UNIVERSITÀ DEGLI STUDI DELL'AQUILA
DIPARTIMENTO DI SCIENZE FISICHE E CHIMICHE

Dottorato di Ricerca in Scienze Fisiche e Chimiche

XXXII ciclo

Titolo della tesi

A numerical study of the Drag parameter for the description of ICME dynamics

SSD FIS/06

Dottorando

Gianluca Napoletano

Coordinatore del corso

Prof. Antonio Mecozzi

Tutor

Prof. Massimo Vellante

Relatori

Dott. Ermanno Pietropaolo

Dott. Dario Del Moro

A.A. 2018/2019

Contents

Abstract

Gli Interplanetary Coronal Mass Ejections (ICME) sono violenti fenomeni legati all'attività del Sole con ripercussioni sull'intera eliosfera, consistenti nell'espulsione di grosse quantità di materiale solare nello spazio interplanetario. La previsione del loro impatto sui diversi corpi del sistema solare è uno degli obiettivi primari della disciplina dello Space Weather.

I modelli numerici di plasma, impiegati per la descrizione dell'evoluzione del plasma interplanetario attraverso accurate simulazioni, possono essere considerati, in linea di principio, il metodo più affidabile per fornire informazioni sullo stato del vento solare e sulle strutture che in esso si muovono. Tuttavia, un simile approccio genera un'enorme quantità di dati, e richiede la disponibilità di grosse risorse computazionali, insieme a lunghi tempi di calcolo.

D'altra parte, semplici modelli empirici, basati su leggi sperimentali e una ragionevole scelta di caratteristiche, contribuiscono a fornire previsioni rapide, e consentono una parametrizzazione della fisica mancante o ancora non sufficientemente compresa. In modo altrettanto importante, questo approccio consente di eseguire previsioni multiple ed indipendenti in breve tempo, con conseguente possibilità di valutare l'incertezza delle previsioni attraverso l'analisi delle statistiche che emergono dalla valutazione dei risultati di insieme. Nonostante ciò possa sembrare un approccio meno diretto rispetto a quello numerico, negli ultimi anni i modelli empirici sono stati notevolmente rivalutati, nell'ambito dello Space Weather, per le procedure di fast warning, nonchè come approccio complementare ai metodi computazionali.

Il tempo di viaggio di un ICME dal Sole alla Terra viene tipicamente calcolato attraverso il modello basato sul drag fluidodinamico (Drag-Based Model), che assume una semplice equazione del moto per l'ICME, definendo la sua accelerazione come $a = -\gamma(v - w)|v - w|$, dove v è la velocità CME, w è la velocità ambientale del vento solare e γ il cosiddetto *parametro di drag* (Vršnak et al., 2013). A dispetto di una complessa dipendenza di questo parametro dalle caratteristiche fisiche e geometriche del plasma, la tipica ipotesi di lavoro impiegata nell'applicare il DBM assume che w e γ siano quantità costanti lontano dal Sole. I valori di input per γ e w sono tipicamente derivati dalla conoscenza preesistente delle condizioni del vento solare o dalla risoluzione del problema inverso, in cui il tempo di percorrenza dell'ICME è noto e le incognite sono i parametri del modello. Nell'approccio 'Ensemble' (Dumbovich et al., 2018; Napoletano et al. 2018), l'incertezza sui valori effettivi di tali input è resa dalle distribuzioni di probabilità.

Nel corso di questo lavoro viene studiato un possibile miglioramento del modello Drag-Based attraverso l'uso dei risultati di simulazioni numeriche e l'analisi di dati sperimentali. La prima sezione presenta il dettaglio delle tecniche di calcolo numerico adoperate per la creazione e l'esecuzione di codici da impiegare per la simulazione numerica della dinamica di fluidi viscosi.

Nella seconda parte, dopo aver descritto la fenomenologia legata alla fisica dei corpi solidi in moto relativo rispetto a fluidi, viene presentato in dettaglio il modello a drag fluidodinamico impiegato per il forecasting, e la procedura per la determinazione dei parametri del modello a partire dai dati sperimentali. I risultati delle simulazioni sono quindi confrontati con i dati sperimentali, allo scopo di procedere on un'interpretazione degli stessi su base fisica. Particolare enfasi è posta sulla questione, tuttora aperta, legata alla possibilità che diversi regimi di velocità relativa tra ICME e vento solare influenzino in maniera dterminante il valore del parametro di drag, come avviene tipicamente per oggetti solidi che si spostano in un fluido esterno.

Introduction

Coronal Mass Ejections are violent phenomena of solar activity with repercussions throughout the entire heliosphere. It is now well-established that Interplanetary Coronal Mass Ejections, i.e. their manifestation into interplanetary space and solar wind, are the major cause of geomagnetic disturbances, with enormous economic consequences on our technology, in addition to severe risks for space missions, telecommunication disturbances and the potential hazard for high-flying aircraft and astronauts during missions of space exploration. All of these effects contributed in making the techniques for monitoring, tracking and predicting the arrival of such events at Earth one of the primary tasks in space-weather forecasting.

Numerical models describing heliospheric processes through accurate simulations of interplanetary plasma evolution, may be thought as the most reliable method to provide very detailed information about structures moving in the solar wind and their related effects. This approach requires a very detailed knowledge of the state of the heliosphere, in order to properly model interplanetary plasma and magnetic field, as well as Coronal Mass Ejections kinematic and dynamic models. All of these quantities must be measured through several instruments, in general with remote sensing observational methods. These measures provide data with associated errors and uncertainties, that the detailed plasma numerical simulations can only ingest after careful conditioning. This implies tremendous computational power, long times and the generation of a huge amount of data.

On the other hand, simple empirical models, based on experimental laws and a reasonable choice of features, contribute in providing quick and fluent forecasts and allow a parameterization of the missing or poorly understood physics. As much important, the possibility to run multiple, independent forecasts in short time allows to evaluate the forecast uncertainty through the analysis of the statistics emerging from ensemble evaluations. Despite this may seem a less direct approach in a world consenting Physics-based numerical simulations, in recent years empirical models have been greatly reevaluated for fast warning procedures in Space Weather and as a complementary approach to computational methods.

The purpose of this work is to study a possible improvement of a widely employed forecasting model, the *Drag-Based Model*, through the use of numerical simulations and experimental data analysis.

This thesis is composed of four chapters and can be divided into two main parts: the first one comprises chapters 1 and 2 and is focused on Mathematical and Computational Physics, dealing with numerical methods for differential equations and applications to fluid dynamics. In the second part (Chapters 3 and 4) I employ such methods for the refinement of a specific forecasting model, discussing the simulation results and comparison with experimental data.

In Chapter 1, the basic elements of numerical methods for Ordinary Differential Equations are presented and discussed in detail; classical and more advanced methods for differential problems, along with discussions about stability and convergence of the presented schemes are also considered.

Chapter 2 presents numerical methods for Partial Differential Equations, and bears on the exposition of Chapter 1. Discretization of partial derivatives and schemes for time advancement of systems of PDEs are presented, and then applied to classical fluid dynamic models as Euler and Navier-Stokes equations, allowing for specific applications and comparison with literature.

Chapter 3 is focused on a study of the drag force on solid bodies immersed in a uniform flow stream. The two-dimensional case of the cylinder in a fluid stream is taken as a principal reference for discussing the outcome of the different fluid regimes and to test our simulations, in order to show the limits and efficiency of the numerical codes at our disposal. Bearing in mind such results, numerical simulations of the drag force on several shapes are performed in the two-dimensional case.

Chapter 4 finally presents the details of the model we employ to forecast the ICME arrival to the Earth or any other point in the heliosphere. This model is based on fluid dynamic assumptions to describe the dynamics an ICME moving into the interplanetary space. We propose a refinement of this model by assuming that the efficiency of the interaction of an ICME with the solar wind may depend on the relative velocity of the structure with respect to the surrounding solar wind. Employing a sample of ICME events, for which we assessed a robust association between remote observations and in-situ measurements, we proceed to an evaluation of the model parameters to test our hypothesis with experimental values.

Chapter 1

Numerical methods

This chapter is mainly dedicated to the theory of the numerical techniques for the construction of the algorithms employed for the numerical simulations discussed in chapters 2 and 3.

Numerical methods for differential equations are certainly a big topic in mathematics, and therefore we must restrict our attention to a very specific choice of subjects, mainly depending on the precise applications we are going to consider. Despite this, this chapter starts from the most basic aspects of numerical methods, employing a rather formal mathematical language, to offer a concise view about this topic.

After recalling some basic elements related to the theory of Ordinary Differential Equations (ODEs), methods for domain discretization, discrete derivative representation and solution of ODEs are presented. Such schemes and methods are then extended to the case of Partial Differential Equations (PDEs), which will be used later to describe the physical problems of our interest. In conclusion, frequency filtering methods to control undesired numerical effects inherent to the method itself are discussed.

1.1 Numerical methods for ODEs

1.1.1 Introduction and notations

During this chapter, the solution to the general differential problem will be indicated by y. For the ease of explaining basic concepts, we will focus on the scalar case: the general theory for a system of ODEs where $y \in \mathbb{R}^d$ is rather directly related to this exposition and can be found in the majority of books about numerical methods for ODEs, e.g. [Lam91].

Throughout this chapter, we will make extensive use of the *big-O* notation:

Definition 1 (Big O notation) *A function $f(h)$ is a $O(h^p)$ as $h \to 0$ if*

$$\exists\, k,\, \delta > 0 : |f(h)| < k\, h^p \quad \forall\, h < \delta$$

meaning that for h sufficiently small, as in a limit evaluation, one can properly substitute f with $k\,h^p$.

Concerning the approximate quantities on the numerical grid, we will make use of the notation $y_n \approx y(t_n)$ to indicate that the quantity y_n approximates an exact (analytic) quantity $y(t_n)$ using a numerical method.

1.1.2 Basic ODE theorems

We start by considering the following first-order Initial Value Problem (hereafter IVP):

$$\begin{cases} y' = f(t,y) & t \in [t_0, T] \\ y(t_0) = y_0 \end{cases} \tag{1.1}$$

Since in general such a problem does not possess a unique solution, (or any solution at all), we will always assume that there always hold the conditions of the following theorem, providing sufficient conditions for a unique solution to exist:

Theorem 1 (Existence and uniqueness) *If the function $f : [t_0, T]\times\mathbb{R} :\to \mathbb{R}$ is continuous and satisfies the Lipschitz condition with respect to the second argument:*

$$\exists\, L \in \mathbb{R}^+ : |f(t,y_2) - f(t,y_1)| \leq L\,|y_2 - y_1| \quad \forall\, t \in [t_0,T] \quad \forall\, y_1, y_2 \in \mathbb{R}$$

then, for any initial condition $y_0 \in \mathbb{R}$, there exists a unique solution to the IVP (1.1) in the time interval $[t_0, T]$.

The hypothesis of this theorem is enough to construct a consistent theory of numerical methods. Indeed it guarantees that all of the problems at hand are well-posed in the Hadamard sense:

Definition 2 (well-posedness) *The IVP (1.1) is said to be well-posed in the Hadamard sense if*

1. *a solution exists,*
2. *the solution is unique,*
3. *the solution's behavior changes continuously with the initial conditions.*

The following theorem gives an insight of the ability of a slightly modified differential system not to let the solution go "too far" from the solution to the original problem.

Theorem 2 (Continuous dependence on initial data) *Let*

$$\begin{cases} y'(t) = f(t, y(t)) \\ y(t_0) = y_0 \end{cases} \quad \text{and} \quad \begin{cases} \tilde{y}'(t) = \tilde{f}(t, \tilde{y}(t)) \\ \tilde{y}(t_0) = \tilde{y}_0 \end{cases} \tag{1.2}$$

be two IVPs with $f, \tilde{f} \in \mathcal{C}^0([t_0, \infty) \times \mathbb{R}) \to D \subseteq \mathbb{R}$*, and let* f *be a Lipschitz function wrt its second argument and Lipschitz constant* L*. If there exists a real value* $\varepsilon > 0$ *such that*

$$|f(t, w) - \tilde{f}(t, w)| \leq \varepsilon \qquad t \geq t_0, \forall w \in D$$

then

$$|y(t) - \tilde{y}(t)| \leq |y_0(t) - \tilde{y}_0(t)|\, e^{L(t-t_0)} + \frac{\varepsilon}{L}\left(e^{L(t-t_0)} - 1\right) \quad \forall t \geq t_0$$

The proof to this theorem is not particularly difficult and can be found in [Jac09] More importantly, the meaning of this theorem is that if the two differential systems are close enough both in the fields and initial conditions, at any time there's an upper limit the respective solutions can be far apart from each other. Notice that only f is required to have the Lipschitz property.

1.2 The discrete problem

Continuum models must be discretized in order to obtain a numerical solution using a computer. Despite the assumption that the continuous problem is well-posed and its solutions have a continuous dependence on the initial conditions, the numerical resolution may suffer from numerical instability due to the method involving a representation with finite precision. In other words, a small error in the initial data, which always exists, may result in much larger errors in the solutions, depending on the way the numerical scheme handles it. In this section, the basic facts related to equation discretization, machine stability in computer representation and problems related to errors in the calculation of numerical solutions are presented.

1.2.1 Time-domain discretization and solution approximation

It is rather obvious that a continuous problem, such as (1.1), on a computer can only be solved at a finite number of times, and the solution can be provided only to a finite precision, due to the limited representation of real numbers on a machine. By *discretization* of the time domain we mean the process of sampling a finite number of points in the interval $[t_0, T]$; since we will not deal with the so-called *adaptive methods*, where the time values are not equally spaced, we will always assume points are separated with a fixed time step h and thus are written:

$$t_n = t_0 + nh \qquad n = 0, 1, 2, \ldots N \tag{1.3}$$

such relation identifies $N + 1$ points in the interval $[t_0, T]$. The integer N is related to the step h through the relation:

$$N = \frac{T - t_0}{h} \tag{1.4}$$

the set of sampled points in the domain are called the *grid* and indicated by the symbol

$$\mathcal{I}_h = \{t_0, t_1, \ldots, t_N\} \tag{1.5}$$

The numerical solution to the IVP on the grid is an ensemble of $N+1$ values $y_0, y_1, \ldots, y_N$ obtained through an appropriate procedure, representative of the true solution at the grid points $\mathcal{I}_h$:

$$y_n \approx y(t_n) \tag{1.6}$$

Mathematically speaking, a numerical method for an IVP is a *difference equation*, i.e. an equation of the form:

$$F(y_n, y_{n+1}, \ldots, y_{n+k}) = 0 \qquad \forall\, n \geq n_0 \in \mathbb{N}_0 \tag{1.7}$$

such a relation involves $k+1$ unknown values to determine all of the (infinite) values indexed from a given integer n_0 on. The integer k is called the stepnumber of the method; if $k = 1$, the method is called a one-step method, while if $k > 1$, the method is called a multistep method. It is understood that, with direct reference to the numerical approximation to the IVP, a finite number of the difference equation unknowns is representative of the approximated values of the solution at all the grid points. For this work, we will only consider one-step methods.

1.2.2 One-step methods

We turn now our attention to how to obtain a difference equation in the process of numerically integrating an IVP. In first instance, one can construct a straightforward approximate method by Taylor expanding the solution. Since the initial value $y_0 = y(t_0)$ is known, this allows to relate it to the unknown value at the nearest grid point $t_1 = t_0 + h$:

$$y(t_0 + h) = y(t_0) + y'(t_0)\, h + O(h^2) \tag{1.8}$$

and substituting the differential equation:

$$y(t_0 + h) = y(t_0) + f(t_0, y_0)\, h + O(h^2) \tag{1.9}$$

If we neglect the $O(h^2)$ term in this equation, the relation between the exact values doesn't hold anymore, but we can think and use it as an exact relation between approximate values of the solution:

$$y_1 = y_0 + f(t_0, y_0)\, h \tag{1.10}$$

where $y_0 \approx y(t_0)$ and $y_1 \approx y(t_1)$, and this relation will be exact except for a $O(h^2)$ term, this being the difference between the exact and approximated equation. This argument holds more in general: if the value y_n at t_n is known, it can be shifted forward to obtain the next one:

$$y_{n+1} = y_n + f(t_n, y_n)\, h \tag{1.11}$$

This method is known as the *forward Euler method*: it is a difference equation with $k = 1$, i.e., a one-step method. Depending on the nature of the function f, it is in general a non-linear method, allowing to solve for y_{n+1} once y_n is known.

Euler method (1.11) is just a particular case of an explicit one-step methods, cast under the general expression:

$$y_{n+1} = y_n + h\,\varphi(t_n, y_n; h) \tag{1.12}$$

and the quantity φ is representative of the approximation to the increment ratio $(y_{n+1} - y_n)/h$ through the chosen numerical method.

1.2.3 Consistency, stability and convergence

From the general form of one-step methods (1.12), one can write:

$$\frac{y_{n+1} - y_n}{h} - \varphi(t_n, y_n; h) = 0 \tag{1.13}$$

The *local truncation error* is defined by putting in (1.13) the exact solution $y(t_n)$ in place of the numerical solution y_n:

$$\mathcal{T}(t_n, y(t_n); h) := \frac{y(t_{n+1}) - y(t_n)}{h} - \varphi(t_n, y(t_n); h) \tag{1.14}$$

This quantity will now be, in general, different from zero. The local truncation error thus represents the difference between the exact increment of the solution and the approximate increment provided by the method through φ. It is a point-dependent quantity, evaluated at each time t_n. Basing on this, we can provide the following definition, properly posing the idea that the discrete method, through the increment function φ, must be a sufficiently accurate representation of the differential system:

Definition 3 (Consistency) *A one-step method is said to be consistent if for the local truncation error holds the limit*

$$\lim_{h \to 0} \mathcal{T}(t, y; h) = 0 \quad \forall\,(t, y) \in [t_0, T] \times \mathbb{R} \tag{1.15}$$

The local truncation error is a measure of how much the numerical differential operator φ, which takes the place of the increment ratio, diverges from the exact one. Therefore, the consistency means that the numerical method must converge toward the exact differentiation as h becomes small.

Another important definition related to one-step methods is:

Definition 4 (Order) *A one-step method is said to be* of order p *if*

$$\mathcal{T}(t, y; h) = O(h^p) \qquad \forall (t, y) \in [t_0, T] \times \mathbb{R} \tag{1.16}$$

It is worth noting that *if a method is consistent, then it is at least of order* $p = 1$ (and conversely, a method of order $p = 1$ is certainly consistent).

Next, we turn to the fundamental question of how reliable may be a representation of an exact problem on a computer and the solution it provides through the application of a numerical method. It must be clear that, due to the truncation error in the machine representation of real numbers, a computer can never represent the actual initial condition and the method (i.e., the difference equation) under consideration. Therefore, a stability criterion describing how much a numerical solution departs from an exact one, must be properly characterized. To proceed in this direction, we first define the *numerical residual operator* associated to the one-step method (1.12) as the function

$$R_h(v_n) := \frac{v_{n+1} - v_n}{h} - \varphi(t_n, v_n; h) \tag{1.17}$$

where v_n is any net function $v_n : \mathcal{I}_h \to \mathbb{R}$. When $v_n = y_n$ (that is the solution obtained from the method itself), it is obviously $R_h(v_n) = 0$. Consider then a different net function v_n such that $v_0 = y_0 + \delta$ and denote $\varepsilon_n := R(v_n) \quad \forall n$, which is then certainly different from zero. We give the following important definition, which can be thought as the numerical counterpart of Theorem 2:

Definition 5 (Zero-stability) *A one-step method (1.12) is called zero-stable if there exists a constant* K*, independent of* h*, such that the following stability inequality holds:*

$$\max_{0 \le n \le N} |y_n - v_n| \le K \left(\delta + \max_{0 \le n \le N} |\varepsilon_n| \right) \tag{1.18}$$

$\forall\, h$ *with* $|h|$ *sufficiently small.*

Zero-stability is only a small stepsize-related property: the last specification is the very reason why the method is called *zero*-stable. Due to this vagueness, more robust stability definitions have been proposed, though they will not be discussed in this work. Zero-stability is a property of the method and is not related to its approximating power. Rather, it characterizes the robustness of the scheme to small perturbations, and is of extreme importance for computational purposes. We may think ε as the difference between the exact (say in "infinite precision") solution y_n to the method and the solution v_n obtained through the same method starting from the approximate initial value $v_0 = y_0 + \delta$:

$$v_{n+1} = v_n + h\, \varphi(t_n, v_n; h) \qquad n = 0, \ldots, N \tag{1.19}$$

The zero-stability property guarantees the solution obtained through an approximation (perturbation) of both initial condition and numerical method (e.g., calculated with a limited representation in floating-point arithmetic) goes not too far from the exact representation. The following theorem provides a sufficient condition for the zero-stability of a numerical method to be guaranteed:

Theorem 3 (Zero-stability theorem) *If $\varphi(t, y; h)$ satisfies a Lipschitz condition respect to y*

$$|\varphi(t, y_2; h) - \varphi(t, y_1; h)| \leq L|y_2 - y_1| t \in [t_0, T] \quad y_1, y_2 \in D \quad h \in [0, h_0] \quad (1.20)$$

then the method is zero stable for $h \leq h_0$.

Finally, we provide the fundamental definition of *convergence of a numerical method*, which translates the idea that a numerical solution $[y_0, y_1, \ldots, y_N]$ must approach the exact solution to the problem $y(t)$ at the grid points as the timestep h is reduced further and further. This concept can be formally expressed as a limit statement as follows:

Definition 6 (Convergence) *A one-step method is said to be convergent if for the difference between the analytic solution evaluated at the net points and the numerical solution holds the limit:*

$$\lim_{h \to 0} \max_{0 \leq n \leq N} |y(t_n) - y_n| = 0 \quad (1.21)$$

In the vast majority of practical cases, the previous properties cannot be verified through direct application of their definitions, since the knowledge of the exact analytic solution of the IVP would be required. The following, fundamental theorem provides the deep connection between all of the discussed properties.

Theorem 4 (Dahlquist Equivalence Theorem) *A one-step method is convergent if and only if it is consistent and zero-stable.*

1.3 Runge-Kutta methods for ODEs

Among all the possible schemes, throughout this work we will adopt and construct time advancement of our codes with the specific class of *Runge-Kutta methods*. A way of introducing them is by considering the following integral expression of the IVP (1.1):

$$y(t_{n+1}) = y(t_n) + h \int_{t_n}^{t_{n+1}} f(s, y(s)) ds \quad (1.22)$$

After selecting s real numbers c_i with the following property

$$0 \leq c_1 \leq \cdots \leq c_s \leq 1$$

a chosen quadrature formula, employing $t_n + c_i h$ as knot points and weights b_i, is then applied to the integral (1.22)

$$\int_{t_n}^{t_{n+1}} f(s, y(s)) ds \simeq h \sum_{i=1}^{s} b_i f(t_n + c_i h, y(t_n + c_i h)) \quad (1.23)$$

indicating with $Y_i \approx y(t_n + c_i h)$ the approximate values of the solution on the subgrid points, the exact relation between approximate values on the grid is

$$y_{n+1} = y_n + \sum_{i=1}^{s} b_i f(t_n + c_i h, Y_i) \tag{1.24}$$

it is clear that an independent relation to evaluate the quantities $Y_i \quad i = 1, 2 \ldots, s$ is now needed. This relation can be introduced through Taylor expansions, or by further application of an integral quadrature approximation with weights a_{ij} over the same knot points, as follows:

$$\begin{aligned} y(t_n + c_i h) &= y(t_n) + \int_{t_n}^{t_n + c_i h} f(s, y(s)) ds \\ &\simeq y(t_n) + h \sum_{j=1}^{s} a_{ij} f(t_n + c_j h, y(t_n + c_j h)) \end{aligned} \tag{1.25}$$

which read as an exact relation on the grid is finally

$$Y_i = y_n + h \sum_{j=1}^{s} a_{ij} f(t_n + c_j h, Y_j) \tag{1.26}$$

Equations (1.24) and (1.26) define a Runge-Kutta time advancement scheme for the first order IVP (1.1). Because of the form of equation (1.24), Runge-Kutta methods are one-step methods, but since they require several evaluations of the function at fractions of the time interval, they are said to be *multistage methods*.

Example (Modified Euler): We consider approximating the integral (1.23) by the well-known *midpoint rule*, meaning $s = 2$, $b_1 = 0$, $b_2 = 1$, $c_1 = 0$ and $c_2 = 1/2$:

$$\int_{t_n}^{t_{n+1}} f\left(s, y(s)\right) ds \simeq h f\left(t_n + \frac{h}{2}, y\left(t_n + \frac{h}{2}\right)\right) \tag{1.27}$$

Since $c_1 = 0$, we can choose $Y_1 = y_n$ and the value of the solution at the midpoint is evaluated through the rectangle rule approximation, meaning $a_{21} = 1/2$, $a_{22} = 0$ in equation (1.25) with $i = 2$, obtaining:

$$y(t + h/2) = \int_{t_n}^{t_n + h/2} f\left(s, y(s)\right) ds \simeq y(t_n) + \frac{h}{2} f(t_n, y(t_n)) \tag{1.28}$$

Therefore the R-K steps (1.24) and (1.26) for this case are written

$$\begin{aligned} y_{n+1} &= y_n + h\, f\left(t_n + \frac{h}{2}, Y_2\right) \\ Y_1 &= y_n \\ Y_2 &= y_n + \frac{h}{2} f(t_n, Y_1) \end{aligned} \tag{1.29}$$

since evaluations of Y_1 and Y_2 are explicit, this method can be written in one equation:

$$y_{n+1} = y_n + h\, f\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}\, f(t_n, y_n)\right) \tag{1.30}$$

This is commonly known as the *modified Euler* method, a one-step two-stage explicit method.

Since any R-K method requires an $s \times s$ matrix and two s-dimensional vectors, it is usually given in terms of a *Butcher's tableau* or *Runge-Kutta matrix*, as shown in Table 1.1. The Butcher's tableau for the example presented above is reported in Table 1.2.

c_1	a_{11}	a_{12}	$\dots$	a_{1s}
c_2	a_{21}	a_{22}	$\dots$	a_{2s}
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
c_s	a_{s1}	a_{s2}	$\dots$	a_{ss}
	b_1	b_2	$\dots$	b_s

Table 1.1: General expression for a Butcher tableau

0	0	0
1/2	1/2	0
	1/6	1/3

Table 1.2: Butcher tableau for the modified Euler method.

By considering the structure of the matrix a_{ij}, three main cases can be observed:

1. $a_{ij} = 0 \quad \forall j \geq i$. (Examples 1. and 2. are particular cases of this condition). The system of equations is in the form

$$\begin{aligned}
Y_1 &= y_n \\
Y_2 &= y_n + a_{21}\, f(t_n + c_1 h, Y_1) \\
Y_3 &= y_n + a_{31}\, f(t_n + c_1 h, Y_1) + a_{32}\, f(t_n + c_2 h, Y_2) \\
&\dots \\
Y_s &= y_n + a_{s1}\, f(t_n + c_1 h, Y_1) + a_{s2}\, f(t_n + c_2 h, Y_2) + \dots \\
&\qquad + a_{s(s-1)}\, f(t_n + c_{s-1} h, Y_{s-1})
\end{aligned}$$

   in this case each equation allows methodically to explicitly solve for the quantity Y_i, which will then be needed to solve the next ones.

2. $a_{ij} = 0 \quad \forall j < i$. Presence of diagonal terms cause the unknown i-th

quantity to figure at both members:

$$
\begin{aligned}
Y_1 &= y_n + a_{11}\, f(t_n + c_1 h, Y_1) \\
Y_2 &= y_n + a_{21}\, f(t_n + c_1 h, Y_1) + a_{22}\, f(t_n + c_2 h, Y_2) \\
Y_3 &= y_n + a_{31}\, f(t_n + c_1 h, Y_1) + a_{32}\, f(t_n + c_1 h, Y_2) + a_{33}\, f(t_n + c_3 h, Y_3) \\
&\dots \\
Y_s &= y_n + a_{s1}\, f(t_n + c_1 h, Y_1) + a_{s2}\, f(t_n + c_2 h, Y_2) + \dots \\
&\qquad + a_{s(s-1)}\, f(t_n + c_{s-1} h, Y_{s-1}) + a_{ss}\, f(t_n + c_s h, Y_s)
\end{aligned}
$$

3. The matrix a_{ij} is complete. In this case each equation of the stage law (1.26) is a relation between Y_i and all of the quantities Y_j. Depending on f, this will be in general a system of implicit non linear algebraic equations.

1.3.1 Convergence, consistency and stability of R-K methods

Since the increment function $\varphi(t, y; h)$ for a Runge-Kutta method is a linear combination of evaluations of the field f, which is assumed to be a Lipschitz function, $\varphi(t, y; h)$ is also a Lipschitz function with respect to y. From the previous theorems, we can conclude that *all Runge-Kutta methods are zero-stable*, implying that *consistency and convergence for a Runge-Kutta method are equivalent properties*.

Any Runge-Kutta method is completely (and uniquely) defined through its Butcher tableau: for this reason, it makes sense to study whether relations between the coefficients c_i, a_{ij}, b_i and the properties of the method exist. First we introduce the following *stage consistency condition* by considering the particular IVP:

$$
\begin{cases}
y'(t) = 1 \\
y(t_0) = y_0
\end{cases}
\tag{1.31}
$$

which has the obvious solution

$$
y(t) = t - t_0 \tag{1.32}
$$

For a general R-K, the stages law (1.26) in this case writes

$$
Y_i = y_n + h \sum_{j=1}^{s} a_{ij} \quad \forall\, i = 1, \dots, N \tag{1.33}
$$

and by setting the requirement for the approximate solutions Y_i at the stages to match the exact solution $y(t_i)$:

$$
\begin{aligned}
y(t_i) = t_n + c_i\, h - t_0 = y_n + h \sum_{j=1}^{s} a_{ij} \\
\implies c_i = \sum_{j=1}^{s} a_{ij} \quad \forall\, i = 1, \dots, N
\end{aligned}
\tag{1.34}
$$

The last equation is the *stage consistency condition for the linear problem*, also known as the *row-sum condition* of the Runge-Kutta matrix. If the row-sum condition is satisfied, $Y_i = y(t_i)$ is exact only for the linear problem, and in general it will not hold for any other IVP. Despite, this is a useful property in some cases (e.g. problems linearization). It is not required for all Runge-Kutta methods to satisfy this condition though.

John C. Butcher (1933) developed an elegant theory relating fundamental properties of R-K methods to precise algebraic conditions on the coefficients. Dealing with details of Butcher's theory would require a dedicated chapter and specific mathematics (e.g. graph theory for derivatives representation) which are not the purpose of this work. We limit ourselves here to present the following important result:

Theorem 5 (Butcher) *A Runge-Kutta method is convergent if and only if*

$$\sum_{i=1}^{s} b_i = 1 \tag{1.35}$$

In spite of the augmented difficulty and computational requirements needed when dealing with implicit methods, such methods are largely employed because of their wide class of stability- and order-related properties, making them useful in many situations. The following theorem provides an example of the limits related to the use of explicit R-K methods:

Theorem 6 (Butcher barrier for explicit R-K methods) *The maximum order p for an s-stages explicit Runge-Kutta method is s if $s \leq 4$. For $s > 5$ there doesn't exist any R-K method of order $p = s$.*

1.4 Numerical methods for PDEs

1.4.1 Introduction and notations

Partial Differential Equations (PDEs) arise in many fields of Physics where the problem at hand involves several independent variables; examples are: Continuum Mechanics, Electrodynamics, Quantum Mechanics and General Relativity. When dealing with PDEs and system of PDEs, it is clear that, even more than for the ODEs case, numerical methods are strictly required to provide solutions.

This section deals with the techniques employed to construct finite difference schemes for PDEs. Our purpose is to provide a method for constructing numerical algorithms or PDEs, relying on numerical methods for ODEs discussed in the previous sections. Due to the vastity of the subject, and in view of the specific applications for this work, we will limit our attention to first-order PDEs, starting from the scalar case. A *Partial Differential Equation* is a relation between a function $u(x^i)$ depending on $n > 1$ variables $x^1, x^2, \ldots, x^n$

and its partial derivatives, that in the most general form has the expression

$$F\left(x^1,\dots,x^n,u,\frac{\partial u}{\partial x^1},\dots,\frac{\partial u}{\partial x^n}\right)=0 \tag{1.36}$$

Since numerical simulation of two-dimensional problems on rectangular grids will be specifically considered later in this work, from now on we will take t,x,y as the independent variables, with the usual meaning of time and rectangular coordinates in space. Extensions of the concepts to higher dimension are immediate and can be found in many textbooks (see e.g. [RM14]).

1.4.2 Classification of first-order PDEs systems

Several definitions are required to proceed:

Definition 7 (Quasi-linear PDE) *A first-order PDE of the form*

$$A(t,x,y,u)\frac{\partial u}{\partial t}+B(t,x,y,u)\frac{\partial u}{\partial x}+C(t,x,y,u)\frac{\partial u}{\partial y}+D(t,x,y,u)=0 \tag{1.37}$$

is called a quasi-linear PDE.

If the derivative-terms coefficients do not explicitly depend on the solution u, we have the following

Definition 8 (Linear PDE) *A first-order PDE of the form*

$$A(t,x,y)\frac{\partial u}{\partial t}+B(t,x,y)\frac{\partial u}{\partial x}+C(t,x,y)\frac{\partial u}{\partial y}+D(t,x,y)=0 \tag{1.38}$$

is called a linear PDE.

