UNIVERSITY OF PALERMO
PHD JOINT PROGRAM:
UNIVERSITY OF CATANIA - UNIVERSITY OF MESSINA
XXXVI CYCLE

DOCTORAL THESIS

Enhancing Computational Fluid Dynamics with Artificial Intelligence: an AI-based Smoothed Particle Hydrodynamics (SPH) Emulator for Lava Flow Modeling

Author:
Eleonora AMATO

Supervisor:
Prof. Gaetana GAMBINO

PhD Coordinator:
Prof. Maria Carmela LOMBARDO

Co-Supervisor:
Dr. Ciro DEL NEGRO

A thesis submitted in fulfillment of the requirements
for the degree of Doctor of Philosophy

in

Mathematics and Computational Sciences

Declaration of Authorship

I, Eleonora AMATO, declare that this thesis titled, “Enhancing Computational Fluid Dynamics with Artificial Intelligence: an AI-based Smoothed Particle Hydrodynamics (SPH) Emulator for Lava Flow Modeling” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: Eleonora AMATO

Date: February 17, 2024

“Per tre cose vale la pena di vivere:
la matematica, la musica e l’amore.”

(Renato Caccioppoli, 1904-1959)

UNIVERSITY OF PALERMO

Abstract

Department of Mathematics and Computer Sciences

Doctor of Philosophy

Enhancing Computational Fluid Dynamics with Artificial Intelligence: an AI-based Smoothed Particle Hydrodynamics (SPH) Emulator for Lava Flow Modeling

by Eleonora AMATO

The combination of Computational Fluid Dynamics (CFD) and Artificial Intelligence (AI) expands the scope of fluid modeling, providing high fidelity and fast simulations. A variety of Eulerian CFD methods integrated with AI has been already successfully presented (*e.g.*, for weather forecasting); on the other hand, the combination of AI and Lagrangian methods remains less consolidated. Smoothed Particle Hydrodynamics (SPH) is a Lagrangian mesh-less CFD numerical method, highly reliable for the simulation of complex fluids. Nevertheless, SPH models exhibit limitations in high-resolution real-time simulations of physical phenomena, due to the high computational costs involved. Specifically, SPH simulations of lava flows are well representative of the difficulties in modeling highly complex fluids. Lava is a fluid with a high physical complexity, generating viscous flows, dependent on temperature and rheology, and it may have significant impacts on the surrounding environment. Thus, it is important to monitor lava flows with accurate and timely forecasting of their spatio-temporal evolution. Here, I present an emulator derived from CFD physics-based models, in which AI algorithms join the equation-based mathematical representation of physics, to solve fluid dynamics problems in shorter times. I developed an AI-based emulator for SPH method, in which the conservation of momentum equation is substituted by an Artificial Neural Network (ANN), which learns from SPH simulations. The ANN is trained to estimate SPH particles interaction forces exploiting as input the state of the particles (position, velocity, density). I verified the reliability of the AI-based emulator to remain as faithful as possible to the SPH reference model. Applications to different kind of fluids are shown, starting from an inviscid fluid, up to the study of a viscous fluid with a thermal component, to finally move towards the description of a lava flow evolution, exploiting the potential of the combined use of numerical and AI models. Simulations and emulations have been compared for each step, reaching a high degree of fidelity, and demonstrating the generalizability of the AI-based emulator, tested over problems with varying levels of complexity, and its robustness to different spatial resolutions.

Acknowledgements

This work was developed within the framework of the Laboratory of Technologies for Volcanology (TechnoLab) at the INGV – National Institute of Geophysics and Volcanology, Etna Volcano Observatory in Catania (Italy).

I am grateful to Dr. Ciro Del Negro, who directed my PhD research at INGV, following every step of my study, guiding my work, and training me as a researcher. I want to also thank Dr. Claudia Corradino and Dr. Vito Zago, INGV researchers who followed and mentored me with perseverance and attention, accompanying me on this journey, working with me, and contributing intensely to my growth. A huge thank you goes to my university Professor Maria Carmela Lombardo, who carefully guided and followed my journey, always believing in me, being present, and supporting me from both a professional and personal point of view. Many thanks also go to my Professor Gaetana Gambino, university tutor, for all the interest always shown in my activities and for actively participating in my doctoral path.

Thanks to my INGV team colleagues and to all my colleagues at INGV, with whom I shared a lot of work, trips for conferences, lunches, coffees, and free moments, making this stay in Catania full of exciting experiences. A huge thank you goes to Domizia, my housemate in Catania, for always being present, believing in me, and cheering for me, making me feel at home. Thanks also to all my university colleagues because, in addition to sharing the degree path, we also shared this intense doctoral journey, always comparing ourselves and celebrating every goal achieved.

Thanks to my lifelong friends, who have always known me and supported me in all the moments that filled the PhD period. Thanks also to my big friend Riccardo, the first colleague I met but much more than a colleague, a great, great friend. Thank you for all the moments spent together, for always sharing our infinite passion for mathematics, and for always being a great point of reference for me.

Thanks to all my family, grandparents, uncles, and cousins. Thank you for always supporting me, following me on this journey, and rejoicing with me in my results. And the most important thanks go to them: Mom, Dad, and my sister Claudia. Special thanks to Claudia, who understood me with just a glance and supported me, accompanying me in every moment. Thank you for always expressing how proud you are of me. Thanks to Mom and Dad for always being by my side, consistently present in all my choices. Thank you for being my guides and teaching me everything. I am grateful to the three of you, because you make everything so full of love.

Contents

List of Figures

Introduction

Mathematical modeling is the process of theoretically describing real-world phenomena in mathematical terms, with equations capable to describe their spatio-temporal evolution (to its best approximation). A model is a representation of the reality, and a mathematical model is composed of a finite set of equations that describes a finite set of variables. Models are idealization of the real phenomena, with some limitation, they can be continuous or discrete, and they can be formulated using exact conservation laws, empirical conservation laws (that depend on the specific physical phenomenon), or hypothetical laws (based on qualitative rules, for example of assumptions that are not based on quantitative laws) (Edwards and Hamson, 1989; Fowler, 1997). Especially for physical phenomena, a mathematical model must be composed of governing equations, sub-models for specific cases (defining and constitutive equations) and relative assumptions and constraints (initial and boundary conditions, classical constraints and kinematic equations). The idea is thus to follow a process in which a problem is identified (with specific variables and constants), a modeling procedure is selected, and a model is formulated. It is then solved to give an answer to the initial physical process, finally verifying it (Meerschaert, 2013).

Throughout mathematical history, the mathematical modeling have been applied to solve several problems in various fields such as biology, chemistry, physics, astrophysics, nuclear physics, and geophysics (Velten, 2009). In all these fields, it is possible to study the mechanics of the fluids that characterize them and the branch of fluid mechanics that studies the behavior of fluids in movement is fluid dynamics. In it, motion and interactions between fluids are studied, with the aim to describe their natural behavior (McCormack, 2012).

The fluid physical aspects are governed by three fundamental principles, that are, the Newton's second law, $F = ma$, with F force, m mass and a acceleration, the principle of mass conservation, and the principle of energy conservation. These principles can be expressed in terms of mathematical governing equations, that can describe and model a system and its evolution (thus, often they are ordinary or partial differential equations or integral equations). The governing equations for fluid dynamics are the Navier-Stokes equations (Temam, 2001).

Navier-Stokes equations completely describe any fluid flow, including the turbulent behavior. Correlated by initial and boundary conditions, in specific cases, they give the fluid velocity field as solution, from which to obtain other quantities that characterize the fluid flow. In the general case, they are composed of five scalar partial differential equations and twenty variables. Due to their complexity and non-linearity, the Navier-Stokes equations almost never admit an analytical solution (*i.e.*, an exact solution). Therefore, a fundamental problem is to develop a mathematical theory to understand and analyze these equations, determining whether they are well-posed, and finding the analytical closed form solution in the general case. In the context of the Clay Mathematical Institute millennium problems (Carlson, Jaffe, and Wiles, 2006), this represents of them (Lemarié-Rieusset, 2018).

Focusing now in the specific Navier-Stokes solvable cases, however, they are complex to be solved, both in terms of resources and calculation times. A way to overcome this problem is to discretize the analytical equations under stability, convergence and consistency conditions (Anderson and Wendt, 1995; Chung, 2002), obtaining a numerical formulation of the same model, that can be solved with less computational costs. Computational Fluid Dynamics (CFD) is the modeling branch to numerically solve fluid dynamics models, recently in great development for the study of fluids.

CFD, therefore, can be used to simulate complex fluids, with high viscosities, thermal components and vorticity. An example of this kind of fluid is lava. Varying physical and rheological parameters, and considering different topographies on which the lava fields evolve, they may travel long distances, overcoming barriers and cooling, and becoming destructive for inhabited areas. Some volcanoes have highly urbanized areas nearby, therefore, it is important to monitor lava flows, following their behavior in space and time. Physical-mathematical models can be used to make accurate predictions of the lava flows evolution reducing in field risks linked to direct analysis, and helping in monitoring and mitigating damages linked to eruptions. However, lava is characterized by complex physical and rhelogical properties, which make it difficult to faithfully simulate in near real-time its flow with analytical models. In cases like this, the use of CFD helps in studying complex flow progress in real-time, predicting possible evolution, and deepening the understanding of physical phenomena.

Moreover, CFD models place limits in high-resolution real-time simulations for complex physical phenomena, due to the high computational costs required. Indeed, increasing the complexity of the system to be simulated, the discretization of analytical models produces highly complex numerical models, with a large number of operations to process and variables to take into consideration. Currently, the development of approaches that combine advantages of physics-based numerical models with intelligent techniques are pushing forward the frontiers of traditional CFD, increasing their fields of application, and opening up to new forms of detailed analysis (Kochkov et al., 2021; Morita et al., 2022).

Artificial Intelligence (AI) (Goodfellow, Bengio, and Courville, 2016; Bonaccorso, 2017) is the branch of computer science that studies systems that learn from huge volume of data, becoming capable of reasoning and solving tasks autonomously. Artificial Intelligence is mainly divided into two classes, Machine Learning (ML) (Bonaccorso, 2017) and Deep Learning (DL) (Goodfellow, Bengio, and Courville, 2016). In ML, the model learns from structured data, in which data features are selected and extracted during the input feature engineering phase, to be opportunely representative of the real phenomenon. In DL, the model learns from unstructured data, as raw images, exploiting the spatial and/or temporal relationships between neighboring data to automatically extract the features useful to describe the phenomenon under analysis. Therefore, DL is a subset of ML. In both cases, the input data are analyzed, extracting some basic information, which is then passed on to the next training step, which elaborates them in a deeper way, extracting significant information about the data. The use of AI models, therefore, allows data to be analyzed in a semi-automatic or fully automatic fashion. After an initial training phase of the ML and DL models, these become capable of identifying patterns or relationship within the processed data, extrapolating useful information for data analysis. The use of AI is widely spreading in every field of research, in particular in those that involve the analysis of a great amount of data in a short time, favoring their automatic processing.

Here, I exploit knowledge in CFD and AI fields to build emulators (Bortnik and Camporeale, 2021; Sofos et al., 2022). They are derived from physics-based models, based on CFD, capable to reproduce their behavior generally by learning a particular dynamic from their simulated data. Their aim is to describe the spatio-temporal evolution of a physical phenomenon with better performances. Emulators open to new levels of analysis (Raissi, Perdikaris, and Karniadakis, 2019; Ummenhofer et al., 2019), advancing functionalities (Vinuesa and Brunton, 2022), and allowing to deepen the understanding of physical phenomena (Mohaghegh et al., 2022). A general way to implement an emulator is to start with a CFD model, in which some blocks of the numerical approach are substituted by or joined with an AI model, trained to learn the CFD behavior, remaining as faithful as possible to the CFD formulations and speeding-up the simulation times (Kasim et al., 2021). To develop these models, I set myself two preliminary objectives. Emulators must show completeness and correctness, reproducing the studied phenomenon in a physically faithful manner, and they must produce fast reproductions of the phenomenon with respect to the analytical reference models, opening up to simulations of phenomena that are not always accessible.

Therefore, I present a CFD emulator, derived from the Navier-Stokes equations, developing its architecture, and validating and testing it over different case studies aimed, finally, at application in the volcanic field. As numerical method, I chose the Smoothed Particle Hydrodynamics (SPH), a particle-based, Lagrangian and mesh-free CFD method, that allows high fidelity simulations of complex fluids (Monaghan, 2005). Due to the heavy computational costs of the SPH method, I combined it with an Artificial Neural Network (ANN), an AI model capable of appropriately approximating any kind of functions, well generalizing to unknown cases. In order to finally reproduce lava flows, particular attention has been paid to fluids with viscous and thermal components. Starting from analytical equation-based SPH simulations, that discretize the Navier-Stokes equations, a set of reference simulated data is sampled, used to train the ANN model. Thus, the AI-based SPH emulator is ready to be used to emulate the SPH fluid system, describing and reproducing the dynamics of the physical phenomenon.

The general pipeline of emulator development follows a three-step approach. Considered a physical problem whose dynamics must be described, the first step involves the SPH formulation of the problem, to understand limits and where to intervene. Therefore, the second step implies the analysis of Artificial Intelligence techniques, to understand in what they can be of support and, therefore, how to intervene. Putting together the knowledge obtained, finally, in the third step, CFD and AI are combined together in the concept of the emulator, to provide a solution to the complexity problems that have been discussed.

The product of the physics-based model is a simulation, *i.e.*, an equation-based reproduction of the dynamics of a fluid; the product of an emulator is an emulation, *i.e.*, an AI-CFD-based reproduction of the dynamics of a fluid. Therefore, a simulation is the reproduction of a real phenomenon totally based on the governing laws that model the phenomenon itself, and it is produced by a discretized version of them; an emulation is the reproduction of a phenomenon, still based on an analytical or numerical model that reproduce it, but in which a part of them is made up of an intelligent model, capable of learning to replicate the governing laws and extrapolating specific laws. The final object is therefore always a representation of reality, and in the first case this takes into account the original physical governing laws, in the second case it comes from their reproduction.

I will apply this approach to different kinds of fluids, validating the capability of the emulator to reproduce their complex dynamics. I will begin with a study of a Newtonian, inviscid and isothermal fluid, and I will progressively increase its complexity. In particular, I will add the viscous component and, then, the thermal model. For all these cases, a study on the feasibility of the problem will be conducted (*Is it possible to emulate a phenomenon with these characteristics?*), simulating the fluid using equation-based models and, subsequently, reproducing it with the emulator, comparing the results. The ability to generalize for cases never seen during the model construction phase and for changes in the model resolution will also be verified. Moreover, studies on some specific features will be conducted, such as the reproduction by the emulator of boundary conditions, and the management of different topographies. Finally, moving towards lava flows simulations, a complex case study will be addressed, that is a look toward an emulator capable of reproducing a highly viscous fluid with thermal behavior typical of lava flows (Giordano and Dingwell, 2003).

Such an approach opens the way to analyzing the dynamics of complex fluids, following their progress in time and space, and considering their interactions with the surrounding environment. Therefore, the combined use of CFD models and AI can enrich the panorama of modeling tools that allow a reliable and fast analysis. This leads going into the study of phenomena that cannot be considered with other methods, better understanding the physics of certain natural events. As mentioned before, applications can be found in different research fields, including the volcanic studies, in which it is important to have reliable and timely simulations, to be able to manage a crisis in time, minimizing damages to people and infrastructures.

This thesis is organized as follows. After this Introduction, Chapter 1 presents mathematical models for fluid dynamics, based on observations or on the governing laws, with the analytical and numerical formulation. Thus, Computational Fluid Dynamics is treated, and the Smoothed Particle Hydrodynamics CFD technique is explored, with possible applications and limits. Chapter 2 presents the Artificial Intelligence models, with the description of some properties, specific components and different techniques. Then, a brief excursus on the state-of-the-art models is shown, with some applications to real data (in particular in the volcanic context), for extrapolating input parameters of the mathematical models using AI and data from Space. Chapter 3 presents the combination of CFD and AI models in the emulators. A brief state-of-the-art of the AI-CFD models is followed by the definition of the AI-based SPH emulator here developed, with the specific technical characteristics. Chapter 4 shows the results of the developed and tested emulator, discussing them. In particular, each case study is analyzed, comparing simulations and emulations and verifying the generalizability of the emulator, laying the foundations for future developments. Finally, in the Conclusion, I sums up all the study, commenting on the results obtained. In addition, two appendices have been included. Appendix A, linked to Chapter 1, shows more in detail the theoretical basis of the SPH method, in particular the mathematical derivation of the SPH approximation and of the spatial gradients discretization, and some attributes are indicated, as kernel choice and SPH initial and boundary conditions setting. Appendix B, linked to Chapter 2, delves into the data feed to AI models to extrapolate the input parameters of the mathematical models.

Chapter 1

Modeling

1.1 Mathematical models for fluid dynamics

Physics is highly complex. Due to the intricate nature of physical phenomena, mathematical models are a key factor for the development of many scientific and technological fields, from fundamental research to applied sciences, helping with accurate predictions of the physical systems evolution.

Nowadays, various mathematical models have developed (Velten, 2009), and one of the widespread and complex field of research in mathematical physics is fluid dynamics. It is the field of physics interested in motion and interactions between fluids (McCormack, 2012), with the aim to describe and reproduce their natural behavior, giving predictions under specific initial and boundary conditions (Dym, 2004; Basmadjian, 2003).

All the physics-based dynamical models require several input parameters to tune the simulations based on models, with respect to the observed natural phenomena in space and time. The data fed to the model can be sampled with experiments in laboratory, with measurements in field or using the remote sensing techniques, that include the use of aircraft, drones, and satellites images from Space. Depending on the final aim of the physical simulations, different kinds of models can be used, from analytical to numerical ones. Models of the first type describe in detail the theoretical dynamics of a system, in a continuous space, and they produce exact solutions. They are not always solvable, making necessary the use of different kinds of models. Therefore, models of the second type discretize in a more simplified version the physical space of the system, obtaining approximations of the exact solution. Approximately in the middle, there are the empirical models, derived from observations, very fast but less accurate. In addition, to monitor a physical phenomenon, the models can be classified also as physics-based or data-driven (Bortnik and Camporeale, 2021). The first area includes analytical or numerical models that are totally based on the physical laws that describe the system, and they require as input the physical parameters to build the model. On the other hand, the second area includes all the models based on the data, and they are built by fitting the data given in input (with statistical or intelligent models), that represent the relative phenomenon, extracting information from them to describe it. Between the physics-based and the data-driven models, there are the combined-ones, that incorporate the physical models with the sampled data and the artificial intelligence to describe a phenomenon. All these models differ in term of accuracy and speed, so that if one increases the other decreases.

In modern times, with the improvement of computer hardware and software, numerical modeling has progressed greatly and it has become widespread for computer simulations. The class of numerical techniques in fluid dynamics is the Computational Fluid Dynamics (CFD) (Anderson and Wendt, 1995). This is the branch of fluid dynamics to numerically solve the relative constitutive equations, and CFD models are able to produce faithful simulations of the fluids behavior. Its downside is that, being very reliable, therefore resorting to advanced calculation techniques, CFD has high computational costs. The balance in level of accuracy and description of the results and in execution times should be selected according to the purpose of the study.

Complex fluids: the lava case study An example of a complex fluid, with a high physical complexity and several impacts on human life, is *lava* (Sigurdsson et al., 1999).

This complex fluid has the ability to reach inhabited areas. The growing urbanization of areas near a volcano increases the threats linked to the lava flows. They have, usually, slow flows and they do not threaten human lives directly. With the variation of the main parameters that characterize the eruption (as rheology and effusion rate), and considering different topographies on which the lava fields evolve, they may travel long distances, overcoming barriers and cooling, with the formation of channels and tunnels. In these cases, this fluid may become destructive for inhabited areas, harming people and infrastructures (Del Negro et al., 2020). For these reasons lava flows need to be studied. However, lava is characterized by several factors determining its behavior, including specific properties of the fluid, such as very high viscosities, thermal components, a non-Newtonian rheology, and solidification processes, and specific properties of the flow such as topography, effusion rate, with the presence of a fixed inlet with variable flow rate, solid-fluid interactions, phase transitions and free-surface flows (Giacomelli and Scandone, 2004). Because of the complex physics of lava and the importance to monitor its progress in space and time, physical-mathematical models can be used to make accurate predictions of the lava flows evolution, helping to monitor hazards and mitigate damages without in field risks. In detail, in volcanic field and in lava fields studies, high fidelity simulations can help on two fronts, to estimate volcanic hazards and to mitigate damages (Cordonnier, Lev, and Garel, 2016). For the first scope, simulations can help in building forecast scenarios of the possible extension and flooded areas with the connected dangers, without in field risks. For the second one, these scenarios can guide the planning of mitigating actions, in order to manage and reduce the damages. Some of the main input parameters are the topography of volcanic area, the eruptive initial and boundary conditions, the lava thermal conditions, and the physical properties of the lava flow (Cordonnier, Lev, and Garel, 2016), and one of the increasingly widespread data acquisition techniques is the use of remote sensing sensors on board drones or satellites (Harris, 2013). Different techniques can be used to process this kind of data (to extract physical parameters linked to the volcano activity), as automatic models, that automatically analyses data and extract information; for more details, see Chapter 2.

Many modeling approaches can be used to simulate lava flows, and fluids in general, with the basic idea of having a model capable of reliably simulating fluids in the shortest time possible, thus obtaining accurate and fast predictive scenarios. To monitor a physical phenomenon, as the volcanic activity, therefore, the techniques can be more physics-based or totally data-driven, or a combination of the two approaches. Here, different types of models will be presented, from the more physics-based up to data-driven models, described in the following chapter, through which the physical input parameters can be extracted or volcanic quantitative analysis can be done. The combined models will be seen in the following chapters, finally obtaining fluids simulations.

This chapter, therefore, will present empirical, analytical, and numerical models for fluid dynamics.

1.2 Physical models

In this section, the different types of physical models, from the ones based on observations, to the one with accurate physical meaning and higher computational costs (high accuracy, low speed), and the ones with high calculation speeds and low physical completeness (low accuracy, high speed), to be chosen based on the fields of application, will be discussed.

1.2.1 Empirical models

An empirical model is a model derived from data observations, measurement and practical experience, so only on experimental data, looking for relationships between the final results and the initial and boundary conditions, without a priori information about the system (Velten, 2009). Empirical models are simplified version of the analytical ones, sometimes losing in the detailing of the system, not always being able to take into account all those real phenomena that require a high descriptive accuracy of the fluid. They are, however, representative of the system and easy to be resolved, with low computational costs.

An example of this kind of models for the volcanic case study seen before is the empirical model for the maximum distance of a lava flow (L), in Eq. 1.1

$$L = 3.11\ E^{0.47}, \tag{1.1}$$

with L the final length of the single channel-fed flow [m] and E the mean discharge rate [$\mathrm{m}^3\ \mathrm{s}^{-1}$] averaged for the whole time of an individual active flow (Calvari and Pinkerton, 1998). This model was calibrated using data related only to the Mt.Etna volcano (Sicily, Italy). An improved version of this model is the following one, Eq. 1.2

$$L = 1.747\ E^{0.47}\ t^{0.29}\ \alpha^{0.26}, \tag{1.2}$$

with t the total emplacement time of the flow [s], and α the mean ground slope [rad] (Calvari and Pinkerton, 1998).

Clearly, a similar model can be obtained for each volcano worldwide, calibrating the parameters with specific data. Nevertheless, this can not be generalized to a theoretical case.

1.2.2 Analytical models

An analytical model is a quantitative representation of the reality, based on principles or theory, thus on mathematical formulas that are solved using classical methods, such as calculus or algebra. In this case, the system behavior can be theoretically studied, obtaining a related analytical formulation (governing laws) and (when possible) an exact solution, expressed in terms of well-known functions (Velten, 2009). An ordinary or partial differential equation of order $n \in \mathbb{N}$, under specific conditions, admits one and only one solution (*Cauchy Problem* and *Cauchy-Kowalevski theorem*, (Bers, John, and Schechter, 1964; Petrovsky, 2012)). Models like these need much work to be developed and solved.

Cauchy problem (simple formulation) Let $f : \mathbb{R} \times \mathbb{R} \longrightarrow \mathbb{R}$ be a function, f is an analytic function in a neighborhood of the point $(x_0, y_0) \in \mathbb{R} \times \mathbb{R}$,

$$\frac{dy}{dx} = f(x, y(x)) \tag{1.3}$$

with

$$y(x_0) = y_0, \tag{1.4}$$

initial condition, then $\exists!\, y(x)$ that satisfies Eqs. 1.3 and 1.4 in some neighborhood of x_0. □

Cauchy problem Let Eq. 1.5 be a set of partial differential equations of unknown functions $u_1(t, x_1, ..., x_n), ..., u_N(t, x_1, ..., x_n)$,

$$\frac{\partial^{n_i} u_i}{\partial t^{n_i}} = f_i \left(t, x_1, ..., x_n, u_1, ..., u_N, ..., \frac{\partial^k u_j}{\partial t^{k_0} \partial x_1^{k_1} ... \partial x_n^{k_n}}, ... \right) \tag{1.5}$$

with $i, j \in \mathbb{N}$, $i, j \in [1, N]$, $k_0 + k_1 + ... + k_n = k \leq n_j$, $k_0 < n_j$. For $t = t_0$, let Eq. 1.6 be the initial conditions,

$$\frac{\partial^k u_i}{\partial t^k} = \phi_i^{(k)}(x_1, .., x_n) \tag{1.6}$$

for $k \in \mathbb{N}$, $k \in [0, n_i - 1]$, with $\phi_i^{(k)}$ prescribed in the same region of the space $(x_1, ..., x_n)$ in which the equation solutions must be found, the Cauchy problem allows to find the unknown functions $u_1(t, x_1, ..., x_n), ..., u_N(t, x_1, ..., x_n)$ that satisfy Eqs. 1.5 and 1.6 (Petrovsky, 2012). □

Cauchy-Kowalevski theorem If all the f_i functions are analytic in some neighborhood of $(t^0, x_1^0, ..., \phi^0_{j,k_0,k_1,...,k_n}, ...)$ and if all the $\phi_j^{(k)}$ functions are analytic in some neighborhood of $(x_1^0, ..., x_n^0)$, then the Cauchy problem has a unique analytic solution in some neighborhood of $(t^0, x_1^0, ..., x_n^0)$ (Petrovsky, 2012). □

At the basis of analytical models there are the governing equations, that describe how some state variables vary in relation to other variables. For fluid dynamics, the governing equations are the Navier-Stokes ones (Temam, 2001).

Looking in detail at a classic set of equations for fluid dynamics, the continuity equation, which expresses the law of conservation of mass, is the following Eq. 1.7

$$\frac{D\rho}{Dt} = -\rho \nabla \cdot \vec{u}, \tag{1.7}$$

where ρ is the density, t is the time, and $\vec{u}$ the velocity. The operator $\frac{D}{Dt}$ is the Lagrangian or material (total) derivative,

$$\frac{D}{Dt} = \frac{\partial}{\partial t} + \vec{u} \cdot \nabla.$$

The conservation of momentum is expressed by the Navier-Stokes equation that, for incompressible fluids ($\frac{D\rho}{Dt} = 0$), is the following Eq. 1.8

$$\rho \frac{D\vec{u}}{Dt} = -\nabla P + \mu \nabla^2 \vec{u} + \vec{G}, \tag{1.8}$$

with $\nabla \cdot \vec{u} = 0$, that can be also written as

$$\rho \frac{D\vec{u}}{Dt} = -\nabla P + \nabla \cdot (\mu \nabla \vec{u}) + \vec{G}, \tag{1.9}$$

when the viscosity is not homogeneous, and it can spatially variate.

Finally, the Navier-Stokes equation for the conservation of momentum for compressible fluids is the following Eq. 1.10

$$\rho \frac{D\vec{u}}{Dt} = -\nabla P + \mu \nabla^2 \vec{u} + \frac{1}{3} \mu \nabla (\nabla \cdot \vec{u}) + \vec{G} \tag{1.10}$$

The pressure P, obtained from the density and not from the Poisson equation, derives from the Cole's equation of state (Cole, 1948), and it can be expressed as Eq. 1.11

$$P(\rho) = c_0^2 \frac{\rho_0}{\gamma} \left(\left(\frac{\rho}{\rho_0} \right)^{\gamma} - 1 \right), \tag{1.11}$$

where c_0 is the speed of sound, ρ_0 is the reference density of the fluid at rest, and γ is the polytropic constant.

The fluid thermal evolution is described by the heat equation, as in Eq. 1.12

$$\frac{DT}{Dt} = \frac{1}{c_p \rho} \nabla (\kappa \nabla T), \tag{1.12}$$

where T is the temperature [K], c_p the specific heat at constant pressure [J kg^{-1} K^{-1}], and κ the thermal conductivity [W m^{-1} K^{-1}].

Rheology Rheology concerns the study of the deformation and flow behavior of materials, linked to the fluids stress–strain behavior (Velten, 2009; Zago, 2019). The strain rate $\dot{\gamma}$ [s^{-1}] is defined as

$$\dot{\gamma} = \frac{1}{2} [\nabla \vec{u} + (\nabla \vec{u})^T], \tag{1.13}$$

and the shear stress τ [Pa] is defined using the linear stress constitutive equation 1.14,

$$\tau = \lambda (\nabla \cdot \vec{u}) \vec{I} + 2\mu \dot{\gamma}, \tag{1.14}$$

with $\lambda \neq 0$ bulk viscosity, and μ dynamic viscosity (Lamé parameters) (Zago, 2019). Here, incompressible fluids are treated, with $\nabla \cdot \vec{u} = 0$. If the shear stress and the strain rate have a linear proportionality (with μ constant of proportionality), the fluid will be named Newtonian (*e.g.*, the water behavior). Otherwise, the fluid will be named non-Newtonian, and, in this case, it is possible to define $\tau = f(\dot{\gamma})$, with f a generic non linear function (*e.g.*, blood, toothpaste, or lava).

An example of non-Newtonian case is the Herschel–Bulkley rheology (Hérault et al., 2011), in which

$$\begin{aligned} \dot{\gamma} &= 0, \text{ if } \tau < \tau_0 \\ \tau &= \tau_0 + k\dot{\gamma}^n, \text{ if } \tau \geq \tau_0 \end{aligned} \tag{1.15}$$

with τ_0 the yield strength [Pa], k a consistency index [Pa s^n], n the flow index (Zhu, Kim, and De Kee, 2005; Hérault et al., 2011).

From these relationships, the following fluids classification is obtained:

- $\tau_0 = 0,\ n = 1 \implies$ Newtonian fluid;
- $\tau_0 = 0,\ n \neq 1 \implies$ Power-law fluid: if $n > 1$, *Dilatant*, if $n < 1$, *Pseudo-plastic*;
- $\tau_0 \neq 0,\ n = 1 \implies$ Bingham fluid.

Some non-Newtonian fluids can be restricted to a subset, named generalized Newtonian fluids, in which it is possible to define an effective viscosity function μ_{eff}, which can then be explained in the Navier-Stokes equations, as $\mu_{eff} = \mu_{eff}(\dot{\gamma})$, such that $\tau = \mu_{eff}(\dot{\gamma})\dot{\gamma}$. For this type of fluids the same form of the Navier–Stokes equations can be still applied.

A Herschel–Bulkley fluid can be traced back to a generalized Newtonian rheology, for example using the regularization in (Zhu, Kim, and De Kee, 2005), and defining an effective viscosity in the following way

$$\begin{aligned} \mu_{eff} = \mu_{eff}(\dot{\gamma}) = \frac{\tau_0}{|\dot{\gamma}|} + k|\dot{\gamma}|^{n-1}, \text{for} |\tau| > \tau_0 \\ |\dot{\gamma}| = \sqrt{[\gamma_{xx}^2 + \gamma_{yy}^2 + \gamma_{zz}^2 + 2(\gamma_{xy}^2 + \gamma_{xz}^2 + \gamma_{yz}^2)]/2}. \end{aligned} \tag{1.16}$$

1.2.3 Numerical models

A numerical model is based on a discretization of the analytical model, for which numerical processes are used to solve a set of equations, applying different iterative numerical algorithms to reach the approximated solution for a specific set of parameter values (Velten, 2009). They are an approximation of complex analytical models, and they could require expensive and complex software and hardware to be developed for complex simulations.

Analytical models are not always solvable (sometimes it is difficult, for example, to define the boundary conditions). An example is the case of the millennium problems (Carlson, Jaffe, and Wiles, 2006), seven mathematical problems (one of which has already been solved) brought to the attention of mathematicians by the Clay Mathematical Institute, awarded a prize of one million dollars for their solution. One of these, for example, involves the Navier-Stokes equations; the problem is precisely to develop a mathematical theory that allows to understand and analyze these equations, determining whether they are well posed and finding the analytical solution in closed form in the general case (Lemarié-Rieusset, 2018). Then one way to solve some analytical problems may be to solve them numerically, discretizing them under specific conditions, which may however require high computational costs. Discretizing the system, under the hypotheses of existence and uniqueness of the analytical solution, the numerical solution can be determined, if the model fulfills the conditions of Consistency, Convergence and Stability (Anderson and Wendt, 1995; Chung, 2002). Briefly,

- Consistency condition involves that the discrete operator tends to the differential operator as the discretization step tends to zero;
- Convergence condition involves that the approximate solution tends to the true solution as the discretization step tends to zero;
- Stability condition involves that, considered two different solutions u_a and u_b of the problem, obtained by considering two different initial vectors u_{0_a} and u_{0_b}, it exists $\delta \in \mathbb{R}$, $\delta > 0$, $\delta \in [0, \delta_0]$ $(\delta_0 > 0)$ such that $\|u_a - u_b\| \leq$

$\delta \| u_{0_a} - u_{0_b} \|$, that is the system is robust to perturbations (of the initial state), thus, the effect of a perturbation is limited in the solution.

If fluid dynamics is the analytical physics field interested in motion and interactions between fluids (Batchelor, 1967; Lamb, 1994[1932]; McCormack, 2012), Computational Fluid Dynamics (CFD) is the numerical branch of fluid dynamics to simulate the behavior of fluids using numerical techniques (Anderson and Wendt, 1995; Chung, 2002). With the continued diffusion and growth of computers, in the last decades, CFD has opened to great advancements in the study of fluids.

Classification of numerical models

The numerical models can be classified as Eulerian or Lagrangian, mesh-free or mesh-based, based on the type of discretization (interpolation points) applied, as follows.

Eulerian The observed object (interpolation point) is a specific part of the domain, fixed in space, through which the system evolves. Therefore, the Eulerian methods discretize the simulation domain into volumes (cells), which remain fixed in space (Batchelor, 1967; Lamb, 1994[1932]) (see Fig. 1.1).

This approach is good for modeling diffusion systems, and examples are the Finite Volumes (FV), or Finite Differences (FD).

Lagrangian The observed object (interpolation point) is a single portion of the fluid, so the nodes of the discretizations moves in space over time. Therefore, in Lagrangian methods the fluid is discretized into small volumes which can move in space, tracking their state in time (Batchelor, 1967; Lamb, 1994[1932]) (see Fig. 1.1).

This approach is good for modeling deformable complex surfaces and an example can be the Finite Element Method (FEM).

Mixed Eulerian-Lagrangian Arbitrary Lagrangian–Eulerian methods (ALE) combine the two approaches, so that it can be possible to handle large deformations (like for pure Lagrangian methods) with higher resolution (like Eulerian methods) (Hirt, 1974; Donea et al., 2004). In this case, for example, the fluid can have an Eulerian treatment and the solid domain a Lagrangian one (Legay, Chessa, and Belytschko, 2006), or, for liquid-gas interactions, the flow of the gas mixture can discretized in an Eulerian way, and the liquid-droplet system in a Lagrangian one, reducing the computational resources linked to the resolution of each droplet (Melnikova, Epikhin, and Kraposhin, 2021).

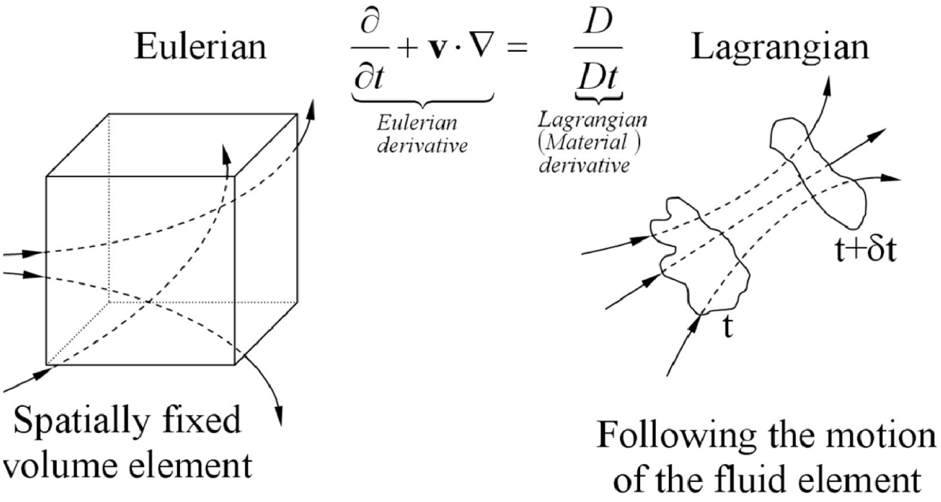

Eulerian
$\underbrace{\frac{\partial}{\partial t} + \mathbf{v} \cdot \nabla}_{\text{Eulerian derivative}} = \underbrace{\frac{D}{Dt}}_{\text{Lagrangian (Material) derivative}}$
Lagrangian
t+δt
t
Spatially fixed volume element
Following the motion of the fluid element

FIGURE 1.1: Sketch of an Eulerian and a Lagrangian numerical method. Source: (Shadloo, Oger, and Le Touzé, 2016)

Mesh-free/Mesh-based Another way to classify numerical methods involves the use of grids or meshes to discretize the space. A mesh is an ordered list of node coordinates, with information about the connectivity between nodes. To make this, a list of nearest neighbors nodes of the central one is built. In case of governing equations to be solved, the mesh is constructed to fill the entire domain over which these equations have to be applied, for their spatial discretization (Mavriplis, 1996).

A subset of the mesh-free (or mesh-less) methods is composed by the particle methods, that discretize the fluid with a finite set of particles, following the changes in their state (*i.e.*, position, velocity, density, temperature) (see Fig. 1.5). Examples can be the Molecular Dynamics (MD), and the Smoothed Particle Hydrodynamics (SPH).

Eulerian and Lagrangian numerical methods can thus be mesh-based or mesh-free. For mesh-based cases, in Eulerian methods, the mesh is fixed in space (see Fig. 1.2). Grids are very usable in this case, but it is difficult to track free surfaces and other complex interfaces. Also the computational load is high, due to the need to discretize the whole domain with the grid. In Lagrangian case, the mesh is fixed to the simulated body and follows it during the simulation, re-adapting itself to the body deformations (see Fig. 1.4). This case is largely applied for boundary conditions on the surface or any fluid fronts. However, it is difficult to track very distorted geometries and large deformations, since a frequent re-meshing is necessary. Mesh-free methods, therefore, overcome these limitations. They are based on the absence of any mesh or grid, exploiting the interactions between each node of the discretized system and all its neighbors. For Eulerian case, this setting is typical of fluid mechanics. It is applied when tracking averaged properties over a control volume of the fluid is need, without a fixed grid. An example is the Eulerian SPH (Lind and Stansby, 2016; Nasar et al., 2021; Chang, 2023), in which the particles assume the role of simple fixed quadrature points (see Fig. 1.3). Thus, the method remains mesh-less (there are no strict requirements on node (particle) connectivity or ordering), and particle distributions uniform. In addition, high accuracy and computational savings are available. For Lagrangian methods, in which it is frequent to simulate large deformations or changing surfaces, mesh-free methods are based on units free to move, and the absence of a mesh allows to avoid the frequent re-meshing, reducing simulation times and approximation errors. Therefore, mesh-free Lagrangian methods are good for modeling large deformations. An examples can be the Discrete Element Method (DEM).

