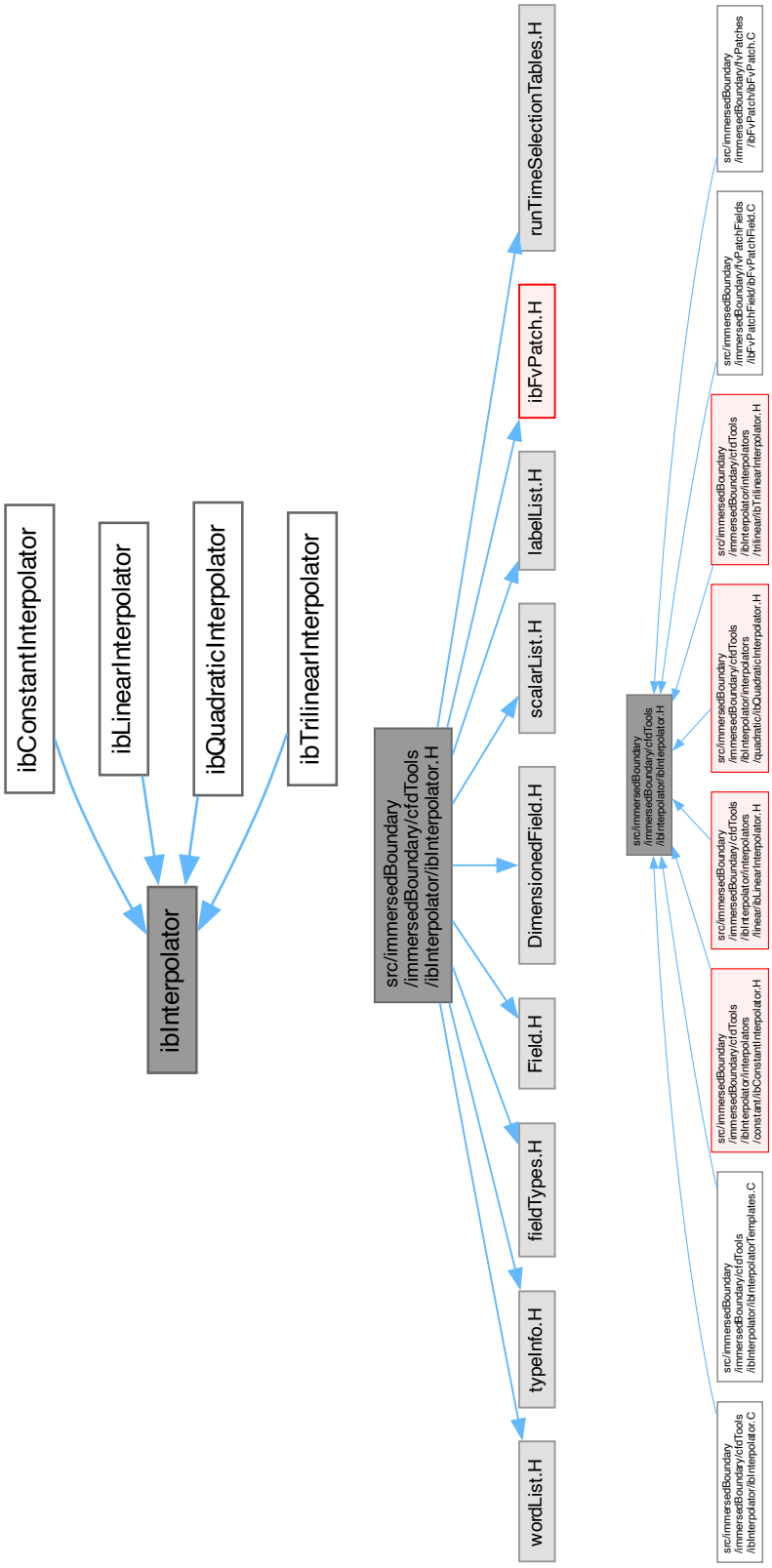


Figure 5.7: IB interpolator framework class inclusion and what depends on `ibInterpolator` class.

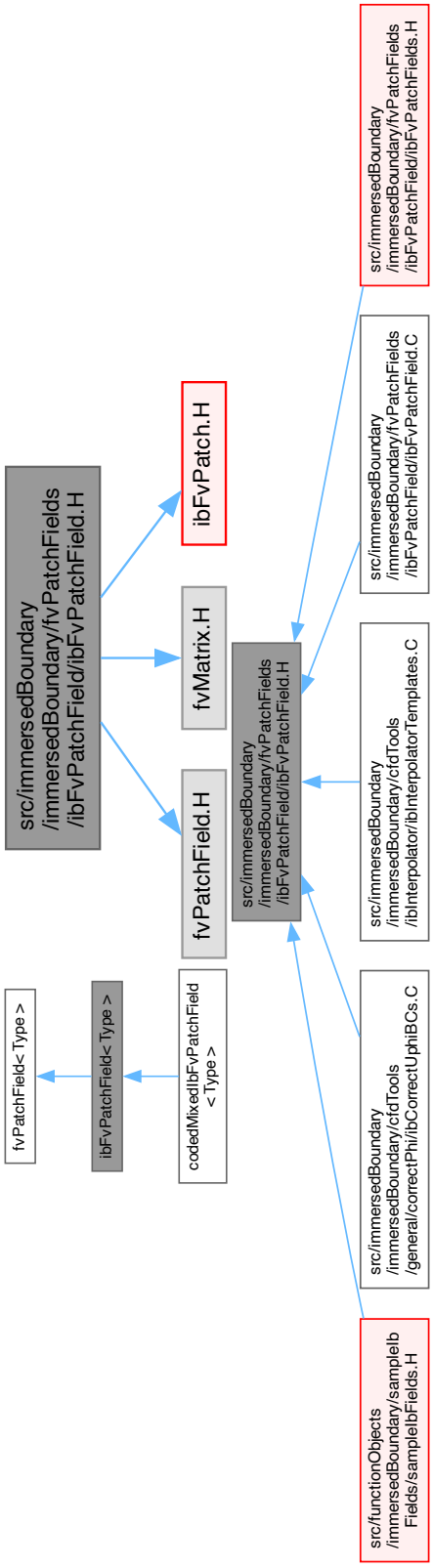


Figure 5.8: IB patch fields what depends on `ibFvPatchField` class.

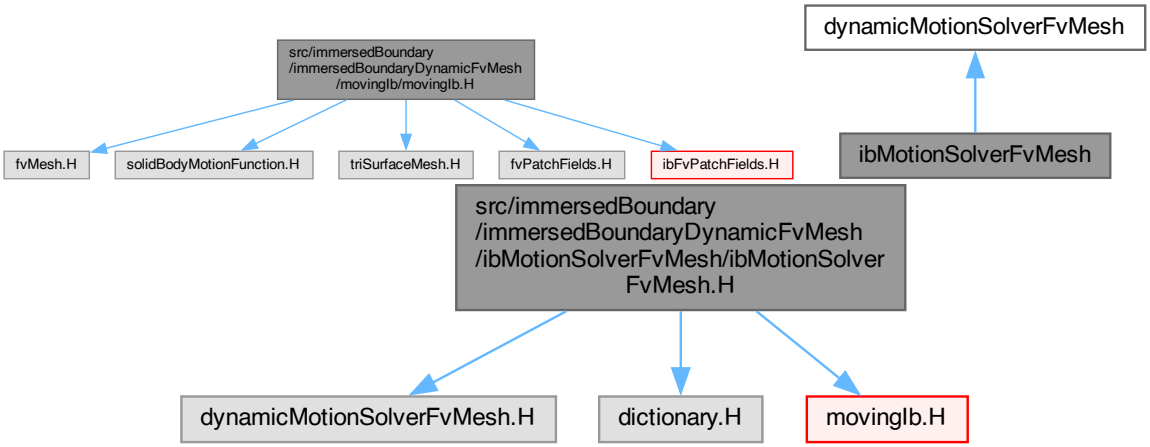


Figure 5.9: IB patch fields class inclusion and what depends on `ibFvPatchField` class.

Chapter 6

Application to industrial problems

In this chapter, we apply the approach described in Chapter 4 and we show that it is able to deal with a wide range of types of geometry and application in the field of polymer processing. We adopted, as described in Chapter 4, a non-conforming approach, hence we considered simple background meshes and we account for the presence of the screws through immersed boundaries described by embedded STL surfaces. Here we present the work that has been done to develop a robust tool to face applications to real industrial geometries of mixing devices. In particular, we consider three different technologies: single- and twin-screw extruders and planetary roller extruder. For each of them we discuss the main criticalities of the application and how it has been faced in practice, employing effective strategies in the development of the tool.

In every case-study we consider a high-viscosity power law polymer with WLF law for temperature dependence (see equations (2.40) and (2.45)). Approximately, the consistency factor of the power law is of the order of $10^2\text{m}^2/\text{s}$ while the exponent is $n \simeq 0.2$. For the WLF law, we considered the activation energy of the order of 10^3K and the reference temperature $T_\alpha \simeq 370$K. For the other material properties we have considered typical values for the rubber: the density is around 10^3kg/m^3, the diffusivity 10^{-1}W/mK and the heat capacity around $2 \cdot 10^2$J/KgK.

These parameters give the idea of the problem complexity. The power law exponent makes the flow problem highly nonlinear and difficult to solve because of great variations of the kinematic viscosity with respect to small variations of shear rate (Figure 2.1a). The other difficulty is that, despite having a very low Reynolds number (commonly between 10^{-4} and 10^{-5}), the problem in temperature is strongly dominated by advection, indeed, the Peclét number of the problem is between 10^5 and 10^6.

A strong assumption has been made for the simulations described in next sections, that is that the flow is single-phase, hence the device is completely full of fluid. This is a common assumption among many literature works despite real processes involve multiphase flows of polymer melt and air.

Claim: the geometries and rheological laws presented here are provided by an industrial partner, for this reason, numerical values of geometrical dimensions and parameters are not reported explicitly, but only in terms of orders of magnitude.

Remark 6.1. All the simulations have been carried out using the open-source CFD library OpenFOAM and an in-house developed toolbox, customized for polymer processes applications, called POLIMIX (PoliMi/polymer Mixing) (see 5.4). In particular, mesh generation was performed using the OpenFOAM tool `blockMesh`, a multi-block hexahedral mesh generator able to build high quality grids. Instead, the generation of IB STL surfaces was performed using the python interface of GMSH or built-in OpenFOAM executable to extract surface triangulations from volume meshes.

In sections 6.1, 6.2 and 6.3 we analyse the results obtained on complex industrial applications: the single-screw, twin-screw and planetary roller extruders, respectively. We report both the numerical solutions obtained from the simulations and results in terms of parallel scalability of the code and numerical performance.

6.1 The Single Screw Extruder

The single-screw extruder (SSE) is the simplest geometry one can consider among continuous mixing devices. It is composed by the barrel and by one screw. The basic operation of a single-screw extruder is rather straightforward. Material enters from the feeding section hopper and flows down into the extruder barrel. The barrel is stationary and the screw is rotating. As a result, frictional forces will act on the material, both on the barrel as well as on the screw surface. These frictional forces are responsible for the forward transport of the material. As the material moves forward, it will heat up as a result of frictional heat generation and because of heat conducted from the barrel heaters. The material then accumulates in the metering section before being pushed through the die.

The geometry that has been considered in our simulation is presented in Figure 6.1. The device is composed by two concentric axisymmetric surfaces. The external barrel is a cylinder of diameter D (of the order of dozens of centimetres) and length 10-15D. The internal surface is the shaft, that is divided in three regions (from left to right in Figure 6.1): the feed, the one with the largest fluid section, compression, the transitional one, and metering, the one after the restriction between barrel ans shaft. On the shaft, the so called screw teeth are attached. Here, we consider teeth with a rectangular shaped section that is then extruded by a helical path along the axial direction of the device (). The two teeth are symmetrical with respect to the axis and have a thickness of dozens of millimetres and form a gap of tenths of millimetres with the barrel, four orders of magnitude less that the device length. Working conditions are between 10∼100RPM. We consider the Immersed Boundary

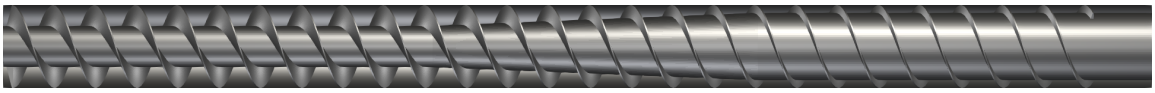


Figure 6.1: SSE rendering.

method to embed the presence of screw teeth, that are the objects that make the geometry difficult to approximate with a conforming grid.

The barrel and the shaft will be conformingly described by the grid because of their simple shape. The strategy to mesh these parts is the following: we generate an azimuthal

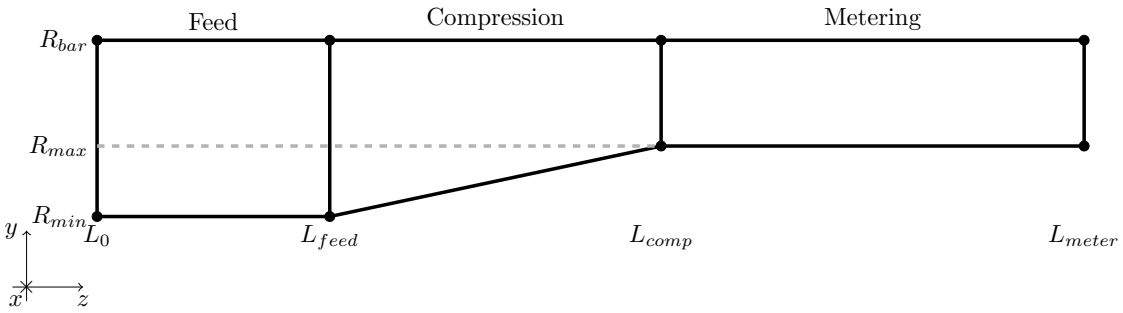


Figure 6.2: Block scheme used by `blockMesh` for the azimuthal section of the SSE (proportions shrinked along z).

section of the grid and then we extrude this section for a complete round. The block scheme of the azimuthal section is showed in Figure 6.2.

As mentioned above, the teeth are generated by extruding a rectangular shaped section on zy-plane with the following parametrization:

$$\begin{cases} x = & r\cos t, \\ y = & r\sin t, \\ z = & \dfrac{L_{pitch}}{2\pi}t, \end{cases} \qquad t \in (0, 2\pi N_{period})$$

where L_{pitch} is the length of the screw pitch. and $N_{period} = L_{meter}/L_{pitch}$.

The mesh discretization is then automatically performed defining a characteristic length h and multiplication factors. Moreover, to optimize the distribution of mesh elements, a strategy to automatically set mesh grading has been implemented. Given the presence of particularly narrow gaps, a rigorous procedure was implemented to place a prescribed number of elements within the gap. Grading level, or expansion ratio, $g \in \mathbb{R}^+$ is defined as the quotient between the length of last and first elements (Figure 6.3).

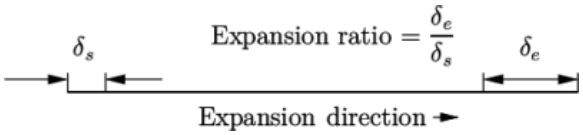


Figure 6.3: Mesh grading along a mesh edge.

Every block edge is defined by a length L_e, the element number N_e, the expansion factor a_g associated to grading level g. The expansion factor a_g is defined as

$$a_g = g^{\frac{1}{N_e - 1}}.$$

Then, the position of the mesh vertices along the block edge can be computed as

$$x_k = x_0 + L_e \frac{1 - a_g^k}{1 - a_g^{N_e}}, \qquad k = 0, \ldots, N_e,$$

where x_0 is the position of the first node of block edge. We then define δ_k as the edge element width:

$$\delta_k = x_{k+1} - x_k = L_e \frac{a_g^k - a_g^{k+1}}{1 - a_g^{N_e}}.$$

Finally, we set N_{gap} as the number of elements that we want inside the gap of width L_{gap}. The grading level that we have to set to satisfy this condition (assuming the gap is at the beginning of the block edge) is the solution of the following nonlinear equation:

$$L_e \frac{1 - g^{\frac{N_{gap}}{N_e - 1}}}{1 - g^{\frac{N_e}{N_e - 1}}} - L_{gap} = 0, \tag{6.1}$$

that is solved using the Newton method. The result is resumed in Figure 4.5.

6.1.1 Dynamic local mesh refinement

OpenFOAM is equipped with an Adaptive Mesh Refinement (AMR) library able to efficiently refine hexahedral grids, both statically and dynamically. The refinement is always performed using hanging nodes, so each hexahedron is split in eight parts by connecting face barycentres. Because of the flexibility that finite volumes have on polyhedral grids, the handling of such refinement procedure is performed efficiently.

In order to better capture the geometry of immersed boundaries, it could be useful to have the possibility of refining in that region. What we did was integrating the library that we developed with the OpenFOAM framework of dynamic mesh refinement.

The refinement criteria are straightforward to choose. We implemented basically two criteria, both distance based. The first criteria selects all cells within a certain spatial distance from the surface. The second selects all cells contained in level c stencil $\mathcal{K}_i^c$, where $K_i \in \{K \in \mathcal{T}_h : K \cap \Sigma \neq \emptyset\}$, so at a certain connectivity distance from the cells cut by the surface.

An example of dynamic refinement on the SSE test case is presented in the series of images of Figure 6.4. Here we applied two nested refinements until the third stencil level.

6.1.2 Results

We present a steady-state simulation of a SSE filled with a high-viscosity power law polymer. The working conditions are the following. The flow is driven by the screw rotation, set at 30RPM, so we set zero pressure ant inflow and outflow. On barrel, shaft and screw teeth we set no slip condition for velocity. The barrel and screw temperatures are set to 323K, as the inflow temperature.

In particular, for this test case, we compared results from both Diffuse Interface and Immersed Boundary methods, introduced in Chapters 3 and 4, to have an idea of how they behave when applied to real industrial applications.

To run these simulations, we have employed a single rotating frame (SRF) to perform a steady-state simulation. We set our frame of reference on the screw, hence from this point of view, the screw remains fixed while the barrel rotates with -30RPM. So, we solved the non-inertial Navier-Stokes equations (2.32) and then we converted the obtained relative velocity

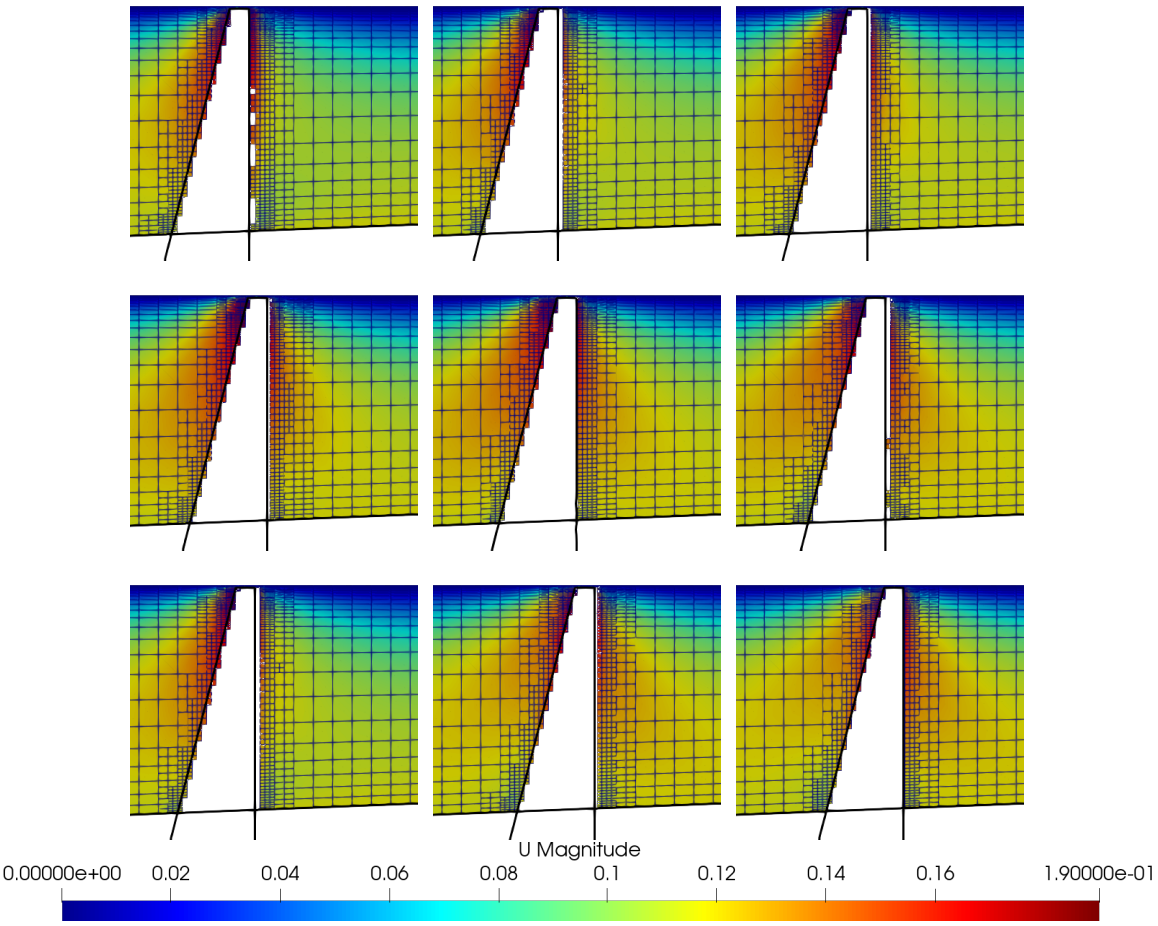


Figure 6.4: Sequence of images representing the evolution of the grid while the screw profile (the white line) advances. The figure represents an azimuthal section with normal directed along x axis.

to the absolute one. Moreover, we employed the SIMPLEX iterations in order to guarantee the stability of the solution.

Remark 6.2.

We have reported some results regarding the two numerical solutions obtained with DIM and IBM in Figures 6.5 and 6.6.

For what regards velocity, the effect of the constant extrapolation, used by the DIM to impose the immersed boundary condition, is visible in the velocity profile near the gap between the barrel and the screw. The velocity value in this region is greater than the one of IBM that, using a quadratic extrapolation, is sharper. For the DIM, this can be interpreted as the effect of "diffusing the interface". On the other hand, the fact that the IBM solution profile is sharper is eventually the gain in accuracy of the IBM with respect to the DIM, showed in Section 4.7.

This difference in velocity profiles reflects also on differences in terms of pressure and temperature. The pressure profile is coherent with the geometry in both cases (see top of Figure 6.6): the pressure raises until the end of the compression region, where it has its peak, then it decreases until the outlet. Pressure gradients are significantly milder in the DIM case than in IBM case. This is due to the fact that the DIM velocity solution shows higher velocity values, that justifies the lower pressure gradients needed to transport mass along the screw axis.

The same considerations can be done on temperature profiles. Looking at Figure 6.5 (bottom) and Figure 6.6 (bottom), temperature is much more distributed in the SSE cavity, being the velocity higher just outside the screw tooth surface. Moreover, in this way, the shear near the tooth is higher. This generates a higher viscous heating and consequently a higher value of temperature near the tooth with respect to what happens in the IBM simulation.

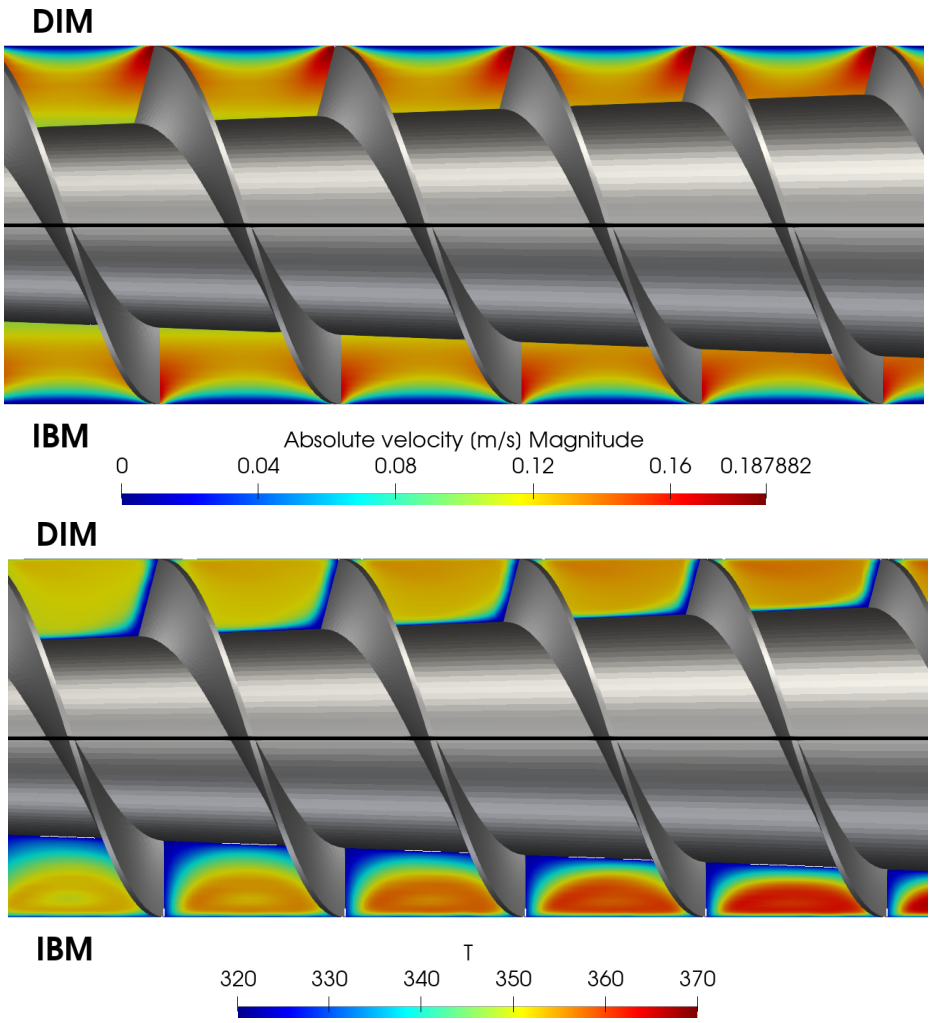


Figure 6.5: Representation of velocity magnitude (top) and temperature (bottom) fields distribution on an azimuthal section of the transitional sector of the SSE. Both DIM and IBM solutions have been represented on top and bottom parts of the extruder channel, respectively.