Definition 9 (First-order quasi-linear PDE system) *A first-order quasi-linear system of PDEs is a set of quasi-linear PDEs in the unknowns $u_1,u_2,\dots,u_m$ having the form:*

$$\begin{cases} A_1^1\dfrac{\partial u^1}{\partial t}+\cdots+A_m^1\dfrac{\partial u^m}{\partial t}+F_1^1\dfrac{\partial u^1}{\partial x}+\cdots+F_m^1\dfrac{\partial u^m}{\partial x}+ \\ \qquad\qquad +G_1^1\dfrac{\partial u^1}{\partial y}+\cdots+G_m^1\dfrac{\partial u^m}{\partial y}+D^1=0 \\ A_1^2\dfrac{\partial u^1}{\partial t}+\cdots+A_m^2\dfrac{\partial u^m}{\partial t}+F_1^2\dfrac{\partial u^1}{\partial x}+\cdots+F_m^2\dfrac{\partial u^m}{\partial x}+ \\ \qquad\qquad +G_1^2\dfrac{\partial u^1}{\partial y}+\cdots+G_m^2\dfrac{\partial u^m}{\partial y}+D^2=0 \\ \vdots \\ A_1^m\dfrac{\partial u^1}{\partial t}+\cdots+A_m^m\dfrac{\partial u^m}{\partial t}+F_1^m\dfrac{\partial u^1}{\partial x}+\cdots+F_m^m\dfrac{\partial u^m}{\partial x}+ \\ \qquad\qquad +G_1^m\dfrac{\partial u^1}{\partial y}+\cdots+G_m^m\dfrac{\partial u^m}{\partial y}+D^m=0 \end{cases} \tag{1.39}$$

which in short notation is

$$A_j^i \frac{\partial u^j}{\partial t} + F_j^i \frac{\partial u^j}{\partial x} + G_j^i \frac{\partial u^j}{\partial y} + D^i = 0 \qquad i = 1, \dots, m \tag{1.40}$$

where Einstein summation convention is employed. We will also make use of the matrix form:

$$A\frac{\partial U}{\partial t} + F\frac{\partial U}{\partial x} + G\frac{\partial U}{\partial y} + D = 0 \tag{1.41}$$

where

$$U = \begin{bmatrix} u^1 \\ u^2 \\ \vdots \\ u^m \end{bmatrix} \qquad D = \begin{bmatrix} D^1 \\ D^2 \\ \vdots \\ D^m \end{bmatrix} \tag{1.42}$$

and A, F and G are the matrices related to the same symbols as in (1.40).

In fluid dynamics application we always deal with a particular case of the system above, normalized form where for each equation only one time derivative of the solution is present, while space derivatives are all still coupled; this case reads

$$\frac{\partial u^i}{\partial t} + F_j^i \frac{\partial u^j}{\partial x} + G_k^i \frac{\partial u^k}{\partial x} + D^i = 0 \qquad i = 1, \dots, m \tag{1.43}$$

We restrict now our attention to the one-dimensional case to spend some words about the classification of PDEs systems, which gives relevant information about the way the solution evolves, and is a topic related to stability criteria in numerical analysis. By considering the system

$$\frac{\partial u^i}{\partial t} + F_j^i \frac{\partial u^j}{\partial x} + D^i = 0 \qquad i = 1, \dots, m \tag{1.44}$$

we provide the following definition:

Definition 10 (Hyperbolic System) *A system of PDEs (1.44) is said to be hyperbolic at a point (t, x) if the matrix F has m real eigenvalues $\lambda_1, \dots, \lambda_m$. The system is said to be strictly hyperbolic if the eigenvalues λ_i are all distinct.*

It is important to stress that, since in the quasi-linear case the elements of the matrix F depend on the position through the solution itself, the hyperbolic character of a PDEs system is not only related to the differential system alone, but may depend on the solution itself through the choice of initial conditions. Hyperbolic systems are particularly meaningful from a physical point of view, since a precise meaning can be associated to the system eigenvalues. This procedure is explained in section 2.2, with particular reference to numerical applications. Several situations arise when the system eigenvalues present specific characteristics, and therefore some generalizations of the definition above are required for higher dimensions. A complete and widely more general discussion about the subject of classification of PDEs systems can be found in [RM14].

1.4.3 Space and time discretization

Same prescriptions as in section 1.2.1 hold for the discretization of the space and time domain. For the continuous problem, we assume the field variables to vary in the domain

$$t \in [0, T] \qquad x \in [0, L_x] \qquad y \in [0, L_y] \tag{1.45}$$

If the integers n_t, n_x, n_y are chosen as the number of points in each of these intervals respectively, the discretization writes:

$$\begin{aligned} t_n &= n\Delta t \qquad n = 1, \ldots, n_t \\ x_i &= i\Delta x \qquad i = 1, \ldots, n_x \\ y_j &= j\Delta y \qquad j = 1, \ldots, n_y \end{aligned} \tag{1.46}$$

where

$$\begin{aligned} \Delta t &= \frac{T}{n_t - 1} \\ \Delta x &= \frac{L_x}{n_x - 1} \\ \Delta y &= \frac{L_y}{n_y - 1} \end{aligned} \tag{1.47}$$

When boundary conditions for the problem require a periodic spatial direction, for example

$$u(t, x + L, y) = u(t, x, y) \tag{1.48}$$

the last grid point coincides with the first one, and the relation between stepsize, interval length and number of points becomes

$$\Delta x = \frac{L_x}{n_x} \tag{1.49}$$

Employing the same symbols as in Chapter 1, the numerical approximation of the solution on the grid through an approximating scheme will then be written

$$u(t, x, y) \approx u_{ij}^n \tag{1.50}$$

1.4.4 Approximation of first derivatives

Now we want to focus our attention on to how to approximate partial derivatives of a function on a discrete domain. All of the methods we are going to use in this work are based on Taylor expansion of functions. The first, most straightforward approximation of a derivative, is obtained by considering the first-order expansion of the function $u(t, x, y)$ with respect to the variable x (all of the other variables will be subtended in the following expressions)

$$u(x + \Delta x) = u(x) + \frac{\partial u}{\partial x}(x)\Delta x + O(\Delta x^2) \tag{1.51}$$

from which

$$\frac{\partial u}{\partial x}(x) = \frac{u(x+\Delta x) - u(x)}{\Delta x} + O(\Delta x) \tag{1.52}$$

By neglecting the $O(\Delta x)$ term, this expression provides an approximation to the first derivative. As usual, an approximate relation between exact values is to be read on the grid as an exact relation between approximate values

$$\left(\frac{\partial u}{\partial x}\right)_i = \frac{u_{i+1} - u_i}{\Delta x} \tag{1.53}$$

where

$$\left(\frac{\partial u}{\partial x}\right)_i \approx \frac{\partial u}{\partial x}(x_i) \tag{1.54}$$

Such approximation to the first derivative is nothing more than a mere substitution of the derivative at a point with the function increment ratio. A more accurate representation is obtained by considering two second-order Taylor expansions about the point x:

$$\begin{aligned} u(x+\Delta x) &= u(x) + \frac{\partial u}{\partial x}(x)\Delta x + \frac{1}{2}\frac{\partial^2 u}{\partial x^2}(x)\Delta x^2 + O(\Delta x^3) \\ u(x-\Delta x) &= u(x) - \frac{\partial u}{\partial x}(x)\Delta x + \frac{1}{2}\frac{\partial^2 u}{\partial x^2}(x)\Delta x^2 + O(\Delta x^3) \end{aligned} \tag{1.55}$$

and subtracting one from the other:

$$\frac{\partial u}{\partial x}(x) = \frac{u(x+\Delta x) - u(x-\Delta x)}{\Delta x} + O(\Delta x^2) \tag{1.56}$$

Therefore, the approximating scheme is

$$\left(\frac{\partial u}{\partial x}\right)_i = \frac{u_{i+1} - u_{i-1}}{2\,\Delta x} \tag{1.57}$$

Notice that even though second-derivative terms have canceled out, expansion to second order in (1.55) has been necessary to explicitly realize that the neglected error term is now an $O(\Delta x^2)$. This scheme is usually referred to as the *central difference approximation.*

Since all the methods for the approximation of first derivatives involve a linear combination of the function values at some points and a division by the step size, to indicate them we will make use of the general notation

$$\frac{\partial u}{\partial x} \approx \frac{\Delta_x[u]}{\Delta x} \tag{1.58}$$

where $\Delta_x[\cdot]$ indicates the operator taking the function $u(x)$ and evaluating a linear combination of its value at discrete points along the x direction. We will

make use of the same symbol applied to u_i to indicate the same linear operation performed on the values of the numerical solution on the grid.

A scheme for derivatives approximation is said to be of *order* p if it comes from neglecting an $O(\Delta x^p)$ term in the expression obtained for the derivative after Taylor expansions. Equivalently, this means that the local truncation error:

$$T(x;\Delta x) = \frac{\partial u}{\partial x} - \frac{\Delta[u]}{\Delta x} = O(\Delta x^p) \tag{1.59}$$

We consider now a more general approach for constructing difference schemes which, at the same time, allows for more direct control of the truncation scheme order. The method mainly relies on the paper [Lel92]. Again, we'll focus on a single differentiation variable omitting the others for writing clarity. The basic assumption is that a linear combination of the first derivatives of a function around a point can be related to a linear combination of the function-central differences around the same point:

$$\sum_{k=1}^{l-1} \alpha_k \left(u'(x_{i-k}) + u'(x_{i+k})\right) + u'(x_i) = \frac{1}{\Delta x}\sum_{k=1}^{l} a_k \left(u(x_{i+k}) - u(x_{i-k})\right) \tag{1.60}$$

By writing this, we're assuming that first derivatives at the same distance from the point x_i are equally weighted. Such expression, and the following considerations, are meaningful at internal points of a given domain, when x_i is sufficiently far from the boundary and all of the terms in the summation are defined. Since the exact function $u(x)$ is involved in Equation (1.60), this equation cannot hold exactly in general. The values of the coefficients α_k and a_k must be constrained by matching LHS and RHS to a required order of approximation, based on Taylor series expansion around the point x_i of all the quantities. As an explicit example, consider the case $l = 2$ for which we have:

$$\begin{aligned}\alpha(u'(x_{i+1}) + u'(x_{i-1})) + u'(x_i) = \\ \frac{a_1}{\Delta x}(u(x_{i+1}) - u(x_{i-1})) + \frac{a_2}{\Delta x}(u(x_{i+2}) - u(x_{i-2}))\end{aligned} \tag{1.61}$$

Expanding in Taylor series about the i-th gridpoint up to the second order gives the expression

$$\begin{aligned}&\alpha\left(u'(x_i) + u''(x_i)\Delta x + O(\Delta x^2)\right) + u(x_i) + \alpha\left(u'(x_i) - u''(x_i)\Delta x + O(\Delta x^2)\right) = \\ &\frac{a_1}{\Delta x}\left(u(x_i) + u'(x_i)\,\Delta x - u(x_i) + u'(x_i)\Delta x + O(\Delta x^2)\right) + \\ &\frac{a_2}{\Delta x}\left(u(x_i) + u'(x_i)\,2\Delta x - u(x_i) + u'(x_i)\,2\Delta x + O(\Delta x^2)\right)\end{aligned}$$

Even terms cancel out while odd terms sum, obtaining

$$(2\alpha + 1)u'(x_i) + O(\Delta x^2) = (2a + 4b)u'(x_i) + O(\Delta x^2) \tag{1.62}$$

Neglecting here the $O(\Delta x^2)$ terms and matching the coefficients of $u'(x_i)$ we obtain the following relation:

$$2\alpha + 1 = 2a_1 + 4a_2 \tag{1.63}$$

Therefore, two of the three parameters can be chosen arbitrarily, always conserving the order 2 of the scheme. If the value $\alpha = 0$ is chosen, the scheme is explicit; if also $a_2 = 0$, then $a_1 = 1/2$ and we obtain the central difference scheme (1.57).

Taylor expansions can be extended to higher orders to provide more accurate schemes and introducing new constraints between the coefficients:

$$\begin{aligned} 6\alpha &= a_1 + 4a_2 \qquad \text{(4-th order, one free parameter)} \\ 5\alpha &= a_1 + 32a_2 \qquad \text{(6-th order, no free parameters)} \end{aligned}$$

When the number of equations matches the number of parameters, no more independent equations can be obtained by further Taylor expansions. For $l = 2$ the maximum possible order of approximation is thus the 6-th order. The general rule is that, since $l - 1$ parameters feature to the LHS of equation 1.60 and l parameters at the RHS, the maximum number of independent equations is $2l - 1$, and since for the central differences the order is doubled at each new equation, this implies that the maximum order for this case is $2(2l-1)$. Finally, in terms of the function approximation on the grid, the general scheme will be written:

$$\sum_{k=1}^{l-1} \alpha_k (u'_{i-k} + u'_{i+k}) + u'_i = \frac{1}{\Delta x} \sum_{k=1}^{l} a_k (u_{i+k} - u_{i-k}) \tag{1.64}$$

1.4.5 First-derivative schemes on periodic domain

In this section some typical schemes for numerical differentiation of a function on a periodic domain, based on equation (1.64), are written explicitly as a reference for computer code implementation.

- $l = 1$

  This case implies $a = 1/2$ and realizes an explicit second order central difference scheme. It is represented in matrix form as

$$\begin{bmatrix} u'_1 \\ u'_2 \\ \vdots \\ u'_{n-1} \\ u'_n \end{bmatrix} = \begin{bmatrix} 0 & a_1 & 0 & \dots & 0 & -a_1 \\ -a_1 & 0 & a_1 & \dots & 0 & 0 \\ \vdots & & & & & \\ 0 & 0 & \dots & -a_1 & 0 & a_1 \\ a_1 & 0 & \dots & 0 & -a_1 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{n-1} \\ u_n \end{bmatrix} \tag{1.65}$$

- $l = 2$

  This case realizes the following constraints:

$$\begin{aligned} &2\alpha + 1 = 2a_1 + 4a_2 \quad \text{(2-nd order)} \\ &6\alpha = a_1 + 4a_2 \quad \text{(4-th order)} \\ &5\alpha = a_1 + 32a_2 \quad \text{(6-th order)} \end{aligned} \tag{1.66}$$

  and is represented in matrix form as:

$$\begin{bmatrix} 1 & \alpha_1 & 0 & \dots & 0 & \alpha_1 \\ \alpha_1 & 1 & \alpha_1 & \dots & 0 & 0 \\ \vdots & & & & & \\ 0 & 0 & \dots & \alpha_1 & 1 & \alpha_1 \\ \alpha_1 & 0 & \dots & 0 & \alpha_1 & 1 \end{bmatrix} \begin{bmatrix} u'_1 \\ u'_2 \\ \vdots \\ u'_{n-1} \\ u'_n \end{bmatrix} = \\ \begin{bmatrix} 0 & a_1 & a_2 & 0 & \dots & -a_2 & -a_1 \\ -a_1 & 0 & a_1 & a_2 & \dots & 0 & -a_2 \\ \vdots & & & & & & \\ a_2 & 0 & \dots & -a_2 & -a_1 & 0 & a_1 \\ a_1 & a_2 & 0 & \dots & -a_2 & -a_1 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{n-1} \\ u_n \end{bmatrix} \tag{1.67}$$

  Equations for the derivatives are coupled and the resolution for the derivative vector requires inversion of a tridiagonal matrix.

- $l = 3$

  This case realizes the constraints:

$$\begin{aligned} &2\alpha_1 + 2\alpha_2 + 1 = 2a_1 + 4a_2 + 6a_3 \quad \text{(2-nd order)} \\ &6\alpha_1 + 24\alpha_2 = a_1 + 4a_2 + 54a_3 \qquad \text{(4-th order)} \\ &5\alpha = a_1 + 32a_2 \quad \text{(6-th order)} \end{aligned} \tag{1.68}$$

  and is represented in matrix form as:

$$\begin{bmatrix} 1 & \alpha_1 & \alpha_2 & 0 & \dots & \alpha_2 & \alpha_1 \\ \alpha_1 & 1 & \alpha_1 & \alpha_2 & \dots & 0 & \alpha_2 \\ \vdots & & & & & & \\ \alpha_2 & 0 & \dots & \alpha_2 & \alpha_1 & 1 & \alpha_1 \\ \alpha_1 & \alpha_2 & 0 & \dots & \alpha_2 & \alpha_1 & 1 \end{bmatrix} \begin{bmatrix} u'_1 \\ u'_2 \\ \vdots \\ u'_{n-1} \\ u'_n \end{bmatrix} = \begin{bmatrix} 0 & a_1 & a_2 & a_3 & 0 & \dots & -a_3 & -a_2 & -a_1 \\ -a_1 & 0 & a_1 & a_2 & a_3 & 0 & \dots & -a_3 & -a_2 \\ \vdots & & & & & & & & \\ a_2 & a_3 & 0 & \dots & -a_3 & -a_2 & -a_1 & 0 & a_1 \\ a_1 & a_2 & a_3 & 0 & \dots & -a_3 & -a_2 & -a_1 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{n-1} \\ u_n \end{bmatrix} \tag{1.69}$$

Equations for the derivatives u'_i are coupled and the resolution for the derivative vector requires inversion of a pentadiagonal matrix.

To conclude, a collection of several schemes commonly employed for numerical differentiation at interior points or on periodic domains are reported in the following tables:

- Explicit schemes ($\alpha = 0$)

Type	l	a_1	a_2	a_3	α_1	α_2	Order
2nd order central difference	1	1/2	-	-	-	-	2
4th order central difference	2	2/3	-1/12	-	0	-	4
6th order central difference	3	3/4	-3/20	1/60	0	0	6

Notice that 8th order explicit schemes are missing since they are possible only for $l > 3$.

- Implicit schemes ($\alpha \neq 0$)

Type	l	a_1	a_2	a_3	α_1	α_2	Order
Classical Padé	2	3/4	0	-	1/4	-	4
6th order tridiagonal	3	7/9	1/36	0	1/3	0	6
8th order tridiagonal	3	25/32	1/20	-1/480	3/8	0	8
4 points 8th order pentadiagonal	3	20/27	25/216	0	4/9	1/36	8
10th order pentadiagonal	3	17/24	101/600	1/600	1/2	1/20	10

1.4.6 Differentiation at boundary points

When some boundaries have no periodic conditions, differentiation at points near the boundary of the domain requires a special treatment due to the lack of an equal or a sufficient number of points from both sides. Similarly to equation

(1.60), we start from the following position defining the one-sided four points approximation for a point at the left boundary:

$$u'_1 + \alpha\, u'_2 = \frac{1}{\Delta x}(a_1 u_1 + a_2 u_2 + a_3 u_3 + a_4 u_4) \tag{1.70}$$

Expanding in Taylor series, we obtain the following constraints between the coefficients for each required order

$$\begin{aligned} &2a_1 + 2a_4 = -(3+\alpha), \quad a_2 - 3a_4 = -2 \quad \text{(2nd order)} \\ &6a_1 = -11 - 2\alpha, \quad 2a_2 = 6 - \alpha \quad \text{(3rd order)} \end{aligned} \tag{1.71}$$

It is worth mentioning that generally the global error of a scheme is dominated by the boundary error.

1.4.7 Approximation of second derivatives

This case follows a similar approach to that for the first derivatives. The main scheme is written starting from the assumption:

$$\sum_{k=1}^{l-1} \alpha_k (u''_{i-k} + u''_{i+k}) + u''_i = \frac{1}{\Delta x^2} \sum_{k=1}^{l} a_k (u_{i+k} - 2u_i + u_{i-k}) \tag{1.72}$$

Proceeding with Taylor expansions, relations between the coefficients α_k and a_k are derived by matching Taylor series of various orders. If $l = 3$, schemes up to 10th order are obtained through the following constraints:

$$\begin{aligned} a_1 + 4a_2 + 9a_3 &= 1 + 2\alpha_1 + 2\alpha_2 && \text{(2nd order)} \\ a_1 + 2^4 a_2 + 3^4 a_3 &= \frac{4!}{2!}(\alpha_1 + 2^2\alpha_2) && \text{(4th order)} \\ a_1 + 2^6 a_2 + 3^6 a_3 &= \frac{6!}{4!}(\alpha_1 + 2^4\alpha_2) && \text{(6th order)} \\ a_1 + 2^8 a_2 + 3^8 a_3 &= \frac{8!}{6!}(\alpha_1 + 2^6\alpha_2) && \text{(8th order)} \\ a_1 + 2^{10} a_2 + 3^{10} a_3 &= \frac{10!}{8!}(\alpha_1 + 2^8\alpha_2) && \text{(10th order)} \end{aligned}$$

More details about second derivative schemes, such as special cases and schemes for second derivatives at boundary points, can be found in [Lel92].

1.4.8 Fourier analysis of difference schemes

Even though compact difference schemes allow for vast control of the choice of the coefficients, some differences arise when choosing different sets of coefficients, and Fourier analysis of errors provides the proper instruments to check

such differences. To give an example of how the Fourier analysis is performed, we consider the simple scheme with $l = 1$

$$u'(x_i) = \frac{a}{\Delta x}(u(x_{i+1}) - u(x_{i-1})\,) \tag{1.73}$$

The Fourier series is defines as:

$$u(x) = \sum_{k=-\infty}^{\infty} \hat{u}\, e^{i\frac{2\pi}{L}kx} \tag{1.74}$$

Expressing both members of (1.73) in Fourier series representation, we then obtain

$$\begin{aligned}
\sum_{k=-n/2}^{n/2} \hat{u}'_k\, e^{ikx} &= \frac{a}{\Delta x}\left(\sum_{k=-n/2}^{n/2} \hat{u}_k\, e^{ik(x+\Delta x)} - \sum_{k=-n/2}^{n/2} \hat{u}_k\, e^{ik(x-\Delta x)}\right) \\
&= \frac{a}{\Delta x}\left(\sum_{k=-n/2}^{n/2} \hat{u}_k\, e^{ikx}\,(e^{ik\Delta x} - e^{-ik\Delta x})\right) \\
&= \frac{a}{\Delta x}\left(\sum_{k=-n/2}^{n/2} 2a\, i \sin(k\Delta x)\, \hat{u}_k\, e^{ikx}\right)
\end{aligned} \tag{1.75}$$

meaning that for any k we have:

$$\hat{u}'_k = i\frac{2a\sin(k\Delta x)}{\Delta x}\hat{u}_k \tag{1.76}$$

This is different from the exact relation expected between Fourier coefficients of a function and its derivative, namely:

$$\hat{u}'_k = iku_k \tag{1.77}$$

The latter would hold if (1.73) were an exact relation between the function and its derivatives. Therefore, to give a comparison criterion, after Fourier transforming a central difference scheme, a *modified wavenumber* K, collecting all of the quantities taking the place of k is defined:

$$\hat{u}'_k := iK(k; \Delta x, a, \alpha)u_k \tag{1.78}$$

For a consistent scheme, it must be

$$\lim_{\Delta x \to 0} K = k$$

This condition can be readily verified for the central difference scheme taken as an example, where $a = 1/2$.

Fourier analysis of a scheme is thus related to the analysis of the modified wavenumber. For a given k, the closer is K to k itself, the better we can

expect is the derivative representation of the wave through the scheme; this depends on the choice of the scheme through the parameters and the stepsize Δx. Free parameters of a scheme can thus be chosen to improve the resolution characteristics, and this analysis allows to check which waves are not sufficiently resolved, eventually according to a tolerance criterion.

More generally, for the scheme with $l = 3$ we have:

$$K(k; \Delta x, a, \alpha) = \frac{2}{\Delta x} \cdot \frac{a_1 \sin(k\Delta x) + a_2 \sin(2k\Delta x) + a_3 \sin(3k\Delta x)}{1 + 2\alpha_1 \cos(k\Delta x) + 2\alpha_2 \cos(2k\Delta x)} \tag{1.79}$$

We may notice that K is periodic in k with period $2\pi/\Delta x$, which is representative of the highest k allowed, and that $K(\pi/\Delta x) = 0$. Therefore it is meaningful to employ the scheme to resolve waves with k up to this value, and restrict the analysis to half of the period. This can also be understood by thinking the function as a signal sampled with frequency $1/\Delta x$ for a time interval L: according to the Nyquist-Shannon theorem, the maximum frequency that can be resolved in the sample is half of the sampling frequency, meaning the wave with $k = \pi/\Delta x$. This limit will hold for the function derivatives as well, but the quality of their reconstruction is further reduced through the use of the finite difference derivative scheme. Figure 1.1 shows several plots of the normalized modified wavenumber $K' = K\Delta x$ versus the normalized wavenumber $k' = k\Delta x$, to get rid of the parametric dependency on Δx. Plot details are provided in the Table below. It may be observed, as expected, that the resolving efficiency generally increases over a wider range of wavelengths by employing implicit, higher order schemes, so that different choice of the coefficients gives a different spectral resolution even though the order remains the same. Anyway, some care must be observed since exceptions may occur.

Plot label	Scheme type	l	a_1	a_2	a_3	α_1	α_2	Order
a	2nd order central diff.	1	1/2	-	-	-	-	2
b	4th order central diff.	2	2/3	-1/12	-	0	-	4
c	6th order central diff.	2	3/4	-3/20	-	1/60	-	6
d	Classical Padé	2	3/4	0	-	1/4	-	2
e	6th order tridiagonal	3	7/9	1/36	0	1/3	0	6
f	8th order tridiagonal	3	25/32	1/20	-1/480	3/8	0	8
g	4 points 8th order pentadiagonal	3	20/27	25/216	0	4/9	1/36	8
h	10th order pentadiagonal	3	17/24	101/600	1/600	1/2	1/20	10

Table 1.3: First derivative schemes of use.

1.5 Time advancement of PDEs

In the previous section, we discussed representation of derivatives by means of an algebraic finite different expression. The first step to proceed with is then to substitute all of the partial derivatives with finite difference quotients. As an

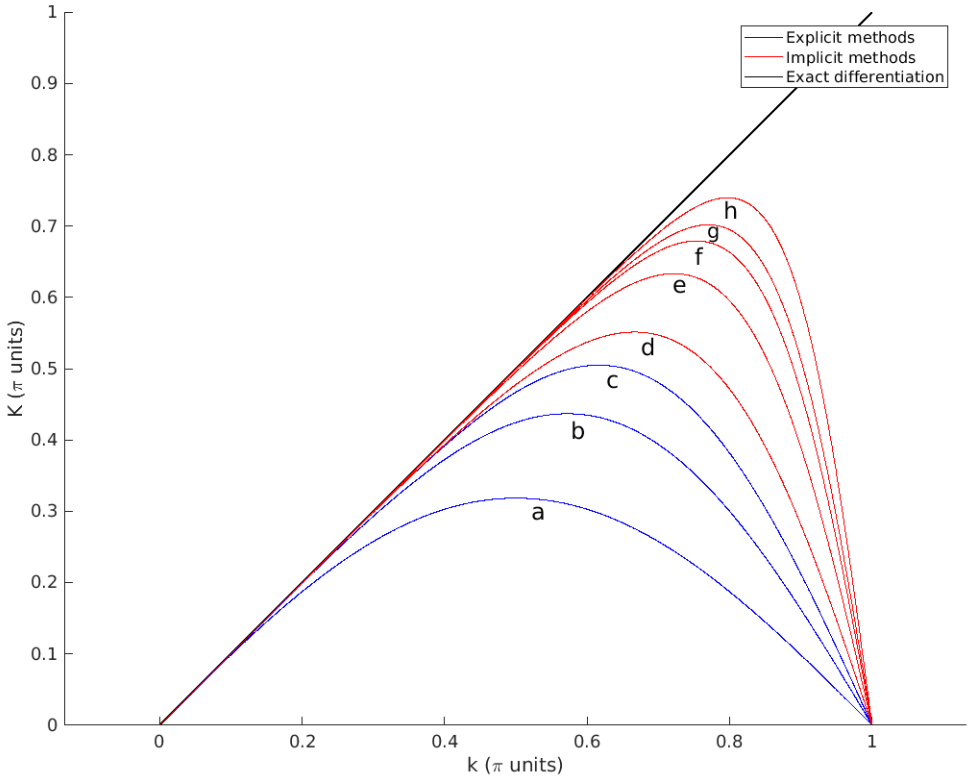


Figure 1.1: Plot of the modified wavenumber K versus the wavenumber k for different schemes. Details are provided in Table 1.4.8.

example, suppose we want to generate a scheme for solving the one-dimensional advection equation

$$\frac{\partial u}{\partial t} + c\,\frac{\partial u}{\partial x} = 0 \tag{1.80}$$

First, we employ some schemes for space and time derivatives,

$$\begin{aligned} \frac{\partial u}{\partial t}(t_n, x_i) &= \frac{\Delta_t[u]}{\Delta t} + O(\Delta t^p) \\ \frac{\partial u}{\partial x}(t_n, x_i) &= \frac{\Delta_x[u]}{\Delta x} + O(\Delta x^m) \end{aligned} \tag{1.81}$$

The PDE is then replaced

$$\frac{\partial u}{\partial t} + c\,\frac{\partial u}{\partial x} = \frac{\Delta_t[u]}{\Delta t} + c\,\frac{\Delta_x[u]}{\Delta x} + O(\Delta t^p, \Delta x^m) = 0 \tag{1.82}$$

And as the local truncation error is neglected at the RHS, the equation is read as a difference equation on the grid, to be solved with algebraic methods.

$$\frac{\Delta_t[u_i^n]}{\Delta t} + c\,\frac{\Delta_x[u_i^n]}{\Delta x} = 0 \tag{1.83}$$

Some consistency, stability and convergence arguments are to be discussed in this new situation. As for the ODE case, given a method through a difference equation, the local truncation error is obtained by substitution of the analytic solution in place of the numerical one:

$$T(t, x, u; \Delta t, \Delta x) := \frac{\Delta_t[u]}{\Delta t} + c\,\frac{\Delta_x[u]}{\Delta x}$$

Definition 11 (Consistency) *A numerical method for a PDE is consistent if*

$$\lim_{\substack{\Delta t \to 0 \\ \Delta x \to 0}} T(t, x, u; \Delta t, \Delta x) = 0 \tag{1.84}$$

It should be clear that any method obtained through the application of discretisation procedures of derivatives is always consistent since, as seen above, the neglected quantity is always a sum of O quantities.

On the other hand, the subject of stability requires a somewhat special treatment, since a stability criterion depends on the form of the difference equation; for this reason, rather than providing more definitions and theorem, we chose to discuss here the more practical Von Neumann stability analysis. To do so, we proceed again with the case study (1.81) specifically employing:

$$\begin{aligned} \Delta_t[u] &= u(x_{i+1}) - u(x_i) \qquad \text{(Forward difference)} \\ \Delta_x[u] &= u(t_{i+1}) - u(x_{i-1}) \quad \text{(Central difference)} \end{aligned} \tag{1.85}$$

The resulting difference equation is

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} + c\,\frac{u_{i+1}^n - u_{i-1}^n}{\Delta x} = 0 \quad , \tag{1.86}$$

and is a consistent numerical method for Equation (1.80) with a local truncation error $T = O(\Delta t, \Delta x^2)$.

In order to discuss stability, we want to check what happens when the exact numerical solution u_i^n, obtained through the numerical method (1.86) starting from a given initial condition, is replaced with a perturbed one:

$$u_i^n \to u_i^n + \varepsilon_i^n \tag{1.87}$$

which contains at each time step and place on the grid, a round off error term ε_i^n due to finite precision representation of the initial condition, and an error coming from finite representation of the difference equation itself at successive times. The difference equation propagating the perturbed numerical solution thus reads:

$$\frac{u_i^{n+1} + \varepsilon_i^{n+1} - u_i^n - \varepsilon_i^n}{\Delta t} + c\,\frac{u_{i+1}^n + \varepsilon_{i+1}^n - u_{i-1}^n - \varepsilon_{i-1}^n}{2\Delta x} = 0 \tag{1.88}$$

and since the numerical solution u_i^n alone exactly satisfies the difference equation,

$$\frac{\varepsilon_i^{n+1} - \varepsilon_i^n}{\Delta t} + c\,\frac{\varepsilon_{i+1}^n - \varepsilon_{i-1}^n}{2\Delta x} = 0 \tag{1.89}$$

i.e. the error ε_i^n also satisfies the difference equation.In particular, we aim to check under what circumstances the error does not grow in time (or at least stays equal):

$$\left|\frac{\varepsilon_i^{n+1}}{\varepsilon_i^n}\right| \leq 1 \tag{1.90}$$

Representation of the error in terms of a discrete Fourier transform in space with an exponential amplitude term is considered at this point. That means representing the grid function ε_i^n as

$$\varepsilon_i^n = \sum_{m=1}^{N/2} A_m(t_n)\, e^{\mathbf{i}k_m x_i} \tag{1.91}$$

where $\mathbf{i}$ is the imaginary unit (not to be confused with the grid index i). substituting in Equation (1.89), after some algebra (see [And95]), we get that

$$\left|\frac{\varepsilon_i^{n+1}}{\varepsilon_i^n}\right| = |A_m(t_n)| = \cos(k_m\Delta x) - \mathbf{i}\,c\frac{\Delta t}{\Delta x}\sin(k_m\Delta x) \qquad \forall m \tag{1.92}$$

and the stability requirement is

$$\left|\frac{\varepsilon_i^{n+1}}{\varepsilon_i^n}\right| = |A_m(t_n)| \leq 1 \implies c\frac{\Delta t}{\Delta x} \leq 1 \tag{1.93}$$

An inequality like this, relating time and space step sizes needed to guarantee stability of a numerical method is called a *Courant-Friedrichs-Lewy condition* (CFL for short hereafter).

As previously anticipated, throughout this work we will consider physical situations described by PDEs systems of the form (1.39). Explicit presence uncoupled time derivative of the solution allows to adapt R-K methods for ODEs in a rather straightforward way in order to generate time advancement schemes for PDEs. For the schemes presented in this work, we proceed in the following manner: first, we construct a hybrid equation where all of the terms involving space derivatives are numerically expressed through a spatial derivative scheme

$$\frac{\partial \mathbf{u}}{\partial t}(t_n) + \mathbf{F}\frac{\Delta_x[\mathbf{u}_{ij}^n]}{\Delta x} + \mathbf{G}\frac{\Delta_y[\mathbf{u}_{ij}^n]}{\Delta y} + \mathbf{D} = 0 \tag{1.94}$$

Then, the term featuring discrete derivative acts as the ODE field f, and the equation is time-advanced through a R-K method.

As an example, let's suppose we want to generate a scheme for solving the one-dimensional advection equation

$$\frac{\partial u}{\partial t} + c\,\frac{\partial u}{\partial x} = 0 \tag{1.95}$$

space derivatives at the RHS are substituted

$$\frac{\partial u}{\partial t} + c\,\frac{\Delta_x[u_i^n]}{\Delta x} = 0 \tag{1.96}$$

the term featuring discrete derivative acts as the ODE field f of Equation, then the time advancement with a R-K method reads

$$\begin{aligned} u_i^{n+1} &= u_i^n + c\,\Delta t \sum_{k=1}^{s} b_k \frac{\Delta_x[u_i^k]}{\Delta t} \\ u_i^k &= u_i^n + \sum_{l=1}^{s} a_{kl} \frac{\Delta_x[u_i^l]}{\Delta t} \end{aligned} \tag{1.97}$$

In complex situations, an explicit stability criterion may be rather difficult or impossible to obtain. In such cases, we will refer to stability conditions of the form:

$$\sigma \frac{\Delta x}{\Delta t} \leq \overline{c} \tag{1.98}$$

where σ is an empirical safety factor, usually ranging from 0 to 1/2, and $\overline{c}$ is the largest characteristic speed of the system solution advancement. Details about this choice can be found in [EFT89] and are employed in the next chapter.