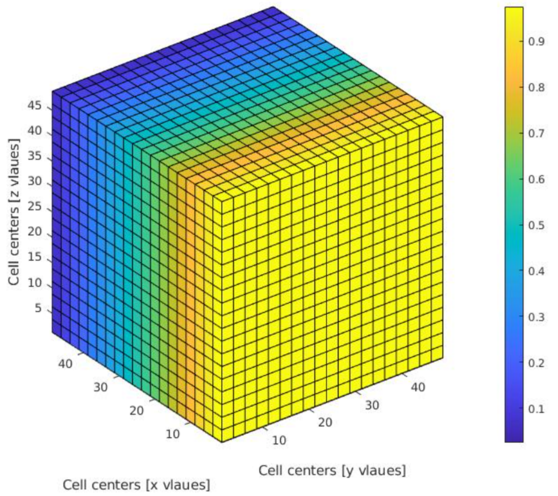


FIGURE 1.2: Example of an Eulerian mesh-based numerical method. Source: here

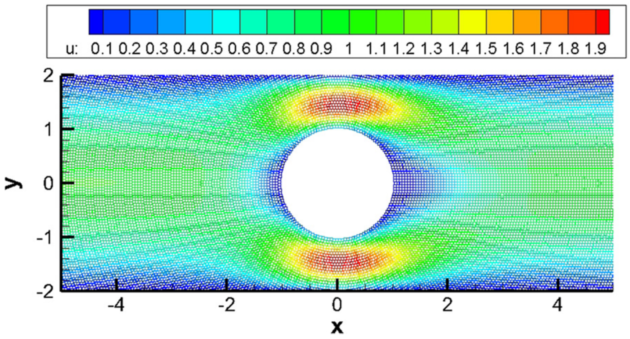


FIGURE 1.3: Example of an Eulerian mesh-free numerical method. Source: (Lind and Stansby, 2016)

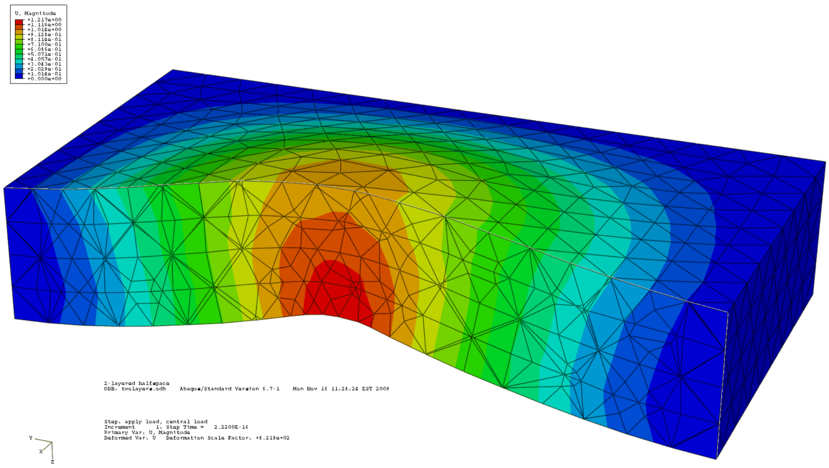

U, Magnitude
+1.217e+00
+1.116e+00
+1.014e+00
+9.128e-01
+8.114e-01
+7.100e-01
+6.085e-01
+5.071e-01
+4.057e-01
+3.043e-01
+2.029e-01
+1.014e-01
+0.000e+00
2-layered halfspace
ODB: twolayers.odb Abaqus/Standard Version 6.7-1 Mon Nov 16 11:28:24 EST 2009
Step: apply load, central load
Increment 1: Step Time = 2.2200E-16
Primary Var: U, Magnitude
Deformed Var: U Deformation Scale Factor: +8.218e+02
Y
X
Z

FIGURE 1.4: Example of a Lagrangian mesh-based numerical method. Source: here

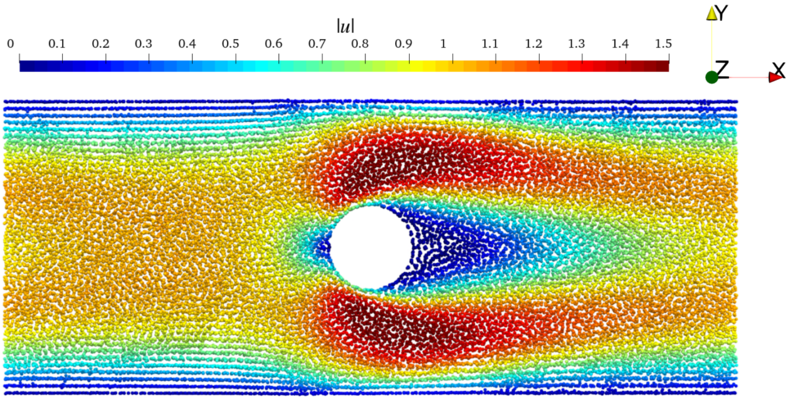


FIGURE 1.5: Example of a Lagrangian mesh-free numerical particle method. Source: here (Rakhsha, Kees, and Negrut, 2021)

1.2.4 A Lagrangian mesh-free method: Smoothed Particle Hydrodynamics

Computational Fluid Dynamics (CFD) is an approach that uses numerical techniques to solve fluid dynamics problems. Also in this case, it is possible to classify Eulerian or Lagrangian, mesh-free or mesh-based methods. Between others, the Lagrangian mesh-free methods are efficient for complex dynamics. Specifically, the Lagrangian nature allows an accurate treatment of flow interfaces, including free surfaces, and interactions with complex geometries. The absence of any grid allows an efficient management of highly dynamics flows undergoing strong mixing (Zago et al., 2018). A largely used CFD Lagrangian mesh-free method is the Smoothed Particle Hydrodynamics (Monaghan, 2005).

Smoothed Particle Hydrodynamics Between the different CFD approaches, Smoothed Particle Hydrodynamics (SPH) is a Lagrangian mesh-free computational particle model to simulate fluids, based on a discrete approximation of the Navier-Stokes equations, using smoothed kernels to weigh interactions between particles. It was introduced firstly by Gingold and Monaghan and Lucy in 1977 (Gingold and Monaghan, 1977; Lucy, 1977) in astrophysics fields and, subsequently, widely used in fluid dynamics (Monaghan, 1992; Monaghan, 2005). It is especially used for complex fluids and free-surface flows simulations (Zago et al., 2017; Zago et al., 2021), because the Lagrangian properties allow a good representation of the changing irregular shape of the flows, interfaces and free surfaces, and the absence of a mesh gives the possibility to manage large deformations while fluid flows, without frequent re-meshings and consequent extra time and approximations of the volume. Therefore, SPH produces high fidelity simulations of complex fluids, as lava flows, catching the kinematic and thermal details of the fluid, and dealing well with flow interfaces and high deformations. It is highly parallelizable, executable on GPUs (Graphics Processing Units), and it is capable to solve viscous and thermal effects (Zago et al., 2017). SPH can have different formulations (Bilotta et al., 2022b; Bilotta et al., 2022a), in different spatial dimensions and with specific physical properties. It can include numerical corrections (Vila, 1999; Zago et al., 2021; Molteni and Colagrossi, 2009; Saikali et al., 2020), chosen according to the characteristics of the simulated fluid and problem and the level of abstraction adopted with respect to the simulated physics.

Next, the SPH formulation in fluid dynamics and the integration scheme used will be presented.

SPH formulation in fluid dynamics SPH formulation is largely applied in fluid dynamics systems, to numerically simulate the fluid behaviors. It can be developed in different space dimensions, as $1D$ case (Price, 2008), $2D$ case (Bilotta et al., 2022a), and $3D$ case (Zago et al., 2017). It is important to underline that the general SPH formulation is the same, while the definition of the kernel and the definition of the mass vary as the spatial dimension varies (the kernel because it follows the normalization property in Eq. A.2, the mass because of its dependence on volume, surface, or similar, as the dimension varies).

Different variants of the SPH discretizations exist (Oger et al., 2007). Here, the following one was chosen (Zago et al., 2018), which is one of the most widespread. The equation of the mass conservation (equation of continuity) is written as

$$\frac{D\rho_i}{Dt} = \sum_j \vec{u_{ij}} \cdot \vec{x_{ij}} F_{ij} m_j + \xi h c_0 \sum_j \Psi_{ij} F_{ij} m_j, \tag{1.17}$$

with $\vec{x}_i, \vec{y}_i$ position coordinates, $\vec{u}_i$ velocity, m_i fixed mass, ρ_i density, $V_i = \frac{m_i}{\rho_i}$ volume and P_i pressure, for a particle i. Then, $\vec{x}_{ij} = \vec{x}_i - \vec{x}_j$ and $\vec{u}_{ij} = \vec{u}_i - \vec{u}_j$. Moreover, c_0 is the speed of sound, h is the smoothing length, defined using the smoothing factor as $s_f = \frac{h}{\Delta p}$, with Δp spatial resolution of the approximation (particle size) and, typically, $s_f = 1.33$ (Zago et al., 2018). $W_{ij} = W(\left|\vec{x}_i - \vec{x}_j\right|, h)$, with $\nabla_j W_{ij} = \vec{x}_{ij} F(|\vec{x}_{ij}|) = \vec{x}_{ij} F_{ij}$ (see Appendix A). The second term on the right hand side of Eq. 1.17 stands for the density diffusion correction (Molteni and Colagrossi, 2009; Antuono, Colagrossi, and Marrone, 2012), improving the pressure and stability of the numerical simulation (especially for quasi-static flows).

Here, the density diffusion formulation of (Antuono, Colagrossi, and Marrone, 2012) was chosen, with ξ density diffusion coefficient, typically $\xi = 0.1$, and with

$$\Psi_{ij} = \begin{cases} 2\left(\frac{\rho_j}{\rho_i} - 1\right) & \text{if} \quad \frac{|P_i - P_j|}{\rho_i \vec{g} |\vec{y}_i - \vec{y}_j|} > 1 \\ 0 & \text{otherwise.} \end{cases} \tag{1.18}$$

The discretization for the inviscid equation of the momentum conservation (with artificial viscosity) is written as

$$\frac{D\vec{u}_i}{Dt} = -\sum_j \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2} + \Pi_{ij} \right) \vec{x_{ij}} F_{ij} m_j + \vec{g}, \tag{1.19}$$

where $\vec{g} = (0, -9.81)^T$ is the gravity vector and Π_{ij} is the artificial viscosity term (Monaghan, 2005), which helps reducing numerical high frequency components of the velocity field (helping in numerical stability), defined as

$$\Pi_{ij} = \begin{cases} -\frac{\alpha\, h\, c_0}{\rho_j} \left(\frac{\vec{u}_{ij} \cdot \vec{x}_{ij}}{|\vec{x}_{ij}|^2 + \epsilon h^2} \right) & \text{if} \qquad (\vec{u}_{ij} \cdot \vec{x}_{ij}) > 1 \\ 0 & \text{otherwise,} \end{cases} \tag{1.20}$$

with $\epsilon = 0.01$, added to avoid a singularity when $|\vec{x}_{ij}|^2 \to 0$ (Monaghan, 2005).

An equivalent kinematic viscosity ν is obtained defining the artificial viscosity coefficient α (Monaghan, 2005) in the following way

$$\alpha = \frac{c_\alpha\, \nu}{h\, c_0}, \tag{1.21}$$

with $c_\alpha = 8$ in $2D$, and $c_\alpha = 10$ in $3D$ (Monaghan, 2005).

The discretization for the viscous equation of the momentum conservation (with physical viscosity) is written as

$$\frac{D\vec{u}_i}{Dt} = -\sum_j \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2} \right) \vec{x_{ij}} F_{ij} m_i + \sum_j \frac{2\mu_{ij}}{\rho_i \rho_j} \vec{u_{ij}} F_{ij} m_i + \vec{g}, \tag{1.22}$$

where μ_{ij} is a parameters for the viscosity (that can be the viscosity of a single particle, or the arithmetic or harmonic mean of two nearby particles).

The pressure P is obtained from the density, using the Cole's equation of state (Eq. 1.11); if c_0 is at least an order of magnitude higher than the maximum velocity of the flow (hydrostatic velocity, that is $u_h = \sqrt{2\|\vec{g}\|h}$, with h the maximum height of the fluid (Torricelli, 1644)), a weakly-compressible fluid is obtained.

Integration scheme Once the spatial derivatives have been evaluated according to Eq. A.8, the Partial Differential Equations (PDEs) are reduced to Ordinary Differential Equations (ODEs) with derivatives over time. These are then integrated adopting a second order predictor-corrector integration scheme (Zago et al., 2021), described by the following steps.

1. Compute accelerations and density derivatives at instant n:
   (a) $\vec{a}^{(n)} = \vec{a}(\vec{x}^{(n)}, \vec{u}^{(n)}, \rho^{(n)})$,
   (b) $\dot{\rho}^{(n)} = \dot{\rho}(\vec{x}^{(n)}, \vec{u}^{(n)}, \rho^{(n)})$.
2. Compute half-step intermediate positions, velocities and densities (predictor):
   (a) $\vec{x}^{(n*)} = \vec{x}^{(n)} + \vec{u}^{(n)} \frac{\Delta t}{2}$,
   (b) $\vec{u}^{(n*)} = \vec{u}^{(n)} + \vec{a}^{(n)} \frac{\Delta t}{2}$,
   (c) $\rho^{(n*)} = \rho^{(n)} + \dot{\rho}^{(n)} \frac{\Delta t}{2}$.
3. Compute corrected accelerations and density derivatives:
   (a) $\vec{a}^{(n*)} = \vec{a}(\vec{x}^{(n*)}, \vec{u}^{(n*)}, \rho^{(n*)})$,

(b) $\dot{\rho}^{(n*)} = \dot{\rho}(\vec{x}^{(n*)}, \vec{u}^{(n*)}, \rho^{(n*)})$.

4. Compute new positions, velocities and densities (corrector):

   (a) $\vec{x}^{(n+1)} = \vec{x}^{(n)} + (\vec{u}^{(n)} + \vec{a}^{(n*)}\frac{\Delta t}{2})\Delta t$,

   (b) $\vec{u}^{(n+1)} = \vec{u}^{(n)} + \vec{a}^{(n*)}\Delta t$,

   (c) $\rho^{(n+1)} = \rho^{(n)} + \dot{\rho}^{(n*)}\Delta t$.

The time step Δt is computed starting from the one for each particle i, Δt_i, requiring to fulfill the CFL-like (Courant-Friedrichs-Lewy) stability conditions (Monaghan, 1992), determined by the acceleration magnitude, the speed of sound and the viscosity term (Eq. 1.23),

$$\Delta t_i \leq \min \left\{ 0.3\sqrt{\frac{h}{||\vec{a}_i||}}, 0.3\frac{h}{c_0}, 0.125\frac{\rho_0 h^2}{\mu_i} \right\}, \tag{1.23}$$

with ρ_0 the initial material density [kg/m^3], so that $\Delta t = \min\, \Delta t_i$ (Zago et al., 2018).

The theoretical basis of the SPH model, in particular the mathematical derivation of the SPH approximation and of the spatial gradients discretization, and the kernel choice and SPH initial and boundary conditions setting are shown in Appendix A.

1.2.5 Numerical applications to lava flows and limits

Mathematical modeling can help in improving lava flow behaviors knowledge, delving into aspects related to spatio-temporal thermal and viscous evolutions. The complex nature of the fluid, make simulation of lava flows an extremely challenging task for CFD. Various computer codes have been developed to reproduce and predict lava flow dynamics, differing in physical implementations, numerical accuracy, and computational costs.

As already seen, lava flow mathematical models and numerical simulations must have a lot of input parameters, that are topography or slope, eruptive input conditions (volume effusion rate, vent geometry and effusion temperature), thermal boundary condition at the top and the bottom of the flow, and physical properties of the lava: density, thermal conductivity, rheology (Cordonnier, Lev, and Garel, 2016).

In the modeling of lava flow hazards (Cappello, Vicari, and Del Negro, 2011; Del Negro, Cappello, and Ganci, 2016), different approaches aim to simplify the modeling complexity, in terms of implementations and higher performances, with reduced dimensionality approaches (Fernández-Nieto et al., 2016), simplified thermal or dynamic models, or using stochastic approaches with little or no physical modeling (Costa and Macedonio, 2005), making possible real-time forecasting, risk mitigation, the production of long-term scenarios (Scifoni et al., 2010).

Mesh-based methods

In the following subsections, a description of the main mesh-based lava flows simulation methods are shown (Cordonnier, Lev, and Garel, 2016).

Channelled models Channelled models implement 1D problems, in which the fluid is confined and the flow advances downwards, only in one direction. They manage a quite complex thermal model, crystallization rate, and rheology models, and the velocity of the flow depends on the channel dimensions and on the rheology of the lava. The main example of 1D channelled lava implementation is FLOWGO (Harris and Rowland, 2001), that models finite amounts of moving lava between stagnant levees. It has no mechanically continuous roof, the moving lava top represents a free surface open to the atmosphere, and its sides and bottom are in contact with levee walls and the emplaced flow base. Thermal aspects, the heat lost, is managed by radiation and convection phenomena at the surface and conduction at the base, and the rheology is computed from its crystal- and temperature-dependence. Therefore, the flow along the channel is stopped when the lava cools down and its rheological properties equal that of a solid.

Models like this are fast and easy to implement, but some limitations arise from the 1D assumption, forcing the local channel width to directly correlate with ground slope.

Cellular Automata The cellular Automata (CA) is a discrete computational model, composed by a regular grid of cells with specific cells state. Selected a central cell, a set of neighborhood is defined, that influence the state of the central one and the system advances according to fixed updating rules (generally, a mathematical function) that determines the new state of each cell in terms of the current state and the states of the neighborhood cells (Schiff, 2011).

Examples of CA implementations for model lava flow emplacement are MAGFLOW (Cappello et al., 2016), SCIARA (Rongo et al., 2016) and FLOW-FRONT (Young and Wadge, 1990).

Using Cellular Automata, the computational domain is discretized with a 2D grid of cells, in which cells are characterized by specific volcanic properties as altitude, lava height and temperature, and lava flow advance. The cooling process is implemented with the evolution of cell properties. The flow of lava from one cell to another depends on the low density and on the slope (difference between two adjacent cells altitudes) and it is stopped with plastic rheology model or following the solidification process.

Models like this are fast and easy to implement. Limitations are encountered in their absence of a detailed vertical description of the lava flow, important in coupling surface and basal heat losses to the bulk rheology evolution, or to model the formation of phenomena like lava tubes.

Depth-averaged models Depth-averaged methods are simplified models that act on spatial dimensionality reduction, using the shallow-water equation (SWE) (Sadourny, 1975), assuming that the horizontal dimension is greater than the vertical one, giving homogeneous properties throughout the fluid section. In volcanic context, from Newtonian to Bingham rheology models can be implemented, with constant viscosity value along the vertical profile. This is a limitation in the rheology aspects into flow dynamics. Depth-averaged codes for volcanological applications include (Macedonio, Costa, and Longo, 2005), VOLCFLOW (Kelfoun and Vargas, 2016) and RHEOLEF (Saramito, 2022). Models like this have an easy implementation, reducing simulation times (Fernández-Nieto et al., 2016).

Nuclear based models Nuclear based models owe their name to his first application in the field of nuclear plants, simulating and optimizing the spreading phase of melted substance in case of nuclear reactor meltdown (Greiner, Maruhn, et al., 1996). They are an evolution of the Cellular Automata, adding the third dimension, that is a discretization of the height of each cell. In volcanic context, LavaSIM (Proietti et al., 2009) is an example for lava simulations. Free surfaces has not been modeled yet, making this a limitation for lava simulations.

CFD OpenFOAM Open Field Operation And Manipulation (OpenFOAM) (Jasak, Jemcov, Tukovic, et al., 2007) is an open-source software package based on finite Volumes, produced by OpenCFD Ltd, and it is applied to solve problem of continuum-mechanics problems, including CFD problems. It is applicable to different fields, as complex fluids, chemical reactions, turbulence, and heat transfer. It is easy to implement and fully parallelized. In particular in volcanic field, specific feature of lava are easy to be implemented, adding equations, solvers and specific applications.

CFD Flow3D Flow3D (Duguay, Lacey, and Gaucher, 2017) is a commercial software, with engineering and Computational Fluid Dynamics applications, distributed by Flow Science Inc. It is based on Volume-of-Fluid algorithms, combined with Level Sets, to model interface tracking feature, making suitable simulation of free surface flows, as lava flows. Limits are that it is slow and it has a high price.

Mesh-free methods

Smoothed Particle Hydrodynamics (SPH) method is largely used to simulate complex fluids and free-surface flows simulations, as lava flows (Zago et al., 2017; Zago et al., 2021), for its Lagrangian mesh-free nature. Catching kinematic and thermal details of the lava, with viscous and thermal effects, and dealing well with flow interfaces and high deformations, it is largely used in volcanic context. It is also highly parallelizable (Zago et al., 2017). An example is GPUSPH (Bilotta et al., 2016), with applications on the evolution of surface temperature and solidification, managing to get velocity and temperature profiles (Hérault et al., 2011). Limits to this approach are the heavy computational costs, in terms of time and computing resources.

Chapter 2

Artificial Intelligence

In the era of constant production of information and signals, new researches are increasingly turning towards a massive use of the Artificial Intelligence (AI) approaches to analyze this great quantity of data.

AI is the branch of computer science technology that simulates the human intelligence processes using machines, especially computer systems, reproducing common human tasks and going beyond. In this way, intelligent systems, capable of reasoning and solving problems, are produced. As for humans, that learn from experience or examples and through mistakes, these systems learn from data and through experience and not through models or corrections superimposed (Goodfellow, Bengio, and Courville, 2016).

Artificial Intelligence is mainly divided into Machine Learning (ML) (Bonaccorso, 2017) and Deep Learning (DL) (Goodfellow, Bengio, and Courville, 2016). In the first one, the model learns from structured data, composed by vectors of features representative of the input state, for example using a list of pixels of an image, exploiting the spectral characteristics of the object captured there (Amato, 2022). In the second one, the model learns from unstructured data, as raw images, automatically extracting the features useful for the relative purpose (ground-truth), exploiting spectral, spatial and/or temporal characteristics of the input data (Amato et al., 2023a). Briefly, in ML, human experts determine the hierarchy of features to understand the differences between data inputs, usually requiring more structured data to learn ("feature engineering" process), in DL the model can ingest unstructured data in its raw form (*e.g.*, text or images), and it automatically determines the hierarchy of features which distinguish different categories of data from one another. Therefore, DL is a subset of ML.

In both cases, the input data are analyzed, extracting some basic characteristics, which are then passed on to the next training step, that elaborates them in a deeper way, extracting ever more general information about the data. This process is in perfect analogy with the human elaboration of concepts through the transition from particular and easy to understand cases to more general and abstract one.

The learning process in ML and DL is the phase in which the model uses techniques to learn an analysis scheme for the system under examination, automatically extrapolating from data a model to describe it. It is divided into three cases, reinforcement, supervised, and unsupervised learning. The first one is based on the use of an agent, that learns by trial and error using feedback and experience in an interactive environment. The external environment provides the initial sets of data and evaluates the final result, with a reward if that result is correct and a penalty if it is not. The learning then proceeds with successive tests and it is based on subsequent corrections of the work, through reinforcements of the more efficient process. The main aim of reinforcement learning is thus to find a suitable action model, to maximize the total cumulative reward of the agent. It is mainly diffuses in gaming, robotics, and similar fields (Wiering and Van Otterlo, 2012). In the supervised learning, the model is trained using a set of data, the training set, labeled with the relative results, typically called the ground-truth, the targets linked to the real meaning of each data. In this case, the model adjusts their parameters exploiting a series of typical characteristics automatically extrapolated from the couple data-label, minimizing the difference between predicted values and real ones, *i.e.*, the loss function. Thus, the learning process in based on the correlation between data and ground-truth. In the unsupervised learning, in contrast, the training data are not labeled and the model searches patterns and schemes directly in the data, without having any ground-truth data. This is very useful for dataset that can not be characterized a priori. In data analysis, the main approaches are the supervised and unsupervised learning (Amato et al., 2021a).

Both ML and DL models can have two main tasks (Bonaccorso, 2017), *i.e.*, classification and regression one. In the first case, the aim is to divide the data into classes, or clusters, determined by intra-classes similarities, that measure the homogeneity of a cluster, and inter-classes dissimilarities, that measure the distance between two clusters (Corradino et al., 2021b; Corradino et al., 2022; Torrisi et al., 2022a; Cariello et al., 2023). Therefore, the output of the model is a discrete number, linked to the class membership. In the second case, the purpose is to find a model that in a better way fits the input data, interpolating them, and that can predict future trends, for data never seen in the training set. Thus, the output is a continuous numeric data (Malik et al., 2021).

In order to conduct a balanced and not polarized analysis, the input dataset is divided into three independent subsets, *i.e.*, a train set, a validation set and a test set, randomly distributing the data in each set. The train set is used to train the model, the second one to validate the results in the train phase and to fit the hyperparameters (parameters that directly control the structure, function, and performance of the model), and the last set is used to test the model with data never seen during in the training phase, calculating the predictive performances of the model. The learning process is composed of different data processing cycles, called epochs, for each of which the model analyzes the data and extrapolate a function that can fit them in an increasingly precise way, calculating an error between the prediction and the target data. Therefore, during the training phase, the model feeds training and validation data and, for each epoch (iteration), processes the data. The learning process is stopped according to different stopping criteria, such as after a predefined number of epochs or if the error is less than a predefined tolerance (here, in the following applications, for the first reason, *i.e.*, after a predefined number of epochs), and the final error is used as a reference for the goodness of the training phase. One of the learning hyperparameter is the learning rate, a tuning parameter that determines the step size at each iteration while moving toward a minimum of a certain loss function, an objective function to learn the analysis scheme. The error is calculated also for the validation set, to verify the capability of the model to learn for data not used during the training phase. Finally, the test set is used to calculate the accuracy of the model, through the test error. If the model has poor performance on training data, *i.e.*, it is not capable to capture the relationship between the data themselves or the data and the targets, the *underfitting* phenomenon will be showing up. If the model performs well on training data, and not on testing data, *i.e.*, the model stores data it has analyzed during the training phase and it is not capable to generalize for data never seen before, the *overfitting* phenomenon will be showing up (Bonaccorso, 2017).

Deep learning needs more data to build a robust model than machine learning.

The goodness of an AI model is calculated over the test set with several performance indices, that compare the model prediction with the real values of the data labeled to evaluate the model. For the classification task, examples of performance indices can be the accuracy index (Eq. 2.1), the precision index (Eq. 2.2), the recall index (Eq. 2.3) and the F1 score (Eq. 2.4), then calculated as percentages (James et al., 2013).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2.2}$$

$$Recall = \frac{TP}{TP + FN} \tag{2.3}$$

$$F1score = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{2.4}$$

where TP (True Positives) are the data correctly predicted to belong to class 1 that actually belong to class 1, TN (True Negatives) are the data correctly predicted to belong to class 0 that actually belong to class 0, FP (False Positives) are the data incorrectly predicted to belong to class 1 that actually belong to class 0, and FN (False Negatives) are the data incorrectly predicted to belong to class 0 that actually belong to class 1. The total number of samples in the test set is given by the sum of TP, TN, FP and FN (James et al., 2013). For the regression task, other performance indices exist, such as the Mean Squared Error (MSE) (Bonaccorso, 2017), defined as follows.

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{2.5}$$

with N number of samples, y_i real value, and $\hat{y}_i$ predicted value.

Another important aspect in AI is the *model explainability*, which is the property of an AI model that can be explained. The branch of AI that deals with this is called Explainable AI (XAI) (Xu et al., 2019), and the idea is to make possible for the human operator to be able to grasp the reasons behind the automatic decisions or predictions of the model. In many real-world applications affected by AI decisions (medical applications, business decision, process optimization, for example) AI models explainability and transparency is essential to make models reliable and truly usable.

On a practical level, the AI techniques used can be the most varied, clustering techniques, *i.e.*, mathematical models that group data with similar characteristics (Xu and Wunsch, 2005), decision trees, that are graphs in which, starting from an input, the relative output can be predicted using if-else conditions over the data (Shaik and Srinivasan, 2019), random forests, that are groups of decision trees, and probabilistic models, which are based on the probability that, given an input, a specific output will emerge (Ghahramani, 2015), all of these in a context of machine or deep learning.

In the following section, an overview of the principal state of the art AI models is presented.

2.1 AI models

An AI model is a mathematical representation of a specific problem, obtained automatically relating input to output data, finding patterns and making predictions. It can be used for different problems, as data classification or value regression and it can be built using different approaches, as ML or DL. Each model can have a different architecture (structure, functional form) and, using a specific algorithm (procedure), an entire automatic framework can be created, that feeds the input data, recognizes patterns or schemes, and extracts information to make decisions.

More rigorously, let $\vec{X} \in \mathbb{R}^p$ be the input space, with $p = 1, 2, ...$ the number of independent input variables, and $\vec{Y} \in \mathbb{R}^d$ the output space, with $d = 1, 2, ...$ the number of dependent output variables. Denoting $\vec{x}$ the vector of components $(x_i)_{i \in \mathbb{N}}$ in the input space, and $\vec{y}$ the vector of components $(y_i)_{i \in \mathbb{N}}$ in the output space, the AI model is a function f (Eq. 2.6) that links the independent variable or input $\vec{x} \in \vec{X}$ with the dependent variable or output $\vec{y} \in \vec{Y}$, with an error component $\vec{\varepsilon} = (\varepsilon_i)_{i \in \mathbb{N}} \in \mathbb{R}$, such that

$$\begin{aligned} f :& \vec{X} \rightarrow \vec{Y} \\ & \vec{x} \rightarrow \vec{y} = f(\vec{x}) + \vec{\varepsilon} \end{aligned} \tag{2.6}$$

with $f(\vec{x}) = \hat{\vec{y}}$ the predicted value, $\vec{y}$ the real value, and $\vec{\varepsilon} = (\varepsilon_i)_{i \in \mathbb{N}}$ the error, such that its components are supposed to be independent and identically distributed with zero expected value, constant variance and with covariance between two of the components equal to zero, regardless of the chosen variables. This describes the error made in predicting the values of $\vec{x}$ with $\vec{y}$ via the function f, *i.e.*, the difference between the predicted and the real value of the output data ($\vec{\varepsilon} = \vec{y} - \hat{\vec{y}}$) (Bonaccorso, 2017).

The Eq. 2.6 is a simplified formulation of the AI models, by way of example. A list of common AI models is shown in the following subsections, with different architectures, applicable to various contexts.

2.1.1 Unsupervised Machine Learning

k-Means This is an unsupervised ML approach. The k-Means (kM) algorithm is a clustering method (Ahmed, Seraj, and Islam, 2020) in which the goal is to divide the elements into k homogeneous groups or classes, identified as clusters, *i.e.*, a collection of data points aggregated together because of certain similarities. The number k of classes must be set a priori. There are two approaches to cluster the data, the partitional method, *i.e.*, a first random division of the elements is carried out, and then the elements are slowly moved within the clusters, obtaining the final class, and the agglomerative hierarchical method, *i.e.*, all the elements are considered as a single cluster and gradually merge or divide them into the various clusters. For each class, it is possible to calculate the centroid ("means" because the aim is to average the data, that is, finding the centroid), *i.e.*, the class center, as the mean value calculation on datasets with known classes. Starting with a first group of randomly selected centroids, which are used as the beginning points for every cluster, the model performs iterative calculations to optimize the positions of the centroids, then allocating every data point to the nearest cluster (in the partitional method, for example). The algorithm is stopped when there is no changes in the centroid positions, or the defined number of iterations has been achieved. Clustering technique is an unsupervised approach, that does not consider the reference class or label. This is due to an a priori lack of the reference targets for the data or because the aim is to obtain that ones for a subsequent supervised task.

x-Means This is a ML unsupervised model, that is an efficient version of the k-Means algorithm. The x-Means (xM) efficiently searches the space of cluster locations and number of clusters to optimize the Bayesian Information Criterion (BIC) or the Akaike Information Criterion (AIC) measure. A threshold is then automatically set based on a specific pixel feature similarity, in terms of their Euclidean distance from each computed centroid (Pelleg, Moore, et al., 2000).

2.1.2 Supervised Machine Learning

Linear regression Considering the model in Eq. 2.6, if $p = 1$ and $d = 1$, then the model is the linear regression (LR). This is one of the simplest supervised ML models. For each sample, that is a couple $(x_i, y_i)_{i \in \mathbb{N}}$, the model is the following one:

$$y_i = \mu(x_i) + \varepsilon_i, \tag{2.7}$$

where μ represents an unknown function to be estimated, and the variables $\varepsilon_i \in \mathbb{R}$ the error component.

Support Vector Machine This is a supervised ML approach. The aim of the Support Vector Machine (SVM) is to find an hyperplane in an N-dimensional space ($N \in \mathbb{N}$ is the number of features), that classifies the data points. The hyperplane chosen must have the maximum margins, *i.e.*, the maximum distance between data points of both classes, to efficiently classify the data. Data points falling on either side of the hyperplane can be attributed to different classes. SVM algorithm have three parameters. The first one is the kernel type, that is a transformation to bring the data in the required form, the second parameter is γ, that is the spread of the kernel (*i.e.*, the decision region, how far the influence of a single training example reaches), and the third is the cost C, that is the inverse of the maximum margin (*i.e.*, the trade-off between smooth decision boundary, namely the hyperplane, and classifying training points correctly) (Noble, 2006).

Random Forest This is a supervised ML approach. It is based on the decision tree algorithm, a model based on simple decision rules inferred from input features during the training phase, to classify the data or make predictions. Random Forest (RF) is made by a number of decision trees, that must be set a priori. Each decision tree is trained independently using samples of the original training dataset and their results are combined to get the final RF outcome, based on the majority voting (Belgiu and Drăguţ, 2016) or averaging them, if the task is classification or regression. During the training phase, RF learns from data the best decision rules to minimize the error between target and model output.

Minimum Distance This is a supervised ML approach. The Minimum Distance (MD) classifier is a method that classifies the data using the shortest distance between the input sample and all the centroids of the classes, assigning the data to the class with the shortest distance. It is mostly used in classification problems. The distance is defined as an index of similarity so that the minimum distance is identical to the maximum similarity. The distances applied can be different, such as the

Euclidean, Normalized Euclidean, Manhattan, or Mahalanobis distance (Zhao et al., 2012).

Classification and Regression Trees This is a supervised ML approach. The Classification and Regression Trees (CART) classifier is based on a decision tree algorithm that can be used for classification or regression predictive problems. This is composed by a binary tree, with input nodes for the input variables, and the if-else rules to analyze the data, assuming that the variables are numeric. The final nodes of the tree contain an output variable, to make a prediction. As for decision trees, to fit a CART model the input data are fed and split following the rules to minimize a cost function. Tree construction ends using a predefined stopping criterion, such as a minimum number of splits or a minimum numbers of nodes (Breiman, 2017).

Artificial Neural Network One of the most used ML models is the Artificial Neural Network (ANN) (Wang and Wang, 2003; Bonaccorso, 2017), that mimics the human brain structure and the way to extract information analyzing input data. ANN is composed by a number of nodes called *artificial neurons*, to simulate the natural neurons of the brain, and a number of weighted connections between nodes, the *edges*. The NN is the only example of architecture that can have supervised or unsupervised learning, analyzing data and labels in the first case and automatically clustering similar data in the second one.

For NN with unsupervised tasks (Becker, 1991), different learning process are applicable. One can be the competitive learning. In this approach, the nodes are put in competition to find the one for which the weight vector is closest to the input vector. Calculating all the distances between the data, the one with minimum distance is chosen. The output neuron with the minimal distance is called the *winner-takes-all*, and, during the learning process, only its weights are updated. A similar formulation, based on spatial proximity between nodes, is the Kohonen one (Dekker, 1994), in which the maps are created, taking into account the location of the nodes. Therefore, considering any node, the nearby ones are their neighborhood, and they are arranged on a square map by associating a function which indicates how much distance that node from the others. Thus, nodes close to each other will specialize in recognizing similar inputs. This leads to the concept of bubbles, *i.e.*, *n* subset of nodes representing the clusters. This allows to better manage the variability between elements, and, in addition, respect to use the other techniques, in this case the number of clusters can be decided afterwards. These networks are based on the behavior of topological maps in the cerebral cortex of the brain.

In cases of NN for supervised tasks (Hush and Horne, 1993), it can be composed by a single neuron, a *perceptron*, with a number of nodes, and a single or binary output. The learning process is an iterative process. For each epoch (iteration, *i.e.*, update cycle in which the training set is presented to the network and the error between the prediction and the target data is calculated), the nodes process the information, the edges connect the nodes, summing-up and passing the information, as synapses do, and storing the knowledge in the weights during the learning process. Comparing the prediction with the real value or the similarity between them, the weights are updated. At the end of the process, the information passes through an activation function, which activates or not the neuron (Goodfellow, Bengio, and Courville, 2016). It is possible to (fully) connect a number of perceptrons (neurons), obtaining a *multilayer perceptron* (MLP), used for supervised aims and commonly named *neural network*. It was demonstrated that the standard multilayer feed-forward networks, with at least a sufficiently wide hidden layer of neurons, can approximate any kind of functions with any accuracy. Therefore, the neural networks can be considered as *universal approximators* (Hornik, Stinchcombe, and White, 1989) of each kind of functions, in the regression task. The multilayer perceptron is organized in consecutive layers with related activation functions. The first layer is named input layer, and it has the same dimensions of the input data. The last layer is the output one, with the dimensions of the output (not necessarily binary). Between input and output layer, each layer of nodes is connected with the previous and next one, composing the hidden layers. The dimensions of each hidden layer varies depending on the NN architecture, with a larger NN (higher number of nodes in the hidden layers) to extract features from more complex data. The number of hidden layers (network depth) again depends on the complexity of the data and of the task. From the layers point of view, thus, the learning process is based on a process of sending information between layers. Each layer extracts simpler features from the data, moved then to the next layers to be processed and used in an higher level. The training follows the backpropagation process (Hecht-Nielsen, 1992), in which the information is processed in the forward direction, from the input to the output. Subsequently, an error between the prediction and the reality is calculated and that error rate is propagated backwards through the neural network layers to update and fine-tune the weights. Each node computes its output from the input by using the *activation function* (Goodfellow, Bengio, and Courville, 2016).

Recurrent Neural Network Recurrent Neural Networks (RNN) are NN in which the neurons are not only connected from the input to the output, but they also follow a cycle, (closed loops, "feedback connections"), in which the output depends on the prior elements and on the future information of the sequence (Medsker and Jain, 2001). They are characterized by the capacity to take information from prior inputs to influence the current input and output, so for their "memory" over the data, and, for this capacity, they are largely used for sequential data, for example for language translation and speech recognition tasks.

One of the main problem of RNNs is the vanishing gradient problem (Basodi et al., 2020). A neural network updates its weights at each iteration, proportionally to the partial derivative of the error function to be minimized. This problem occurs when the gradient becomes small, vanishing to zero. In cases like this, the weights can not be updated, sometimes stopping also the training process.

2.1.3 Deep Learning

Convolutional Neural Network This is a DL approach. The architecture of a Convolutional Neural Network (CNN) (Goodfellow, Bengio, and Courville, 2016) is similar to the one of the multilayer perceptron, with some developments. The idea is to mimic the connectivity pattern of neurons in the human brain and the DL approaches were inspired by the organization of the Visual Cortex. Individual human neurons respond to stimuli only in a restricted region of the visual field known as the *Receptive Field*. A collection of such fields overlap to cover the entire visual area. An artificial CNN takes as input a data, often an image, assigns importance (learnable weights and biases) to various features, capturing spatial and temporal dependencies through the application of relevant filters, sweeping and covering the entire data. Compared to an ANN, a CNN has an higher complexity in terms of neural architecture and, thus of data that can be processed. Belonging to the deep learning class of algorithms, the CNN takes directly the unstructured data as input and autonomously calculates the features within the data and segment them. Two new types of layers are present, the *convolutional* and the *pooling* ones. The first one generates a feature map by sliding a convolutional kernel (*i.e.*, a filter) over the input data (*e.g.*, an image) and recognizing patterns in them; the second one downsamples the feature map, calculating some statistics on neighboring samples (*e.g.*, the maximum or the average value of the region), from the input overlapped by the kernel. This is done to reduce overfitting processes, introducing translation invariances of the samples. The number of layers and the kinds of filters must be set a priori (Li et al., 2021).