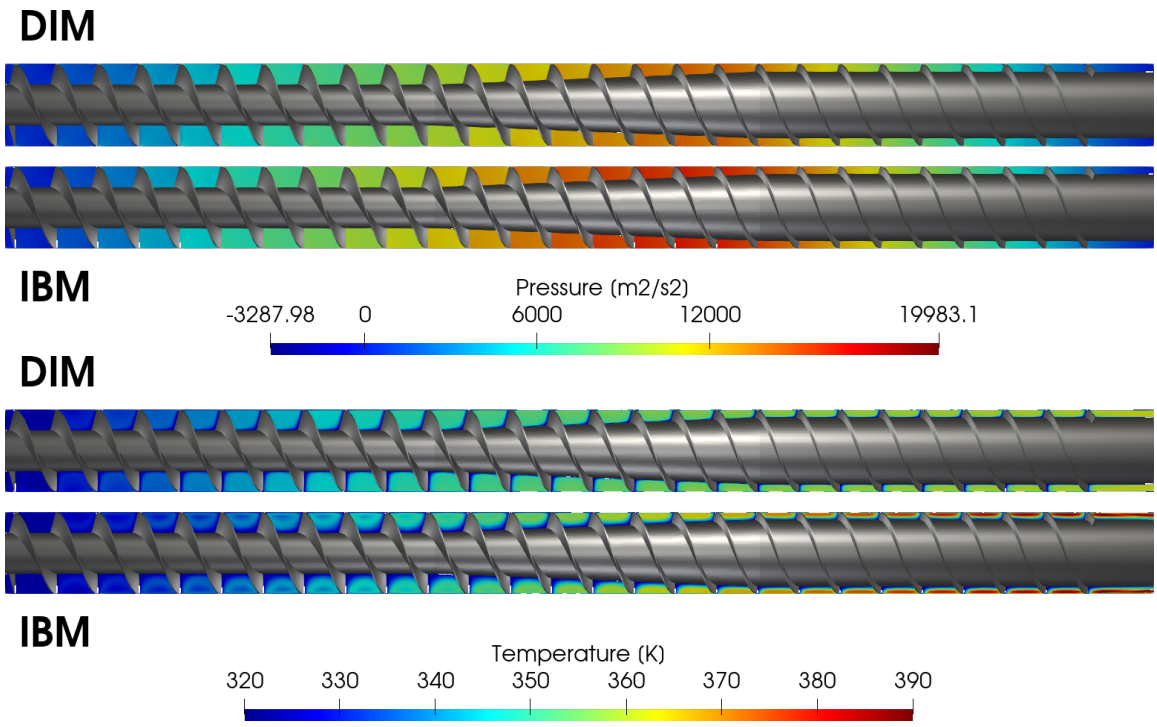


Figure 6.6: Representation of pressure (top) and temperature (bottom) fields distribution on an azimuthal section of the SSE.

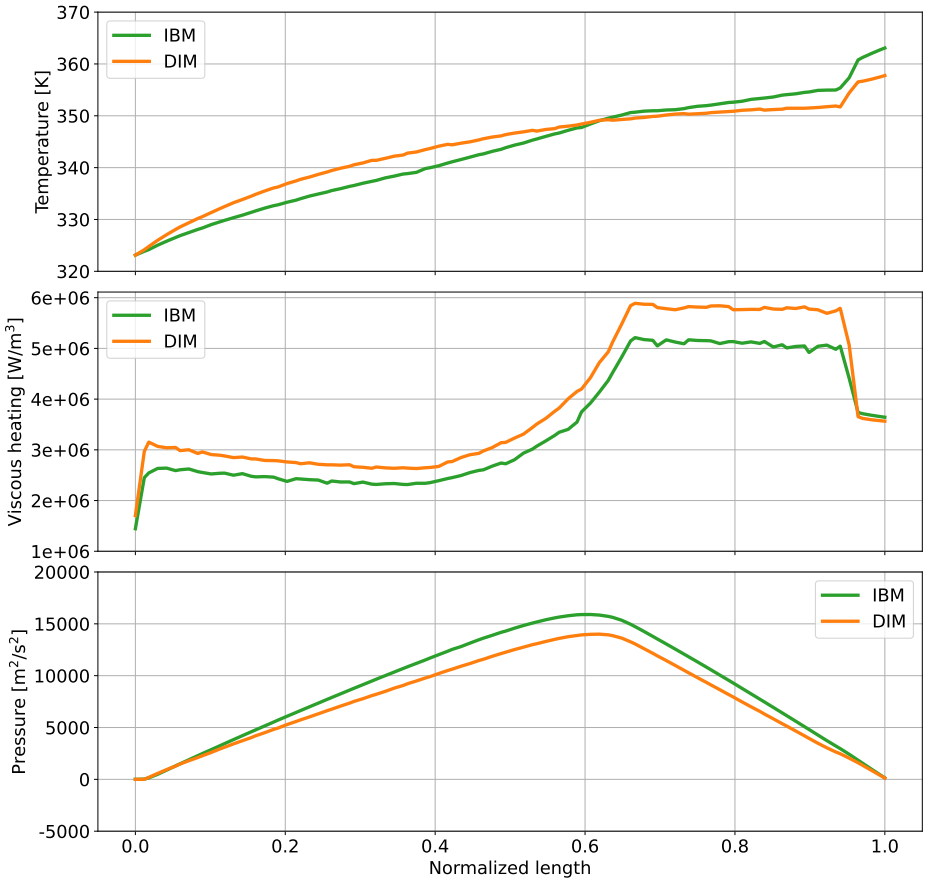


Figure 6.7: Pressure, viscous heating and temperature sectional averages for DIM and IBM solutions.

6.1.2.1 Influence of filler fraction

In this section, we performed a numerical experiment in which we considered the fluid which viscosity is characterised by the same power law of the other experiments in this chapter and we vary the dependence of the viscosity from the filler. We remark that the power law parameters, employed for the numerical experiments of this chapter, correspond to a filler fraction ϕ_f=0.05.

In particular, we considered a Maron-Pierce dependence (equation 2.50) with a maximum volume fraction $\phi_M = 0.35$ and the filler fraction of the fluid $\phi_f = 0.03, 0.05, 0.07$). For each value of ϕ_f we ran a simulation with the same setup of previous section.

To compare the results we computed the sectional averages of temperature, viscous heating and pressure to see how they change with respect to the amount of filler present in the compound. The averages are reported in Figure 6.8. As expected, a higher filler concentration makes the compound harder to be pushed towards the outflow because the value of viscosity increases. Also the shear increases and consequently the viscous heating and the temperature.

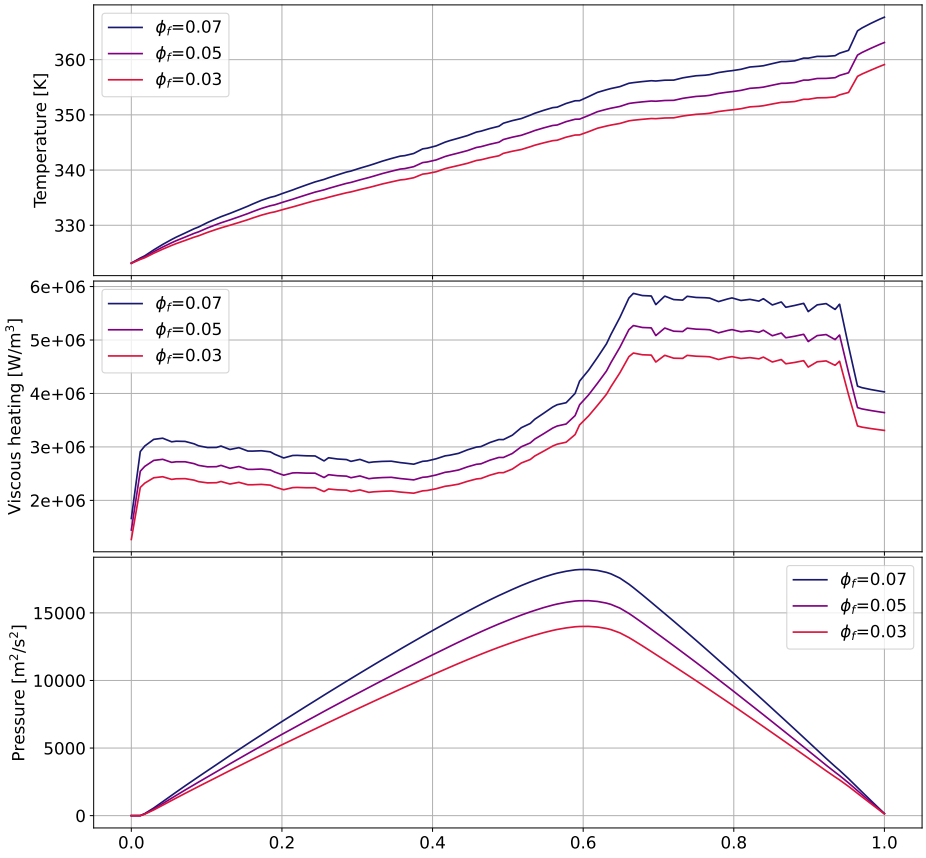


Figure 6.8: Pressure, viscous heating and temperature sectional averages for three levels of filler volume fraction.

6.1.2.2 Parallel scalability analysis

In order to analyse the parallel performances of our code, we consider the single-screw extruder test case introduced in the previous section as a representative large scale problem of interest. Scalability is a crucial aspect of our implementation for two reasons. The first is related to the type of processes that we want to simulate, that in general requires a large number of degrees of freedom to have a good approximation of the physical phenomena and to correctly approximate IB surfaces.

The second is associated to the fact that a method like the IBM presented in Section 4.2, typically requires more parallel communications than a standard method. Indeed, it can happen that an IB cell is near to a decomposed domain interface. If this is the case, we have to look for stencil cells also in the other processor. Hence, every time that we have to perform the WLS interpolation, we have to retrieve cell values from other processors. So, verifying at which extent our method shows parallel scalability properties is crucial to guarantee the possibility to use the method in solving real industrial problems.

We consider the same SSE test case of previous section in its steady-state version and performed a scalability analysis on the CINECA HPC Galileo100, which consists of 554 computing nodes each 2 x CPU Intel CascadeLake 8260, with 24 cores each, 2.4 GHz, 384GB RAM.

We consider a mesh of $\simeq$22 millions of cells and we run simulations for 50 steps of SIMPLEX algorithm parallelizing the case from 2 processor to 2048 processors. We define the speedup s_n of a simulation with n cores as

$$s_n = \frac{T_{n_0}}{T_n},$$

where T_n is the wall time of simulation n and n_0 is the simulation with the smallest number of cores. We also consider both IB and DI methods to characterize the impact of IBM interpolation procedure on the parallel efficiency. Notice that DIM has the same parallel performance of the standard OpenFOAM version, because the imposition of the embedded boundary condition is performed with using surface information.

We report the results in Figure 6.9. The efficiency, represented by the dashed lines, has been computed considering wall times from 16 to 512 cores simulations (between 1.5 millions and 50 thousands degrees of freedom per core), because simulation time begins to raise when the number of cells per processor is below $50 \cdot 10^3$. These results tell us that, using IBM, we loose the 20% of performance with respect to DIM. We did expect a similar result, because the IBM procedure requires a significant number of WLS interpolations during a SIMPLEX iteration. However, looking to the right graph, we can see that the simulation times of IBM are of the same order of magnitude of the ones of DIM. Moreover, for the same mesh size, the IBM results are significantly more accurate than DIM ones, so the reduced performance is more than balanced by the gain in accuracy.

We also performed a test to measure the performance improvement from the original implementation present in the OpenFOAM unofficial fork `foam-extend-4.0`. Due to limitations that we experimented using the original IBM when dealing with anisotropic grid, we had to employ a different setup for the SSE test case, relaxing the number of degrees of freedom

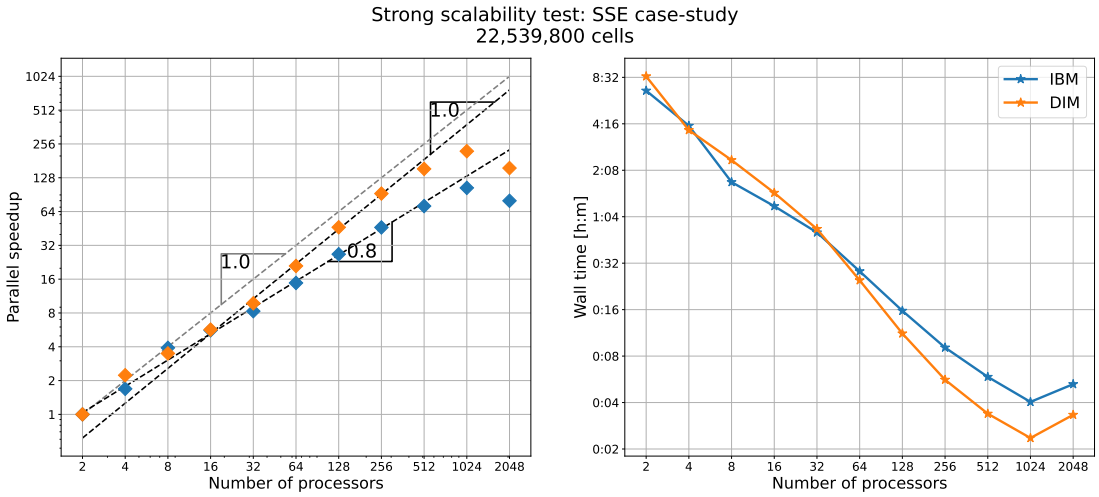


Figure 6.9: Log–log representation of the code parallel performance employing the SIMPLEX solver and both IB and DI methods. Left: simulation speed-up with respect to the number of processors (reference is 2 cores and linear solver is PBiCG with GAMG preconditioner). Right: the behaviour of the simulation time increasing the number of processors.

and the gap width in order to be able to have stable simulations. Scalability analysis has been performed on the MOX HPC cluster, which consists of 5 computing nodes, each 20 x CPU Intel Xeon E5-4610v2 @2.30GHz, with 32 cores each, 256GB RAM per node.

We report the results in Figure 6.10. While IBM and DIM present scalability features similar to the previous case, we can now observe that the original IBM implementation is not optimized for parallel applications. Indeed, its efficiency is much lower than the one of the other methods. Moreover, the computational time to solve the problem begins to increase when is below 250 thousands of cells per processor, five times more with respect to what we previously observed in Figure 6.9.

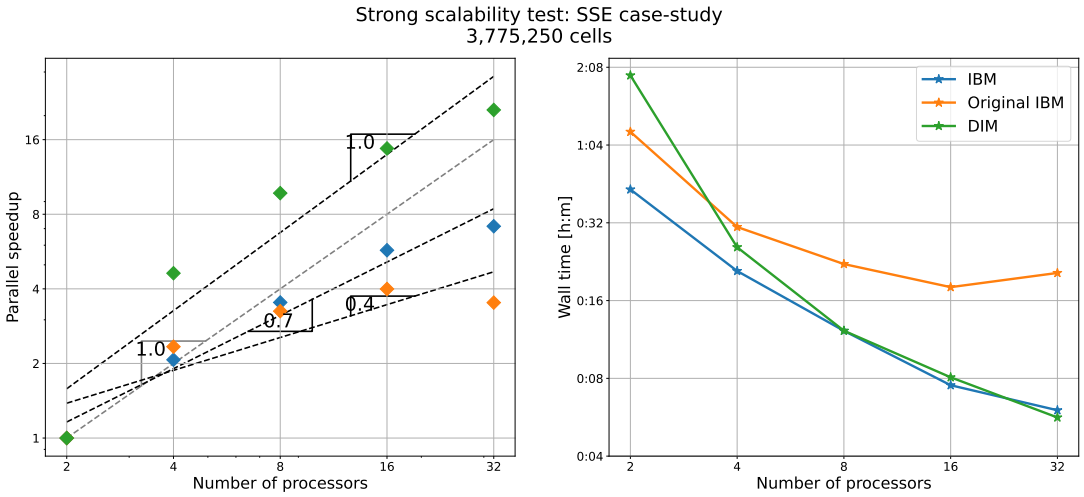


Figure 6.10: Log–log representation of the code parallel performance employing the SIMPLE solver and IB, in its new and original implementations, and DI methods. Left: simulation speed-up with respect to the number of processors (reference is 2 cores and linear solver is PCG with DILU/DIC preconditioner). Right: the behaviour of the simulation time increasing the number of processors.

6.1.2.3 Quantitative comparison on a short SSE sector

A further validation step of our single-screw simulations is obtained comparing results from OpenFOAM and from the commercial code ANSYS Polyflow, one of the current benchmark software for polymer processes.

We consider a short SSE sector as a computational domain. We run five different simulations as described in Table 6.1. We considered three meshes: the first is the coarsest one, where the grading is applied; the second and the third are regular meshes with two and three nested refinements, respectively.

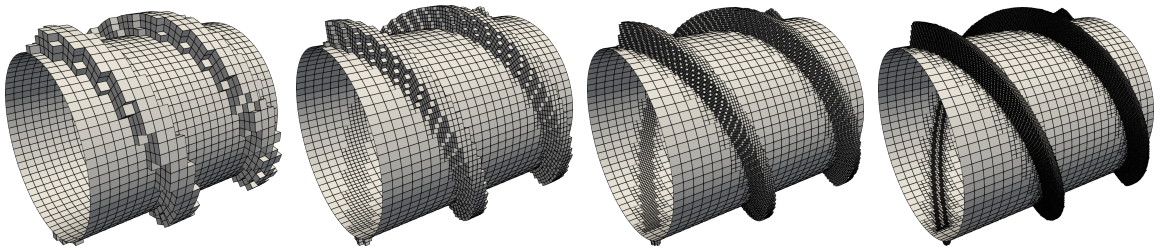


Figure 6.11: Grid visualization of the SSE sector of increasing local mesh refinement around the teeth IB surface. From no refinement (top left) to three nested levels of refinement (bottom right).

Mesh cells	**Polyflow**	**DIM**	**IBM**
0.1M	Coarse	Coarse	Coarse
2.1M	–	2 refinements	2 refinements
9.1M	–	3 refinements	–

Table 6.1: Sum up of the run simulations.

Polyflow has been used to simulate only on the coarse mesh because simulations on heavier meshes were unaffordable due to their huge virtual memory requisites. IBM was tested only on the last mesh refinement because it is comparable to the solution of DIM using the finest mesh.

We first compared some quantities sampled along a line along axial direction between the inner and outer radius (Figure 6.12). By the fact that we could not run the two finer simulations on Polyflow, we cannot see the trend of Polyflow solution so we can only make qualitative considerations.

The solution profiles computed along the sample line show that the five numerical solutions are qualitatively similar. A deeper look into the graphs shows that the solutions of DIM and Polyflow on the coarse mesh are comparable. Moreover, the sequence of DIM solutions tend to the IBM solution, as expected.

The last aspect we want to mention is related to performance point of view. We compared the number of degrees of freedom used, the number of processors, the wall time and the CPU time (wall time times number of processors). We measure the *efficiency* index as the CPU time (in seconds) over the number of degrees of freedom, basically the seconds spent on each DOF. From Table 6.2, the outcome is that we have overcome the performance of Polyflow, taking into account that it cannot even handle the same meshes that can be handled by OpenFOAM DIM and IBM. Moreover, despite DIM shows a lower computational time, we have shown that in just 1.5 times the computational cost of DIM we are able to obtain a significantly more accurate solution with IBM.

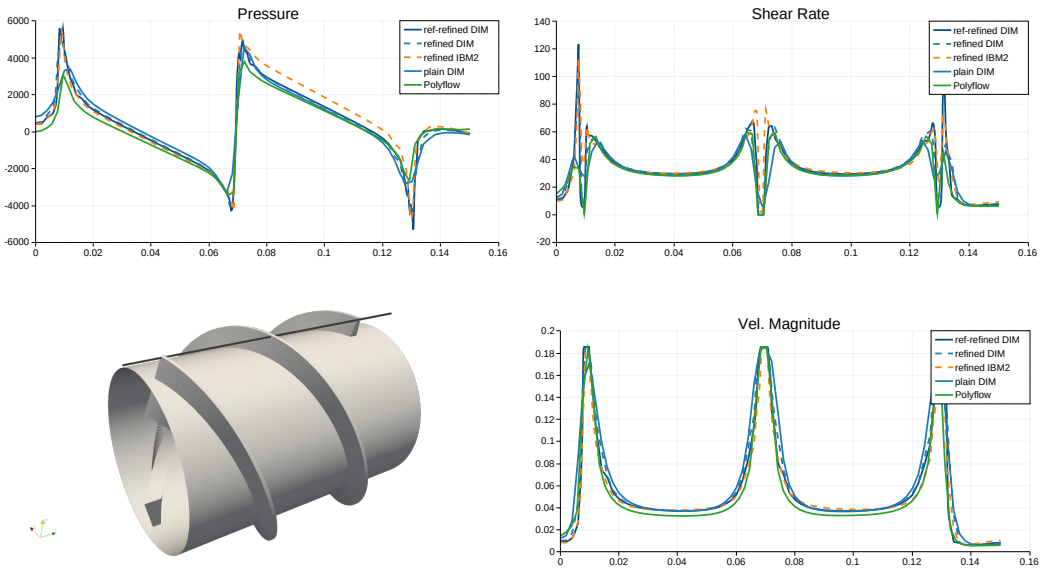


Figure 6.12: Sampling of pressure, shear rate and velocity magnitude along the line represented in lower left figure.

	Polyflow	DIM	Ref DIM	Ref-Ref DIM	IBM	Ref IBM
DOFs	0.8M	0.1M	2.1M	9.1M	0.1M	2.1M
# Proc	10	1	10	56	1	10
CPU Time	30h	5m	3h	42h	10m	65h
Wall Time	3h	5m	22m	46m	10m	6.5h
efficiency (s/#DOF)	0.135	0.003	0.00514	0.017	0.006	0.17

Table 6.2: Performance data for each simulation

6.2 The Twin Screw Extruder

A twin screw extruder (TSE) is a continuous mixing machine with two screws. TSE presents an increased number of design variables with respect to SSE, such as direction of rotation and degree of intermeshing.

Larger heat transfer area and better mixing ability allow good control of stock temperatures, residence times and positive conveying, that are key elements in the extrusion of thermally sensitive materials. On the other hand, this geometry complexity makes the TSE more difficult to be simulated when compared to the SSE. For these reasons, barrels and screws have been designed with removable elements, in order to allow arbitrary sequences of elements along the shaft. This modular design, therefore, is highly flexible and allows process optimization, unfortunately increasing the costs. Hence, numerical modelling plays a fundamental role in designing custom elements patterns for specific applications. From now on, we consider a case-study of a self-wiping intermeshing co-rotating TSE [118].

The section of a TSE is sketched in Figure 6.13. The device is composed by three parts: the barrel, represented by two adjacent cylinders of total width $2L_C$, of the order of dozens of centimetres, and length $30L_C$ and the two screws positioned in the two channels. The gap between screws and barrel is of the order of tens of millimetres, as in the SSE case. As mentioned before, the screws can be composed by different elements. In our case we consider two types of elements: the transport modules and the kneading modules, represented in the upper image of Figure 6.14. Both types are derived from the same profiles and the difference resides in the flight angles. The transport modules are devoted to conveying the fluid towards the die with different velocities depending on the flight angle, however, their mixing action is limited. The mixing action is prevalently performed by kneading modules, that are basically screw elements with a 90 degrees flight angle. They can be also of different lengths: long modules are better for distributive mixing, while short ones for dispersive mixing.