1.6 Filtering techniques

When put in the context of an evolution equation or system, numerical schemes may cause additional effects which affect the evolution of the numerical solution. This section deals with techniques to employ when the numerical resolution of a PDEs system specifically causes the numerical solution to originate numerical, nonphysical oscillations.

By developing a central difference scheme through Taylor series, even derivatives always cancel out: this means that all the neglected quantities contain a leading order term featuring an odd derivative. For this reason, *central schemes are never dissipative schemes*. In turn, this means that the main effect of the neglected terms must be dispersive: the numerical system works as if for the numerical solution, an associated equation with additional odd derivatives were to be solved (see [And95]).

Such numerical effect is clearly manifested when dealing with extreme situations, usually close to the boundaries and when applying the central difference scheme over very sharp function gradients.

The most straightforward way to eliminate oscillations is by taking average of the considered quantity. A very common one is the so called *moving average*, which essentially consists of a low-pass filter giving from the u at each point the averaged quantity U_i defined as

$$U_i = \frac{u_{i-1} + u_{i+1}}{2} \tag{1.99}$$

as can be understood, this averaging method causes an oscillation over a length $2\Delta x$ to be smeared out.

In order to generalize this concept and have a full control over the spectral response of a frequency-filtering technique, we follow ([Lel92]) and employ the following averaging scheme:

$$\begin{aligned} &\beta(U_{i-2} + U_{i+2}) + \alpha(U_{i-1} + U_{i+1}) + U_i = \\ &\quad a\,u_i + \frac{b}{2}(u_{i-1} + u_{i+1}) + \frac{c}{2}(u_{i-2} + u_{i+2}) + \frac{d}{2}(u_{i-3} + u_{i+3}) \end{aligned} \tag{1.100}$$

where U is the filtered grid function and u the original one. Constraints between the coefficients are now found by choosing appropriate conditions on the transfer function associated to (1.100) and its derivatives, so that the filtering can be realized to be sufficiently efficient over a proper range of wavelengths. In our codes we employ the 6th order filtering scheme with the following coefficients at internal points:

$$\begin{aligned} a &= \frac{1}{16}(11 + 10\alpha) \\ b &= \frac{1}{32}(15 + 34\alpha) \\ c &= \frac{1}{16}(-3 + 6\alpha) \\ d &= \frac{1}{32}(1 - 2\alpha), \end{aligned}$$

while at boundary points, filtering is realized with the following 5 point explicit 4th order filtering formulas:

$$\begin{aligned} U_1 &= \frac{15}{16}u_1 + \frac{1}{16}(4u_2 - 6u_3 + 4u_4 - u_5) \\ U_2 &= \frac{3}{4}u_2 + \frac{1}{16}(u_1 - 6u_3 - 4u_4 + u_5) \\ U_3 &= \frac{5}{8}u_3 + \frac{1}{16}(-u_1 + 4u_2 + 4u_4 - u_5) \end{aligned}$$

We refer the reader to [Lel92] for details.

Chapter 2

Computational Fluid Dynamics

The purpose of this chapter is to present the applications of the numerical schemes and procedures to construct numerical solvers based on finite difference schemes for systems of PDEs. Finite difference schemes, as presented in the previous chapter, are employed here, providing an introductory and effective approach to numerical methods for PDEs. Some special attention is given to boundary conditions and to stability control near the edge of the computational domain, which often originates several undesired numerical effects affecting the overall code stability and results. We give reference to fluid dynamic applications of the numerical methods, in view of the study of the drag force over rigid body in a fluid stream to be discussed in chapters 3 and 4.

2.1 Recap about Fluid Dynamics equations

We start by recalling some basic features about continuum mechanics, which can be found in most of the literature about this subject (see e.g. [RM14]). Continuum mechanics deals with the motion of materials modeled as a continuous mass rather than as discrete particles. Basic continuum mechanics equations are obtained by application of mass, momentum and energy conservation principles to the continuous system, and typical additional assumptions about function regularity allow for a differential formulation of the resulting equations. As explained in the previous chapter, all of the fluid quantities are fields, i.e., functions of (t, x, y, z). Conservation of mass gives the following equation

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0 \tag{2.1}$$

where, as usual, ρ denotes the mass density and $\mathbf{v}$ the velocity field. Momentum balance law leads to the equation

$$\rho \left(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} \right) = \nabla \cdot \mathbf{T} + \rho \mathbf{b} \tag{2.2}$$

where $\mathbf{T}$ is a $(2,0)$ symmetric tensor known as the *Cauchy stress tensor* and $\mathbf{b}$ is an external force field per unit mass. By substituting the continuity equation into the momentum balance equation, the so called *conservative formulation* of (2.2) is obtained:

$$\frac{\partial(\rho\mathbf{v})}{\partial t} + \nabla(\rho\,\mathbf{v}\otimes\mathbf{v}) = \nabla\cdot\mathbf{T} + \rho\mathbf{b} \tag{2.3}$$

Both forms will be employed throughout this chapter. An energy balance equation is based on the first law of thermodynamics:

$$\Delta U = Q + W^e \tag{2.4}$$

where Δe is the variation of the system internal energy, Q is the amount of heat exchanged with its exterior, and W^e is the work done on the system by its surrounding. In order to formulate an energy balance equation for the continuous, such law is considered per unit volume and time, then applied to any fluid element c of the continuum C under consideration. We assume that the heat per unit volume q exchanged by each element with its surrounding can be described through a flux vector $\mathbf{h}$ such that

$$q(c) = \int_{\partial c} -\,\mathbf{h}(\mathbf{x},t)\cdot\hat{n}\,d\sigma$$

We are not taking into consideration any external heat source.

We need to include a kinetic energy term per unit volume K(c) in addition to the system internal energy per unit volume e(c), so that (each term being a function of c, omitted for clarity):

$$\frac{de}{dt} + \frac{dK}{dt} = \frac{dq}{dt} + P^e \tag{2.5}$$

where $P^e(c)$ is the mechanical power per unit volume transferred to c from its surrounding. As usual, proceeding by substitution of integral expressions for each quantity and applying multidimensional integral theorems, the following differential form for the energy conservation equation is obtained:

$$\frac{\partial E}{\partial t} + \nabla\cdot(E\mathbf{v}) = -\nabla\cdot(p\mathbf{v}) + \nabla\cdot(\mathbf{T}\cdot\mathbf{v}) - \nabla\cdot\mathbf{h} + \rho\,\mathbf{b}\cdot\mathbf{v} \tag{2.6}$$

where

$$E = e + \frac{\rho\,(\mathbf{v}\cdot\mathbf{v})}{2} \tag{2.7}$$

is the total (internal + kinetic) energy per unit volume, and p is the fluid pressure. If the fluid density varies in space and time, the fluid is said to be *compressible*. Viceversa, if the density is constant throughout the fluid motion, the fluid is *incompressible*. By considering the continuity equation in Lagrangian formulation:

$$\frac{d\rho}{dt} + \nabla\cdot\mathbf{v} = 0 \tag{2.8}$$

we have that

$$\nabla\cdot\mathbf{v} = 0 \tag{2.9}$$

2.1.1 Euler equations

Euler equations is a system of non–linear PDEs that govern the dynamics of a compressible fluid, such as gases or liquids at high pressures, for which viscous stresses and heat flux are neglected. Starting from equations (2.1), (2.3) and (2.6), Euler equations are obtained under the perfect fluid assumption, for which the stress tensor is given by

$$\mathbf{T} = -p\,\mathbf{I}_3 \tag{2.10}$$

Then, the adiabatic assumption consists in assuming a zero heat flux vector; this means that each fluid element is an isolated system. In the following we will also neglect the presence of external body forces, obtaining the system:

$$\begin{aligned} &\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0 \\ &\frac{\partial (\rho \mathbf{v})}{\partial t} + \nabla (\rho\,(\mathbf{v} \otimes \mathbf{v}) + p\,\mathbf{I}_3) = \mathbf{0} \\ &\frac{\partial E}{\partial t} + \nabla \cdot [(E + p)\mathbf{v}] = 0 \end{aligned} \tag{2.11}$$

Being (2.11) a system of five equations in six unknown quantities (ρ, u, v, w, p, e), an additional closure relation is needed. Thermodynamic and/or constitutive equations usually provide such a closure. Under the ideal gas assumption, it is sufficient to employ a caloric equation, i.e. a thermodynamic relation between the system internal energy, volume and pressure at thermodynamic equilibrium (see [Tor09]). Notice that this is an independent relation from an equation of state, which relates pressure, volume and temperature [1]. For the ideal gas considered here, the caloric equation of state reads:

$$e = \frac{p}{(\gamma - 1)} \tag{2.12}$$

where γ is the adiabatic index. Substituting such expression in the energy equation and making use of the continuity equation, we can put the Euler equations in the following conservative form, where each one is expressed as a time derivative plus a divergence term:

$$\begin{aligned} &\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0 \\ &\frac{\partial (\rho \mathbf{v})}{\partial t} + \nabla \cdot (\rho\,\mathbf{v} \otimes \mathbf{v} + p\,\mathbf{I}_3) = \mathbf{0} \\ &\frac{\partial (\rho^{-\gamma+1} p)}{\partial t} + \nabla \cdot [(\rho^{-\gamma+1} p)\mathbf{v}] = 0 \end{aligned} \tag{2.13}$$

As explained later, this is a convenient form for numerical codes.

[1] The equation of state for the ideal gas $p = \rho k_B T$ may be additionally employed to determine the fluid temperature as a function of pressure and density.

2.1.2 Navier-Stokes Equations

We employ now the well-known Newton's assumption stating that the stress in a moving fluid is linearly related to fluid deformations. From a formal point of view, it is possible to represent this situation by introducing the symmetric part of the velocity gradient tensor or deformation tensor $\mathbf{D}$

$$\mathbf{D} = \frac{\nabla \mathbf{v} + (\nabla \mathbf{v})^T}{2} \tag{2.14}$$

Then, by setting

$$\mathbf{T} = k_0 \, \mathbf{I} + k_1 \, \mathbf{D} \tag{2.15}$$

it is possible to show that k_0 and k_1 must be function of ρ and that the stress tensor can be cast into

$$\mathbf{T} = (-p + \lambda)\mathbf{I} + 2\mu\mathbf{D} \tag{2.16}$$

where λ and μ are called the *bulk* viscosity and the *dynamic* or *shear* viscosity, respectively. For more details see [RM14] and references therein. We will employ these equations under the additional assumption of a constant dynamic viscosity and the condition

$$\lambda = -\frac{2}{3}\,\mu$$

(see [Bat00] for a derivation and specifications about).

By introducing (2.16) in equation (2.3), we can thus describe the fluid through the following continuity and momentum equations

$$\begin{aligned} &\frac{\partial \rho}{\partial t} + \nabla \cdot \rho \mathbf{v} = 0 \\ &\frac{\partial (\rho \mathbf{v})}{\partial t} + \nabla \cdot (\rho \mathbf{v} \otimes \mathbf{v}) - \mu \nabla \mathbf{u} - (\lambda + \mu)\nabla(\nabla \cdot \mathbf{v}) + \nabla p = \mathbf{0} \end{aligned} \tag{2.17}$$

Additional relations are needed to close the system. Since for this work we will consider Navier-Stokes equations in the isentropic, isothermal case, rather than explicitly manipulating the energy equation to establish a closure relation between ρ and p, it is convenient to set the adiabatic constraint directly through

$$p\rho^{\gamma} = \text{const.} \tag{2.18}$$

As a result, the local speed of sound is given by

$$c_s^2 = \frac{dp}{d\rho} = \frac{\gamma p}{\rho} \tag{2.19}$$

and by considering the ideal gas equation of state

$$p = \rho k_B T \tag{2.20}$$

we obtain the relation between sound speed and temperature:

$$c_s^2 = \gamma k_B T \tag{2.21}$$

meaning that the isothermal assumption can be considered by setting a constant speed of sound in the fluid. Such an assumption is usually valid in low Mach number regimes, for which internal heat generation by friction is negligible.

2.2 Treatment of boundary conditions in numerical simulations

One of the main problems to be faced with when performing numerical simulations is the presence of a finite boundary, requiring a different numerical scheme to calculate to calculates the derivatives there. Even if a precise boundary condition is imposed, spurious solutions arise from the boundaries for numerical reasons, and may affect the solution at inner points, especially when performing long time simulations.

As will be shown in this section, Euler equations (2.13) are a hyperbolic system of quasi-linear PDEs. Consequently, the system evolves its variables as a propagation of waves. It is this special feature which allows us to use techniques that deal with the waves supported by the system, rather than directly treating the original system variables. When approaching the boundary domain, outward propagating waves behave following the solution at the boundary and interior points immediately close to it. On the other hand, inward propagating waves at the boundary need the use of nearest points outside the domain, and this is impossible, since the solution there is unknown. A way to overcome this problem is to impose the incoming wave-solution.

2.2.1 Non-reflecting characteristic boundary conditions for hyperbolic PDEs

We start by considering a one-dimensional system of m PDEs of the form:

$$\frac{\partial U}{\partial t} + F(U)\frac{\partial U}{\partial x} = 0 \tag{2.22}$$

In the general case, such equations will be coupled through the matrix F, depending on all of the unknown variables u^j contained in the vector U. Let P be the matrix making F diagonal through a similarity transformation:

$$\Lambda = PFP^{-1} \tag{2.23}$$

where

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \dots & & & \\ 0 & \dots & 0 & \lambda_m \end{bmatrix} \tag{2.24}$$

therefore P transforms all the components of the solution vector U to a new base

$$w^i = P^i_j u^j \tag{2.25}$$

or, in matrix notation

$$W = PU \tag{2.26}$$

Bearing this in mind, we multiply through $I = P^{-1}P$ and write (2.22) as

$$\frac{\partial U}{\partial t} + P^{-1}\left(PF\frac{\partial U}{\partial x}\right) = O \tag{2.27}$$

at this point it is common to introduce the vector $L = (\mathcal{L}^1, \ldots, \mathcal{L}^m)$ defined as

$$L := PF\frac{\partial U}{\partial x} = \Lambda\frac{\partial W}{\partial x} \tag{2.28}$$

so that the original system of equations is written in the form

$$\frac{\partial U}{\partial t} + P^{-1}L = O \tag{2.29}$$

or, multiplying by P and substituting $\mathcal{L}$

$$\frac{\partial W}{\partial t} + \Lambda\frac{\partial W}{\partial x} = O \longleftrightarrow \frac{\partial w^i}{\partial t} + \lambda_i\frac{\partial w^i}{\partial x} = 0 \tag{2.30}$$

Obviously, due to matrix multiplications, these expressions are nothing but a different arrangement of the original variables after a linear combination of them. Quantities w^i defined through (2.25) are called the *characteristic variables* of the system and, following the latter equation, they evolve according to an advection equation with characteristic speed λ_i. It is important to bear in mind, though, that characteristic variables are not independent one from the other since the characteristic equations (2.30) are coupled through the λ_is that are functions of all of the w^i. The quantities $\mathcal{L}^i$, are generally referred to as the *characteristic wave amplitude*. We are interested in their expression, written in terms of the original set of variables u by using equation (2.28), as this identifies which particular combination of them is moving with the corresponding system characteristic speed.

In order to apply the problem boundary conditions with the method of the characteristic variables, a code evolving a PDEs system on a one-dimensional finite domain $[a, b]$ must include a section performing the following operations:

1. determine the characteristic speeds λ_i at the boundary points a and b;
2. according to the sign of λ_i, if w^i is an entering wave, set a specific value to the wave amplitude $\mathcal{L}^i$, according to the problem's boundary conditions;
3. employ equation (2.29) at the boundary for time advancement of the original variables u^i in a and b.

Notice that this approach also allows for the treatment of time-dependent boundary conditions (e.g. entering waves, inlets). In order to realize non-reflecting boundary conditions, meaning that no waves must enter from the boundary and that anything approaching the boundary leaves the domain undisturbed, a first approach would be to set $\mathcal{L}_i = 0$ for entering waves.

In two and more dimensions, the situation becomes slightly more complicated due to a specific projection of the incoming wave from the boundary to be considered. Only the component of a characteristic wave normal to the boundary is affected by non-physical numerical problems originated from the boundary. If we consider the two-dimensional system:

$$\frac{\partial U}{\partial t} + F\frac{\partial U}{\partial x} + G\frac{\partial U}{\partial y} = O \tag{2.31}$$

on the rectangular domain $[0, L_x] \times [0, L_y]$, and we want to constrain waves entering from the left and right edge of the domain $[0, y]$, $[L_x, y]$, $y \in [0, L_y]$, the procedure is performed by considering the x-projected characteristic waves:

$$w_x^i = (P_x)_j^i u^j \tag{2.32}$$

P_x being the matrix making only F diagonal through a base change. To advance the solution, the equation to be considered at the x boundaries is then the x-projected characteristic equation:

$$\frac{\partial U}{\partial t} + P_x^{-1} L_x + G\frac{\partial U}{\partial y} = O \tag{2.33}$$

This procedure is usually called *the method of projected characteristics.* and can be extended to any boundary shape through frame rotation at each point, but we will only consider rectangular domains in the course of this work.

In the next section explicit evaluation of the characteristic variables for the Euler equations system and application of this method will be shown.

2.3 Euler Equations codes

Numerical differentiation, filtering and time advancement schemes, as well as the techniques to manage boundary conditions have been employed to develop a code to simulate Euler equations on a 1- or 2-dimensional domain with open boundaries. In the following section, equations and boundary conditions are presented in detail, together with the results for some standard tests.

2.3.1 1D Euler Equations code structure

For the one-dimensional case, we want to numerically solve the problem (2.11) following the system of equations:

$$\begin{cases} \dfrac{\partial \rho}{\partial t} + \dfrac{\partial \rho u}{\partial x} = 0 \\ \dfrac{\partial \rho u}{\partial t} + \dfrac{\partial}{\partial x}(\rho u^2 + p) = 0 \\ \dfrac{\partial}{\partial t}(p\rho^{1-\gamma}) + \dfrac{\partial}{\partial x}(u\rho^{1-\gamma}p) = 0 \end{cases} \tag{2.34}$$

over the space domain $\Omega = [0, L]$ and time domain $[0, T]$, given initial conditions $\rho_0(x)$, $u_0(x)$, $p_0(x)$ and open boundary conditions. To this end, it is always convenient to put the equations in non-dimensional form to adapt numerical values to characteristic variability ranges in numerical simulations. For Euler equations, this is achieved by introducing a characteristic speed U, a characteristic length L and a reference density $\overline{\rho}$, so that the following dimensionless position, velocity, time, density and pressure have order unity

$$\begin{aligned} &\mathbf{x}^* = \mathbf{x}/L, \quad \mathbf{v}^* = \mathbf{v}/U, \quad t^* = \frac{t}{L/U} \\ &\rho^* = \frac{\rho}{\overline{\rho}} \quad p^* = \frac{p}{\overline{\rho}\, U^2} \end{aligned} \tag{2.35}$$

and similarly, the dimensionless operators:

$$\frac{\partial}{\partial \mathbf{x}^*} = L\,\frac{\partial}{\partial \mathbf{x}} \quad \frac{\partial}{\partial t^*} = \frac{L}{U}\frac{\partial}{\partial t} \tag{2.36}$$

It is easy to see that, when switching to dimensionless variables, Euler equations remain the same. Indeed, this group of equations has a scale-invariant property: if $u(x,t)$, $\rho(x,t)$, $p(x,t)$ is a solution, then

$$\frac{u}{U}\left(\frac{x}{L}, \frac{t}{\tau}\right), \quad \frac{\rho}{\overline{\rho}}\left(\frac{x}{L}, \frac{t}{\tau}\right), \frac{u}{U}\left(\frac{x}{L}, \frac{t}{\tau}\right),$$

is also a solution of the equations when considered in the scaled dimensionless variables.

Conservative formulation (2.13) is adopted for all internal points, allowing a better numerical stability when dealing with sharp gradients. The system of equations is solved numerically by applying a central difference schemes to compute the spatial derivatives. This is done as follows:

$$\begin{cases} \dfrac{\partial \rho}{\partial t} + \dfrac{\Delta_x[\rho u]}{\Delta x} = 0 \\ \dfrac{\partial \rho u}{\partial t} + \dfrac{\Delta_x[\rho u^2 + p]}{\Delta x} = 0 \\ \dfrac{\partial}{\partial t}(p\rho^{1-\gamma}) + \dfrac{\Delta_x[u\, p\, \rho^{1-\gamma}]}{\Delta x} = 0 \end{cases} \tag{2.37}$$

where, as presented in Chapter 1, Δ_x is the operator representative of a chosen spatial differencing scheme, while Δx is the discretization step of the one-dimensional domain Ω. For time advancement, a Runge-Kutta scheme of given order is then employed.

To facilitate the discussion about non-reflecting conditions at the boundary points $0, L$ we adopt Euler equations in their primitive formulation:

$$\begin{aligned} &\frac{\partial \rho}{\partial t} + \frac{\partial(\rho\, u)}{\partial x} = 0 \\ &\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + \frac{1}{\rho}\frac{\partial p}{\partial x} = 0 \\ &\frac{\partial p}{\partial t} + \rho c^2 \frac{\partial u}{\partial x} + u\frac{\partial p}{\partial x} = 0 \end{aligned} \tag{2.38}$$

where

$$c_s^2 = \frac{dp}{d\rho} = \frac{\gamma\, p}{\rho} \tag{2.39}$$

The above system can be rewritten in the following matrix formulation, which is particularly useful for the diagonalization procedures

$$\frac{\partial}{\partial t}\begin{bmatrix}\rho \\ u \\ p\end{bmatrix} + \begin{bmatrix} u & \rho & 0 \\ 0 & u & 1/\rho \\ 0 & \rho c^2 & u \end{bmatrix} \frac{\partial}{\partial x}\begin{bmatrix}\rho \\ u \\ p\end{bmatrix} = \begin{bmatrix}0 \\ 0 \\ 0\end{bmatrix} \tag{2.40}$$

Here the calculation of the characteristic variables follows. Comparing (2.40) with (2.22), the eigenvalues of matrix F are:

$$\begin{aligned} \lambda_0 &= u \\ \lambda_2 &= u + c \\ \lambda_3 &= u - c \end{aligned}$$

and since for a physical system $c = \sqrt{\frac{dp}{d\rho}}$ is real because $\frac{dp}{d\rho} > 0$, the system of equations is strictly hyperbolic. Matrix F eigenvectors are

$$\begin{aligned} \mathbf{V}_0 &= (1, 0, 0) \\ \mathbf{V}_+ &= (1/c^2, 1/c\rho, 1) \\ \mathbf{V}_- &= (1/c^2, -1/c\rho, 1) \end{aligned} \tag{2.41}$$

and therefore the matrices that transform the base are

$$P^{-1} = \begin{bmatrix} 1 & \frac{1}{c^2} & \frac{1}{c^2} \\ 0 & 1/c\rho & -1/c\rho \\ 0 & 1 & 1 \end{bmatrix} \qquad P = \begin{bmatrix} 1 & 0 & -1/c^2 \\ 0 & c\rho/2 & 1/2 \\ 0 & -c\rho/2 & 1/2 \end{bmatrix} \tag{2.42}$$

from which the characteristic variables are immediately obtained

$$w_0 = \frac{\partial \rho}{\partial x} - \frac{1}{c^2}\frac{\partial p}{\partial x} \tag{2.43}$$

$$w_+ = \frac{1}{2}\left(c\rho\frac{\partial u}{\partial x} + \frac{\partial p}{\partial x}\right) \tag{2.44}$$

$$w_- = \frac{1}{2}\left(c\rho\frac{\partial u}{\partial x} - \frac{\partial p}{\partial x}\right) \tag{2.45}$$

$$\tag{2.46}$$

They satisfy the following advection equations

$$\begin{aligned} &\frac{\partial w^0}{\partial t} + u\frac{\partial w^0}{\partial x} = 0 \\ &\frac{\partial w^+}{\partial t} + (u+c)\frac{\partial w^+}{\partial x} = 0 \\ &\frac{\partial w^-}{\partial t} + (u-c)\frac{\partial w^-}{\partial x} = 0 \end{aligned}$$

The first equation states that the quantity w^0 is conserved along curves with slope u, i.e.:

$$\frac{dw^0}{dt} = 0 \quad \text{on} \quad \frac{dx}{dt} = u \tag{2.47}$$

It is interesting to note that this can be associated with some physical meaning. By recalling the following expression for the entropy of a thermodynamical system:

$$s = c_v \ln\left(\frac{p}{\rho^\gamma}\frac{\rho_0^\gamma}{p_0}\right) \tag{2.48}$$

where quantities p_0 and ρ_0 are there for dimensional consistency and are inessential for the purpose of the following differentiation over a generic curve:

$$ds = c_v\left(\frac{dp}{p} - \gamma\frac{d\rho}{\rho}\right) \tag{2.49}$$

carrying out γ/ρ to recognize dw^0 we finally get:

$$ds = \frac{c_v}{\rho}\left(\frac{\rho\, dp}{\gamma\, p} - d\rho\right) = \frac{c_v}{\rho}\left(\frac{dp}{c^2} - d\rho\right) = -\frac{c_v}{\rho}dw^0 = 0 \tag{2.50}$$

Therefore (2.47) is nothing more than entropy conservation along the fluid stream, in accordance with the fact that if no heat exchange occur, each fluid element can be thought as an adiabatic system.

Characteristic variables w^+ and w^- represent the acoustic waves of the system, a combination of pressure and velocity gradient propagating (forward and backward) at the sound speed referred to the local fluid.

$$\begin{aligned} \frac{dw^+}{dt} = 0 \quad \text{on} \quad \frac{dx}{dt} = u + c \\ \frac{dw^-}{dt} = 0 \quad \text{on} \quad \frac{dx}{dt} = u - c \end{aligned} \tag{2.51}$$

Euler equations in primitive formulation with the explicit presence of the characteristic waves' amplitudes, to be employed at the boundary points, are then:

$$\begin{aligned} &\frac{\partial \rho}{\partial t} + \mathcal{L}_0 + \frac{1}{c^2}(\mathcal{L}_+ + \mathcal{L}_-) = 0 \\ &\frac{\partial u}{\partial t} + \frac{1}{c\,\rho}(\mathcal{L}_+ - \mathcal{L}_-) = 0 \\ &\frac{\partial p}{\partial t} + (\mathcal{L}_+ + \mathcal{L}_-) = 0 \end{aligned}$$

In order to apply non-reflecting boundary conditions, it is necessary to check which characteristic waves are incoming from the boundary under consideration. To do that it is necessary to evaluate the sign of the eigenvalues, with the significance of characteristic waves speed at the boundary points. As an example, let's consider a subsonic flow approaching the right boundary, for which the characteristic speeds will be:

$$\begin{aligned} \lambda_0 &= u > 0 \\ \lambda_- &= u - c < 0 \\ \lambda_+ &= u + c > 0 \end{aligned}$$

meaning that only the characteristic acoustic wave w_- will be incoming from outside of the boundary, and non-reflecting boundary condition will consist in imposing zero amplitude $\mathcal{L}_-$ for this wave. Then the whole set of boundary conditions at the right boundary point will be:

$$\mathcal{L}_- = 0$$
$$\mathcal{L}_+ = \frac{1}{2}(u+c)\left(\frac{\partial p}{\partial x} + c\rho\frac{\partial u}{\partial x}\right)$$
$$\mathcal{L}_0 = u\left(\frac{1}{c^2}\frac{\partial p}{\partial x} + \frac{\partial \rho}{\partial x}\right)$$

The numerical code for 1-dimensional Euler equation is reported in Appendix B.

2.3.2 2D Euler equations code structure

We now solve numerically the problem (2.11) which in cartesian coordinates reads:

$$\begin{cases} \dfrac{\partial \rho}{\partial t} + \dfrac{\partial \rho u}{\partial x} + \dfrac{\partial \rho v}{\partial y} = 0 \\ \dfrac{\partial \rho u}{\partial t} + \dfrac{\partial}{\partial x}(\rho u^2 + p) + \dfrac{\partial}{\partial y}(\rho uv) = 0 \\ \dfrac{\partial \rho v}{\partial t} + \dfrac{\partial}{\partial x}(\rho uv) + \dfrac{\partial}{\partial y}(\rho v^2 + p) = 0 \\ \dfrac{\partial}{\partial t}(p\rho^{1-\gamma}) + \dfrac{\partial}{\partial x}(u\rho^{1-\gamma}p) + \dfrac{\partial}{\partial y}(v\rho^{1-\gamma}p) = 0 \end{cases} \tag{2.52}$$

over the space domain $\Omega = [0, L_x] \times [0, L_y]$ and time domain $[0, T]$, given initial conditions $\rho_0(x, y)$, $p_0(x, y)$, $u_0(x, y)$, $v_0(x, y)$ and open boundary conditions.

As for the one-dimensional case, we adopt for all internal points the conservative formulation (2.52). There the equations system is numerically solved by applying a central difference scheme:

$$\begin{cases} \dfrac{\partial \rho}{\partial t} + \dfrac{\Delta_x[\rho u]}{\Delta x} + \dfrac{\Delta_x[\rho v]}{\Delta y} = 0 \\ \dfrac{\partial \rho u}{\partial t} + \dfrac{\Delta_x[\rho u^2 + p]}{\Delta x} + \dfrac{\Delta_y[\rho uv]}{\Delta y} = 0 \\ \dfrac{\partial \rho v}{\partial t} + \dfrac{\Delta_x[\rho uv]}{\Delta x} + \dfrac{\Delta_y[\rho v^2 + p]}{\Delta y} = 0 \\ \dfrac{\partial}{\partial t}(p\rho^{1-\gamma}) + \dfrac{\Delta_x[u\, p\, \rho^{1-\gamma}]}{\Delta x} + \dfrac{\Delta_y[v\, p\, \rho^{1-\gamma}]}{\Delta y} = 0 \end{cases} \tag{2.53}$$

With the same meaning for the symbols as explained for the one-dimensional case. For time advancement, a Runge-Kutta scheme of given order is then employed.

In order to implement non-reflecting boundary conditions on boundary points $\partial\Omega$ we adopt Euler equations in primitive formulation, expressed in the following matrix form, which is particularly useful for diagonalization procedures

$$\frac{\partial}{\partial t}\begin{bmatrix}\rho\\u\\v\\p\end{bmatrix}+\begin{bmatrix}u&\rho&0&0\\0&u&0&1/\rho\\0&0&u&0\\0&\rho c^2&0&u\end{bmatrix}\frac{\partial}{\partial x}\begin{bmatrix}\rho\\u\\v\\p\end{bmatrix}+\begin{bmatrix}v&0&\rho&0\\0&v&0&0\\0&0&v&1/\rho\\0&0&\rho c^2&v\end{bmatrix}\frac{\partial}{\partial y}\begin{bmatrix}\rho\\u\\v\\p\end{bmatrix}=\begin{bmatrix}0\\0\\0\\0\end{bmatrix} \tag{2.54}$$

Explicit calculation of the projected characteristic variables along the x direction is now shown. Due to the symmetry of the equations when switching between x and y directions, results on y boundaries are formally the same with the appropriate indices inversion. Comparing with equation (2.22), the eigenvalues of matrix F are:

$$\begin{aligned}\lambda_0 &= u\\ \lambda_+ &= u+c\\ \lambda_- &= u-c\end{aligned}$$

The system is again a hyperbolic one, though not *strictly* hyperbolic, as the eigenvalue λ_0 has an algebraic and geometric multiplicity of two. Eigenvectors of th matrix F are:

$$\begin{aligned}\mathbf{V}_0 &= (1,0,0,0)\\ \mathbf{V}_1 &= (0,0,1,0)\\ \mathbf{V}_+ &= (1/c^2, 1/c\rho, 0, 1)\\ \mathbf{V}_- &= (1/c^2, -1/c\rho, 0, 1)\end{aligned} \tag{2.55}$$

and consequently the matrices that transform the base are:

$$P^{-1}=\begin{bmatrix}1&0&\frac{1}{c^2}&\frac{1}{c^2}\\0&0&\frac{\rho}{c}&-\frac{\rho}{c}\\0&1&0&0\\0&0&1&1\end{bmatrix}\qquad P=\begin{bmatrix}1&0&0&-\frac{1}{c^2}\\0&0&1&0\\0&\frac{c\rho}{2}&0&\frac{1}{2}\\0&-\frac{c\rho}{2}&0&\frac{1}{2}\end{bmatrix} \tag{2.56}$$

Hence, Euler equations in primitive formulation to employ at the left and right boundary, with explicit presence of the x-projected characteristic amplitudes, read:

$$\frac{\partial \rho}{\partial t} + \mathcal{L}_0 + \frac{1}{c^2}(\mathcal{L}_+ + \mathcal{L}_-) + \frac{\partial(\rho v)}{\partial y}$$
$$\frac{\partial u}{\partial t} + \frac{1}{c\,\rho}(\mathcal{L}_+ - \mathcal{L}_-) + v\frac{\partial u}{\partial y} = 0$$
$$\frac{\partial v}{\partial t} + \mathcal{L}_1 + v\frac{\partial v}{\partial y} + \frac{1}{\rho}\frac{\partial p}{\partial y} = 0$$
$$\frac{\partial p}{\partial t} + (\mathcal{L}_+ + \mathcal{L}_-) + \rho c^2\frac{\partial v}{\partial y} + v\frac{\partial p}{\partial y} = 0$$

with the $\mathcal{L}$ terms being:

$$\mathcal{L}_0 = u\left(-\frac{1}{c^2}\frac{\partial p}{\partial x} + \frac{\partial \rho}{\partial x}\right)$$
$$\mathcal{L}_1 = u\frac{\partial v}{\partial x}$$
$$\mathcal{L}_+ = \frac{1}{2}(u + c)\left(\frac{\partial p}{\partial x} + c\rho\frac{\partial u}{\partial x}\right)$$
$$\mathcal{L}_- = \frac{1}{2}(u - c)\left(\frac{\partial p}{\partial x} - c\rho\frac{\partial u}{\partial x}\right)$$

The same procedure as for the one-dimensional case applies for the treatment of waves approaching or coming from the boundary.