Generative Adversarial Network A Generative Adversarial Network (GAN) is a model in the context of Generative AI, that is the branch of AI capable of generating data (Goodfellow et al., 2014). GANs are composed by two NN, *i.e.*, a Generator (G) and a Discriminator (D). Given a train set, G has the task of generating data with the same probability distribution as those of the train set, maximizing the probability of misleading D. On the other hand, considered a sample, D has the task of discriminating whether it comes from the real train set or whether it was instead generated by G, maximizing the probability of correctly classifying real data and data generated by G.

Since neural networks are universal approximators (Hornik, Stinchcombe, and White, 1989), GANs are asymptotically consistent, that is, if an infinite amount of data is considered, the estimator will give the correct result for the parameter being estimated (in this case, the sample to be discriminated) (Goodfellow, 2016).

Long short-term memory network Long Short-Term Memory (LSTM) networks (Hochreiter and Schmidhuber, 1997) are RNN, which do not suffer of the vanishing gradient problem. They are NN with a new block, capable to learn when to remember and when to forget information. Due to this block, these networks are capable to process information with a long-term time dependence. They are largely used to process time series, for example, for Natural Language Processing (Chowdhary and Chowdhary, 2020).

LSTM are composed by a cell and three gates, *i.e.*, an input gate, an output gate, and a forget gate. The cell stores information over a determined interval of time, remembering the values also for several iterations. The input gate defines which new information must be fed to the network to be processed, the output gate evaluate which processed information must be selected for the network output, and, finally, the forget gate decides which information to discharge. All the three gates manage these processes giving a 1 value to the positive assignments (*e.g.*, add new input or output information, discharge values) and a 0 value otherwise. LSTM units partially solve the vanishing gradient problem, because they allow gradients to flow also unchanged. They, however, do not solve the exploding gradient problem, that is the opposite problem of the vanishing one, *i.e.*, when the gradient tends to infinity (it "explodes") (Calin, 2020).

Transformer A transformer is a very recent DL architecture (Lin et al., 2022), largely used in sequential processing (*e.g.*, words or audio sequences). Input sequences are fed to the model, that splits them into adjacent blocks encoded as *tokens*, and converted in vectors. Each token contextualized in the sequence via a parallel multi-head attention mechanism (Vaswani et al., 2017), with the others (unmasked) tokens. This allows to amplify the signal for key tokens and reduce the one of the others.

All this process makes possible to process very long sequences, maintaining the general context and meaning of the sequence.

2.2 AI for extrapolating mathematical models input parameters: initial and boundary conditions

In modeling, one of the most delicate aspects is the search of parameters to input to the model. Therefore, also for fluid dynamics models, the initial and boundary conditions are needed to make the mathematical problem uniquely solvable (Velten, 2009). Moving to CFD, *i.e.*, dicretizing the system, under the hypotheses of existence and uniqueness of the solution, the numerical solution can be determined. In this context, however, every discrete scheme must respect the conditions of Consistency, Convergence and Stability (Anderson and Wendt, 1995; Chung, 2002), to make sense. Also in this case, initial and boundary conditions must be set.

In the volcanic field, monitoring is referred to both following the manifestations of the eruption once it has started, as well as forecasting the areas potentially threatened by lava in an effusive scenario. Different aspects can be monitored, *e.g.*, the current state of the activity, the probable evolution of the lava flow field, and the potential impact of lava flows for surrounding areas. Different CFD models can be applied to monitor lava flows, reproducing the flow behavior and making predictions and possible scenario. These models need initial and boundary conditions to be implemented, that can be obtained with on field observations and remote sensing techniques, as satellite data. This great quantity of data must be processed to extract information linked to a volcanic phenomena, for example vent position, effusion rate of the lava eruption, or extension of the lava field, *i.e.*, initial and boundary conditions of the phenomenon. To process data more easily, automatic artificial intelligence techniques can be implemented. These techniques reduce data processing times, obtaining results in near real-time, they are also well usable with images (for example, satellite data), also taking advantage of cloud computing.

To monitor a physical phenomena, as the volcanic activity, the techniques are different, from more physics-based to totally data-driven, and each technique requires different data inputs. The physics-based techniques include model-based monitoring. These models can be analytical or numerical, they are totally based on the physical laws that describe a system and they require as input the physical parameters to build the model. The data-driven ones include all the models based on the data, and they are built fitting the data given in input, that represent the relative phenomenon, extracting information from them to describe it. The data fed to the model can be obtained with laboratory experiments, with measurements on field or using the remote sensing techniques, that include the use of aircraft, drones, and, mainly, satellites images (Harris, 2013). The first subdivision of the volcanic applications is, therefore, between Near Real-Time (NRT) models, for a time sensitive analysis (*e.g.*, for active lava flow, ongoing), and Not Time-Critical (NTC) ones, for analysis that not need to be fast (*e.g.*, for cooled or in cooling lava flow).

The increasing availability of open-source satellite data and current developments in cloud computing and data-driven approaches have made the monitoring of volcanic hazards from Space more feasible for volcano observatories. Various forms of geospatial datasets are constantly accumulated and captured by the different forms of sensors and devices and managing such an enormous dataset is becoming a challenge. Automated techniques have been designed to process geospatial datasets with minimal human interference, exploiting the potential of the AI techniques. In the specific context of AI applied to satellite images, ML model takes as input the spectral intensities of the pixels, exploiting the spectral characteristics of the monitored area under investigation, in the different spectral bands (*e.g.*, three-bands RGB images or multispectral ones). In contrast, DL models take as input the entire image, exploiting not only the spectral response of the pixels, but also spatial and/or temporal relationships between data, to automatically extract significant features.

Satellite data inputs for ML approaches can be used to classify or to segment the objects. In the first case, for each given input is assigned a label (*e.g.*, 1 or 0) indicating the class it belongs to; in the second case, the object is divided with respect to the background, identifying its contours and assigning a semantic meaning, also linked to spatial features. In cases of satellite data, it is also possible to combine the two approaches. In the first step, to classify the pixels of a satellite image with machine learning, a *pixel-based approach* is applied. The pixel-based approach (Zerrouki and Bouchaffra, 2014) is founded on the classification of each pixel of the image under analysis one after the other, deciding which class each pixel belongs to. In cases like this, the model uses as input the specific features of the pixels, *i.e.*, the spectral response of the portion of the object acquired in that pixel, contained in the bands of the pixels. In the pixel-based algorithms, a scene-by-scene classification approach (Corradino et al., 2021a) is often applied, based on the use of a single image to apply the model, dividing it for the training and test phase. A part of the entire image (a set of pixels) is used to train the model, and the model becomes capable to classify all the rest of the pixels of the image. In this way, finally, the objects of the scene are also segmented, because the satellite image pixels are georeferenced, thus the classified pixels are repositionable in their original place of the scene, obtaining an image where each pixel is classified and the objects are all divided respect to the background, following the spectral characteristics and the spatial position, and with defined contours. Therefore, they are segmented. As opposed to the scene-by-scene approach, it is also possible to feed to the model the entire image, classifying the whole scene as a single input. This is mostly used in deep learning approaches, for which the segmentation is also possible, again using the entire image as input and a mask with each object of the scene divided as target, exploiting not only the pixel spectral features, but also the spatial characteristics of the scene (Bonaccorso, 2017; Goodfellow, Bengio, and Courville, 2016). Therefore, AI models can be applied for extrapolating initial and boundary parameters of the mathematical models. In particular, in the context of volcanic eruptions monitoring, the combined use of AI and cloud computing can help to process satellite data, extracting spatial and spectral characteristics of the lava flows. Moreover, analytical models can be applied to extrapolate some volcanic quantitative parameters, as the area and volume of a lava flows.

Extracting lava flow information by combining field observations, satellite data, analytical and intelligence models and cloud computing has immediate applications to the real-time monitoring of eruptions using numerical modeling, exploiting them to extract the mathematical models initial and boundary parameters. The following subsections will show applications of AI for extrapolating these parameters. Appendix B will show some additional technical details of the satellite data used.

2.2.1 Model inputs from satellite and ML

Different kinds of input can be extracted by satellite data using ML or DL models, to be given in input to the physical-mathematical models. Examples are the spectral data, as the spectral response of the lava, the spatial data, as the map of the lava flows, or a binary index for the presence or not of volcanic anomalies. The following subsections will show these input data.

Spectral data as input

The spectral data given in input to a physical-mathematical, but also ML model for volcano monitoring, is the spectral response. I exploited the theoretical basis of physical-mathematical models that describe the spectral responses of the surfaces and the satellite optical images (from visible to infrared bands), acquired by ESA Sentinel-2 MSI satellites, to investigate the behavior of the spectral response of lava flows (Amato et al., 2023b).

In general, the spectral response (Blackett, 2017; Head, Maclean, and Carn, 2013) is the absolute reflectance or radiance of an object as a function of solar radiation wavelength under specific environmental conditions. Reflectance spectroscopy is the technique used to analyze spectral data in the visible and infrared region of the spectrum, to identify different materials based on their reflectance characteristics. Thus, in volcanic area, a deeper knowledge about properties of lava flows can be inferred by investigating this spectral measure. Moreover, the growing number of satellite images available, with different spatial, temporal, and spectral resolutions, allows to explore the spectral characteristics of lava flows from Space (Harris, 2013).

The spectral response of surfaces covered by lava flows acquired by satellite sensors is affected by several factors, that are *exogenous*, mostly associated to the passing of years, like weathering, with revegetation and oxidation processes, and *endogenous*, like lava chemical composition and physical characteristics. In particular, old basaltic lava flows have a higher value of reflectance compared to the younger ones. This is mostly due to weathering and vegetation recovery (*e.g.*, lichen and moss), oxidative processes, water reflectance and soil coverage (*i.e.*, tephra fall-out) (Spinetti et al., 2009; Corradino et al., 2019a). Physical properties, as the different lava textures due to the slope changes in the lava path, the thickness, temperature, grain size, roughness of the lava flows and viscosity, may also affect the measured spectral response of a lava flow (Li et al., 2015; Rogic et al., 2022). Lava flows also differ in chemical composition. It is possible to chemically classify the lava in term of weight percent of silica (SiO_2), *i.e.*, the relative percentage in 100 g of compound, in macro areas, that are ultrabasic lava (37 to 45 wt% of SiO_2, *e.g.*, foiditic case), basic lava (45 to 52 wt% of SiO_2, *e.g.*, basaltic case), intermediate lava (52 to 63 wt% of SiO_2, *e.g.*, andesitic case) and acid lava (63 to 77 wt% of SiO_2, *e.g.*, rhyolitic case) (Le Bas, 2000). It is questionable whether changes in physics and chemical composition of the lava flows affect the spectral response.

All objects emit electromagnetic radiation if their temperature is above 0 K. The intensity of the radiation is dependent on some properties of the object, as the surface temperature and the emissivity. Hence, if the radiation emitted is known, the surfaces can be characterized from that one. Satellite sensors typically measure upwelling electromagnetic radiation in terms of spectral radiance L [W m^{-2}sr^{-1}m^{-1}] based on pre-launch or in-orbit calibration. The dependence of the overall spectral radiance L on wavelength λ [μm] and temperature T [K] is given by the Planck's formula (Planck, 1901), in Eq. (2.8)

$$L(\lambda, T) = \frac{2hc^2}{\lambda^5(exp(\frac{hc}{\lambda kT}) - 1)}, \tag{2.8}$$

with $h = 6.6256 \cdot 10^{-34}$ J s the Planck constant, $c = 2.9979 \cdot 10^8$ m s^{-1} the light speed and $k = 1.38 \cdot 10^{-23}$ J K^{-1} the Boltzmann gas constant. The Eq. (2.8) is related to a blackbody, *i.e.*, a body that totally emits the energy absorbed; in contrast, a real body emits only a portion of the total absorbed energy. The ability of a real surface to emit energy is given by the emissivity, Eq. (2.9)

$$\epsilon(\lambda) = 1 - \rho(\lambda). \tag{2.9}$$

The emissivity can also be defined as the ratio of the energy radiated from an object's surface to the energy radiated from a blackbody at the same temperature (*i.e.*, the emitting ability of a surface), with the following Eq. (2.10)

$$\epsilon(\lambda) = \frac{L(\lambda, T)}{L_{BB}(\lambda, T)}, \tag{2.10}$$

with $L(\lambda, T)$ the real object spectral radiance, and $L_{BB}(\lambda, T)$ the blackbody spectral radiance.

The reflectance $\rho(\lambda)$ (Eq. 2.11) is the ratio between energy (solar radiation) reflected and incident a body,

$$\rho(\lambda) = \frac{\pi L(\lambda, T) d^2}{ESUN_\lambda \sin\theta}, \tag{2.11}$$

with d the distance between Earth and Sun (in astronomical unit), $ESUN_\lambda$ the solar irradiance in [W m^{-2}m^{-1}], and θ the sun elevation, in degrees. $\rho(\lambda) \in [0, 1]$, in general, and $\rho(\lambda) = 0$, for a blackbody (Harris, 2013).

In absence of terrestrial heat sources, any observations of the Earth surface at low wavelengths are dominated by the reflected component. Depending on the surface reflectivity properties, different surfaces reflect different amounts of solar irradiance in the spectrum generating specific spectral responses (Harris, 2013; Blackett, 2017). Satellite remote sensors measure the Top of Atmosphere (TOA) radiance or reflectance of a surface at different wavelengths.

I investigated a variety of lava flows from different volcanoes worldwide using satellite data from visible to infrared bands, conducting a triple analysis, temporal-, physical-, and chemical- based, to search for specific patterns and determine more closely the factors affecting their spectrum (Amato et al., 2023b).

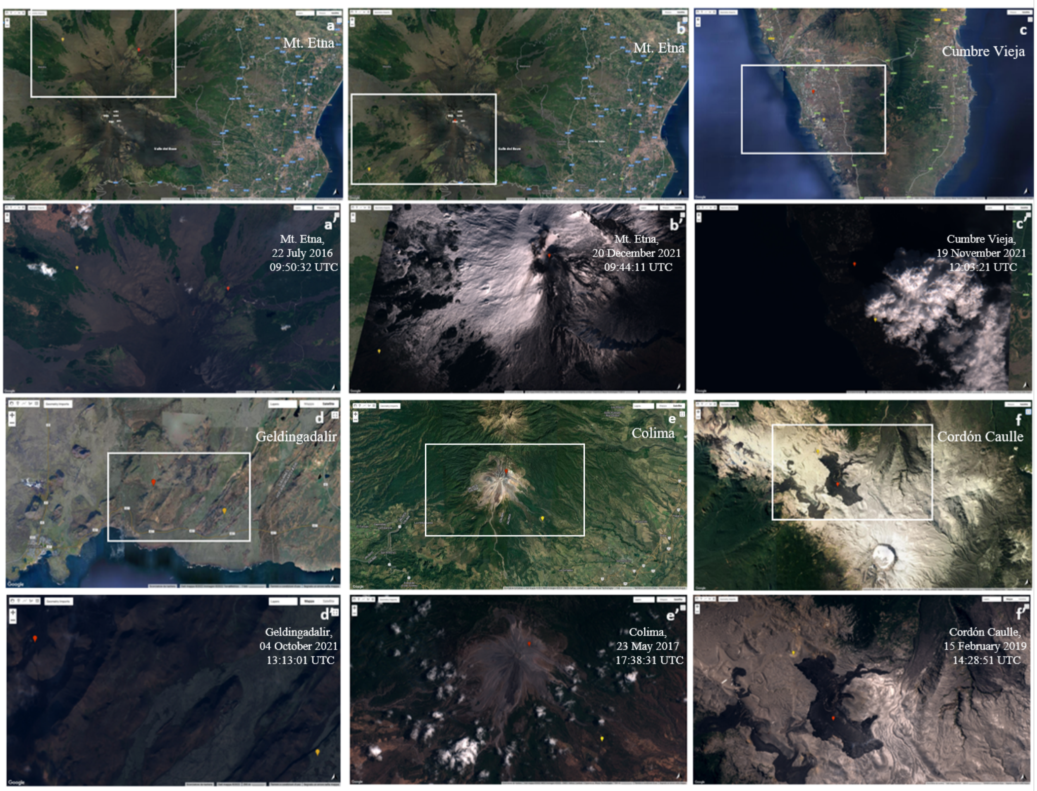
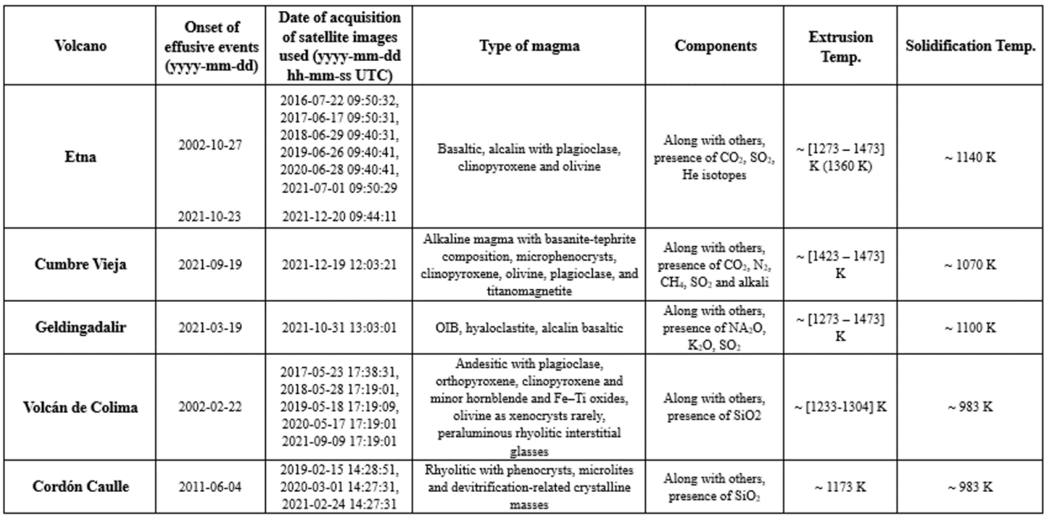


FIGURE 2.1: Volcanoes (a-f) and white box S2-MSI images in visible bands (a′-f′), map pin for the points (red: lava, yellow: background). All the images are obtained via Google Earth Engine.

The selected volcanoes are Mt.Etna, in Sicily (Italy), Cumbre Vieja, in La Palma (Canary Islands, Spain), Geldingadalir, in Reykjanes Peninsula (Iceland), Volcán de Colima, between Colima and Jalisco (Mexico), and Puyehue-Cordón Caulle volcano (Chile), to compare the spectral response of different kinds of lava. Figure 2.1 shows the volcanoes under examination (Fig. 2.1 a-f), with an example of S2-MSI image used for each case study (Fig. 2.1 a′-f′), with a visualization in the visible bands (S2 bands: red "B4", green "B3", blue "B2") of the white box portion of the volcano, red map pins for the points chosen for the lava case and yellow ones for the background case.

The data used for the analysis are the images acquired by the Sentinel-2 satellites (S2) from the European Space Agency (ESA) Copernicus mission, composed by a couple of identical satellites launched in 2015 and 2017 (S2A, S2B). As a reminder, the S2 MultiSpectral Instrument (MSI) has a good temporal resolution (2 - 5 days) and spectral resolution (13 spectral bands) and an high spatial resolution (10 - 60 m). I analyzed the S2-MSI Top of Atmosphere (TOA) reflectance data, a dimensionless measurement that describes the ratio between reflected and incident solar radiation on a given surface. All the spectral responses are collected via Google Earth Engine (GEE), a cloud-based platform capable to visualize and analyze satellite data in a rapid and efficient way (Gorelick et al., 2017).



Volcano	Onset of effusive events (yyyy-mm-dd)	Date of acquisition of satellite images used (yyyy-mm-dd hh-mm-ss UTC)	Type of magma	Components	Extrusion Temp.	Solidification Temp.
Etna	2002-10-27 2021-10-23	2016-07-22 09:50:32, 2017-06-17 09:50:31, 2018-06-29 09:40:31, 2019-06-26 09:40:41, 2020-06-28 09:40:41, 2021-07-01 09:50:29 2021-12-20 09:44:11	Basaltic, alcalin with plagioclase, clinopyroxene and olivine	Along with others, presence of CO_2, SO_2, He isotopes	~ [1273 – 1473] K (1360 K)	~ 1140 K
Cumbre Vieja	2021-09-19	2021-12-19 12:03:21	Alkaline magma with basanite-tephrite composition, microphenocrysts, clinopyroxene, olivine, plagioclase, and titanomagnetite	Along with others, presence of CO_2, N_2, CH_4, SO_2 and alkali	~ [1423 – 1473] K	~ 1070 K
Geldingadalir	2021-03-19	2021-10-31 13:03:01	OIB, hyaloclastite, alcalin basaltic	Along with others, presence of NA_2O, K_2O, SO_2	~ [1273 – 1473] K	~ 1100 K
Volcán de Colima	2002-02-22	2017-05-23 17:38:31, 2018-05-28 17:19:01, 2019-05-18 17:19:09, 2020-05-17 17:19:01 2021-09-09 17:19:01	Andesitic with plagioclase, orthopyroxene, clinopyroxene and minor hornblende and Fe–Ti oxides, olivine as xenocrysts rarely, peraluminous rhyolitic interstitial glasses	Along with others, presence of SiO2	~ [1233-1304] K	~ 983 K
Cordón Caulle	2011-06-04	2019-02-15 14:28:51, 2020-03-01 14:27:31, 2021-02-24 14:27:31	Rhyolitic with phenocrysts, microlites and devitrification-related crystalline masses	Along with others, presence of SiO_2	~ 1173 K	~ 983 K

FIGURE 2.2: Volcanoes under analysis, with the dates of the events and the S2-MSI images used, and properties of the lava analyzed. (Del Negro et al., 2004; Cappello et al., 2019; Amato et al., 2021a; Amato, 2022; Carracedo et al., 2022; Corradino et al., 2022; Zierenberg et al., 2021; Savov, Luhr, and Navarro-Ochoa, 2008; Heap et al., 2014; Magnall et al., 2019)

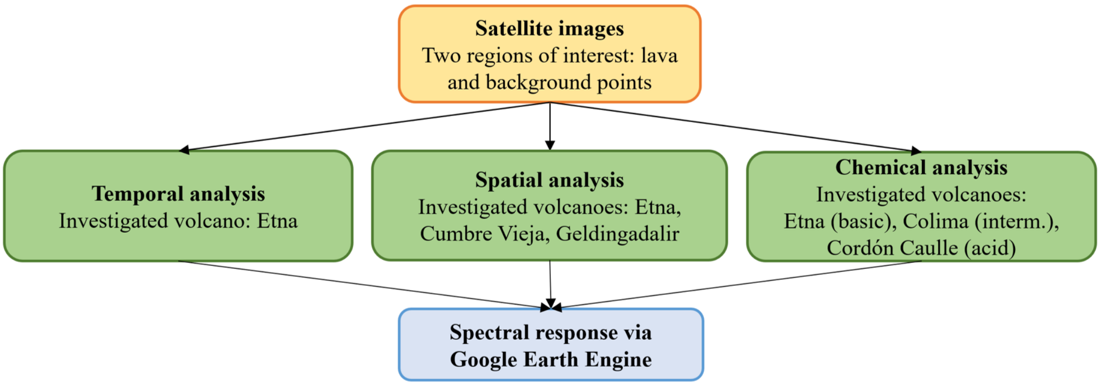

Satellite images
Two regions of interest: lava and background points
Temporal analysis
Investigated volcano: Etna
Spatial analysis
Investigated volcanoes: Etna, Cumbre Vieja, Geldingadalir
Chemical analysis
Investigated volcanoes: Etna (basic), Colima (interm.), Cordón Caulle (acid)
Spectral response via Google Earth Engine

FIGURE 2.3: Flow chart of the spectral response analysis methodology

I captured the spectral response of the points for the S2-MSI TOA reflectance bands between the visible and the infrared, *i.e.*, blue, green, red bands (B2, B3, B4, with spatial resolution 10 m), red-edge bands (B5, B6, B7, B8A, with spatial resolution 20 m), and Near InfraRed band (B8, with spatial resolution 10 m). Figure 2.2 shows a table with the volcanoes used as case studies, the relative dates of the events and of the S2-MSI images processed, and the properties of the relative lava analyzed. A flow chart giving a telling insight into the methodology adopted for this study is shown in Fig. 2.3.

For each volcano investigated, I considered two regions of interest, the area covered by recent lava flows (*i.e.*, "lava") not superimposed by subsequent lava flows, and the surrounding older volcanic rocks (*i.e.*, "background"), as control points. After I have picked a point for each region of interest (see Fig. 2.1), considering a neighborhood of radius 20 m of the point for the analysis, I have monitored the evolution of the spectral response for each sample. Firstly, I conducted a temporal analysis focusing on lava spectral changes in time (years), investigating the basaltic lava flow erupted at Mt.Etna on 27 October 2002 from a lateral fissure (see Fig. 2.2). Secondly, I conducted a physical analysis comparing the eruptions occurred in 2021 at volcanoes with same chemical composition (basic) and different physical properties, *i.e.*, Mt.Etna (23 October 2021 eruption), Cumbre Vieja (19 September 2021 eruption), and Geldingadalir (19 March 2021 eruption). More in detail, the lava chemical composition of the Cumbre Vieja 2021 eruption is *basanite*, ranging from ultrabasic to basic (Castro and Feisel, 2022) (see Fig. 2.2). Finally, I analyzed the spectral response in time of lava flows with different chemical composition, *i.e.*, basic, intermediate, and acid one, with samples from basaltic Etna (27 October 2002 eruption), andesitic Colima (22 February 2002 eruption) and rhyolitic Cordón Caulle (04 June 2011 eruption) volcanoes (see Fig. 2.2).

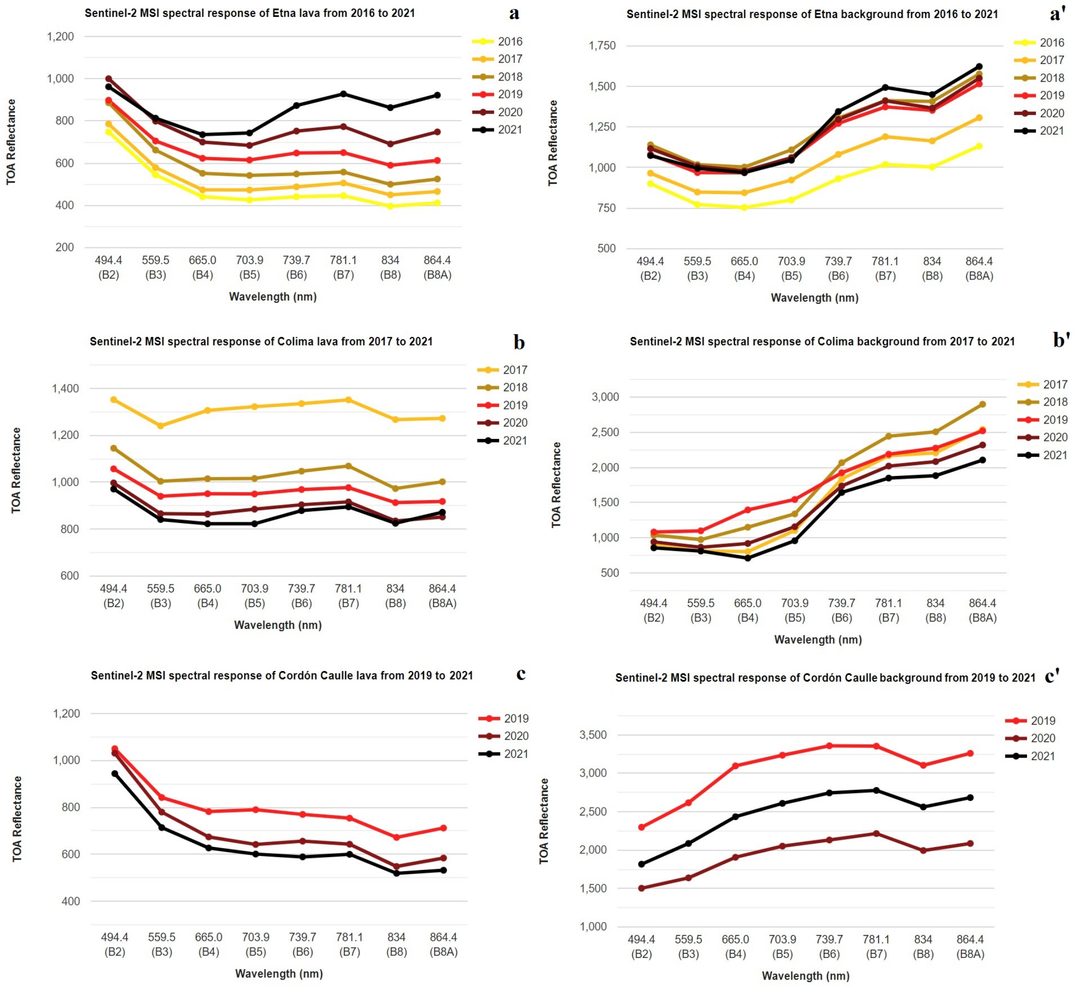


FIGURE 2.4: Spectral response of the lava (left) and background (right) points referred to the S2-MSI TOA reflectance images. a, a') Basic Mt.Etna 2002 lava flow, from 2016 (lower color gradation) to 2021 (higher color gradation), with growing lava spectral trend. b, b') Intermediate Colima 2002 eruption, from 2017 (lower color gradation) to 2021 (higher color gradation), with decreasing lava spectral trend (note that b graph is differently scaled to avoid squashed lines). c, c') Acid 2011 Cordón Caulle eruption, from 2019 (lower color gradation) to 2021 (higher color gradation), with decreasing lava spectral trend. All the data are obtained via Google Earth Engine, with reflectance scaled by 10000. Figure taken from (Amato et al., 2023b).

Temporal changes of the spectral response of the basic lava flow emplaced at Mt.Etna in 2002 are shown in Fig. 2.4 (a, a'). In this case, it is possible to see that the spectral response of Etnean basaltic lava increases significantly with over time (years) and a similar trend is not present in the background. A second analysis was conducted comparing different lava chemical compositions, searching for possible different trends (*e.g.*, for example different trends over time) that may be linked to this different composition. Moving to the intermediate and acid case, I found that the lava spectral response has an opposite trend, decreasing over time. Indeed, for the andesitic case, the spectral response of the 2002 Colima lava decreases in time and a specific trend is not present in the background (Fig. 2.4 b, b'). Also for the rhyolitic case, the spectral response of the 2011 Cordón Caulle lava decreases over time (years) and a specific trend is not present in the background (Fig. 2.4 c, c′).

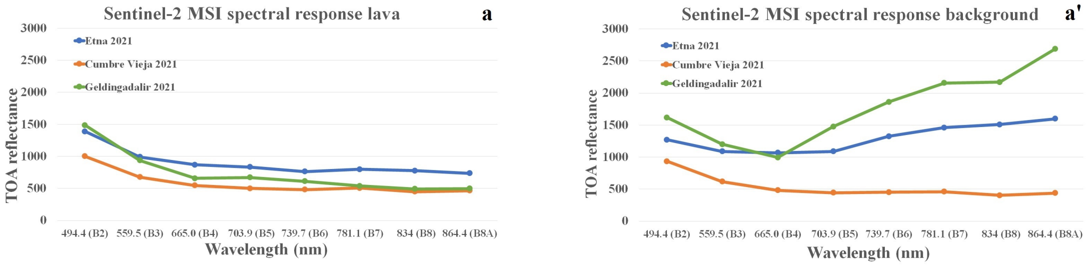


FIGURE 2.5: Spectral response of the lava (a) and background (a') points referred to the S2-MSI TOA reflectance images of Etna, Cumbre Vieja and Geldingadalir volcanoes, with narrower ranges of variability for lava case. All the data are obtained via Google Earth Engine, with reflectance scaled by 10000. Figure taken from (Amato et al., 2023b).

In addition, a physical analysis was conducted to investigate the changes of the spectral response of recent lava erupted in 2021 from ultrabasic-basic volcanoes (*i.e.*, Mt.Etna, Cumbre Vieja and Geldingadalir), placed in different locations and with different physical features (*e.g.*, medium thickness: Etna ∼ 2 m (INGV weekly bulletins at www.ct.ingv.it), Cumbre Vieja ∼ 12 m (Carracedo et al., 2022), Geldingadalir ∼ 30 m (Pedersen, 2016)). Figure 2.5 highlights the lava and background spectral responses for these volcanoes. For the lava case (Fig. 2.5 a), the reflectance curves are confined to a narrower range than the background one (Fig. 2.5 a'), with an average difference between maximum and minimum for all the bands of 330 for lava and 1130 for background.

Some points can be discussed. Temporal analysis of basaltic lava flows in the case of Mt.Etna (Fig. 2.4 a, a') highlights that the spectral response increases significantly over time, showing an ascending order of the reflectance curves of the years, and a random trend for the background control point (to confirm that the specific lava trends depend only on the properties of the lava, they are not present in the background cases). Comparing this result with (Spinetti et al., 2009), that shows similar results, the "young" lava of my research shows the same growing trend in time, with a peak at ∼ 780 nm. The differences at lower wavelengths with respect to (Spinetti et al., 2009) depend on the different type of sensor adopted thereby, namely the active higher spatial resolution LiDAR sensor. Changing the lava chemical composition, going toward an acidic composition, I found that the spectral response pattern is completely opposite, with a decreasing lava trend in time (Fig. 2.4 b, b', c, c′). When I consider lava flows with similar silica content and amount of time from the emplacement (Fig. 2.5), I can neglect the temporal and chemical dependencies and I can focus only on the different physical properties. The trends are similar and they remain in a narrow range across all the bands. The lava case presents reduced variability (Fig. 2.5 a), due to similar chemical composition (ultrabasic to basic one), respect to the background points (Fig. 2.5 a'), for which the larger range of variation is probably due to greater variability of background properties. In particular, results show that the variability of lava from different geographic areas is negligible, whereas the higher variability of the surrounding volcanic rocks (background) can be done by the different geographic location and consequently characteristic climate of each site. For each case study, uncertainty in the measured satellite reflectance due to the instruments used could affect measured spectral responses; this is the reason why I have considered for each map pin also its neighborhood of radius 20 m for the analysis.

To sum up, I have characterized changes occurring in the lava spectral response trends over time and with different physical and chemical characteristics, in the visible-infrared portion of the spectrum. The results were obtained using solely satellite observations and exploiting the theoretical physical-mathematical and chemical models that describe the spectral responses of the surfaces.

Work in progress Future works could bring to an extension of the study to lava flows with chemical composition which were not here analyzed, *e.g.*, ultrabasic lava flows with the passing of years, and to a physical analysis, described here for the basic case, also for other intermediate and acid cases in comparison with each other.

Spatial data as input

The spatial data given in input to a volcanic monitoring model may be several, linked to the area extension of the lava flow, the vent position, the maximum length or altitude reached, and so on. In cases of volcano monitoring, a spatial information linked to the areal extension of the lava flow may help to quantify the hazard linked to an eruption, and this map can be inferred by a ML algorithm. The information linked to the state of the volcano is another important input for the models. In volcanic context, a satellite image can be classified via a DL approach, detecting the elements inside, *e.g.*, the possible volcanic thermal anomalies in the scene, and giving a prediction of the volcanic activity.

I have developed two algorithms to reach these aims. The first approach allows to spatially map the lava flow field, obtaining the areal extension and a quantification of the lava flow area for a given eruption. To do this, I developed some ML models depending on different parameters, so as to be able to have a lava flow mapping in all the weather and eruptive conditions. The second approach exploits the advantages of a DL model to automatically detect the presence or not of volcanic anomalies, classifying an entire satellite image. These models can thus be used together, in a unique framework, starting from the second approach. Indeed, with the DL model, the volcanic anomalies can be detected and, subsequently, mapped using the ML approaches. These models are detailed below.

Mapping with ML A thermal anomaly may be referred to as a hotspot when it has a relatively high temperature in comparison to a reference value, *e.g.*, its surroundings area in the scene. Hotspot detection algorithms exploit spectral and spatial differences with respect to the background areas, analyzing the image with statistical or fixed-threshold approaches (*i.e.*, a priori setting of a threshold or statistical models), or data-driven ones (*i.e.*, machine or deep learning).

Threshold-based approaches compare the input data with a range of value, set a priori with a preliminary study over a group of data (Genzano, Pergola, and Marchese, 2020), or statistically selected, calculating mean and standard deviation for each image (Coppola et al., 2016). These kinds of techniques, computationally light, do not allow generalization to different test cases (*e.g.*, mapping lava flows from other volcanoes), because the thresholds are strictly dependent on the dataset used to obtain them and they are not applicable to cases other than those present in the original dataset or similar. To overcome problems like this, machine learning techniques are recommended. I have developed a framework to track in space and time lava flows behavior using ML approaches.

General mapping framework Mapping lava flows means spatially identifying the areal extension of the lava field, obtaining a map of the lava field. This brings also to a quantitative estimation of the relative area. To have a final approach applicable to different weather and eruptive conditions, I have developed a set of ML models, interchangeable and capable to return a map of the hotspots in different conditions. I have initially considered the Random Forest (RF), a totally data-driven model to classify data, exploiting if-else conditions over the samples with high generalization performances in classification tasks. It is trained over a set of images, exploiting visible to infrared spectral features, relevant in detecting volcanic anomalies, to map the anomalies in the scene and segment the lava flows and the background. I have demonstrated this ability sampling the training data from a set of S2-MSI satellite images and testing the model over different images. An approach like this returns an automatic pretrained model, capable to detect subtle/low to intense anomalies, and ready to be used when a new image is available, from all the volcanoes around the world (Corradino et al., 2022).

In cases in which the RF approach is not applicable, for the specificity of the eruptive phenomena under analysis, *e.g.*, when the lava flow is cooled and the RF model is not capable to detect it for the absence of thermal anomalies, or when there are not S2 data available for that eruption (S2 model is actually trained only over S2 data), I have applied a different approach. In particular, scene-by-scene ML approaches are more usable respect to the precedent one. They are trained over a set of pixels of the satellite image and classify the remaining part, using, for example, K-Means, SVM, MD, and CART models, to exploit the specific characteristics of the specific scene under analysis (Amato, 2022).

In addition, when also these ML approaches are not applicable, because of bad weather conditions or lack of satellite data, I have developed a cumulative approach or a best fitting model, to obtain an estimation of the areal extension of the lava flows (Amato, 2022). In the first case, I have used a combination of images related to a temporal range in which different eruptions happened, accumulating a number of lava field in a unique high-spatial resolution satellite image. The other approach is the best fitting one, in which high-temporal resolution satellite images are considered, extracting a parameter linked to the radiant power of the lava field. This parameter is linked to the areal extension of the field. Therefore, a best fitting model can be applied to extract the value of the area that in a better way fit the temporal series of data.

Incidentally, a more in-depth lava flow spectral analysis may allow to construct a unique model for hotspot and cooled lava flow mapping, collecting a range of spectral values for different volcanic features and cases. In this way, it could be possible to develop a model capable to analyze the specific spectral response of each element in the scene and classify the data in each condition. This is one of the future development of the mapping approaches.

Some additional details of the developed framework will be provided in the following paragraphs.

Mapping with a pretrained model I developed a pretrained model, *i.e.*, an algorithm trained over a limited number of volcanic satellite images, that becomes capable to classify pixels and segment images never seen before. This model was applied to different volcanoes around the world, that are Cumbre Vieja, in La Palma (Canary Islands, Spain), Mt.Etna and Stromboli, in Sicily (Italy), Geldingadalir, in Reykjanes Peninsula (Iceland), and Pacaya, in Guatemala, classifying pixels with anomalies and background. This supervised classifier exploits spectral information provided by S2-MSI satellite images to learn discriminating information between thermal anomalies (class 1, named "thermal anomaly") and background (class 0, named "background"), being able to detect also lower intensities changes, and different background spectral features, *e.g.*, vegetation, snow, buildings, and/or clouds (Corradino et al., 2022).