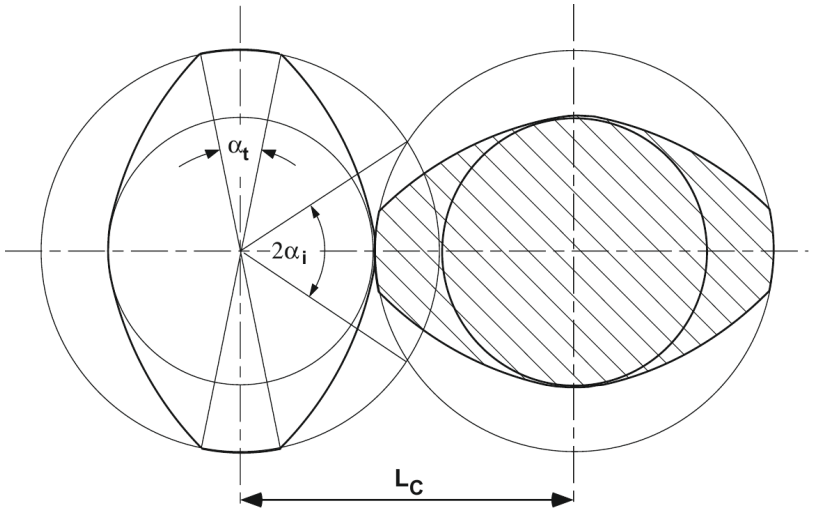


Figure 6.13: Geometrical description of an axial section cut [118].

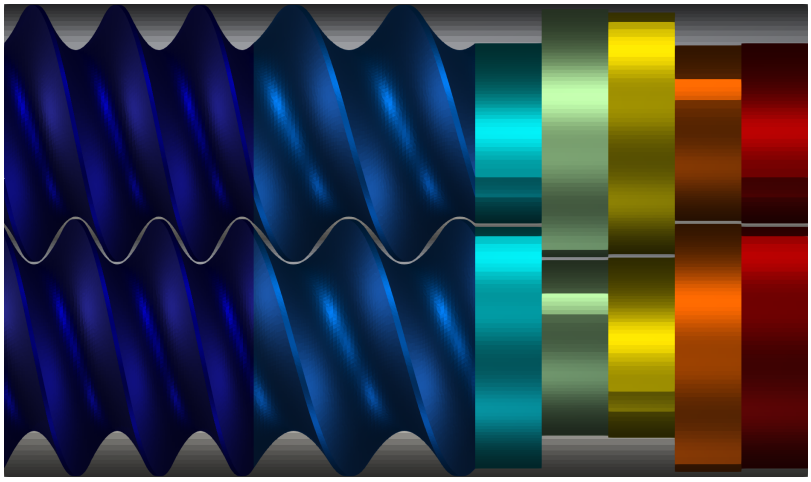


Figure 6.14: Examples of TSE modules, view from above. Different colours are the different modules: the first and the second are two transport modules of different pitches; the last five are kneading modules with different rotation angle.

For the simulation setup, we use a conforming grid for the barrel while we use IBM for approximating the presence of the two screws. The strategy to mesh the barrel is the following. We first generate the axial section of the left part using the block scheme of Figure 6.15, then we mirror the grid with respect to the intermeshing region. The reason to use such a block scheme is to have the highest flexibility possible in handling boundary layers and refinement in intermeshing zone, where we have the narrowest gaps between the screws. The same grading strategy of SSE case has been employed. The outcome of such a block division is represented in Figure 6.15. Notice that, employing the top-right and bottom-right blocks configuration, we can almost arbitrarily refine the intermeshing zone without affecting mesh size and quality of external parts. As for the SSE, TSE geometries present small gaps and so multiscale physical phenomena. However, by the presence of

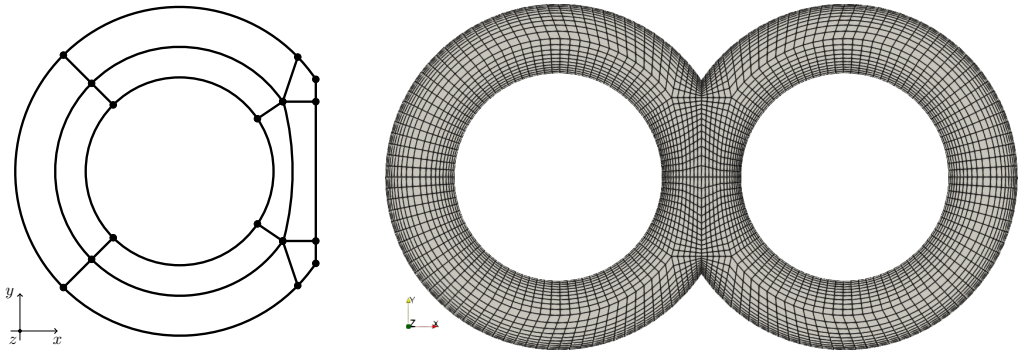


Figure 6.15: Left: Block scheme used by `blockMesh` for the axial section of the TSE. Right: sectional cut of the TSE sectional grid.

the intermeshing zone, there are more regions where small gaps can be found. Moreover, the screws rotate faster, so we have stronger velocity gradients and therefore a tougher nonlinearity to solve.

With respect to the IBM implementation, the big difference between the two devices is given by the strong interaction and proximity of the two screws in TSEs. In the SSE case-study, there where multiple geometries, i.e. the two teeth, that where treated by different IB surfaces and they were far and the influence of one tooth on the other was negligible. For the TSE, we have to handle the interaction between the two screws and the interaction between the various modules of each screw, in general the interaction between multiple immersed boundaries.

With opportune CAD tools, it is not difficult to construct a monolithic STL file of such geometries (Figure 6.14), however, as already mentioned in Section 4.4, we decided to develop a strategy to consider every single module as a single independent IB surface to be more efficient in geometry description.

6.2.1 Results

We present a time dependent simulation of the revolution of some TSE sectors filled with a high-viscosity power law polymer. In particular we considered two transport modules with different pitches, five kneading modules and another transport module. The working conditions are the following. The flow is driven by the screw rotation, set at 100RPM, so we set zero pressure on both inflow and outflow. On barrel and screws we set no slip condition for velocity. The barrel and screws temperatures are set to 323K as the inflow temperature.

The have simulated three complete revolutions in order to let the flow and the temperature field to stabilize. We have represented averages evolutions in time in Figure 6.16 and, to assess whether or not the solution reached an asymptotic behaviour, we also looked to the increment of the the sectional averages between subsequent time steps. We computed the following estimator: let $\overline{V}(z,t)$ be the average of a quantity V computed on section at axial coordinate z at time t, the increment reads:

$$\text{incr(t)} = \max_z \left| \overline{V}(z, t^{n+1}) - \overline{V}(z, t^n) \right|. \tag{6.2}$$

However, for this type of flows, we do not expect this error estimator to vanish as time goes to infinity, because the process is periodic and it does not necessarily reach a steady-state solution. Namely, we accept a slight difference in average between one time step and the other. The increments are reported in Figure 6.17. For flow rate, pressure and viscous heating we have periodic patterns developing in time. On the other hand, for temperature we can see a significant evolution in time heating, in average, the compound up to 4°and increasing. From these observations, we deduce that three revolutions are not enough to reach a periodic flow state.

Remark 6.3. IBM implementation is not yet optimized to work in time dependent simulations because of the fact that, at each time step, the STL file changes position and all the IB sets and stencils have to be recomputed. Despite being parallelized, this procedure has a significantly higher cost with respect to the solution time of the current linear system. While

in SSE and PRE geometries (see previous section 6.1 and next section 6.3), this aspect does not have a remarkable impact on simulation time, because the solid and IB cells occupation is largely smaller than other mesh subsets, in the TSE the whole screws are contained in the computational domain (Table 6.3). This computational cost difference is particularly evident if comparing SSE and TSE computational times.

	SSE	TSE	PRE
No. of D.o.F.s	6.76 Mln	3.3 Mln	2.9 Mln
% of IB+Solid cells	9.7%	41.8%	11.6%
Simulation time	1.5 days	5 days	1.5 days

Table 6.3: Fraction of IB and solid cell with respect to total number of degrees of freedom per case study. All the simulations were run on 56 processors.

The results for flow rate, average pressure and average viscous heating suggest that there are no significant changes in time for velocity and pressure because viscous heating and flow rate mainly depend by them and they are in practice constant during time. On the contrary, we have an increasing temperature profile in time, that does not seem to have completely reached a constant profile. The errors trends, represented in Figure 6.17, confirm the second hypothesis, indeed we can observe a rise of the errors while reaching the final time step. Hence, three revolutions are not enough to reach a fully periodic flow condition.

Moreover, after the first time steps, the flow rate settles around a uniform value on the whole domain, with an average value of 0.047 [m^3/s] and a standard deviation of 0.00084 [m^3/s], which confirms the conservation of the mass with an error up to $\simeq$0.1%.

We have reported the evolution of the temperature inside the TSE in Figure 6.18. The small gap sizes in combination with high rotational velocities of the screws produce a lot of viscous heating, that increases the melt temperature. A particular increase of temperature is detected in gaps the two screws, in particular between kneading modules. In the latter region, the velocity gradients are greater, due to the width of the modules crests, that generates a higher viscous dissipation.

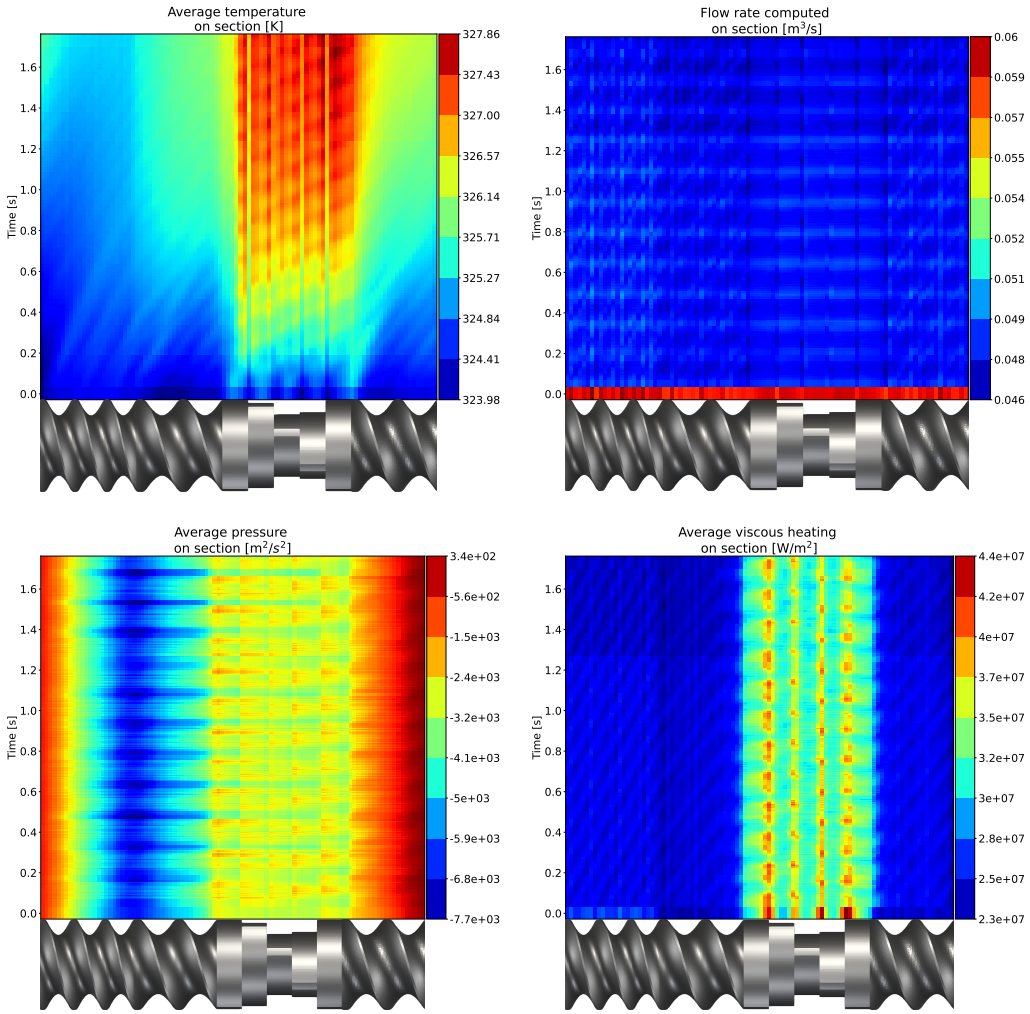


Figure 6.16: Evolution in time of some quantities evaluated on several axial sections. On the x axis of each figure, the screw rendering has been represented in order to enlighten the flow features with respect to the geometrical feature. Top left: temperature average. Top right: flow rate. Bottom left: pressure average. Bottom right: viscous heating average.

6.3 The planetary roller extruder

Single-screw and twin-screw extruders are widely used in extrusion processes and they have been studied extensively. Nevertheless, there exist other extruders with more complex geometries and kinematics, which are customized for specific tasks. Here, we present a numerical study of the planetary roller extruder (PRE). The PRE is a multi-screw extruder composed by a central spindle (sun) and the barrel (ring) with variable number of smaller spindles (planets) between them. The rotation of the sun drives the one of the planets thanks to their gear-like shapes.

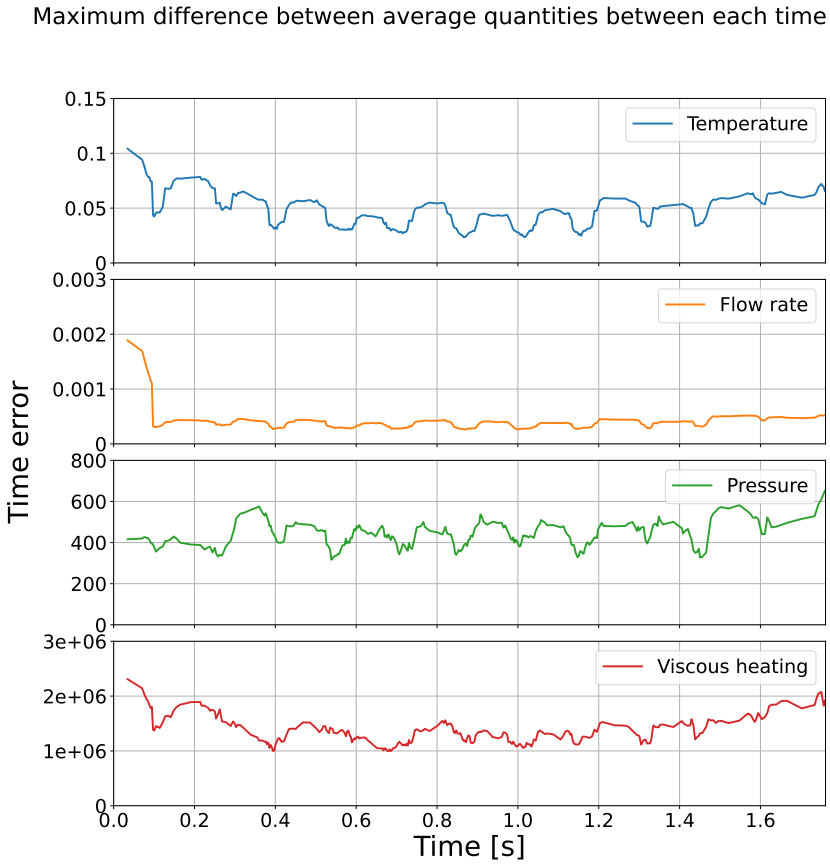


Figure 6.17: Evolution in time of error computed by equation 6.2.

Moreover, its design allows a large number of configurations, that makes the PRE a powerful mixing tool configurable for many types of processes. First, the number of contiguous modules can vary. Second, the number and the type of planets can vary between one module and another.

The flow in a PRE is driven by both drag forces and pressure gradients because the spindles are characterized by a helical gearing to allow the transport of the fluid along its axis. Moreover, the gearing increases significantly the contact surface between the fluid and the spindles with respect to single- and twin- screw extruders, which allows the drag to noticeably drive the flow.

A planetary roller extruder is able to masticate, mix, homogenize, disperse and de-gas highly viscous substrates. In the end, a PRE is the optimal choice to perform continuous mixing.

Very few numerical explorations have been done on this type of device. What will be presented next seems to be novel in the literature. As far as we know, the only work about numerical simulations is exposed in [134], where a sector of a PRE is considered and both the

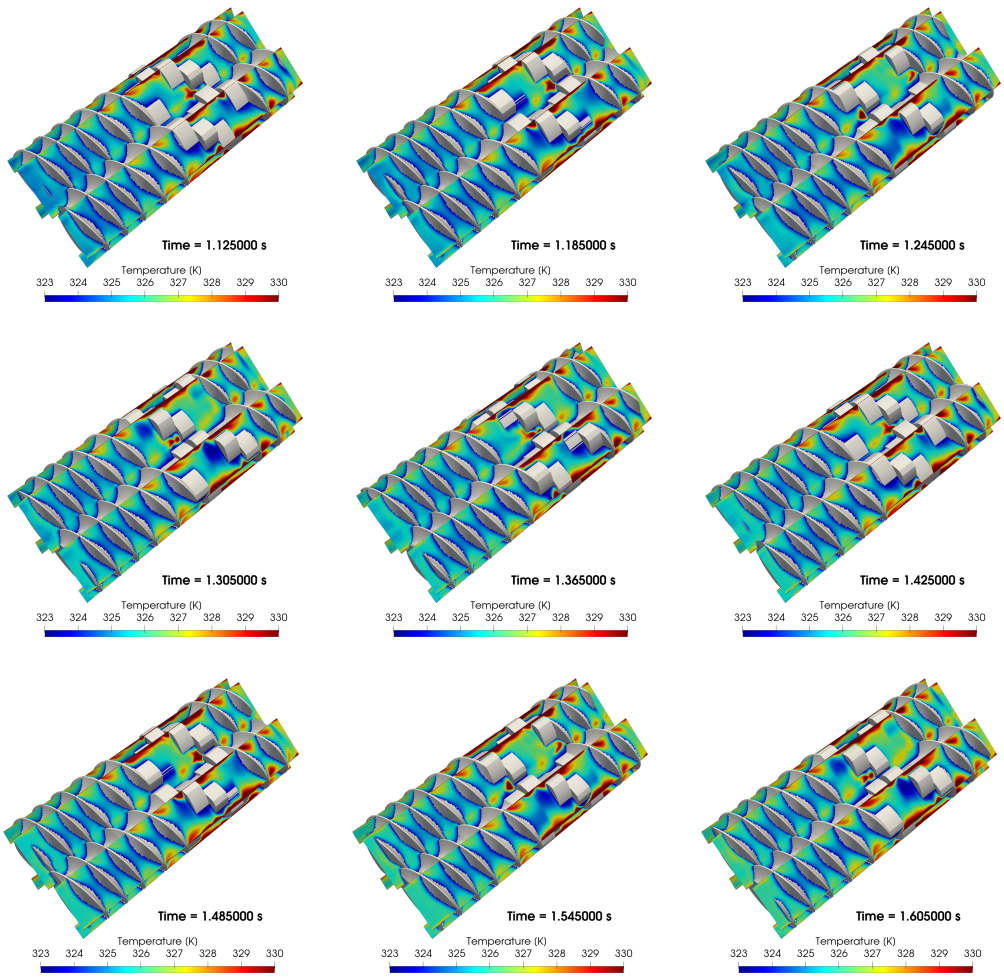


Figure 6.18: Temperature distribution in the twin-screw extruder at different times.

sun and the ring and the spindles are approximated using the mesh superposition technique implemented in ANSYS Polyflow.

Here we first propose a new methodology to simulate this device approximating both sun and ring with a conforming mesh, while spindles are embedded using the IBM. An automatized procedure to build a conforming mesh of a PRE is implemented and explained. Another novel feature of our approach is the inclusion of the temperature in the simulations. On the sun and the ring temperature is prescribed, while the spindles are considered as solid with a certain heat conductance. Finally we show also a time dependent approach in which we employ a sliding mesh technique to move sun and ring.

In this framework, a mesh convergence study is exposed in terms of average and integral quantities and a weak scaling analysis for parallel efficiency is presented.

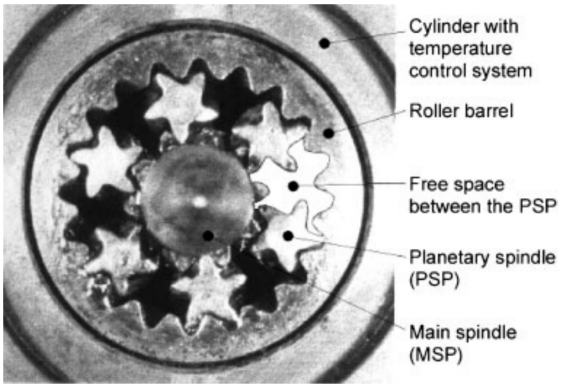


Figure 6.19: Example of a planetary roller extruder [96].

6.3.1 PRE kinematics

As stated in the introduction section, the motion is completely driven by the sun shaft rotation. We let ω_s be the angular velocity of the sun. Knowing that the teeth contact points are all on the pitch circles, we can determine the kinematics of each device part. Let ω_r be the angular velocity of the ring, ω_p the angular velocity of the planets around their barycentre and ω_c the angular velocity of the carrier, i.e. of the barycentres of the planets around the origin. We represent these information in Figure 6.20.

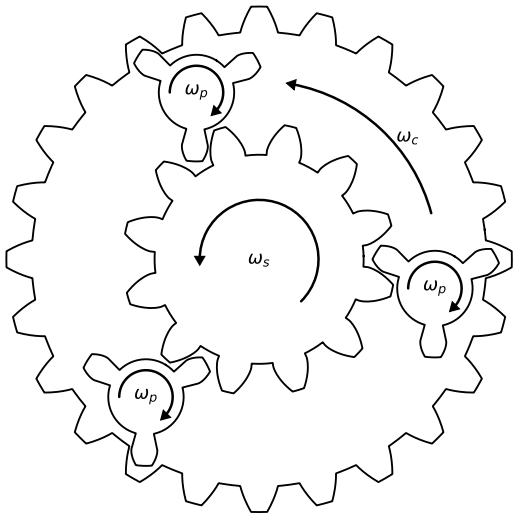


Figure 6.20: Profiles of a PRE section with 12 and 24 teeth on sun and ring, respectively. It is designed with three planets that have three teeth each. The kinematics is also represented with the angular velocity of each gear.

If the origin as the reference frame, we fix ω_s, let $\omega_r = 0$ and R_p be the planet radius,

then the angular velocities are defined as

$$\begin{aligned} \omega_c =& \frac{1}{2}\frac{R_s}{R_s + R_p}\omega_s, \\ \omega_p =& -\frac{R_s + R_p}{R_p}\omega_c. \end{aligned} \tag{6.3}$$

To simplify the implementation of this kinematics, we set the new reference frame moving with carrier, hence with the planets barycentres, performing eventually simulation using the MRF solvers of OpenFOAM (c.f. equation 2.35). In this way, we now are on a non-inertial frame of reference and, setting $\tilde{\omega}_c = 0$, the velocities read:

$$\begin{aligned} \tilde{\omega}_s =& \omega_s - \omega_c, \\ \tilde{\omega}_r =& -\omega_c, \\ \tilde{\omega}_p =& \omega_p - \omega_c. \end{aligned} \tag{6.4}$$

Now, sun and ring rotate in opposite direction, while the planets only rotate around their axes. Ad hoc strategies adopted to simulate the moving PRE will be explained in nest sections.

Remark 6.4. Motion and kinematics in the PRE are produced by the contact between sun and planets and ring and planets. Indeed, by the way in which the geometry is built, planets, sun and ring would touch in order to rotate, while in real geometries there exists an inter-meshing region in which the fluid passes. In our simulations contact mechanics between the gears is not taken into account. Thus, to model the gaps in which the fluid passes between planets and sun and ring, we have "shrank" planets by a factor of 5%.

6.3.2 Building a conforming grid

To generate a conforming mesh of a PRE, gear profiles have to be considered. In particular, we considered spur-gear teeth and their parametrization (Figure 6.21). Spur gears have teeth parallel to the axis of rotation and are used to transmit motion from one shaft to another, parallel, shaft [28]. In the PRE design, they are also twisted along the shaft axis by a certain angle to perform the transport of the fluid. Looking at Figure 6.21, we define the generator radius of a tooth as R_{gen}, that is the radius of the circle between the addendum and the dedendum. We define the generator radii of sun and ring as $R_{s,gen}$ and $R_{r,gen}$.

Here we present a fully automatic strategy to build the conforming mesh of a PRE sun and ring. The developed strategy is parametrized with respect to the number of teeth of sun, ring and planets. The teeth profiles are parametrized with respect to generator radii, addenda and dedenda.

The workflow for building a conforming mesh is to construct separate teeth and then to merge them together to build the whole computational grid. In Figure 6.22, we have represented the spur-gear profiles of the teeth of each piece of the PRE: sun, ring and planets. For every piece, the tooth is characterised by two involutes, i.e. the curved segments, that connect the internal and external circular segments.