2.3.3 Stability

when dealing with numerical codes, it is necessary to establish a stability criterion relating the time step and mesh size. As explained in section 1.5, a precise evaluation of a CFL condition would require a detailed analysis of each numerical scheme, which is out of the purpose of this work.

In order to define a stability criterion, instead, we employ a reasonable approach based on the fastest moving wave speed proposed by [EFT89]. For a one-dimensional problem, the request is that the numerical domain be always contained in the analytic domain:

$$\Delta t \leq \frac{\sigma \Delta x}{\max_i(|c_i|)} \tag{2.57}$$

Where c is the local sound speed, and σ is a safety factor collecting all of the omitted scheme details, and is empirically determined to be of order 1/10, and always less than one half. Here we evaluate this maximum on the entire domain,

and use the multidimensional generalization:

$$\Delta t \leq \sigma \frac{\sqrt{\Delta x^2 + \Delta y^2}}{\max_{ij}(|\mathbf{v}_{ij}| + c_{ij})} \tag{2.58}$$

2.4 Euler equations code tests

In this section, some standard tests are performed, with the codes described in the previous section and reported in Appendix B.

2.4.1 Sod shock tube test

It is a classical test that can be employed to check the filtering scheme efficiency and its limits in managing shock propagation when extremely steep gradients in the solution arise. The Sod shock tube problem is the evolution of the 1-dimensional Euler equations in the spatial domain $[0, L]$ with the following initial conditions:

$$\rho_0(x) = \begin{cases} 1 & \text{if} \quad x < L/2 \\ 0.125 & \text{if} \quad x > L/2 \end{cases}$$

$$p_0(x) = \begin{cases} 1 & \text{if} \quad x < L/2 \\ 0.1 & \text{if} \quad x > L/2 \end{cases}$$

$$u(x) = 0 \quad \forall x \in [0, L]$$

From a physical point of view, the system can be thought as fluid initially at rest separated in the middle by a membrane abruptly breaking. An extremely steep gradient as the exact one set on initial density and pressure is not manageable by finite difference methods. To appropriately simulate such conditions, it is required a smoothing of the initial conditions on a length scale much less than characteristic length scales of the problem. A typical choice to represent a step with a smooth function is a hyperbolic tangent profile:

$$\tanh\left(\frac{x}{\Delta L}\right)$$

Where ΔL is representative of the characteristic length of variation of the hyperbolic tangent. Consequently, $\Delta x << \Delta L << L$ are qualitative conditions for the numerical algorithm to solve on this length scale and allowing to represent the shock fronts as limits of a smooth numerical solution. Since in a span of $3\Delta L$ a hyperbolic tangent varies from 0 to $\simeq 99.5\%$ of its asymptotic value, left and right values y_L, y_R of the initial conditions can be set by solving the

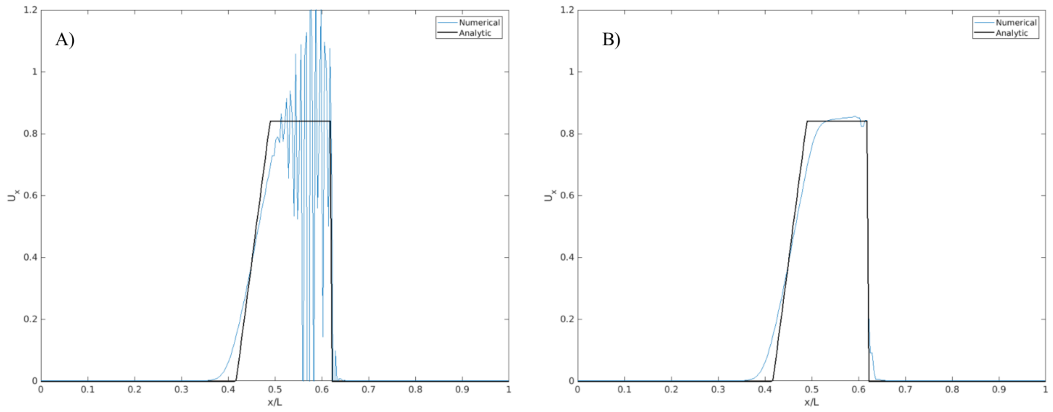


Figure 2.1: A) Example of the extreme dispersive oscillations produced in the numerical solution to the Sod problem for the velocity profile at time $t = 0.065$ near the shock front without filtering. B) Numerical solution to the Sod tube problem with filtering applied every other time advancement step. Analytic solution profile (black) is shown for comparison.

following system

$$\begin{aligned} y_L &= \alpha + \beta \tanh\left(\frac{L/2 - \alpha\Delta L}{\Delta L}\right) \\ y_R &= \alpha + \beta \tanh\left(\frac{L/2 + \alpha\Delta L}{\Delta L}\right) \end{aligned} \tag{2.59}$$

with $\alpha \geq 3$. This problem is of interest for it admits an explicit analytic (non classical) solution, which we will make use of for comparison with our numerical results. Details about the solution method of this Riemann problem can be found in [Tor09] and related references. A code computing this solution is reported in Appendix B.

Figure 2.4.1 presents a comparison between the unfiltered and filtered solutions, showing the filter efficiency and its limits in reducing numerical dispersion near very steep gradients. Despite this test represents an extreme situation, the code behaves rather efficiently.

2.4.2 Acoustic pulse propagation

Euler equations can be effectively employed to model acoustic phenomena, where we are mainly interested in sound waves propagation rather than effects related to viscosity and heat transfer.

In this simulation, the code for simulating 2D Euler equations is employed to compute the evolution of a Gaussian-shaped perturbation in pressure and density, representative of an acoustic perturbation in a still fluid.

We consider the fluid initially at rest and with background density $\overline{\rho} = 1$ and pressure $\overline{p} = 1/\gamma$ (so that $c = 1$ in the still fluid). Initial values for velocity,

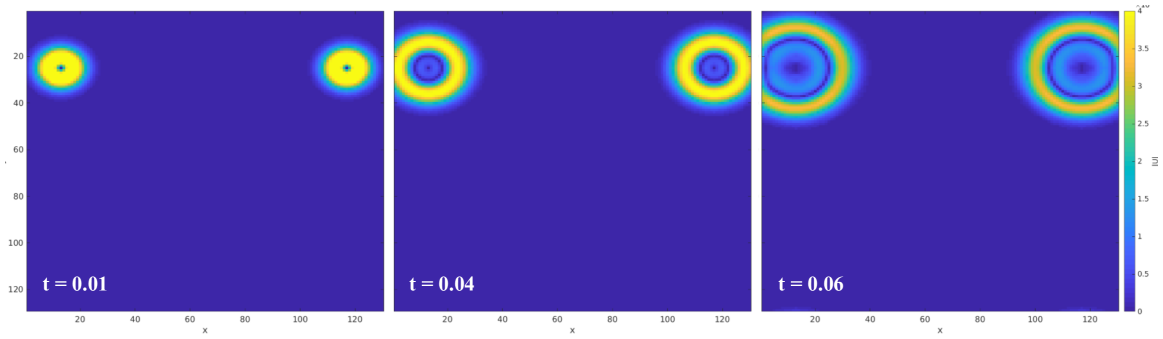


Figure 2.2: Magnitude of the velocity field at three different times for a simulation of two acoustic pressure pulses. y direction no spurious numerical reflections when fluid approaches x boundary are greatly reduced.

pressure and density are then written as follows:

$$\begin{aligned} u_0(x,y) &= 0 \\ v_0(x,y) &= 0 \\ p_0(x,y) &= \frac{1}{\gamma}(1+\varepsilon\, g_0(x,y)) \\ \rho_0(x,y) &= 1+\varepsilon\, g_0(x,y) \end{aligned} \tag{2.60}$$

where $g_0(x,y)$ is the Gaussian function centered in (x_0,y_0), around which the perturbation is localized:

$$g_0(x,y) = e^{-\alpha\,[(x-x_0)^2+(y-y_0)^2]} \tag{2.61}$$

and ε and α are parameters to be employed to set the pulse perturbation amplitude and variance, respectively [BB02].

The main purpose of this simulation is to test code isotropy in the two dimensional case and efficiency of non reflecting boundary conditions. Figure 2.2 shows the magnitude of the velocity field at three different times for the evolution of two acoustic pulses of the form (2.60) localized near the $x = 0, L_x$ boundaries. The two dimensional code, with periodic y direction and open x direction with characteristic-based non reflecting boundary conditions is reported in Appendix B. No significant numerical effects are produced when the fluid perturbation approaches the edges.

2.5 Navier-Stokes equations code structure

2.5.1 Numerical schemes for Navier-Stokes equations

A *predictor-corrector* scheme is a multi-step approach making use of a combination of implicit and explicit schemes. The explicit scheme is employed to predict the solution at a time step, and the the implicit scheme uses this result

to refine the solution. As an example, let us consider an ODE time advancement through the following second-order *implicit Euler* method:

$$y_{n+1} = y_n + h\, f(t_{n+1}, y_{n+1}) \tag{2.62}$$

The unknown quantity y_{n+1} requires algebraic iterations at each time advancement step and may be computationally expensive to solve for. Then the predictor-corrector idea is to estimate the value of y_{n+1} featuring on the RHS of the previous equation by means of an explicit method, for example the explicit Euler:

$$y^p_{n+1} = y_n + h\, f(t_n, y_n) \tag{2.63}$$

and to substitute it in the implicit method:

$$y_{n+1} = y_n + h\, f(t_{n+1}, y^p_{n+1}) \tag{2.64}$$

In this way the use of implicit solvers is no longer necessary, as the implicit method is practically turned into an explicit one. As can be understood, its stability limitations are mainly inherited from the explicit method employed.

Due to ease of computational implementation and recognized effectiveness we employ here the widely studied Mac Cormack method [And12] for PDEs to solve the isothermal, isentropic compressible Navier-Stokes equations (2.17). This scheme puts together the space discretization and time advancement and works as follow. Consider an evolution equation of the form

$$\frac{\partial q}{\partial t} + \frac{\partial q}{\partial x} = 0 \tag{2.65}$$

and indicate forward and backward difference with the symbols

$$\Delta^+[q] = q^n_{i+1} - q^n_i \tag{2.66}$$

$$\Delta^-[q] = q^n_i - q^n_{i-1} \tag{2.67}$$

then the explicit time advancement of the numerical solution consists of the predictor step

$$q^* = q^n - \Delta t \frac{\Delta^+[q]}{\Delta x} \tag{2.68}$$

followed by the corrector step written as

$$q^{n+1} = \frac{1}{2}\left(q^n + q^* \Delta t - \frac{\Delta^-[q]}{\Delta x}\right) \tag{2.69}$$

Using the symbol Δ^0 to indicate the central difference scheme:

$$\Delta^0_x[u] = \frac{1}{2}(u_{i+1} - u_{i-1}) \tag{2.70}$$

Application to Navier-Stokes equations (2.17) of the Mac Cormack method for the two-dimensional problem in cartesian cooordinates gives the following scheme:

$$\rho^*_{ij} = \rho^n_{ij} - \Delta t \left(\frac{\Delta^+_x[\rho u]}{\Delta x} + \frac{\Delta^+_y[\rho u]}{\Delta y} \right)$$

$$\begin{aligned}(\rho u)^*_{ij} =&(\rho u)^n_{ij} + \Delta t \Bigg(- \frac{\Delta^+_x[\rho u^2 + c^2\rho]}{\Delta x} - \frac{\Delta^+_y[\rho uv]}{\Delta y} + \\ &+ (2\mu+\lambda)\frac{\Delta^+_x[\Delta^-_x[u]]}{\Delta x^2} + \mu\frac{\Delta^+_y[\Delta^-_y[u]]}{\Delta y^2} + (\mu+\lambda)\frac{\Delta^0_x[\Delta^0_y[v]]}{\Delta x \Delta y} \Bigg) \\ (\rho v)^*_{ij} =&(\rho v)^n_{ij} + \Delta t \Bigg(- \frac{\Delta^+_x[\rho uv]}{\Delta x} - \frac{\Delta^+_y[\rho v^2 + c^2\rho]}{\Delta y} + \\ &+ (2\mu+\lambda)\frac{\Delta^+_x[\Delta^-_x[v]]}{\Delta x^2} + \mu\frac{\Delta^+_y[\Delta^-_y[v]]}{\Delta y^2} + (\mu+\lambda)\frac{\Delta^0_x[\Delta^0_y[u]]}{\Delta x \Delta y} \Bigg)\end{aligned}$$

for the predictors, and

$$2\rho^{n+1}_{ij} = \rho^n_{ij} + \rho^*_{ij} - \Delta t \left(\frac{\Delta^-_x[(\rho u)^*]}{\Delta x} + \frac{\Delta^-_y[(\rho v)^*]}{\Delta y} \right)$$

$$\begin{aligned}2(\rho u)^{n+1}_{ij} =&(\rho u)^n_{ij} + (\rho u)^*_{ij} + \Delta t \Bigg(- \frac{\Delta^+_x[(\rho u^2 + c^2\rho)^*]}{\Delta x} - \frac{\Delta^+_y[(\rho uv)^*]}{\Delta y} + \\ &+ (2\mu+\lambda)\frac{\Delta^+_x[\Delta^-_x[u^*]]}{\Delta x^2} + \mu\frac{\Delta^+_y[\Delta^-_y[u^*]]}{\Delta y^2} + (\mu+\lambda)\frac{\Delta^0_x[\Delta^0_y[v^*]]}{\Delta x \Delta y} \Bigg) \\ 2(\rho v)^{n+1}_{ij} =&(\rho v)^n_{ij} + (\rho v)^*_{ij} + \Delta t \Bigg(- \frac{\Delta^+_x[(\rho uv)^*]}{\Delta x} - \frac{\Delta^+_y[(\rho v^2 + c^2\rho)^*]}{\Delta y} + \\ &+ (2\mu+\lambda)\frac{\Delta^+_x[\Delta^-_x[v^*]]}{\Delta x^2} + \mu\frac{\Delta^+_y[\Delta^-_y[v^*]]}{\Delta y^2} + (\mu+\lambda)\frac{\Delta^0_x[\Delta^0_y[u^*]]}{\Delta x \Delta y} \Bigg)\end{aligned}$$

for the correctors. The symmetry between predictor and corrector step in the Mac Cormack scheme allows for the backward and forward difference along the various directions to be switched. This switching balances the scheme during the whole time-stepping procedure and prevents asymmetries in the truncation error of the flow variables, which can result in spurious perturbations (see [And95]). [PH06b] and [JR76] found that the forward/backward progressions for the 2D and 3D cases as reported in table 2.5.1 are particularly effective in improving stability, especially for problems with moving discontinuities.

A semi-empirical stability criterion for the Mac Cormack scheme applied to the compressible Navier-Stokes equation was given by [JP97]

$$\Delta t \leq \frac{\sigma}{1 + 2\,Re_\Delta} \left[\frac{|u|}{\Delta x} + \frac{|v|}{\Delta y} + c\sqrt{\frac{1}{\Delta x^2} + \frac{1}{\Delta x^2}} \right]^{-1} \tag{2.71}$$

x derivative at predictor stage	x derivative at corrector stage
F	B
B	F
$x-y$ derivative at predictor stage	$x-y$ derivative at corrector stage
F−F	B−B
F−B	B−F
B−B	F−F
B−F	F−B
$x-y-z$ derivative at predictor stage	$x-y-z$ derivative at corrector stage
F−F−F	B−B−B
B−B−F	F−F−B
F−F−B	B−B−F
B−F−B	F−B−F
F−B−F	B−F−B
B−F−F	F−B−B
F−B−B	B−F−F
B−B−B	F−F−F

Table 2.1: Appropriate sequences for spatial derivative schemes at predictor and corrector stage for the 1D, 2D and 3D Mac Cormack scheme.

where σ is a safety factor and

$$Re_{\Delta} = \min\left(\frac{\rho u \Delta x}{\mu}, \frac{\rho v \Delta y}{\mu}\right) \tag{2.72}$$

is the minimum mesh Reynolds Number (see section 3.1.1). Our code adopts a similar stability criterion.

2.5.2 Non-Reflecting Boundary Conditions for Navier-Stokes equations

Extending what previously discussed about non-reflecting boundary conditions for Euler equations is not straightforward in the compressible NS equations case. The main reason is that the Navier-Stokes equations (2.17) are no longer a first-order hyperbolic PDEs system, for which the theory of characteristic waves is meaningful and well-posed. The addition of the viscous terms significantly changes the mathematical nature of the system; consequently, viscous Navier-Stokes equations do not propagate physical quantities in exactly the same wave-like manner. Therefore, we cannot can no longer strictly speak of characteristic waves. On the other hand, from a physical point of view, we would agree saying that the propagation of waves in a real fluid is a well-observed phenomena, despite of the presence of viscosity. As a consequence, it is necessary to perform a proper analysis leading to a separation of wave behavior and viscous effect on different timescales and/or wavelengths. Specifically, it can be shown that viscosity effects are important for shorter wavelengths. For our purposes, though, we are not going to deal here with the precise theory allowing operator

splitting into the different scales involved, instead, following [LP92], we adopt the reasonable approximation that waves are associated only with the hyperbolic part of the Navier-Stokes equations system. "Waves" associated with viscosity dependent diffusion processes are neglected at the boundaries. This is reasonable as viscosity effects are important for short wavelengths phenomena, which we are not going to include in our simulations. Additionally, when starting our simulations from the domain center, these waves are also rapidly dampened before reaching the boundaries. Such assumption at the boundaries leads to results analogous to the Euler equations case and is called the *Local One Dimensional Inviscid* method (LODI hereafter), which details and applications are explained below.

It is important to stress that this method was a first approach to the problem, and indeed several authors, later on, pointed out its weaknesses and important refinements (see e.g., [LD10]). Despite, we chose to use this method as it results to be well-suited to our purposes.

2.5.3 One-dimensional NS equations in characteristic form

Starting with one-dimensional NS equations in primitive rather than conservative formulation is particularly useful for this case:

$$\begin{aligned} &\frac{\partial \rho}{\partial t} + \frac{\partial \rho u}{\partial x} = 0 \\ &\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + \frac{c^2}{\rho}\frac{\partial \rho}{\partial x} = \nu \frac{\partial^2 u}{\partial x^2} \end{aligned} \tag{2.73}$$

Written in matrix form:

$$\frac{\partial}{\partial t}\begin{bmatrix}\rho \\ u\end{bmatrix} + \begin{bmatrix} u & \rho \\ c^2/\rho & u \end{bmatrix} \frac{\partial}{\partial x}\begin{bmatrix}\rho \\ u\end{bmatrix} = \begin{bmatrix}0 \\ d\end{bmatrix} \tag{2.74}$$

matrix eigenvalues are

$$\begin{aligned} \lambda_+ &= c + u \\ \lambda_- &= c - u \end{aligned}$$

where $d = \nu \frac{\partial^2 u}{\partial x^2}$; characteristic variables are forward- and backward-propagating acoustic waves. Is easily found that

$$P = \frac{1}{2}\begin{bmatrix} \frac{c}{\rho} & 1 \\ -\frac{c}{\rho} & 1 \end{bmatrix} \qquad P^{-1} = \begin{bmatrix} \frac{\rho}{c} & -\frac{\rho}{c} \\ 1 & 1 \end{bmatrix} \tag{2.75}$$

Following the Local Inviscid approach, we neglect the viscous terms d at the boundaries and consider through (2.75) the system (2.73) in characteristic form:

$$\begin{aligned} &\frac{\partial \rho}{\partial t} + \frac{\rho}{c}(\mathcal{L}_+ - \mathcal{L}_-) = 0 \\ &\frac{\partial u}{\partial t} + (\mathcal{L}_+ + \mathcal{L}_-) = 0 \end{aligned} \tag{2.76}$$

where

$$\begin{aligned} \mathcal{L}_+ &= \frac{(u+c)}{2}\left(\frac{\partial u}{\partial x} + \frac{c}{\rho}\frac{\partial \rho}{\partial x}\right) \\ \mathcal{L}_- &= \frac{(u-c)}{2}\left(\frac{\partial u}{\partial x} - \frac{c}{\rho}\frac{\partial \rho}{\partial x}\right) \end{aligned}$$

As usual, boundary conditions are imposed by adequately setting the amplitude of the characteristic waves, as explained previously.

2.5.4 Two-dimensional NS equations in characteristic form

Following the same approach employed in the preceding section, we extend the treatment of boundary condition to the two-dimensional compressible NS equations starting from primitive formulation:

$$\begin{aligned} &\frac{\partial \rho}{\partial t} + \frac{\partial \rho u}{\partial x} + \frac{\partial \rho v}{\partial y} = 0 \\ &\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + \frac{1}{\rho}\frac{\partial p}{\partial x} = \nu \Delta u + \frac{(\lambda+\mu)}{\rho}\frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right) \\ &\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + \frac{1}{\rho}\frac{\partial p}{\partial y} = \nu \Delta v + \frac{(\lambda+\mu)}{\rho}\frac{\partial}{\partial y}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right) \end{aligned} \tag{2.77}$$

in matrix formulation they read

$$\frac{\partial}{\partial t}\begin{bmatrix}\rho \\ u \\ v\end{bmatrix} + \begin{bmatrix} u & \rho & 0 \\ c^2/\rho & u & 0 \\ 0 & 0 & u \end{bmatrix}\frac{\partial}{\partial x}\begin{bmatrix}\rho \\ u \\ v\end{bmatrix} + \begin{bmatrix} v & 0 & \rho \\ 0 & v & 0 \\ c^2/\rho & 0 & v \end{bmatrix}\frac{\partial}{\partial y}\begin{bmatrix}\rho \\ u \\ v\end{bmatrix} = \begin{bmatrix}0 \\ d_x \\ d_y\end{bmatrix} \tag{2.78}$$

eigenvalues of F matrix are

$$\begin{aligned} \lambda_0 &= u \\ \lambda_+ &= u + c \\ \lambda_- &= u - c \end{aligned}$$

and transformation matrices are

$$P = \frac{1}{2}\begin{bmatrix} 0 & 0 & 1 \\ \frac{c}{2\rho} & \frac{1}{2} & 0 \\ -\frac{c}{2\rho} & \frac{1}{2} & 0 \end{bmatrix} \qquad P^{-1} = \begin{bmatrix} 0 & \frac{\rho}{c} & -\frac{\rho}{c} \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix} \tag{2.79}$$

from which the characteristic waves' amplitudes are

$$\begin{aligned} \mathcal{L}_0 &= u\frac{\partial v}{\partial x} \\ \mathcal{L}_+ &= \frac{(u+c)}{2}\left(\frac{\partial u}{\partial x} + \frac{c}{\rho}\frac{\partial \rho}{\partial x}\right) \\ \mathcal{L}_- &= \frac{(u-c)}{2}\left(\frac{\partial u}{\partial x} - \frac{c}{\rho}\frac{\partial \rho}{\partial x}\right) \end{aligned}$$

The x-projected form is thus:

$$\frac{\partial}{\partial t}\begin{bmatrix} \rho \\ u \\ v \end{bmatrix} + \begin{bmatrix} (\mathcal{L}_+ + \mathcal{L}_-)\frac{\rho}{c} \\ \mathcal{L}_+ - \mathcal{L}_- \\ \mathcal{L}_0 \end{bmatrix} + \begin{bmatrix} \frac{\partial \rho v}{\partial y} \\ v\frac{\partial u}{\partial y} \\ v\frac{\partial u}{\partial y} + \frac{1}{\rho}\frac{\partial p}{\partial y} \end{bmatrix} = \begin{bmatrix} 0 \\ d_u \\ d_v \end{bmatrix} \tag{2.80}$$

The Local One Dimensional Inviscid (LODI) method for the two-dimensional case consists in neglecting in equation (2.77) both the transverse and viscous terms at the boundaries, and applying the desired boundary condition through the amplitude $\mathcal{L}$ of the characteristic waves. Even though this may seem an oversimplification, [LP92] showed that it gives excellent results for a wide range of situations. In a more recent paper, [LD10] showed the weaknesses of this approach and discussed possible solutions and improvements to this method. Since at the boundaries the Mac Cormack scheme is not immediately applicable due to the presence of forward and backward differencing, we employed a second-order Runge-Kutta scheme with second-order unilateral first derivatives to compute the quantities involved in the definition of each characteristic variable. The full code is reported in Appendix B.

2.6 Test codes

2.6.1 Lid-driven cavity flow

It is a widely used benchmark problem employed for testing numerical algorithms that solve compressible or incompressible Navier-Stokes equations on a two-dimensional domain. The problem consists of a square cavity filled with

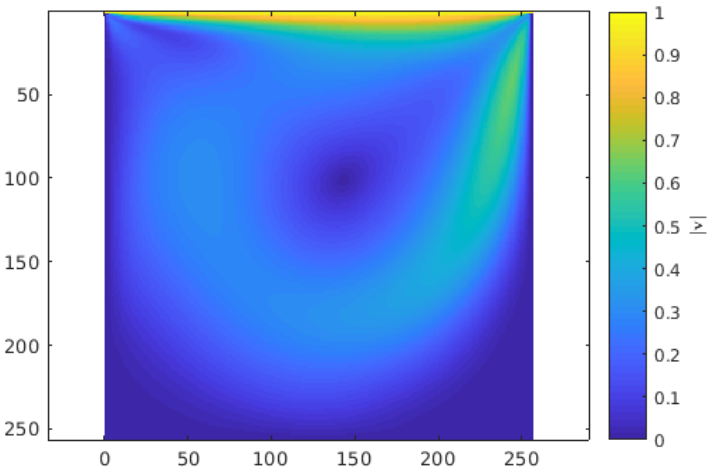


Figure 2.3: Magnitude of the velocity field $\mathbf{v}$ normalized to the lid velocity U_0 at the stationary state for the lid driven cavity with $Re = 400$, $Ma = 0.1$.

fluid, and a sliding lid carries the topmost fluid layer with a constant velocity. As a consequence of the friction, the underlying fluid starts moving, and several vortices are generated until the system reaches a stationary state. No-slip condition is applied at lateral and bottom walls, while at the top boundary the x component of velocity u is set to a constant value U_0. Figure 2.3 shows the simulation result for the velocity field magnitude in the domain. The stationary state is established through a tolerance criterion in the solution variations from one time step to the successive. $L = 1$, $\rho = 1$, $c = 1$, $U_0 = 0.1$ and $Re = 400$ are employed for this simulation. Figure 2.4 shows the comparison between x-component of velocity evaluated on the vertical line passing through the middle of the cavity, as calculated with the algorithm, and the papers [Hou+95], [PH06a] under the same conditions, but employing different schemes. In table 2.6.1, the results for the position of the main vortex are compared. Such excellent agreement between these several methods is remarkable.

	x_V	y_V
Mac Cormack algorithm	0.559 ± 0.004	0.394 ± 0.004
Hou et al. 1995 [Hou+95]	0.5608	0.3922
Perrin et al. 2006 [PH06a]	0.57 ± 0.02	0.39 ± 0.02

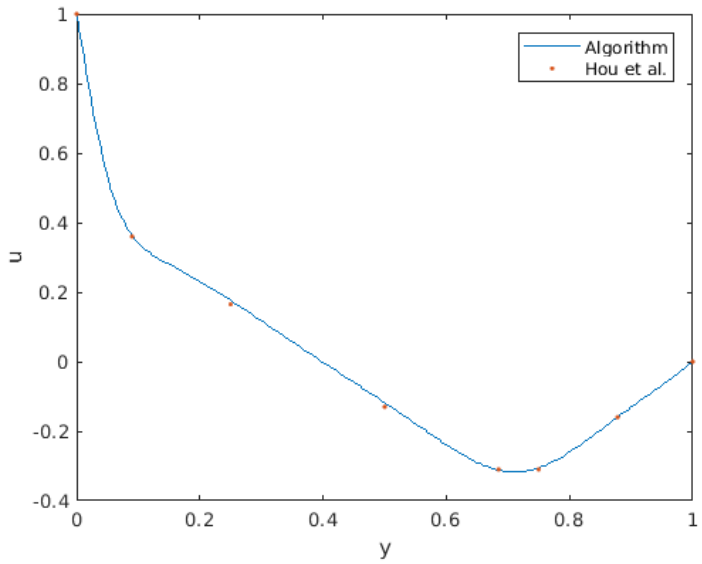


Figure 2.4: Comparison of results for the lid-driven cavity flow with $Re = 400$, $Ma = 0.1$ obtained from the algorithm and [Hou+95]. The shown values are u normalized to the lid velocity U_0 along a vertical line through the center of the domain.

Chapter 3

Fluid dynamical drag

This chapter serves as an introduction to the study of the resistance exerted on a body moving in a fluid stream. Due to the vastness of this topic, only specific aspects, of interest for this work, will be considered. Topics will be covered both with qualitative explanations and discussing some results obtained from numerical simulations.

A particular attention is reserved to the dynamical effects of a fluid stream over a circular cylinder, for which a vast literature exists, allowing to employ it as a reference situation to compare our simulations with. Finally, two-dimensional simulations of the drag force over different shapes have been performed in view of the applications to the space weather forecasting model discussed in Chapter 4.

3.1 Forces on a rigid body in a fluid stream

3.1.1 The Reynolds number

The *Reynolds number* is a particularly useful reference quantity, representing the relative importance of the viscous to the inertial forces in the fluid in a scale-independent manner, meaning that fluids with the same Reynolds number are in a similar dynamic condition. It is employed to check for transition to laminar to turbulent regime in the fluid motion and compute the scales over which dissipative effects become important.

Here, for convenience, we introduce it by starting from the following incompressible Navier-Stokes equations:

$$\begin{aligned} \nabla \cdot \mathbf{v} &= 0 \\ \frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} &= -\frac{1}{\rho_0} \nabla p + \nu \, \Delta \mathbf{v} \end{aligned} \tag{3.1}$$

where $\nu = \mu/\rho_0$ is the so called kinematic viscosity. Similarly to Chapter 2, we introduce the reference quantities L, U and ρ_0, with the significance of a fluid

characteristic length and speed and the constant density of the incompressible fluid, so that the dimensionless quantities

$$\mathbf{x}^* := \frac{\mathbf{x}}{L}, \qquad \mathbf{v}^* := \frac{\mathbf{v}}{U}, \qquad t^* := \frac{t}{L/U}, \tag{3.2}$$

have the order of magnitude of unity. For a solid body placed in a fluid, L identifies a characteristic dimension of the body (e.g. a cross section) and U the uniform velocity of the fluid particles that are very far from the body. With these substitutions, Navier-Stokes equations (3.1) take the following dimensionless form:

$$\begin{gathered} \nabla \cdot \mathbf{v}^* = 0 \\ \frac{\partial \mathbf{v}^*}{\partial t^*} + \mathbf{v}^* \cdot \nabla^* \mathbf{v}^* = -\nabla^* p^* + \frac{1}{Re}\, \Delta^* \mathbf{v}^* \end{gathered} \tag{3.3}$$

where Re is the *Reynolds number*:

$$Re = \frac{UL}{\nu} \tag{3.4}$$

3.1.2 Drag and lift forces

We know from experience that the presence of a solid body in a fluid stream changes the fluid dynamics, deflecting the flow aside and originating eddies and wakes in certain situations. Even though the fluid deflection can be observed in a rather large region around the body, the fluid resistance to the motion of solid objects basically relies in a microscopic interaction between the fluid and the body surface, and viscosity effects are properly confined to a very narrow area around the body called the boundary layer. Recalling the definition of the stress tensor, the total force on the body can be naturally evaluated as the integrated pressure and shear force distribution on the body surface through the surface integral

$$\mathbf{R} = \int_\Sigma (\mathbf{T} \cdot \hat{n} - p\hat{n})\, d\sigma \tag{3.5}$$

Specific components of this force are usually of relevance for applications: if $\hat{\mathbf{u}}$ is the direction of the relative motion with respect to the fluid, the *drag force* is the component of the total force in such direction. In aerodynamics applications, it is the flight-directed force, always tending to retard the motion of the object. The component along a direction perpendicular to $\hat{\mathbf{u}}$ is usually called the *lift* force, because it often refers to the force along the vertical direction. Throughout this work, we will generally refer to the lift force as the component of $\mathbf{R}$ along a direction $\hat{\mathbf{u}}_\perp$ perpendicular to the relative motion. For these forces, it is common to define the following *drag* and *lift dimensionless coefficients*:

$$C_D = \frac{F_D}{\frac{1}{2}\rho\, U^2 A} \tag{3.6}$$

$$C_L = \frac{F_L}{\frac{1}{2}\rho U^2 A} \tag{3.7}$$

Where $F_D = \mathbf{R} \cdot \hat{\mathbf{u}}$ and $F_L = \mathbf{R} \cdot \hat{\mathbf{u}}_\perp$. Interest in these quantities relies in the fact (provable with dimensional reasoning, see [Sch76] for details) that they only depend on the dimensionless quantity formed with U, ρ, L and μ, i.e. on the Reynolds number. When performing fluid dynamics experiments and numerical simulations, this allows to make use of a *similarity principle*, i.e. two systems act in the same way relative to their characteristic scales if they have the same Reynolds number, condensing all the details of a specific physical situation in just one parameter.

When dealing with two-dimensional cases, it is generally convenient to consider a total force per unit length in the third dimension through the line integral

$$\mathbf{r} = \int_{\partial C} p\,\hat{n}\,dl \tag{3.8}$$

where now $\hat{n}$ is the normal to the body contour ∂C in the xy plane. The drag force per unit length is

$$f_D = \mathbf{r} \cdot \hat{\mathbf{u}}_\perp$$

and the drag coefficient is computed as

$$C_D = \frac{f_D}{\frac{1}{2}\rho U^2 d} \tag{3.9}$$

where d is a characteristic linear dimension of the cross-section.

3.1.3 The role of viscosity and the D'Alembert paradox

In the previous section some emphasis, the role of shear stresses in generating the drag force on a solid body in a stream received some emphasis. The set of Euler equations obtained under the assumption of a perfect fluid, without a role of viscosity, cannot be employed to compute the drag force on a body. As an example, we show here how to compute the exact analytic solution for the stationary Euler equations describing the fluid flow over a circular cylinder placed in an asymptotically uniform stream, and how such a situation predicts no drag at all on the cylinder, despite any physical expectation.