The RF is trained over ESA S2-MSI Level-1C TOA reflectance images. The TOA reflectance measurement is converted to radiance in [W $m^{-2}sr^{-1}\mu m^{-1}$], to consider how much radiation is emitted by a thermal anomaly, and not how much solar irradiation is reflected by the body, and six-band images are fed to the classifier as input data. In detail, two different classifiers have been developed and compared; the first one (RF1) with a input feature vector composed by three bands (red B4, SWIR1 B11, and SWIR2 B12), and three normalized indices, defined as *Normalized Hotspot Indices (NHI)* (Genzano, Pergola, and Marchese, 2020), that are $NHI(SWIR1, NIR)$, $NHI(SWIR2, SWIR1)$, and $NHI(SWIR2, NIR)$, with

$$NHI(x, y) = \frac{x - y}{x + y}. \tag{2.12}$$

The second classifier (RF2) takes as inputs images composed by three visible bands, one NIR band and two SWIR bands (blue B2, green B3, red B4, NIR B8A, SWIR1 B11, SWIR2 B12), better discriminating volcanic anomalies with respect to heterogeneous backgrounds through the spectral response of the surfaces. The target labels have been obtained with a manual inspection of the high-resolution satellite images. 100 trees are chosen to compose the two RFs, that is a good trade-off between complexity and performances of the model. I used three performances indices to quantify the goodness of the models, *i.e.*, accuracy (Eq. 2.1), precision (Eq. 2.2), recall (Eq. 2.3). In addition, I conduct a *feature importance* analysis, that is a study to understand how much a variable is discriminative for the classification task of the RF model. For this analysis the features are selected with Gini impurity or information gain and each feature decreases the impurity of the split (Breiman, 2001; Menze et al., 2009). The feature with the highest degrowth is selected for the internal node. This degrowth can be measured, obtaining an average value over all the trees, that is the final value of importance (Menze et al., 2009).

I tested the model over different eruptions, 21st of February 2021 Mt.Etna eruption, 27th of July 2019 Stromboli eruption, 25th of September 2021 Cumbre Vieja eruption, and 31st of October 2020 Pacaya eruption, and compared the results of the two RF (RF1 and RF2) and with a traditional threshold-based algorithm (Marchese et al., 2019). Fig. 2.6 and 2.7 show the outputs over the test set of the fixed-threshold model (FT), RF1 and RF2. I also calculated the average performances indices, comparing the results of FT, RF1 and RF2, as it is shown in the Table 2.1.

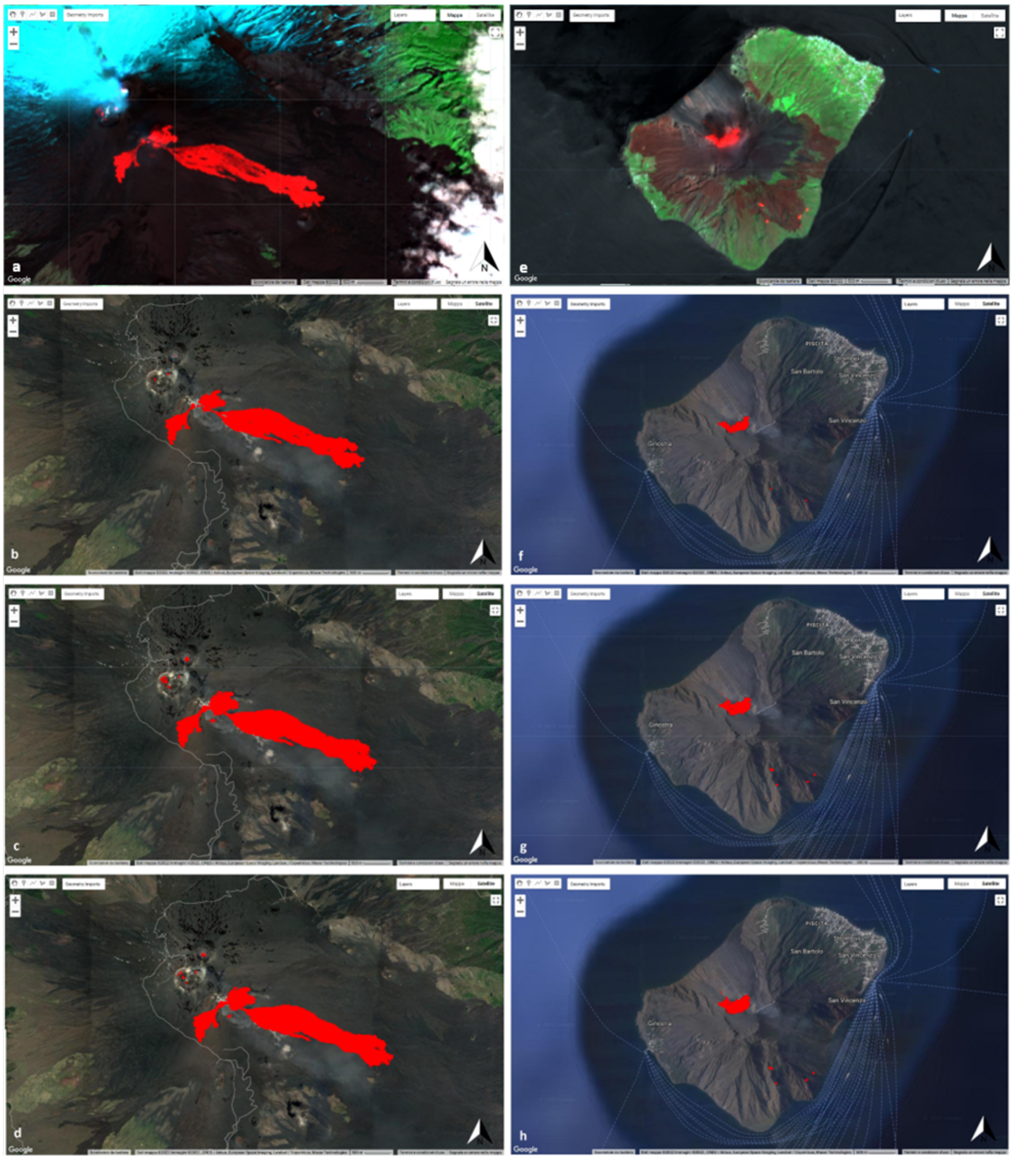


FIGURE 2.6: S2-MSI False RGB (B12-B8A-B5) image, used as test case. Comparison between fixed-threshold, RF1 and RF2 outputs for 21st of February 2021 Mt.Etna eruption (a-d), and 27th of July 2019 Stromboli eruption (e-h). All the images are captured via Google Earth Engine. Figure taken from (Corradino et al., 2022)

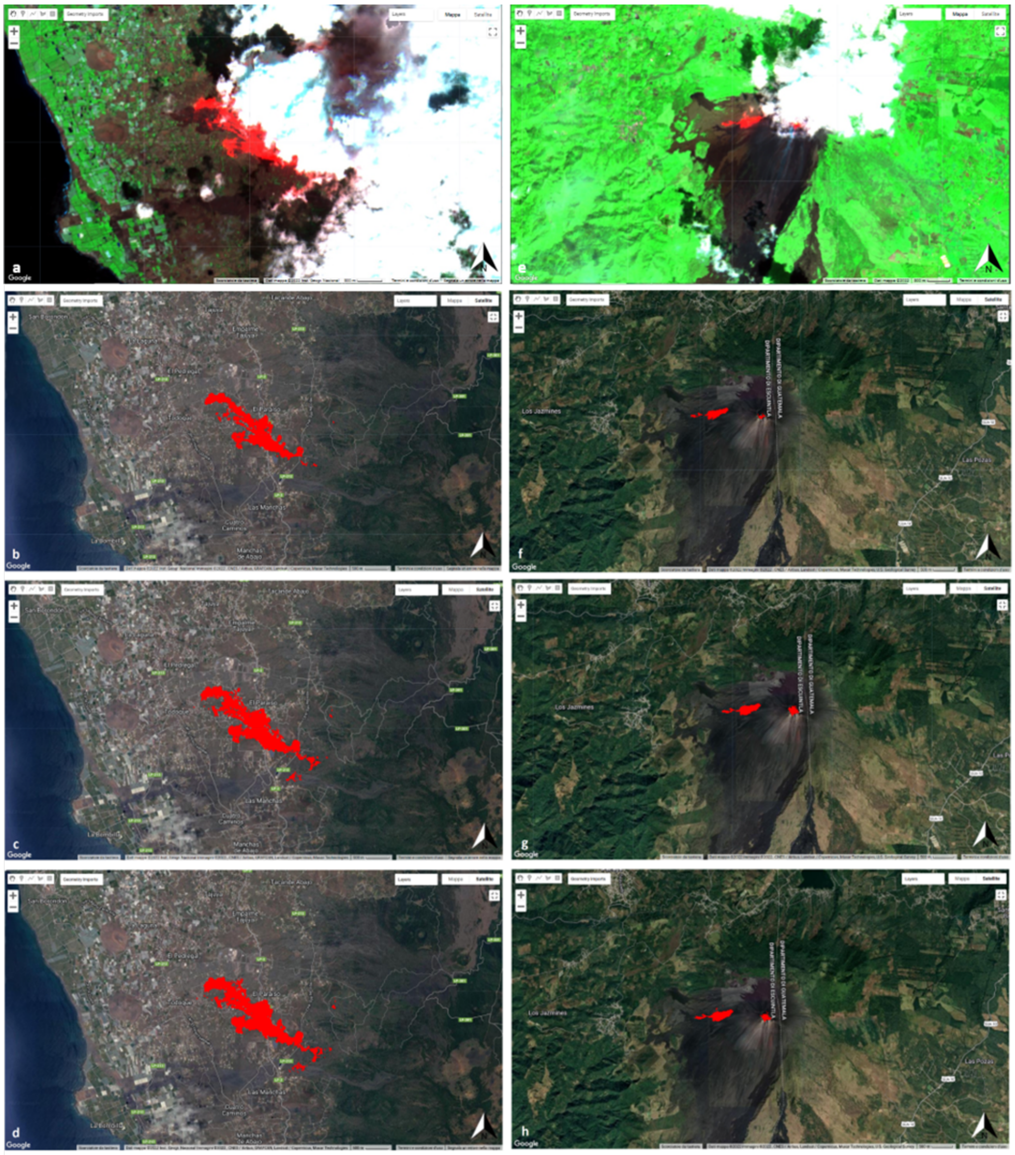


FIGURE 2.7: S2-MSI False RGB (B12-B8A-B5) image, used as test case. Comparison between fixed-threshold, RF1 and RF2 outputs for 25th of September 2021 Cumbre Vieja eruption (a-d), and 31st of October 2020 Pacaya eruption (e-h). All the images are captured via Google Earth Engine. Figure taken from (Corradino et al., 2022)

TABLE 2.1: Average performance indices comparison between fixed-threshold (FT), RF1 and RF2 outputs

	FT	RF1	RF2
Accuracy	0.83	0.90	0.91
Precision	0.99	0.92	0.91
Recall	0.79	0.87	0.89

Comparing the results of the RF1 and RF2 models with each other and with respect to the FT one, the RFs performs better than threshold-based models. In terms of performances indices, RF2 is better than RF1, because the use of visible bands helps in recognizing the background pixels spectral response, differentiating them respect to the anomalies pixels, especially when reflected radiance become dominant, for example for low emitted radiance. In term of feature importance, the RF1 features are almost all equal important; in contrast, in RF2 the infrared bands (NIR, SWIR1, SWIR2) are more important than the others, in recognizing the anomalies.

Mapping with scene-by-scene models and fitting If RF is not applicable to map anomalies, due to extremely low thermal anomalies (*i.e.*, in cooling or cooled lava flows) or lack of S2 data, a scene-by-scene ML approach can be applied. I developed several models of this type, that are the unsupervised K-Means, and the supervised SVM, MD, and CART, training them to classify pixels and segment lava and background in every satellite image, exploiting the specific spectral responses of the pixels of the scene under analysis.

The different ML models are applied to classify each pixel of the scene in two classes, *i.e.*, "lava" and "background", finally segmenting the scene. These models are trained with different types of high-spatial resolution satellite data (TOA reflectance images), with different sensors, *e.g.*, Sentinel-1 SAR and -2 MSI, Landsat 8 OLI/TIRS, and Terra ASTER images, due to the relative availability. Also the number of bands used can be variable, based on need and availability, as each model is trained on a Scene-by-Scene (SbS) basis. Therefore, different combinations of bands are used, 3D images (*e.g.*, NIR, SWIR1, SWIR2 bands) or 6D images (blue, green, red, NIR, SWIR1, SWIR2), with a combination of different bands also using mathematical operations between bands to combine the information (Corradino et al., 2021a) or the NHI indices (see Eq. 2.12). As for the RF model, the target data are obtained manually drawing polygons over the image, choosing the pixels with the aim to emphasize specific features of the relative classes.

The models are applied over the 2020-2021 Mt.Etna paroxysmal summit sequence. The output maps obtained were compared with the relative ones published for each event in the INGV weekly bulletins (at https://www.ct.ingv.it/), and a good agreement was found. In Fig. 2.8 the maps of the test events are shown.

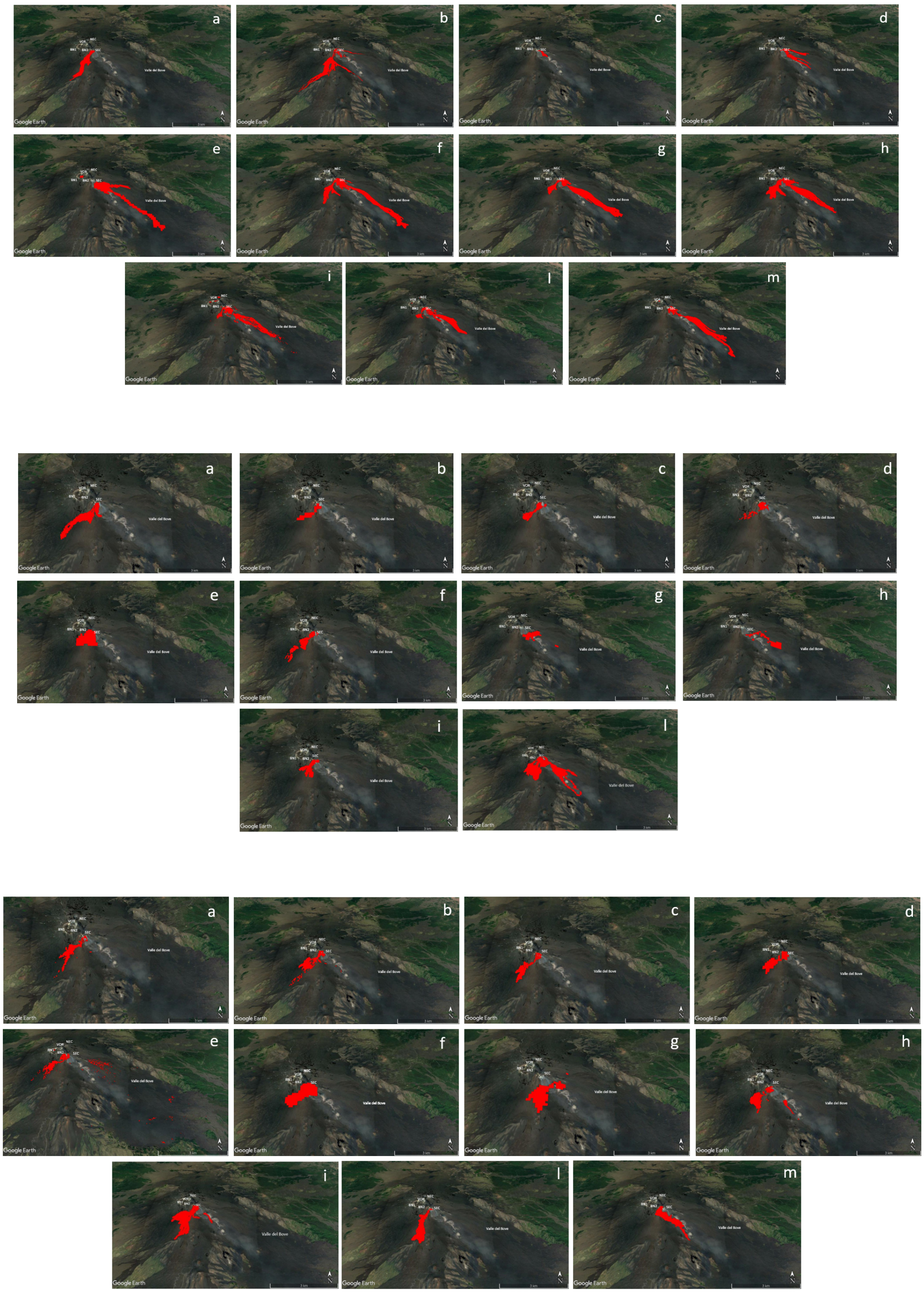


FIGURE 2.8: Upper block: events of 14 December 2020 (a), 22 December 2020 (b), 17 January 2021 (c), 18 January 2021 (d), 16 February 2021 (e), 18 February 2021 (f), 21 February 2021 (g), 23 February 2021 (h), 24 February 2021 (i), 28 February 2021 (l), 02 March 2021 (m). Middle block: events of 19 May 2021 (a), 21 and 22–23 May 2021 (b), 24 May 2021 (c), 30 May 2021 (d), 02 June 2021 (e), 12 June 2021 (f), 14 June 2021 (g), 16 June 2021 (h), 17 June 2021 (i), 19–27 June 2021 (l). Bottom block: events of 01 July 2021 (a), 04 July 2021 (b), 06 July 2021 (c), 08 July 2021 (d), 14 July 2021 (e), 20 July 2021 (f), 31 July 2021 (g), 08 August 2021 (h), 29 August 2021 (i), 21 September 2021 (l), 23 October 2021 (m). Figure taken from (Amato, 2022)

Due to its extreme flexibility, such an approach is applicable both to cases of hot thermal anomaly, and cases of cooling or already cooled lava, with different weather and eruptive conditions. The only cases in which this approach is not applicable are when there are no satellite images available for that event, or when the images are completely covered by clouds. In cases like these, a *cumulative approach* can be applied. An example of cumulative approach is the one applied for the lava fields between the 19 and the 27 June 2021 (Amato et al., 2021a). In that period, the adverse weather conditions made it impossible to use the satellite images, because totally covered by clouds, making indistinguishable the different lava flow field for each event. I have however estimated the total areal coverage of the lava flows of this period with a cumulative approach. In detail, the ratio between two SAR images, fused with a Sentinel-2 image of the 23 June 2021 (with the NI index for the bands used), an ASTER image of the 25 June, and a Landsat 8 image of the same date (with the NI index for the bands used) have been used to emphasize the specific characteristics and increase the information available (Corradino et al., 2021a). The bands of the image obtained have been normalized in the same range and some filters were used to improve the performances of the segmentation, finally obtaining the cumulative map for the entire period.

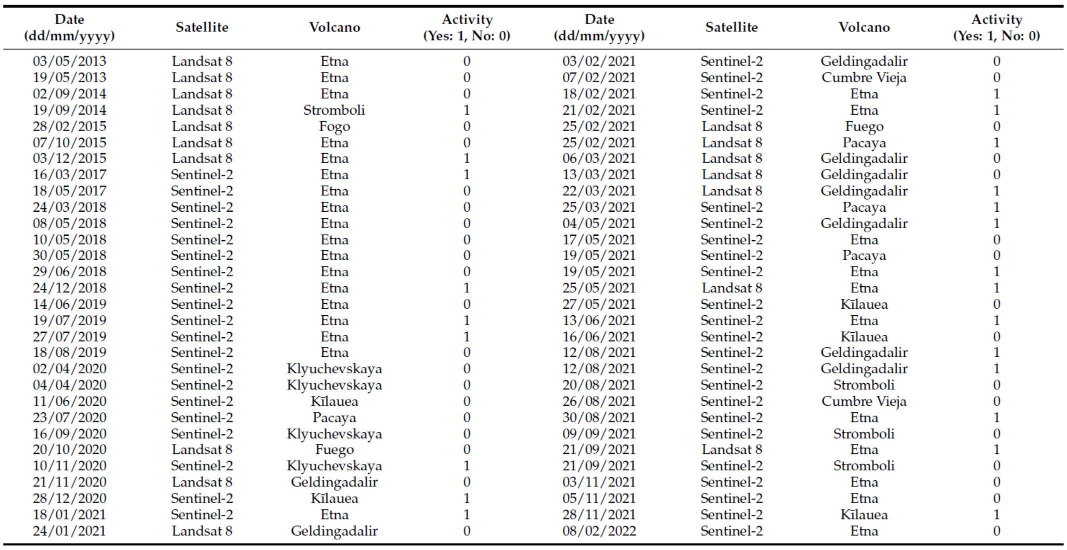
Detecting with DL The state of a volcano is a fundamental parameter in volcano monitoring. Therefore, detecting whether high temperature volcanic features are present in a scene can be an important model input parameter to recognize the volcanic behavior.

I developed an automatic monitoring system, based on a DL model, to detect the presence of volcanic thermal anomalies associated with eruptive activity, linked to active flows (such as lava flows, pyroclastic flows, warm lahars) and erupting vents (such as lava lake, lava fountains, hot fumaroles). In particular, I explored the potential of a DL Convolutional Neural Network (CNN), giving in input polar high-spatial resolution satellite data in the infrared bands (*i.e.*, NIR, SWIR1, SWIR2) and applying advanced learning techniques to reduce the training times and improve the accuracy of the model (Amato et al., 2023a). I applied the *transfer learning* approach, that is a learning technique based on the concept to transfer the knowledge of a pretrained model to a new domain. By exploiting the fine-tuning approach of the parameters using new images, it is possible to readjust the model to a new domain with the same task, without wasting time or computational resources (Yang et al., 2020). I have also applied an *ensemble learning*, another learning approach in which multiple models are combined together, to achieve better predictive performances than when using a model alone (Lin et al., 2017).

I have chosen the SqueezeNet CNN model (Iandola et al., 2016); this is a compact and fast CNN, largely applied for classification tasks. Its principal repeating module, the "fire" module, is sequentially composed by a "squeeze" convolutional module (with three 1×1 convolution filters), a ReLU activation function, an "expand" module (with four 1×1 convolution filters and four 3×3 convolution filters), a ReLU activation function. At the end of the architecture, a Softmax function is used for the classification. The SqueezeNet most famous version is already trained over ImageNet dataset, composed of around 14 million annotated images belonging to 1000 different classes (http://www.image-net.org/, (Deng et al., 2009)). Thus, I downloaded a pretrained SqueezeNet version over this dataset from the SqueezeNet website (https://github.com/forresti/SqueezeNet, model 1.0 for this study) and retrained it with the transfer learning technique to readjust the parameters of the network to the new image dataset.

For this reason, I created a new volcanic dataset. I built the volcanic dataset by sampling 135 images from the Sentinel-2 satellite and 65 images from the Landsat 8 satellite, balancing about the same number of images from each sensor between class 1 and class 0. The images belong to the 9 different volcanoes, Cumbre Vieja, Mt.Etna, Pico do Fogo, Fuego, Geldingadalir, Kilauea, Klyuchevskaya, Pacaya and Stromboli, acquired between 2013 and 2022, divided into two classes. The first class is related to the presence of volcanic thermal anomalies and it is defined as "Yes Anomaly", containing images with lava flows and active vents; the second one is linked to the absence of anomalies and it is defined as "No Anomaly", containing images with volcanoes at rest. A total of 100 images have been sampled for the first class and a total of 100 images for the second class. Generally speaking, a CNN needs a balanced training datasets of images to differentiate into classes by the presence or absence of volcanic thermal anomalies. This is why I have chosen the same number of images for the two classes. For representativeness of the scenes, I choose images for class 0 with different characteristics, *e.g.*, presence of clouds, snows, sea in the scene, or building areas. I have assigned every image to the correct class by carefully analyzing the eruptive activity of each volcano both using the volcanological bulletins (*e.g.*, from Smithsonian Institution, Global Volcanism Program, available online: https://volcano.si.edu/, or from INGV National Institute of Geophysics and Volcanology, available online: https://www.ct.ingv.it/) and manually inspecting the scenes. Similarly, I have defined the "ground truth" labels (*i.e.*, the real labels of the scenes) either examining the volcanological bulletins or manually inspecting the scenes. The complete dataset used for the transfer learning approach is available at the link: https://zenodo.org/record/7944343.ZGSp8HZBy5c, with the doi: 10.5281/*zenodo*.7944343.

Once the dataset is built, the first step in the classification process is to convert the TOA reflectance S2 and L8 images into the .png format (in the 0–255 scale) to feed the SqueezeNet model. Subsequently, the volcanic dataset is randomly divided into three independent subsets: a train set, a validation set and a test set and the pretrained model is then trained again over the training and validation set, with a small number of epochs and a limited number of images (with a learning rate alpha of 0.001, a batch size of 25 and 5 epochs for the fine-tuning (Goodfellow, Bengio, and Courville, 2016)). Finally, the SqueezeNet model is run 10 times over the test set (Fig. 2.9), combining the 10 outputs to obtain the final ensemble model. For each image of the test set, this ensemble model gives as output the mode of the 10 outputs of the 10 SqueezeNet models combined, improving the performances of the process. The goodness of the final ensemble model is quantified calculating the accuracy (Eq. 2.1), the precision (Eq. 2.2), the recall (Eq. 2.3) and the F1-score (Eq. 2.4) over the test set. The ensemble model reaches an accuracy of 98.3%, with a unique image misclassified over the total 60 images in the test set, a precision of 100.0%, a recall of 95.7%, and a F1 score of 97.8%, indicating a high-fidelity of the results. I chose the accuracy index as a representative index to compare the different SqueezeNet models, and the Fig. 2.10 shows the accuracy of the 10 different SqueezeNet models.



Date (dd/mm/yyyy)	Satellite	Volcano	Activity (Yes: 1, No: 0)	Date (dd/mm/yyyy)	Satellite	Volcano	Activity (Yes: 1, No: 0)
03/05/2013	Landsat 8	Etna	0	03/02/2021	Sentinel-2	Geldingadalir	0
19/05/2013	Landsat 8	Etna	0	07/02/2021	Sentinel-2	Cumbre Vieja	0
02/09/2014	Landsat 8	Etna	0	18/02/2021	Sentinel-2	Etna	1
19/09/2014	Landsat 8	Stromboli	1	21/02/2021	Sentinel-2	Etna	1
28/02/2015	Landsat 8	Fogo	0	25/02/2021	Landsat 8	Fuego	0
07/10/2015	Landsat 8	Etna	0	25/02/2021	Landsat 8	Pacaya	1
03/12/2015	Landsat 8	Etna	1	06/03/2021	Landsat 8	Geldingadalir	0
16/03/2017	Sentinel-2	Etna	1	13/03/2021	Landsat 8	Geldingadalir	0
18/05/2017	Sentinel-2	Etna	0	22/03/2021	Landsat 8	Geldingadalir	1
24/03/2018	Sentinel-2	Etna	0	25/03/2021	Sentinel-2	Pacaya	1
08/05/2018	Sentinel-2	Etna	0	04/05/2021	Sentinel-2	Geldingadalir	1
10/05/2018	Sentinel-2	Etna	0	17/05/2021	Sentinel-2	Etna	0
30/05/2018	Sentinel-2	Etna	0	19/05/2021	Sentinel-2	Pacaya	0
29/06/2018	Sentinel-2	Etna	0	19/05/2021	Sentinel-2	Etna	1
24/12/2018	Sentinel-2	Etna	1	25/05/2021	Landsat 8	Etna	1
14/06/2019	Sentinel-2	Etna	0	27/05/2021	Sentinel-2	Kīlauea	0
19/07/2019	Sentinel-2	Etna	1	13/06/2021	Sentinel-2	Etna	1
27/07/2019	Sentinel-2	Etna	1	16/06/2021	Sentinel-2	Kīlauea	0
18/08/2019	Sentinel-2	Etna	0	12/08/2021	Sentinel-2	Geldingadalir	1
02/04/2020	Sentinel-2	Klyuchevskaya	0	12/08/2021	Sentinel-2	Geldingadalir	1
04/04/2020	Sentinel-2	Klyuchevskaya	0	20/08/2021	Sentinel-2	Stromboli	0
11/06/2020	Sentinel-2	Kīlauea	0	26/08/2021	Sentinel-2	Cumbre Vieja	0
23/07/2020	Sentinel-2	Pacaya	0	30/08/2021	Sentinel-2	Etna	1
16/09/2020	Sentinel-2	Klyuchevskaya	0	09/09/2021	Sentinel-2	Stromboli	0
20/10/2020	Landsat 8	Fuego	0	21/09/2021	Landsat 8	Etna	1
10/11/2020	Sentinel-2	Klyuchevskaya	1	21/09/2021	Sentinel-2	Stromboli	0
21/11/2020	Landsat 8	Geldingadalir	0	03/11/2021	Sentinel-2	Etna	0
28/12/2020	Sentinel-2	Kīlauea	1	05/11/2021	Sentinel-2	Etna	0
18/01/2021	Sentinel-2	Etna	1	28/11/2021	Sentinel-2	Kīlauea	1
24/01/2021	Landsat 8	Geldingadalir	0	08/02/2022	Sentinel-2	Etna	0

FIGURE 2.9: List of the images contained in the test set, with date, satellite, volcano, and ground truth activity. Figure taken from (Amato et al., 2023a)

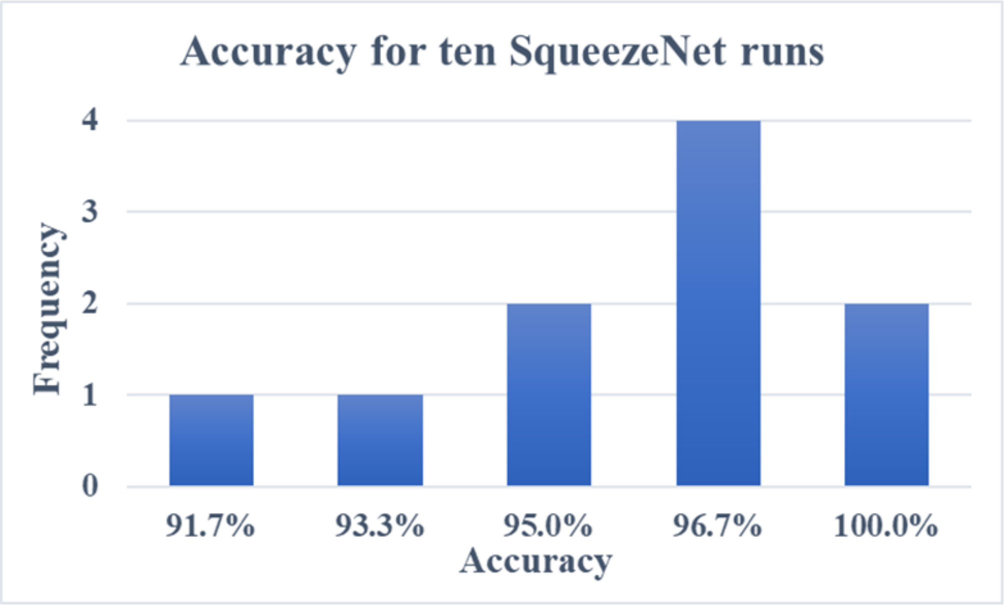


FIGURE 2.10: Accuracy of the 10 SqueezeNet models, retrained over the volcanic dataset. Figure taken from (Amato et al., 2023a)

The Fig. 2.11 shows examples of text images, misclassified by some of the singles SqueezeNet models, but classified in the correct way by the final ensemble one. In particular, Fig. 2.11 (b/e) is the unique image that remains misclassified also by the ensemble SqueezeNet, the others have been correctly classified.

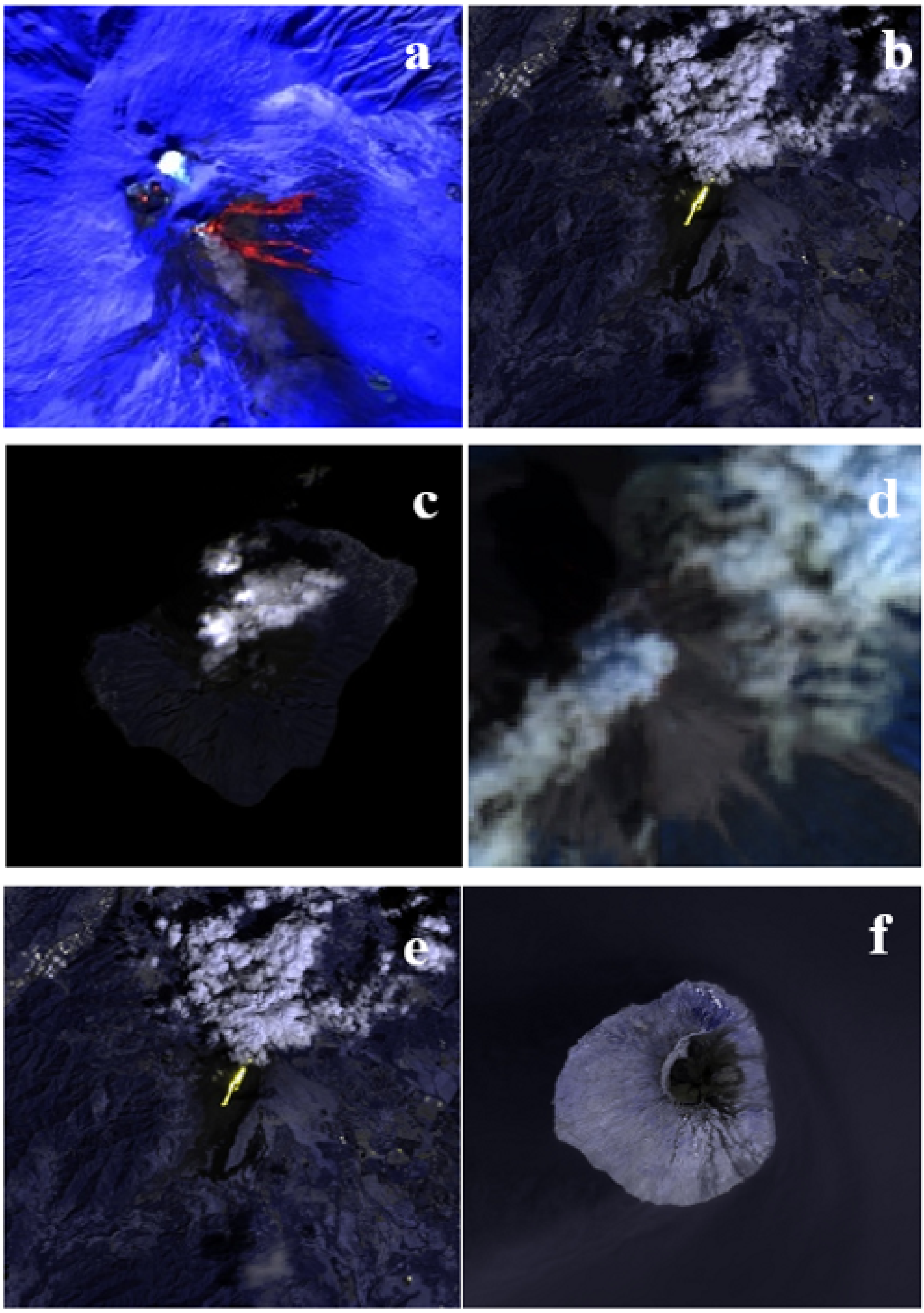


FIGURE 2.11: Examples of test case images, misclassified by three different fine-tuning runs: R1 case (a,b), images labeled as class 1, classified as belonging to class 0; R2 case (c,d), images labeled as class 0, classified as belonging to class 1; R3 case (e), image labeled as class 1, classified as belonging to class 0 and (f), image belonging to class 0, classified as belonging to class 1. All the images were downloaded from GEE. Figure taken from (Amato et al., 2023a)

In addition, I compared the result of the DL CNN SqueezeNet model with other traditional or ML approaches. The Squeezenet (SN) model has been compared with a fixed-based threshold method (Marchese et al., 2019; Genzano, Pergola, and Marchese, 2020), a statistical threshold method (Coppola et al., 2016), an unsupervised X-Means ML model (Bonaccorso, 2017), and a supervised Random Forest ML model (Corradino et al., 2022). Fig. 2.12 shows a comparison between the capability to detect an anomaly for the different models. In detail, when an anomaly is present in the scene under analysis, the ground truth dark red bar is present for that specific day; in contrast, when the anomaly is absent, that bar is absent. Moreover, when a model detects an anomaly, the bar with the color related to that model appears in the graph, otherwise, it is absent. Therefore, a good performance of each model analyzed is reached when the dark red ground truth bar is surmounted by all of the five light and dark green, orange, yellow, blue bars related to the outputs of the five models analyzed (*i.e.*, fixed threshold, statistical threshold, X-Means, RF, SqueezeNet), *e.g.*, 19 July 2019 and 10 November 2020 cases. Similarly, a good performance is also obtained when there are no bars related to a specific date, *e.g.*, 19 May 2013 and 29 June 2018 cases. All other cases present some misclassifications.

Fig. 2.13 shows a comparison between the accuracy of each model analyzed, calculated over the same test set (Fig. 2.9), using Sentinel-2 and Landsat 8 data (accuracy percentages here rounded to the unit for ease of analysis). It is worth noting that the RF model was trained only over S2 data, thus only a comparison between the S2 data is possible. Fig. 2.13 (a) makes in comparison Sentinel-2 data. The SqueezeNet (blue bar) has an accuracy of 100%, the fixed threshold method (light green bar) of 98%, the statistical threshold method (dark green bar) has a lower accuracy of 74% and the X-Means (orange bar) of 60%. The RF model (yellow bar), in this case, has the same accuracy of the SqueezeNet one. However, the proposed model is readily available for S2 and L8 data, allowing for the detection of thermal anomalies reducing the processing dead times and taking advantage of the first available acquisition between the S2-MSI and L8-OLI. Fig. 2.13 (b) makes in comparison Landsat 8 data. The SqueezeNet (blue bar) has an accuracy of 94% (only one L8 image misclassified), the fixed threshold method (light green bar) and the statistical threshold method (dark green bar) have a lower accuracy (respect to the S2 case) of 94% and 71% and the X-Means (orange bar) has an higher accuracy of 65%, respect to the S2 case. Fig. 2.13 (c) compares all the data, that are Sentinel-2 and Landsat 8. The SqueezeNet (blue bar) has an accuracy of 98% (only one L8 image misclassified), the fixed threshold method (light green bar) and the statistical threshold method (dark green bar) have an accuracy of 93% and 73% and the X-Means (orange bar) has an accuracy of 62%, respect to the S2 case.

Therefore, the SqueezeNet model has the highest accuracy (100%) with Sentinel-2 images and almost the maximum accuracy (94%) with Landsat 8 images, similar to the fixed threshold method. However, the SqueezeNet model has the advantage of be used with both Sentinel-2 and Landsat 8 data indifferently (satellites with sensors with similar spatial and spectral resolution), reaching the highest accuracy (98%) compared to the other methods in that case.