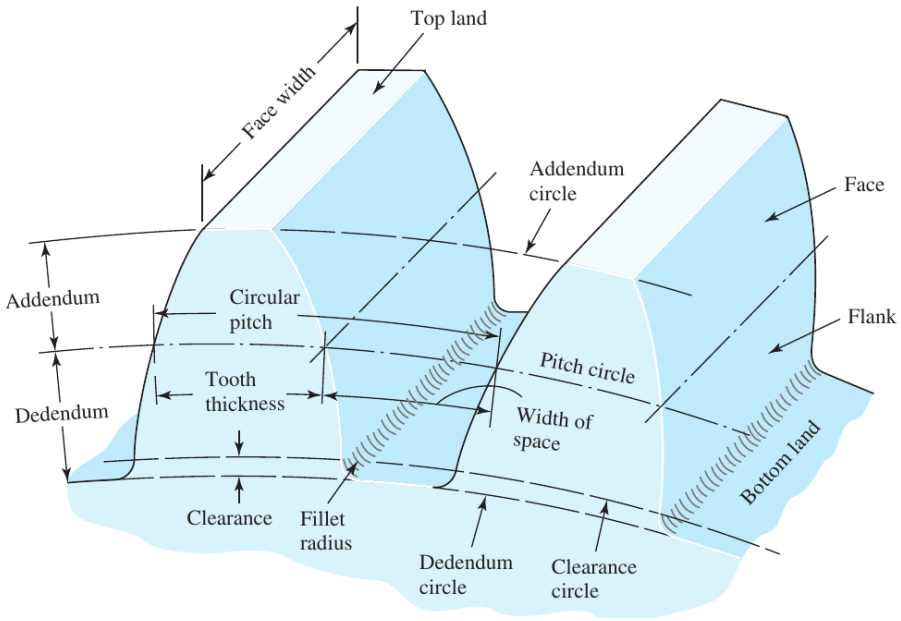


Figure 6.21: Schematics of spur gear teeth [28].

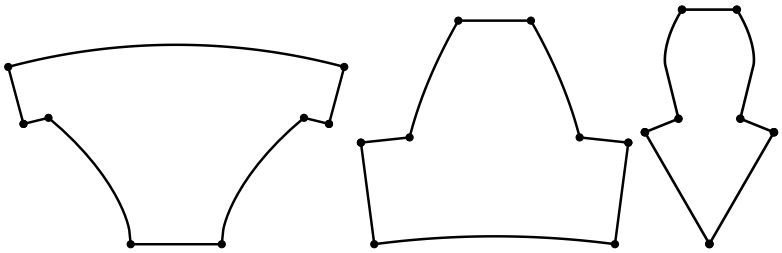


Figure 6.22: Profiles of sun, ring and planet teeth, from left to right, respectively.

Once we have created the mesh of each single piece, we are able to "sew" them together (Figure 6.23). We first set sun and ring pieces at the right distances from the origin. Then, for the sun piece we repeat the shape azimuthally for N_s times and we do the same with the ring piece for N_r times. For the planet we repeat the piece 3 times, interspersed by a plain circular piece. Merging then sun and ring together, positioning the planets in the correct positions, we obtain the final configuration for our PRE case-study (Figure 6.20).

For the sun and ring teeth blocks, we manage to decouple the mesh refinement of the teeth and core region, generating a boundary layer along teeth profile. As show in Figure 6.24, we set $R_{s,mid}$ and $R_{r,mid}$ the radii dividing core region from the sun and ring teeth boundary layer regions, respectively. We define also $R_{mid} = \frac{1}{2}(R_{s,mid} + R_{r,mid})$ as the radius of the interface between ring and sun meshes, where we will do the merging.

In Figure 6.23, we have represented the merging procedure that brings to the final PRE grid configuration. The algorithm is the following. Let N_s and N_r are the sun and ring number of teeth, respectively, we create the ring and sun teeth meshes separately and then

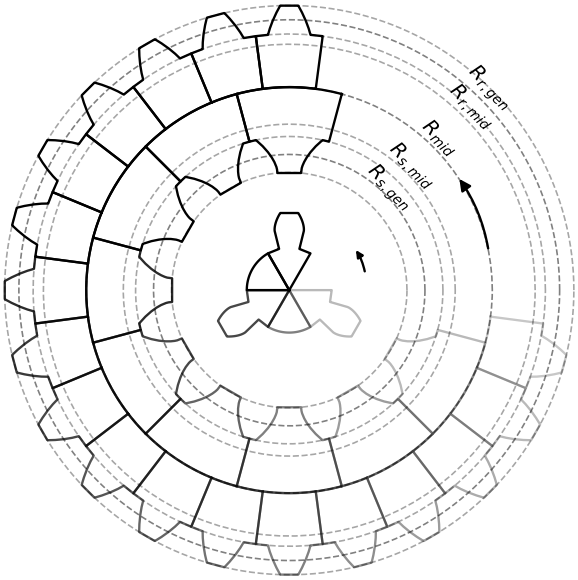


Figure 6.23: PRE conforming mesh construction procedure. Sun and ring building blocks are merged together. The same is done for a planet (in the centre of the PRE section).

we add iteratively:

for $n = 0, n < N_s, n++$ **do**
  $newsun =$ rotate$(sun, n \cdot 2\pi/N_s)$
  $sun =$ merge$(sun, newsun)$
end for
for $n = 0, n < N_r, n++$ **do**
  $newring =$ rotate$(ring, n \cdot 2\pi/N_r)$
  $ring =$ merge$(ring, newring)$
end for
$pre =$ merge$(sun, ring)$

Before performing the last merging, sun and ring parts are twisted by a pitch angle. Mesh twisting is performed applying opposite rotation angles for sun and ring, in order to make the planet fit. Thus, we have employed the following strategy: we twist sun by θ_s, ring by θ_r.

Let $\mathbf{p}_0 = [x_0, y_0, z_0]^\intercal$ be a point in the original mesh and let l_p be the pitch length. We obtain the point $\mathbf{p}_1 = [x_1, y_1, z_1]^\intercal$ in the new configuration by $\mathbf{p}_1 = \mathrm{R}(\theta)\mathbf{p}_0$, where $\mathrm{R}(\theta)$ is

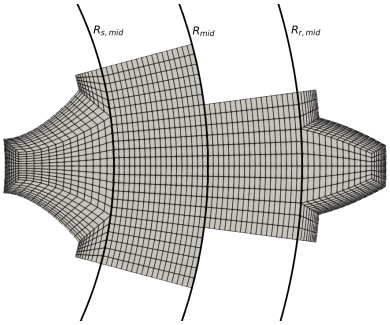


Figure 6.24: Sun and ring teeth grids example. Boundary layer and mesh interfaces are represented by the circular lines.

the rotation matrix of angle θ. θ is defined by the following relationship:

$$\begin{aligned}
\theta_s(z_0) =& 2\pi \frac{z_0 - z_{min}}{N_s l_p}, \\
\theta_r(z_0) =& 2\pi \frac{z_0 - z_{min}}{N_r l_p}, \\
r =& \sqrt{x_0^2 + y_0^2}, \\
\theta(r) =& \begin{cases} \theta_r, & r \geq R_{r,mid}, \\ \theta_s, & r \leq R_{s,mid}, \end{cases}
\end{aligned} \tag{6.5}$$

The twisting procedure is represented in Figure 6.25 while the result of the twisting of two sun and ring blocks is represented in Figure 6.26.

Remark 6.5. The former procedure has been performed using the OpenFOAM built-in tools `mergeMeshes` and `stitchMesh`. Moreover, the procedure has been automatized with respect to the choice of sun, ring and planet number of teeth to have the highest flexibility possible in the construction of such geometries.

6.3.3 Results

In this section we present some results regarding simulations of a PRE sector. We first present a case in which the solution is a steady-state and then we present the time dependent case, in which geometries rotate. Simulating the case in steady-state means that we discard the time derivative in the equations and impose the effective rigid rotations on the boundary conditions. We also performed a weak scaling analysis, i.e. varying number of cores maintaining constant the number of d.o.f.s per core, along with a grid convergence analysis, comparing various integral quantities of interest. Finally, we performed a time dependent simulation of the PRE, combining the dynamic mesh library of OpenFOAM and our IBM implementation.

First, we introduce the geometrical setup. As described in the previous section, we considered a PRE with 12 teeth on the sun, 24 teeth on the ring and 3 teeth on each one of

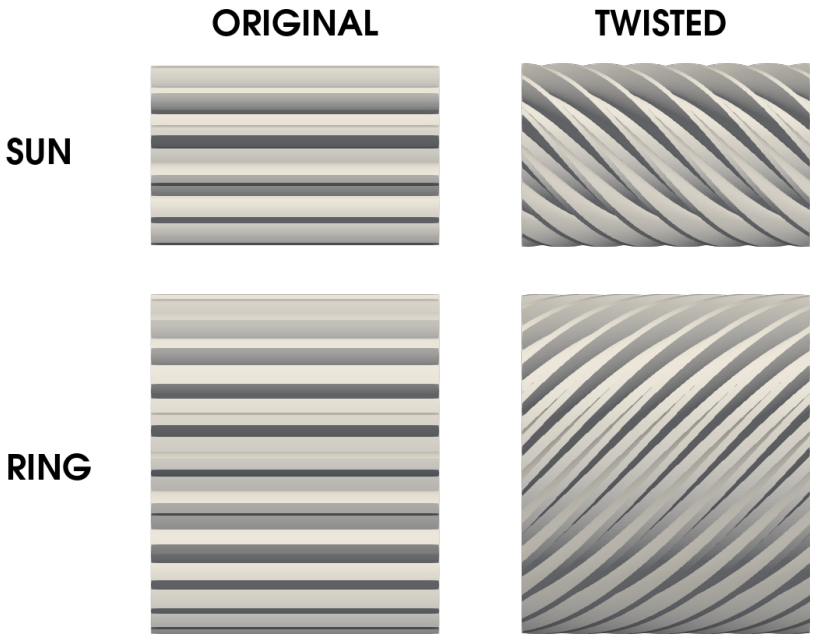


Figure 6.25: Twist transformation of sun and ring in opposite directions.

the three planets. However, running a simulation using the mesh built with the procedure described in previous section is not straightforward because the interface between sun and ring parts presents non-conforming mesh faces, due to opposite twist directions.

Hence, we adopted the following strategy. Due to the fact that sun and ring parts are twisted in opposite directions, the interface between sun and ring present non-conforming elements. To deal with this situation, we exploit the structure and the features of the finite volume method, that allows to connect two pieces of mesh without loosing a significant amount of numerical accuracy. This because they are base only on flux balance, that is preserved also in case of nonconforming elements between different boundaries. The strategy involves the so called periodic, or cyclic, boundary conditions, that map the solution on the two sides of the non-conforming interface, using flux balance, to compute a continuous solution. This feature of FVM clearly increases the flexibility in terms of mesh generation and usage.

Refer now to Figure 6.26. Once the computational mesh is built, we need to implement a strategy to move our PRE geometry (or to deal with non-conforming interfaces, in steady-state case). We employed the so called sliding interfaces. While the ring is fixed, the sun moves with a rotational velocity ω_s. This means that we have to "split" the mesh into two parts: the sun part, i.e. all the cells with centre within R_{mid}, and the ring part, i.e. the remaining cells. Then, with the same strategy of mesh construction, using periodic boundary conditions between sun and ring part, we are able to reconstruct fluxes across the sliding interface (the mid line in Figure 6.24). Sun part is then rotated at each time step with respect to its angular velocity and the fields are re-mapped using ALE strategies (c.f. Section 2.1.4.2). On the other hand, planets are handled using the IBM library. To this aim, the

toolbox has been integrated with the dynamic mesh framework of OpenFOAM, in order to be able to use the built-in tools without rewriting additional libraries.

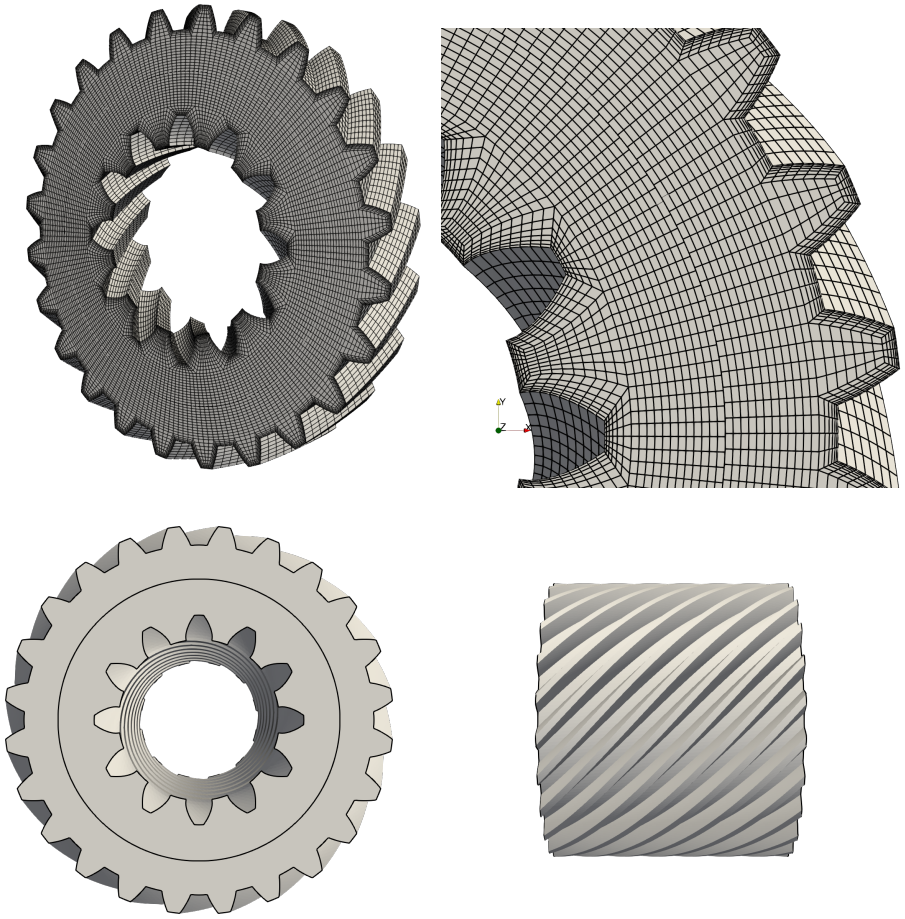


Figure 6.26: Top: grid visualization of a PRE sector (left) and grid after one time step, to enlighten the non conformity of cell edges when moving the mesh (right). Bottom: geometry with subdivision of sun and ring parts by sliding interface (left) and side view of the geometry (right).

Remark 6.6. In OpenFOAM, the periodic boundary conditions between two non-conforming interfaces are called `cyclicAMI`, cyclic Arbitrary Mesh Interface, and the sliding interfaces are called `baffles`.

Case number	0	1	2	3	4
Number of D.o.F ($\cdot 10^3$)	400	900	1700	2400	4400

6.3.3.1 Quasi-static simulations

We first considered the simulation of the PRE in a steady-state framework. We fixed sun rotation ω_s and computed planet rotational velocity $\tilde{\omega}_p$ accordingly. We then used the carrier velocity ω_c in the inertial frame to compute the rotation of planets with respect to the origin. We then impose a flow rate at the inlet and zero mean pressure at the outlet.

For the temperature, we impose an average temperature at the inlet (in a mixed condition flavour, Dirichlet at inflow cells and Neumann at outflow cells) and adiabatic condition at the outlet. We fixed temperatures for ring and sun, that are thermally controlled, and we considered planets free to conduce heat, hence implementing an interface condition for the IBM. In particular, this passage consists in solving two problems on the same computational domain.

On $\Gamma_F \cup \Gamma_{\text{IB}}$ we solve the advection-diffusion equation with viscous heating source using the high viscosity polymer physical properties. While, on Γ_S we solve an advection-diffusion equation with no sources using the physical properties of steel. So, recalling problem (2.22), the energy equation reads:

$$\begin{aligned} \mathbb{1}_{\Gamma_F \cup \Gamma_{\text{IB}}} \left(\rho \frac{\partial c_p T}{\partial t} + \rho \mathbf{u} \cdot \nabla (c_p T) - \nabla \cdot (k \nabla T) - \tau : \mathbf{D} \right) = \\ = \mathbb{1}_{\Gamma_S} \left(-\rho^s \frac{\partial c_p^s T}{\partial t} - \rho^s \mathbf{u} \cdot \nabla (c_p^s T) + \nabla \cdot (k^s \nabla T) \right), \end{aligned} \tag{6.6}$$

where parameters with s at the superscript are the physical parameters of steel and $\mathbb{1}$ is the indicator functions. We approached the problem using the SIMPLEX algorithm (c.f. Section 5.3), to guarantee stability and robustness. More detailed results on quasi-static simulations, together with a comparison against the time dependent solution, will be reported in Section 6.3.3.2.

Convergence We performed a mesh convergence analysis considering just a shorter PRE sector to reduce the computational complexity. The boundary conditions are the same as described above.

We considered five levels of refinement. The number of degrees of freedom per each case is as follows:

$$\frac{N_{dof}}{10^3} = 400, 900, 1700, 2400, 4400.$$

The meshes where built using the procedure described in Section 6.3.2 varying the number of azimuthal cells of a single tooth (c.f. Figure 6.24) and the number of cells in other directions accordingly.

Then, to measure convergence, we sampled 20 surfaces in axial directions and we measure some global and integral quantities of interests: the mean temperature, the integral of

viscous heating, the pressure gradient and the flow rate. Results are reported in Figure 6.27. The analysis confirms a convergent behaviour. We computed the relative error between each solution against the finest. Let $\overline{V}^n(z)$ be the average of a quantity V computed on section at axial coordinate z for simulation n, the relative error for simulation n reads:

$$\text{err}_n = \frac{\max_z \left| \overline{V}^4(z) - \overline{V}^n(z) \right|}{\max_z \left| \overline{V}^n(z) \right|}, \qquad n = 0, 1, 2, 3. \tag{6.7}$$

The error behaviour is reported in Figure 6.27.

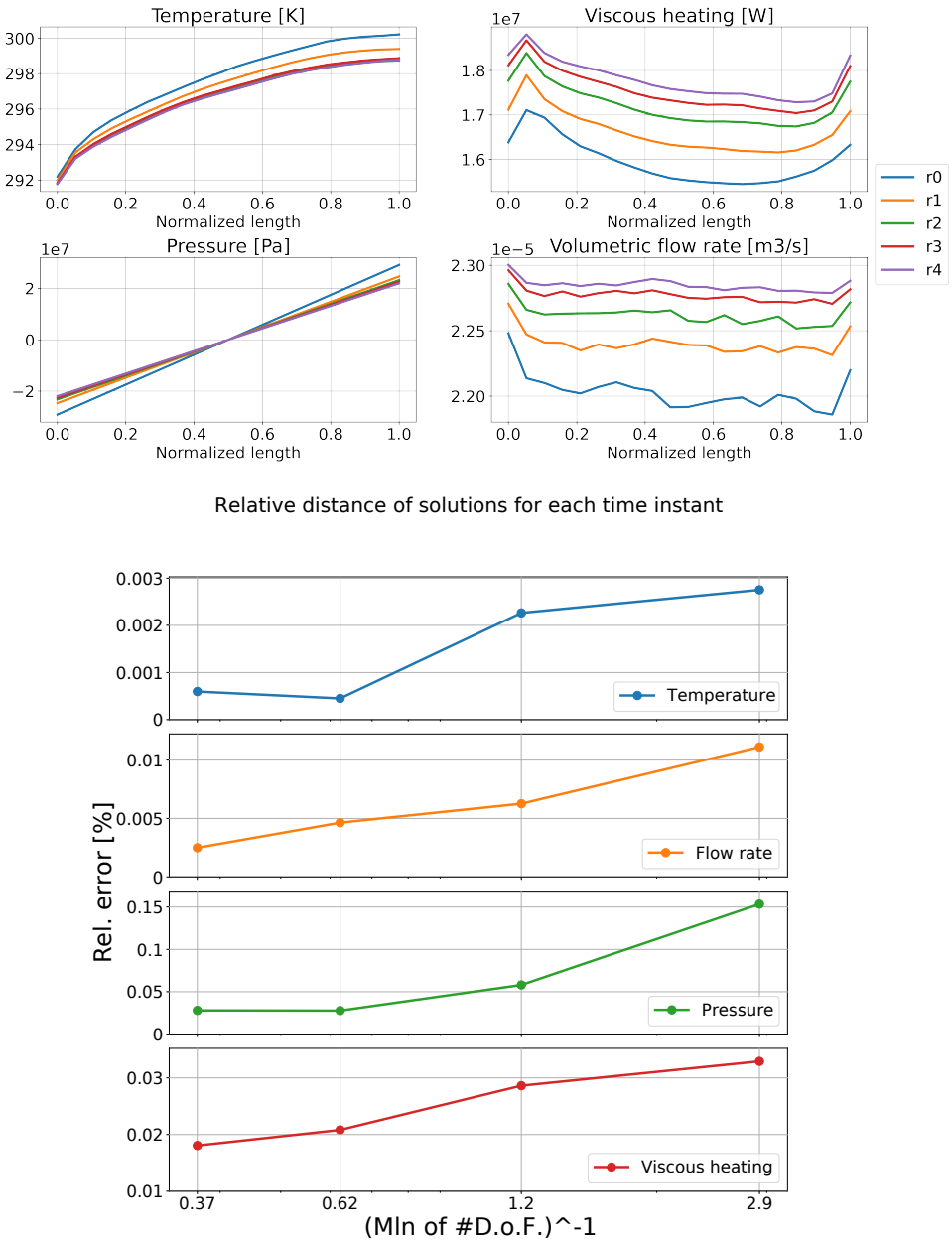


Figure 6.27: Top: global measures of some quantities of interests sampled at different axial positions. $r_i, i = 0, \dots, 4$ represent the different grid levels, from the coarser to the finer. Bottom: relative distance between finest and coarser solutions, computed with equation 6.7.

Weak parallel scaling On the same group of simulations of the convergence analysis, we performed a weak scaling analysis to assess the parallel efficiency of the code. We considered the five simulations and we simulated them keeping constant the number of degrees of freedom per processor, indicatively a hundred thousand.

We measured the simulation time for each simulation. It is represented in Figure 6.28. What is worth to notice is that the curve is reaches a plateau. This means that the code is able to solve increasingly larger problems using the same amount of work per node, that is the idea behind weak scaling analysis.

The simulation time difference between each case diminishes as the problem size increases. This is a consequence of the complexity of the problem, indeed, refining mesh size, the problem is harder to solve, due to the fact that temperature and velocity gradients increase and so convergence is tougher to be reached. On the other hand, when mesh is enough fine, efficiency stabilizes because the problem difficulty does not increase any more. The stabilization of the simulation time means that, with an enough fine mesh, the amount of work remains the same, keeping the same number of d.o.f. per node.

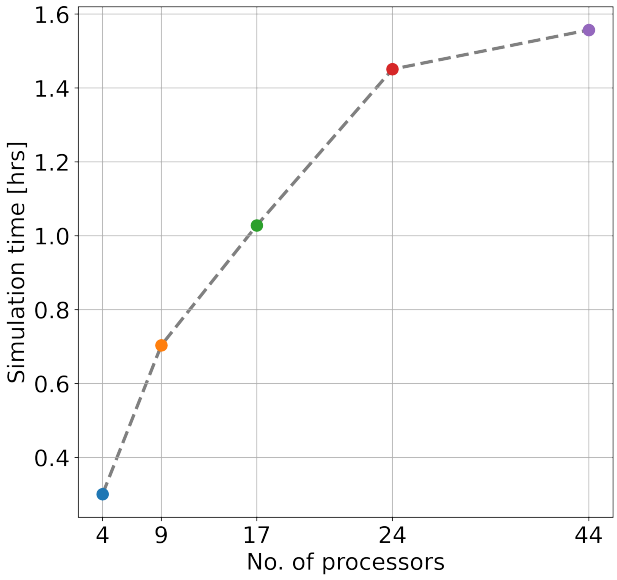


Figure 6.28: Simulation time for each test case increasing mesh size and and keeping the same number of d.o.f.s on each core.

Comparison with a commercial tool As in Section 6.1, we compared our results with the ones of the commercial software ANSYS Polyflow. In particular, we want to enlighten the performance gain that we experiment using our new OpenFOAM toolbox. In Figure 6.29, the number of degrees of freedom and the simulation times are reported for both software.

Polyflow uses a FEM discretization, hence, on the same grid, it has more degrees of freedom than OpenFOAM. However, we were not able to run any simulation after the third, because of computational weight and robustness issues relative to nonlinearity.

OpenFOAM is able to deal with the nonlinearity in every simulation, thanks to the use of SIMPLEX and PIMPLEX algorithms (c.f. Section 5.3). Moreover, it outperforms Polyflow in terms of computational time also if considering the same number of d.o.f..