We consider Euler equations (2.11) in the stationary two-dimensional case and under the assumption for the fluid to be irrotational and incompressible, the latter being a condition representative for low Mach numbers:

$$\begin{aligned} &\nabla \cdot \mathbf{v} = 0 \\ &\nabla \times \mathbf{v} = \mathbf{0} \\ &(\mathbf{v} \cdot \nabla)\,\mathbf{v} + \frac{\nabla p}{\rho_0} = 0 \end{aligned} \tag{3.10}$$

with the velocity and pressure conditions at infinity in the stream direction

$$\lim_{x\to\infty} p = p_0 \qquad \lim_{x\to\infty} \mathbf{v} = U_0\,\hat{\mathbf{x}} \tag{3.11}$$

and the *impermeable boundary conditions* on the cylinder, expressed as

$$\mathbf{v}\cdot\hat{n} = 0 \quad \text{on} \quad \partial C \tag{3.12}$$

C is a circle of radius R centered at the origin, representative of the cylinder section. Incompressibility and irrotational constraints are sufficient to determine the velocity field, while the momentum equation is employed to relate the pressure to the flow field. In order to proceed, the problem can be placed in $\mathbb{R}^2$ or any unlimited simply connected region $\mathcal{D}$ containing C, so that the relations above are equivalent to state that the two closed differential forms

$$\begin{aligned} \omega_1 &= u\,dx + v\,dy \\ \omega_2 &= -v\,dx + u\,dy \end{aligned} \tag{3.13}$$

are integrable: there exist two functions φ (velocity potential) and ψ (stream potential) such that

$$d\varphi = \omega_1 \tag{3.14}$$

$$d\psi = \omega_2 \tag{3.15}$$

Furthermore, for the irrotational flow condition on $\mathcal{D}$ the velocity field $\mathbf{v}$ can be expressed as the gradient of a function, this implies with (3.14) that such function must be φ itself. Hence, curves $\varphi = const.$ are at any point normal to the velocity field, and since $\nabla\varphi\cdot\nabla\psi = 0$, curves $\psi = const.$ follow the flow lines. The linearity of equations (3.10) allows to add different flow potentials to construct more general fluid flows. We consider the following velocity and stream potentials:

$$\varphi(x,y) = U_0\,x\Big(1 + \frac{a^2}{x^2+y^2}\Big) \tag{3.16}$$

$$\psi(x,y) = U_0\,y\Big(1 - \frac{a^2}{x^2+y^2}\Big) \tag{3.17}$$

obtained from the superposition of the potential for a uniform flow parallel to the x axis and the dipole potential (a source-sink combination), where U_0 is the uniform flow velocity, and a is a parameter related to the dipole strength (see [RM14]). It is convenient to transform into polar coordinates

$$\varphi(r,\theta) = U_0\,r\,cos\theta\Big(1 + \frac{a^2}{r^2}\Big) \tag{3.18}$$

$$\psi(r,\theta) = U_0\,r\,sin\theta\Big(1 - \frac{a^2}{r^2}\Big) \tag{3.19}$$

$$\tag{3.20}$$

Consequently, we obtain the velocity field by taking the gradient of the kinetic potential

$$\begin{aligned} v_r(r,\theta) &= \frac{\partial \varphi}{\partial r} = U_0\, cos\theta \left(1 - \frac{a^2}{r^2}\right) \\ v_\theta(r,\theta) &= \frac{1}{r}\frac{\partial \varphi}{\partial \theta} = -U_0\, sin\theta \left(1 + \frac{a^2}{r^2}\right) \end{aligned} \tag{3.21}$$

It is seen that $v_r = 0$ for all points at $r = a$, so that, if $a = R$, the velocity field matches the boundary condition and therefore must be the solution for the flow in the domain outside of the circle C. Figure 3.1 presents a comparison of the streamlines between the analytic solution and the laboratory experiment for a low Reynolds number, which, conversely, is a viscosity-dominated situation.

Despite such apparent qualitative agreement, this simple case is sufficient to present a significant result to bear in mind when perfect fluid equation are employed to compute the net force on a rigid body. To obtain the pressure from the velocity field, we consider in (3.10) the momentum equation and apply the following vector differentiation rule

$$(\mathbf{v} \cdot \nabla)\mathbf{v} = (\nabla \times \mathbf{v}) \times \mathbf{v} + \frac{1}{2}\nabla(\mathbf{v} \cdot \mathbf{v}) \tag{3.22}$$

The quantity

$$(\nabla \times \mathbf{v}) \times \mathbf{v} = -\nabla \left(\frac{1}{2}(\mathbf{v} \cdot \mathbf{v}) + \frac{p}{\rho_0}\right) \tag{3.23}$$

is null because of the irrotational assumption, meaning that

$$\frac{\rho_0}{2}(\mathbf{v} \cdot \mathbf{v}) + p$$

is constant over all the fluid domain. This result is nothing but a particular case of the well-known Bernoulli's equation for the stationary, incompressible and irrotational case. Finally, since the problem boundary conditions prescribe pressure and velocity very far from the cylinder, this constant value can be quantified as

$$\frac{\rho_0}{2}(\mathbf{v} \cdot \mathbf{v}) + p = \frac{\rho_0}{2}U_0^2 + p_0 \tag{3.24}$$

From equation (3.21)

$$\mathbf{v} \cdot \mathbf{v} = 4\, U_0^2 \sin^2\theta \tag{3.25}$$

the pressure in space depends only on θ and has the expression

$$p(\theta) = \frac{\rho_0\, U_0^2}{2}\left(1 - 4\, \sin^2\theta\right) + p_0 \tag{3.26}$$

We can now compute the total force per unit length on the cylinder through equation (3.8)

$$\mathbf{r} = \int_{\partial C} p\, \hat{n}\, dl = \int_0^{2\pi} p(\theta)\, \hat{n}\, R d\theta \tag{3.27}$$

and substituting $\hat{n} = \cos\theta\,\hat{\mathbf{x}} + \sin\theta\,\hat{\mathbf{y}}$ it can be easily found that

$$\mathbf{r} = R\left[\int_0^{2\pi} p(\theta)\cos\theta\hat{\mathbf{x}}\,d\theta + \int_0^{2\pi} p(\theta)\sin\theta\,\hat{\mathbf{y}}d\theta\right] = \mathbf{0} \tag{3.28}$$

meaning that the fluid does not exert any force - no drag nor lift - on the cylinder.

This somewhat counter-intuitive result is a particular example of a more general one that also applies to the compressible fluid case and over a body of arbitrary shape, known as the *D'Alembert paradox*: *the irrotational flow of a perfect fluid gives zero drag on any obstacle placed in a fluid stream.* Notice that a lift force can be non zero by dropping the irrotational fluid assumption.

We conclude that the perfect fluid description cannot be employed to explain and quantify details about the drag force, which is of our interest in this work. Numerical simulation aimed at determining the drag coefficient must necessarily employ the Navier-Stokes equations, since the fluid viscosity in the vicinity of the body is the real cause of the body drag force.

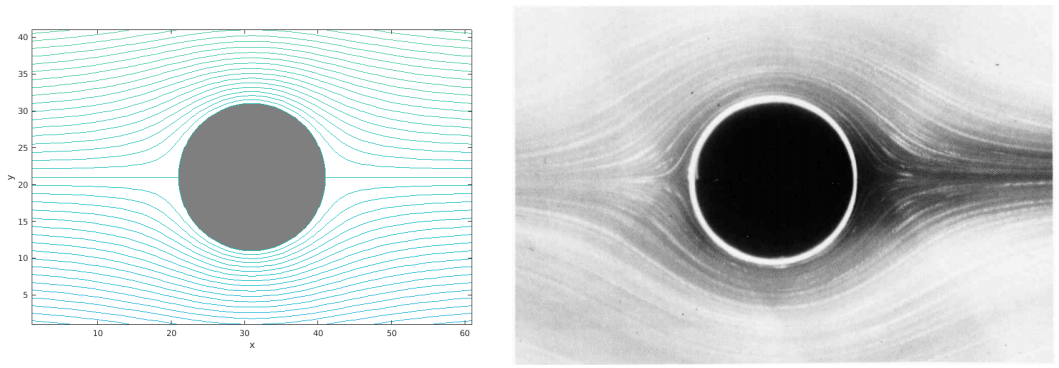


Figure 3.1: Visual comparison of analytic streamlines around a cylinder in uniform nonviscous flow (left) and experimental (right) for $Re = 0.16$. Picture from [Dyk88]).

3.2 The relation between drag and viscosity from experiments

Because of the similarity principle, it is meaningful to perform experiments to determine the relationship between the drag and the characteristics of a fluid stream by plotting a C_D vs Re curve.Typical examples, widely studied both from a theoretical and an experimental viewpoint, are the uniform flow past a cylinder and a sphere. In this section, we will consider the first case, which can be studied as a two-dimensional problem, and present a comparison between experimental and numerical results obtained by employing numerical

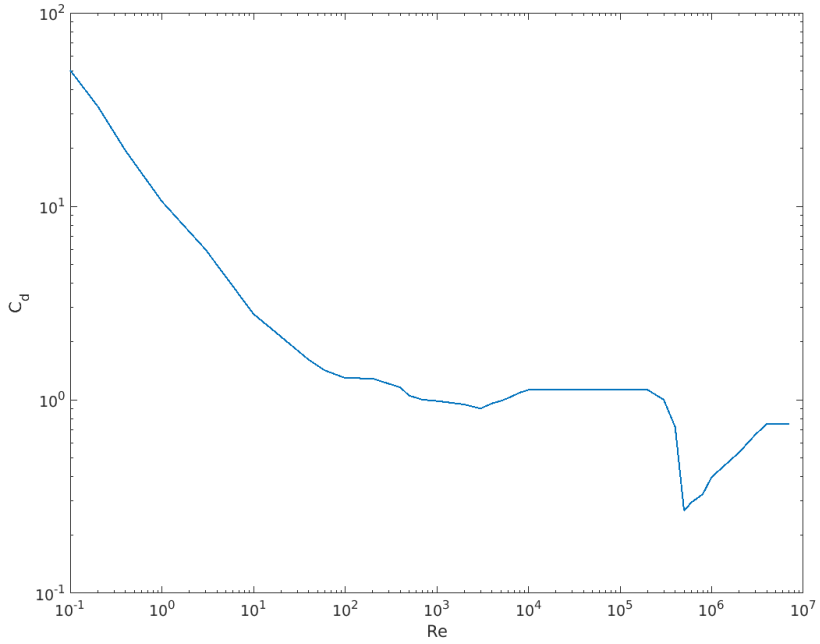


Figure 3.2: Plot of the experimental drag coefficient versus the Reynolds number for a Cylinder in a stream generated using the experimental results taken from [Sch76].

schemes and techniques developed in the previous chapters, and the COMSOL Multiphysics® simulator.

3.2.1 Cylinder in a stream

Figure 3.2 shows the experimental results for the drag coefficient C_d as a function of the Reynolds number Re for a cylinder in a fluid stream. As expected, a complicated behaviour occurs and several regimes, involving specific physical aspects, are observed. Salient features depending on the Reynolds number are:

- $Re < 0.5$: For very low Reynolds numbers, the effect of viscosity is dominant over all other forces. Smooth, regular flow patterns are to be expected in this case, as shown in the right panel of Figure 3.1.

  From a theoretical point of view, the case of a stationary fluid with vanishing (but non zero) viscosity is called a *creeping flow* or *Stokes' flow* and can be managed in principle using the incompressible stationary Navier-Stokes equation in the approximation obtained by neglecting convective (inertial) terms. For the two-dimensional problem, this leads to the fol-

lowing second-order linear system:

$$\begin{aligned} &\frac{\partial u}{\partial x} + \frac{\partial v}{\partial v} = 0 \\ &\mu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} - \frac{\partial p}{\partial x}\right) = 0 \\ &\mu\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} - \frac{\partial p}{\partial y}\right) = 0 \end{aligned} \tag{3.29}$$

Despite the simplification expected in having a linear system of differential equations with constant coefficients, unfortunately the study of creeping flows in two dimensions leads to the so-called *Stokes paradox*, stating that there is no way to find a velocity field satisfying at the same time the no-slip condition on the circle and the free stream condition far away from it. There is not enough space here to consider historical and analytic details about the solution of this problem; we then limit ourselves to consider the following scaling law for the drag force per unit length on a cylinder at small Re, obtained through the analysis of the creeping flow equations [Lag]:

$$f_D \simeq \frac{\mu U_0}{Log(\frac{U_0 d}{\nu})} = \frac{\mu U_0}{Log(Re)} \tag{3.30}$$

giving for the drag coefficient

$$C_d \simeq \frac{2}{Log(Re)Re} \tag{3.31}$$

A linear-like decreasing trend, consistent with such relation, is observed Figure 3.2 for low Reynolds numbers.

- $1 < Re < 100$: The system develops a stable pair of vortices on the downwind side of the cylinder. The high pressure originated by these stable vortices is responsible for the high drag on the cylinder in this range of Reynolds numbers. When increasing the value of Re in this range, the drag decreases and the eddies elongate downstream up to a breaking point. Figure 3.3 shows a graphical comparison between the streamlines obtained through simulation of stationary compressible Navier-Stokes equations and the experimental results for the same Reynolds number.

- $100 < Re < 10^3$: the pressure stability between the upper and the lower eddies easily breaks under the effect of the slightest asymmetry in the fluid stream approaching the cylinder. Such instabilities grow and the tail starts to oscillate, giving rise to the so-called Von Karman vortex. A stationary model cannot be employed anymore to properly describe such situation. Even though unsteady, this type of flow is periodic, repeating itself at some time interval. Figure 3.5 shows our results from the two-dimensional

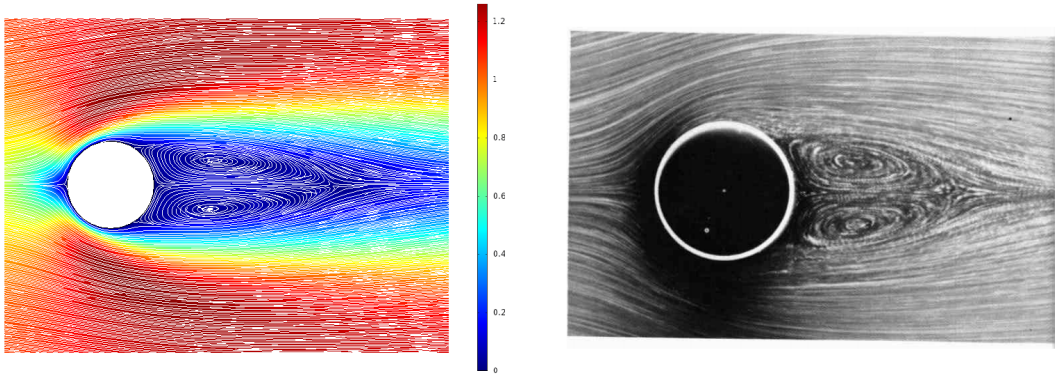


Figure 3.3: Visual comparison of streamlines around a cylinder in a viscous flow at low Reynolds number of value 26. Numerical simulation (left) where the colorbar represents magnitude of velocity normalized to inlet speed. Experimental (right). Picture from [Dyk88].

simulations of the drag coefficient in time at Re = 1000 after the initial transient.

- $10^3 < Re < 10^4$: the flow in the boundary layer on the upstream side of the cylinder maintains itself ordered and laminar. As the flow turns onto the downstream side of the cylinder, the turbulent wake starts. Such wake is not as wide as for the Von Karman vortex, so the drag is slightly less.

- $Re > 10^4$: the system has turned its behaviour to a turbulent, chaotic regime. Of particular interest in the regime of very high Reynolds numbers is the so-called *drag crisis*, where the cylinder experiences a drop in the drag resistance. This is associated with a transition from well-organized vortex shedding to a randomized one for super-critical Reynolds numbers, eventually returning to well-organized shedding at the post-critical Reynolds number with high drag force coefficients.

  In order to follow such instabilities growth and development, fine space and time simulations or turbulence models are required or, as an alternative, turbulence model for describing the evolution of averaged values of the field quantities. These topics are out of the purpose of this work.

Figure 3.6 shows the comparison between experimental results and our numerical simulations of the drag coefficient. Numerical values obtained for C_D agree within a 10 % tolerance with experimental values up to $Re \simeq 10^5$. At higher Reynolds number, our codes are not able to sufficiently reproduce instability growth and turbulence effects, smoothing out such effects so that C_D remains a constant.

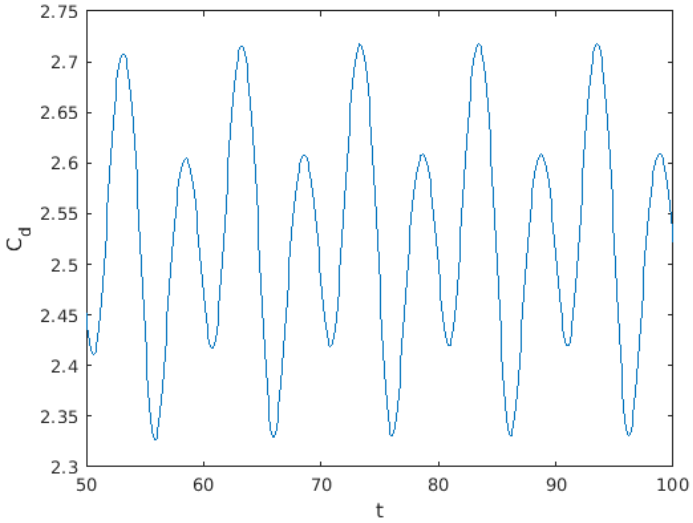


Figure 3.4: Values of the drag coefficient on a cylinder as a function of time. For this simulation, originating Von Karman vortex street, $Re = 1000$. The time-averaged value for C_d is 2.53. Values related to the transient evolution at the beginning of the simulation are not shown.

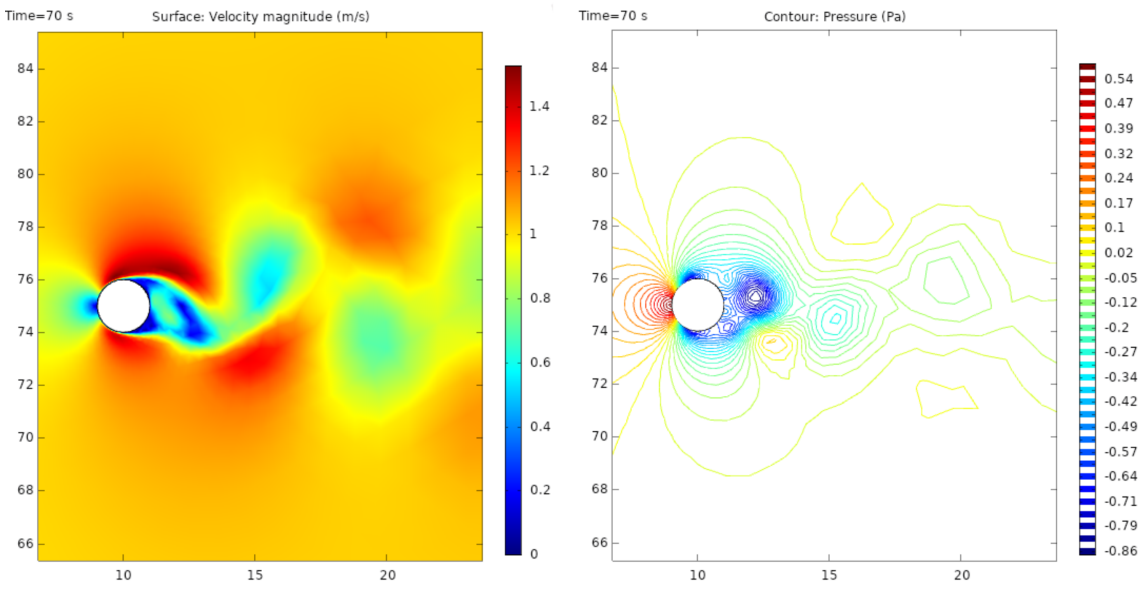


Figure 3.5: Representation of the velocity and pressure field after Von Karman Vortex are formed, obtained from the numerical simulation of the flow over a cylinder through the COMSOL Multiphysics®software.

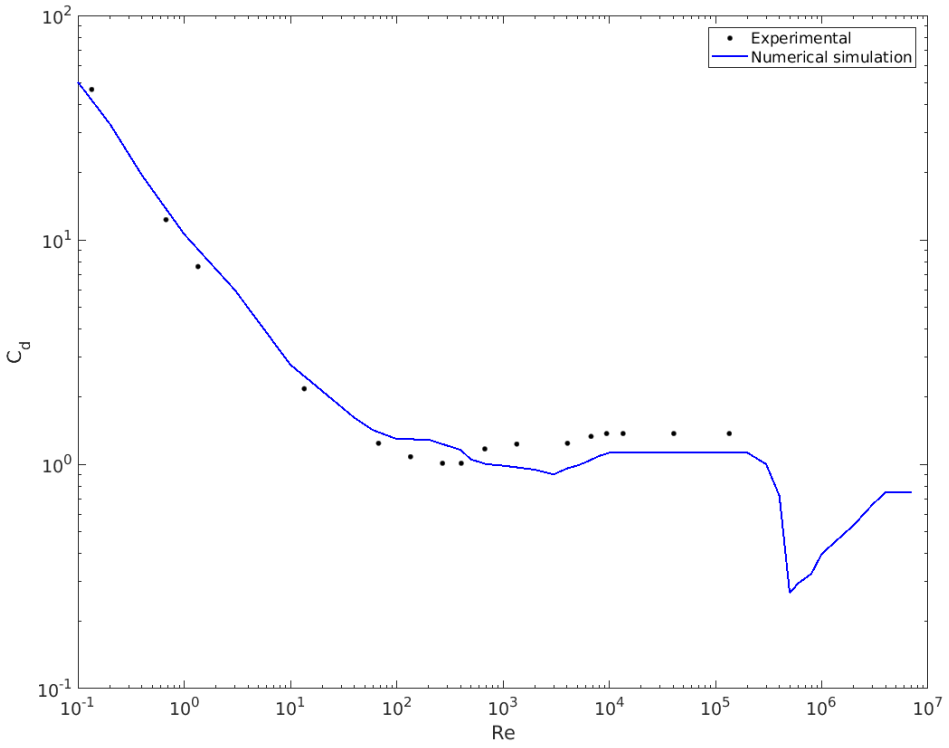


Figure 3.6: Numerical values (black dots) and experimental values (blue line) for the drag coefficient of a cylinder in a uniform flow.

3.3 Numerical simulations of drag on different shapes

The comparison of the numerical results with the experimental ones presented in the previous section, although restricted only to a specific case, provided a test of our codes and showed the limits of our simulations. Bearing in mind such level of performance for the code at our disposal, we performed numerical simulations to compute the Re vs C_D curve for several two-dimensional solid body shapes in a flowing stream. The purpose of such simulations, and the reasons behind the choice of these shapes will be made more manifest in the next chapter.

Table 3.1 summarises the results, while Appendix A contains shape details and drag coefficient plots. Some comments about these results are necessary. First, we noticed that the behaviour of the drag coefficient for all shapes is in qualitative agreement with the overall result obtained for the drag on a cylinder, presenting high values for small Reynolds numbers, a decreasing trend and finally settling to a constant value. Asymmetric shapes with respect to an axis perpendicular to the stream flow (shapes 4 and 5) offer a different amount of drag depending on the direction of the approaching fluid stream. In particular, a shape presenting a clear concavity to the flow stream (shape 5) causes a higher drag force. This has to be expected as the fluid tends to pile up and finds a bigger surface to interact with in the shape inward depression. For such cases. a "forward" and "backward" drag coefficients are thus distinguished, the latter being when the flow stream comes towards the concavity.

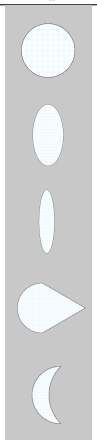
#	Shape	C_d^F	C_d^B
1		1.3	-
2		2.0	-
3		1.7	-
4		1.0	1.5
5		2.2	3.6

Table 3.1: Drag coefficient at high Reynolds number found from numerical simulations for the listed two-dimensional shapes.

Chapter 4

The drag-based model for ICME propagation

In this chapter, our focus is on the study of a classical model providing an essential description of Coronal Mass Ejections propagation in interplanetary space. After a general introduction to Coronal Mass Ejections and their effects on the near-Earth environment, we show in detail an analytical, fluid dynamical-based model widely employed to describe ICME interplanetary propagation. Basing on the outcomes from numerical simulations of the motion of solid bodies in an external flow, considered as representative of ICME moving into the solar wind, we attempt an improvement to this model by dropping the assumption of a constant drag efficiency irrespective of the relative motion of the ICME with respect to the solar wind. A list of ICME events collecting data from remote CME observations and their arrival to the Earth is employed to perform an inversion procedure allowing to compute the model parameters for each event and to test our hypothesis with experimental values.

4.1 Coronal Mass Ejections and Interplanetary Coronal Mass Ejections

Coronal Mass Ejections (hereafter CMEs) are highly energetic eruptions of matter in plasma-state and magnetic field from the Sun. These spectacular events of solar activity can move into interplanetary space a mass larger than 10^{13}kg with a velocity up to several thousands of kilometres per second, which means a kinetic energy that may exceed $10^{25}J$. Following [AJ+84] and [Sch96], a CME may be more precisely defined as *"an observable change in coronal structure that occurs on a time scale of a few minutes and several hours and involves the appearance and outward motion of a new, discrete, bright, white-light feature in the coronagraph field of view"*. Figure 4.1 shows a CME event observed by the *Large Angle Spectrometric Coronagraph* (LASCO) on board the *Solar and Heliospheric Observatory* spacecraft (SOHO; Brueckner et al. 1995). This is

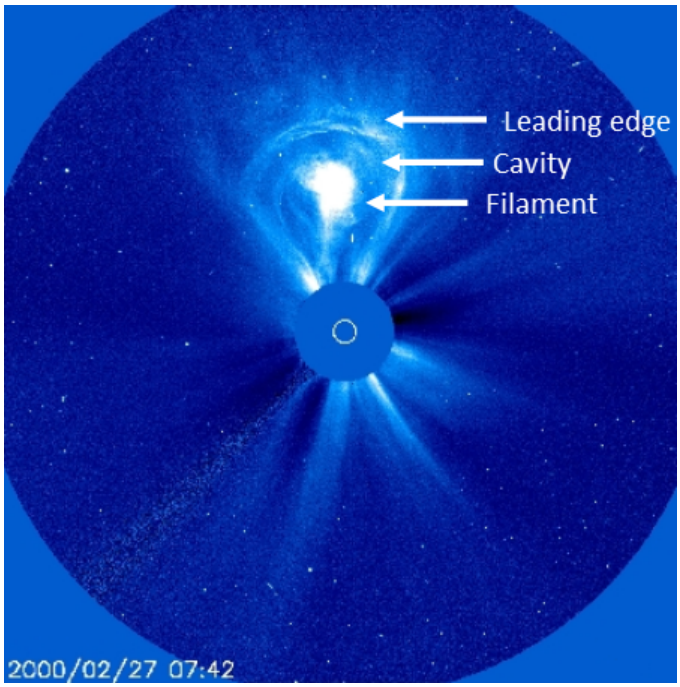


Figure 4.1: A "classic" three-part CME, often associated with filament eruptions from the polar crown, as observed by the LASCO coronagraph on board SOHO. Arrows indicate the leading edge, the cavity and the filament within. Image from the SOHO webpage courtesy of NASA/ESA.

a typical CME often associated with filament eruptions from the polar corona, featuring a leading-edge followed by a dark cavity region with a bright filament within. Other possible structures commonly occur, as jets and narrow CMEs with no resemblance to the three-part structure, CMEs with a clear flux-rope morphology and halo CMEs.

CMEs have been observed to erupt from any region of the corona, even though they are more often associated with lower latitude regions and with the heliospheric current sheet (the surface where the polarity of the Sun's magnetic field changes). Their occurrence is strongly dependent on the solar activity: CMEs typically erupt from the Sun with a frequency of around once a day to about three per week during solar minimum, and around four or five times per day during solar maximum. CMEs from active regions have been observed to be in close association with major solar flares, but they can also come from filament channels in the quiet Sun.

There is uncertainty in the composition of the plasma generated during a CME; a general agreement is that the key to understanding CMEs composition must rely in their formation mechanism. The reason is that, by assuming that the formation and launch of a CME from the Sun are magnetically dominated, most of the mass of a CME must be that of the region where the closure of the CME flux rope or magnetic cloud and its subsequent launch takes place. If a

CME initiates in the lower solar atmosphere such as the photosphere, then its abundance is expected to be similar to that of this region, with a dominance of hydrogen, helium and other lighter elements. On the other hand, if a CME initiates in the corona, then it is expected to be mostly comprised of coronal material and to contain heavy ions that are formed at very high temperatures, as revealed by spectrographic studies of the solar corona. Unfortunately, this picture is not so simple, since we know that emerging regions in the corona, such as from new active regions, are composed of material originating at deeper layers, and therefore a CME initiating in the corona could be composed of photospheric plasma. As a result, the actual CME composition is highly variable with contributions coming from many regions on the Sun.

The heliospheric counterpart of a CME, i.e. a CME when it is at much larger distances from the Sun (say a couple of tenth of solar radii), is referred to as an *Interplanetary Coronal Mass Ejection* (ICME). ICMEs characteristics, such as velocity, plasma composition and magnetic field, can be directly measured with instruments on-board spacecraft crossed on their path, and so much more is known about them than for CMEs. We know that ICMEs have large masses and contain a magnetic field, but they are generally not as fast as CMEs. It is almost certainly due to the significant deceleration imposed on fast CMEs by the surrounding solar wind during their interplanetary travel, and considering a model for the description of such an interaction is one of the subjects of this chapter. However, many ICMEs are still *supersonic*, meaning that they often cause shocks in the interplanetary medium which are subject to other secondary effects, such as solar energetic particles (SEPs) acceleration and electromagnetic radio bursts. It is the ICME and its shock that impact with the Earth and can cause a variety of phenomena known under the name of Space Weather effects.

With regard to ICMEs particle composition, direct measurements from the early 1970s revealed a Helium abundance enhancement following interplanetary shocks, and high ionisation states of Oxygen and Iron, such as Fe^{10+} and even Fe^{16+}. These high-temperature ions are generally regarded to originate low in the solar corona or from heating during the launch of the CME. ICMEs also contain cooler ions such as He^{+}, Mg and Ne, and their low temperatures are consistent with an origin from the filaments that erupted behind the CME. Despite the high variability in their composition, it is also possible to find regularities: for example, He^{++} to H^{+} ratio and Fe^{16+} enhancements are common features to many ICMEs. Unfortunately such detailed knowledge of the content of particles of a ICME does not allow us to make any statement about the composition of a CME at the Sun, since the interplanetary medium, through which the CME has often spent several days travelling before arriving at the observing spacecraft, also contains material which may interact with the CME. It is not well-established whether or not there can be some matter exchange between the CME plasma and the surrounding plasma it will encounter into interplanetary space (in principle, if the CME is a closed magnetic structure, its material

cannot interact with the environment, except via magnetic reconnection); if the case of no-matter interaction were true, the composition observed at 1 AU must be the same as that originating at the Sun.

4.2 ICMEs dynamics

The physical reasons for which CMEs originate and are launched into interplanetary space are still not clear. Likewise, once a CME leaves the Sun, the proper mechanism by which it propagates through the heliosphere is also not well understood. The main reasons under such missing knowledge are a general lack of observational evidence and the primary difficulty in approaching to a complete description of the ICME dynamics in a theoretical way. On the other hand, for the consequences of their impact on the Earth, methods for the prediction of the arrival of Earth-directed events are a topic of primary importance in Space Weather. For this reason techniques for the forecasting of ICME arrivals, mostly based on the observational evidence at our disposal, have equally developed on statistical, empirical and numerical basis.

Observations through coronagraphs of early phases of massive eruptions of plasma from the Sun found that emerging CMEs have velocities in the range between 100 and well beyond 2000 km/s [HBC94]. On the other hand, observations during the phase where solar eruptions have reached a considerable distance from the Sun revealed that ICMEs have velocities that typically differ by only about 100-200 km/s from the solar wind speed [Car04]. During their interplanetary journey, CMEs initially faster than the solar wind experience a deceleration, while those slower than the solar wind are accelerated. Hence, the forces acting on the ICME in the interplanetary medium must lead to an equalization of the ICME and solar wind velocities.

Models for the description of the dynamics of a CME assume that their generation near the Sun and consequent launch and propagation into interplanetary space are determined mainly by the Lorentz force due to the Interplanetary Magnetic Field, by the inward gravitational force of the Sun and by its interaction with the solar wind. [Che96] and [Vrs90] showed using of MHD models that all of these three interactions actually govern the dynamics of an erupting flux rope, and predicted that the Lorentz force reaches its maximum within a distance of several solar radii and becomes negligible at more considerable distances. Gravity is also negligible over several solar radii and can in no way account for the observed acceleration and deceleration of the ICME, leaving the interaction with the solar wind as the fundamental one in determining the ICME propagation.

4.3 The Drag-Based model

All of the previous considerations have left the interaction with the solar wind as the fundamental one in determining the ICME propagation. In order to describe such interaction adequately, several mechanisms based on a drag force acting on the ICME, analogous to the force expected on a solid body immersed in a fluid, have been proposed ([Car+96], [Vrs+12]), and, due to their effectiveness, are still investigated.

The Drag-Based model for ICME propagation [Vrs+12] starts from the assumption that the fundamental interaction driving ICME propagation in interplanetary space is the one with the solar wind, and that this can be modeled as a drag-like force acting on the ICME structure as a whole.

Starting from the expression (3.6) for the drag coefficient, we assume the drag force exerted by the solar wind on an ICME is written as:

$$\vec{F}_D = -\rho_W A\, C_d\, (\vec{v} - \vec{w})|\vec{v} - \vec{w}| \tag{4.1}$$

where $\vec{v}$ and $\vec{w}$ are the ICME and solar wind velocities, ρ_W is the solar wind mass density, A is the ICME cross-section in the direction of relative motion to the fluid and C_D is the drag coefficient, which collects at once the geometry and the other physical features of the interaction. It must be observed that the values of A and C_D, as well as w and ρ_W, are time and position-dependent in the general case. In the following, we will consider a stationary and purely radial solar wind so that $\vec{w} = w(r)\,\hat{r}$ and $\rho_W = \rho_W(r)$.