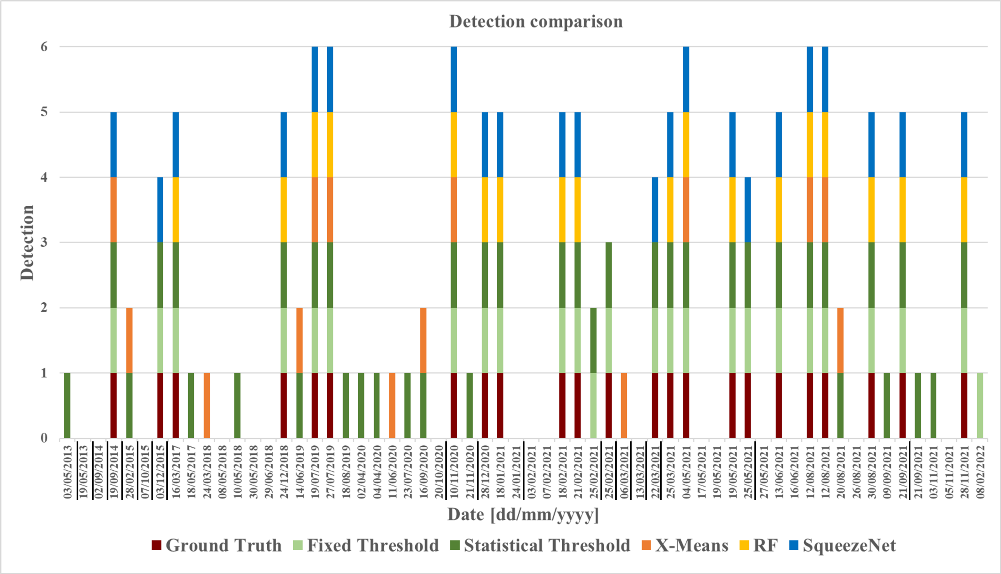


FIGURE 2.12: Comparison between the detection capability of the DL SqueezeNet model (blue bars), and the fixed-based (light green bars) and statistical (dark green bars) thresholds, the unsupervised (X-Means, orange bars) and supervised (RF, yellow bars) machine learning techniques. The Ground Truths are shown with dark red bars. Underlined data are related to Landsat 8 images used for those cases; otherwise, Sentinel-2 images are used. Figure taken from (Amato et al., 2023a)

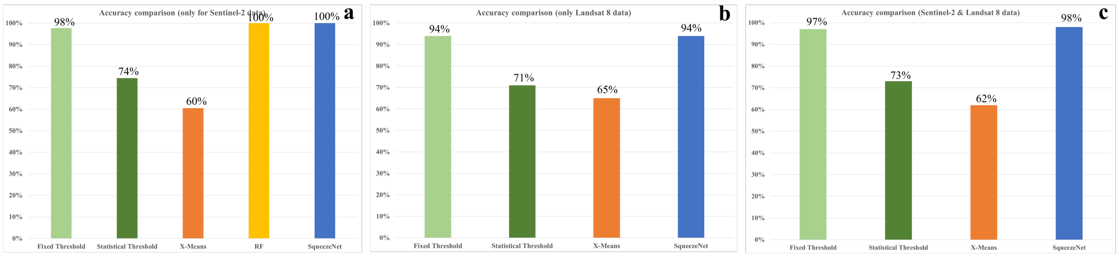


FIGURE 2.13: Accuracy comparison between DL SqueezeNet model, and threshold-based and ML models. Figure taken from (Amato et al., 2023a)

Summing up, with the DL SqueezeNet model, it is possible to detect thermal anomalies with high accuracy in less time, taking the first image available between the Sentinel-2 and Landsat 8 data, without the need to wait for the temporal step between two acquisitions for a single satellite (Sentinel-2 or Landsat 8).

2.2.2 Model inputs from satellite and analytical models

Collaterally to the estimation of volcanic input data from satellite using AI, it is possible to extract some other quantitative parameters linked to eruptive activity, to combine them with the first ones mentioned, using analytical models. In this way, several quantitative parameters can be extracted by satellite data using analytical methods, obtaining values to be given in input to the physical-mathematical models. Examples are the volume and area estimation of the lava fields. The following subsections will show these techniques.

Volume estimation from satellite data

I exploited the Volcanic Radiative Power (VRP) satellite product (Harris, 2013; Wooster et al., 2005) and the Time-Averaged Discharge Rate (TADR) (Coppola et al., 2013), from satellite, and analytical methods, to obtain the volume of the lava flows (Amato et al., 2021b). The VRP describes the radiative heat power emitted by an incandescent surface, for instance fire or volcanic lava. The TADR is an index that represents the lava extrusion rate, namely the amount of lava erupted during an eruption, averaged over a given time period. I considered the VRP satellite product directly obtained from the Space Agencies and I applied an analytical method to estimate this parameter. Satellite data can be processed to obtain it (*e.g.*, see (Corradino et al., 2019b; Vicari et al., 2011), with the MIR method (Wooster, Zhukov, and Oertel, 2003)), exploiting for example the Spinning Enhanced Visible and InfraRed Imager (SEVIRI) sensor, which is placed in Meteosat Second Generation (MSG) satellites. From the VRP, the TADR can be then calculated. This is, subsequently, useful to estimate the volume of erupted lava, multiplying it to the entire temporal range of the event. Some details are now given.

The VRP (in watt) is defined as in the Eq. 2.13

$$VRP_{true} = \epsilon\sigma A_{pixel} \sum_i p_i T_i^4, \tag{2.13}$$

where A_{pixel} is the pixel size (4.5 10^6 m^2 for the resampled SEVIRI pixels), p_i is the portion of the pixel at temperature T_i (in K), ϵ is the emissivity and σ is the Stefan-Boltzmann constant (5.67 10^{-8} W m^{-2}K^{-4}).

Exploiting the Eq. 2.13 and the following one, the Eq. 2.14,

$$VRP = \epsilon\sigma(T_{surf}^4 - T_a^4)A, \tag{2.14}$$

using the Stefan cooling problem for T_{surf} (Harris, 2013; Ganci et al., 2012; Wooster, Zhukov, and Oertel, 2003), it is possible to write

$$t = \left(\left(\left(\frac{T_h - T_a}{\epsilon\sigma(T_{surf}^4 - T_a^4) + h_c(T_{surf} - T_a)} - \frac{R_c R_r}{R_c + R_r}\right)k\right)\frac{1000}{2\lambda}\right)^2 \frac{1}{\kappa}, \tag{2.15}$$

with A the areal extension, $R_c = \frac{1}{h_c}$ and $R_r = \frac{T_h - T_a}{\epsilon\sigma(T_h^4 - T_a^4)}$, $\epsilon = 0.98$ emissivity, $\sigma = 5.670367\ 10^{-8}$ Wm^{-2}K^{-4} Stefan-Boltzmann constant, $h_c = 50$ Wm^{-2}K^{-1} convective heat transfer coefficient, $T_a = 298.15$ K ambient temperature, $T_h = 1273.15$ K initial temperature of lava, $k = 1$ Wm^{-1}K^{-1} thermal conductivity, $\lambda = 0.421$, $\kappa = 0.5$ mm^{2}s^{-1} thermal diffusivity.

To have a solution, I wrote the Eq. 2.15 in the biquadratic formulation Eq. 2.16

$$aT_{surf}^4 + bT_{surf} - c = 0, \tag{2.16}$$

with

$$a = \epsilon\sigma,$$

$$b = h_c,$$

$$c = \frac{(\epsilon\sigma T_a^4 + h_c T_a)(2\lambda\sqrt{\kappa t}(R_c + R_r) + 1000kR_cR_r) + (T_h - T_a)1000k(R_c + R_r)}{2\lambda\sqrt{\kappa t}(R_c + R_r) + 1000k(R_cR_r)}.$$

The acceptable real and positive analytical solution that I have calculated is the following Eq. 2.17

$$T_{surf} = -\frac{1}{2}\sqrt{\frac{\alpha}{\beta} - \frac{\gamma}{\alpha}} + \frac{1}{2}\sqrt{\frac{2b}{a\sqrt{\frac{\alpha}{\beta} - \frac{\gamma}{\alpha}}} - \frac{\alpha}{\beta} + \frac{\gamma}{\alpha}}, \tag{2.17}$$

with

$$\alpha = \sqrt[3]{\sqrt{3}\sqrt{256a^3c^3 + 27a^2b^4} + 9ab^2},$$
$$\beta = \sqrt[3]{2}\sqrt[3]{9}a,$$
$$\gamma = 4\sqrt[3]{\frac{2}{3}}c.$$

With this formulation, I can calculate the punctual values for the Eq. 2.14.
The TADR (in $\mathrm{m^3s^{-1}}$) can be calculated following the Eq. 2.18

$$TADR = \frac{VRP}{c_{rad}}, \tag{2.18}$$

with $c_{rad} = \frac{6.45\ 10^{25}}{X_{SiO_2}^{10.4}}$, X_{SiO_2} silica lava content, for example $X_{SiO_2} = 47.9$ wt% for the Mt.Etna case.

According to (Coppola et al., 2013), the strong effects that the bulk rheology has on the spreading and cooling processes of active lava must be taken into account in the calculation of the c_{rad}. Therefore, an uncertainty of ±50% c_{rad} must be considered. The final TADR thus is as in Eq. 2.19

$$TADR = \frac{TADR_{min} + TADR_{max}}{2}, \tag{2.19}$$

with

$$TADR_{min} = \frac{VRP}{c_{rad_{min}}},$$
$$TADR_{max} = \frac{VRP}{c_{rad_{max}}},$$

and $c_{rad_{min}} = 0.5\ c_{rad}$ and $c_{rad_{max}} = 1.5\ c_{rad}$.
Finally, I estimated the volume following the procedure composed by the following Eqs. 2.20, 2.21 and 2.22:

$$TADR_{mean} = \frac{TADR_i + TADR_{i-1}}{2}, \tag{2.20}$$

$$V_i = TADR_{mean} \cdot \Delta t, \tag{2.21}$$

$$Volume = \sum V_i, \tag{2.22}$$

with $TADR_k$ the $TADR$ at time $t = k$, and Δt the temporal range between two observations. All the results have to be incremented of a 30% in order to have the values for bulk and air bubbles.

Area estimation from satellite data

Once obtained the map of the lava field with ML approaches, it is also possible to calculate the area of the lava flow. The algorithm that I have developed to reach this aim consists of three steps. Considering the ML classification of the pixels, that returns a reference number which indicates whether the pixel belongs to the "background" or "lava" class (*i.e.*, 0 or 1), I masked only the "lava" pixels, respect to the other ones. After calculating the area of a single pixel of the satellite image under analysis (considering the spatial resolution of the relative satellite sensor), I then summed the number of pixels classified as "lava" and multiplied this numerical value by the area of the single pixel, obtaining a final estimation of the total area of the lava field (Amato, 2022).

When no images are available, it is not possible to extract a map for the lava flow using the ML techniques. In cases like this, a best fitting approach is a possible alternative way to have an estimation of the area of the lava field. In detail, geostationary high-temporal resolution satellite sensors, as the ESA EUMETSAT SEVIRI, follow a part of the terrestrial globe acquiring images with a temporal resolution in order of minutes. From these data, it is thus possible to extract parameters capable to follow the volcanic phenomenon. One of these is the VRP, fed as input to the best fitting model. The SEVIRI spectral radiance data in the Middle Infrared (MIR) at 3.9 μ m and Thermal Infrared (TIR) at 12.0 μ m, are processed to obtain this parameter (Corradino et al., 2019b; Vicari et al., 2011), calculated using the MIR method (Wooster, Zhukov, and Oertel, 2003). The VRP (in watt) is defined as in Eq. 2.13.

Thus, it is possible to consider the VRP physical formulation in Eq. 2.13, considering only the effect of the thermal anomalies (using A_{hot} and T_{hot}, to identify the hotspot pixels area and their temperature). It is also necessary to add the background contribution, that should be subtracted for real remote sensing data. In this way, it is possible to write $VRP = QA$, with $Q = \epsilon\sigma(T^4_{surf} - T^4_{background})$, where T_{surf} is computed with the Stefan Cooling Problem methodology (Ganci et al., 2012; Harris, 2013). Therefore, using the VRP product obtained and processed starting from the SEVIRI measurements, that is the VRP_{Seviri}, the best fitting algorithm is defined like in Eq. 2.23

$$\min_{A_{hot}} \frac{1}{m} \left(\sum_{i=1}^{m} (Q_i A_{hot} - VRP_{Seviri_i}) \right)^2, \tag{2.23}$$

minimizing the function inside the parenthesis to obtain the best area A_{hot} for the reference data (VRP_{Seviri}), with m the number of data in the final vector to minimize. The input vector of measurements is composed by the data belonging to the cooling curves associated with the eruptive (paroxysmal) event, *i.e.*, the data between the pick value of VRP during the eruption and the values of VRP near to the zero (Ganci et al., 2012). Minimizing this quantity, it is possible to finally estimate the better area for that specific set of data, *i.e.*, the area of the lava field for that eruption (Amato, 2022).

Thickness estimation from satellite data

Considering volume and area estimations, the average thickness of the lava flow can be obtained (Amato et al., 2021a) as in Eq. 2.24

$$\overline{Thickness} = \frac{Volume}{Area}. \tag{2.24}$$

To sum up, combining satellite images, machine learning techniques and analytical formulations, the volume, area, and thickness of lava fields can be estimated, and used as input to the model for the volcano monitoring.

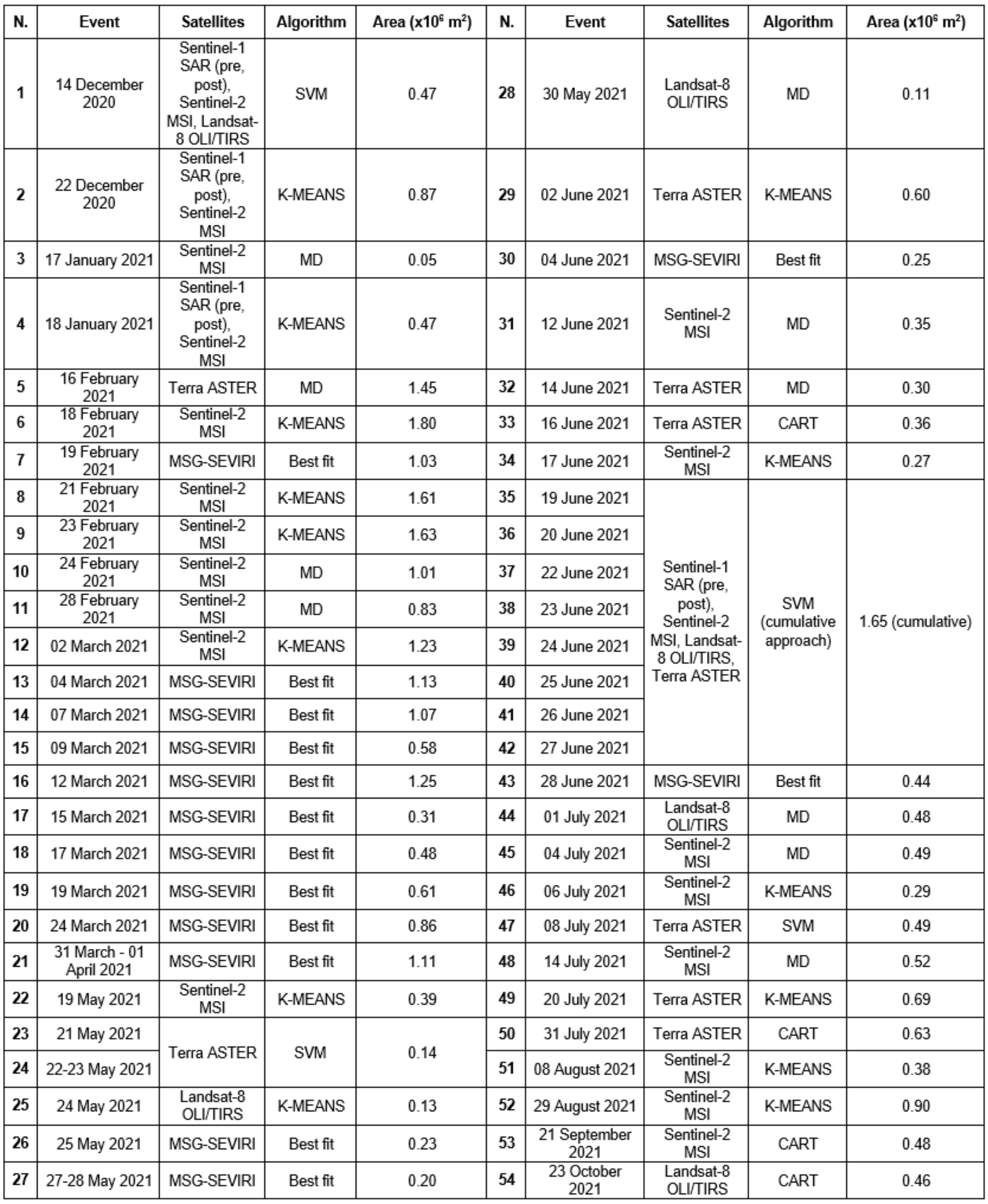
2.2.3 Incorporated results

Combining these different techniques and results, I have shown some incorporated results for the 2020–2021 paroxysmal summit events of Mt.Etna (Italy), with a list of quantitative parameters and maps (Amato, 2022).

Starting from the quantitative parameters, for example, I have considered two Mt.Etna eruptions, occurred on 18th February 2021 and 2nd March 2021. For the first one, I have found a volume estimation of 3.1 10^6 m^3 and an area estimation of 1.8 10^6 m^2. Combining these values, the lava field under analysis has a thickness of 1.7 m. For the second one, I have found a volume estimation of 2.0 10^6 m^3 and an area estimation of 1.2 10^6 m^2. Combining these values, the lava field under analysis has a thickness of 1.7 m. These results are in agreement with the known estimations (using the weekly INGV bulletins, available at https://www.ct.ingv.it/), *i.e.*, for the 18th February 2021, an area estimation of 2.0 10^6 m^2, and a volume estimation of 4.0 10^6 m^3, obtained imposing a thickness of 2.0 m. For the 2nd March 2021, the values are an area estimation of 1.2 10^6 m^2, and a volume estimation of 1.8 10^6 m^3, obtained imposing a thickness of 1.5 m.

Moreover, Fig. 2.14 shows a list of the 2020–2021 paroxysmal summit events of Mt.Etna (Italy), of the satellite used for obtaining the maps or the area estimation, of the algorithm chosen to obtain the areal extension and the relative outputs (shown in Fig. 2.8). In detail, ML models, the cumulative approach and the best fitting model have been used.

Finally, the 2020-2021 summit sequence of Mt.Etna eruptions produces a great quantity of lava fields, many of which overlapped one another. To demonstrate this statement, I created a lava flows map, overlapping all the maps obtained with the ML techniques. The different shades of grey indicate the number of lava flows overlapped, using a greyscale from black (that indicates a 0 value, no lava in the pixel) to white (that indicates a 26 value, so 26 lava flows overlapped in that pixel). It is worth noting that computing the area of all the events overlapped, without considering the thickness of each event and the region with more lava flows, the result is 5.40 km^2. In contrast, summing all the areas for the same ML maps events (see Fig. 2.14 for the single areas), the area is 21.53 km^2, confirming the idea that more events are overlapped. In addition, a map like this can be seen as a risk map, highlighting the volcanic areas most affected by lava inundation.



N.	Event	Satellites	Algorithm	Area (x10^{6} m^{2})	N.	Event	Satellites	Algorithm	Area (x10^{6} m^{2})
1	14 December 2020	Sentinel-1 SAR (pre, post), Sentinel-2 MSI, Landsat-8 OLI/TIRS	SVM	0.47	28	30 May 2021	Landsat-8 OLI/TIRS	MD	0.11
2	22 December 2020	Sentinel-1 SAR (pre, post), Sentinel-2 MSI	K-MEANS	0.87	29	02 June 2021	Terra ASTER	K-MEANS	0.60
3	17 January 2021	Sentinel-2 MSI	MD	0.05	30	04 June 2021	MSG-SEVIRI	Best fit	0.25
4	18 January 2021	Sentinel-1 SAR (pre, post), Sentinel-2 MSI	K-MEANS	0.47	31	12 June 2021	Sentinel-2 MSI	MD	0.35
5	16 February 2021	Terra ASTER	MD	1.45	32	14 June 2021	Terra ASTER	MD	0.30
6	18 February 2021	Sentinel-2 MSI	K-MEANS	1.80	33	16 June 2021	Terra ASTER	CART	0.36
7	19 February 2021	MSG-SEVIRI	Best fit	1.03	34	17 June 2021	Sentinel-2 MSI	K-MEANS	0.27
8	21 February 2021	Sentinel-2 MSI	K-MEANS	1.61	35	19 June 2021	Sentinel-1 SAR (pre, post), Sentinel-2 MSI, Landsat-8 OLI/TIRS, Terra ASTER	SVM (cumulative approach)	1.65 (cumulative)
9	23 February 2021	Sentinel-2 MSI	K-MEANS	1.63	36	20 June 2021			
10	24 February 2021	Sentinel-2 MSI	MD	1.01	37	22 June 2021			
11	28 February 2021	Sentinel-2 MSI	MD	0.83	38	23 June 2021			
12	02 March 2021	Sentinel-2 MSI	K-MEANS	1.23	39	24 June 2021			
13	04 March 2021	MSG-SEVIRI	Best fit	1.13	40	25 June 2021			
14	07 March 2021	MSG-SEVIRI	Best fit	1.07	41	26 June 2021			
15	09 March 2021	MSG-SEVIRI	Best fit	0.58	42	27 June 2021			
16	12 March 2021	MSG-SEVIRI	Best fit	1.25	43	28 June 2021	MSG-SEVIRI	Best fit	0.44
17	15 March 2021	MSG-SEVIRI	Best fit	0.31	44	01 July 2021	Landsat-8 OLI/TIRS	MD	0.48
18	17 March 2021	MSG-SEVIRI	Best fit	0.48	45	04 July 2021	Sentinel-2 MSI	MD	0.49
19	19 March 2021	MSG-SEVIRI	Best fit	0.61	46	06 July 2021	Sentinel-2 MSI	K-MEANS	0.29
20	24 March 2021	MSG-SEVIRI	Best fit	0.86	47	08 July 2021	Terra ASTER	SVM	0.49
21	31 March - 01 April 2021	MSG-SEVIRI	Best fit	1.11	48	14 July 2021	Sentinel-2 MSI	MD	0.52
22	19 May 2021	Sentinel-2 MSI	K-MEANS	0.39	49	20 July 2021	Terra ASTER	K-MEANS	0.69
23	21 May 2021	Terra ASTER	SVM	0.14	50	31 July 2021	Terra ASTER	CART	0.63
24	22-23 May 2021				51	08 August 2021	Sentinel-2 MSI	K-MEANS	0.38
25	24 May 2021	Landsat-8 OLI/TIRS	K-MEANS	0.13	52	29 August 2021	Sentinel-2 MSI	K-MEANS	0.90
26	25 May 2021	MSG-SEVIRI	Best fit	0.23	53	21 September 2021	Sentinel-2 MSI	CART	0.48
27	27-28 May 2021	MSG-SEVIRI	Best fit	0.20	54	23 October 2021	Landsat-8 OLI/TIRS	CART	0.46

FIGURE 2.14: 2020–2021 paroxysmal summit events of Mt.Etna (Italy), of the satellite used for obtaining the images, of the algorithm chosen to obtain the areal extension and the relative results (rounded to the second decimal digit). Figure taken from (Amato, 2022).

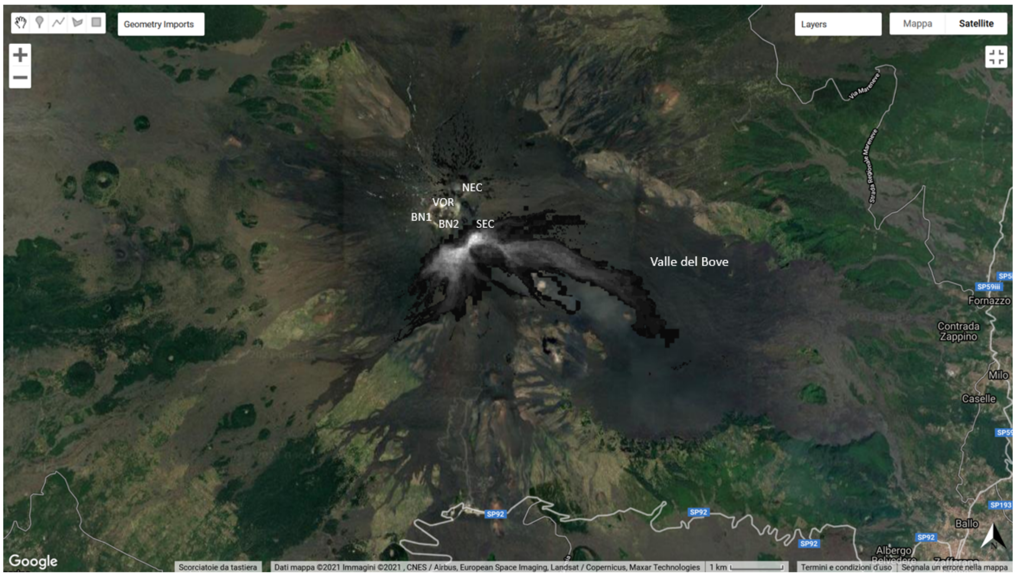

Geometry Imports
Layers
Mappa
Satellite
NEC
VOR
BN1
BN2
SEC
Valle del Bove
Fornazzo
Contrada Zappino
Milo
Caselle
SP92
SP193
Ballo
Albergo Belvedere
Google
Scorciatoie da tastiera
Dati mappa ©2021 Immagini ©2021 , CNES / Airbus, European Space Imaging, Landsat / Copernicus, Maxar Technologies
1 km
Termini e condizioni d'uso
Segnala un errore nella mappa

FIGURE 2.15: 2020–2021 paroxysmal summit events maps of Mt.Etna (Italy) overlapped (between December 2020 and October 2021, except for the 19 February, the period between 04 March and 01 April, 25 May, 28 May, 04 June and 28 June 2021 events). Greyscale from black (*i.e.*, 0 value, no lava in the pixel) to white (*i.e.*, 26 value, so 26 lava flows overlapped in that pixel). Figure taken from (Amato, 2022).

Work in progress Future researches will bring to the production of a quantitative risk map, using these methodologies, estimating the inundated areas and their probability, to an integration of different other satellite sensors, with greater spatial or temporal resolution, to improve the analysis, and to an applications of these generalized approaches to other volcanoes.

Chapter 3

Emulators

Numerical simulations of fluids are of great importance to study the dynamics of the physical systems and predict their possible evolution. Among all the Computational Fluid Dynamics (CFD) models, SPH is a consolidated Lagrangian mesh-free CFD method for the high fidelity in the numerical simulations of complex fluids. The Lagrangian property allows a good representation of the changing irregular shape of the flow, and an easy treatment of interfaces, including free surfaces. The mesh-free characteristic makes possible to manage large fluid deformations (*e.g.*, of lava while flowing), which would require frequent re-meshing (*i.e.*, the reconstruction of the mesh to follow the dynamics of the fluid), with consequent extra time and approximations of the volume. However, SPH simulations are computationally expensive and highly accurate simulations require long execution times and a high demand for computational resources. Simulations must be reliable and fast to be operationally useful.

Speed-ups of the simulations could be obtained in two ways, by simplifying the model, defining reduced problems and setting, or boosting the computational resources, improving the hardware and software support. Simplified models are typically adopted for real-world applications. In cases like this, the downside is the need to simulate complex physical dynamics with highly simplified models, losing in the reproducibility of real phenomena. Otherwise, wanting to follow the second way to speed up the models, with a computational resources boosting, the necessary resources are not always available, or not always it is possible to follow this way. A new point of view is then discussed, substituting the simplification of the models and the constantly upgrading of the hardware with the use of AI to reduce the amount of computations, maintaining the high-fidelity of the results.

Nowadays, there is a growing interest in the use of Artificial Intelligence (AI) (Winston, 1984) to push forward the frontiers of CFD, allowing new forms of analysis (Bortnik and Camporeale, 2021; Ummenhofer et al., 2019) and improving the performances of solvers (Kim et al., 2019). The speed-up of the simulations has, therefore, the final aim to enhance physics knowledge of natural phenomena. The use of highly accurate numerical methods to simulate a phenomenon poses the problem of slow processing, which makes the use of these methods complex for the study of highly complex phenomena in real-time. Hence, the need to combine models like these with AI, to improve their performance. On the other hand, the increasing widespread use of AI-CFD combined approaches makes necessary to rigorously validate the outputs of these combined models, with a deeper analysis of the reliability of the use of AI models in physics simulations.

Some solutions have been recently proposed, introducing the combined use of CFD models and AI to simulate fluids (Bortnik and Camporeale, 2021). Machine (ML) and Deep Learning (DL) models can be trained over CFD models, *e.g.*, SPH, to learn how to estimate the forces between particle pairs, replacing or joining the physical models. In this way, it is also possible to efficiently reproduce the dynamics of a physical system with far less computations.

Eulerian models already give very reliable and advanced results, *e.g.*, Deep Emulator Network SEarch (DENSE) model for weather predictions (Kasim et al., 2021) or MeshGraphNet for fluid dynamics simulations (Tobias Pfaff, 2021). The Lagrangian ones, especially for the lack of grids, are less consolidated, requiring to fill the absence of pre-established connections between nodes (Alexiadis, 2023). In addition, some of these approaches use simplified "vanilla" CFD models while others reach high-fidelity qualitative but not quantitative results, that are not the main scope. SPH technique is a Lagrangian good mesh-free method that gives high fidelity numerical simulations of complex fluids and, for this reason, can be used as reference for this kind of models (Zago et al., 2017; Zago et al., 2019).

A class of architecture largely used in fluid dynamics implementations are the Physics-Informed Neural Networks (PINN) (Raissi, Perdikaris, and Karniadakis, 2019; Cai et al., 2021). These NN are capable to efficiently learn from the data, exploiting physical-based formulations in the training or validation phase to validate the fitted models. In detail, the physical laws can be integrated in the networks architectures or in the loss functions, combining the error minimization with the physical constraints (Beucler et al., 2021). In this way, it is possible to solve forward and inverse fluid dynamics formulations through the pattern or scheme present in the data, also following the physical bases of the phenomena under analysis ("Physics-Informed"). In cases like this, so, the physical models are used with the AI in a way in which the models are seen as constraints of the neural networks (Li et al., 2023).

An alternative approach to reduce the computational cost of the simulations is the use of Reduced Order Models (ROM) (Lassila et al., 2014; Dar, Baiges, and Codina, 2023), that approximate large-scale systems with smaller ones, maintaining a high-fidelity in the reproduction of the physical phenomena. In the context of mesh-based models, for examples, this approach can be followed using a coarser grid; in contrast, for other kind of models, a larger time step or the use of *projections* can help to reduce the dimensionality of the problem. Generically speaking, the aim of the ROMs is to find a latent low-dimensional space to represent the original Full Order Model (FOM) dynamics, in an offline phase, and to solve the system in this reduced space, in an online phase. The first phase is computationally heavy but it is implemented once. The second phase can be repeated and it is lighter that the first one. Galerkin projections and Singular Value Decomposition (SVD) are examples of the projections (Dar, Baiges, and Codina, 2023). Therefore, in physics-based methods the idea is to follow the mathematical or physical meaning of the underlying model (looking at the governing equations). In contrast, in data-driven models, the input-output data relationship is exploited to construct an accurate ROM to represent the underlying system. An example of approach linked to this idea can be the Hypercomplex or Quaternion MultiLayer Perceptrons (HMLPs or QMLPs) (Arena et al., 1994), that are neural networks applied in the context of the Quaternion Algebra ($\mathbb{H}$), reducing the overall number of model parameters and the consequent computational load. Thus, a HMLP is a multilayer perceptron (MLP) in which input, output, weights and biases are quaternions, and the functions in the NNs are quaternion-valued sigmoidal functions. In addition, HMLPs with one hidden layer are considered universal interpolators in $\mathbb{H}$, for this reason they can be largely used for real word applications (Hornik, Stinchcombe, and White, 1989; Buscarino et al., 2023). Another model capable to reduce the scale of the system can be found in the Autoencoders, that are models capable to learn an efficient coding of unlabeled data (in the concept of unsupervised learning), based on a dimensional bottleneck between input and output (Kramer, 1992). They were firstly introduced as a nonlinear generalization of principal components analysis (PCA) (Kramer, 1991). This kind of models are composed by two blocks, an encoder and a decoder. The encoder compresses the input data, mapping them in a reduced space with a specific codification for this transformation. The decoder reconstructs the data from the encoded representation, learning and exploiting the precedent codification. They have multiple aims, dimensionality reduction, feature learning, and also data generation or anomaly detection. A famous example of autoencoder is the U-Net, composed by a contracting part and an expansive part, which gives it the U-shaped architecture (Ronneberger, Fischer, and Brox, 2015).

Artificial Intelligence can also be used to "emulate" part of the simulation process (Bortnik and Camporeale, 2021). In detail, the idea is to start from a totally physics-based model, based on CFD numerical schemes (*e.g.*, SPH method), where some blocks of the numerical approach are joined or substituted by AI models, that can learn from the CFD simulated data a particular dynamic and reproduce the behavior of the system. A typical ML model for combined AI-CFD Lagrangian models to emulate the physical ones is the one based on the Artificial Neural Networks (ANN), algorithms that mimic the human brain structure and the way to analyze data to extract information. Models obtained with such approach are then called *emulators*, *e.g.*, DENSE for weather predictions (Kasim et al., 2021). In particular, an emulator can take in input the state of the cells or particles obtained from the CFD model and can learn how to estimate simulation quantities, as forces or densities, speeding-up the simulations while maintaining the same physical descriptive accuracy of the phenomenon.

A NN approach for Lagrangian cases is based on the Graph NN (GNN), models to process data using graphs to compensate the lack of a spatial structure in the dataset. They are largely used in fluid dynamics, *e.g.*, the Lagrangian Fluid Graph Networks (FGN) (Li and Farimani, 2022), in which particles are the nodes and the interactions between them are the edges. Similarly, it is possible to define an Encoder-Processor-Decoder approach, based of GNN, to learn an operator, *i.e.*, the Graph Network-based Simulator (GNS), to simulate the dynamic of a physical system (Sanchez-Gonzalez et al., 2020). A downside of the GNN is the need for re-meshing after large deformations, that is a computationally expensive job.

Another group of AI-CFD Lagrangian models uses Convolutional Neural Networks (CNN). These take into account the spatial distribution of the Lagrangian nodes by performing spatial convolutions over restricted neighborhoods, with (convolutional) kernel functions. CNN are basically designed to work with structured data divided in grids, as the nodes of an Eulerian numerical method or the image pixels. Therefore, their application to Lagrangian mesh-free methods is not painless, and needs specific expedients. One is to replace the process of iterating over the neighboring particles, typical in SPH simulations, with the convolutions of the network. Using the convolutions, these models exploit the global position of the particles, with a strong dependence on the spatial features of the data used for the training. This aspect influences the quality of the results and the generalization abilities of the model.

Finally, another AI-CFD model is the one that acts as a local emulator, that is, at the level of the particles, or over particle-pairs interactions. In this case, the model learns how to reproduce part of the physical law of the system, substituting part of the numerical approach with some AI model. To consider the spatial distribution of the particles, the model exploits the nearby particles at each iteration, as in the classical SPH technique. Therefore, the network plays the same role of the reference equations and the spatial distribution of the particles is handled with iterations over neighbors. Different AI architectures are already be used to locally emulate the interactions of the physical-mathematical models between particles.

In (Ladickỳ et al., 2015), a Random Forest (RF) model is employed to estimate (extrapolate) the particle state (position, velocity) from the one at the precedent time step, speeding-up the simulations. In detail, this RF takes as input a feature vector that represents the setting of the system, composed by pressure, viscosity, and surface tension terms, and incompressibility constrains. The aim is to speed-up SPH simulations, so the model learns and reproduces the behavior of each particle over time steps longer than those utilized in simulation phase. The output is the updated particles state relative to the time step presented at training, with a good speed-up of the method with real-time performances. In this work, three different integration schemes are explored, reaching a good level of qualitative results. The RF architecture is known for the inference speed, however, one of the main problem is the common limit of these models for regression tasks, *i.e.*, it is not capable to extrapolate the model far outside the data observed during training, that is linked to the consistency and generalization ability of the model (Graw, Wood, and Phrampus, 2021). To overcome this problem, the use of ANN is advised. In (Alexiadis, 2023), an ANN is used to reproduce the particle-particle interactions. This takes in input the state of the particle (composed by relative position, relative velocity, and the density of two particles, made for each couple of particles) and it is trained to reproduce the behavior of the fluid. In this case, the network works on particle pairs, therefore the global spatial distribution of the particles in the training dataset is not fully considered, as it is for CNN. The emulator reproduces the reference equations. In (Alexiadis, 2023) work, three different CFD approaches are explored to obtain the reference data (SPH, Molecular Dynamics, and Discrete Element Method). All the three approaches reach a good fidelity of the simulations and a good generalization ability, but they are based on a simplified version of the CFD models, with little physical meaning (the so called "vanilla" versions), that limits the reproducibility of complex fluids. Moreover, a model inversion application is shown, training the ANN to estimate particle interaction forces, while only knowing total forces per particle. However, the main aim of this work is not the speed-ups of the simulations.

I decided to work on the use of an emulator capable to reproduce the behavior of different kind of fluids, faithfully reproducing the physics of the natural phenomenon under analysis. In the first instance, the aims of my work are twofold, to have a physically correct and complete model, aimed at faithfully reproducing physical phenomena, and to speed-up the simulations, to obtain them in a short time. By unifying the two purposes, therefore, the ultimate goal of this study is to improve knowledge of the physics behind natural phenomena, combining physical-mathematical models with artificial intelligence. This approach is based of the use of CFD techniques, in particular the SPH model, where the reference model is in line with SPH formulations adopted in practical applications, coupled with a specific AI architecture, *i.e.*, the ANN, developed for physically consistent and fast simulations, to obtain an emulator that works at the level of the particles (the local one). This approach emphasizes the design aspects that have a strong impact on the physical fidelity to the reference CFD model. For the *universal approximators theorem* (Hornik, Stinchcombe, and White, 1989), it is demonstrable that the standard multilayer feedforward networks, with at least a sufficiently wide hidden layer of neurons, can approximate any kind of functions with any accuracy. For this and the precedent observations, thus I decided to use this architecture. The idea is to demonstrate that whit this approach is possible to remain as faithful as possible to the reference physical model, so to have completeness and correctness of the physics, and to reduce the simulation times, in order to have a faster fluid simulation respect to the classical numerical techniques.

In the following sections, the proposed idea and the specific technical details are given, delving into the models used, the innovations introduced from a conceptual and practical point of view and the frameworks developed.

3.1 Fluid simulations with SPH and AI

An AI-based Computational Fluid Dynamics (AI-CFD) emulator is a model that combines the use of CFD approaches with the Artificial Intelligence techniques to simulate fluids, improving the performances of the numerical models and their power of reproduction, and therefore in the study, of physical phenomena. In the CFD techniques, as SPH, part of the numerical code is combined with an AI algorithm, that can join an equation-based CFD model or replace parts of it, emulating the relative behavior. The AI model is trained over the CFD data to learn a way to reproduce the fluid dynamics of the system. Thus, emulators mimic CFD models with explicit mathematical laws, which represent the emulated entity. This is in contrast to generic data-driven approaches, where the data can be experimental and the physical laws unknown (Torrisi et al., 2022b; Amato et al., 2023b). For emulators, regression applications of AI are generally used, as the aim is to generate continuous quantities to simulate a physical phenomenon.

3.1.1 AI-CFD model

The general pipeline that I have structured for this approach is the following one. Firstly, an equation-based SPH simulation is conducted, using the analytical model that discretizes the Navier-Stokes equations. From this simulation, a set of reference simulated data are sampled, to feed to the ANN model. The AI-based SPH emulator is, then, trained over the sampled data, learning how to reproduce the fluid behavior. Finally, the trained AI-CFD model is used to emulate the SPH fluid system, reproducing the dynamics of the physical phenomenon. The following Fig. 3.1 is for illustrative purposes, only to highlight these steps.