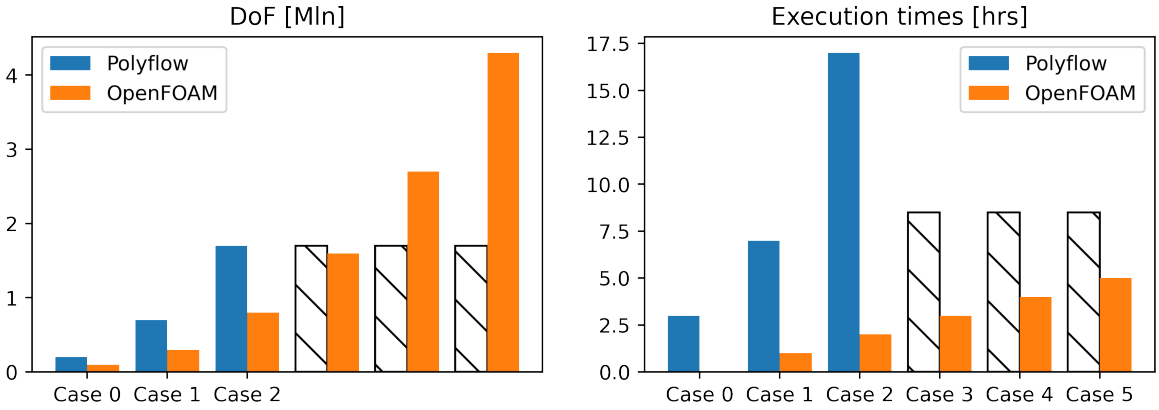


Figure 6.29: Comparison between OpenFOAM and Polyflow computational times.

6.3.3.2 Time dependent simulations

Here we present the results of the time dependent simulations of the PRE. To simulate the moving PRE, we employed the strategy mentioned above, in which we move sun and ring with angular velocities of opposite signs and the planets are moved by the IBM, setting the problem in a non-inertial frame of reference.

The simulations velocity and pressure fields have been initialized using the steady-state isothermal solution of the problem, imposing the actual rigid rotations on the PRE components. The fields have been initialized with the solution of a quasi-static simulation with the same boundary conditions prescribed for the time dependent simulation. As for the TSE, we have simulated 50 complete revolutions of the sun in order to let the flow to stabilize. We have represented averages evolutions in time in Figure 6.30 and we have reported the computed by equation 6.2.

From the evolution of global quantities in time, we can notice that the periodic flow regime has been fully reached. Also time errors stabilizes at the end of the time frame. In particular, while flow rate, pressure and viscous heating stabilize after 20 seconds, temperature evolves until 50 seconds. This shows that the slight variations in the flow field can significantly influence the temperature field, due to the high advection dominance of the problem. We can also notice that temperature and viscous heating averages have similar features because the more the fluid heats, the more viscosity decreases and so the viscous heating.

The axial pressure variation decreases in time because the fluid heats and it becomes easier to be pushed towards the outflow. Finally, The flow rate is uniform value on the

whole domain, with an average value of 2.63e-05 [m^3/s] and a standard deviation of 4.39e-07 [m^3/s], which confirms the conservation of the mass with an error up to $\simeq$1%.

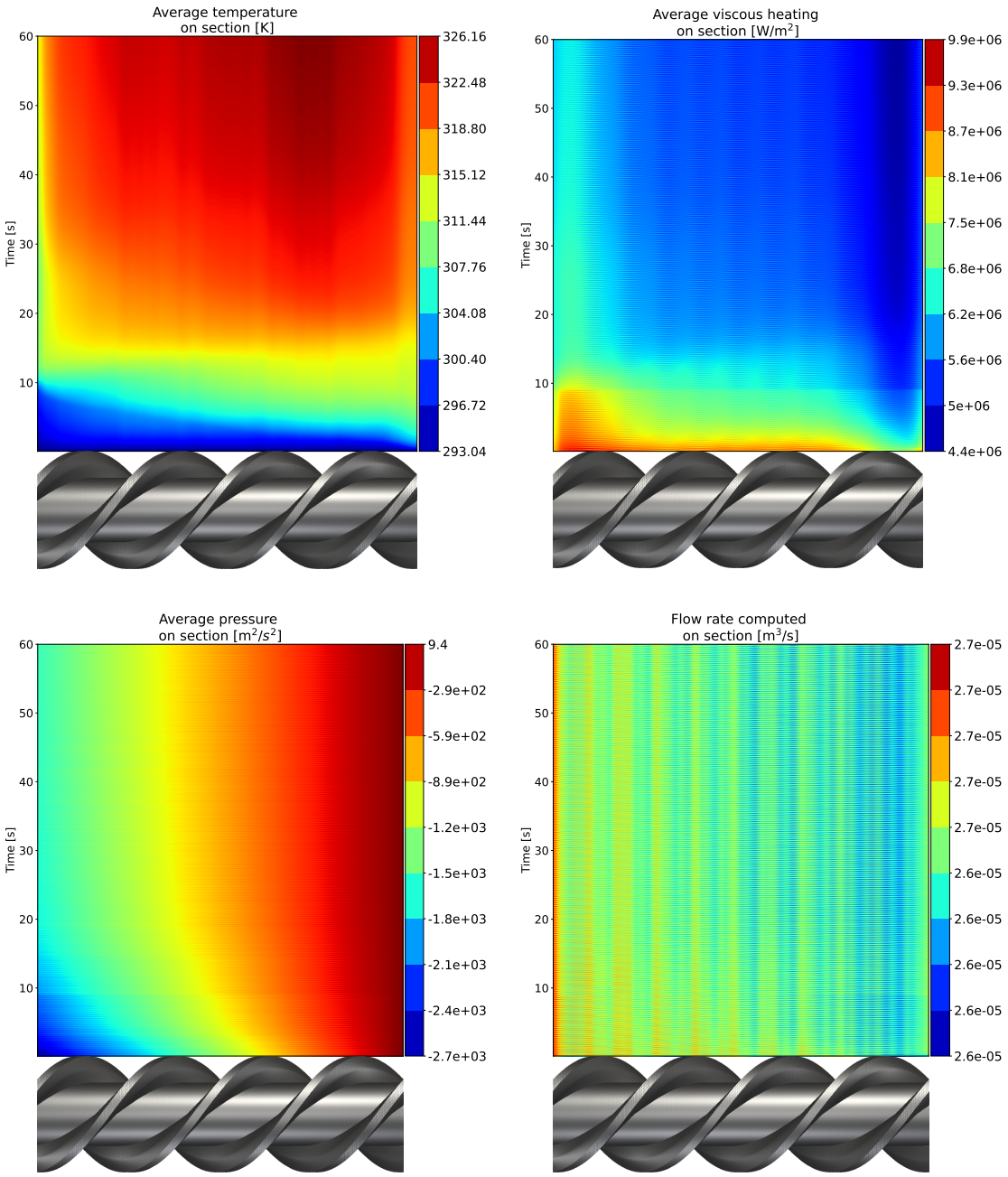


Figure 6.30: Evolution in time of some quantities evaluated on several axial sections. On the x axis of each figure, the screw rendering has been represented in order to enlighten the flow features with respect to the geometrical feature. Top left: temperature average. Top right: flow rate. Bottom left: pressure average. Bottom right: viscous heating average.

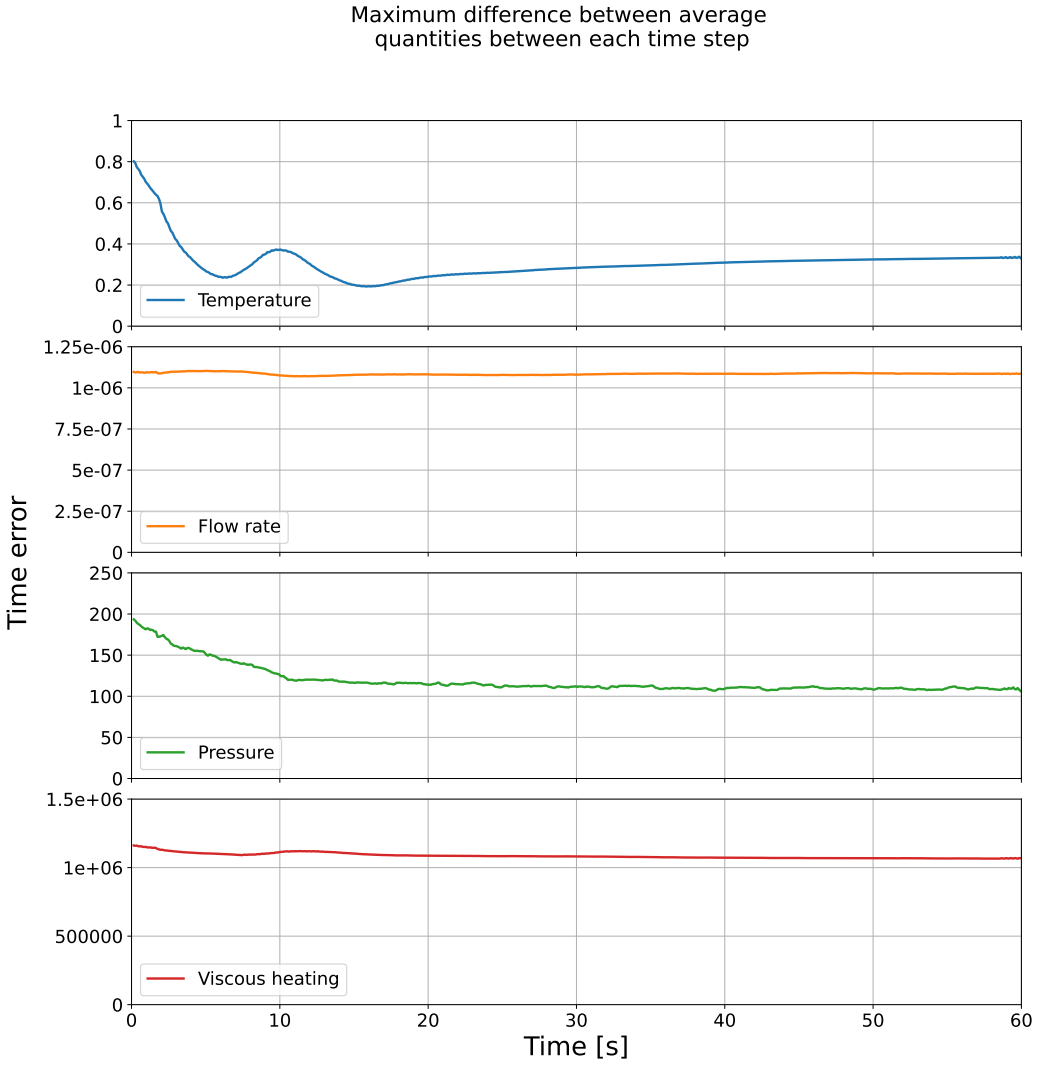


Figure 6.31: Evolution of the increment in time for each averaged quantity.

Finally, we report some slices (see Figure 6.32) of the PRE fluid domain with the distributions of the absolute velocity, pressure and temperature. The slicing has been performed on the axial direction, and on the radial direction. Of the two slices on the radial direction, one crosses a planet and the other is in a zone far from planets. We have reported also the evolution of the temperature inside the PRE in Figure 6.36.

The small gaps sizes and the high portion of contact surface between planets, sun and ring produce an increase of temperature in time but also along the extruder axis. However, oppositely with respect to TSE, the temperature distribution is more uniform and free of localized peaks. This confirms the increased ability of the PRE in mixing. For what concerns

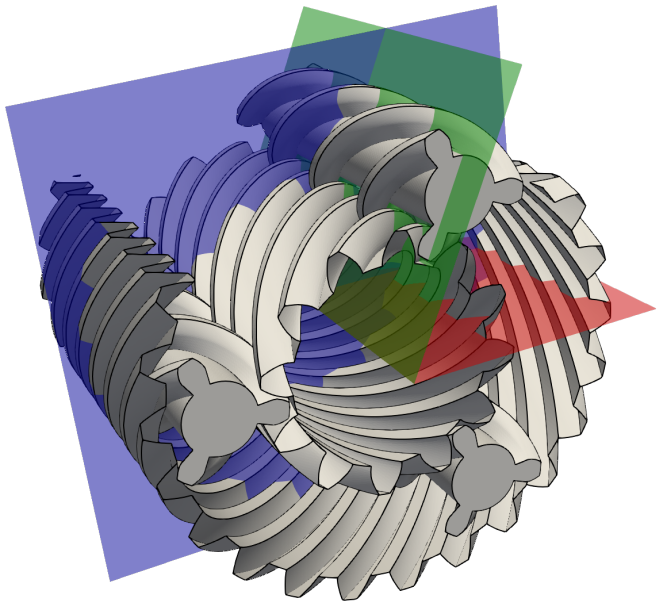


Figure 6.32: Representation of sampling slices on the PRE geometry.

the axial slice, we reported the fields in Figure 6.33.

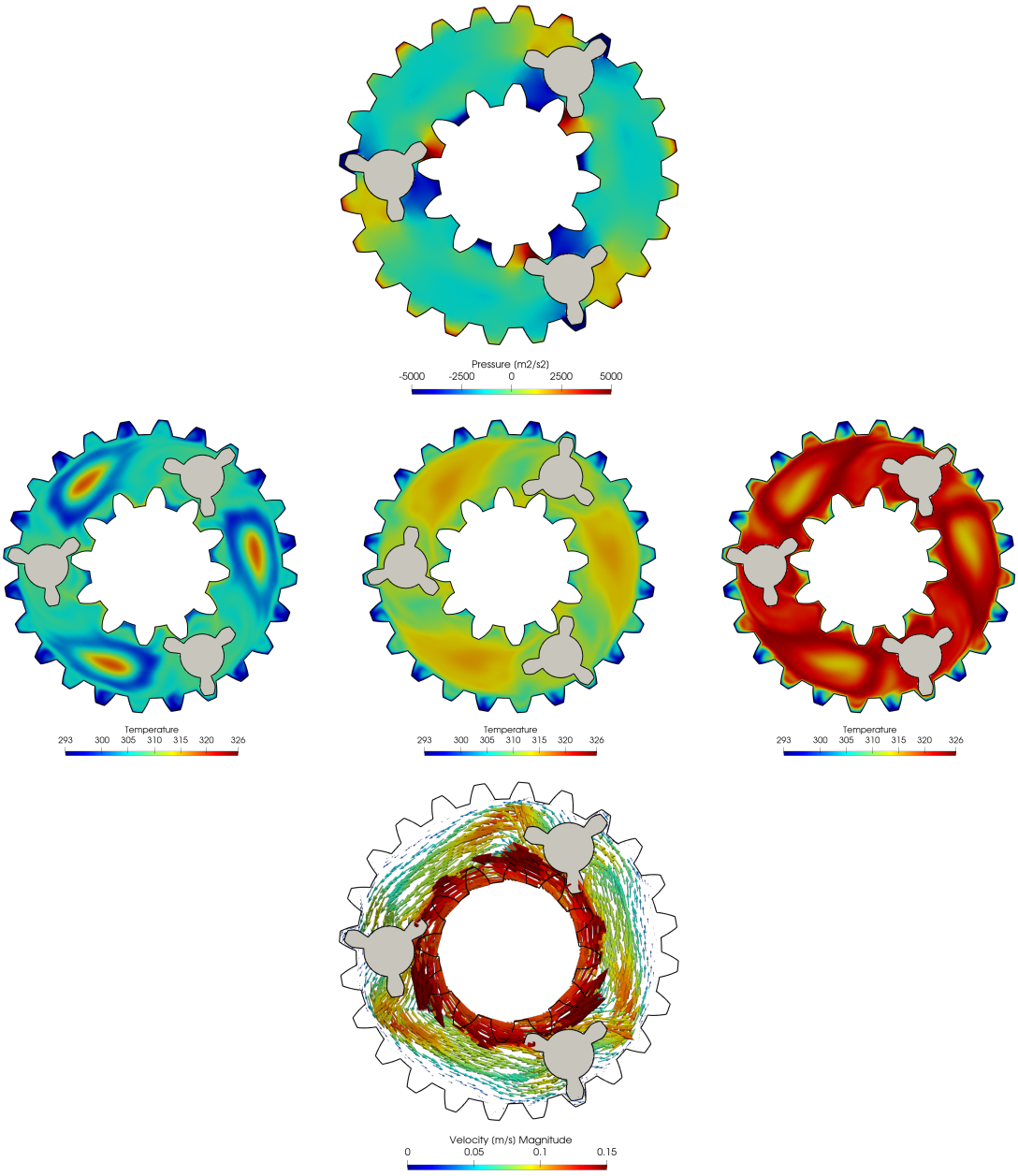


Figure 6.33: Pressure, velocity and temperature sampled on an axial slice after 20 seconds. For the temperature we reported also time instants 5 and 10 seconds in order to show the evolution of the field.

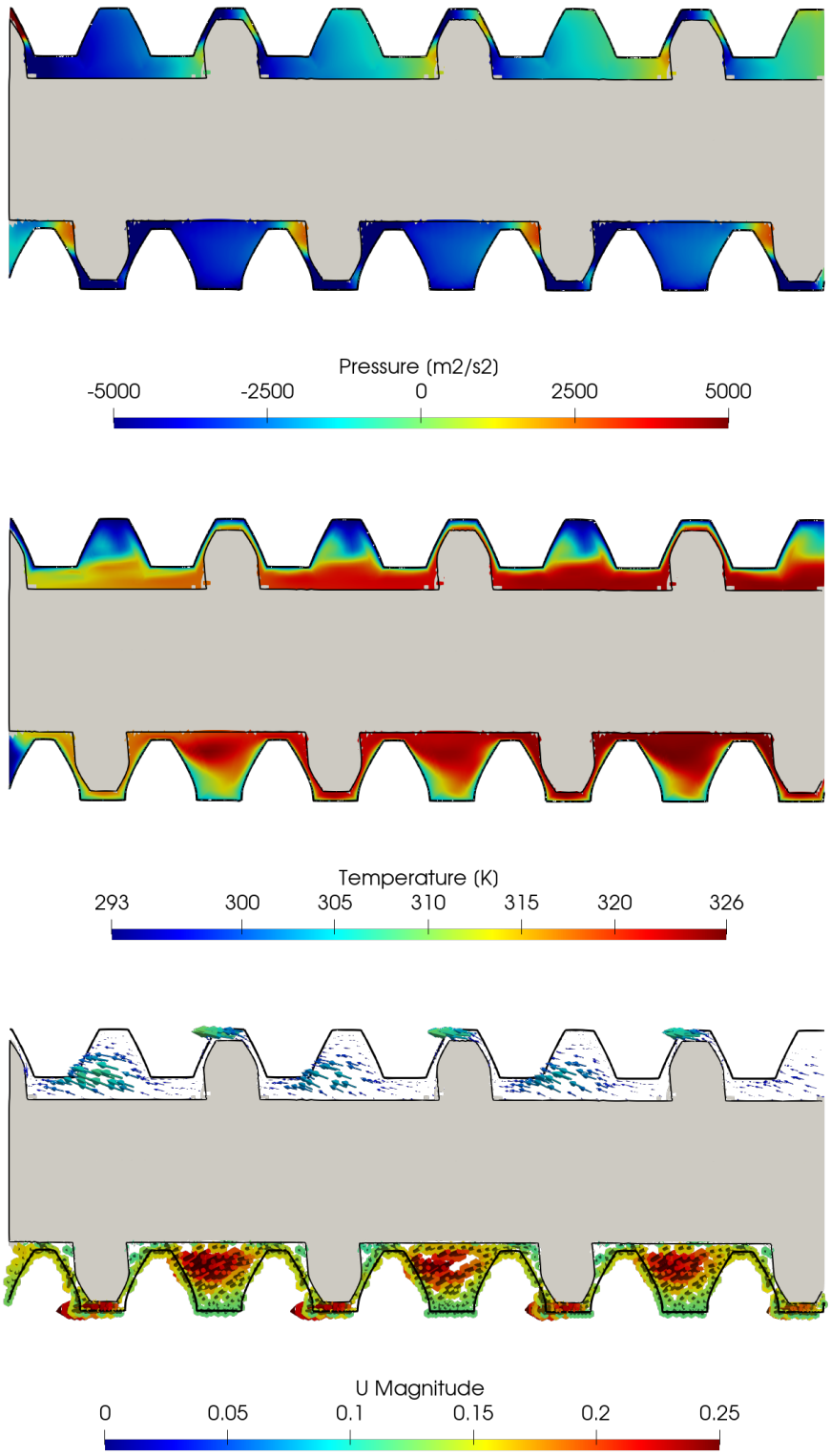


Figure 6.34: Pressure, velocity and temperature sampled on a longitudinal slice far from the planets after 20 seconds.

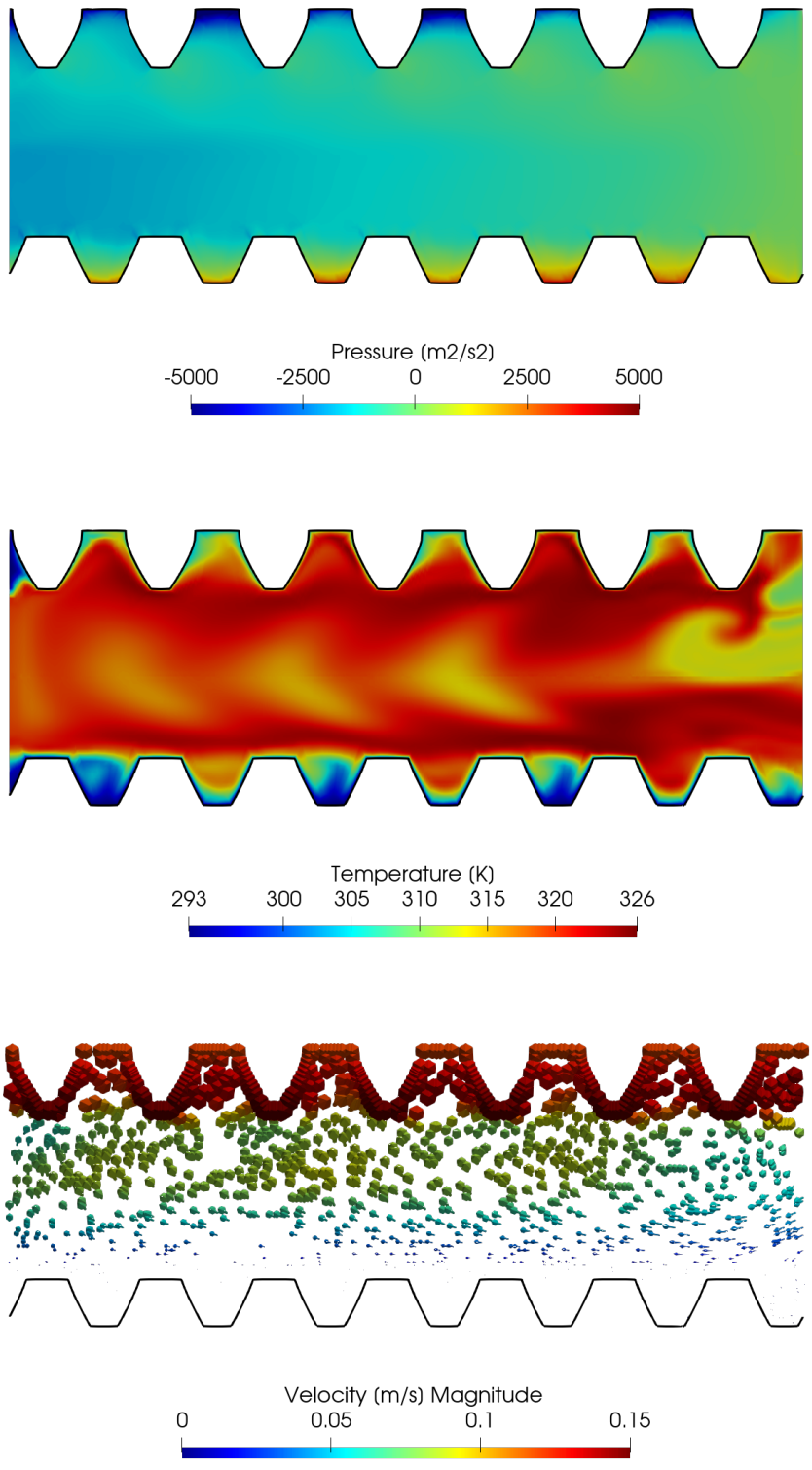


Figure 6.35: Pressure, velocity and temperature sampled on a longitudinal slice crossing a planet after 20 seconds.