The next step is to insert this expression for the drag force into the classical Newton motion law

$$M\frac{d\vec{v}}{dt} = \vec{F}_D \tag{4.2}$$

and consider a polar coordinates system (r, θ, ϕ) with the origin at the centre of the Sun to arrive at the following second-order ordinary differential equation for the radial distance $r(t)$:

$$\frac{d^2r}{dt^2} = -\gamma \left(\frac{dr}{dt} - w\right)\left|\frac{dr}{dt} - w\right| \tag{4.3}$$

where, after rearranging some terms

$$\gamma := \frac{\rho_W A}{M}\, C_D \tag{4.4}$$

is the so-called *drag parameter*. Solar wind speed and drag parameter in equation (4.3) are position-dependent quantities, so it is necessary to consider their spatial behaviour. For the solar wind, we assume a constant radial speed during the whole ICME propagation. This may be considered a valid choice, since by applying the continuity equation to a radially directed and time-independent solar wind flow we obtain:

$$\frac{1}{r^2}\frac{d(r^2 w \rho_W)}{dr} = 0 \tag{4.5}$$

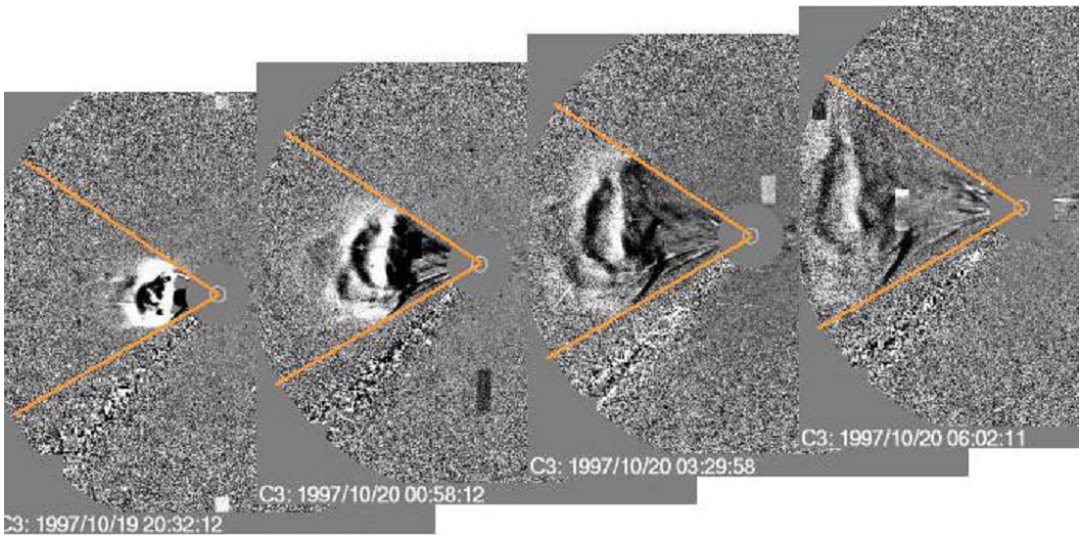


Figure 4.2: Self-similar expansion of the limb CME of 19 October 1997 seen by LASCO. The angle $\varphi = 65°$ between the outer edges of the opposing flanks of the limb CME is well maintained (at least within the LASCO-C3 FOV). From Schwenn *et al.* 2005 [Sch+05].

which implies $r^2 w \rho_W$ is a constant value. Next, we employ the empirical model derived by Leblanc *et al.* [LDB98] for the solar wind electron density n_e in the inner heliosphere:

$$n_e = \frac{8.0 \times 10^7}{r^6} + \frac{4.1 \times 10^6}{r^4} + \frac{3.3 \times 10^5}{r^2} \tag{4.6}$$

where r is the radial distance from the Sun expressed in solar radii, and the result for n_e is in cm^{-3}. The first two terms are needed to account the measured behaviour of the solar wind density for small distances and are irrelevant beyond $R \simeq 20\, R_\odot$. Hence, for the distances to be considered in evaluating ICME propagation, the main contribution to the solar wind density comes from the third term. Equation (4.6), with the principal dependence from the $1/r^2$ term for interplanetary distances, leads to $w \approx$ const. within a few percents.

About the drag parameter, it is also necessary to discuss all of the quantities featuring in equation (4.4) to define its spatial behaviour. A well-established observative fact relevant for discussing the general behaviour of the ICME cross section A is the so-called *"self-expansion"*, meaning that expanding CMEs are maintain shapes during their propagation. A representation of this property from coronagraph observations is shown in Figure 4.2, taken from [Sch+05]. In the same work, they discuss how the observed shapes of the vast majority of ICMEs appear to be consistent with nearly perfect circular or elliptical cross-sections. Since self-expansion does preserve the ICME angular width, this leads to the conclusion that the area A from the radial direction is expected to scale as r^2 as the ICME expands while propagating; a rough estimation may be $A \simeq \pi(r\varphi)^2$ in considering for the ICME a cone shape where φ is the half-

angular width.

With a dependence $\rho_W \propto 1/r^2$ for the solar wind density and $A \propto r^2$ for the ICME cross-section area, all of the terms multiplying C_D in equation (4.4) may be considered a constant quantity. To complete this discussion, then, the last analysis regards the behaviour of C_D during the ICME interplanetary motion. Since the physical environment and the ICME shape and speed change during the propagation, we expect that the drag force undergoes some variations as well, and that this is indeed the most complicated matter to evaluate consistently. An attempt to establish the behaviour of C_D as a function of the heliospheric distance was first made by [Car04] through MHD simulations of the motion of magnetic flux tubes (representative of the ICMEs) through a magnetized plasma (representative of the solar wind). His numerical results predicted that the drag coefficient is approximately constant and of the order of unity during the propagation of fast, heavy and dense ICMEs, which usually are those of interest for space weather effects. Different results were found for lighter and less dense CMEs, where the simulated drag coefficient appears to be an increasing function of the heliospheric distance with values $C_D \simeq 10$. Following Cargill's results, a constant value $C_D = 1$ is the assumption usually adopted for the majority of applications employing the drag-based model (see also [VZ]). In the following we will reconsider the values for the drag coefficient C_D according to the outcome from fluid dynamics simulations of the flux over a supposed shape of the CME.

To conclude, when assuming $C_D = 1$, an estimation of typical γ values is readily made by considering $M \approx 10^{12} \div 10^{13}$ kg for the mass and a typical limb-CME half angular width of $\varphi = 30°$, obtaining at a distance $r = 20\,R_\odot$ a cross-section $A = \pi(r\varphi)^2 \approx 10^{20}\mathrm{m}^2$. Then, employing a solar wind density $n \approx 10^9\,\mathrm{m}^{-3}$ at this distance (as predicted by equation (4.6) with a typical particle composition $[H] \approx 96\%$, $[He] \approx 4\%$), one obtains for $\gamma \approx 0.2 \div 2 \times 10^{-7}$ km^{-1}. Notice that the drag efficiency is therefore higher for lighter ICMEs.

4.3.1 The Drag-Based Model with constant solar wind and drag parameter

Under the assumption of constant solar wind and drag parameter motivated on physical basis in the previous section, the ordinary differential equation (4.3) allows for an analytic solution for r and v as a function of time. It is nothing more than a calculus exercise to show that such a solution is:

$$r(t) = \pm\frac{1}{\gamma}\ln\Big[1 \pm \gamma(v_0 - w)t\Big] + wt + r_0 \tag{4.7}$$

$$v(t) = \frac{v_0 - w}{1 \pm \gamma(v_0 - w)t} + w \tag{4.8}$$

As expected, the solution for v has tends to adjust to the solar wind speed, with a faster rate for larger values of the drag parameter. For examples of solutions

refer to [Vrs+12].

4.4 Evaluation of the ICME fluid dynamic regime

A consistent evaluation of typical Reynolds' numbers for an ICME moving in the solar wind can be performed appealing to fluid dynamics and macroscopic plasma arguments. Our interest in evaluating this quantity resides in establishing which expected makes it possible to compare the results from our simulations with experimental values.

We start by defining the Reynolds number for an ICME, seen as a solid obstacle placed in a uniform flow in the most natural way, that is by directly re-adapting the general definition of Reynolds number presented in equation (3.4):

$$Re = \frac{\rho_{SW}\, v_{CME}\, L_{CME}}{\mu_p} = \frac{v_{CME}\, L_{CME}}{\nu_p} \tag{4.9}$$

where L_{CME} is now a characteristic length for the CME as seen from the flow, v_{CME} is its relative speed to the solar wind, μ_p is the plasma dynamic viscosity coefficient and $\nu_p = \mu_p/\rho_{SW}$ is the plasma kinematic viscosity. In this equation the solar wind viscosity coefficient μ_p is indeed the most delicate parameter to be evaluated, since it is a macroscopic quantity collecting several processes occurring into the plasma. We remember that the viscosity in a fluid is a measure of the macroscopic force per unit area that a fluid layer produces on another. It is well-known that the solar wind is a collisionless plasma above $\simeq 10R_{\odot}$, meaning that the Coulomb collision mean free path in the solar wind ($\simeq 1AU$) is longer than the macroscopic scale lengths. Despite of this, a fluid description of the solar wind is well accepted since a macroscopic, coherent behaviour on scales of our interest can still be retrieved by considering processes other than particle-particle Coulomb collisions, basing on appropriate alternative mechanisms to be adequately evaluated.

For our purpose of providing an estimation of ν_p in equation (4.9), we will refer to the theory of collisionless viscosity proposed by [PBK96], where the viscosity in the plasma is retrieved by considering the dynamic effect of a flux of protons originating from one plasma layer and impinging on another. These protons interact both with other protons and with tangled magnetic fields embedded in the flow, the latter being the actual interaction causing them to scatter. Since this model contains a lot of precise details, from the distribution of tangled magnetic field starting from statistical topics, and the consequent averaging procedures to be taken to define the macroscopic plasma quantities, we must refer to the books [MW99], [CT91] and the paper [PBK96] for the full theoretical treatment of this topic.

Basing on this approach, in [SLB12] they provide the following relation for

the plasma dynamic viscosity:

$$\mu_p = \frac{2}{15}\sqrt{6}\rho_p v_{rms}\lambda \tag{4.10}$$

where ρ_p is the proton mass density, $v_{rms} = \frac{3k_B T_p}{m_p}$ is the proton root mean square speed, and λ is the effective mean free path relative to collisions between the protons and the magnetic kinks, considered to be equal to the coherence length of the magnetic field irregularities (the shortest length scale over which the magnetic field has a defined structure) and evaluated, basing on the work [CH89], as

$$\lambda = \frac{684}{\sqrt{n_e}}\,\mathrm{km} \tag{4.11}$$

Substituting the previous laws we find that the general trend for ν_p is that of a linearly increasing function with the heliocentric distance; by considering for the solar wind a proton temperature $T_p = 10^5$K, over typical ICME propagation distances this relation predicts values for ν_p ranging from $200\,\mathrm{km}^2/\mathrm{s}$ to a value of $\nu \approx 800\,\mathrm{km}^2/\mathrm{s}$ near the Earth. By considering scale laws with the heliocentric distance for all of the quantities involved in equation (4.9), the Reynolds number, and therefore the drag coefficient, is a slowly varying function with the r and may be considered to a first approximation a constant value during the ICME propagation.

At this point, it is possible to consistently evaluate some typical values of the Reynolds number for an ICME propagating in the solar wind. Substituting in equation (4.9) a general value halfway Sun-Earth for the plasma viscosity of $\nu_p \simeq 600\mathrm{km}^2/\mathrm{s}$ and $v_{CME} \simeq 500\mathrm{km/s}$, $L_{CME} \simeq 0.2AU$ for the ICME, this relation generally predicts values $Re > 10^6$, in accordance with [SLB12].

This result limits the validity of our simulations, for high values of the Reynolds number require to take into account turbulence models to correctly compute drag forces, as discussed in section 3.2.1. In the following we will then proceed by considering the drag coefficient C_D for the highest values for Re allowed by our simulations, which we may consider to be the best data for high Re at our disposal. Table 4.1 finally presents the computed drag parameter on several shapes representative of the ICME section in the ecliptic plane, evaluated according to the considerations above and our two-dimensional simulations of the drag force.

The order of magnitude of these values for γ_c agrees with the estimated range of values for γ. By the way, it is worth noting that equation (4.4) also involves the ICME mass, for which we are forced to consider an interval of values rather than precise mass evaluations, which would be required for each case to complete the picture of consistent experimental evaluations of the drag parameter. With regard to this point, Gopalswamy ([Gop+09], [Gop+12] and references therein) proposed interesting models for the estimation of ICME mass based

on observations have been proposed and will be successively integrated in this study in order to introduce a fully consistent, observational-based evaluation of the drag parameter to include in our forecasting models.

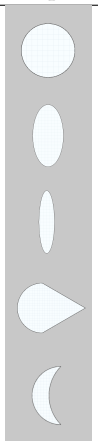
#	Shape	C_d^F	C_d^B	γ^F ($\times 10^{-7}$km^{-1})	γ^B ($\times 10^{-7}$km^{-1})
1		1.3	-	$0.25 \div 2.5$	-
2		2.0	-	$0.4 \div 4$	-
3		1.7	-	$0.35 \div 3.5$	-
4		1.0	1.5	$0.2 \div 2$	$0.3 \div 3$
5		2.2	3.6	$0.45 \div 4.5$	$0.7 \div 7$

Table 4.1: Drag coefficient and drag parameter at high Reynolds number found from numerical simulations for the listed two-dimensional shapes.

4.5 Comparison with experimental values

4.5.1 w and γ from model inversion

In order to compare some of our assumptions about the behaviour of the drag parameter with experimental data, it is necessary the availability of a sample of ICME for which several characteristics quantities, such as those needed to compute the drag parameter, are known. As discussed in the previous section, observational data from remote observations are affected by several sources of errors and lack of proper measuring procedures. To get an estimation of the drag parameter for a sample of ICME events, we proceed with the method proposed by [Vrs+12]. This is an interesting approach which also allows obtaining observational statistics that may be included in probabilistic approaches to forecasting via the Drag-Based Model (see, e.g. [Nap+18], [Dum+18]).

The method works as follows: rather than employing the Drag-Based model directly to estimate time of arrival and impact velocity at the target distance, the drag-based model solution can be employed, when travel time and impact velocity are known, to leave the solar wind speed and the drag parameter as unknown values. According to [Vrs+12], it is not possible to analytically inverse equations (4.7) and (4.8); for the drag parameter it simply holds that:

$$\pm\,\gamma = \frac{v_0 - v_d}{(v_0 - w)(v_d - w)T} \tag{4.12}$$

where the plus or minus sign has to be chosen depending on $v_0 > w$ or $v_0 < w$, respectively. This equation requires the solar wind speed, which can be computed by employing numerical methods to solve the implicit algebraic equation in w:

$$\frac{(v_d - w)(v_0 - w)\,T}{v_0 - v_1} \ln\left[\frac{v_0 - v_1}{v_1 - w} + 1\right] + wT + r_0 - r_1 = 0 \tag{4.13}$$

As explained in section 4.1, observations of early phases of a CME launch from the Sun and successive recording of their in-situ signature involve different methods and observable quantities. Bearing in mind that few methods exist for ICME en-route tracking, to proceed with the inversion method presented above, we need a sample of events for which a safe and affordable association between remote observations of their origin at the Sun and their consequent arrival at Earth.

For this purpose, we selected and combined information from two databases:

- The CACTUS database, a system autonomously detecting Coronal Mass Ejections in image sequences from LASCO (see [Rob04]). Quantities of our interest provided by this catalog are the CME initial time and the projected velocity of the CME in the coronagraph FOV. Extensive details about these quantities, their precise definitions and measurement methods can be found online at the CACTUS webpage (`http://sidc.oma.be/cactus`).

- The list of ICME compiled by I. Richardson and H. Cane, which reports the occurrence of Interplanetary Coronal Mass ejections in the near-Earth solar wind based on ACE and WIND data. Quantities of our interest provided by this catalogue are the estimated start time of the ICME based on plasma and magnetic field observations, the mean ICME plasma speed during its passage, the maximum solar wind speed during the period from the occurring of the first disturbance (shock or geomagnetic effects) to the trailing edge of the ICME.

  Extensive details about these quantities, their precise definitions and measure methods can be found online at the Richardson and Cane ICME catalog (`http://www.srl.caltech.edu/ACE/ASC/DATA/level3/icmetable2.htm`).

The final event list we compiled is reported in Appendix C and contains the following data:

- CME starting date
- CME initial speed v_0
- ICME arrival date
- ICME plasma mean velocity v_1 at 1AU.

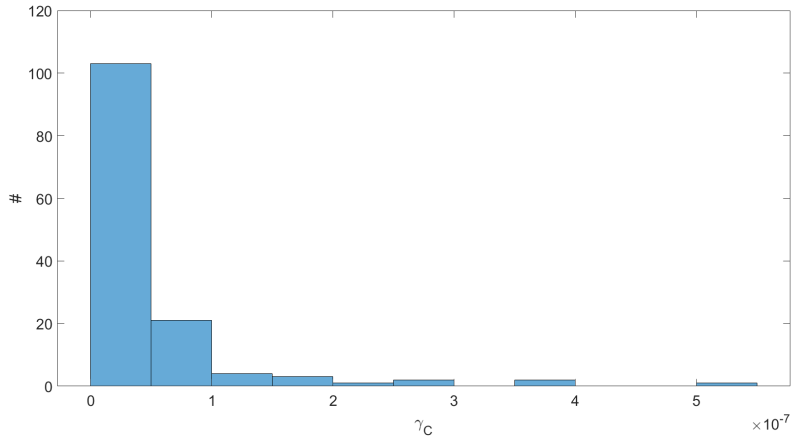


Figure 4.3: Histogram of the values for the drag parameter obtained from DBM inversion.

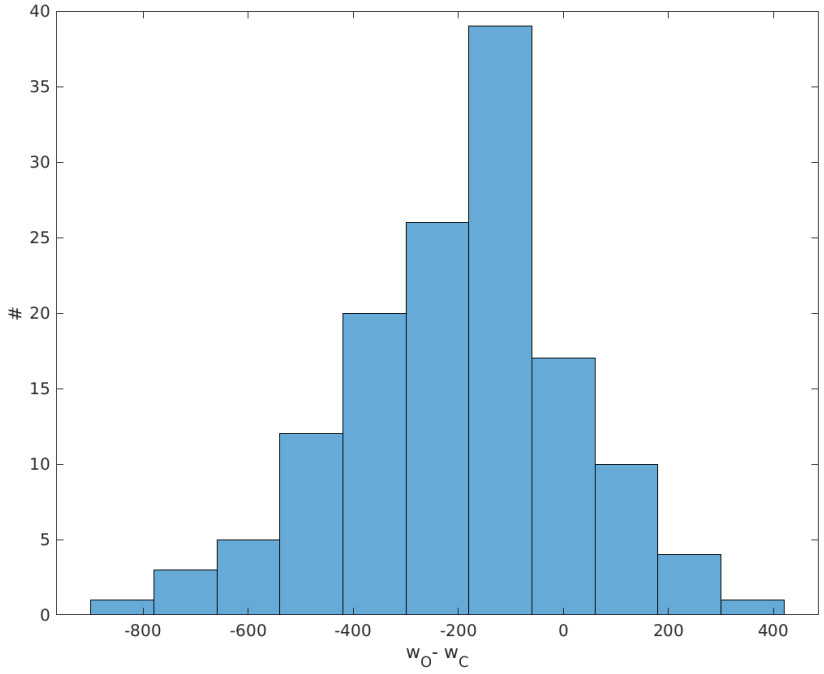


Figure 4.4: Histogram of the difference between the observed solar wind speed and the DBM-computed solar wind speed for our sample of events.

Figure **??** shows the distribution of values for the drag parameter γ obtained from the Drag-Based Model inversion technique applied to these data.

Richardson and Cane's list also provides the observed value of the solar wind speed w_O during the ICME passage. Knowledge of this quantity allows for a comparison of the solar wind speed obtained from the inversion procedure with the solar wind speed actually measured at Earth. Figure 4.4 shows the distribution of values for the difference between the solar wind speed obtained from the Drag-Based Model inversion w_C and the observed value of the solar wind speed w_O during the ICME passage. Mean value and standard deviation of this distribution are $< \Delta w >= -202\,\mathrm{km/s}$ and $\sigma(\Delta w) = 215\,\mathrm{km/s}$ respectively, indicating that the Drag-based model inversion, although providing values comparable with typical solar wind speeds, tends to overestimate the actual solar wind speed of about 200 km/s.

In order to compare these results with our hypothesis that an asymmetrically shaped CME would lead to a different value of the drag parameter, depending on the acceleration or deceleration regime of the ICME, we plotted in Figure 4.5 the computed values of the drag parameter γ_c versus the ICME initial relative speed to the solar wind $(v_0 - w_c)$. In this plot, points for which $w_c - v_0 < 0$ correspond to ICMEs pushed from ahead, i.e. braked ICMEs, while points for which $w_c - v_0 > 0$ correspond to ICMEs pushed from behind by the solar wind. In case the ICME behaves as an asymmetric solid body in the solar wind, a lower drag parameter must be expected in the first case in comparison to the second.

Looking at the plot in Figure 4.5 it is necessary to pinpoint that the net increase of the values of γ_c near $(v_0 - w) \to 0$ is a mathematical artefact due to the presence of the denominator term $(v_0 - w)$ in equation (4.12), which is not precisely compensated by the other terms because of experimental errors, error propagation during the inversion procedure and intrinsic limits of the Drag-Based Model when representing experimental data under these circumstances. In fact, comparing with equation (4.3), setting the initial condition $v_0 = w$ the drag force vanishes, and for this particular case the drag parameter may take any value, going to infinity when approaching $|v_0 - w| \to 0$ in a limit procedure.

To proceed, we then selected a subset of the obtained γ_c values by employing the results from the evaluation of the solar wind speed distribution, employed to define a cutoff value on the relative velocity, where we consider that only data from the drag-based model inversion for which $|v_0 - w| > \sigma_W$ are to be considered from the Drag-Based Model inversion. This relies on the fact that the standard deviation of the $w_o - w_c$ distribution is taken as a measure of the error that the Drag-Based Model inversion procedure commits when employed to compute relative speeds. We reasonably assume that an error of the same order of magnitude holds for the quantity $v_0 - w_c$, defining our cutoff criterion that excludes differences in relative speed below this interval of values. Our

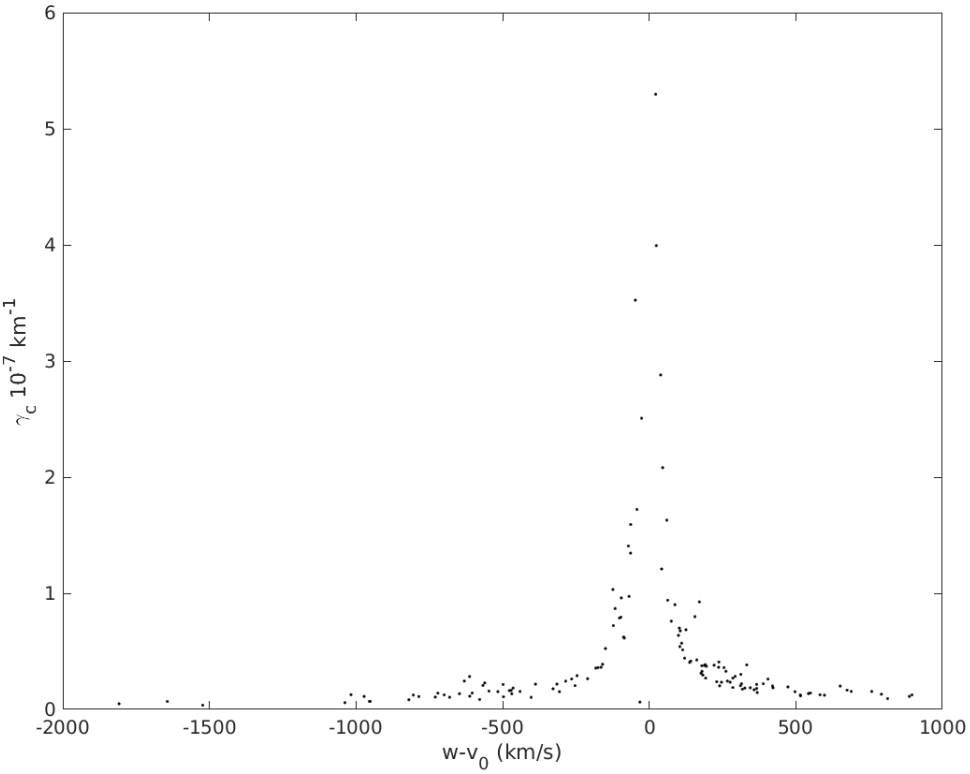


Figure 4.5: Computed values for the drag parameter γ_c versus the ICME initial relative speed to the solar wind obtained through DBM equations inversion.

results for the drag parameter in the two regimes are:

$$\gamma^F = (0.15 \pm 0.06) \times 10^{-7}\,\mathrm{Km}^{-1} \quad \text{(Forward wind)}$$
$$\gamma^B = (0.21 \pm 0.08) \times 10^{-7}\,\mathrm{Km}^{-1} \quad \text{(Backward wind)}$$

As can be seen, the values obtained for the drag parameter in the two regimes are barely comparable within the experimental errors and apparently consistent with the values of the drag parameter computed for shape 4 (cone model). Because of the errors in the estimation of these parameters, arising both from uncertainties in the association of remote to in-situ events and to measurement errors on all of the quantities involved in the inversion procedure, this result cannot be considered conclusive in supporting our hypothesis that a backward wind produces a higher drag on the ICME structure respect to a forward wind, even though the difference found in the mean values of the drag parameter may be considered a possible first indication of an actual asymmetric behaviour of the ICME. Because of such a result, we believe that this investigation certainly deserves further studies, which may lead to a better definition of the drag parameter and refinement of the Drag-Based Model.

Conclusions and future work

The realisation of forecasting models in space weather is a rather difficult task because of the complexity underlying all of the physical processes involved in the interplanetary phenomena, which are still poorly understood. Effective forecasting models as the Drag-Based require the definition and tuning of several parameters describing in an average manner such detailed physics, and their definition must rely on reasonable assumption and empirical considerations for which it is fundamental to proceed with experimental evaluations and comparisons.

Throughout this work, we aimed at the refinement of a specific parameter of a forecasting model. We employed numerical techniques to evaluate the dynamics of a Coronal Mass Ejection, modeled as a solid body moving in a fluid environment, and proposed the hypothesis that in case of an asymmetric behaviour and shape of the body, the evaluation of the drag parameter can be refined by considering the regime of motion relative to the ambient fluid, representative of the solar wind.

Our results suggest an agreement with this hypothesis. We want to point out, though, that our method of validation required data with a high grade of reliability, obtained after a chain of procedures that undoubtedly affected the quality and quantity of final usable data. For this reason, we intend to reconsider the statistical results and improve the quality of the input data to this study by employing a larger sample of events, to be obtained through a refinement of the methods we employed for the evaluation and association of the remotely observable quantities with in-situ data.

This study also allows for a wide range of possible improvements by reconsidering the numerical methods employed. More sophisticated and efficient numerical fluid dynamics simulations can be employed to further improve the precision on our estimation of the drag coefficient, by extending the number of possible ICME shapes, by considering the three-dimensional cases, and simulating in a non-uniform solar wind density and velocity distribution. Numerical simulations of turbulence models may allow for a more accurate evaluation of the drag coefficient in the range of very high Reynolds number, that must be included when considering structures moving in the solar wind.

Concerning future studies about this forecasting method, we intend to include a consistent evaluation of the ICME mass based on remote observations, to perfect the value of the drag parameter to introduce in the forecasting. As discussed in section 4.5, evaluating the ICME mass is neither simple nor unambiguous, and thus proper methods, based on the most recent research about this subject, have to be included.

We observed that when employing the Drag-Based Model, the drag parameter is only one of the quantities involved in the forecasting procedure. The solar wind speed is also a parameter of great relevance in the forecasting results. We

believe that is also necessary a dedicated study, analogous to the one performed for the drag parameter and aimed to establish proper evaluation procedures for this quantity, in order to improve the overall forecasting efficiency through this method.

Appendix A

C_d vs Re profiles

This appendix is a reference collecting plots of the drag coefficient versus Reynolds number for the various shapes employed to compile table 4.4. Dots are representative of the simulated values.

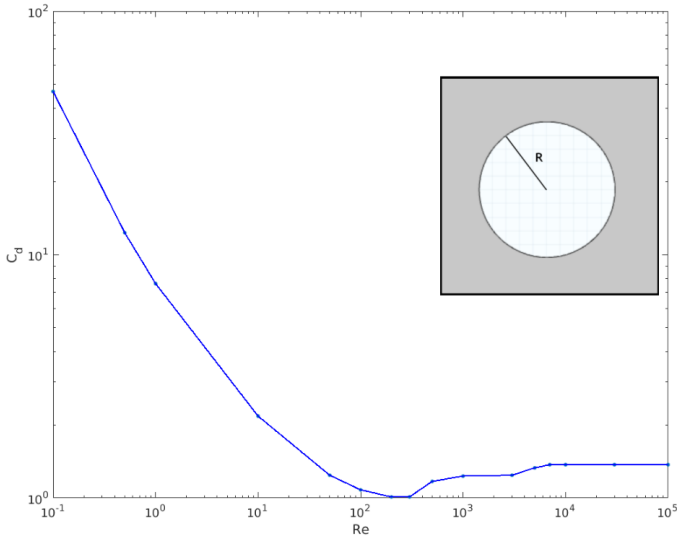


Figure A.1: C_d vs Re profile for the cylinder (shape 1).

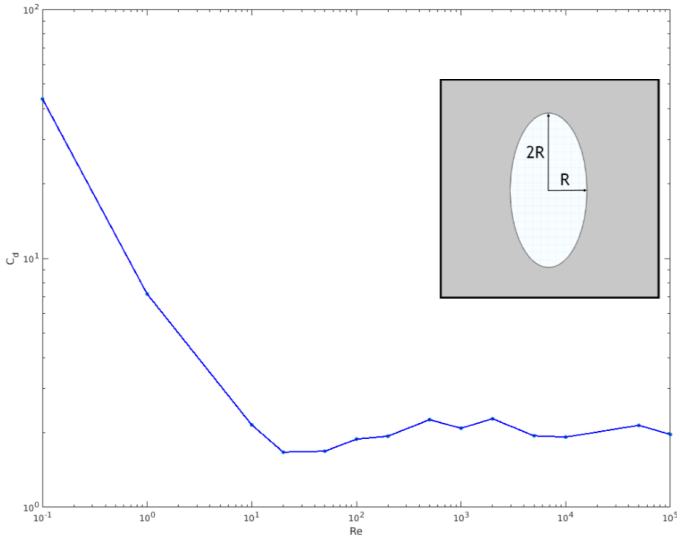


Figure A.2: C_d vs Re profile for the the helix model with eccentricity 0.5 (shape 2).

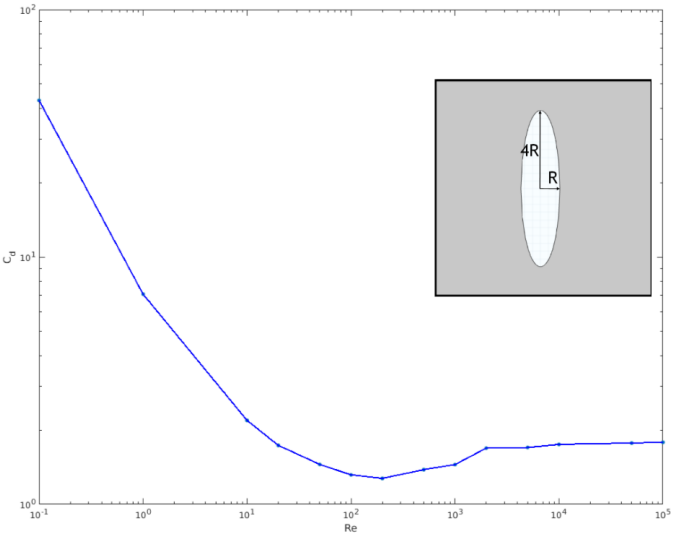


Figure A.3: C_d vs Re profile for the the helix model with eccentricity 0.25 (shape 3).

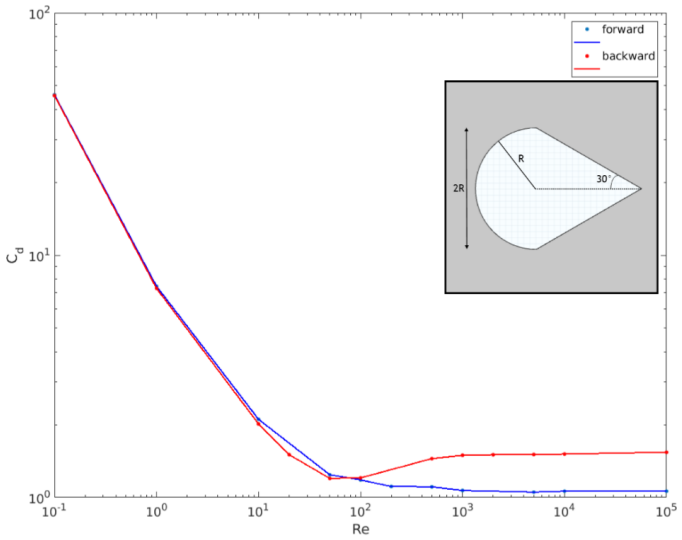


Figure A.4: C_d vs Re profile for the Cone Model (shape 4).

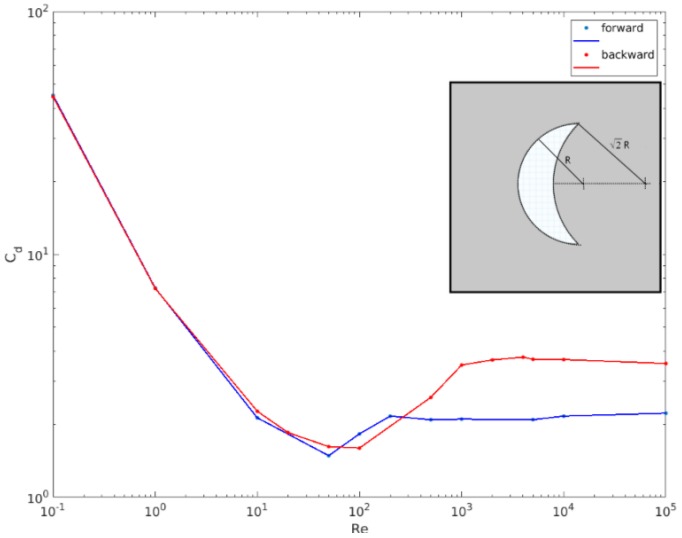


Figure A.5: C_d vs Re profile for the shield model (shape 5).