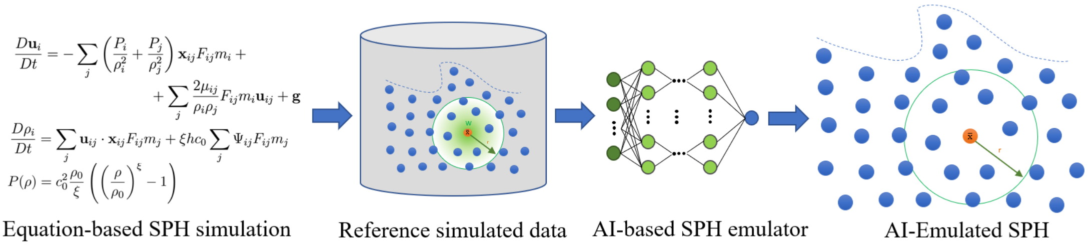

$\frac{D\mathbf{u}_i}{Dt} = -\sum_j \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2}\right)\mathbf{x}_{ij}F_{ij}m_i + $
$+ \sum_j \frac{2\mu_{ij}}{\rho_i\rho_j}F_{ij}m_i\mathbf{u}_{ij} + \mathbf{g}$
$\frac{D\rho_i}{Dt} = \sum_j \mathbf{u}_{ij}\cdot\mathbf{x}_{ij}F_{ij}m_j + \xi h c_0 \sum_j \Psi_{ij}F_{ij}m_j$
$P(\rho) = c_0^2\frac{\rho_0}{\xi}\left(\left(\frac{\rho}{\rho_0}\right)^{\xi} - 1\right)$
Equation-based SPH simulation
Reference simulated data
AI-based SPH emulator
AI-Emulated SPH

FIGURE 3.1: Pipeline of the proposed approach

More in detail, in this study I followed a 2D SPH formulation applied to complex fluids, and I substituted the conservation of momentum equation with an Artificial Neural Network (ANN, multilayer perceptron), training the latter to reproduce individual particle-particle interactions (Zago et al., 2023b). This emulator has a dual aim, to reach the completeness and correctness of the physics and to reduce the simulation times. To obtain this, the ANN is trained using a feature vector that which can well represent the interactions between particles, therefore composed of the relative position and velocity between them and their densities. The output of the ANN is the estimated particles forces. Incidentally, in the SPH context, so often "force" replaces the term "acceleration", using them indistinctly, as they are related by the particle density, which is a known quantity.

Briefly, in the equation-based simulation, for each simulation step, the state of the particles is considered, building the particle neighbors list for each particle and obtaining the feature input vector, from which the force is calculated by the equation-based SPH formulations and integrated to have the new positions. Fig. 3.2 shows a scheme of the SPH process. In the emulator, the derivative calculation of the momentum equation is substituted by the ANN. The simulations advance over time with step Δt, named the *time step*, repeating each computation to derive and integrate. Clearly, Δt directly affects the duration of a simulation. Computing derivatives is typically the slowest part of the simulation, as it involves a large amount of computations. A smaller Δt (that improves the simulations) produces more simulation steps, thus longer simulation times. Therefore, the combination of CFD approaches and AI (in the emulator) could be a way to reproduce the fluid behavior reducing the computational load of the simulation.

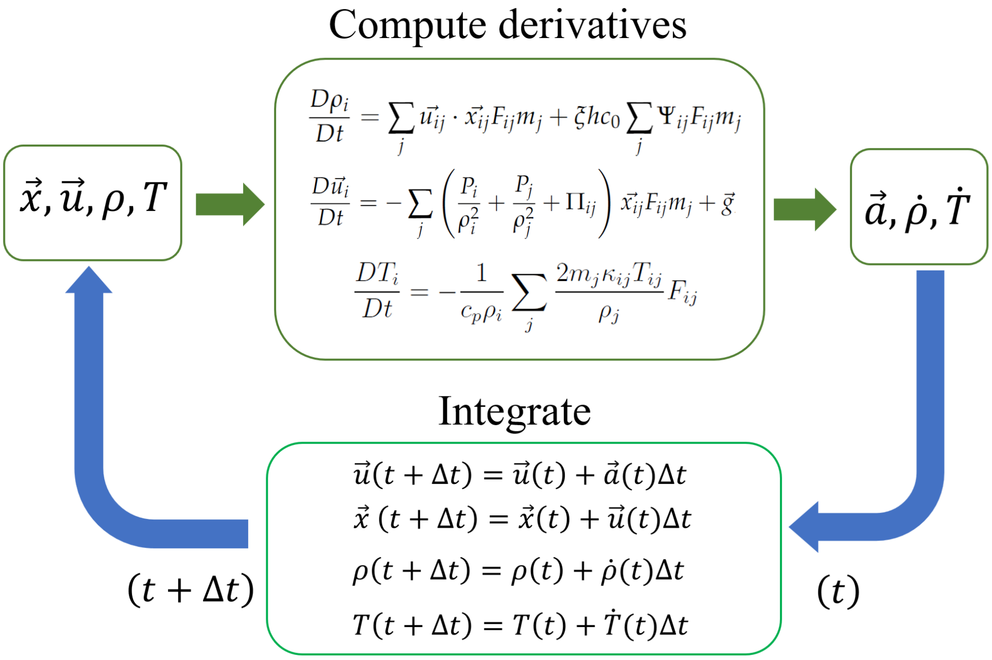

Compute derivatives
$\vec{x}, \vec{u}, \rho, T$
$\frac{D\rho_i}{Dt} = \sum_j \vec{u_{ij}} \cdot \vec{x_{ij}} F_{ij} m_j + \xi h c_0 \sum_j \Psi_{ij} F_{ij} m_j$
$\frac{D\vec{u}_i}{Dt} = -\sum_j \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2} + \Pi_{ij} \right) \vec{x_{ij}} F_{ij} m_j + \vec{g}$
$\frac{DT_i}{Dt} = -\frac{1}{c_p \rho_i} \sum_j \frac{2 m_j \kappa_{ij} T_{ij}}{\rho_j} F_{ij}$
$\vec{a}, \dot{\rho}, \dot{T}$
Integrate
$\vec{u}(t+\Delta t) = \vec{u}(t) + \vec{a}(t)\Delta t$
$\vec{x}\,(t+\Delta t) = \vec{x}(t) + \vec{u}(t)\Delta t$
$\rho(t+\Delta t) = \rho(t) + \dot{\rho}(t)\Delta t$
$T(t+\Delta t) = T(t) + \dot{T}(t)\Delta t$
$(t+\Delta t)$
(t)

FIGURE 3.2: SPH process scheme

ANN architecture

In the developed emulator, I choose an Artificial Neural Network (ANN, multilayer perceptron) as AI model.

In cases like the one presented here, the relationship between inputs and output data is known in the specific analytical equations that govern the fluids (Eq. 1.19, which recalls Eqs. 1.11 and 1.20). Looking at the equations, it is clear that the input to the network must include particle positions, velocities and densities. In general, the first two quantities are $2D$ vectors, while input variables of the ANN should be scalars. Therefore, I derived the scalar quantities from the variables of interest, following also how these variables figure in the reference equations (Amato, Zago, and Del Negro, 2024). Then, I used three input variables, *i.e.*, the relative position between two particles (x_{ij}), the normal component of the relative velocity ($u_{n_{ij}}$), and the product of the two densities ($\rho_i \cdot \rho_j$). Indeed, in the case of an inviscid fluid (Eq. 1.19), the pressure component of the Navier-Stokes equation is proportional to the relative position and the artificial viscosity is proportional to the normal component of the relative velocity. The first two quantities have the same shape of (Alexiadis, 2023); in contrast, the third one substitutes the individual densities used in that case, following the way in which the densities appear in the reference equations. In this way, the computational load is reduced, due to the less number of input features, and their maintain the symmetry of the particles interactions, the independence of the model from the local configuration of the particles, and the high fidelity to the original SPH formulation, allowing to exactly use some classical SPH features, *e.g.*, the boundary models. The output estimated vectors are by construction oriented as the vector of relative position between the interacting particles, which is known. Therefore, the network will produce in output the module of the acceleration vector, subsequently multiplied by the unit vector of the relative position. The structure of the NN has been empirically defined, with the use of three hidden layers with, respectively, input-output dimensions of $9-27$, $27-27$, $27-9$ neurons, wide and deep enough to handle the high complexity of the physics represented in the dataset. The activation function used is the Exponential Linear Unit (ELU) function (Clevert, Unterthiner, and Hochreiter, 2015).

The emulator presented here has two main advantages respect to the examples already presented in the state of the art. Firstly, the use of the ANN, considered as an universal approximator with the capability to extend the learned behavior beyond the interval presented during training, allows to increase the generalization ability; secondly, the choice of the features in a spatial independent way, allows the model to learn the values of particle-particle interaction without any dependence on local particles configurations, considering only the specific values and not directly modeling the spatial behavior of the fluid. In this way, the emulator learns to reproduce the numerical relationships between the variables in the reference model, predicting interactions of particles without a predominant direction, namely there are *no anisotropy effects* in the model training. Therefore, when the emulator is used to generalize, as long as the values of the variables are in ranges where they hold physics similar to that learned from the reference simulation, the emulator should be able to produce physically meaningful results, without the need to use a large number of training samples to satisfy all the possible geometric configurations that may arise.

I used the predictor-corrector integration scheme (see subsection 1.2.4), and I have noticed that training the network with input SPH quantities sampled during the predictor phase of the integration (see subsection 1.2.4), and using the SPH reference outputs sampled during the corrector phase of the integration (see subsection 1.2.4), improves the emulator stability. This could be explained thinking about the similarity of this approach to an implicit integration scheme. Although this procedure introduces a deviation from the reference integration scheme, it is possible that this deviation is negligible with respect to the achieved benefit in terms of stability (Amato, Zago, and Del Negro, 2024).

Training phase

The ANN learns how to approximate the relationship between input features and output presented by the training dataset. It is important that this dataset is representative of the type of physics that the emulator will handle. The dataset for the training is thus generated by simulating an inviscid dam break. This choice is due to two factors. Firstly, it is a classic benchmark case for SPH, widely used in simulations for methodological and application purposes, therefore it represents an easy-to-use case study to compare SPH-based results. In addition, the dam break presents a wide range of flow conditions, evolving from the initially violent flow of the transient, toward a static steady state; therefore, this training dataset is representative for the different flow conditions. The reference simulation is discretized with around 3420 particles, evolving for 50.0 s, sampled at 0.5 s, totally sampling around 3.42 million of data points (in case of spatial step $\Delta p = 0.75$ m).

For the entire simulation, all the particles states and particle-particle interaction forces are sampled and collected in different arrays. The particles states array for the entire simulation is fed to the network as input, the particle-particle interaction forces are used as reference output.

The network is trained in order to minimize a total loss function, composed by the sum over all the neighbor lists for all the particles and time steps, namely of all the single L_i losses (see Eq. 3.1, that is, the loss for a single particle and neighbor list for a single time step) (Alexiadis, 2023). This loss function was chosen to adjust the estimate of the particle-particle interaction forces so that their summation for a particle equals the reference (target) total force on the particle.

$$L_i = \sum_i \left\| F_i^{target} - \sum_{j \neq i}^{N_{l,i}} f_{ij}^{ANN} \frac{\vec{x}_{ij}}{\|\vec{x}_{ij}\|} \right\|^2, \tag{3.1}$$

with F_i^{target} the reference total force simulated with the SPH equation-based formulation for the i-th particle, and f_{ij}^{ANN} the single particle-particle force estimated by the ANN.

Look-up table As in (Alexiadis, 2023), to reduce the execution time of the emulator during runtime, the technique to map the ANN outputs over a look-up table has been adopted (Groeneveld et al., 2007). The table is constructed only once at the end of the training phase by mapping the input-output relationship of the obtained model across a predefined set of values, and storing the corresponding outputs. During the emulation phase, the values of the ANN output are reconstructed directly from this table, with a linear interpolation. Although this process introduces some errors in the estimation of the output, it can reduce the execution time of the emulated part, and keep it independent from the complexity of the ANN (Amato, Zago, and Del Negro, 2024).

Boundary Conditions

For the simulation, I have adopted the Dynamic boundary model (Crespo, Gómez-Gesteira, Dalrymple, et al., 2007) (see Appendix A), in which density is treated equally to fluid particles, and velocity is set to zero. Some parallel layers of extra boundary particles are placed along analytical boundaries, spaced at intervals of Δp, obtaining an analytical boundary. The number of extra particle layers should be enough to complete the support of the smoothing kernel when it is centered on the out-most fluid particle. With the Dynamic boundary model, the no-slip conditions are obtained.

Chapter 4

Results and Discussion

Various fluids have been studied, relating to different theoretical and application contexts, with related results and discussions.

The goal is to demonstrate the applicability of an approach like this, validating the physical goodness of the model and the ability to generalize to cases not seen during the study, and showing the speed of the simulations, greater than traditional methods.

4.1 Validation and Generalization

To train and validate the model and to test the generalization capability, I firstly used a classical SPH benchmark as test case, that is the inviscid or the viscous dam break and compared results obtained with the equation-based SPH model (hereafter simulations) and the AI-based SPH emulator (hereafter emulations). The dam break flow was chosen because it is an easy-to-use case study to compare results, and as it presents a wide range of flow conditions, that make it representative to train the network and emulate very different scenarios.

4.1.1 Validation: inviscid dam break

The first test case taken into consideration is the inviscid dam break (Amato, Zago, and Del Negro, 2024), with artificial viscosity coefficient $\alpha = 0.01$. The physical parameters used are density $\rho = 1.00\ \mathrm{kg/m^3}$, polytropic constant $\gamma = 1$, and artificial speed of sound $c_0 = 560.29\ \mathrm{m/s}$, that is 20 times higher than the hydrostatic velocity, $c_0 = 20\sqrt{2\|\vec{g}\|h}$, with h the maximum height of the fluid (Torricelli, 1644). The fluid is set in a box which is 90 m long and 80 m high, and the fluid has an initial shape of width 30 m and height 40 m. The particles are discretized by using a spatial step of $\Delta p = 0.75$ m (which will be referred to as Medium Resolution, MR), with 3423 particles. To verify the generalization capability of the approach, two other spatial steps are also used, $\Delta p = 0.5$ m (High Resolution, HR) and $\Delta p = 1.0$ m (Low Resolution, LR), resulting in 6681 and 2151 particles, respectively. The initial density is set by using an inversion of the state equation (Equation 1.11) and considering a hydrostatic pressure profile. The fifth order Wendland smoothing kernel was used (Wendland, 1995) (see Appendix A).

In order to incorporate a high quantity of training information for the flow reproducibility, 50.0 s of fluid evolution are simulated, sampling the whole particle set, composed by fluid and boundary particles, every 0.5 s. Of these, an 80% is used for the training phase, and the remaining part for the validation one. It is possible to notice that, including boundary particles within the dataset, the quality of the results are improved in terms of particle disorder. This is likely due to a better representativeness of surface particles.

The model was trained over 10000 epochs, with a learning rate α in the order of 10^{-4} (Goodfellow, Bengio, and Courville, 2016). The Adam optimizer and the Mean Squared Error (MSE) loss function (Eq. 3.1) are used, with a final training and validation loss in the order of 10^{-3}.

The momentum equation is emulated, *i.e.*, substituted with the ANN, emulating thus also the computation of the kernel and the equation of state. The learned parameters (as the Δp, the smoothing length, or the speed of sound) are fixed for the emulator and a new model needs to be created when they need to be changed, retraining the network.

In Fig. 4.1 a simulation (equation-based SPH reproduction of the dynamics of the fluid) and an emulation (AI-CFD emulator reproduction of the dynamics of the fluid) of an inviscid fluid with a medium resolution is shown. The fluid is posed in a box with a regular topography and the frame shown is at time $t = 3.5$ s. The particles are colored by velocity magnitude [m/s]. A good agreement is found between the simulation and the emulation, within the resolution of the method (that is of the order of the dimension of the particle, *i.e.*, the Δp).

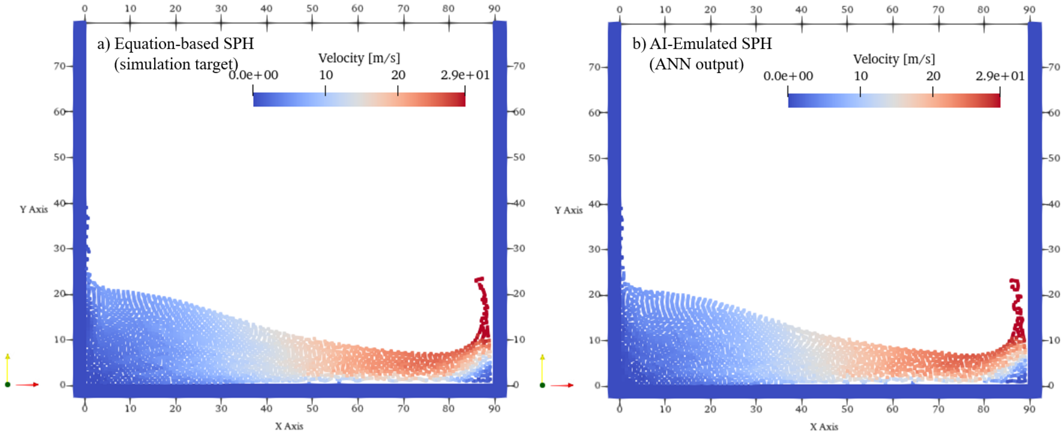


FIGURE 4.1: Simulation (a) and emulation (b) of an inviscid fluid with medium resolution (MR, $\Delta p = 0.75$ m) in a box with regular topography at time $t = 3.5$ s. The particles are colored by velocity magnitude [m/s].

In Fig. 4.2 other frames of the simulation and emulation are compared, noting that all the major flow features evolving over time in the original simulation (first row) are present also in the emulation (second row). Some expected discrepancies are shown in the different content of flow details, that is, the major features are present, some minor detail, especially in the case of chaotic elements as droplets or splashes, get lost.

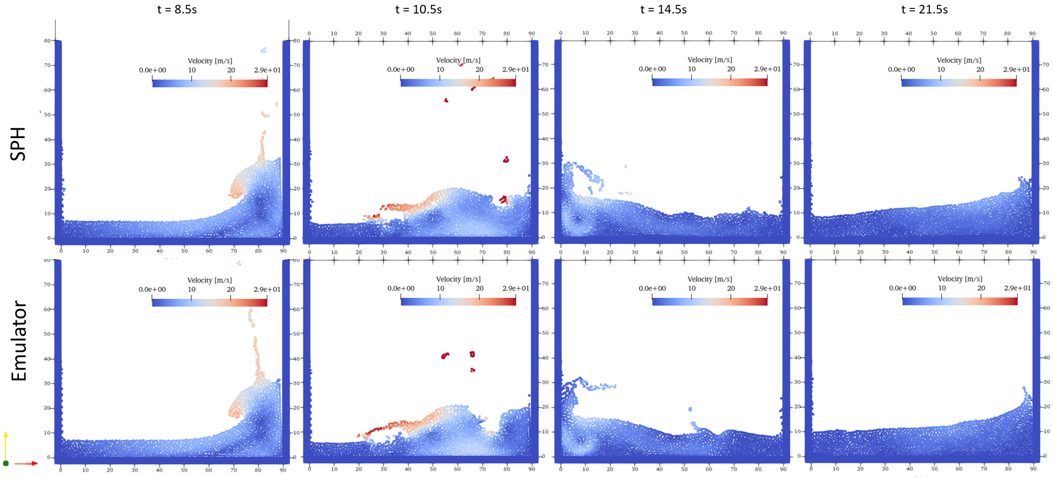


FIGURE 4.2: Simulation (first row) and emulation (second row) of an inviscid fluid with medium resolution (MR, $\Delta p = 0.75$ m) in a box with regular topography at time $t = 8.5$ s, $t = 10.5$ s, $t = 14.5$ s, and $t = 21.5$ s. The particles are colored by velocity magnitude [m/s]. Figure taken from (Amato, Zago, and Del Negro, 2024).

To quantify the goodness of the model, I compared the dam break fronts in time in the SPH simulation and in the AI-based emulation, measuring the position of the rightmost particle of fluid over time (Fig. 4.3, with SPH reference in blue solid line, and emulator one in orange dashed line). In addition, the relative local particles configuration was shown (Fig. 4.4, with SPH simulation particles in blue, and AI-based emulation particles in orange). The two Figures show a good match between the fronts, compatible with the scale of the particle size, and a very strong agreement of the overall profile of the flow.

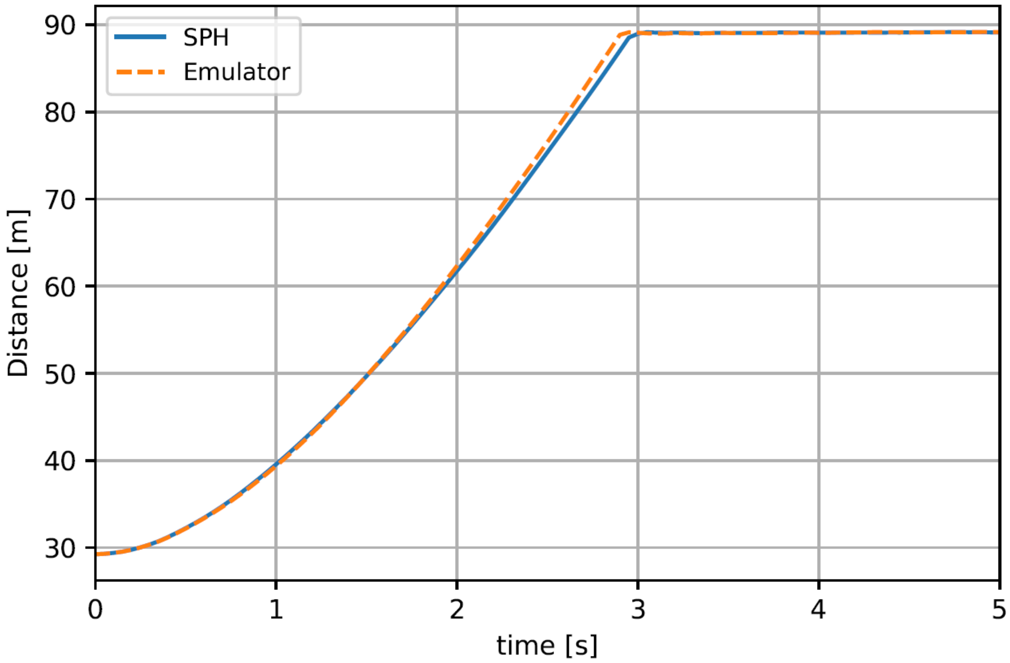


FIGURE 4.3: Inviscid MR dam break fronts comparison over time. SPH model is shown with blue solid line, emulator with orange dashed line. Figure taken from (Amato, Zago, and Del Negro, 2024).

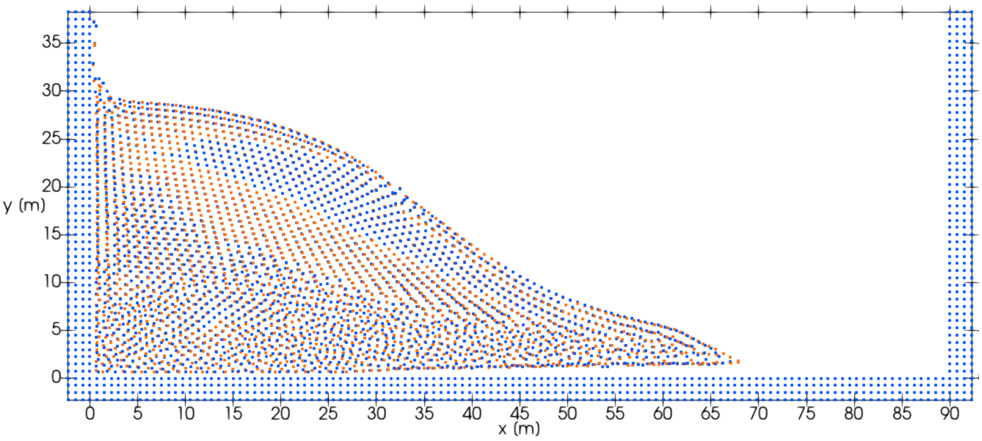


FIGURE 4.4: Inviscid MR dam break relative particles configuration at time $t = 2.2$ s. SPH simulation is shown with blue particles, AI-based emulation with orange particles. Figure taken from (Amato, Zago, and Del Negro, 2024).

4.1.2 Generalization

When considering AI models, it is important to analyze their robustness to changes in some parameters and their ability to generalize reproducing problems which they have not been trained on.

In the first place, a preliminary study has been conducted, shown in Fig. 4.5. Starting from the top left block and moving clockwise, I firstly simulated a dam break in a larger regular topography (box dimension 160 m × 80 m, fluid dimension 60 m × 30 m), with the equation-based SPH formulation (Fig. 4.5 a). The data were sampled and given in input to the ANN, to train the model and reproduce the same problem (Fig. 4.5 b). The trained model has then been used to emulate the same fluid problem in a different topography, with the presence of bumps, obtaining very good results (Fig. 4.5 c). Finally, to verify the ability of the model to be used in problems different respect to the one used in the training phase, I simulated with classical SPH formulation the same setting with bumps, obtaining a good agreement with the emulated case (Fig. 4.5 d).

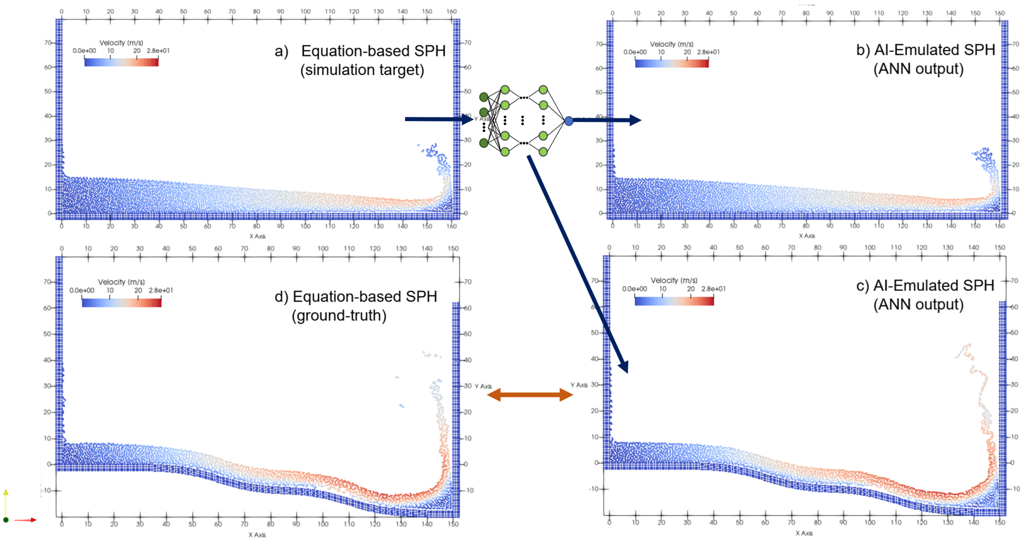


FIGURE 4.5: Inviscid MR dam break generalization capability. (a) Equation-based SPH simulation of the inviscid dam break in a larger regular topography, (b) AI-emulation of the dam break in regular topography, using the model trained with the same problem, (c) AI-emulation of a different problem, *i.e.*, the inviscid dam break in an irregular topography with bumps, (d) equation-based SPH simulation of the same problem, *i.e.*, the inviscid dam break in an irregular topography with bumps, to compare the results. The particles are colored by velocity magnitude [m/s].

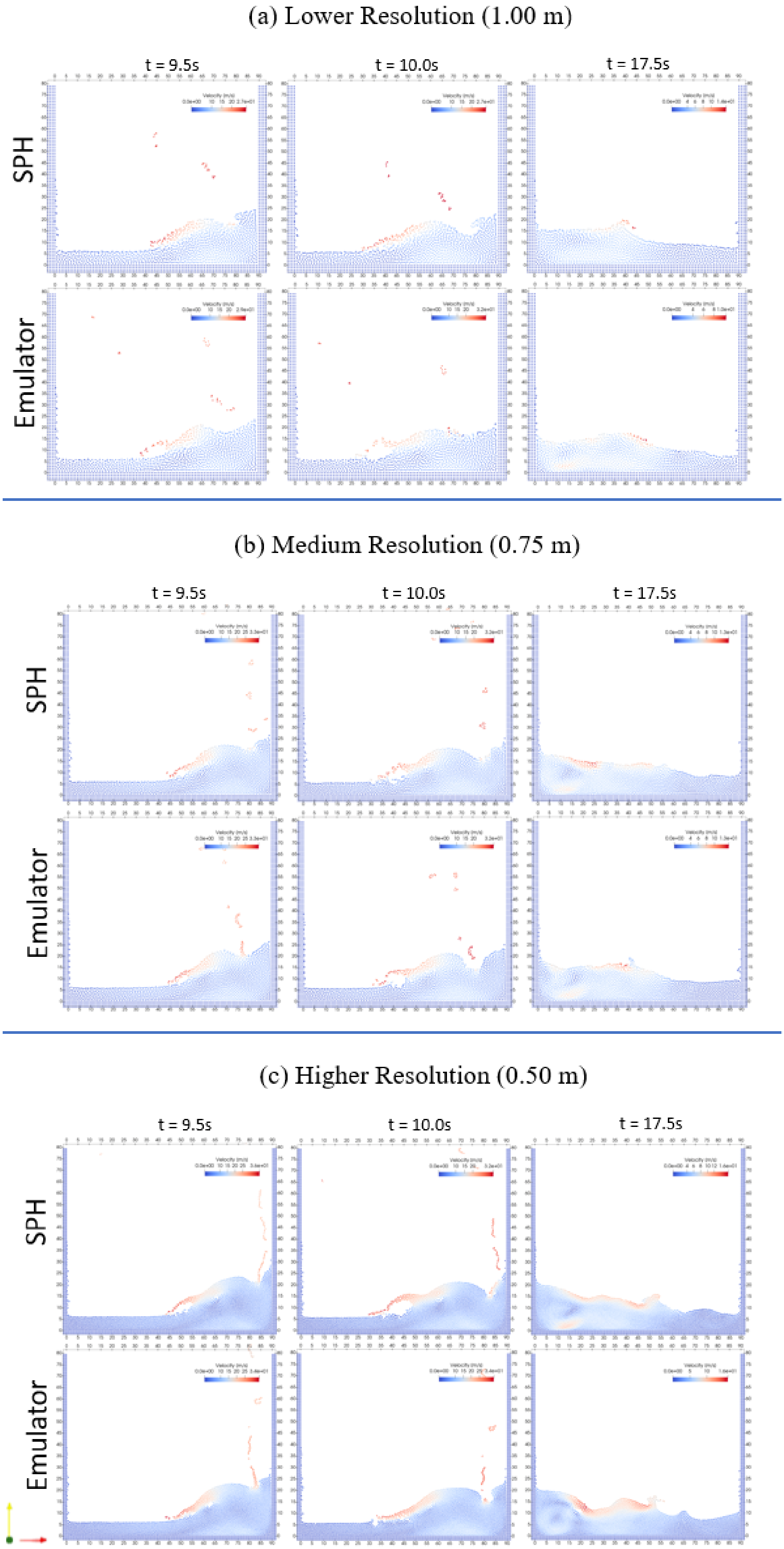
After this test, to strengthen the analysis, I tested the ability of the model to generalize and resolve the formation of flow features, considering different spatial resolutions and different problem settings.

Testing for different spatial resolution: lower and higher resolution

To test the robustness respect to the spatial resolution changes, I repeated the same framework also with different spatial resolutions, repeating the entire process (*i.e.*, equation-based SPH simulation, training, and AI-based SPH emulation) for each resolution. This kind of resolution influences the fidelity of the emulator, because the flow features can become comparable to the particle size, and therefore to the intrinsic errors (numerical and/or approximation). More in detail, in case of lower resolutions, the particle size becomes more important within the domain because it is comparable to the scale of flow features, making them particularly subject to errors; in case of higher resolution, more sensitive flow details are generated, enforcing the chaotic nature of the flow, which becomes thus particularly sensitive to errors.

Following this analysis, simulation and emulation outputs are in accordance both in the two cases, with the formation of the major flow features, like the initial advancement of the fluid front, and the interaction with the walls (formation of a vortex). Some discrepancies between the simulated and emulated results, that have been founded, are always compatible with the sensitivity of the model (Δp).

Fig. 4.6 shows the comparison between Lower Resolution (LR), with $\Delta p = 1.0$ m (Fig. 4.6 a), Medium Resolution (MR), with $\Delta p = 0.75$ m (Fig. 4.6 b), and Higher Resolution (HR), with $\Delta p = 0.5$ m (Fig. 4.6 c), for three significant times, $t = 9.5$ s, $t = 10.0$ s and $t = 17.5$ s. Some features can be found in the comparison between LR, MR and HR. Even if minor flow features are not always replicated changing the resolution, especially with its reduction, major features, as the impinging jet formation after the dam break reaches the wall, are in general maintained, and each difference in the flow is dissipated moving towards the final steps (*e.g.*, $t = 17.5$ s), when also the chaotic features disappear.


(a) Lower Resolution (1.00 m)

(b) Medium Resolution (0.75 m)

(c) Higher Resolution (0.50 m)

FIGURE 4.6: Time series of an inviscid dam break flow, comparison LR (a) vs MR (b) vs (c) HR. The particles are colored by velocity magnitude [m/s]. Figure taken from (Amato, Zago, and Del Negro, 2024).

Generalizing for different flow complexity: still fluid and flow with waterfalls

To test the generalization capability of the model also for different problem settings, I used the model trained over the inviscid dam break in a rectangular box for new problems, with the same physics of the original one, but with different levels of complexity. In particular, two new geometries have been set, with lower and higher complexity respect to the reference training set. The first one, with a lower complexity, is composed by a still volume of fluid in a greater tank. This test is necessary because the model has been trained over fluid in motion, but must also be able to deal with fluid at rest. The fluid in this case is with different dimensions, it is a 90 m × 15 m domain of fluid (Fig. 4.7), with the same physical and simulation properties of the training fluid, and I used the emulator previously trained over the dam break to reproduce 200.0 s of evolution. In addition, I also simulated the same problem with the original equation-based SPH model, comparing the results. Even if individual particles assume a different configuration, the overall behavior of the fluid is consistent with the expectations (it remains stationary) (4.8).

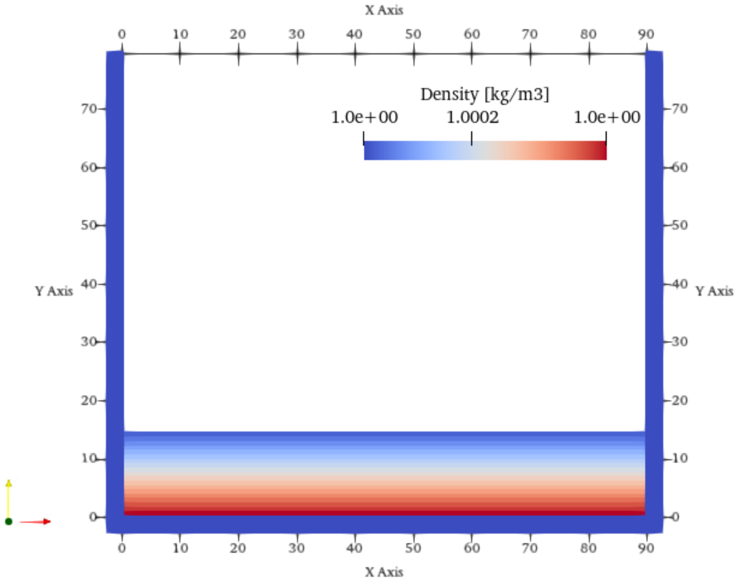


FIGURE 4.7: Initial inviscid still fluid simulation setup with particles colored by density [kg/m^3], with a minimum value around 0.99997 [kg/m^3] and a maximum value around 1.00049 [kg/m^3], not visible in the legend.

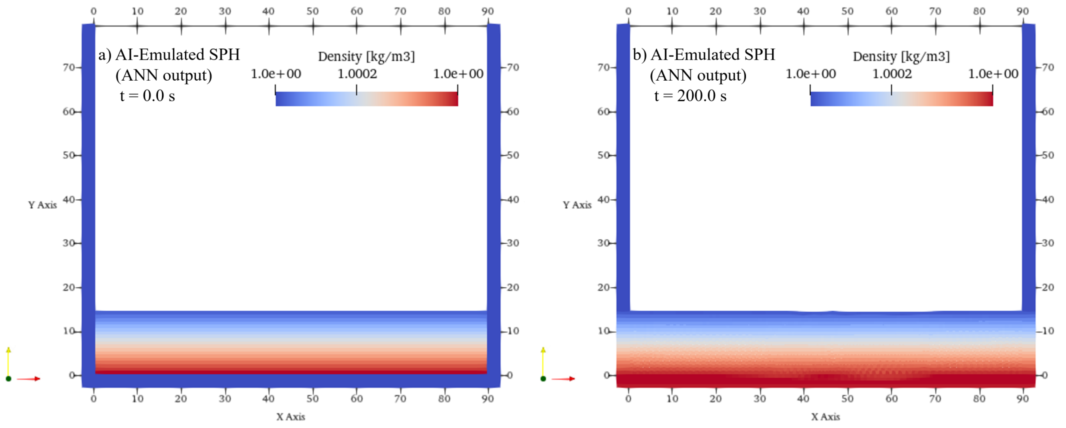


FIGURE 4.8: Emulation of an inviscid still fluid at time $t = 0.0$ s (a), and $t = 200.0$ s (b). Particles are colored by density [kg/m^3].

The second one, more complex than the original problem, is composed by a dam break in an elevated shelf, therefore forced to produce waterfalls during the initial phase, starting the motion from an higher elevation. In this case, the box has a 150 m × 80 m dimension, and the fluid has the same dimension of the original dam break used for the training (Fig. 4.9). 50.0 s of evolution have been simulated. Also in this case, the simulation and emulation are in agreement (Fig. 4.10), and the sequence of major flow dynamics is preserved, despite some different details. The Fig. 4.10 shows this simulation and emulation of the inviscid dam break with waterfalls at medium resolution. The droplets are strongly chaotic phenomena, which therefore can not coincide between simulation and emulation; conversely, the global behavior is well reproduced by the emulator.

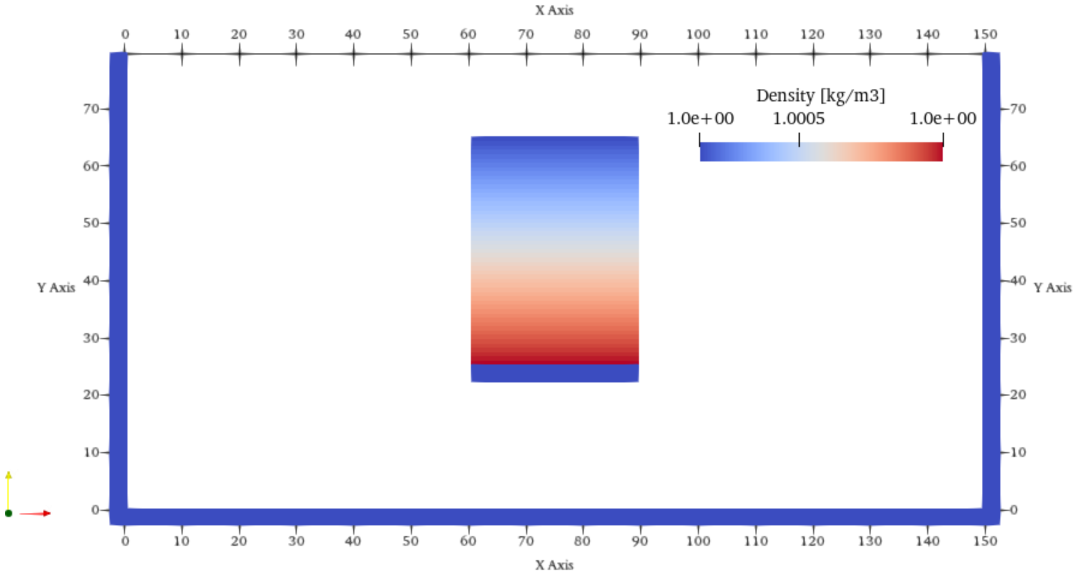


FIGURE 4.9: Initial inviscid dam break with waterfalls simulation setup with particles colored by density [kg/m^3], with a minimum value around 0.993 [kg/m^3] and a maximum value around 1.042 [kg/m^3], not visible in the legend.