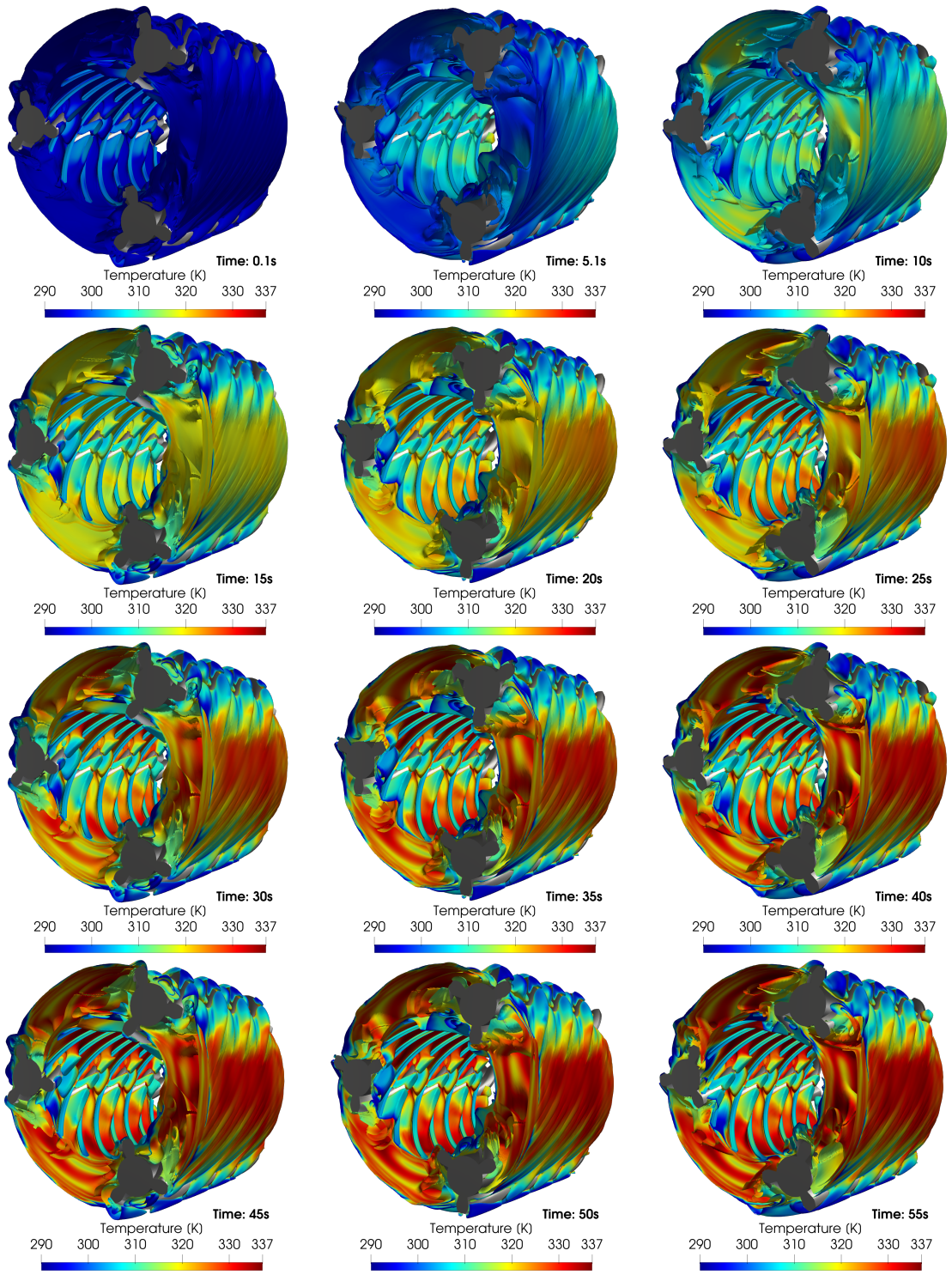


Figure 6.36: Temperature distribution evaluated for different time instants on relative velocity magnitude contours.

Chapter 7

Conclusions and perspectives

The recent advances in technology, in particular in the development of digital twins for industrial process design, simulation and monitoring, require a continuous development also for the underlying numerical and physical modelling. Even being in possess of sufficient computational power, the request of running high fidelity numerical simulations of real world phenomena is increasing. Hence, besides the computational efficiency, also the approximation algorithms have to be enhanced in accuracy, robustness and in the ability of describing the features of the problem in object.

In this work, we have investigated the Immersed Boundary method as an effective numerical tool for the prediction of complex phenomena involved in industrial polymer mixing processes. The complexity given by the geometries and the kinematics of industrial mixing devices make, in most cases, necessary to implement a non-conforming approach, especially when the problem has to be considered in a time dependent framework. Another reason is that the former method has shown to be able to handle a wide variety of problems due to its flexibility in embedding the geometries into the computational domain.

Moreover, the context of the Finite Volume method allows to robustly and efficiently dealing with the strong nonlinearity involved. Indeed, with an enough powerful computing system, real industrial problems are affordable with this tools in a relatively low computational time.

7.1 Main outcomes

In the present work, the research has been directed towards the development and analysis of a simulation tool able to deal with industrial mixing processes of polymer melt. In particular, we consider the Finite Volume method combined with the Immersed Boundary method to deal with the computational, physical and geometrical complexities of the problem. The work focused on the analysis of the combination of these methods.

We first developed a convergence analysis considering a simplified scenario, in which we applied the Box method combined with the Diffuse Interface method to the Poisson and the linear Stokes problems. The Box method formulation allowed us to work with the Finite Volume method in a variational framework. In particular for the Stokes problem, we were

able to prove, with the aid of numerical assessments, optimal error estimates for velocity and pressure of the Rhie-Chow stabilized Box method. We also included in the analysis, the contribution to the error of discrete approximations of fluxes through a consistency analysis.

Then, combining the Box method with the Diffuse Interface method, we analysed how the presence of a non-conforming boundary influences the numerical error. We found that introducing a non-conformal approximation of boundary conditions makes the convergence rate suboptimal with respect to the conforming method. In this analysis we exploited the dependence of the error on mesh and penalization parameters. In this way, we found that the non-conforming imposition of the boundary conditions introduces an error that depends only on neighbouring mesh elements size, which means that, employing an enough deep local mesh refinement, we are able to recover the original convergence rate of conforming methods.

Due to the high computational cost implied by this local mesh refinement, a more accurate approach than the Diffuse Interface method has been considered. We did this by implementing an Immersed Boundary method that able to recover the optimal convergence rate without resorting to refinements, but rather using a correction strategy for the imposition of non-conforming boundary conditions. A quadratic weighted least-square approximation uses the solution values in the neighbours of the immerse surface in order to impose a corrected, or shifted, solution to Immersed Boundary cells. This approach has proven to be accurate, robust and able to deal with several configurations of the immersed surfaces.

Then, the Immersed Boundary method showed its capability of dealing with complex problems in solving realistic industrial applications. Within this part, we showed the potentiality of our simulation tool in mainly two aspects. The computational efficiency in dealing with large scale problems, for which we assess the parallel scalability of the code, and the comparison with the commercial software ANSYS Polyflow, a standard tool for the simulation of polymer processing. In this comparison, we showed how our implementation overcomes the computational limitations of Polyflow on representative industrial problems.

In particular, we employed the Immersed Boundary method on three industrial mixing devices. The single-screw extruder geometry, a relatively simple multi-scale problem, in comparison to other industrial devices, the twin-screw extruder, a geometry that shows narrow intermeshing regions between the screws and that is built by several modules of different shapes and, finally, a more recent mixing technology, namely the planetary roller extruder. Our approach in handling complicated geometries and kinematics is novel in the literature and proves that the IBM implementation can be effectively adopted in the industrial practice. A more systematic validation of these result will be carried out.

7.2 Perspectives and future developments

In this work, our analysis of the Box method sets a base workflow to deal with classical Finite Volume methods in a variational framework in the context of non-conforming approach. This work can potentially be extended to other differential problems in the context of computational fluid dynamics, like the Navier-Stokes equations [133] and the non-Newtonian Stokes and Navier-Stokes equations, where the main difficulty is represented by the nonlin-

ear nature of the problem.

For what regards the Diffuse Interface, the adopted approach to prove error estimates can be potentially extended to any direct-forcing non-conforming approach. This is because the error devoted to the presence of an embedded geometry contributes to the global error only when estimating the interpolation error on mesh elements belonging to the discrete diffuse interface.

Major efforts in this research has been devoted towards the non-conforming imposition of boundary conditions. However, accurate approximations of mixing processes of polymer melt is subordinated to a good approximations of temperature and velocity boundary layers generated by the nonlinearity of the problem. Our Immersed Boundary method implementation showed some difficulties in well capturing these boundary layers. Moreover, the computational cost of the IBM remains high in general, because of the large number of least-squares interpolations that have to be performed on each IB cell during the computations.

These interpolation procedures also suffer when not enough points are employed in the stencil, which implies the solution of relatively large linear systems that are computationally onerous to invert at each time step. Indeed, IB subsets and stencils must be computed each time the embedded surface is moved. Hence, more accurate and robust interpolations have to be developed in order to better capture boundary layers and to make the method robust on the various geometric configuration that can be encountered when simulating industrial mixing processes, such as the narrow intermeshing region of the twin-screw extruder or to the thin gaps of the PRE, two examples of devices that have to be simulated in time dependent runs.

Some alternatives to our least-squares implementation are proposed in literature, such as radial basis functions [128] or physics-based interpolations, like divergence-free interpolations [12] and upwind inverse weighted distance interpolation [21].

Concerning the effective adoption of this kind of simulation tools in the industrial mixing practice, special attention should be paid in the future to two aspects. The first is the analysis of mixing ability of an industrial device. This aspect is of paramount importance when analysing mixing processes and it is highly depending on accurate CFD simulations of the mixing flow. A good mixing ability should combine both dispersive and distributive mixing. Dispersive mixing means how well the mixer is able to break a particulate in smaller pieces inside a polymer compound. Distributive mixing means how uniform the particulate is dispersed in the fluid. Higher mixing ability will result in a better integration of the filler in the compound.

This kind of studies are often performed using Lagrangian Particle Tracking, where a cloud of particles is introduced in a flow and their path is traced while they move in the fluid. Then, sampling certain quantities on each particle, like the stress under which a particle undergoes, the mixing index or the residence time, an analysis of how the particle is dispersed and eroded can be brought out. In the framework of Immersed Boundary methods, ad-hoc strategies have to be implemented in order to take into account the presence of a non-conforming boundary when tracing particles in the fluid [130, 132].

The second aspect that should deserve attention in the near future concerns the approximation we made to the degree of filling of the extruders. In our simulations we assumed that

the machines were completely filled with the fluid. In reality, the mixers have empty parts, especially at the inlet, where the flow is pumped in. Then, while the fluid is transported along the mixer length, the filled sectional area gradually increases until it is completely filled at the end of the device, where the polymer melt accumulates before being forced out through the die. Hence, free-surface effects have to be taken into account in order to obtain reliable results from simulations.

Among the many methods that are employed to deal with multiphase flows, the Volume of Fluid (VOF) method is a valid choice in order to track the evolution of the free-surface boundary within the mixing domain. Many challenges hide behind the simulation of multiphase non-Newtonian flows. For example, the interaction between the fluid, the air and the walls [84], that form a triple point in which boundary conditions are difficult to be modelled, or the fact that slip conditions can raise between the fluid and the walls. Moreover, in the context of this work, it would be challenging to study the interplay between VOF and IBM [111, 127].

Appendix A

Review of the finite volume method on general polyhedral grids

In sections A.1, A.2 and A.4 we introduce the basic notation for the OpenFOAM FVM, their discretization and the correction schemes employed in order to recover optimal convergence when dealing with general grids.

A.1 Preliminaries

To analyse the FVM, we consider as model problem the advection-diffusion equation. First consider the following elliptic problem, in conservative form. Let $\Omega \in \mathbb{R}^d$ be a Lipschitz polygonal domain and let $\Gamma = \partial\Omega = \Gamma^D \cup \Gamma^N$ be its boundary. The problem reads:

$$\begin{cases} -\nabla \cdot (k\nabla u - \mathbf{b}u) = f, & \text{in } \Omega, \\ u = g_D, & \text{on } \Gamma^D, \\ k\nabla u \cdot \mathbf{n} = g_N, & \text{on } \Gamma^N, \end{cases} \tag{A.1}$$

where $k \in L^\infty(\Omega), \mathbf{b} \in [L^\infty(\Omega)]^d, f \in L^2(\Omega)$ are the diffusivity, velocity field and source term, respectively, and $g_D \in H^{\frac{1}{2}}(\Omega)$ and $g_N \in H^{-\frac{1}{2}}(\Omega)$ are the Dirichlet and Neumann boundary data.

Let us now define the following functional space

$$\begin{aligned} \mathcal{V} &= H^1(\Omega), \\ \mathcal{V}_{\Gamma^D} &= \{v \in \mathcal{V} : v|_{\Gamma^D} = 0\}, \\ \mathcal{V}_{\Gamma^D, g_D} &= \{v \in \mathcal{V} : v|_{\Gamma^D} = g_D\}, \\ \mathcal{Q} &= L^2(\Omega). \end{aligned}$$

Now we multiply problem (A.1) by a test function $v \in \mathcal{Q}$ and integrate by parts obtaining the weak formulation of the problem: find $u \in \mathcal{V}_{\Gamma^D, g_D}$ such that

$$\begin{aligned} -\int_\Omega \nabla \cdot (k\nabla u)v + \int_\Omega \nabla \cdot (\mathbf{b}u)v &= (f, v)_\Omega \\ B(u, v) = a(u, v) + b(u, v) &= (f, v)_\Omega, \quad \forall v \in \mathcal{V}_{\Gamma_D}, \end{aligned} \tag{A.2}$$

where $(\cdot,\cdot)_\Omega$ is the standard scalar product in Ω, $B, a, b : \mathcal{V} \times \mathcal{Q} \to \mathbb{R}$ and $f \in L^2(\Omega)$.

To prove the well-posedness of problem (A.2), we recall the following result:

Theorem A.1.1 (Banach-Nečas-Babuška)**.** *Let $\mathcal{V}$ be Banach space and $\mathcal{Q}$ be a reflexive Banach space. Let $B \in \mathcal{L}(\mathcal{V} \times \mathcal{Q}, \mathbb{R})$ and $f \in \mathcal{V}^*$. Then the problem: find $u \in \mathcal{V}_{\Gamma_D, g_D}$ s.t.*

$$B(u, v) = (f, v)_\Omega, \quad \forall v \in \mathcal{Q},$$

is well-posed if and only if

$$\exists \alpha > 0 : \quad \inf_{v \in \mathcal{V}} \sup_{q \in \mathcal{Q}} \frac{B(v, w)}{\|v\|_{\mathcal{V}} \|q\|_{\mathcal{Q}}} \geq \alpha.$$

Moreover, the following holds:

$$\|u\|_{\mathcal{V}} \leq \frac{1}{\alpha} \|f\|_{\mathcal{V}^*}.$$

Well-posedness of problem A.2 comes from Theorem A.1.1. If the bilinear form B is considered setting $v = u$, then it will be continuous and coercive by the standard estimates for weak elliptic problems and the *inf–sup* condition is satisfied choosing $q = v$. Thus, the solution satisfies the following stability estimate:

$$\|u\|_{H^1} \leq C(\nu, \mathbf{b}) \left[\|f\|_{L^2} + \|g_N\|_{L^2(\Gamma_N)} + \|g\|_{H^{1/2}(\Gamma_D)} \right]. \tag{A.3}$$

Moreover, if the coefficients of the elliptic operator are Lipschitz continuous and the domain Ω is bounded, the solution will be $H^2(\Omega')$, $\forall \Omega' \subset\subset \Omega$.

A.2 The finite volume method

The finite volume method is traditionally introduced for the discretization of general conservation laws. A conservation law expresses the conservation of some physical quantity such as mass, momentum or energy. In general, it is convenient to adopt an Eulerian approach, so considering a region of space and writing the balance of the integral variation in this region of the quantity involved.

In the finite volume method, the domain is divided into control volumes and then a local balance is written on each control volume. Using the Gauss–Green theorem the formulation becomes a balance of fluxes over control volumes boundaries. Then, the fluxes on the boundary are discretized using appropriate numerical discretization schemes involving the discrete solution unknowns with a collocated approach.

The finite volume method is know also for its flexibility in handling general polyhedral meshes. On the other hand, mesh plays a fundamental role in terms of accuracy of numerical schemes. Thus we give a definition for the finite volume mesh and we require some properties on its regularity.

We recall first mesh definitions given in Chapter 4.

Definition A.2.1 (Finite volume mesh)**.** *Let $\Omega \subset \mathbb{R}^d$, where $d = 3$, be a Lipschitz bounded domain. Let $\mathcal{T}_h$ be a polyhedral tessellation of Ω.*

1. *Each polyhedron, cell, of $\mathcal{T}_h$ is denoted by K and is such that $\Omega = \bigcup K_i$ and $K_i \cap K_j = \emptyset$, $\forall K_i, K_j \in \mathcal{T}_h$. Let $h_K = diam(K)$ be the diameter of K and $h = \max_K\{h_K\}$. Let $\mathbf{k}$ and $|K|$ be the barycentre and the volume of cell K, respectively.*

2. *Let $\mathcal{F}_h$ be the set of faces F of $\mathcal{T}_h$. Let $\mathcal{F}_h = \mathcal{F}_h^o \cup \mathcal{F}_h^\partial$, where $\mathcal{F}_h^o$ is the set of internal faces and $\mathcal{F}_h^\partial$ is the set of boundary faces. Let $\mathbf{f}$ be the barycentre of face F and $|F|$ be its area.*

3. *Let $\mathcal{E}_h$ be the set of edges E of $\mathcal{T}_h$, with midpoint $\mathbf{e}$ and length $|E|$.*

4. *Let $\mathcal{P}_h$ be the set of vertices $\mathbf{p}$ of $\mathcal{T}_h$.*

Definition A.2.2 (Mesh connectivity)**.** *Consider a cell K_i. Let $\mathcal{F}_i = \{F : K_i \cap F \neq \emptyset\}$ be the set of faces of cell K_i. Moreover, denote by F_{ij} the face in common between cell K_i and K_j and denote by F_i^∂ each face of K_i living on $\partial\Omega$. Let also $\mathcal{K}_i$ be the set of cells adjacent to cell K_i and let, analogously, $\mathcal{E}_i$ and $\mathcal{P}_i$ as the sets of edges and vertices of K_i, respectively.*

Then, for each face F_{ij} associate a unit normal vector $\mathbf{n}_{ij}$, directed from cell i to cell j, a distance h_{ij} between centres $\mathbf{k}_i$ and $\mathbf{k}_j$, and a *diamond* D_{ij} as the polyhedron formed by the face vertices and the correspondent cell barycentres $\mathbf{k}_i$ and $\mathbf{k}_j$. Moreover, let D_{ij}^i, D_{ij}^j be the parts of the diamond D_{ij} belonging to cells K_i, K_j, respectively.

We now establish some regularity properties on the finite volume mesh.

Definition A.2.3 (Mesh assumptions and regularity)**.** $\forall K_i \in \mathcal{T}_h, F_{ij} \in \mathcal{F}_h, \exists C_1, C_2, C_3, C_4 > 0$*, independent of h:*

$$\begin{aligned} card(\mathcal{F}_i) \leq & C_1, \\ C_2 h^d \leq & |K_i| \leq h^d, \\ |F_{ij}| \leq & C_3 h^{d-1}, \\ h_{ij} \simeq & h_{K_j}, \text{ for some } K_j \in \mathcal{K}_i. \end{aligned}$$

Moreover, we define the following regularity indicators.

1. *Non-orthogonality: the cosine of the angle between $\mathbf{h}_{ij}$ and $\mathbf{n}_{ij}$,*

$$\tau_{ij} = \frac{\mathbf{h}_{ij} \cdot \mathbf{n}_{ij}}{h_{ij}}.$$

   It measures how much the two vectors are far from being parallel.

2. *Skewness: let $\mathbf{x}_{ij}$ be the intersection point between face F_{ij} and vector $\mathbf{h}_{ij}$. Let $\mathbf{m}_{ij} = \mathbf{f}_{ij} - \mathbf{x}_{ij}$, then the skewness factor reads*

$$\psi_{ij} = \frac{\|\mathbf{m}_{ij}\|}{h_{ij}}.$$

   It measures how far the intersection is from the face barycentre.

3. *Uniformity: let* $h_{ij}^* = \|\mathbf{x}_{ij} - \mathbf{k}^*\|, * = i, j$*, the local uniformity measure of a face reads*

$$\xi_{ij} = \frac{h_{ij}^i}{h_{ij}^i + h_{ij}^j}.$$

The more is near to 0.5, the more is uniform.

4. *Chunkiness:*

$$\delta = \frac{h}{h_m},$$

where $h_m = \min_K h_K$.

Once defined the computational mesh we can define a functional framework on it. When dealing with finite volume approximation, we consider a collocated approach, in which functions are defined by values in the barycentres of mesh cells and their fluxes across mesh faces are approximated using numerical schemes.

A function v_h is a piecewise constant function on tessellation $\mathcal{T}_h$ if it can be expressed as follows:

$$v_h = \sum_{K \in \mathcal{T}_h} v_K \mathbb{1}_K, \qquad v_K \in \mathbb{R} \, \forall K \in \mathcal{T}_h, \tag{A.4}$$

where $\mathbb{1}_K$ is the indicator function of cell K.

For this type of functions, we can define the finite volume counterparts of continuous spaces L^2 and H^1. The definition of discrete L^2 is straightforward:

$$\mathcal{Z}_h = \left\{ v_h \in L^2(\Omega) : \, v_h|_K \in \mathbb{P}^0(K), \, \forall K \in \mathcal{T}_h \right\},$$

where $\mathbb{P}^n$ is the space of polynomials of degree n.

For the definition of a discrete H^1, first we define the following discrete space of piecewise constant functions over face diamonds:

$$\mathcal{Z}_h^* = \left\{ v_h \in L^2(\Omega) : \, v_h|_{D_{ij}} \in \mathbb{P}^0(D_{ij}), \, \forall F_{ij} \in \mathcal{F}_h \right\}.$$

Then, we consider the orthogonal face normal gradient operator:

$$\nabla_h^\perp : \mathcal{T}_h \to \mathcal{F}_h, \qquad \nabla_{h,ij}^\perp v_h = \frac{v_i - v_j}{\tau_{ij} h_{ij}}, \qquad F_{ij} \in \mathcal{F}_h. \tag{A.5}$$

Using this operator, we are allowed to define the discrete H^1 space as

$$\mathcal{W}_h = \left\{ v_h \in \mathcal{Z}_h : \, \nabla_h^\perp v_h \in \mathcal{Z}_h^* \right\}.$$

Before writing the FVM formulation of a differential problem we need some more ingredients: face interpolators and face and cell gradients.

For the face interpolator we consider a weighted average between the two adjacent cell values. Let us define the following operator:

$$\mathcal{I}_h : \mathcal{T}_h \mapsto \mathcal{F}_h, \qquad \mathcal{I}_{h,ij} v_h = w_{ij} v_i + (1 - w_{ij}) v_j, \qquad F_{ij} \in \mathcal{F}_h, \tag{A.6}$$

where $w_{ij} \in (0,1)$ is the interpolation weight. Unless specified, for now on we consider the standard linear interpolation where

$$w_{ij} = \frac{h_{ij}^j}{h_{ij}^i + h_{ij}^j}. \tag{A.7}$$

It is worth noticing that, this interpolation scheme evaluates the solution in point $\mathbf{x}_{ij}$ (c.f. Definition A.2.3). For this reason, for general meshes, the consistency of the scheme is influenced by the skewness of the face. If it is significantly greater than zero, then we are evaluating the interpolation in a point that is far from the face barycentre. If this is the case, a specific correction can be applied using the skewness vector $\mathbf{m}_{ij}$ of face F_{ij}:

$$\mathcal{I}_{h,ij} v_h = w_{ij} v_i + (1 - w_{ij}) v_j + \mathbf{m}_{ij} \cdot \nabla_{F_{ij}} v_h, \tag{A.8}$$

where $\nabla_{F_{ij}}$ is some evaluation of the gradient on face F_{ij}, usually treated in an explicit way. The operator $\nabla_{F_{ij}}$ will be defined later on, when we introduce FVM gradient operators.

For what regards gradients, there exist two types of gradient: cell gradient and face normal gradient. The first derives from Gauss-Green gradient theorem: let $K_i \in \mathcal{T}_h, v \in H^1(\Omega)$,

$$\int_{K_i} \nabla v \mathrm{d}x = \int_{K_i} \nabla \cdot (\mathrm{I} v) \mathrm{d}x = \int_{\partial K_i} \mathrm{I} v \cdot \mathbf{n}_i \mathrm{d}s,$$

where I is the identity matrix. If we plug $v_h \in \mathcal{W}_h$ into the last expression we obtain the cell gradient operator, defined as

$$\nabla_h : \mathcal{T}_h \mapsto \mathcal{T}_h, \qquad \nabla_{h,i} v_h = \frac{1}{|K_i|} \sum_{K_j \in \mathcal{K}_i} |F_{ij}| \, \mathcal{I}_{h,ij} v_h \mathbf{n}_{ij}, \qquad K_i \in \mathcal{T}_h. \tag{A.9}$$

The second is the face normal gradient, usually discretized through a centred finite difference between two cell centres. We have already defined it in equation (A.5), in its orthogonal form. However, this formula does not computes accurately the normal gradient if the non-orthogonality is lower than 1 because the segment between two cell centres would not be aligned with the face normal vector. If this is the case, a correction is added to the numerical scheme in the following way. Let us write the face unit normal vector as the sum of two contributions: the first is parallel to $\mathbf{h}_{ij} = \mathbf{k}_j - \mathbf{k}_i$ and the second is computed accordingly. Thus, the scheme is applied along the first contribution and a correction is then performed along the second one. This is called *under-relaxation*:

$$\mathbf{n}_{ij} = \mathbf{n}_\perp + (\mathbf{n}_{ij} - \mathbf{n}_\perp),$$

where

$$\mathbf{n}_\perp = \frac{\mathbf{h}_{ij}}{\mathbf{h}_{ij} \cdot \mathbf{n}_{ij}} = \frac{\mathbf{h}_{ij}}{h_{ij} \tau_{ij}}. \tag{A.10}$$

The (non-orthogonally) corrected face normal gradient scheme finally reads:

$$\widetilde{\nabla}_h^\perp : \mathcal{T}_h \mapsto \mathcal{F}_h, \qquad \widetilde{\nabla}_{h_{ij}}^\perp v_h = \nabla_{h,ij}^\perp v_h + \nabla_{F_{ij}} v_h \cdot (\mathbf{n}_{ij} - \mathbf{n}_\perp), \qquad F_{ij} \in \mathcal{F}_h, \tag{A.11}$$

where $\nabla_{F_{ij}} v_h = \mathcal{I}_{h,ij} \nabla_h v_h$.