Appendix B

MATLAB codes

B.1 Functions for numerical differentiation

B.1.1 First derivatives

Periodic boundary

```
% Numerical first derivative of a function on x periodic domain
% Reference: Lele 1992

function df = der1_per(x,f)
%% Domain and matrices definitions
nx = length(x);
dx = (x(end)-x(1))/(nx-1);
A = zeros(nx);
B = zeros(nx,1);

alpha = 1/3;
a = 2/3*(alpha+2);
b = 1/3*(4*alpha-1);

% boundary equations i = 1, nx and 2, nx-1
%(because of b != 0,  terms f_i-2 and f_i+2 are involved)
B(1)= a*(f(2)-f(nx-1))/(2*dx) + b*(f(3)-f(nx-2))/(4*dx);
B(nx) = B(1);
B(2)= a*(f(3)-f(1))/(2*dx) + b*(f(4)-f(nx-1))/(4*dx);

B(nx-1)= a*(f(1)-f(nx-2))/(2*dx) + b*(f(2)-f(nx-3))/(4*dx);
% internal equations i = 3 to nx-2
B(3:nx-2,1) = a*(f(4:nx-1)-f(2:nx-3))/(2*dx) + b*(f(5:nx)-f(1:nx-4))/(4*dx);

    %% Constructing tridiagonal matrix
    % boundary rows
```

```
    A(1,1:2) = [1 alpha];   A(1,nx) = alpha;
    A(nx,nx-1:nx) = [alpha 1]; A(nx,1) = alpha;
    for i = 2:nx-1
            A(i,i-1:i+1)=[alpha 1 alpha];
    end
    %% Solving tridiagonal matrix
    df = A\B;
    df = df';
end
```

Non periodic boundary

```
% Numerical first derivative of a function on x domain
% with boundary point treatment
% Reference: Lele 1992

function df = der1_BC(x,f)

    %% Domain and matrices definitions
    nx = length(x);
    dx = (x(end)-x(1))/(nx-1);
    A = zeros(nx);
    B = zeros(nx,1);

    % internal points from 3 to nx-2
    alpha_int = 1/3;
    a = 2/3*(alpha_int+2);
    b = 1/3*(4*alpha_int-1);
    B(3:nx-2,1) = a*(f(4:nx-1)-f(2:nx-3))/(2.0*dx) + b*(f(5:nx)-f(1:nx-4))/(4*dx);

    % quasi-boundary points 2 and nx-1
    alpha_qb = 1/4;
    a = 6*alpha_qb;
    B(2) = a*(f(3)-f(1))/(2.0*dx);
    B(nx-1) = a*(f(nx)-f(nx-2))/(2.0*dx);

    % boundary points 1 and nx
    alpha_b = 3;
    a = -(11+2*alpha_b)/6;
    b = (6-alpha_b)/2;
    c = (2*alpha_b-3)/2;
    d = (2-alpha_b)/6;
    B(1) = 1/dx*(a*f(1)+b*f(2)+c*f(3)+d*f(4));
    B(nx) = -1/dx*(a*f(nx)+b*f(nx-1)+c*f(nx-2)+d*f(nx-3));

    %% Solving tridiagonal matrix

    % boundary equations
    A(1,1:2) = [1 alpha_b];
    A(nx,nx-1:nx) = [alpha_b 1];
```

```
    % quasi boundary
    A(2,1:3)=[alpha_qb 1 alpha_qb];
    A(nx-1, nx-2:nx)=[alpha_qb 1 alpha_qb];
    % internal points
    for i = 3:nx-2
        A(i,i-1:i+1)=[alpha_int 1 alpha_int];
    end

    df = A\B;  % solve the system A*df = B
    df = df';

end
```

B.2 Filtering functions

```
function A = init_filter_matrix(alpha_f,nx)
% initialization of filtering matrices

% boundary equations
    A(1,1) = 1;
    A(2,2) = 1;
    A(3,3) = 1;
    A(nx,nx) = 1;
    A(nx-1,nx-1) = 1;
    A(nx-2,nx-2) = 1;

% internal points
    for i = 4:nx-3
            A(i,i-1:i+1)=[alpha_f 1 alpha_f];
    end
    A = sparse(A);
end


% Filtering out high frequencies from the function
% Reference Lele 1992 eq. C.2.5 with beta = 0
% (requires init_filter_matrix.m to be executed first)

function f_filt = filter_BC1d_call(f,alpha_f)
global A
nx = length(f);
B = zeros(nx,1);

% VI order filtering scheme
% internal points coefficients
a = (11+10*alpha_f)/16;
b = (15+34*alpha_f)/32;
c = (-3+6*alpha_f)/16;
d = (1-2*alpha_f)/32;
```

```
B(4:nx-3) = a*f(4:nx-3)+ ...
            b/2*( f(5:nx-2)+f(3:nx-4) )+ ...
            c/2*( f(6:nx-1)+f(2:nx-5) ) + ...
            d/2*( f(7:nx)+f(1:nx-6) );

% LHS boundary points
B(1) = 15/16*f(1) + 1/16*(4*f(2) - 6*f(3) + 4*f(4) - f(5));
B(2) = 3/4*f(2) + 1/16*(f(1)+6*f(3)-4*f(4)+f(5));
B(3) = 5/8*f(3) + 1/16*(-f(1)+4*f(2)+4*f(4)-f(5));
% RHS boundary points
% no sign change (as in the derivative)?
B(nx-2) = 5/8*f(nx-2) + 1/16*(-f(nx)+4*f(nx-1)+4*f(nx-3)-f(nx-4));
B(nx-1) = 3/4*f(nx-1) + 1/16*(f(nx)+6*f(nx-2)-4*f(nx-3)+f(nx-4));
B(nx) = 15/16*f(nx) + 1/16*(4*f(nx-1)-6*f(nx-2)+4*f(nx-3)-f(nx-4));

%% Solving tridiagonal matrix
    f_filt = A\B;       % solve system A*f_filt = B
    f_filt = f_filt';
end
```

B.3 1D Euler equations

B.3.1 Main section

```
% 1D Euler equations on open domain
% main file

%% Initialization and options

% Global variables
global dx dt nx gamma rk_ord
global M alpha_filter err_flag Filter
global i_filter ndt_filter filter_type

% Run continuation options
Continue = false;
t_max_new = % new final time

% Filtering options
Filter = true;
ndt_filter = 4;     % filter every ndt_filter time steps
filter_type = 2;    % Filter type selector
                    % 1 - Nonlinear filter from [Engquist et al. 1988]
                    % 2 - Lele 1992
alpha_filter = 0.4; % alpha filter parameter in Lele 1992 eq.(C.2.5)
                    % keep below 0.5, for filter_type = 2 only.
% Parameters
gamma = 5/3;
```

```
M = 1;              % particle mass

% Space time domain definition
Lx = % box length
nx = % number of spatial points
dt = % time step
t_max = % final time

dx = Lx / (nx-1);
x = (0:nx-1)*dx;

% Time advancement options
n_dt_out = 1;       % output the solution every n_dt_out time steps
rk_ord = 4;         % Runge-Kutta scheme order

%% Initial conditions
% Initial conditions must be provided through an external function
% condinit(~) giving [Sn, Den, U_x, P, T] as output

    if Continue == false
    [Sn, Den, U_x, P, T] = condinit(~);
    % output variables initialization
    Sn_n = Sn;
    Den_n = Den;
    U_x_n = U_x;
    P_n = P;
    T_n = T;

    % Run continuation case
    % starts from the variables at the end of the previous simulation
    elseif Continue == true
        if t_max_new <= t_max
            fprintf('New final time <= to previous one!')
        end
    Sn = Sn_n(:,n_out);
    Den = Den_n(:,n_out);
    U_x = U_x_n(:,n_out);
    P = P_n(:,n_out);
    T = T_n(:,n_out);
    end

    % CFL condition evaluation and stabiliy WARNING at initial time
    S_max = 0;
    for i = 1:nx
        S = sqrt(gamma*P(i)/Den(i)) + abs(U_x_n(i)); % S = c + |u|
        if S > S_max
            S_max = S;
        end
    end
    sigma = 0.15;    % empirically evaluated, < 0.5
    if dt > 0.15*dx/S_max
```

```
        fprintf('WARNING: CFL condition is not satisfied \n')
        fprintf('numerical instability is expected! \n')
        fprintf('Suggested dt <= %s \n',num2str(0.5*dx/S_max))
    end

%% Time advancement
tic
    % counters initialization
if Continue == false
    t = 0;
    i_out = 0;
    i_filter = 0;
    n_out = 1;  % outputs counter
    n = 0;      % total outputs counter
elseif Continue == true
t_max = t_max_new;
end

    % MAIN LOOP
    err_flag = 0;       % error flag
    while t < t_max
        n = n + 1;
        fprintf('step number %s, t = %s -> %s \n',...
            num2str(n), num2str(t), num2str(t+dt))

        % time advancement
        [Sn, Den, U_x, P, T] = advance_Eul1d_NRBC(Sn,Den,U_x,P,T);

        % filtering counter.
        i_filter = i_filter + 1;
        t = t + dt;
        if i_filter == ndt_filter + 1
            i_filter = 0;   % reset filtering counter
        end

        % output counter
        i_out = i_out + 1;
        if i_out == n_dt_out
            i_out = 0;
            n_out = n_out + 1;
            if err_flag == 0
                Sn_n(:,n_out) = Sn;
                Den_n(:,n_out) = Den;
                U_x_n(:,n_out) = U_x;
                P_n(:,n_out) = P;
                T_n(:,n_out) = T;
            elseif err_flag == 1
    fprintf('Exiting numerical simulation due to imaginary result \n')
    fprintf('for solution at time %s \n',num2str(t))
                break
            end
```

```
        end
    end

    if err_flag == 0
        fprintf('Simulation completed! \n')
        fprintf('Number of iterations: %s \n', num2str(n))
        fprintf('Numer of outputs: %s \n', num2str(n_out))
        fprintf('Elapsed time: %s s \n', num2str(toc))
    end

%% Output variables storage
 savefile = 'output.mat';
 save(savefile,'x','dx','t','dt','n_dt_out','U_x_n','Den_n','P_n','T_n');
%% Note
% Output solution variables are in the form V_n(i,nt)
% time t is related to time step index nt through
% t(nt) = (nt-1)*dt*n_dt_out
```

B.3.2 Time advancement function

```
function [Sn_n1, Den_n1, U_x_n1, P_n1, T_n1] = ...
    advance_Eul1d_NRBC(Sn_n, Den_n, U_x_n, P_n, T_n)
% This function produces numerical solution to 1D Euler equations
% at time t + dt for the variables Sn, Den, U_x, P, T
% (subscript _n1) starting from those at time t (subscript _n)
%
% Time advancement is performed through a Runge-Kutta of order rk_ord.
% Details can be found in Canuto et al., "Spectral methods
% in fluid dynamics" I ed. page 109, and Jameson, Schmidt, Turkel (1981)
%
% RK scheme is the following:
%   U = U_n
%   for i = rk_ord:1:-1
%       U = U_n + 1/i * dt * F(U)    (*)
%   end
%   U_n+1 = U  (**)

% Load global variables
global gamma rk_ord nx dx dt
global M err_flag
global i_filter ndt_filter Filter alpha_filter filter_type

x = (0:nx-1)*dx;
% amplitude of characteristic waves inizialization
L0 = zeros(2,1);
Lp = zeros(2,1);
Lm = zeros(2,1);

%% RK time advancement routine
for i = rk_ord:-1:1
```

```
%% 0. Calculating system characteristic variables

    cl = sqrt(gamma*P_n(1)/Den_n(1));
    cr = sqrt(gamma*P_n(nx)/Den_n(nx));
    dPdx = der1_BC(x,P_n);
    dDendx = der1_BC(x,Den_n);
    dUdx = der1_BC(x,U_x_n);
    % left boundary
    if U_x_n(1) >= 0
        L0(1) = 0;
    else
        L0(1) = U_x_n(1)*(dDendx(1)-1/cl^2*dPdx(1));
    end
    if U_x_n(1)+cl >= 0
        Lp(1) = 0;
    else
        Lp(1) = (U_x_n(1)+cl)*(dPdx(1)+cl*Den_n(1)*dUdx(1));
    end
    if U_x_n(1)-cl >= 0
        Lm(1) = 0;
    else
        Lm(1) = (U_x_n(1)-cl)*(dPdx(1)-cl*Den_n(1)*dUdx(1));
    end
    % right boundary
    if U_x_n(nx) <= 0
        L0(2) = 0;
    else
        L0(2) = U_x_n(nx)*(dDendx(nx)-1/cr^2*dPdx(nx));
    end
    if U_x_n(nx)+cr <= 0
        Lp(2) = 0;
    else
        Lp(2) = (U_x_n(nx)+cr)*(dPdx(nx)+cr*Den_n(nx)*dUdx(nx));
    end
    if U_x_n(nx)-cr <= 0
        Lm(2) = 0;
    else
        Lm(2) = (U_x_n(nx)-cr)*(dPdx(nx)-cr*Den_n(nx)*dUdx(nx));
    end

%% 1. Adiabatic equation advancement
    SnU_n = - Sn_n .* U_x_n;
    RHS = der1_BC(x,SnU_n);

        % RHS of entropy equation
        RHS(1) = Den_n(1)^(1-gamma)*...
            ((Lp(1)+Lm(1))/gamma+(1-gamma)/gamma*cl^2*L0(1));
        RHS(1) = - RHS(1);
        RHS(nx) = Den_n(nx)^(1-gamma)*...
            ((Lp(2)+Lm(2))/gamma+(1-gamma)/gamma*cr^2*L0(2));
```

```
        RHS(nx) = - RHS(nx);

    % time advancement
    Sn_n1 = Sn_n + RHS' * dt / i;

%% 2. Continuity equation advancement

    nU_n = Den_n .* U_x_n;
    RHS = - der1_BC(x,nU_n);

        % RHS of continuity equation
        RHS(1) = L0(1)+ (Lp(1)+Lm(1))/cl^2;
        RHS(1) = - RHS(1);
        RHS(nx) = L0(2)+ (Lp(2)+Lm(2))/cr^2;
        RHS(nx) = - RHS(nx);

    % time advancement
    Den_n1 = Den_n + RHS' * dt /i;

    % checking density to be positive,
    % otherwise imaginary terms are produced while computing
    % P_n1 = Sn_n1 ./ (Den_n1.^(1-gamma));
    for k = 1:nx
        if Den_n1(k) <= 0
            err_flag = 1;
            break
        end
    end

%% 3. Momentum equation advancement

    %  x component of momentum equaiton in conservative form
        % RHS = - [ @x(rho * U_x * U_x) + @p/@x ]

        % x derivative
        Aij = - Den_n .* (U_x_n .* U_x_n) - P_n/M;  % Axx
        RHS = der1_BC(x, Aij);

        % RHS of momentum equation
        RHS(1) = U_x_n(1)*(L0(1)+(Lp(1)+Lm(1))/cl^2)+(Lp(1)-Lm(1))/cl;
        RHS(1) = - RHS(1);
        RHS(nx) = U_x_n(nx)*(L0(2)+(Lp(2)+Lm(2))/cr^2)+(Lp(2)-Lm(2))/cr;
        RHS(nx) = - RHS(nx);

    % time advancement
    nU_x_n = Den_n .* U_x_n;
    nU_x_n1 = nU_x_n + RHS' * dt /i;

    %% 4. Filtering

    if Filter == true && i_filter == ndt_filter
```

```
        if filter_type == 2
            Sn_n1 = filter_BC(Sn_n1,alpha_filter)';
            Den_n1 = filter_BC(Den_n1,alpha_filter)';
            nU_x_n1 = filter_BC(nU_x_n1,alpha_filter)';
        elseif filter_type == 1
            Sn_n1 = filter_NL1d(Sn_n1);
            Den_n1 = filter_NL1d(Den_n1);
            nU_x_n1 = filter_NL1d(nU_x_n1);
        end
    end

    %% 5. Indirect quantities
    % quantities not directly featuring under time derivative

    U_x_n1 = nU_x_n1./Den_n1;
    P_n1 = Sn_n1 ./ (Den_n1.^(1-gamma));
    T_n1 = P_n1 ./ Den_n1;    % ideal gas EOS
    % Note: divide by kB*M to convert it to Kelvin.

    %% 6. Updating input quantities for the next RK cycle step
    Sn_n = Sn_n1;
    Den_n = Den_n1;
    U_x_n = U_x_n1;
    P_n = P_n1;
    T_n = T_n1;
end
```

B.4 Analytic solution to the Sod shock tube problem

```
%% Analytic solution to the Sod shock tube problem

%% Initial conditions and parameters
gamma = 5/3;
% left data state
uL = 0;
rhoL = 1;
pL = 1;
% right data state
uR = 0;
rhoR = 0.125;
pR = 0.1;
% derived quantities
cL = sqrt(gamma*pL/rhoL);    % sound speed in left undisturbed region
cR = sqrt(gamma*pR/rhoR);    % sound speed in right undisturbed region
AR = 2/((gamma+1)*rhoR);     % data-dependent constant
BR = (gamma-1)/(gamma+1)*pR;% data-dependent constant

%% Resolution of implicit equation for the pressure in the star region
```

```
f_L = @(p) (2*cL)/(gamma-1)*( (p/pL)^((gamma-1)/(2*gamma)) -1 );
f_R = @(p) (p-pR)*sqrt(AR/(p+BR));
f = @(p) f_L(p) + f_R(p);
pSTAR = fzero(f,0.55);
uSTAR = 0.5*(f_R(pSTAR)-f_L(pSTAR));

%% Generation of analytical solution on a spacetime grid
% space
L = 1;      % box size
x_min = -L/2;
x_max = L/2;
nx = 260;
x = linspace(x_min,x_max,nx);
% time
t_max = 0.51;
dt = 2*0.00025;
t = 0:dt:t_max;
nt = length(t);
% speed parameters
rhoSTAR_L = rhoL*(pSTAR/pL)^(1/gamma);
cSTAR_L = sqrt(gamma*pSTAR/rhoSTAR_L);
SHL = uL - cL;
STL = uSTAR-cSTAR_L;
SR = uR + cR*((gamma+1)/(2*gamma)*pSTAR/pR+(gamma-1)/(2*gamma))^(0.5);

% solutions initialization
u = zeros(nx,nt);
rho = zeros(nx,nt);
p = zeros(nx,nt);

% solution in rarefacion wave and star region

% rarefaction fan
rho_fanL = @(z) rhoL*((2/(gamma+1)+(gamma-1)/(gamma+1)...
    /cL*(uL-z)))^(2/(gamma-1));
u_fanL = @(z) 2/(gamma+1)*(cL + 0.5*(gamma-1)*uL + z);
p_fanL = @(z) pL*(rho_fanL(z)/rhoL)^gamma;
% star region, before and after contact discontinuity
% rho in star region from rarefaction fan tail to contact discontinuity
rhoSTAR_fanL = rhoL*(pSTAR/pL)^(1/gamma);
 % rho in star region from contact discontinuity to shock
num =  rhoR*(pSTAR/pR+(gamma-1)/(gamma+1));
den = (gamma-1)/(gamma+1)*pSTAR/pR + 1 ;
rhoSTAR_shockR = num/den;
% more to the right the shock front holds the
% unperturbed initial state uR,pR,rhoR

%% calculations
for j = 1:nt
    for i = 1:nx
        Z = x(i)/t(j);
```

```
        if Z <= SHL
            rho(i,j)= rhoL;
            u(i,j) = uL;
            p(i,j) = pL;
        elseif Z >= SHL && Z <= STL
            rho(i,j) = rho_fanL(Z);
            u(i,j) = u_fanL(Z);
            p(i,j) = p_fanL(Z);
        elseif Z >= STL && Z <= uSTAR
            rho(i,j) = rhoSTAR_fanL;
            u(i,j) = uSTAR;
            p(i,j) = pSTAR;
        elseif Z > uSTAR && Z <= SR
            u(i,j) = uSTAR;
            p(i,j) = pSTAR;
            rho(i,j) = rhoSTAR_shockR;
        elseif Z >= SR
            rho(i,j) = rhoR;
            u(i,j) = uR;
            p(i,j) = pR;
        end
    end
end

%% Animation
 figure
 for j=1:nt
 subplot(2,2,1)
 plot(x+0.5*L,u(:,j))
 xlim([0,L])
 ylim([0,1.2])
 xlabel('x')
 ylabel('u')

 subplot(2,2,2)
 plot(x+0.5*L,rho(:,j))
 xlim([0,L])
 ylim([0,1.2])
 xlabel('x')
 ylabel('\rho')

 subplot(2,2,3)
 plot(x+0.5*L,p(:,j))
 xlim([0,L])
 ylim([0,1.2])
 xlabel('x')
 ylabel('p')

 pause(0.001)
 end
```

B.5 2D Euler equations

B.5.1 Main section

```
% 2D Euler equations on x-open y-periodic domain
% main file

clear all %#ok<CLALL>
%% Initialization and options

% Global variables
global dx dy dt nx ny gamma rk_ord
global M alpha_filter err_flag
global Filter i_filter ndt_filter filter_type

% Run continuation options
Continue = false;
t_max_new = 0.25;   % new final time

% Filtering options
Filter = true;
ndt_filter = 4;     % filter every ndt_filter time steps
filter_type = 2;    % Filter type selector
                    % 1 - Nonlinear filter from [Engquist et al. 1989]
alpha_filter = 0.4; % alpha filter parameter in Lele 1992 eq.(C.2.5)
                    % keep below 0.5, for filter_type = 2 only.

% Parameters
gamma = 5/3;    % adiabatic index
M = 1;          % particle mass

% Space time domain definition
Lx =    % box length in x dimension
Ly =    % box length in x dimension
nx =    % number of spatial points along x
ny =    % number of spatial points along y
dt =    % time step
t_max = % final time


dx = Lx / (nx-1);   % open direction
dy = Ly / ny;       % periodic direction

x = (0:nx-1)*dx;     % N.B. goes from 0 to Lx
y = (0:ny-1)*dy;    % N.B. goes from 0 to Ly-dy

% Time advancement options
n_dt_out = 1;   % output the solution every n_dt_out time steps
rk_ord = 4;     % Runge-Kutta scheme order
```

```
%% Initial conditions
% Initial conditions must be provided through an external function
% condinit(...) giving [Sn, Den, U_x, P, T] as output

    if Continue == false
    [Sn, Den, U_x, U_y, P, T] = condinit(~);
    % output variables initialization
    Sn_n = Sn;
    Den_n = Den;
    U_x_n = U_x;
    U_y_n = U_y;
    P_n = P;
    T_n = T;
    Umod_n = sqrt(U_x.^2+U_y.^2);

    % Run continuation case
    % starts from the variables at the end of the previous simulation
    elseif Continue == true
        if t_max_new <= t_max
            fprintf('New final time <= to previous one!')
        end
    Sn = Sn_n(:,:,n_out);
    Den = Den_n(:,:,n_out);
    U_x = U_x_n(:,:,n_out);
    U_y = U_y_n(:,:,n_out);
    P = P_n(:,:,n_out);
    T = T_n(:,:,n_out);
    end

    %% CFL condition evaluation and stability warning at initial time

    % Reference [E.F. Toro 1989]
    S_max = 0;
    for i = 1:nx
        for j = 1:ny
            S = sqrt(gamma*P(i,j)/Den(i,j)) + Umod_n(i,j); % S = c + |u|
                if S > S_max
                   S_max = S;
                end
        end
    end
if dt > 0.15*sqrt(dx^2+dy^2)/S_max
   fprintf('WARNING: CFL condition not satisfied \n')
   fprintf('numerical instability expected! \n')
   fprintf('Suggested dt <= %s \n',num2str(0.15*sqrt(dx^2+dy^2)/S_max))
end

%% Time advancement
tic
    % counters initialization
```

```
if Continue == false
    t = 0;
    i_out = 0;
    i_filter = 0;
    n_out = 1;  % outputs counter
    n = 0;      % total outputs counter
elseif Continue == true
t_max = t_max_new;
end

    % MAIN LOOP
    err_flag = 0;       % error flag
    while t < t_max
    n = n + 1;
    fprintf('step numero %s, t = %s -> %s \n',...
        num2str(n), num2str(t), num2str(t+dt))

    % time advancement
[Sn, Den, U_x, U_y, P, T]=advance_Eul2d_NRBCx(Sn, Den, U_x, U_y, P, T);

% filtering counter
    i_filter = i_filter + 1;
    t = t + dt;
    if i_filter == ndt_filter + 1
       i_filter = 0;    % reset filtering counter
    end

    % output counter
i_out = i_out + 1;
if i_out == n_dt_out
   i_out = 0;
   n_out = n_out + 1;
   if err_flag == 0
        Sn_n(:,:,n_out) = Sn;
        Den_n(:,:,n_out) = Den;
        U_x_n(:,:,n_out) = U_x;
        U_y_n(:,:,n_out) = U_y;
        P_n(:,:,n_out) = P;
        T_n(:,:,n_out) = T;
        Umod_n(:,:,n_out) = sqrt(U_x_n(:,:,n_out).^2+...
                            U_y_n(:,:,n_out).^2);
   elseif err_flag == 1
        fprintf('Exiting numerical simulation due to imaginary result \n')
        fprintf('for solution at time %s \n',num2str(t))
        break
   end
end
end

    if err_flag == 0
        fprintf('Simulation completed! \n')
```

```
        fprintf('Number of iterations: %s \n', num2str(n))
        fprintf('Number of outputs: %s \n', num2str(n_out))
        fprintf('Elapsed time: %s s \n', num2str(toc))
    end

%% Output variables storage
    savefile = 'output.mat';
    save(savefile,'x','dx','y','dy','t','dt','n_dt_out','U_x_n','U_y_n',...
        'Umod_n','Den_n','P_n','T_n');
%% Note
% Output solution variables are in the form V_n(i,j,nt)
% time t is related to time step index nt through
% t(nt) = (nt-1)*dt*n_dt_out
```

B.5.2 Time advancement function

```
function [Sn_n1, Den_n1, U_x_n1, U_y_n1, P_n1, T_n1] = ...
    advance_Eul2d_NRBCx(Sn_n, Den_n, U_x_n, U_y_n, P_n, T_n)

% This function produces numerical solution to 2D Euler equations
% at time t + dt for the variables Sn, Den, U_x, U_y, P, T
% (subscript _n1) starting from those at time t (subscript _n)
%
% Time advancement is performed through a Runge-Kutta of order rk_ord.
% Details can be found in Canuto et al., "Spectral methods
% in fluid dynamics" I ed. page 109, and Jameson, Schmidt, Turkel (1981)
%
% RK scheme is the following:
%   U = U_n
%   for i = rk_ord:1:-1
%       U = U_n + 1/i * dt * F(U)    (*)
%   end
%   U_n+1 = U  (**)

%% Load global variables

% Caricamento variabili globali
global gamma rk_ord nx ny dt
global M  err_flag
global i_filter ndt_filter filter_type alpha_filter

% amplitude of characteristic waves initialization
    if iBC == 4
       L01 = zeros(2,ny);
       L02 = zeros(2,ny);
       L_p = zeros(2,ny);
       L_m = zeros(2,ny);
    end

%% RK time advancement routine
```

```
for i = rk_ord:-1:1
%% 0. Calculating system characteristic variables

        der = der1x_BC(U_y_n);
        L01(1,:) = U_x_n(1,:).*der(1,:);      % vorticity wave
        L01(2,:) = U_x_n(nx,:).*der(nx,:);

        % L02 = Ux*(-1/c^2*@p/@x+@rho/@x)     % entropy wave
        c_inv2 = 1/gamma*Den_n./P_n;
        der = -c_inv2.*der1x_BC(P_n)+der1x_BC(Den_n);
        L02(1,:) = U_x_n(1,:).*der(1,:);
        L02(2,:) = U_x_n(nx,:).*der(nx,:);

        % L_+ = 1/2*(Ux+c)*(@p/@x+c*rho*@Ux/@x)
        c = sqrt(gamma*P_n./Den_n);
        der = der1x_BC(P_n)+c.*Den_n.*der1x_BC(U_x_n);
        L_p(1,:) = 0.5*(U_x_n(1,:)+c(1,:)).*der(1,:);
        L_p(2,:) = 0.5*(U_x_n(nx,:)+c(nx,:)).*der(nx,:);

        % L_- = 1/2*(Ux-c)*(@p/@x-c*rho*@Ux/@x)
        der = der1x_BC(P_n)-c.*Den_n.*der1x_BC(U_x_n);
        L_m(1,:) = 0.5*(U_x_n(1,:)-c(1,:)).*der(1,:);
        L_m(2,:) = 0.5*(U_x_n(nx,:)-c(nx,:)).*der(nx,:);

        % NRBC at boundaries

        % left boundary
         for j = 1:ny
             if U_x_n(1,j) >= 0
             L01(1,j) = 0;
             L02(1,j) = 0;
             end

             if U_x_n(1,j)-c(1,j) >= 0
             L_m(1,j) = 0;
             end

             if U_x_n(1,j)+c(1,j) >= 0
             L_p(1,j) = 0;
             end
         end

         % right boundary
         for j = 1:ny
             if U_x_n(nx,j) <= 0
             L01(2,j) = 0;
             L02(2,j) = 0;
             end

             if U_x_n(nx,j)-c(nx,j) <= 0
             L_m(2,j) = 0;
```

```
            end

            if U_x_n(nx,j)+c(nx,j) <= 0
            L_p(2,j) = 0;
            end
        end
    end

%% 1. Adiabatic equation advancement

    SnU_n = Sn_n .* U_x_n;
    RHS = der1x_BC(SnU_n);

    % x direction derivative
        c2 = gamma*P_n./Den_n;
        RHS(1,:) = (L_p(1,:)+L_m(1,:))/gamma+(1-gamma)/gamma*(c2(1,:).*L02(1,:));
        RHS(1,:) = Den_n(1,:).^(1-gamma).* RHS(1,:);
        RHS(nx,:) = (L_p(2,:)+L_m(2,:))/gamma+(1-gamma)/gamma*(c2(nx,:).*L02(2,:))
        RHS(nx,:) = Den_n(nx,:).^(1-gamma).* RHS(nx,:);

    % y direction derivative
    SnU_n = Sn_n .* U_y_n;
    RHS = RHS + der1y_per(SnU_n);

    % time advancement
    Sn_n1 = Sn_n - RHS * dt / i;


%% 2. Continuity equation advancement

    nU_n = Den_n .* U_x_n;
    RHS = der1x_BC(nU_n);

    % x direction derivative
        RHS(1,:) = L02(1,:) + (L_p(1,:)+L_m(1,:)).*c_inv2(1,:);
        RHS(nx,:) = L02(2,:) + (L_p(2,:)+L_m(2,:)).*c_inv2(nx,:);

    % y direction derivative
    nU_n = Den_n .* U_y_n;
    RHS = RHS + der1y_per(nU_n);

    % time advancement
    Den_n1 = Den_n - RHS * dt /i;

    % checking density to be positive,
    % otherwise imaginary terms are produced while computing
    % P_n1 = Sn_n1 ./ (Den_n1.^(1-gamma));

    for k = 1:nx
        for l = 1:ny
            if Den_n1(k,l) <= 0
```

```
                err_flag = 1;
                break
            end
        end
    end

%% 3. Momentum equation advancement

    % y component of momentum equation in conservative form
        % RHS = [ @x(rho U_x * U_y) + @y(rho * U_y * U_y) - @p/@y

        % x direction derivative
        Aij =  Den_n .* (U_x_n .* U_y_n);  % Axy
        RHS = der1x_BC(Aij);

            RHS(1,:) = L01(1,:).*Den_n(1,:)+ U_y_n(1,:).*(L02(1,:)+...
                (L_p(1,:)+L_m(1,:)).*c_inv2(1,:));
            RHS(nx,:) = L01(2,:).*Den_n(nx,:)+ U_y_n(nx,:).*(L02(2,:)+...
                (L_p(2,:)+L_m(2,:)).*c_inv2(nx,:));

        % y direction derivative
        Aij =  Den_n .* (U_y_n .* U_y_n) + P_n/M; % Ayy
        RHS = RHS + der1y_per(Aij);

    % time advancement
    nU_y_n = Den_n .* U_y_n;
    nU_y_n1 = nU_y_n - RHS * dt /i;

    % y component of momentum equation in conservative form
        % RHS = - [ @x(rho U_x * U_x) + @y(rho * U_y * U_x) ] - @p/@x

        % x direction derivative
        Aij =  Den_n .* (U_x_n .* U_x_n) + P_n/M;  % Axx
        RHS = der1x_BC(Aij);

            RHS(1,:) = (L_p(1,:)-L_m(1,:)).*sqrt(c_inv2(1,:))+...
                U_x_n(1,:).*(L02(1,:)+(L_p(1,:)+L_m(1,:)).*c_inv2(1,:));
            RHS(nx,:) = (L_p(2,:)-L_m(2,:)).*sqrt(c_inv2(nx,:))+...
                U_x_n(nx,:).*(L02(2,:)+(L_p(2,:)+L_m(2,:)).*c_inv2(nx,:));

        % y direction derivative
        Aij = Den_n .* (U_y_n .* U_x_n); % Ayx
        RHS = RHS + der1y_per(Aij);

    % time advancement
    nU_x_n = Den_n .* U_x_n;
    nU_x_n1 = nU_x_n - RHS * dt /i;

    %% 4. Filtering

    if Filter == true && i_filter == ndt_filter
```

```
        if filter_type == 2
            Sn_n1 = filter_BC2d(Sn_n1,alpha_filter);
            Den_n1 = filter_BC2d(Den_n1,alpha_filter);
            nU_x_n1 = filter_BC2d(nU_x_n1,alpha_filter);
            nU_y_n1 = filter_BC2d(nU_y_n1,alpha_filter);
        elseif filter_type == 1
            Sn_n1 = filter_NL2d(Sn_n1);
            Den_n1 = filter_NL2d(Den_n1);
            nU_x_n1 = filter_NL2d(nU_x_n1);
            nU_y_n1 = filter_NL2d(nU_y_n1);
        end
    end

    %% 5. Indirect quantities
    % quantities not directly featuring under time derivative

    U_x_n1 = nU_x_n1./Den_n1;
    U_y_n1 = nU_y_n1./Den_n1;
    P_n1 = Sn_n1 ./ (Den_n1.^(1-gamma));
    T_n1 = P_n1 ./ Den_n1;    % perfect gas equation of state
    % Note: divide by kB*M to convert it to Kelvin.

    %% 6. Updating quantities for the next RK cycle step
    Sn_n = Sn_n1;
    Den_n = Den_n1;
    U_x_n = U_x_n1;
    U_y_n = U_y_n1;
    P_n = P_n1;
    T_n = T_n1;
end
```