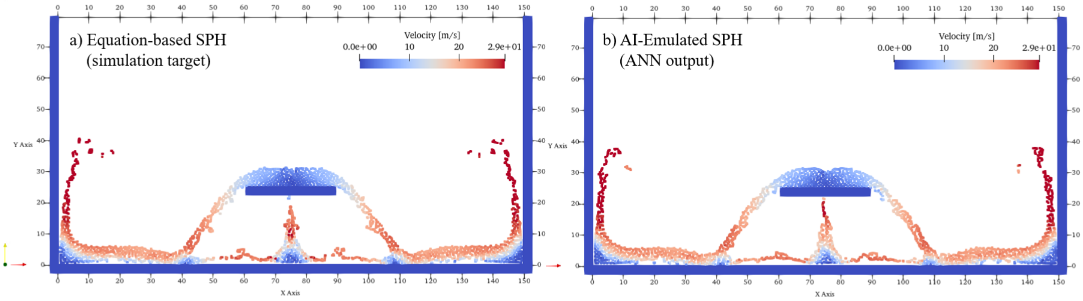


FIGURE 4.10: Simulation (a) and Emulation (b) of an inviscid fluid with medium resolution ($\Delta p = 0.75$ m) in a box with regular topography and an elevated shell, producing waterfalls. Simulation at time $t = 4.6$ s. The particles are colored by velocity magnitude [m/s]. Figure taken from (Amato, Zago, and Del Negro, 2024).

In addition, I have combined the two tests, repeating the generalization process at different resolutions. In this way, it is possible to prove the robustness to changes in the spatial resolution for the generalization still fluids and waterfalls problems. The different problems maintain the good agreement between simulation and emulation. One limitation encountered when increasing flow complexity is the presence of some instabilities at the impact of jets or drops, occurring when also the spatial resolution is increased. However, these limitations are inherently conservative, because they primarily influence the numerical execution of the simulation and do not lead to physically unrealistic results.

Therefore, the model is capable to generalize over problems never seen during the training phase. This is realistically due to its independence from local spatial features.The peculiarity of using an emulator is that it reproduces particle-particle interactions, rather than directly modeling the spatial behavior of the fluid, thus the numerical relationships between the variables in the reference model. Therefore, as long as the values of the variables are in ranges where it holds the same physics learned from the reference simulation, the emulator should be able to generalize, producing physically meaningful results.

Reproducibility of Boundary Conditions

Boundary Conditions is one of the main SPH open problems, to make SPH a mature method (see the *SPH "Grand Challenges"* here (Vacondio et al., 2021)). Therefore, I proved the capacity of the emulator to reproduce the boundary conditions as in the reference SPH model (Amato, 2023). Using as reference the a simulated dam break, it is possible to notice that accurate boundaries allow to produce the same features during simulation and emulation. Fig. 4.11 shows simulation and emulation of an inviscid classic dam break fluid in a box with regular topography at time $t = 2.0$ s, with the reproduction of significant features also by the emulator in a laminar structure. In addition, when some non-linearity (as the onset of a vortex at time $t = 16.0$ s) is triggered in the fluid by the presence of the box walls, also in this case the matching of simulation and emulation persists, as Fig. 4.12 shows. Thus, the matching of simulation and emulation persists up to the production of consistent non-linear features, in a swirling structure. Therefore, I demonstrated the ability of the emulator to reproduce the SPH boundary conditions, catching also the non-linearity of the fluid behavior.

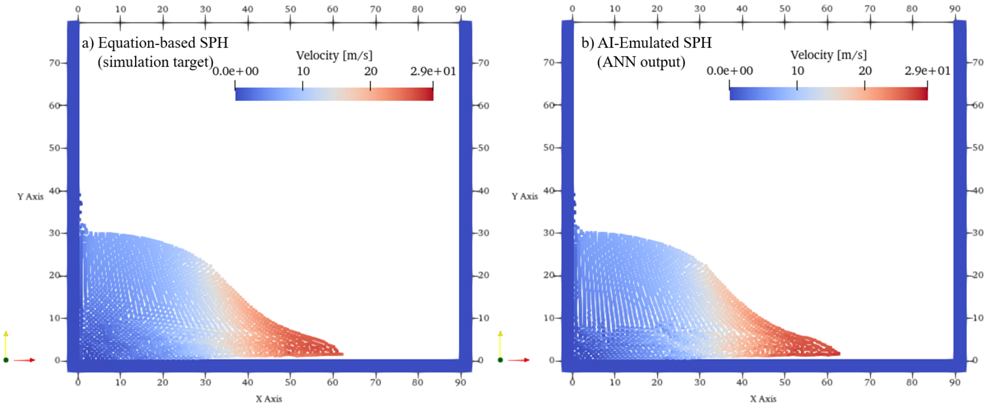


FIGURE 4.11: Simulation (a) and emulation (b) of an inviscid fluid with medium resolution ($\Delta p = 0.75$ m) in a box with regular topography at time $t = 2.0$ s with a laminar structure. The particles are colored by velocity magnitude [m/s].

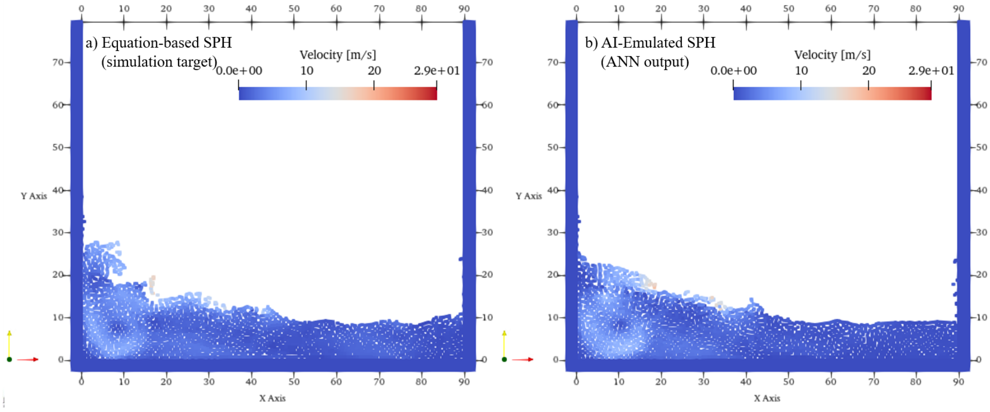


FIGURE 4.12: Simulation (a) and emulation (b) of an inviscid fluid with medium resolution ($\Delta p = 0.75$ m) in a box with regular topography at time $t = 16.0$ s with a swirling structure. The particles are colored by velocity magnitude [m/s]. Non-linearities (as vortices) are triggered in the fluid by the box walls

Therefore, this emulator is capable of faithful reproducing traditional SPH simulations and of generalizing to different problems with varying levels of complexity.

4.1.3 Validation: viscous dam break

After a first validation for inviscid fluids, I applied and validated the emulator for simulations of viscous fluids.

Also in this case, a 2*D* weakly compressible SPH formulation based on the Navier-Stokes equations is used (Rogers et al., 2020; Bilotta et al., 2022b; Zago et al., 2023a). The reference equations are the same of the inviscid case, with the equation of the conservation of the mass as in Eq 1.17 (that recalls the Eq. 1.18), and the density obtained with an inversion of the Cole's equation of state for the pressure (Eq. 1.11), considering a hydrostatic pressure profile. The equation of the conservation of the momentum, in this case, presents also the viscous term, as in Eq. 1.22. Also in this case, the equation of momentum is substituted by the ANN in the emulator.

The physical parameters used for the simulations are the same of the inviscid case, *i.e.*, density $\rho = 1.0$ kg/m^3, and artificial speed of sound $c_0 = 560.29$ m/s (20 times higher than the hydrostatic velocity, $c_0 = 20\sqrt{2\|\vec{g}\|h}$, with h the maximum height of the fluid (Torricelli, 1644)), with a physical viscosity $\mu = 15.0$ Pa s. The fluid box dimensions are 90 × 80 m, and the fluid has an initial shape of 30 × 40 m. The particles are discretized by using a spatial step of $\Delta p = 0.75$ m (Medium Resolution, MR), with 3423 particles. Also in this case, the Wendland kernel was used (Wendland, 1995) (see Appendix A).

A new training is necessary, to teach the network the new behavior of the viscous fluid. To make this, 50.0 s of fluid evolution are simulated and sampled every 0.5 s (fluid and boundary particles together), obtaining a dataset with around 3.42 millions of particle data (with $\Delta p = 0.75$ m). An 80% of this is used for the training phase, while the remaining part for the validation one. The training involves 10000 epochs, with a learning rate α in the order of 10^{-4} (Goodfellow, Bengio, and Courville, 2016), the use of the Adam optimizer and the Mean Squared Error (MSE) loss function (Eq. 3.1). Also in this case, a final training and validation loss in the order of 10^{-3} was obtained.

In Fig. 4.13 the simulation (equation-based SPH reproduction of the dynamics of the fluid) and emulation (AI-CFD emulator reproduction of the dynamics of the fluid) is shown. The fluid is in a box with regular topography and the frame show is at time $t = 2.5$ s. The particles are colored by velocity magnitude [m/s]. A good agreement is found between the simulation and the emulation, to the level of sensitivity of the instrument (Δp), reproducing in a good way the main features of the simulation.

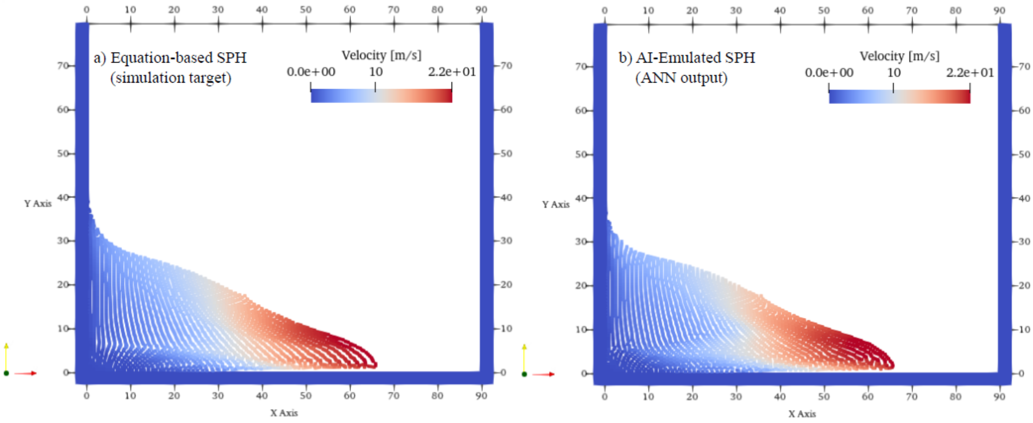


FIGURE 4.13: Simulation (a) and emulation (b) of a viscous fluid with medium resolution ($\Delta p = 0.75$ m) in a box with regular topography at time $t = 2.5$ s. The particles are colored by velocity magnitude [m/s].

Work in progress I am currently working on an extension of the ANN for the viscous case. Until now, the force to be estimated has been the vector with the component of the acceleration, by construction oriented as the vector of relative position between the interacting particles. Therefore, the output of the NN has been the module of the acceleration vector, multiplied in the following step by the unit vector of the relative position. The idea is to extend the network, using the same architecture for the input and hidden layers, and changing only the output layer, adding a second component for the acceleration. In this way, it will be possible to estimate the two parts of the acceleration obtained via the conservation of momentum equation, that are the one linked to the pressure term and the second one linked to the velocity component. Therefore, the network will give in output two terms for the accelerations (for the viscous case), *i.e.*, $\hat{f}_{ij_a}, \hat{f}_{ij_b}$, the first one equal to the actual case, so oriented as the vector of relative position between the interacting particles, the second one oriented as the vector of relative velocity between particles. Therefore, the final output of the ANN will be two-dimensional, and it will be element-wise multiplied by a vector composed by the unit vector of the relative position and the unit vector of the relative velocity. The aim is to consider also the contribution of the velocity in the accelerations estimation, which in the viscous term of the momentum equation takes on a fundamental role for the calculation of accelerations. A possible second aim could be to exploit this considerations to make an inverse analysis. In detail, the idea is to extrapolate the viscous term of the acceleration of a set of data from the ANN and, considering its theoretical formulation (see Eq. 1.22), to invert this term and obtain the viscosity μ_{ij} of the relative fluid.

To obtain an estimation of the pressure and viscous term of the forces, I changed also the second input feature to the NN for the viscous case, the one linked to the velocity component, searching for a more representative feature for the viscous term. I analyzed the single components of the relative velocity vector, *i.e.*, the tangential and normal component ($u_{t_{ij}}, u_{n_{ij}}$). With the $u_{t_{ij}}$, the advancement of the front is closer to the real one, but there is more disorder; with the $u_{n_{ij}}$, the global shape of the emulated flow is more similar than the simulated one, but the position of the simulated and emulated particles does not match over time. In order to consider the two behaviors together, I used the module of the relative velocity $\|\vec{u}_{ij}\|$, to consider the dependence to the entire velocity vector in the conservation of momentum equation for the viscous term. With this new feature, the behavior is better than using the single components, even if the emulation front advances more slowly over time than the simulation one and the emulation is disordered (see Fig. 4.14).

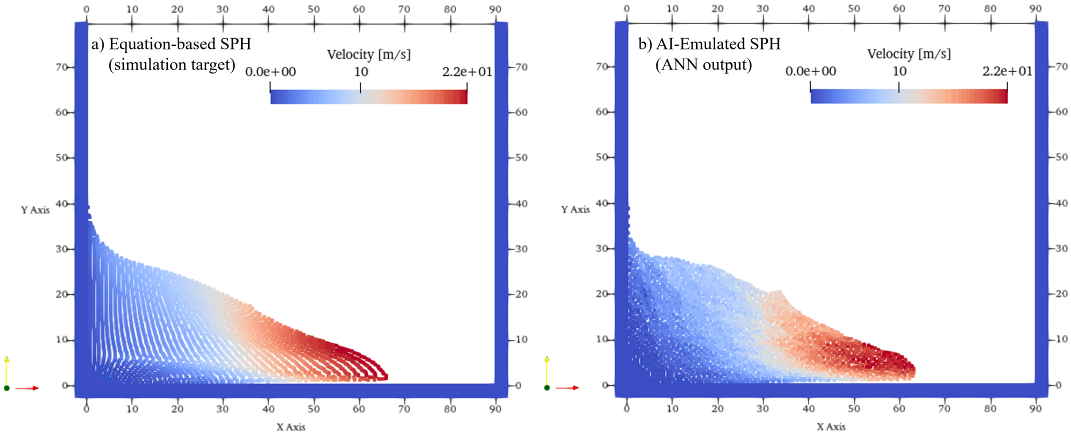


FIGURE 4.14: Simulation (a) and emulation (b) of a viscous fluid with medium resolution ($\Delta p = 0.75$ m) in a box with regular topography at time $t = 2.5$ s and the estimation of the two acceleration components. The particles are colored by velocity magnitude [m/s].

4.1.4 Towards generalization to a complex case: the lava flow

Moving towards lava flows reproductions, I conducted an exploratory study emulating a viscous fluid with thermal component (numerically treated) over an irregular topography, that can recall a volcano flank. I used the model trained over the viscous dam break with a regular topography in the box (see Fig. 4.15), also testing the generalization capabilities of the viscous model.

Thermal model

In order to take into account also the temperature of the fluid particles, the thermal evolution is described by the heat equation, as in Eq. 4.1:

$$\frac{DT}{Dt} = \frac{1}{c_p \rho} \nabla(\kappa \nabla T), \tag{4.1}$$

with T temperature [K], c_p the specific heat at constant pressure [J kg^{-1} K^{-1}], and κ the thermal conductivity [W m^{-1} K^{-1}].

So, I added the SPH discretization of the thermal equation in the implementation (Zago et al., 2018), as in the Eq. 4.2. Currently this term is only numerically treated, namely the temperature is considered as a tracker, as a fluid property, without any coupling with other terms.

$$\frac{DT_i}{Dt} = -\frac{1}{c_p \rho_i} \sum_j \frac{2 m_j \kappa_{ij} T_{ij}}{\rho_j} F_{ij}, \tag{4.2}$$

with T_{ij} the relative temperature between particles. The following physical parameters are used, $c_p = 1000.0$ J kg^{-1} K^{-1}, $\kappa_{ij} = 500.0$, value chosen to accentuate the heating dynamics (very conductive fluid), $T_{fluid} = 293.15$ K the temperature of the fluid and $T_{wall} = 273.15$ K the temperature of the boundaries.

The time step Δt has been calculated so far to fulfill the CFL-like (Courant-Friedrichs-Lewy) stability conditions (Monaghan, 1992), determined by the acceleration magnitude, the speed of sound and the viscous term (see Eq. 1.23). Adding the thermal term, it is necessary to add also the respective stability condition, as in Eq. 4.3

$$\Delta t_i \leq \min \left\{ 0.3 \sqrt{\frac{h}{||\vec{a}_i||}}, 0.3 \frac{h}{c_0}, 0.125 \frac{\rho_0 h^2}{\mu_i}, 0.1 \frac{\rho_0 c_p\, h^2}{\kappa} \right\}. \tag{4.3}$$

In Fig. 4.15 is shows the same simulation and emulation of the viscous fluid case (Fig. 4.13), now numerically tracking also the temperature of the particles, starting with $T_{fluid} = 293.15$ K and $T_{wall} = 273.15$ K. The particles are so colored by temperature [K].

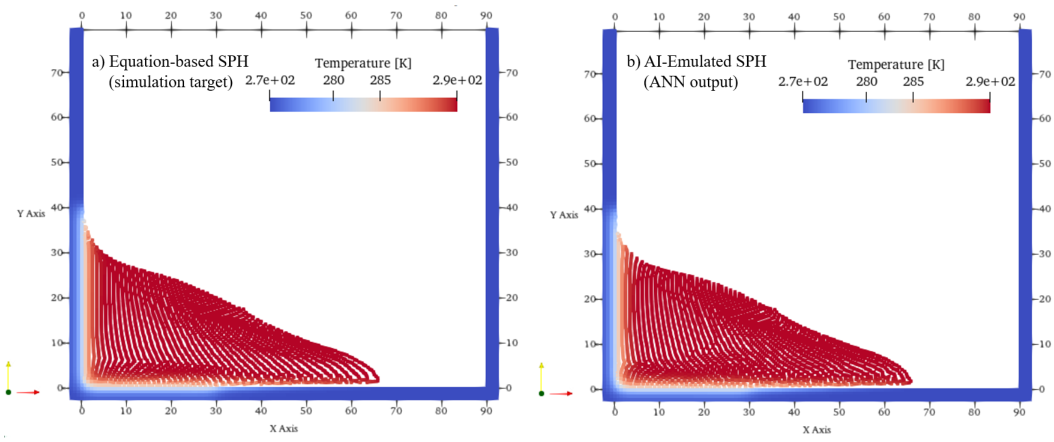


FIGURE 4.15: Simulation (a) and emulation (b) of a viscous fluid with medium resolution ($\Delta p = 0.75$ m) in a box with regular topography and thermal component at time $t = 2.5$ s. The particles are colored by temperature [K].

Fig. 4.16 shows the results of the emulation using a model trained over a different problem respect to the one emulated, *i.e.*, over SPH viscous simulation with a regular topography in the box. The Fig. 4.16, thus, shows a viscous fluid with medium resolution ($\Delta p = 0.75$ m) in an irregular topography at time $t = 12.0$ s. The particles are colored by temperature [K].

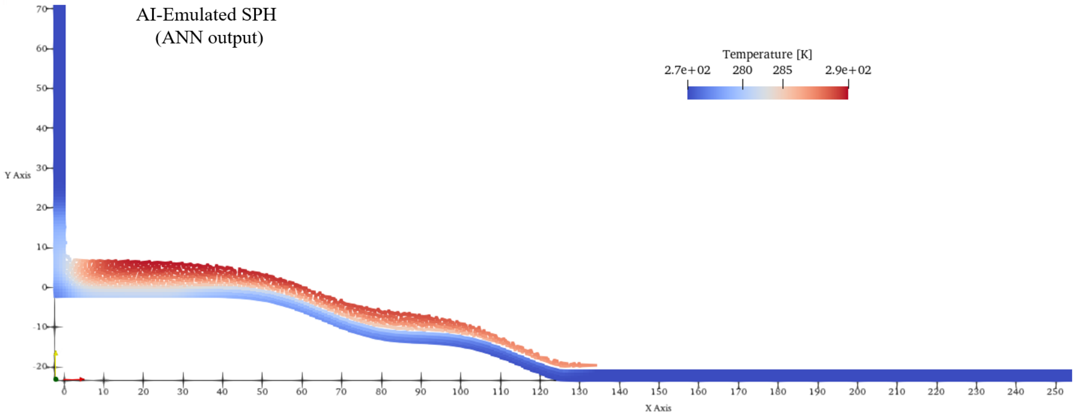


FIGURE 4.16: Emulation of a viscous fluid with medium resolution ($\Delta p = 0.75$ m) in a box with irregular topography and thermal component at time $t = 12.0$ s. The particles are colored by temperature [K]. The model used here was trained over SPH viscous simulation with a regular topography in the box.

Work in progress I am currently working on the viscosity-temperature coupling. Fluids have a characteristic behavior, which links viscosity to temperature. An example of this behavior is found in lava fluid, in which the viscosity is linked to the temperature in a way that when temperature decreases, viscosity increases. The aim of my recent studies is therefore to implement this behavior, through a function that links the two components, and to train the network (with specific representative features) to reproduce it. To do this, the followed steps are to define a function (Eq. 4.4) to model the particle viscosity (μ) with the temperature (T) changes, that is governed by the thermal equation (Eq. 4.2), coupling the two parameters.

$$\mu_i(T_i) = f(T_i). \tag{4.4}$$

Subsequently, it is necessary to search for a characteristic input feature of the NN, linked this behavior, to train the network to emulate it.

Modeling this coupled behavior, starting with an experimental case, in with the viscosity is linearly dependent to the temperature, I numerically coupled temperature and viscosity defining the Eq. 4.5,

$$\mu_i(T_i) = \alpha T_i, \tag{4.5}$$

with $\alpha \in \mathbb{R}$, here $\alpha = 1.5$ to have a behavior in which when temperature decreases, viscosity decreases (behavior hereafter called *direct*).

The viscosity calculated via the Eq. 4.5 is then fed to the conservation of momentum equation (Eq. 1.22), coupling temperature and viscosity. At this point, it is already possible to simulate this behavior with the SPH formulation with a temperature-dependent viscosity, thus not constant.

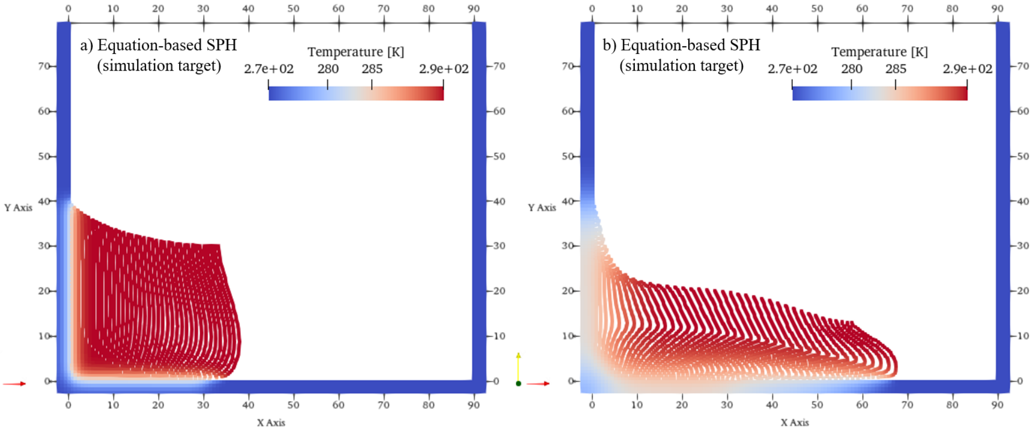


FIGURE 4.17: Simulation of a viscous fluid with medium resolution ($\Delta p = 0.75$ m) and direct viscosity-temperature dependence in a regular box, at (a) time $t = 3.0$ s, and (b) time $t = 24.0$ s. The particles are colored by temperature [K].

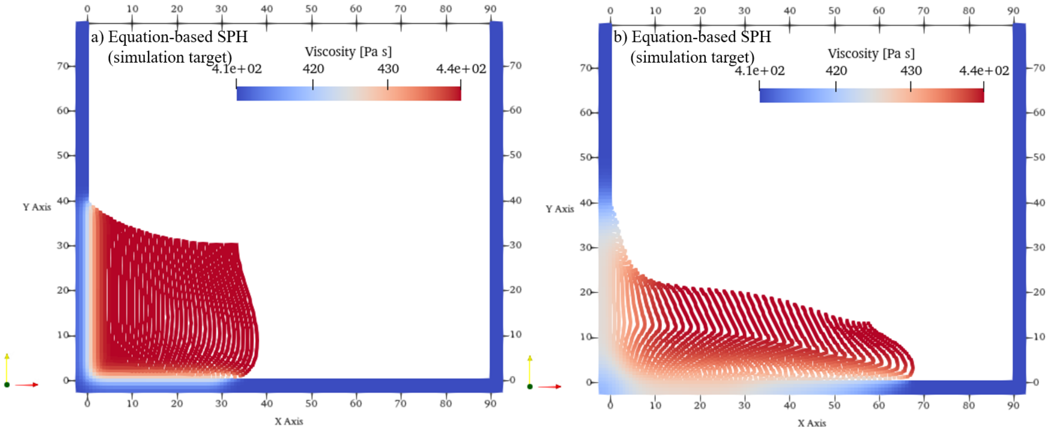


FIGURE 4.18: Simulation of a viscous fluid with medium resolution ($\Delta p = 0.75$ m) and direct viscosity-temperature dependence in a regular box, at (a) time $t = 3.0$ s, and (b) time $t = 24.0$ s. The particles are colored by viscosity [Pa s].

Fig. 4.17 and Fig. 4.18 show the simulation of the viscous fluid with medium resolution ($\Delta p = 0.75$ m) and direct viscosity-temperature dependence in a regular box, at time $t = 3.0$ s (Figs. a), and time $t = 24.0$ s (Figs. b), with the particles colored by temperature [K] (Fig. 4.17) and viscosity [Pa s] (Fig. 4.18). In this case, the temperature of the fluid is set to $T_{fluid} = 293.15$ K, and the temperature of the wall to $T_{wall} = 273.15$ K. As it is possible to see, in this case the fluid cools down the wall, and it flows faster, due to the specific functional dependence of viscosity and temperature, that is, when temperature decreases, viscosity decreases. The viscosity, in this case, has a minimum value around 409.7 Pa s and a maximum value around 439.7 Pa s. In addition, the boundaries represent the unique cooling mechanism, and so they have an effect in the fluid near them.

I also simulate an opposite and more realistic case, an example case in which when temperature decreases, viscosity increases (behavior hereafter called *inverse*).

I numerically coupled temperature and viscosity defining the Eq. 4.6,

$$\mu_i(T_i) = c_0 e^{\frac{1}{T_i}}, \tag{4.6}$$

with $c_0 = 438.41$, a constant of proportionality between the initial viscosity at temperature $T_0 = 293.15$ K in case of exponential inversion ($\mu_0 = 1.5 \cdot T_0 = 439.73$ Pa s) and linear coupling ($\mu_0 = e^{1/T_0} = 1.00$ Pa s).

The viscosity calculated via the Eq. 4.6 is used with the same procedure of the precedent case.

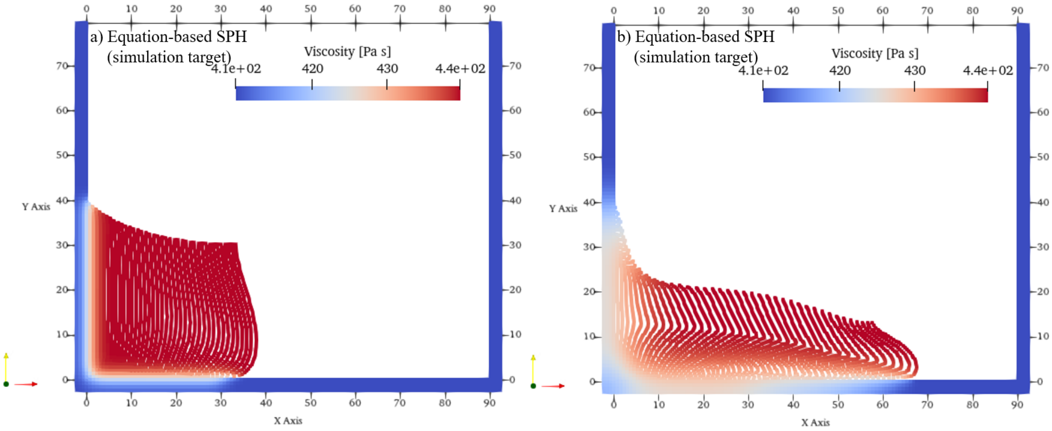


FIGURE 4.19: Simulation of a viscous fluid with medium resolution ($\Delta p = 0.75$ m) and inverse viscosity-temperature dependence in a regular box, at (a) time $t = 3.0$ s, and (b) time $t = 24.0$ s. The particles are colored by temperature [K].

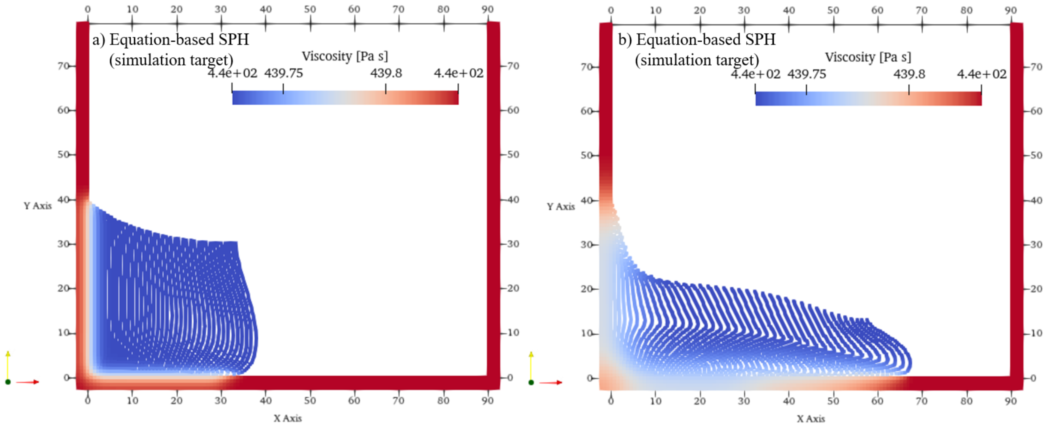


FIGURE 4.20: Simulation of a viscous fluid with medium resolution ($\Delta p = 0.75$ m) and inverse viscosity-temperature dependence in a regular box, at (a) time $t = 3.0$ s, and (b) time $t = 24.0$ s. The particles are colored by viscosity [Pa s].

Fig. 4.19 and Fig. 4.20 show this case, that is the simulation of the viscous fluid with medium resolution ($\Delta p = 0.75$ m) and the inverse viscosity-temperature dependence in a regular box, at time $t = 3.0$ s (Figs. a), and time $t = 24.0$ s (Figs. b), with the particles colored by temperature [K] (Fig. 4.19) and viscosity [Pa s] (Fig. 4.20). As it is possible to see, in this case the fluid cools down the wall and when temperature decreases, viscosity increases, due to the specific functional dependence of viscosity and temperature.

Moving toward a more complex case, more representative, for example, of the lava behavior, I implemented a functional form for the viscosity similar to (Zago et al., 2018), as in Eq. 4.7

$$\mu_i(T_i) = 10^{\left(-5+\frac{4000}{T_i-610}\right)}. \tag{4.7}$$

In this case I used a boundary model considered as thermal source (see A.3), *i.e.*, the boundary has a specific temperature and it remains constant for the entire simulation or emulation.

The viscosity calculated via the Eq. 4.7 is fed to the conservation of momentum equation (Eq. 1.22), coupling temperature and viscosity. In addition, I compared this case with a case representative of a fluid with independent temperature and viscosity.

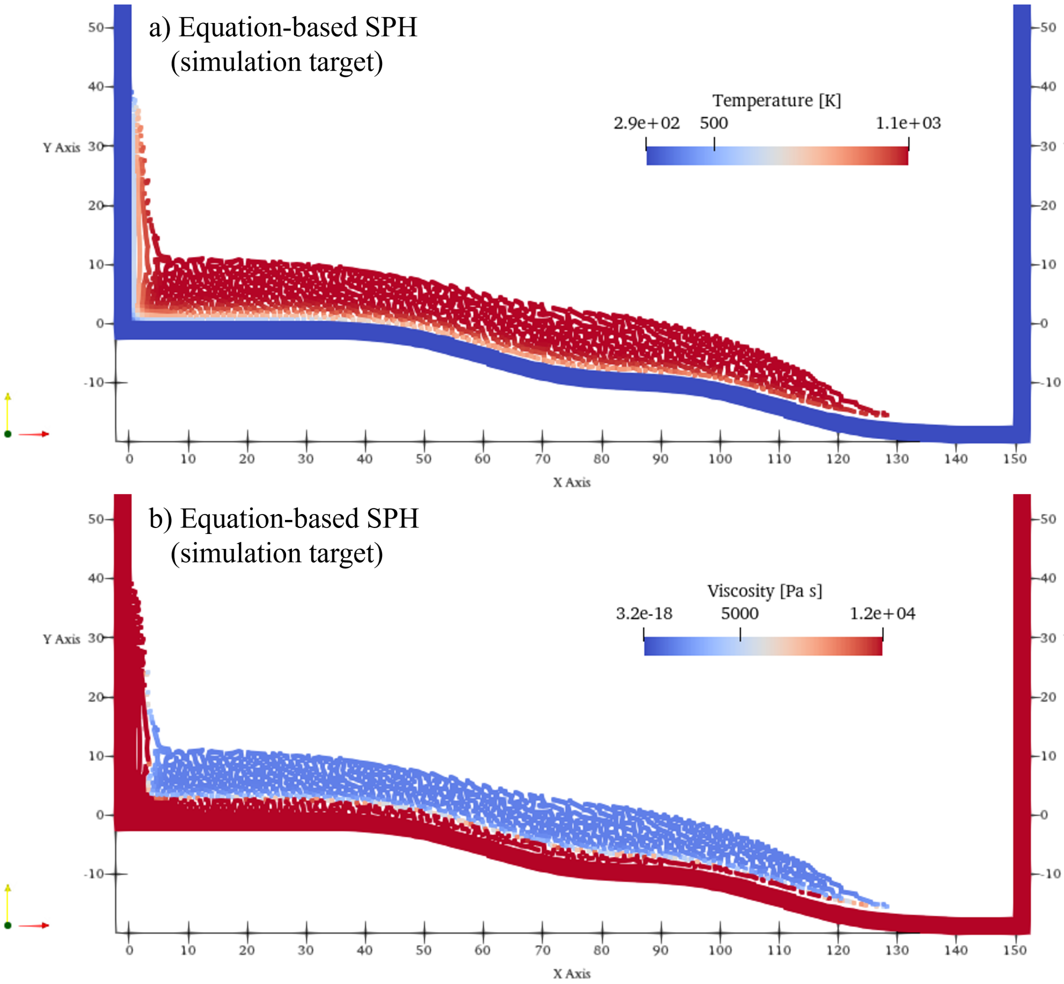


FIGURE 4.21: Simulation of a viscous fluid with medium resolution ($\Delta p = 0.75$ m) and viscosity-temperature dependence (similar to lava case) in a box with bumps, at time $t = 30.0$ s. The particles are colored by temperature [K] (a) viscosity [Pa s] (b).

Fig. 4.21 shows the case with viscosity temperature-dependent, following the Eq. 4.7, with the temperature of the fluid set equal to $T_{fluid} = 1100.0$ K, and the temperature of the wall to $T_{wall} = 290.0$ K. The density is set as $\rho = 100.0$ kg m^{-3}. The particles are colored by temperature [K] (Fig. 4.21 a) and viscosity [Pa s] (Fig. 4.21 b). As it is possible to see, in this case, when temperature decreases, viscosity increases, in an inverse behavior, typical of lava flows. In this way, the flow of lava is also reduced as the viscosity increases.

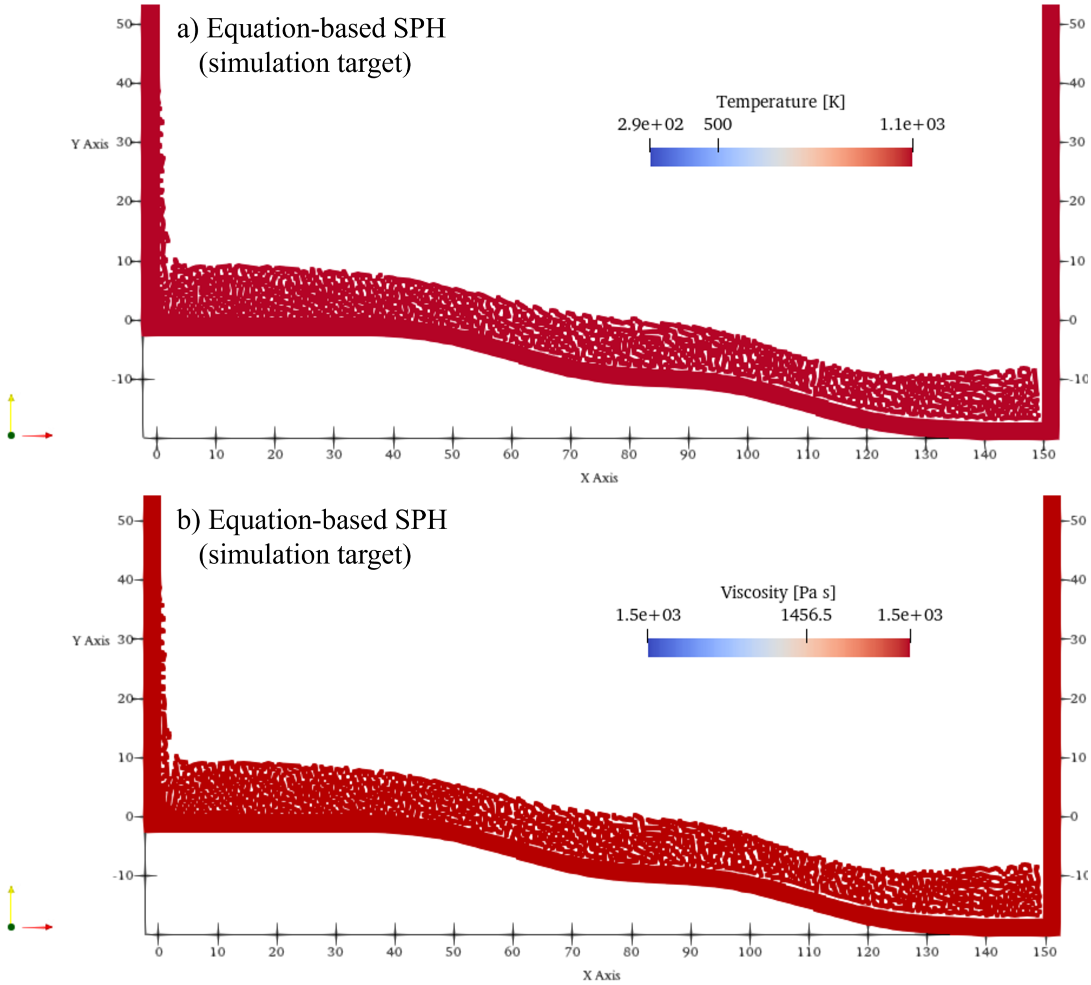


FIGURE 4.22: Simulation of a viscous fluid with medium resolution ($\Delta p = 0.75$ m) and viscosity-temperature independence in a box with bumps, at time $t = 30.0$ s. The particles are colored by temperature [K] (a) viscosity [Pa s] (b).