Remark A.1. [Boundary approximation of face normal gradient] On boundary faces, the form of the scheme changes. the non-orthogonality on boundary faces vanishes because the neighbour cell centre is ideally at infinite distance, hence $\mathbf{h}_{ij}$ is aligned with the face normal.

When on Dirichlet faces, the scheme drops back to its simplest form (A.5) using the Dirichlet value instead of v_j and h_{ij}^i instead of h_{ij}. On the other hand, when on Neumann faces, the whole scheme is substituted with the Neumann boundary value.

Remark A.2. [Face interpolation on boundary] As for the face normal gradient, we need to change the interpolation for boundary faces. The skewness of boundary faces is usually neglected.

When on Dirichlet faces, the result of the interpolation is Dirichlet value itself. On the other hand, when on Neumann faces, the weight w_{ij} becomes equal to one and the boundary value is extrapolated using a first order Taylor expansion:

$$\mathcal{I}_{h,ij} v_h = v_i + \tau_{ij} h_{ij} g_{N,ij},$$

where $g_{N,ij}$ will be the Neumann boundary value.

Now, discrete norms for FVM spaces can be defined.

Definition A.2.4 (Discrete norms)**.** *Let $v_h \in \mathcal{W}_h$,*

$$\begin{aligned}
\|v_h\|_{L^2} &= \sqrt{\sum_{K \in \mathcal{T}_h} |K| \, |v_K|^2}, \\
|v_h|_* &= \sqrt{\sum_{F \in \mathcal{F}_h} |D_{ij}| \left|\nabla_h^\perp v_h\right|^2}, \\
\|v_h\|_* &= \sqrt{\|v_h\|_{L^2}^2 + |v_h|_*^2}.
\end{aligned}$$

Moreover, these norms satisfy some properties:

Proposition A.2.1 (Discrete norms properties)**.** *The following properties hold: $\forall v_h \in \mathcal{W}_h$,*

$$\begin{aligned}
\|v_h\|_{L^2} &\leq C_p \, |v_h|_* \,, \\
|v_h|_* &\leq 2 \frac{1}{\underset{F_{ij} \in \mathcal{F}_h}{h_{ij}}} \|v_h\|_{L^2} \,.
\end{aligned}$$

Proof. 1. By [43, Lemma 5.1], that states that $\forall v_h \in \mathcal{W}_h$, $\exists C_p > 0$:

$$\|v_h\|_{L^2} \leq C_p \, |v_h|_* \,. \tag{A.12}$$

2.

$$
\begin{aligned}
|v_h|_* = & \left(\sum_{F_{ij} \in \mathcal{F}_h} \int_{F_{ij}} \frac{h_{ij}}{d} \left| \frac{v_i - v_j}{\tau_{ij} h_{ij}} \right|^2 \mathrm{d}s \right)^{\frac{1}{2}} \\
\leq & \frac{1}{\min\limits_{F_{ij} \in \mathcal{F}_h} \tau_{ij} h_{ij}} \left(\sum_{F_{ij} \in \mathcal{F}_h} \int_{F_{ij}} \frac{h_{ij}}{d} \left(|v_i|^2 + |v_j|^2 \right) \right)^{\frac{1}{2}} \\
\leq & 2(\tau_m h_m)^{-1} \|v_h\|_{L^2} = \frac{2\delta}{\tau_m h} \|v_h\|_{L^2} ,
\end{aligned}
$$

where δ is the chunkiness factor.

□

A.3 Finite volume formulation for advection-diffusion problems

In order to derive the FVM formulation of the problem, let

$$
\begin{aligned}
\mathcal{Z}_{h,\Gamma^D} &= \{ v_h \in \mathcal{Z}_h : \mathcal{I}_{h,ij}(v_h) = g_D|_{\mathbf{f}_{ij}}, \forall F_{ij} \in \mathcal{F}_h^D \}, \\
\mathcal{W}_{h,\Gamma^D} &= \{ v_h \in \mathcal{W}_h : v_h \in \mathcal{Z}_{h,\Gamma^D} \},
\end{aligned}
$$

be a discrete functional space, where $\mathcal{F}_h^D$ is the set of faces correspondent to the Dirichlet boundary Γ^D. Let $v_h \in \mathcal{Z}_h$ be a test function. Multiply problem (A.1) by v_h and integrate by parts:

$$
-\sum_{K_i \in \mathcal{T}_h} \int_{K_i} \nabla \cdot (k \nabla u - \mathbf{b}u) v_i \mathrm{d}s = -\sum_{K_i \in \mathcal{T}_h} \sum_{F_{ij} \in \mathcal{F}_i} \int_{F_{ij}} (k \nabla u - \mathbf{b}u) v_i \cdot \mathbf{n}_{ij} \mathrm{d}s = \sum_{K_i \in \mathcal{T}_h} (f, v_h)_{K_i}.
$$

The problem can be rewritten by defining the jump operator $[\![v_h]\!]_{ij} = v_i - v_j$ as

$$
-\sum_{F_{ij} \in \mathcal{F}_h} \int_{F_{ij}} (k \nabla u - \mathbf{b}u) \cdot \mathbf{n}_{ij} \, [\![v_k]\!]_{ij} \, \mathrm{d}s = \sum_{K_i \in \mathcal{T}_h} (f, v_h)_{K_i}.
$$

In the classical finite volume framework, the flux on the left-hand-side is approximated by a discrete operator. In this case we consider two operators: the face normal gradient (A.11) for the diffusion term and the face interpolator (A.8) for the convective term. The FVM formulation finally reads: find $u_h \in \mathcal{W}_{h,\Gamma^D}$, such that, $\forall v_h \in \mathcal{Z}_h$,

$$
\begin{aligned}
-\sum_{F_{ij} \in \mathcal{F}_h} \int_{ij} k_{ij} \nabla_h^{\perp} u_h \, [\![v_h]\!]_{ij} \, \mathrm{d}s + \sum_{F_{ij} \in \mathcal{F}_h} \int_{ij} \mathbf{b}_{ij} \cdot \mathbf{n}_{ij} \mathrm{d}s \mathcal{I}_h u_h \, [\![v_h]\!]_{ij} \, \mathrm{d}s &= \sum_{K_i \in \mathcal{T}_h} (f, v_h)_{K_i}, \\
B_h(u_h, v_h) = a_h(u_h, v_h) + b_h(u_h, v_h) &= \sum_{K_i \in \mathcal{T}_h} (f, v_h)_{K_i},
\end{aligned}
\tag{A.13}
$$

where $k_{ij}, \mathbf{b}_{ij}$ are evaluations of coefficients on face F_{ij}.

Notice that $B_h, a_h, b_h : \mathcal{Z}_h^0 \to \mathbb{R}$ are the discrete counterparts of B, a, b, respectively (c.f. equation A.2).

A.4 Convergence analysis

The convergence analysis for finite volumes is performed comparing the numerical solution against the continuous one and against the L^2-projection of the latter one on each cell. In order to obtain an estimate, we notice that the following holds:

$$B_h(u_h, v_h) = B(u, v_h).$$

Now we set $\Pi_{L^2} u$ to be the L^2-projection of u on each cell,

$$\Pi_{L^2} u = \sum_{K \in \mathcal{T}_h} \frac{1}{|K|} \int_K u \mathrm{d}x,$$

and $v_h = u_h - \Pi_{L^2} u$, then we can write the following relationship:

$$B_h(u_h - \Pi_{L^2} u, u_h - \Pi_{L^2} u) = B(u, u_h - \Pi_{L^2} u) - B_h(\Pi_{L^2} u, u_h - \Pi_{L^2} u). \tag{A.14}$$

We now require some property on B_h in order to ensure eventually the convergence of the method.

Definition A.4.1 (Continuity)**.** *B_h is continuous if*

$$\exists M > 0 : |B_h(u_h, v_h)| \leq M \left|u_h\right|_* \left|v_h\right|_*, \quad \forall u_h, v_h \in \mathcal{Z}_h.$$

Definition A.4.2 (Coercivity)**.** *B_h is coercive if*

$$\exists \alpha > 0 : |B_h(u_h, u_h)| \geq \alpha \left|u_h\right|_*^2, \quad \forall u_h \in \mathcal{Z}_h.$$

Definition A.4.3 (Consistency)**.** *Let $R_{ij}(u)$ be the flux residual on a face F_{ij}, defined as follows:*

$$R_{ij}(u) = \frac{1}{|F_{ij}|} \int_{F_{ij}} \left[-k_{ij} \nabla_h^\perp \Pi_{L^2} u + (k \nabla u + \mathbf{b}_{ij} \mathcal{I}_h \Pi_{L^2} u - \mathbf{b} u) \cdot \mathbf{n}_{ij} \right] [\![v_h]\!]_{ij} \, \mathrm{d}s.$$

The residual is consistent with order p if

$$\exists C > 0 : \left\| R_{ij}(u) \right\|_{L^2} = \sqrt{\sum_{F_{ij} \in \mathcal{F}_h} |D_{ij}| \left| R_{ij}(u) \right|^2} \leq C h^p.$$

Proposition A.4.1. *If B_h is consistent, in the sense of Definition A.4.3, then*

$$\exists C > 0 : \quad |B(u, v_h) - B_h(\Pi_{L^2} u, v_h)| \leq C h^p \left|v_h\right|_*, \quad \forall u \in \mathcal{V}_{\Gamma^D}, v_h \in \mathcal{Z}_h.$$

Proof. By Hölder's inequality and by the fact that $[\![v_h]\!]_{ij}$ is a number, we have:

$$\begin{aligned}
|B(u, v_h) - B_h(\Pi_{L^2} u, v_h)| = & \left| \sum_{F_{ij} \in \mathcal{F}_h} \int_{F_{ij}} \left[-k_{ij} \nabla_h^\perp \Pi_{L^2} u + (k \nabla u + \mathbf{b}_{ij} \mathcal{I}_h \Pi_{L^2} u - \mathbf{b} u) \cdot \mathbf{n}_{ij} \right] [\![v_h]\!]_{ij} \right| \mathrm{d}s \\
\leq & \sum_{F_{ij} \in \mathcal{F}_h} |F_{ij}| \left| R_{ij}(u) \right| |v_i - v_j| \\
= & \sum_{F_{ij} \in \mathcal{F}_h} d \, |D_{ij}| \left| R_{ij}(u) \right| \left| \frac{v_i - v_j}{\tau_{ij} h_{ij}} \right| \\
\leq & d \left\| R_{ij}(u) \right\|_{L^2} \left|v_h\right|_* \leq C h^p \left|v_h\right|_*,
\end{aligned}$$

where we exploited the volume of diamond D_{ij} as

$$\frac{|\tau_{ij} h_{ij}|\,|F_{ij}|}{d}.$$

□

Before proving a convergence estimate for the FVM solution, we need to verify that B satisfies Definitions A.4.1 and A.4.2, assuming that A.4.3 holds. For the sake of simplicity we consider the orthogonal and non-skew case.

Lemma A.4.1. *The bilinear form B_h of problem* (A.13) *is continuous.*

Proof. Let us consider the expression of B_h in the general case:

$$\begin{aligned} B_h(u_h, v_h) &= \sum_{F_{ij}\in\mathcal{F}_h} |F_{ij}| \left[-k_{ij} \widetilde{\nabla}_h^{\perp} u_h + \mathbf{b}_{ij}\cdot\mathbf{n}_{ij} \mathcal{I}_h u_h \right] [\![v_h]\!] \\ &= \sum_{F_{ij}\in\mathcal{F}_h} -k_{ij}\,|F_{ij}| \left(\frac{u_j - u_i}{\tau_{ij} h_{ij}} + \nabla_F u_h \cdot (\mathbf{n}_{ij} - \mathbf{n}_{\perp}) \right) [\![v_h]\!]_{ij} \\ &\quad + \mathbf{b}_{ij}\cdot\mathbf{n}_{ij}\,|F_{ij}|\,(w_{ij} u_i + (1-w_{ij}) u_j + \mathbf{m}_{ij}\cdot\nabla_F u_h)\,[\![v_h]\!]_{ij}\,. \end{aligned} \tag{A.15}$$

Now, assuming $\mathbf{n}_{ij}$ s.t. $\mathbf{b}_{ij}\cdot\mathbf{n}_{ij} \geq 0$, $\forall F_{ij} \in \mathcal{F}_h$, orthogonality and non-skewness the latter equation reduces to

$$\begin{aligned} B_h(u_h, v_h) &= \sum_{F_{ij}\in\mathcal{F}_h} -k_{ij}\,|F_{ij}| \left(\frac{u_j - u_i}{\tau_{ij} h_{ij}} \right) [\![v_h]\!]_{ij} \\ &\quad + \mathbf{b}_{ij}\cdot\mathbf{n}_{ij}\,|F_{ij}|\,(w_{ij} u_i + (1-w_{ij}) u_j)\,[\![v_h]\!]_{ij} \\ &\leq \sum_{F_{ij}\in\mathcal{F}_h} \frac{|F_{ij}|}{h_{ij}} \left[\left(\frac{k_{ij}}{\tau_{ij}} + \mathbf{b}_{ij}\cdot\mathbf{n}_{ij} w_{ij} \right) u_i - \left(\frac{k_{ij}}{\tau_{ij}} - \mathbf{b}_{ij}\cdot\mathbf{n}_{ij}(1-w_{ij}) \right) u_j \right] [\![v_h]\!]_{ij} \\ &\leq \sum_{F_{ij}\in\mathcal{F}_h} \frac{|F_{ij}|}{h_{ij}} \left| \frac{k_{ij}}{\tau_{ij}} + \mathbf{b}_{ij}\cdot\mathbf{n}_{ij} w_{ij} \right| [\![u_h]\!]_{ij}\,[\![v_h]\!]_{ij} \\ &\leq \sum_{F_{ij}\in\mathcal{F}_h} h_{ij}\,|F_{ij}|\,C(k,\mathbf{b}) \frac{[\![u_h]\!]_{ij}}{h_{ij}} \frac{[\![v_h]\!]_{ij}}{h_{ij}}, [\![v_h]\!]_{ij} \end{aligned} \tag{A.16}$$

that, by Cauchy-Schwarz and Hölder inequalities, gives the result. □

Lemma A.4.2. *The bilinear form B_h of problem* (A.13) *is coercive.*

Proof.

$$\begin{aligned} B_h(u_h, u_h) &= \sum_{F_{ij}\in\mathcal{F}_h} |F_{ij}| \left[-k_{ij} \nabla_{F_{ij}} u_h + \mathbf{b}_{ij}\cdot\mathbf{n}_{ij} \mathcal{I}_h u_h \right] [\![u_h]\!] \\ &= \sum_{F_{ij}\in\mathcal{F}_h} -k_{ij}\,|F_{ij}| \left(\frac{u_j - u_i}{\tau_{ij} h_{ij}} + \nabla_F u_h \cdot (\mathbf{n}_{ij} - \mathbf{n}_{\perp}) \right) [\![u_h]\!]_{ij} \\ &\quad + \mathbf{b}_{ij}\cdot\mathbf{n}_{ij}\,|F_{ij}|\,(w_{ij} u_i + (1-w_{ij}) u_j + \mathbf{m}_{ij}\cdot\nabla_F u_h)\,[\![u_h]\!]_{ij}\,. \end{aligned} \tag{A.17}$$

Using now

$$(wb+(1-w)a)(b-a)=\left(w-\frac{1}{2}\right)(a-b)^2+\frac{1}{2}\left(b^2-a^2\right),$$

we obtain:

$$\begin{aligned}
B_h(u_h,u_h) &= \sum_{F_{ij}\in\mathcal{F}_h}|F_{ij}|\,\tau_{ij}h_{ij}\left[k_{ij}+\mathbf{b}_{ij}\cdot\mathbf{n}_{ij}\tau_{ij}h_{ij}\left(w_{ij}-\frac{1}{2}\right)\right]\left(\frac{u_j-u_i}{\tau_{ij}h_{ij}}\right)^2\\
&\quad+\sum_{F_{ij}\in\mathcal{F}_h}|F_{ij}|\,\frac{\mathbf{b}_{ij}\cdot\mathbf{n}_{ij}}{2}\left(u_i^2-u_j^2\right)\\
&\quad+\sum_{F_{ij}\in\mathcal{F}_h}|F_{ij}|\left[-k_{ij}(\mathbf{n}_{ij}-\mathbf{n}_\perp)+\mathbf{b}_{ij}\cdot\mathbf{n}_{ij}\mathbf{m}_{ij}\right]\cdot\nabla_F u_h(u_i-u_j)\\
&= \sum_{F_{ij}\in\mathcal{F}_h}d\,|D_{ij}|\left[k_{ij}+\mathbf{b}_{ij}\cdot\mathbf{n}_{ij}\tau_{ij}h_{ij}\left(w_{ij}-\frac{1}{2}\right)\right]\left(\frac{u_j-u_i}{\tau_{ij}h_{ij}}\right)^2\\
&\quad+\sum_{K_i\in\mathcal{T}_h}u_i^2\int_{K_i}\nabla\cdot\mathbf{b}\\
&\quad+\sum_{F_{ij}\in\mathcal{F}_h}|F_{ij}|\left[-k_{ij}(\mathbf{n}_{ij}-\mathbf{n}_\perp)+\mathbf{b}_{ij}\cdot\mathbf{n}_{ij}\mathbf{m}_{ij}\right]\cdot\nabla_F u_h(u_i-u_j).
\end{aligned}\tag{A.18}$$

By the assumption of orthogonality and non-skewness the equation becomes

$$\begin{aligned}
B_h(u_h,u_h) &= \sum_{F_{ij}\in\mathcal{F}_h}d\,|D_{ij}|\left[k_{ij}+\mathbf{b}_{ij}\cdot\mathbf{n}_{ij}\tau_{ij}h_{ij}\left(w_{ij}-\frac{1}{2}\right)\right]\left(\frac{u_j-u_i}{\tau_{ij}h_{ij}}\right)^2\\
&\quad+\sum_{K_i\in\mathcal{T}_h}u_i^2\int_{K_i}\nabla\cdot\mathbf{b}.
\end{aligned}\tag{A.19}$$

To obtain coercivity we need $\nabla\cdot\mathbf{b}\geq 0$ for the second addendum and, for the first one, that

$$k_{ij}+\mathbf{b}_{ij}\cdot\mathbf{n}_{ij}\tau_{ij}h_{ij}\left(w_{ij}-\frac{1}{2}\right)\geq 0\qquad\Longrightarrow\qquad\frac{k_{ij}}{\tau_{ij}h_{ij}\mathbf{b}_{ij}\cdot\mathbf{n}_{ij}}\geq\frac{1-w_{ij}}{2},$$

that means

$$\mathbb{P}\mathrm{e}_h\leq\frac{2}{1-w_{ij}}$$

or

$$w_{ij}\geq 1-\frac{2}{\mathbb{P}\mathrm{e}_h}.$$

□

Thus, the following convergence result holds.

Theorem A.4.1. *Let $u \in \mathcal{V}_{\Gamma^D}$ be the solution of problem* (A.2) *and let $u_h \in \mathcal{W}_{h,\Gamma^D}$ be the solution of problem* (A.13)*. If $\Pi_{L^2} u \in \mathcal{Z}_h$ is the L^2-projection of u on each cell of mesh $\mathcal{T}_h$, B_h is continuos, coercive and consistent with order p, then*

$$\|u - u_h\|_{L^2} + |\Pi_{L^2} u - u_h|_* \leq C(h + \frac{1}{\alpha} h^p).$$

Moreover, the solution exists, it is unique and stable,

$$|u_h|_* \leq \frac{1}{\alpha} \|f\|_{L^2}.$$

Proof. Assume B_h satisfies continuity, coercivity and consistency with order p, Definitions A.4.1, A.4.2 and A.4.3. Using Proposition A.4.1:

$$\begin{aligned} \alpha \, |\Pi_{L^2} u - u_h|_*^2 \leq & B_h(\Pi_{L^2} u - u_h, \Pi_{L^2} u - u_h) \\ \leq & \, |B_h(\Pi_{L^2} u, \Pi_{L^2} u - u_h) - B(u, \Pi_{L^2} u - u_h)| \\ \leq & C h^p \, |\Pi_{L^2} u - u_h|_* . \end{aligned} \tag{A.20}$$

For what regards the L^2-projection we use the following result from [25]: let $G \subset \mathbb{R}^d$ be a bounded convex domain and let $u \in \mathcal{V}$, then

$$\|u - \Pi_{L^2} u\|_{L^2(G)} \leq C_d \frac{\text{diam}(G)^d}{|G|^{1-\frac{1}{d}}} \|\nabla u\|_{L^2(G)} .$$

Let $G = K$,

$$\begin{aligned} \int_\Omega (u - \sum_{K \in \mathcal{T}_h} \Pi_{L^2} u_K \mathbb{1}_K)^2 = & \int_\Omega \left(\sum_{K \in \mathcal{T}_h} (u - \Pi_{L^2} u_K) \mathbb{1}_K \right)^2 \leq \int_\Omega \sum_{K \in \mathcal{T}_h} |u - \Pi_{L^2} u_K|^2 \, \mathbb{1}_K \\ = & \sum_{K \in \mathcal{T}_h} \int_K |u - \Pi_{L^2} u_K|^2 \leq \sum_{K \in \mathcal{T}_h} C_d^2 \frac{h_K^{2d}}{h_K^{2d\left(1-\frac{1}{d}\right)}} \|\nabla u\|_{L^2(K)}^2 \\ \leq & C_d^2 h^2 \|\nabla u\|_{L^2} , \end{aligned} \tag{A.21}$$

from which follows, by Lemma A.2.1,

$$\begin{aligned} \|u - \Pi_{L^2} u\|_{L^2} \leq & \|u - u_h\|_{L^2} + \|\Pi_{L^2} u - u_h\|_{L^2} \\ \leq & \|u - u_h\|_{L^2} + |\Pi_{L^2} u - u_h|_* \\ \leq & C(h + \frac{1}{\alpha} h^p) \|u\|_{H^1} . \end{aligned} \tag{A.22}$$

Finally, stability follows from coercivity of B and by continuity of F:

$$\alpha \, |u_h|_*^2 \leq |B_h(u_h, u_h)| = |F(u_h)| \leq \|f\|_{L^2} \, |u_h|_* . \tag{A.23}$$

□

Remark A.3. Regarding Theorem A.4.1, we enlighten that the FVM error estimates is very similar to estimates proven for classical finite element methods. We have two terms involve: the interpolation and the consistency errors, respectively h and h^p.

Remark A.4. For the convergence analysis we drop the hypothesis of non-orthogonality and skewness of the mesh. This is in fact reasonable when proving continuity and coercivity because all the corrections for grid irregularities go on the right hand side and are treated explicitly.

This because OpenFOAM implements almost always iterative algorithms, so that iterating on the same problem is like resolving it with a fixed point approach. In practice, the mesh irregularities play a role similar to the nonlinearity that a problem can present. Hence, being the corrections on the right-hand-side of the equation, they do not enter in matrix construction and then neither they do influence continuity and coercivity of B_h.

However, those corrections influence the convergence of the FVM by playing a role in the consistency estimate. Indeed, in Proposition A.4.1, what is to be estimated is the difference between the continuous fluxes and the finite volume schemes applied to the L^2 projection of the continuous solution. This can be seen in the presence of operators $\nabla_h^{\perp}$ and $\mathcal{I}_h$, that contain the non-orthogonality and skewness corrections.

Bibliography

[1] I. Aavatsmark, T. Barkve, Ø. Bøe, and T. Mannseth. Discretization on unstructured grids for inhomogeneous, anisotropic media. Part I: Derivation of the methods. *SIAM Journal on Scientific Computing*, 19:1700–1716, 1998.

[2] I. Aavatsmark, T. Barkve, Ø. Bøe, and T. Mannseth. Discretization on unstructured grids for inhomogeneous, anisotropic media. Part II: Discussion and numerical results. *SIAM Journal on Scientific Computing*, 19:1717–1736, 1998.

[3] R. A. Adams and J. F. F. Fournier, editors. *Sobolev Spaces*, volume 140 of *Pure and Applied Mathematics*. Elsevier, 2003.