B.6 2D Navier-Stokes equations (Mac Cormack scheme)

B.6.1 Main section

```
% 2D isothermal compressible Navier-Stokes equations
% with Mac Cormack scheme on  x-open y-periodic domain
clear

%% Initialization and options

% global variables
global i_filter ndt_filter filter A a

% Grid domain definition

nx = % number of spatial points along x
ny = % number of spatial points along y
Lx = % box length in x direction
```

```
Ly = % box length in y direction

dx = Lx/(nx-1); % open direction
dy = Ly/ny;     % periodic direction

x = 0:dx:Lx;
y = 0:dy:Ly-dy;

dt =    % time step
t_max = % final time

% Filtering options
filter = 0;
i_filter = 0;
ndt_filter = 4;     % filter the solution every n_filter time steps
[A,a] = init_filter_matrix(0.45,nx); % filtering operator on x direction

% Fluid parameters

mu = 0.001;
lambda = -mu/3;
C = 1;

% time advancement options and counters initialization
t = 0;          % initial time
n_out = 1;      % counter for the number of outputs
n_dt_out = 1;   % output the solution every n_dt_out time steps
out_counter =   % counter for the total number of time steps performed

%% Initial conditions
% Initial conditions must be provided through an external function
% condinit(...) giving [rho, U_x, U_y, P] as output
[rho, U_x, U_y, P] = condinit(~);

U_x_n = U_x;
U_y_n = U_y;
rho_n = rho;
P_n = P;
Umod_n = sqrt(U_x.^2 + U_y.^2);

%% Time advancement main loop
while t <= t_max

    % FF/BB

      [rho, U_x, U_y, P] = advanceNS_MC_FFBB(rho, U_x, U_y, P,...
          mu, lambda, C, nx, ny, dx, dy, dt);
      i_filter = i_filter + 1;
      t = t + dt;
      out_counter = out_counter + 1;
```

```
    if out_counter == n_dt_out
        out_counter = 0;
        n_out = n_out + 1;

        U_x_n(:,:,n_out) = U_x;
        U_y_n(:,:,n_out) = U_y;
        rho_n(:,:,n_out) = rho;
        P_n(:,:,n_out) = P;
        Umod_n(:,:,n_out) = sqrt(U_x.^2 + U_y.^2);

        fprintf(['t = ', num2str(t),'\n'])
    end

% FB/BF

    [rho, U_x, U_y, P] = advanceNS_MC_FBBF(rho, U_x, U_y, P,...
        mu, lambda, C, nx, ny, dx, dy, dt);

    t = t + dt;
    out_counter = out_counter+1;

    if out_counter == n_dt_out
        out_counter = 0;
        n_out = n_out + 1;

        U_x_n(:,:,n_out) = U_x;
        U_y_n(:,:,n_out) = U_y;
        rho_n(:,:,n_out) = rho;
        P_n(:,:,n_out) = P;
        Umod_n(:,:,n_out) = sqrt(U_x.^2 + U_y.^2);

        fprintf(['t = ', num2str(t),'\n'])

    end

% BB/FF

[rho, U_x, U_y, P] = advanceNS_MC_BBFF(rho, U_x, U_y, P,...
    mu, lambda, C, nx, ny, dx, dy, dt);

    t = t + dt;
    out_counter = out_counter+1;

    if out_counter == n_dt_out
        out_counter = 0;
        n_out = n_out + 1;

        U_x_n(:,:,n_out) = U_x;
        U_y_n(:,:,n_out) = U_y;
        rho_n(:,:,n_out) = rho;
```

```
            P_n(:,:,n_out) = P;
            Umod_n(:,:,n_out) = sqrt(U_x.^2 + U_y.^2);

            fprintf(['t = ', num2str(t),'\n'])

        end

    % BF/FB

    [rho, U_x, U_y, P] = advanceNS_MC_BFFB(rho, U_x, U_y, P,...
        mu, lambda, C, nx, ny, dx, dy, dt);

        t = t + dt;
        out_counter = out_counter+1;

        if out_counter == n_dt_out
            out_counter = 0;
            n_out = n_out + 1;

            U_x_n(:,:,n_out) = U_x;
            U_y_n(:,:,n_out) = U_y;
            rho_n(:,:,n_out) = rho;
            P_n(:,:,n_out) = P;
            Umod_n(:,:,n_out) = sqrt(U_x.^2 + U_y.^2);

            fprintf(['t = ', num2str(t),'\n'])
        end
end

%% Output variables storage
savefile = 'output.mat';
save(savefile,'x','y','t','nx','ny','dx','dy','dt','U_x_n',...
    'U_y_n','Umod_n','rho_n','P_n');
```

B.6.2 Time advancement functions

(Modify the order of forward and backward differentiation to produce each scheme)

```
function [rho_n1, U_x_n1, U_y_n1, P_n1] = ...
advanceNS_MC_FBBF(rho_n,U_x_n,U_y_n,P_n,mu,lambda,C,nx,ny,dx,dy,dt)

global A a filter i_filter ndt_filter

c1 = dt/dx;
c2 = dt/dy;
c3x = (2*mu+lambda)*dt/dx^2;
c3y = mu*dt/dx^2;
c4x = mu*dt/dy^2;
c4y = (2*mu+lambda)*dt/dy^2;
```

```
c5 = dt*(mu+lambda)/4/dx/dy;

rhoU_x_n = rho_n.*U_x_n;
rhoU_y_n = rho_n.*U_y_n;

% Predictors calculation

 % Characteristic based B.C. for predictors
 % with LODI method (Poinsot, Lele 1992)
    U_b = zeros(ny,2);
    rho_b = zeros(ny,2);
    L_p = zeros(ny,2);
    L_m = zeros(ny,2);


    U_b(:,1) = U_x_n(:,1);
    rho_b(:,1) = rho_n(:,1);
    U_b(:,2) = U_x_n(:,end);
    rho_b(:,2) = rho_n(:,end);

    for i = 1:ny
% left boundary

    % acoustic wave +
    if U_b(i,1) + C >= 0
        L_p(i,1) = 0;
    else
        L_p(i,1) = 0.5*(U_b(i,1)+C)*(rho_b(i,1)*L1s_x(U_x_n(i,1:3)) +...
            C/rho_b(i,1)*L1s_x(rho_n(i,1:3)))/dx;
    end
% acoustic wave -
if U_b(i,1) - C >= 0
    L_m(i,1) = 0;
else
    L_m(i,1) = 0.5*(U_b(i,1)-C)*(rho_b(i,1)*L1s_x(U_x_n(i,1:3)) -...
        C/rho_b(i,1)*L1s_x(rho_n(i,1:3)))/dx;
end


% right boundary

% acoustic wave +
if U_b(i,2) + C <= 0
    L_p(i,2) = 0;
else
    L_p(i,2) = 0.5*(U_b(i,2)+C)*(rho_b(i,2)*R1s_x(U_x_n(i,end-2:end)) +...
        C/rho_b(i,2)*R1s_x(rho_n(i,end-2:end)))/dx;
end

% acoustic wave -
if U_b(i,2) - C <= 0
```

```
    L_m(i,2) = 0;
else
    L_m(i,2) = 0.5*(U_b(i,2)-C)*(rho_b(i,2)*R1s_x(U_x_n(i,end-2:end)) -...
        C/rho_b(i,2)*R1s_x(rho_n(i,end-2:end)))/dx;
end

    end

% fractional time step advancement with Runge-Kutta JST method, taken as
% predictor quantity
rho_left_p = rho_b(:,1) - dt/2*(L_p(:,1)-L_m(:,1))/C.*rho_b(:,1);
U_left_p = U_b(:,1) - dt/2*(L_p(:,1)+L_m(:,1));
rho_right_p = rho_b(:,2) - dt/2*(L_p(:,2)-L_m(:,2))/C.*rho_b(:,2);
U_right_p = U_b(:,2) - dt/2*(L_p(:,2)+L_m(:,2));


% internal points

% continuity equation predictor

rho_p = rho_n - c1*Fdiff_x(rho_n.*U_x_n) - c2*Bdiff_y(rho_n.*U_y_n);

% x momentum equation predictor

rhoU_x_p = rhoU_x_n - c1*Fdiff_x(rhoU_x_n.*U_x_n + P_n) -...
    c2*Bdiff_y(rhoU_x_n.*U_y_n) + c3x*C2diff_x(U_x_n) +...
    c4x*C2diff_y(U_x_n) + c5*C2diff_xy(U_y_n);

% y momentum equation predictor

rhoU_y_p = rhoU_y_n - c1*Fdiff_x(rhoU_y_n.*U_x_n) -...
    c2*Bdiff_y(rhoU_y_n.*U_y_n + P_n) + c3y*C2diff_x(U_y_n) +...
    c4y*C2diff_y(U_y_n) + c5*C2diff_xy(U_x_n);

    % updating of predictors at boundaries
    rho_p(:,1) = rho_left_p;
    rho_p(:,end) = rho_right_p;
    rhoU_x_p(:,1) = rho_p(:,1).*U_left_p;
    rhoU_x_p(:,end) = rho_p(:,end).*U_right_p;


    % filtering all quantities

    if filter == 1
        if i_filter == ndt_filter
            for j = 1:nx
                rho_p(:,j) = filter_per(rho_p(:,j));
                rhoU_x_p(:,j) = filter_per(rhoU_x_p(:,j));
                rhoU_y_p(:,j) = filter_per(rhoU_y_p(:,j));
            end
            for i = 1:ny
```

```
                rho_p(i,:) = filter_BC1d_call(rho_p(i,:),A,a);
                rhoU_x_p(i,:) = filter_BC1d_call(rhoU_x_p(i,:),A,a);
                rhoU_y_p(i,:) = filter_BC1d_call(rhoU_y_p(i,:),A,a);
            end
        end
    end

    % calculation of indirect predictors
    U_x_p = rhoU_x_p./rho_p;
    U_y_p = rhoU_y_p./rho_p;

% Correctors (time advancement) calculation

% Characteristic based B.C. for correctors
% with LODI method (Poinsot, Lele 1992)

    U_b(:,1) = U_x_p(:,1);
    rho_b(:,1) = rho_p(:,1);
    U_b(:,2) = U_x_p(:,end);
    rho_b(:,2) = rho_p(:,end);

    for i = 1:ny

% left boundary

    % acoustic wave +
    if U_b(i,1) + C >= 0
        L_p(i,1) = 0;
    else
        L_p(i,1) = 0.5*(U_b(i,1)+C)*(rho_b(i,1)*L1s_x(U_x_p(i,1:3)) +...
            C/rho_b(i,1)*L1s_x(rho_p(i,1:3)))/dx;
    end
    % acoustic wave -
    if U_b(i,1) - C >= 0
        L_m(i,1) = 0;
    else
        L_m(i,1) = 0.5*(U_b(i,1)-C)*(rho_b(i,1)*L1s_x(U_x_p(i,1:3)) -...
            C/rho_b(i,1)*L1s_x(rho_p(i,1:3)))/dx;
    end

% right boundary
    % acoustic wave +

    if U_b(i,2) + C <= 0
        L_p(i,2) = 0;
    else
        L_p(i,2) = 0.5*(U_b(i,2)+C)*(rho_b(i,2)*R1s_x(U_x_p(i,end-2:end)) +...
            C/rho_b(i,2)*R1s_x(rho_p(i,end-2:end)))/dx;
    end
    % acoustic wave -
    if U_b(i,2) - C <= 0
```

```
        L_m(i,2) = 0;
    else
        L_m(i,2) = 0.5*(U_b(i,2)-C)*(rho_b(i,2)*R1s_x(U_x_p(i,end-2:end)) -...
            C/rho_b(i,2)*R1s_x(rho_p(i,end-2:end)))/dx;
    end

    end

% fractional step advancement with Runge-Kutta JST method, taken as
% predictor quantity
rho_left_c = rho_b(:,1) - dt/C*(L_p(:,1)-L_m(:,1)).*rho_b(:,1);
U_left_c = U_b(:,1) - dt*(L_p(:,1)+L_m(:,1));% - c3x(L1s_2x(U_x));
rho_right_c = rho_b(:,2) - dt/C*(L_p(:,2)-L_m(:,2)).*rho_b(:,2);
U_right_c = U_b(:,2) - dt*(L_p(:,2)+L_m(:,2));% - c3x(L1s_2x(U_x));


% internal points

% continuity equation corrector

rho_n1 = rho_n + rho_p - c1*Bdiff_x(rhoU_x_p) - c2*Fdiff_y(rhoU_y_p);
rho_n1 = 0.5*rho_n1;

% x momentum equation corrector

rhoU_x_n1 = rhoU_x_n + rhoU_x_p - c1*Bdiff_x(rhoU_x_p.*U_x_p+C^2*rho_p)+...
        - c2*Fdiff_y(rhoU_x_p.*U_y_p) + c3x*C2diff_x(U_x_p) +...
        c4x*C2diff_y(U_x_p) + c5*C2diff_xy(U_y_p);
rhoU_x_n1 = 0.5*rhoU_x_n1;

% y momentum equation corrector

rhoU_y_n1 = rhoU_y_n + rhoU_y_p - c1*Bdiff_x(rhoU_y_p.*U_x_p) +...
        - c2*Fdiff_y(rhoU_y_p.*U_y_p+C^2*rho_p) + c3y*C2diff_x(U_y_p)+...
        c4y*C2diff_y(U_y_p) + c5*C2diff_xy(U_x_p);
rhoU_y_n1 = 0.5*rhoU_y_n1;

% updating of correctors at boundaries
rho_n1(:,1) = rho_left_c;
rho_n1(:,end) = rho_right_c;
rhoU_x_n1(:,1) = rho_n1(:,1).*U_left_c;
rhoU_x_n1(:,end) = rho_n1(:,end).*U_right_c;

% filtering all quantities
if filter == 1
    if i_filter == ndt_filter
        i_filter = 0;
        for j = 1:nx
            rho_n1(:,j) = filter_per(rho_n1(:,j));
            rhoU_x_n1(:,j) = filter_per(rhoU_x_n1(:,j));
            rhoU_y_n1(:,j) = filter_per(rhoU_y_n1(:,j));
```

```
        end
        for i = 1:ny
            rho_n1(i,:) = filter_BC1d_call(rho_n1(i,:),A,a);
            rhoU_x_n1(i,:) = filter_BC1d_call(rhoU_x_n1(i,:),A,a);
            rhoU_y_n1(i,:) = filter_BC1d_call(rhoU_y_n1(i,:),A,a);
        end
    end
end

% indirect correctors
U_x_n1 = rhoU_x_n1./rho_n1;
U_y_n1 = rhoU_y_n1./rho_n1;
P_n1 = rho_n1*C^2;

end
```

Appendix C

List of events

The following table collects the data obtained from joining LASCO catalog and Richardson and Cane near-Earth ICME list, employed for the Drag-Based Model inversion procedure performed in Chapter 4.

Event #	CME start date	v_0 (Km/s)	ICME arrival date[1]	v_1 (Km/s)
1	1997-04-07 14:27	664	1997-04-11 06:00	552
2	1997-08-30 01:30	264	1997-09-03 13:00	405
3	1997-09-28 01:08	226	1997-10-01 16:00	580
4	1997-10-06 15:28	274	1997-10-10 22:00	430
5	1997-11-04 06:10	1058	1997-11-07 04:00	640
6	1997-12-06 10:27	309	1997-12-10 18:00	460
7	1997-12-26 02:31	350	1997-12-30 10:00	430
8	1998-01-02 23:28	411	1998-01-07 01:00	480
9	1998-01-25 15:26	1572	1998-01-29 20:00	557
10	1998-04-29 16:58	843	1998-05-02 05:00	780
11	1998-11-09 18:18	310	1998-11-13 02:00	520
12	1999-04-13 03:30	290	1999-04-16 18:00	520
13	1999-06-24 13:31	777	1999-06-27 22:00	760
14	1999-07-03 19:54	685	1999-07-06 21:00	620
15	1999-08-09 03:26	248	1999-08-12 03:00	615
16	1999-08-17 13:31	1057	1999-08-20 23:00	510
17	1999-10-18 00:06	969	1999-10-21 08:00	561
18	2000-01-18 17:54	454	2000-01-22 17:00	530
19	2000-02-08 09:30	559	2000-02-11 16:00	630
20	2000-02-10 02:30	756	2000-02-12 12:00	915
21	2000-02-12 04:31	857	2000-02-14 12:00	815
22	2000-02-17 20:06	770	2000-02-21 05:00	560
23	2000-05-10 20:06	670	2000-05-13 17:00	603
24	2000-05-13 12:26	546	2000-05-16 23:00	500
25	2000-05-21 07:26	678	2000-05-24 12:00	650
26	2000-05-31 08:06	375	2000-06-04 22:00	403

27	2000-06-06 15:54	3229	2000-06-08 12:00	1007
28	2000-07-07 10:26	535	2000-07-11 02:00	609
29	2000-07-14 10:54	1909	2000-07-15 19:00	1500
30	2000-07-23 05:30	599	2000-07-27 02:00	490
31	2000-07-25 03:30	442	2000-07-28 12:00	550
32	2000-08-06 22:30	186	2000-08-10 19:00	530
33	2000-08-09 16:30	515	2000-08-12 05:00	830
34	2000-09-05 05:54	461	2000-09-08 12:00	530
35	2000-10-02 20:26	598	2000-10-05 13:00	756
36	2000-10-09 23:50	903	2000-10-13 16:00	590
37	2000-10-25 08:26	1201	2000-10-28 21:00	565
38	2000-11-03 18:26	379	2000-11-06 17:00	660
39	2000-12-18 11:50	437	2000-12-23 00:00	380
40	2001-02-28 14:50	731	2001-03-04 04:00	610
41	2001-03-25 17:06	1168	2001-03-28 17:00	850
42	2001-03-28 12:50	607	2001-03-31 05:00	690
43	2001-03-29 10:26	1103	2001-04-01 04:00	700
44	2001-04-06 19:30	616	2001-04-08 14:00	1050
45	2001-04-10 05:30	519	2001-04-11 22:00	1290
46	2001-04-11 13:31	768	2001-04-13 09:00	990
47	2001-04-26 12:30	590	2001-04-28 14:00	1040
48	2001-07-05 03:54	400	2001-07-09 02:00	520
49	2001-09-28 08:54	651	2001-10-01 08:00	710
50	2001-10-09 11:30	1145	2001-10-12 04:00	780
51	2001-10-19 16:50	855	2001-10-21 20:00	870
52	2001-11-04 16:35	1624	2001-11-06 12:00	1250
53	2001-11-17 05:30	421	2001-11-19 22:00	680
54	2001-11-22 23:30	1484	2001-11-24 14:00	1320
55	2002-03-15 23:06	1121	2002-03-19 05:00	667
56	2002-04-15 03:50	755	2002-04-17 16:00	750
57	2002-04-17 08:26	1387	2002-04-20 00:00	863
58	2002-05-08 13:50	279	2002-05-11 15:00	610
59	2002-05-16 00:50	339	2002-05-20 10:00	420
60	2002-07-15 20:30	2160	2002-07-18 12:00	955
61	2002-07-29 12:07	606	2002-08-02 06:00	500
62	2002-08-16 12:30	1253	2002-08-19 12:00	766
63	2002-09-05 16:54	671	2002-09-08 04:00	880
64	2002-09-17 08:06	879	2002-09-19 20:00	910
65	2003-05-29 01:27	974	2003-05-30 22:00	1078
66	2003-06-14 01:54	652	2003-06-17 10:00	650
67	2003-10-22 08:30	590	2003-10-24 21:00	760
68	2003-10-26 17:54	1362	2003-10-28 02:30	1331
69	2003-10-28 11:30	600	2003-10-29 11:00	2185

70	2003-11-18 08:50	1982	2003-11-20 10:00	886
71	2004-01-20 00:06	1386	2004-01-22 08:00	850
72	2004-01-21 04:54	979	2004-01-23 23:00	720
73	2004-07-20 13:31	650	2004-07-22 18:00	920
74	2004-07-22 07:31	937	2004-07-24 14:00	890
75	2004-07-23 16:06	1222	2004-07-25 20:00	890
76	2004-07-25 14:54	1188	2004-07-27 02:00	1302
77	2004-09-12 00:36	1041	2004-09-14 15:00	960
78	2004-09-14 10:10	450	2004-09-18 12:00	500
79	2004-11-04 23:30	819	2004-11-07 22:00	720
80	2004-11-07 16:54	1478	2004-11-09 20:00	830
81	2004-11-10 02:26	552	2004-11-12 08:00	1080
82	2005-01-05 15:30	445	2005-01-08 21:00	570
83	2005-01-20 06:54	1631	2005-01-21 19:00	1210
84	2005-02-17 00:06	1134	2005-02-20 12:00	500
85	2005-05-16 13:50	310	2005-05-20 03:00	488
86	2005-05-26 15:06	511	2005-05-30 01:00	630
87	2005-06-09 14:36	460	2005-06-12 15:00	650
88	2005-07-07 17:06	545	2005-07-10 10:00	720
89	2005-08-05 08:54	270	2005-08-09 00:00	477
90	2005-08-31 11:30	917	2005-09-02 18:00	840
91	2006-07-06 08:54	657	2006-07-10 21:00	488
92	2006-08-16 16:30	1312	2006-08-20 13:00	620
93	2006-08-26 20:57	443	2006-08-30 20:00	440
94	2006-12-13 02:54	1858	2006-12-14 22:00	1180
95	2006-12-14 22:30	954	2006-12-17 00:00	980
96	2008-12-12 08:54	190	2008-12-17 03:00	350
97	2010-02-07 03:54	590	2010-02-11 08:00	450
98	2010-04-08 04:54	281	2010-04-12 01:00	520
99	2010-05-24 14:06	382	2010-05-28 19:00	500
100	2011-03-03 05:48	238	2011-03-06 09:00	600
101	2011-08-02 06:36	1014	2011-08-05 05:00	670
102	2011-08-04 04:12	1143	2011-08-06 22:00	1100
103	2011-09-06 23:05	281	2011-09-10 03:00	680
104	2011-09-24 12:48	1351	2011-09-26 20:00	870
105	2011-10-27 12:00	619	2011-11-02 01:00	350
106	2011-11-09 13:30	433	2011-11-13 10:00	650
107	2011-11-26 07:00	979	2011-11-29 00:00	670
108	2012-01-18 12:24	369	2012-01-21 06:00	650
109	2012-01-19 14:36	846	2012-01-22 23:00	670
110	2012-03-07 00:24	418	2012-03-09 03:00	1220
111	2012-03-13 17:36	454	2012-03-15 17:00	960
112	2012-05-12 00:00	755	2012-05-16 16:00	370

113	2012-07-02 08:36	1196	2012-07-05 00:00	730
114	2012-07-04 17:24	538	2012-07-09 00:00	480
115	2012-09-28 00:00	904	2012-10-01 00:00	590
116	2012-10-27 16:24	250	2012-11-01 00:00	440
117	2012-11-09 15:12	647	2012-11-13 08:00	530
118	2012-11-20 12:12	729	2012-11-24 12:00	510
119	2012-11-23 13:36	415	2012-11-26 12:00	660
120	2013-01-13 12:00	208	2013-01-17 16:00	420
121	2013-03-15 07:12	299	2013-03-17 15:00	890
122	2013-04-11 07:24	954	2013-04-14 17:00	660
123	2013-06-23 22:36	378	2013-06-28 02:00	470
124	2013-07-09 15:12	379	2013-07-13 05:00	563
125	2013-09-29 21:45	258	2013-10-02 23:00	800
126	2013-10-06 14:43	585	2013-10-09 09:00	780
127	2013-12-12 03:36	265	2013-12-15 16:00	510
128	2014-02-04 00:36	306	2014-02-08 01:00	470
129	2014-02-12 06:12	435	2014-02-16 05:00	530
130	2014-04-02 13:48	800	2014-04-05 22:00	610
131	2014-04-18 13:25	773	2014-04-21 07:00	920
132	2014-06-04 15:24	417	2014-06-08 20:00	570
133	2014-08-15 18:12	458	2014-08-19 16:00	500
134	2014-09-10 18:00	461	2014-09-12 22:00	920
135	2014-09-12 19:24	741	2014-09-17 02:00	410
136	2014-12-17 05:00	584	2014-12-22 04:00	380
137	2015-03-15 02:36	824	2015-03-17 13:00	840
138	2015-06-21 02:36	1418	2015-06-23 02:00	1040
139	2015-06-22 18:36	1391	2015-06-25 10:00	960
140	2015-07-10 02:24	191	2015-07-13 06:00	580
141	2015-08-12 15:12	804	2015-08-15 21:00	640
142	2015-09-04 19:24	490	2015-09-08 00:00	620
143	2015-09-18 05:00	723	2015-09-21 08:00	850
144	2016-01-14 23:24	162	2016-01-19 10:00	440
145	2016-04-10 11:00	223	2016-04-14 09:00	460
146	2016-07-17 12:48	424	2016-07-20 07:00	710
147	2016-11-05 04:24	423	2016-11-10 00:00	430
148	2017-05-23 05:36	286	2017-05-27 22:00	400
149	2017-07-14 01:25	994	2017-07-16 15:00	810
150	2017-09-04 20:36	864	2017-09-07 20:00	820
151	2017-09-06 12:24	872	2017-09-08 11:00	1210

[1] For the ICME arrival time we considered the ICME plasma time from Richardson and Cane near-Earth ICME list.

Bibliography

[JR76] T. L. Holst J. Tannehill and J. V. Rakich. "Numerical Computation of Two-Dimensional Viscous Blunt Body Flows with an Impinging Shock." In: *AIAA* 14.2 (1976), pp. 204–211.

[Sch76] Hermann Schlichting. *Boundary-Layer Theory*. Mac Graw Hill, 1976.

[AJ+84] Hundhausen A.J. et al. "Coronal mass ejections observed during the solar maximum mission - Latitude distribution and rate of occurrence." In: *J. Geophys. Res.* 89 (1984), pp. 2639–2646.

[Dyk88] M. Van Dyke. *An album of fluid motion*. The parabolic press, 1988.

[CH89] W. A. Coles and J.K. Harmon. "Propagation observations of the solar wind near the Sun". In: *Astrophys. J.* 337 (1989), pp. 1023–1034.

[EFT89] E.F.Toro. "A CFL Condition For Characteristic Based Methods." In: *Applied Mathematics Letters* 2.1 (1989), pp. 57–60.

[Vrs90] B. Vrsnak. "Eruptive instability of cylindrical prominences". In: *Sol. Phys.* 129.A4 (1990), pp. 295–312.

[CT91] S. Chapman and T.G.Cowling. *The mathematical theory of non-uniform gases*. Cambridge Mathematical Library, 1991.

[Lam91] J. D. Lambert. *Numerical Methods for Ordinary Differential Systems: The Initial Value Problem*. John Wiley Sons Inc, 1991.

[Lel92] S. Lele. "Compact finite difference schemes with spectral-like resolution". In: *Journal of Computational Physics* 103.1 (1992), pp. 16–42. DOI: https://doi.org/10.1016/0021-9991(92)90324-R.

[LP92] S. K. Lele and T. J. Poinsot. "Boundary Conditions for Direct Simulations of Compressible Viscous Flows". In: *Journal of Computational Physics* 101 (1992), pp. 104–129.

[HBC94] A. J. Hundhausen, J. T. Burkepile, and O. C. St. Cyr. "Speeds of coronal mass ejections: SMM observations from 1980 and 1984-1989". In: *J. Geophys. Res.* 99.A4 (1994), pp. 6543–6552. DOI: https://doi.org/10.1029/93JA03586.

[And95] John D. Anderson. *Computational Fluid Dynamics*. McGraw-Hill Education. 1995.

[Hou+95] S. Hou et al. "Simulation of cavity flow by lattice Boltzmann method." In: *Journal of Computational Physics* 118 (1995), pp. 329–347.

[Car+96] P. Cargill et al. "Magnetohydrodynamic simulations of the motion of magnetic flux tubes through a magnetized plasma." In: *Journal of Geophysical Research* 101.A3 (1996), pp. 4855–4870. DOI: 10.1029/95JA03769.

[Che96] J. Chen. "Theory of prominence eruption and propagation: Interplanetary consequences". In: *Journal of Geophysical Research* 101.A12 (1996), pp. 27499–27520. DOI: 10.1029/96JA02644.

[PBK96] P.Subramanian, P. A. Becker, and M. Kafatos. "Ion viscosity mediated by tangled magnetic fields: An application to black hole accretion disks". In: *Astrophys. J.* 469 (1996), pp. 784–793.

[Sch96] R .Schwenn. "An essay on terminology, myths,- and known facts: Solar transient-flare-CME-driver gas-piston-BDE-magnetic cloud-shock wave-geomagnetic storm". In: *Astrophysics and Space Science* 243.1 (1996), pp. 187–193.

[JP97] D.A. Anderson J. C. Tannehill and R.H. Pletcher. *Computational Fluid Mechanics and Heat Transfer*. Cambridge Mathematical Library. Taylor & Francis, 1997.

[LDB98] Yolande Leblanc, George A. Dulk, and Jean-Louis Bougeret. "Tracing the Electron Density from the Corona to 1 AU." In: *Solar Physics* 183.1 (1998), pp. 165–180. DOI: https://doi.org/10.1023/A:1005049730506.

[MW99] D. Mihalas and B. W.-Mihalas. *Foundations of Radiation Hydrodynamics*. Dover Publications, 1999.

[Bat00] G. Batchelor. *An Introduction to Fluid Dynamics*. Cambridge Mathematical Library. Cambridge University Press, 2000. ISBN: 978-0521663960.

[BB02] C. Bogey and C. Bailly. "Three-dimensional non-reflective boundary conditions for acoustic simulation: far field formulation and validation test cases." In: *Acta Acustica united with Acustica* 88.4 (2002), pp. 463–471.

[Car04] Peter J. Cargill. "On the Aerodynamic Drag Force Acting on Interplanetary Coronal Mass Ejections." In: *Solar Physics* 221.1 (2004), pp. 135–149. DOI: https://doi.org/10.1023/B:SOLA.0000033366.10725.a2.

[Rob04] D. Robbrecht E.; Berghmans. "Automated recognition of coronal mass ejections (CMEs) in near-real-time data". In: *Astronomy and Astrophysics* 425 (2004), pp. 1097–1106. DOI: 10.1051/0004-6361:20041302.

[Sch+05] R. Schwenn et al. "The association of coronal mass ejections with their effects near the Earth." In: *Ann. Geophys.* 23 (2005), pp. 1033–1059. DOI: 10.5194/angeo-23-1033-2005.

[PH06a] A. Perrin and H. H. Hu. "An Explicit Finite-Difference Scheme for Simulation of Moving Particles." In: *Journal of Computational Physics* 212.1 (2006), pp. 166–187.

[PH06b] A. Perrin and Howard H. Hu. "An Explicit Finite-Difference Scheme for Simulation of Moving Particles." In: *Journal of Computational Physics* 212.1 (2006), pp. 166–187. DOI: 10.1016/j.jcp.2005.06.021.

[Gop+09] N. Gopalswamy et al. "The Expansion and Radial Speeds of Coronal Mass Ejections". In: *Central European Astrophysical Bulletin* 33.1 (2009), pp. 115–124. DOI: 2009CEAB...33..115G.

[Jac09] Z. Jackiewicz. *General Linear Methods for Ordinary Differential Equations*. John Wiley Sons Inc, 2009.

[Tor09] Eleuterio F. Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*. Springer, 2009.

[LD10] Cheng Lin and Tang Dengbin. "Navier-Stokes Characteristic Boundary Conditions for Simulation of Some Typical Flows." In: *Applied Mathematical Sciences* 4.18 (2010), pp. 879–893.

[And12] Bengt Andersson. *Computational Fluid Dynamics for Engineers*. Cambridge University Press, 2012. DOI: https://doi.org/10.1017/CBO9781139093590.

[Gop+12] N. Gopalswamy et al. "The Relationship Between the Expansion Speed and Radial Speed of CMEs Confirmed Using Quadrature Observations of the 2011 February 15 CME". In: *Sun and Geosphere* (2012). URL: arXiv:1205.0744.

[SLB12] Prasad Subramanian, Alejandro Lara, and Andrea Borgazzi. "Can solar wind viscous drag account for coronal mass ejection deceleration?" In: *Geophysical Research Letters* 39 (2012). DOI: https://doi.org/10.1029/2012GL053625.

[Vrs+12] B. Vrsnak et al. "Propagation of Interplanetary Coronal Mass Ejections: The Drag-Based Model". In: *Solar Phys* 285 (2012), pp. 295–315. DOI: 10.1007/s11207-012-0035-4.

[RM14] A. Romano and a. Marasco. *Continuum Mechanics using Mathematica*. Birkhauser, 2014.

[Dum+18] Mateja Dumbovich et al. "The Drag-based Ensemble Model (DBEM) for Coronal Mass Ejection Propagation". In: *The Astrophysical Journal* 854.2 (2018). DOI: https://doi.org/10.3847/1538-4357/aaaa66.

[Nap+18] G. Napoletano et al. "A probabilistic approach to the drag-based model". In: *J. Space Weather Space Clim.* 8 (2018). DOI: https://doi.org/10.1051/swsc/2018003.

[Lag] P.-Y. Lagrée. *Small Re flows, $\varepsilon = Re << 1$*. URL: http://www.lmm.jussieu.fr/~lagree/COURS/M2MHP/petitRe.pdf.

[VZ] B. Vrsnak and T. Zic. *THE DRAG-BASED MODEL (DBM) with constant solar wind speed*. URL: https://ccmc.gsfc.nasa.gov/models/CDBM_info.pdf. (accessed: 30.09.2019).