[4] R. Agroum, S. M. Aouadi, C. Bernardi, and J. Satouri. Spectral discretization of the navier-stokes equations coupled with the heat equation. *ESAIM: Mathematical Modelling and Numerical Analysis*, 49(3):621–639, 2015.

[5] R. Agroum, C. Bernardi, and J. Satouri. Spectral discretization of the time-dependent Navier–Stokes problem coupled with the heat equation. *Applied Mathematics and Computation*, 268:59–82, 2015.

[6] P. Angot, C.H. Bruneau, and P. Fabrie. A penalization method to take into account obstacles in viscous flows. *Numerische Mathematik*, 81:497–520, 1999.

[7] R. Araya, G. R. Barrenechea, and F. Valentin. Stabilized finite element methods based on multiscale enrichment for the Stokes problem. *SIAM Journal on Numerical Analysis*, 44(1):322–348, 2006.

[8] K. Aziz and A. Settari. Petroleum reservoir simulation. 1979.

[9] I. Babuška. Error-bounds for finite element method. *Numerische Mathematik*, 16(4):322–333, 1971.

[10] I. Babuška. The finite element method with penalty. *Mathematics of Computation*, 27:221–228, 1973.

[11] R. E. Bank and D. J. Rose. Some error estimates for the box method. *SIAM Journal on Numerical Analysis*, 24(4):777–787, 1987.

[12] Y. Bao, A. Donev, B. Griffith, D. McQueen, and C. Peskin. An immersed boundary method with divergence-free velocity interpolation and force spreading. *Journal of Computational Physics*, 347:183–206, 2017.

[13] M. W. Batton and H. Rust. Adhesive processing using a planetary extruder. ENTEX Rust & Mitschke GmbH, Bochum, Germany.

[14] Y. Bazilevs and T. Hughes. NURBS-based isogeometric analysis for the computation of flows about rotating components. *Computational Mechanics*, 43:143–150, 12 2008.

[15] M. Behr and T. Tezduyar. The shear-slip mesh update method. *Computer Methods in Applied Mechanics and Engineering*, 174:261–274, 05 1999.

[16] B. C. Bell and K. S. Surana. p-version least squares finite element formulation for two-dimensional, incompressible, non-Newtonian isothermal and non-isothermal fluid flow. *International Journal for Numerical Methods in Fluids*, 18(2):127–162, 1994.

[17] M. Belliard and C. Fournier. Penalized direct forcing and projection schemes for Navier–Stokes. *Comptes Rendus Mathematique*, 348(19):1133–1136, 2010.

[18] S. Berrone, A. Bonito, R. Stevenson, and M. Verani. An optimal adaptive fictitious domain method. *Mathematics of Computation*, 88(319):2101–2134, 2019.

[19] S. Berrone and E. Süli. Two-sided a posteriori error bounds for incompressible quasi-Newtonian flows. *IMA Journal of Numerical Analysis*, 28(2):382–421, 2007.

[20] M. Bessemoulin-Chatard, C. Chainais-Hillairet, and F. Filbet. On discrete functional inequalities for some finite volume schemes. *IMA Journal of Numerical Analysis*, 35(3):1125–1149, 2014.

[21] A. S. Bharadwaj and S. Ghosh. Data reconstruction at surface in immersed-boundary methods. *Computers & Fluids*, 196:104236, 2020.

[22] D. Boffi and L. Gastaldi. A finite element approach for the immersed boundary method. volume 81, pages 491–501. 2003. In honour of Klaus-Jürgen Bathe.

[23] L. Bonaventura, A. Iske, and E. Miglio. Kernel-based vector field reconstruction in computational fluid dynamic models. *International Journal for Numerical Methods in Fluids*, 66(6):714–729, 2011.

[24] C. Börgers and O. B. Widlund. On finite element domain imbedding methods. *SIAM Journal on Numerical Analysis*, 27(4):963–978, 1990.

[25] M. V. Borsuk and V. A. Kondratiev. Elliptic boundary value problems of second order in piecewise smooth domains. volume 69. North-Holland Mathematical Library, 2006.

[26] S. C. Brenner and L. R. Scott. *The Mathematical Theory of Finite Element Methods*, volume 15 of *Texts in Applied Mathematics*. Springer, 2008.

[27] F. Brezzi and J. Pitkäranta. *On the Stabilization of Finite Element Approximations of the Stokes Equations*, pages 11–19. Vieweg+Teubner Verlag, Wiesbaden, 1984.

[28] R. Budynas and K. Nisbett. Shigley's mechanical engineering design. 2014.

[29] M. Burger, O. Elvetun, and M. Schlottbom. Analysis of the diffuse domain method for second order elliptic boundary value problems. *Foundations of Computational Mathematics*, 2014.

[30] E. Burman and A. Ern. An Unfitted Hybrid High-Order Method for Elliptic Interface Problems. *SIAM Journal on Numerical Analysis*, 56(3):1525–1546, 2018.

[31] E. Burman and P. Hansbo. Fictitious domain finite element methods using cut elements: I. A stabilized Lagrange multiplier method. *Computer Methods in Applied Mechanics and Engineering*, 199(41-44):2680–2686, 2010.

[32] E. Burman and P. Hansbo. Fictitious domain finite element methods using cut elements: II. A stabilized Nitsche method. *Applied Numerical Mathematics. An IMACS Journal*, 62(4):328–341, 2012.

[33] Z. Cai. On the finite volume element method. *Numerische Mathematik*, 58(1):713–735, 1990.

[34] A. Cangiani, Z. Dong, E. H. Georgoulis, and P. Houston. *hp-Version Discontinuous Galerkin Methods on Polygonal and Polyhedral Meshes*. Springer Cham, 2017.

[35] P. Chatzipantelidis. A finite volume method based on the crouzeix–raviart element for elliptic PDE's in two dimensions. *Numerische Mathematik*, 82(3):409–432, 1999.

[36] P. Chatzipantelidis. Finite volume methods for elliptic PDE's : a new approach. *ESAIM: Mathematical Modelling and Numerical Analysis - Modélisation Mathématique et Analyse Numérique*, 36(2):307–324, 2002.

[37] G. Chesshire and W. D. Henshaw. Composite overlapping meshes for the solution of partial differential equations. *Journal of Computational Physics*, 90(1):1–64, 1990.

[38] P. G. Ciarlet. *The finite element method for elliptic problems*, volume 40 of *Classics in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2002.

[39] M. L. Combrinck, L. N. Dala, and I. I. Lipatov. Eulerian derivation of non-inertial navier–stokes and boundary layer equations for incompressible flow in constant pure rotation. *European Journal of Mechanics - B/Fluids*, 65:10–30, 2017.

[40] R. K. Connelly and J. L. Kokini. Examination of the mixing ability of single and twin screw mixers using 2D finite element method simulation with particle tracking. *Journal of Food Engineering*, 79(3):956–969, 2007.

[41] Alessandro Cortesi. Mathematical modelling of rheology of filled compounds. Master's thesis, Politecnico di Milano, 2021.

[42] J. A. Cottrell, T. J. R. Hughes, and Y. Bazilevs. *Isogeometric analysis*. John Wiley & Sons, Ltd., Chichester, 2009. Toward integration of CAD and FEA.

[43] Y. Coudière, J.P. Vila, and P. Villedieu. Convergence rate of a finite volume scheme for a two dimensional convection-diffusion problem. *ESAIM: Mathematical Modelling and Numerical Analysis - Modélisation Mathématique et Analyse Numérique*, 33(3):493–516, 1999.

[44] Y. Coudière and P. Villedieu. Convergence rate of a finite volume scheme for the linear convection-diffusion equation on locally refined meshes. *ESAIM: Mathematical Modelling and Numerical Analysis - Modélisation Mathématique et Analyse Numérique*, 34(6):1123–1149, 2000.

[45] M. Cui and X. Ye. Superconvergence of finite volume methods for the stokes equations. *Numerical Methods for Partial Differential Equations*, 25:1212 – 1230, 2009.

[46] J. Deteix and D. Yakoubi. Improving the pressure accuracy in a projection scheme for incompressible fluids with variable viscosity. *Applied Mathematics Letters*, 79:111–117, 2018.

[47] J. Deteix and D. Yakoubi. Shear rate projection schemes for non-Newtonian fluids. *Computer Methods in Applied Mechanics and Engineering*, 354:620–636, 2019.

[48] R. M. Dhenge, J. J. Cartwright, M. J. Hounslow, and A. D. Salman. Twin screw granulation: Steps in granule growth. *International Journal of Pharmaceutics*, 438(1):20–32, 2012.

[49] D. Di Pietro and J. Droniou. A hybrid high-order method for leray–lions elliptic equations on general meshes. *Mathematics of Computation*, 86, 2016.

[50] J. J. Díaz, P. J. Garcia Nieto, A. Bello-García, J. Muñoz, and J. Ordieres-Meré. Finite volume modeling of the non-isothermal flow of a non-newtonian fluid in a rubber's extrusion die. *Journal of Non-Crystalline Solids*, 354:5334–5336, 2008.

[51] B. Dominic. *Existence Theory for Generalized Newtonian Fluids*. Academic Press, 2017.

[52] J. Donea, S. Giuliani, and J.P. Halleux. An arbitrary lagrangian-eulerian finite element method for transient dynamic fluid-structure interactions. *Computer Methods in Applied Mechanics and Engineering*, 33(1):689 – 723, 1982.

[53] J. Droniou and R. Eymard. A mixed finite volume scheme for anisotropic diffusion problems on any grid. *Numerische Mathematik*, 105(1):35–71, 2006.

[54] H. Elman, V. Howle, J. Shadid, R. Shuttleworth, and R. Tuminaro. A taxonomy and comparison of parallel block preconditioners for the incompressible Navier–Stokes equations. *Journal of Computational Physics*, 227:1790–1808, 2008.

[55] E. Erik Burman and P. Hansbo. Edge stabilization for the generalized Stokes problem: A continuous interior penalty method. *Computer Methods in Applied Mechanics and Engineering*, 195(19):2393–2410, 2006.

[56] A. Ern and J.L. Guermond. *Theory and Practice of Finite Elements*. Applied Mathematical Sciences. Springer New York, 2004.

[57] A. Ern and M. Vohralík. *A Unified Framework for a posteriori Error Estimation in Elliptic and Parabolic Problems with Application to Finite Volumes*, volume 4, pages 821–837. 2011.

[58] R. Ewing and T. Lin. On the accuracy of the finite volume element method based on piecewise linear polynomials. *SIAM Journal on Numerical Analysis*, 39, 2002.

[59] R. Eymard, T. Gallouët, and R. Herbin. Finite volume methods. *Handbook of numerical analysis*, 7(January):713–1018, 2000.

[60] E. A. Fadlun, R. Verzicco, P. Orlandi, and J. Mohd-Yusof. Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations. *Journal of Computational Physics*, 161(1):35–60, 2000.

[61] M. Feistauer, J. Felcman, M. Lukácová-Medvid'ová, and G. Warnecke. Error estimates for a combined finite volume–finite element method for nonlinear convection–diffusion problems. *SIAM Journal on Numerical Analysis*, 36(5):1528–1548, 1999.

[62] C. Fernandes, L. L. Ferrás, F. Habla, O. S. Carneiro, and J. M. Nóbrega. Implementation of partial slip boundary conditions in an open-source finite-volume-based computational library. *Journal of Polymer Engineering*, 39(4):377–387, 2019.

[63] L. L. Ferrás, J. M. Nóbrega, and F.T. Pinho. Implementation of slip boundary conditions in the finite volume method: new techniques. *International Journal for Numerical Methods in Fluids*, 72(7):724–747.

[64] J.D. Ferry. *Viscoelastic Properties of Polymers*. Springer, Dordrecht, 1980.

[65] J. H. Ferziger, M. Perić, and R. L. Street. *Computational Methods for Fluid Dynamics*. Springer Nature Switzerland AG 2020, 2020.

[66] S. Forte, L. Preziosi, and Vianello M. *Meccanica dei Continui*. Springer, 2019.

[67] G. Galdi. *An Introduction to the Mathematical Theory of the Navier–Stokes Equations*, volume I. Springer New York, 2011.

[68] Y. Gao, Y. Li, G. Yuan, and Z. Sheng. New finite volume element methods in the ale framework for time-dependent convection–diffusion problems in moving domains. *Journal of Computational and Applied Mathematics*, 393:113537, 2021.

[69] V. Girault and P. A. Raviart. Finite Element Methods for Navier-Stokes equations – Theory and Algorithms. In *Springer Series in Computational Mathematics*, 1986.

[70] C. G. Gogos and Z. Tadmor. *Principles of Polymer Processing*. Wiley, 2013.

[71] C. Greenshields and H. Weller. *Notes on Computational Fluid Dynamics: General Principles*. CFD Direct Ltd, Reading, UK, 2022.

[72] J.L. Guermond, P. Minev, and J. Shen. An overview of projection methods for incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, 195(44):6011–6045, 2006.

[73] W. Hackbusch. On first and second order box schemes. *Computing. Archives for Scientific Computing*, 41(4):277–296, 1989.

[74] A. Hansbo and P. Hansbo. An unfitted finite element method, based on Nitsche's method, for elliptic interface problems. *Computer Methods in Applied Mechanics and Engineering*, 191(47):5537–5552, 2002.

[75] J. Helmig, M. Behr, and S. Elgeti. Boundary-conforming finite element methods for twin-screw extruders: Unsteady - temperature-dependent - non-newtonian simulations. *Computers & Fluids*, 190:322–336, 2019.

[76] J. Helmig and S. Elgeti. *A Sliding Interface Approach with Application to Twin-Screw Extruders*. Universitätsbibliothek der RWTH Aachen, 2019.

[77] J. Hinz, J. Helmig, M. Möller, and S. Elgeti. Boundary-conforming finite element methods for twin-screw extruders using spline-based parameterization techniques. *Computer Methods in Applied Mechanics and Engineering*, 361:112740, 2020.

[78] J. Hinz, M. Möller, and C. Vuik. Spline-based parameterization techniques for twin-screw machine geometries. *IOP Conference Series: Materials Science and Engineering*, 425(1):012030, 2018.

[79] C. W. Hirt, A. A. Amsden, and J. L. Cook. An arbitrary Lagrangian-Eulerian computing method for all flow speeds. volume 135, pages 198–216. 1997.

[80] P. Houston, C. Schwab, and E. Süli. Discontinuous hp-finite element methods for advection-diffusion-reaction problems. *SIAM Journal on Numerical Analysis*, 39(6):2133–2163, 2002.

[81] T. J. R. Hughes, W. K. Liu, and T. K. Zimmermann. Lagrangian-Eulerian finite element formulation for incompressible viscous flows. *Computer Methods in Applied Mechanics and Engineering*, 29(3):329–349, 1981.

[82] G. Iaccarino. Immersed boundary technique for turbulent flow simulations. *Applied Mechanics Reviews*, 56, 05 2003.

[83] T. Ikeno and T. Kajishima. Finite-difference immersed boundary method consistent with wall conditions for incompressible turbulent flow simulations. *Journal of Computational Physics*, 226(2):1485–1508, Oct 2007.

[84] M. Jamshidzadeh, F. Ein-Mozaffari, and A. Lohi. Local and overall gas holdup in an aerated coaxial mixing system containing a non-Newtonian fluid. *AIChE Journal*, 66(11):e17016, 2020.

[85] H. Jasak. *Error Analysis and Estimation for Finite Volume Method with Applications to Fluid Flow*. PhD thesis, Imperial College, University of London, 1996.

[86] H. Jasak, D. Rigler, and Ž. Tuković. Design and implementation of immersed boundary method with discrete forcing approach for boundary conditions. 11th. World Congress on Computational Mechanics - WCCM XI, 5th. European Congress on Computational Mechanics - ECCM V, 6th European Congress on Computational Fluid Dynamics - ECFD VI, 2014.

[87] F. Juretić and A. D. Gosman. Error analysis of the finite-volume method with respect to mesh type. *Numerical Heat Transfer, Part B: Fundamentals*, 57(6):414–439, 2010.

[88] B. Kadoch, D. Kolomenskiy, P. Angot, and K. Schneider. A volume penalization method for incompressible flows and scalar advection–diffusion with moving obstacles. *Journal of Computational Physics*, 231(12):4365–4383, 2012.

[89] E. N. Karatzas, G. Stabile, L. Nouveau, G. Scovazzi, and G. Rozza. A reduced-order shifted boundary method for parametrized incompressible Navier–Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 370:113273, 2020.

[90] T. Kataoka, T. Kitano, M. Sasahara, and K. Nishijima. Viscosity of particle filled polymer melts. *Rheologica Acta*, 17(2):149–155, 1978.

[91] J. Kettemann, I. Gatin, and C. Bonten. Verification and validation of a finite volume immersed boundary method for the simulation of static and moving geometries. *Journal of Non-Newtonian Fluid Mechanics*, 290:104510, 2021.

[92] N. Kim, H. Kim, and J. Lee. Numerical analysis of internal flow and mixing performance in polymer extruder i: Single screw element. *Korea-Australia Rheology Journal*, 18:143–151, 2006.

[93] I. M. Krieger and T. J. Dougherty. Mechanism for non-Newtonian flow in suspensions of rigid particles. *Transactions of the Society of Rheology*, 3:137–152, 1959.

[94] R. J. LeVeque. *Numerical methods for conservation laws (2. ed.)*. Lectures in mathematics. Birkäuser, 1992.

[95] X. Li, J. Lowengrub, A. Rätz, and A. Voigt. Solving PDEs in complex geometries: a diffuse domain approach. *Communications in Mathematical Sciences*, 7(1):81–107, 2009.

[96] A. Limper, S. Seibel, and G. Fattmann. Compounding unit planetary roller extruder. *Macromolecular Materials Engineering*, 287:815–823, 2002.

[97] R. K.S. Liu, K.C. Ng, and T. W.H. Sheu. A volume of solid implicit forcing immersed boundary method for solving incompressible Navier-Stokes equations in complex domain. *Computers & Fluids*, 218:104856, 2021.

[98] C. W. Macosko and R. G. Larson. *Rheology: principles, aeasurements, and applications*. Advances in interfacial engineering series. Wiley-VCH, 1994.

[99] A. Main and G. Scovazzi. The shifted boundary method for embedded domain computations. Part I: Poisson and Stokes problems. *Journal of Computational Physics*, 372:972–995, 2018.

[100] S. H. Maron and P. E. Pierce. Application of Ree-Eyring generalized flow theory to suspensions of spherical particles. *Journal of Colloid Science*, 11:80–95, 1956.

[101] D. M. C. Martins, D. M. S. Albuquerque, and J. C. F. Pereira. Continuity constrained least-squares interpolation for SFO suppression in immersed boundary methods. *Journal of Computational Physics*, 336:608–626, 2017.

[102] S. Middleman. *Fundamentals of Polymer Processing*. McGraw-Hill, 1977.

[103] R. Mittal and G. Iaccarino. Immersed boundary methods. *Annual Review of Fluid Mechanics*, 37(1):239–261, 2005.

[104] M. Mohan Rai. A conservative treatment of zonal boundaries for euler equation calculations. *Journal of Computational Physics*, 62(2):472–503, 1986.

[105] M. Mohan Rai. An implicit, conservative, zonal-boundary scheme for euler equation calculations. *Computers & Fluids*, 14(3):295–319, 1986.

[106] F. Moukalled, L. Mangani, and M. Darwish. *The Finite Volume Method in Computational Fluid Dynamics: An Advanced Introduction with OpenFOAM and Matlab*. Springer Publishing Company, Incorporated, 1st edition, 2015.

[107] G Negrini, N. Parolini, and M. Verani. A diffuse interface box method for elliptic problems. *Applied Mathematics Letters*, 120:107314, 2021.

[108] J. Nitsche. Über ein variationsprinzip zur lösung von dirichlet-problemen bei verwendung von teilräumen, die keinen randbedingungen unterworfen sind. In *Abhandlungen aus dem mathematischen Seminar der Universität Hamburg*, volume 36, pages 9–15. Springer, 1971.

[109] G. Pascazio and M. Napolitano. A staggered-grid finite volume method for the vorticity-velocity equations. *Computers & Fluids*, 25(4):433–446, 1996.

[110] S. V. Patankar. *Numerical Heat Transfer and Fluid Flow*. Electro Skills Series. Hemisphere Publishing Corporation, 1980.

[111] H. V. Patel, S. Das, J. A. M. Kuipers, J. T. Padding, and E. A. J. F. Peters. A coupled volume of fluid and immersed boundary method for simulating 3d multiphase flows with contact line dynamics in complex geometries. *Chemical Engineering Science*, 166:28–41, 2017.

[112] C. S. Peskin. Flow patterns around heart valves: A numerical method. *Journal of Computational Physics*, 10(2):252–271, 1972.

[113] C. S. Peskin. The immersed boundary method. *Acta Numerica*, 11:479–517, 2002.

[114] A. J. Poslinski, M. E. Ryan, R. Gupta, S. G. Seshadri, and F. J. Frechette. Rheological behavior of filled polymeric systems i. Yield stress and shear-thinning effects. *Journal of Rheology*, 32:703–735, 1988.

[115] A. Quarteroni and R. Ruiz-Baier. Analysis of a finite volume element method for the stokes problem. *Numerische Mathematik*, 118(4):737–764, 2011.

[116] A. M. Quarteroni. *Numerical Models for Differential Problems*. Springer International Publishing, AG 2017, 2017.

[117] A. M. Quarteroni and A. Valli. *Numerical Approximation of Partial Differential Equations*. Springer Publishing Company, Incorporated, 1st ed. 1994. 2nd printing edition, 2008.

[118] C. Rauwendaal. *Polymer Extrusion*. Carl Hanser Verlag GmbH & Company KG, 2014.

[119] C. M. Rhie and W. L. Chow. Numerical study of the turbulent flow past an airfoil with trailing edge separation. *AIAA Journal*, 21(11):1525–1532, 1983.

[120] J. Rudloff, M. Lang, M. Bastian, K. Kretschmer, P. Heidemeyer, and M. Koch. Analysis of the process behavior of co-kneaders. *AIP Conference Proceedings*, 2055(1):020007, 2019.

[121] T. Sakurai, K. Yoshimatsu, N. Okamoto, and K. Schneider. Volume penalization for inhomogeneous Neumann boundary conditions modeling scalar flux in complicated geometry. *Journal of Computational Physics*, 390:452–469, 2019.

[122] S. Salsa. *Partial Differential Equations in Action: From Modelling to Theory*. Springer Publishing Company, Incorporated, 2nd edition, 2015.

[123] R. Scardovelli and S. Zaleski. Direct numerical simulation of free-surface and interfacial flow. *Annual Review of Fluid Mechanics*, 31(1):567–603, 1999.

[124] M. Schlottbom. Error analysis of a diffuse interface method for elliptic problems with Dirichlet boundary conditions. *Applied Numerical Mathematics. An IMACS Journal*, 109:109–122, 2016.

[125] J. L. Steger and J. A. Benek. On the use of composite grid schemes in computational aerodynamics. *Computer Methods in Applied Mechanics and Engineering*, 64(1):301–320, 1987.

[126] A. F. Stephansen. Discontinuous Galerkin methods and a posteriori error analysis for heterogenous diffusion problems. Technical report, 2007.

[127] X. Sun and M. Sakai. Numerical simulation of two-phase flows in complex geometries by using the volume-of-fluid/immersed-boundary method. *Chemical Engineering Science*, 139:221–240, 2016.

[128] F. Toja-Silva, J. Favier, and A. Pinelli. Radial basis function (RBF)-based interpolation and spreading for the immersed boundary method. *Computers & Fluids*, 105:66–75, 2014.

[129] Y.H. Tseng and J. H. Ferziger. A ghost-cell immersed boundary method for flow in complex geometry. *Journal of Computational Physics*, 192(2):593–623, 2003.

[130] M. Uhlmann. An immersed boundary method with direct forcing for the simulation of particulate flows. *Journal of Computational Physics*, 209(2):448–476, 2005.

[131] H. K. Versteeg and W. Malalasekera. *An introduction to computational fluid dynamics: the finite volume method*. Harlow, 2007.

[132] Z. Wang, J. Fan, and K. Luo. Combined multi-direct forcing and immersed boundary method for simulating flows with moving particles. *International Journal of Multiphase Flow*, 34(3):283–302, 2008.

[133] J. Wen, Y. He, and X. Zhao. Analysis of a new stabilized finite volume element method based on multiscale enrichment for the Navier-Stokes problem. *International Journal of Numerical Methods for Heat & Fluid Flow*, 26:2462–2485, 2016.

[134] J. Winck and S. Frerich. Numerical simulation of fluid flow and mixing dynamics inside planetary roller extruders. *International Polymer Processing*, 36(5):508–518, 2021.

[135] J. Xu and Q. Zou. Analysis of linear and quadratic simplicial finite volume methods for elliptic equations. *Numerische Mathematik*, 111(3):469–492, 2009.

[136] X. Ye. On the relation between finite volume and finite element methods applied to the Stokes equations. *Numerical Methods for Partial Differential Equations*, 17:440 – 453, 2001.

[137] S. Zhang, X. Zhao, and S. Bayyuk. Generalized formulations for the Rhie–Chow interpolation. *Journal of Computational Physics*, 258:880–914, 2014.