Fig. 4.22 shows, in contrast, a case with viscosity temperature-independent. The temperature of the fluid and of the wall are equally set to $T_{fluid} = T_{wall} = 1100.0$ K. The density is set as $\rho = 100.0$ kg m^{-3}. The particles are colored by temperature [K] (Fig. 4.22 a) and viscosity [Pa s] (Fig. 4.22 b). As it is possible to see, in this case, temperature and viscosity are constant and the flow of lava is faster then the precedent case (Fig. 4.22).

The work I am currently conducting aims to modify the ANN in order to train it to emulate this behavior. One of the first steps is the addition of a new input feature, as well as to the relative position between particles, the normal component of their relative velocity and the product of their densities, which can provide information about the thermal term and better characterize this behavior. One candidate is the temperature of the particles, directly extracted by Eq. 4.2. I also noticed that, as for the choice of the other features, looking at the equation to emulate (Eq. 1.22), the viscosity term appears. Thus, I could used the viscosity feature as fourth feature, that could be more representative than the temperature of the dynamics of the system and in a better way it could provide the correct information to the network to learn this dynamics.

4.2 Speed-up

Reducing the computational load of the numerical equations, substituting them with the ANN and the use of the look-up table (see Subsection 3.1.1), the time to estimate the forces is *halved*.

Existing some limits in the reduction of the computational load of the numerical part, I also acted in removing computational load in time. For the emulated block, it is possible to use a longer time step than that used in the reference simulation by training the ANN to predict the force with the ground truth reference over a longer time step. Thus, I have trained the NN in this way, with the aim to learn how to use a longer time step than that used in the reference simulation for the emulated predictions. Formulating the ANN estimator as follow (Eq. 4.8)

$$a_{ij}^{t+k\Delta t} = ANN(x_{ij}^{t},\, u_{n_{ij}}^{t},\, \rho_i^{t} \cdot \rho_j^{t}), \tag{4.8}$$

I used a $k = 5$ times larger step for the training. In this way, the results are stable, obtaining a gain in term of simulation time.

It is important to notice that using a two-step predictor-corrector integration scheme (see Subsection 1.2.4), that being a double-step approach is more accurate and stable (Anderson and Wendt, 1995; Chung, 2002), it was possible to reach and implement this larger step.

Multiplying the two speed-up contributions just discussed here, I have actually obtained a total speed-up of the emulated block equal to *10 times*.

Work in progress I am currently working on a further speeding-up of the model, modifying the code from a technical and conceptual point of view. The first ideas are to move all the code on GPU (and not just the training block) and to continue with the code optimization process. In addition, I am trying to act directly on the implementation of some parts of the model, improving the choices of parameters and hyperparameters to simplify the learning process. In addition, I am also working on an extension of the simulated sampled data, including older time samples and further points to extend the estimate to longer time steps. The stability limits of such models are linked to the propagation of information (see Eq. 4.3). By widening the time step or considering sampled data more distant in space or time (so the propagation of information can be better followed), it should be possible to better follow the evolution of these systems.

Conclusion

Following the evolution in time and space of complex fluids is a hard task for fluid dynamics models. I have exploited the combined knowledge in CFD and AI fields to build an emulator, that learns from CFD simulations the behavior of the CFD reference model, solving fluid dynamics problems with higher fidelity and in shorter times, preserving the physical consistence to the reference model (both correctness and completeness). Therefore, I validated the AI-CFD-based emulator, comparing emulations with equation-based simulations, using different case studies, from inviscid flows to highly complex fluids, such as lava.

The spatio-temporal evolution of lava flows depends on its properties, as non-Newtonian rheology, thermal dependence and solid-fluid interactions, and the topography over which the lava can flow. The complexity of lava makes difficult to faithfully describe it in near real-time using CFD models. This is, therefore, one of the exemplary cases that motivates the use of combined AI-CFD models to follow the spatio-temporal dynamics of a fluid.

Starting from the CFD Smoothed Particle Hydrodynamics (SPH) formulation of an analytical fluid dynamic problem, the numerical equation of momentum conservation has been substituted by an Artificial Neural Network (ANN), trained to extrapolate the behavior of this numerical equation. In detail, ANN has been trained to estimate the SPH particles interaction forces exploiting as input the state of each particle, which includes relative position of nearby particles, their relative velocity, and their densities. I have chosen the SPH method because it allows high fidelity simulations of complex fluids, and I have combined it with an ANN, because it is capable of appropriately approximating any kind of functions, well generalizing to unknown cases. I started with an inviscid and isothermal fluid, finally arriving at the development of an emulator capable of modeling complex highly viscous fluids, with a thermal component that varies following the behavior of the lava fields.

I conducted a study in AI and modeling state-of-the-art, testing specific applications, highlighting their advantages and limits. I explored different mathematical models in fluid dynamics, and their relative discretized version, increasing the accuracy of the numerical representation and, as a counterpart, their computational costs. I also studied several AI techniques to automatic analyze huge quantity of data. I applied different ML and DL techniques to analyze volcanic data from Space, deriving information on active lava flows, as areal extension and volume of lava erupted, obtaining eruptive parameters to feed as input to the mathematical models for volcano monitoring.

The limits in numerical modeling, mainly due to the high computational costs, and the need to deeply investigate physical phenomena, led to design an emulator that leverages on the combination of SPH and AI models. The AI-based SPH emulator has been constructed to ensures symmetric particle interactions, to be independent from local particles configurations and to allow an exact use of classical SPH formulations, including boundary conditions. I validated and tested the emulator, showing and discussing different study cases. Firstly, I analyzed inviscid and isothermal fluids, verifying that the emulator takes well into account the reproducibility of major features and of boundary conditions. Simulations and emulations have been compared, showing a good agreement. I reached a speed-up of the emulated block of one order of magnitude with respect to the numerical model. Moreover, I demonstrated the generalizability of the AI-based emulator, testing the model over problems with varying levels of complexity, and the robustness to different spatial resolutions, reproducing the entire procedure (simulation, training, emulation) with different resolutions. I progressively added the viscous and thermal models, to eventually simulate lava flows. Also in these cases, simulations and emulations have been compared for each step, obtaining a high degree of fidelity, and demonstrating the capacity to reproduce the main features and to generalize to new cases out of the training set.

The emulator presented here is capable to faithfully reproduce SPH simulations in shorter times with the use of an ANN. It is applicable to fluids with different viscosity and thermal properties, managing different resolutions and being capable to generalize to problems with different complexity. It exploits the potential of the combined use of numerical and AI models, enhancing CFD simulations and opening to a deeper understanding of physical phenomena.

Appendix A

SPH method

In this Appendix the theoretical basis of the SPH are presented, starting from the mathematical derivation of the SPH approximation and of the spatial gradient and second order derivative discretization, up to the kernel choice and the SPH initial and boundary conditions setting.

A.1 Mathematical derivation of the SPH approximation

The mathematical derivation of the SPH approximation will be shown.

A.1.1 SPH field discretization

Considering a field f defined on a domain Ω, and defining a Dirac's delta distribution δ, the value assumed by f at any location $\vec{x}^* \in \Omega$, according to Dirac's delta sampling property, can be the following one

$$f(\vec{x}^*) = \int_\Omega f(\vec{x})\delta(\vec{x} - \vec{x}^*)d\vec{x}. \tag{A.1}$$

Considering a family of functions $W(\cdot, h)$ (smoothing kernels), that use $h > 0$ (smoothing length) to be parameterized, these functions can approximate the Dirac's delta, in such a way that

$$\int_\Omega W(\vec{x} - \vec{x}^*, h)d\vec{x} = 1, \tag{A.2}$$

$$lim_{h\to 0} W(\vec{x} - \vec{x}^*, h) = \delta(\vec{x} - \vec{x}^*), \tag{A.3}$$

with the limit in the sense of distributions.

Using this formulations, the SPH field is obtained as follows,

$$f(\vec{x}^*) \approx \sum_{i=1}^{N} f(\vec{x}_i)W(\vec{x}_i - \vec{x}^*, h)V_i, \tag{A.4}$$

approximating the integral with a summation over a finite set of points, that are the i particles, with $i \in \mathbb{N}$, $1 < i < N$, at positions $\vec{x}_i$, with a fixed mass m_i, a density ρ_i, and then a volume $V_i = \frac{m_i}{\rho_i}$. In addition, the smoothing kernel $W(\cdot, h)$ is chosen to have a compact support; in this way, the summation is only over a small neighborhood of the central particle $\vec{x}^*$ (Liu and Liu, 2003).

Eq. A.4 shows that SPH formulation approximates Dirac's delta with the smoothing kernels; this approximation vanishes for $h \to 0$. In addition, Eq. A.4 discretizes the domain using a finite set of particles, controlled by the average inter-particle spacing Δp and this approximation vanishes for $\Delta p \to 0$. Finally, to ensure the consistency for the method, it is necessary to impose the following property, that is that Δp must tend to zero faster than h, with a constant ratio, typically $\frac{h}{\Delta p} \in [1.3, 1.5]$ (Liu and Liu, 2003).

A.1.2 SPH spatial gradient discretization

To discretize the spatial gradient of the field f, it is necessary to use a smoothing kernel with a radial symmetry, $W(r,h)$, with $r = |\vec{x} - \vec{x}^*|$. With this in mind, it is possible to approximate at any given point the field gradient ∇f by convolving it with a smoothing kernel,

$$\nabla f(\vec{x}^*) \approx \int_\Omega \nabla_{\vec{x}} f(\vec{x}) W(|\vec{x} - \vec{x}^*|, h) d\vec{x}. \tag{A.5}$$

Applying the Green's theorem at Eq. A.5, with $\Sigma = \partial\Omega$, the Eq. A.6 is obtained,

$$\nabla f(\vec{x}^*) \approx \int_\Sigma f(\vec{x}) W(|\vec{x} - \vec{x}^*|, h) \vec{n} d\Sigma - \int_\Omega f(\vec{x}) \nabla_{\vec{x}} W(|\vec{x} - \vec{x}^*|, h) d\vec{x}, \tag{A.6}$$

with $\vec{n}$ the normal of the domain.

Starting from this equation, the SPH approximation of the gradient is then obtained as in Eq. A.7, because the first integral in the right part of the Eq. A.6 is equal to zero if $\vec{x}^*$ is far from the boundary, in the second integral, $-\nabla_{\vec{x}} W(|\vec{x} - \vec{x}^*|, h) = \nabla_{\vec{x}^*} W(|\vec{x} - \vec{x}^*|, h)$ for symmetry (Zago et al., 2018), and the integrals are discretized using summations over N neighbors particles, for $i \in \mathbb{N}$.

$$\nabla f(\vec{x}^*) \approx \sum_i^N f(\vec{x}_i) \nabla_{\vec{x}^*} W(|\vec{x}_i - \vec{x}^*|, h) V_i. \tag{A.7}$$

This formulation allows to calculate an approximation of the spatial gradient of the field f, knowing only the gradient of the smoothing kernel, instead of the field gradient (Zago, 2019).

In the SPH formulation, the relationship between a central particle and all its neighbors is analyzed. Thus, the Eq. A.7 can be applied to a particle j in such a way,

$$\nabla f(\vec{x}_j) \approx \sum_i^N (f(\vec{x}_i) - f(\vec{x}_j)) \nabla_j W(|\vec{x}_i - \vec{x}_j|, h) V_i, \tag{A.8}$$

symmetrized with the $-f(\vec{x}_j)$ term.

The gradients are calculated as the sum of the properties on neighboring particles, weighted by the kernel gradient, so the latter can be written as in Eq. A.9 (for the radial symmetry), defining $\vec{x}_{ij} = \vec{x}_i - \vec{x}_j$ and $W_{ij} = W(|\vec{x}_i - \vec{x}_j|, h)$,

$$\nabla_j W_{ij} = \frac{\vec{x}_{ij}}{|\vec{x}_{ij}|} \frac{\partial W(r,h)}{\partial r}\bigg|_{r=|\vec{x}_{ij}|}, \tag{A.9}$$

and it is possible to set the analytical expression A.10

$$F(r) = \frac{1}{r} \frac{\partial W(r,h)}{\partial r}, \tag{A.10}$$

such that $\nabla_j W_{ij} = \vec{x}_{ij} F(|\vec{x}_{ij}|) = \vec{x}_{ij} F_{ij}$.

A.1.3 SPH spatial second order derivative discretization

The discretization of the second order derivative can be done exploiting the first order derivative of the kernel (Brookshaw, 1985; Zago, 2019), and not applying two times the process applied for first derivatives, due to the sensitivity to particle disorder and being very noisy.

Firstly, a Taylor approximation of the field f can be done, as follows

$$f(\vec{x}_i) = f(\vec{x}_j) + \nabla f\Big|_{\vec{x}_j}(\vec{x}_i - \vec{x}_j) + \frac{1}{2}\frac{\partial^2 f}{\partial \vec{x}_l \partial \vec{x}_m}(\vec{x}_i - \vec{x}_j)_l(\vec{x}_i - \vec{x}_j)_m + \mathcal{O}(\vec{x}_i - \vec{x}_j)^3, \tag{A.11}$$

with x_i, x_j positions of i and j particles and l and m first and second component of the position vector. Eq. A.11 is multiplied by $\frac{\vec{x}_{ij}\nabla_j W_{ij}}{|\vec{x}_{ij}|}$ and integrated over the domain Ω. In this way, the first order term is equal to zero, due to the spherical symmetry of W, and the second order term is equal to δ_{lm}. Then,

$$\nabla^2 f\Big|_{\vec{x}_j} \approx -2\int_\Omega \frac{f(\vec{x}_j) - f(\vec{x}_i)}{|\vec{x}_{ij}|^2}\vec{x}_{ij}\nabla_j W_{ij} d\vec{x}_i^3,$$

in which the integral can be discretized with the sum over i, and the term $d\vec{x}_i^3$ with the volume $V_i = \frac{m_i}{\rho_i}$ (in 3D), obtaining, the SPH formulation as in Eq. A.12

$$\nabla^2 f\Big|_{x_j} \approx -2\sum_i \frac{f(\vec{x}_j) - f(\vec{x}_i)}{|\vec{x}_{ij}|^2}\vec{x}_{ij}\nabla_j W_{ij}\frac{m_i}{\rho_i} = -2\sum_i \frac{f_j - f_i}{|\vec{x}_{ij}|^2}\vec{x}_{ij}\nabla_j W_{ij}\frac{m_i}{\rho_i}, \tag{A.12}$$

with $\nabla^2 f\Big|_{\vec{x}_j} = 0$, if f constant.

Similarly,

$$\nabla \cdot (Q(x_i)\nabla f(x_i)) \approx -2\int_\Omega \frac{(Q_j + Q_i)(f_j - f_i)}{|\vec{x}_{ij}|^2}\vec{x}_{ij}\nabla_j W_{ij} d\vec{x}_i^3,$$

has an SPH formulation as

$$\nabla \cdot (Q\nabla f) \approx -2\sum_i \frac{(Q_j + Q_i)(f_j - f_i)}{|\vec{x}_{ij}|^2}\vec{x}_{ij}\nabla_j W_{ij}\frac{m_i}{\rho_i} = -2\sum_i (Q_j + Q_i)(f_j - f_i)F_{ij}\frac{m_i}{\rho_i}, \tag{A.13}$$

with Q that may show a spatial variation, and $F_{ij} = \frac{\nabla_j W_{ij}}{\vec{x}_{ij}}$, (Monaghan, 1992; Brookshaw, 1985).

A.2 SPH kernels

As for the choice of the different schemes in the finite difference methods, in SPH the kernel choice is fundamental. Different types of kernel are available, and each of these must satisfy Eqs. A.2 and A.3. In addition, it must be positive and defined on a compact support, monotonically decreasing with the increase of distance away from the particle, it must be even and sufficiently smooth. Let be $\delta \in \mathbb{N}$, defining a cut-off radius R, $R = \delta \cdot h$, with h smoothing length, and a variable $q = |\vec{x}_{ij}|/R$, $0 \leq$

$q \leq \delta$, different SPH smoothing kernels can be defined. The kernels depend on the h parameter, that is the smoothing length, defined using the smoothing factor as $s_f = \frac{h}{\Delta p}$, with Δp spatial resolution of the approximation (particle size) and, typically, $s_f = 1.33$ (Zago et al., 2018).

A first example of kernel is the Lucy one (Lucy, 1977), with radius $\delta = 1$, defined as in Eq. A.14 and Eq. A.15,

$$W(q) = \frac{\alpha}{h^2}(1+3q)(1-q)^3, \tag{A.14}$$

$$F(q) = \frac{\alpha}{h}\frac{3c_{l1} - 3c_{l2}(1+3q)}{qh}, \tag{A.15}$$

in $2D$, with $\alpha \approx 1.592$ Lucy constant, and

$$c_{l1} = c_{l2}(1-q),$$

$$c_{l2} = (1-q)^2.$$

One of the most used SPH kernel is the Wendland one (Wendland, 1995), which has been observed to be beneficial for free-surface simulations (Macia Lang et al., 2011). It has radius $\delta = 2$, and it is defined as in Eq. A.16 and Eq. A.17,

$$W(q) = c_{w1}(1-0.5q)^4(1+2q), \tag{A.16}$$

$$F(q) = c_{w2}(q-2)^3, \tag{A.17}$$

with

$$c_{w1} = \frac{7}{4h^2\pi},$$

$$c_{w2} = \frac{35}{32h^4\pi},$$

in $2D$, and

$$c_{w1} = \frac{21}{16h^3\pi},$$

$$c_{w2} = \frac{5c_{w1}}{8h^2},$$

in $3D$.

Finally, another kernel is the Gaussian one (Monaghan, 1992), with radius $\delta = 3$, as in Eq. A.18 and Eq. A.19

$$W(q) = \frac{1}{c_{g1}}e^{-q^2} - e^{-\delta^2}, \tag{A.18}$$

$$F(q) = -\frac{1}{c_{g2}}e^{-q^2}, \tag{A.19}$$

with

$$c_{g1} = h^2\pi(1 - e^{-(1+\delta^2)}),$$

in $2D$,

$$c_{g1} = h^3\pi(\sqrt{\pi}Erf(\delta) - \frac{2}{3}e^{-\delta(3+2\delta^2)}),$$

in $3D$ ($Erf()$ is the error function), and

$$c_{g2} = h^2\frac{c_{g1}}{2}.$$

A.3 Initial and boundary conditions

In modeling, the initial and boundary conditions are the setting of all the initial and at borders states and parameters, to make the mathematical problem uniquely solvable (Velten, 2009). In SPH initialization phase, the problem is implemented, creating the set of particles for the fluid and the walls.

The initial values of each particle is assigned, to define the initial state of the variables (position, velocity, acceleration, density, temperature, viscosity). Also the physical parameters are set, as the gravity, the speed of sound, the topography, the initial viscosity and temperature, and the numerical parameters, as the spatial resolution of the simulation, the number of particles used for the discretization, the time step, and the kernel used. Finally, the chosen boundary conditions are applied.

In physics, mechanical boundary conditions (BC), *i.e.*, those implemented imposing a value to the velocity and the pressure, can be of several types, as the no-slip and free-slip (Velten, 2009; Zago, 2019). In the first case, the boundary velocity is set to be zero, and this condition is obtained assigning a zero normal velocity and a physical sliding tangential velocity at the boundary (sometimes equal to zero). In the second case, the free-slip BC, the flow is free to slip over the boundaries, and this condition is obtained imposing a zero tangential shear stress along the boundaries. In an SPH context, boundary conditions are typically assigned by means of boundary particles, which act on the velocity and pressure of the nearby fluid particles. The boundary particles are used to complete the support of the smoothing kernel and to apply BCs; for example, for a smoothing kernel of radius $\delta = 2$ and a smoothing factor $s_f = 1.33$, three layers of particles are necessary.

In addition, generically speaking, a *boundary layer* is a layer of fluid in the immediate vicinity of a boundary, that can be altered by the presence of it (in particular its viscosity) (Schlichting and Gersten, 2016). In the SPH context, the boundary layer is often smaller than δp (Cleary and Monaghan, 1993), so it is studied with semi-analytical models (Ferrand et al., 2013), adding a corrective viscosity term that models it.

The way in which the properties of the boundary particles are set, determines the boundary models, *i.e.*, the specific SPH configurations of the BCs. Examples of SPH boundary models are the Dummy (Adami, Hu, and Adams, 2012) and Dynamic boundary models (Crespo, Gómez-Gesteira, Dalrymple, et al., 2007). In the first case, the density is computed to achieve a pressure that matches the Shepard averaged pressure of the neighboring fluid (Eq. A.20), and the velocity is obtained by adding to the wall velocity ($\vec{u}_w$) the opposite of the Shepard-averaged velocity of the neighboring fluid (Eq. A.21).

$$P_i = \frac{\sum_j P_j W_{ij} + \vec{g} \sum_j \rho_j \vec{x}_{ij} W_{ij}}{\sum_j W_{ij}}. \tag{A.20}$$

$$\vec{u}_i = \vec{u}_w - \frac{\sum_j \vec{u}_j W_{ij}}{\sum_j W_{ij}}, \tag{A.21}$$

For the second case, the dynamic boundary models, the boundary density is treated equally to the fluid particles one, and the velocity is defined in a specific way, for example equal to zero (obtaining, thus, no-slip boundary conditions). Due to the implementation of the density, in this last case, the transition from a dry to wet states can leads to a phenomenon in which the increasing in density brings the fluid to "lift" the boundary, and, conversely, the fluid impinging on a boundary and subsequently flowing away brings to a reduction of the density. This phenomenon is known as *instability (or transition) dry-wet or wet-dry* (Zago et al., 2018). In addition, fluid particles could penetrate the walls. Other kinds of BC can be the Periodic ones, to reproduce infinitely extended domains, in a way in which when a particle arrives near a periodic boundary it crosses it, disappears and reappears on the opposite side of the periodic domain, maintaining the same state; also the neighbors search suffers the periodic properties of the domain, applying it also on the opposite side of it. Another type of BC is the Open Boundary (OB) or Non-Reflective conditions (Ferrand et al., 2017), used to add to the simulation domain a region not real but that can exchange matter and information with the actual implemented domain (waves propagation, with a wall that absorbs mass and wave), for example for fluid inlet. Generalized Riemann Invariants (Ferrand et al., 2017) are computed from the prescribed boundary conditions and they are used to extrapolate information from the inside of the domain, avoiding spurious reflections, and maintaining a continuity between the two sides of the periodic domain.

Moving to thermal boundary conditions, a first case is the thermal source, that is, the entire boundary domain D maintains the same temperature T_a in the entire simulation time, and any heat exchanges are not constrained, *i.e.*, $T(\vec{x}, t) = T_a$, $\forall \vec{x} \in D$, $\forall t \in \mathbb{R}$ (Zago, 2019). An alternative is the adiabatic one, *i.e.*, $\nabla T \cdot \vec{n} = 0$, with $\vec{n}$ the vector normal to the boundary region D, thus $T_i = \frac{\sum_j T_j (x_j - x_i) \cdot \vec{n} F_{ij} V_i}{\sum_j (x_j - x_i) \cdot \vec{n} F_{ij} V_i}$ is the temperature to be assigned to the wall particle i (Zago, 2019). Finally, another case is the sponge layer (a thermal open boundary), that is an absorbing boundary, *i.e.*, the boundary is considered a body in which the heat of the fluid propagates, therefore it must be thick enough to propagate it during the simulation. In this case, thus, $T(0, 0) = T_0$, and $T(-H, t) = T_0$, with T_0 the initial temperature, and H the thickness of the boundary, and $\frac{4\kappa_w t_{end}}{\rho_w c_{p_w} H^2} < 1$ (CFL-like condition), with κ_w the wall thermal conductivity, t_{end} the end of the simulation, ρ_w the wall density, c_{p_w} the wall specific heat at constant pressure (Hérault, 2008).

Appendix B

Volcano monitoring from Space

B.1 Satellite general aspects

The changes in surface of the Earth (a natural or anthropomorphic event that modifies its appearance, *e.g.*, a landslide or eruption, or the introduction of a new human construction) and, in particular, volcanic activity in even the most isolated areas can be observed via the satellite data, providing spectral estimations for the Earth's surface features without the risks linked to reach an area potentially threatened by an eruption.

Satellites are divided into geosynchronous and polar ones (Harris, 2013). Satellites in geosynchronous orbits (GEO) circle the Earth from West to East, following Earth's rotation (taking about 24 hours) by traveling at exactly the same rate as the Earth. This makes GEO satellites to be "stationary" over a fixed position. In order to perfectly match Earth's rotation, the speed of GEO satellites should be about 3 km per second at an altitude of 35786 km. They are so much farther from Earth's surface compared to many other satellites. A geostationary satellite is a geosynchronous satellite orbiting with zero inclination to the equatorial plane. In contrast, satellites in polar orbits usually travel the Earth from North to South, passing roughly over the Earth's poles. Satellites in polar orbits do not have to pass the North and South Poles precisely; even a deviation within the range [20° - 30°] is still classified as a polar orbit, and they have low Earth orbit, with an altitudes between 200 to 1000 km. Sun-synchronous orbit is a particular kind of polar orbit which allows the satellite to be synchronous with the Sun, this means that the satellite always visits the same spot at the same local time.

Different kinds of resolution are defined in satellite remote sensing images (Harris, 2013), that are:

- spectral resolution, which is the ability of a sensor to discern among wavelengths;
- spatial resolution, which is the size of an image pixel, representing an area on the Earth's surface;
- temporal resolution, which indicates how frequently a satellite can provide observation of the same area on the Earth (*satellite revisit time* is the time elapsed between observations of the same point on Earth by a satellite);
- radiometric resolution, which is the amount of information in each pixel, *i.e.*, the number of bits that represent the recorded energy.

The theoretical background of the satellite acquisition is based on the spectral response of the surfaces. All the objects with a temperature above 0 K emit electromagnetic radiations. Satellite sensors will be differentiate to be passive or active sensors, if they do not emit radiation themselves, or if they emit radiations (Harris, 2013). Examples are the multispectral or hyperspectral active sensors and the LiDAR or RADAR passive sensors. A blackbody is an ideal object that absorbs all incident radiation and emits energy following Planck's law (Planck, 1901).

B.1.1 Examples of satellites

In a geophysical context, the satellites most used are classifiable respect to the temporal and spatial resolution. In general, the most common satellites have an inverse proportionality between spatial and temporal resolution (Harris, 2013).

High temporal and low spatial resolution Starting from satellite data with high temporal resolution and low spatial one, used for near real-time analysis, the Spinning Enhanced Visible and Infrared Imager (SEVIRI) is the most used in European zones. It is on board the geostationary Meteosat Second Generation (MSG) satellites (from ESA EUMETSAT) and it measures radiations in 12 spectral bands, from the Visible (VIS) to the InfraRed (IR) part of the spectrum. Its spatial resolution depends on the distance from the sub-satellite ground point (thus, latitude, longitude) and on viewing zenith angle. It is down to 1 km for the High-Resolution Visible channel and of 3 km at sub-satellite point for standard channels. In a volcanic context, it is possible to extract a parameter for the monitoring of volcanic activity, that is the Volcanic Radiative Power (VRP), a parameter that describes the quantity of heat emitted by a volcanic product, measured using a combination of the spectral radiance (Planck, 1901) in the Middle Infrared (MIR) (at 3.9 μm) and Thermal InfraRed (TIR) (at 12.0 μm).

High spatial and low temporal resolution Satellites with high spatial resolution and low temporal one are many, *e.g.*, Sentinel-1 and Sentinel-2 from ESA, Landsat-8, Landsat-9 and Terra from NASA.

Sentinel-1 (S1) and Sentinel-2 (S2) satellite polar missions are part of the Copernicus programme of the European Space Agency (ESA). The S1 constellation consists of two identical satellites both equipped with C-band Synthetic Aperture Radar (SAR) instruments (frequency 5.405 GHz, wavelength $\sim$ 5.55 cm), providing data in dual or single polarization. Also S2 consists of a constellation of two identical satellites. S1 sensors acquire images with a spatial resolution of 10 m to 40 m (according to the operational mode), in C-band Wide swath (IW) mode, with a temporal resolution of 12 days, with a 6-day exact repeat cycle at the equator, and greater at higher latitudes. S2 satellites are equipped with MultiSpectral Instrument (MSI) with 13 bands in the Visible, Near InfraRed (NIR), and ShortWave InfraRed (SWIR) part of the spectrum, and with "Aerosols", "Water vapor", "Cirrus" bands, at spatial resolutions of 10, 20 and 60 m, and they have a revisit frequency for each satellite of 10 days, resulting in a revisit frequency of 5 days for the constellation.

Landsat 8 (L8) and Landsat 9 (L9) are twin polar satellites launched in the frame of a joint National Aeronautics and Space Administration (NASA) and United States Geological Survey (USGS) mission. L8 was launched in 2013 and L9 in 2021. The two twins satellites have a temporal resolution of 16 days and carry the same two types of sensors, named Operational Land Imager (respectively, OLI for L8 and OLI-2 for L9) and Thermal InfraRed Sensor (TIRS for L8 and TIRS-2 for L9). OLI/OLI-2 are characterized by 9 spectral bands, in the Visible, Near InfraRed and ShortWave InfraRed, and a "Coastal aerosol" band (spatial resolution of 30 m), and with a Panchromatic band and a "Cirrus" band at a spatial resolution of 15 m; TIRS/TIRS-2) have 2 bands of 100 m spatial resolution.

Finally, the NASA EOS AM-1 Terra polar satellite is equipped with the Advanced Spaceborne Thermal Emission and Reflection Radiometer (ASTER) sensor with high spatial, spectral and radiometric resolution. The ASTER high spatial resolution is of 15 to 90 m and the spectral resolution is of 14 spectral bands, including Visible and Near InfraRed radiation (VNIR), ShortWave InfraRed radiation (SWIR) and Thermal InfraRed radiation (TIR). The temporal resolution is 16 days.

Bibliography

Adami, Stefan, Xiangyu Y Hu, and Nikolaus A Adams (2012). “A generalized wall boundary condition for smoothed particle hydrodynamics”. In: *Journal of Computational Physics* 231.21, pp. 7057–7075. URL: https://www.sciencedirect.com/science/article/abs/pii/S002199911200229X.

Ahmed, Mohiuddin, Raihan Seraj, and Syed Mohammed Shamsul Islam (2020). “The k-means algorithm: A comprehensive survey and performance evaluation”. In: *Electronics* 9.8, p. 1295. URL: https://www.mdpi.com/2079-9292/9/8/1295.

Alexiadis, Alessio (2023). “A minimalistic approach to physics-informed machine learning using neighbour lists as physics-optimized convolutions for inverse problems involving particle systems”. In: *Journal of Computational Physics* 473, p. 111750. URL: https://www.sciencedirect.com/science/article/pii/S0021999122008130.

Amato, Eleonora (2022). “Machine learning and best fit approach to map lava flows from space”. In: *Il nuovo cimento C* 45, pp. 1–12. URL: https://www.sif.it/riviste/sif/ncc/econtents/2022/045/04/article/15.

— (2023). “How a CFD Emulator Can Resolve the Boundary Conditions in a Viscous Flow”. In: *IEICE Proceedings Series* 76.B4L-42, p. 383. URL: https://www.ieice.org/publications/proceedings/summary.php?expandable=13&iconf=NOLTA&session_num=B4L-4&number=B4L-42&year=2023.

Amato, Eleonora, Vito Zago, and Ciro Del Negro (2024). “A physically consistent AI-based SPH emulator for computational fluid dynamics”. In: *Nonlinear Engineering* 13.1, p. 20220359. URL: https://doi.org/10.1515/nleng-2022-0359.

Amato, Eleonora et al. (2021a). “Mapping lava flows at Etna Volcano using Google Earth Engine, open-access satellite data, and machine learning”. In: *2021 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*. IEEE, pp. 1–6. URL: https://ieeexplore.ieee.org/abstract/document/9591110.

— (2021b). “Mapping of lava flows from the Mount Etna 2020-2021 paroxysmal events combining machine learning and satellite remote sensing techniques”. In: *IV conferenza Rittmann Giovani Ricercatori*, pp. 1–28. URL: https://doi.org/10.17605/OSF.IO/6Q9KM.

— (2023a). “A Deep Convolutional Neural Network for Detecting Volcanic Thermal Anomalies from Satellite Images”. In: *Remote Sensing* 15.15, p. 3718.

— (2023b). “Spectral analysis of lava flows: temporal and physicochemical effects””. In: *Il Nuovo Cimento C* 46, pp. 1–4. URL: https://www.sif.it/riviste/sif/ncc/econtents/2023/046/05/article/0.

Anderson, John David and John Wendt (1995). *Computational fluid dynamics*. Vol. 206. Springer. URL: https://link.springer.com/book/10.1007/978-3-540-85056-4.

Antuono, Matteo, Andrea Colagrossi, and Salvatore Marrone (2012). “Numerical diffusive terms in weakly-compressible SPH schemes”. In: *Computer Physics Communications* 183.12, pp. 2570–2580. URL: https://www.sciencedirect.com/science/article/pii/S0010465512002342.

Arena, Paolo et al. (1994). "Neural networks for quaternion-valued function approximation". In: *Proceedings of IEEE international symposium on circuits and systems-ISCAS'94*. Vol. 6. IEEE, pp. 307–310. URL: https://ieeexplore.ieee.org/abstract/document/409587.

Basmadjian, Diran (2003). "Mathematical modeling of physical systems: an introduction". In: *(No Title)*. URL: https://www.amazon.com/Mathematical-Modeling-Physical-Systems-Introduction/dp/0195153146.

Basodi, Sunitha et al. (2020). "Gradient amplification: An efficient way to train deep neural networks". In: *Big Data Mining and Analytics* 3.3, pp. 196–207. URL: https://ieeexplore.ieee.org/abstract/document/9142152.

Batchelor, George Keith (1967). *An introduction to fluid dynamics*. Cambridge university press. URL: https://books.google.it/books?hl=it&lr=&id=Rla7OihRvUgC&oi=fnd&pg=PR18&dq=+Batchelor,+G.+K.+(1973).+An+Introduction+to+Fluid+dynamics.+Cambridge,+U.K.:+Cambridge+University+Press.+pp.+71%E2%80%9373.+ISBN+978-0-521-09817-5.+OCLC+847527173.&ots=hksZyOPVAz&sig=DNILT2KYvzy1za3G8ePIRaqcHmo&redir_esc=y#v=onepage&q&f=false.

Becker, Suzanna (1991). "Unsupervised learning procedures for neural networks". In: *International Journal of Neural Systems* 2.01n02, pp. 17–33. URL: https://www.worldscientific.com/doi/abs/10.1142/s0129065791000030.

Belgiu, Mariana and Lucian Drăguţ (2016). "Random forest in remote sensing: A review of applications and future directions". In: *ISPRS journal of photogrammetry and remote sensing* 114, pp. 24–31. URL: https://www.sciencedirect.com/science/article/abs/pii/S0924271616000265.

Bers, Lipman, Fritz John, and Martin Schechter (1964). *Partial differential equations*. American Mathematical Soc. URL: https://books.google.it/books?hl=it&lr=&id=UEVY9MmIfrYC&oi=fnd&pg=PR17&dq=P.R.+Garabedian,+%22Partial+differential+equations%22+,+Wiley+(1964)&ots=oMJ3wK4WkP&sig=GO3dSsUBxuBKIVDiFMHCymFiKXs&redir_esc=y#v=onepage&q&f=false.

Beucler, Tom et al. (2021). "Enforcing analytic constraints in neural networks emulating physical systems". In: *Physical Review Letters* 126.9, p. 098302. URL: https://journals.aps.org/prl/abstract/10.1103/PhysRevLett.126.098302.

Bilotta, Giuseppe et al. (2016). "GPUSPH: a Smoothed Particle Hydrodynamics model for the thermal and rheological evolution of lava flows". In: *Geological Society, London, Special Publications* 426.1, pp. 387–408. URL: https://www.lyellcollection.org/doi/full/10.1144/SP426.24.

Bilotta, Giuseppe et al. (2022a). "A numerically robust, parallel-friendly variant of BiCGSTAB for the semi-implicit integration of the viscous term in Smoothed Particle Hydrodynamics". In: *Journal of Computational Physics* 466, p. 111413. URL: https://www.sciencedirect.com/science/article/abs/pii/S0021999122004752.

Bilotta, Giuseppe et al. (2022b). "Fast, feature-rich weakly-compressible SPH on GPU: coding strategies and compiler choices". In: *arXiv preprint arXiv:2207.11328*. URL: https://arxiv.org/abs/2207.11328.

Blackett, Matthew (2017). "An overview of infrared remote sensing of volcanic activity". In: *Journal of Imaging* 3.2, p. 13. URL: https://www.mdpi.com/2313-433X/3/2/13.

Bonaccorso, Giuseppe (2017). *Machine learning algorithms*. Packt Publishing Ltd. URL: https://books.google.it/books?hl=it&lr=&id=_-ZDDwAAQBAJ&oi=fnd&pg=PP1&dq=bonaccorso+Machine+learning+algorithms&ots=epmzD3EE6G&sig=AnfFTQ3BOhepOnq_doejsAuxXzQ&redir_esc=y#v=onepage&q=bonaccorso%20Machine%20learning%20algorithms&f=false.

Bortnik, Jacob and Enrico Camporeale (2021). “Ten ways to apply machine learning in the Earth and space sciences”. In: *AGU Fall Meeting Abstracts*. Vol. 2021, IN12A–06. URL: https://ui.adsabs.harvard.edu/abs/2021AGUFMIN12A..06B/abstract.

Breiman, L (2001). *Random forests Mach Learn 45 (1): 5–32*.

Breiman, Leo (2017). *Classification and regression trees*. Routledge. URL: https://doi.org/10.1201/9781315139470.

Brookshaw, Leigh (1985). “A method of calculating radiative heat diffusion in particle simulations”. In: *Publications of the Astronomical Society of Australia* 6.2, pp. 207–210. URL: https://www.cambridge.org/core/journals/publications-of-the-astronomical-society-of-australia/article/abs/method-of-calculating-radiative-heat-diffusion-in-particle-simulations/8E440E5C36A2D7C5F73FDDCB5F4CF95A.

Buscarino, Arturo et al. (2023). “Hypercomplex Multilayer Perceptron for Planetary Orbits Prediction”. In: *2023 9th International Conference on Control, Decision and Information Technologies (CoDIT)*. IEEE, pp. 1427–1431. URL: https://ieeexplore.ieee.org/abstract/document/10284281.

Cai, Shengze et al. (2021). “Physics-informed neural networks (PINNs) for fluid mechanics: A review”. In: *Acta Mechanica Sinica* 37.12, pp. 1727–1738. URL: https://link.springer.com/article/10.1007/s10409-021-01148-1.

Calin, Ovidiu (2020). *Deep learning architectures*. Springer. URL: https://link.springer.com/book/10.1007/978-3-030-36721-3.

Calvari, Sonia and Harry Pinkerton (1998). “Formation of lava tubes and extensive flow field during the 1991–1993 eruption of Mount Etna”. In: *Journal of Geophysical Research: Solid Earth* 103.B11, pp. 27291–27301. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/97JB03388.

Cappello, Annalisa, Annamaria Vicari, and Ciro Del Negro (2011). “Assessment and modeling of lava flow hazard on Mt. Etna volcano”. In: *Bollettino di Geofisica Teorica e Applicata*. URL: https://www.earth-prints.org/handle/2122/7697.

Cappello, Annalisa et al. (2016). “MAGFLOW: a physics-based model for the dynamics of lava-flow emplacement”. In: *Geological Society, London, Special Publications* 426.1, pp. 357–373. URL: https://www.lyellcollection.org/doi/full/10.1144/SP426.16.

Cappello, Annalisa et al. (2019). “Changing eruptive styles at the south-east crater of Mount Etna: Implications for assessing lava flow hazards”. In: *Frontiers in Earth Science* 7, p. 213. URL: https://www.frontiersin.org/articles/10.3389/feart.2019.00213/full.

Cariello, Simona et al. (2023). “Cascading Machine Learning to Monitor Volcanic Thermal Activity Using Orbital Infrared Data: From Detection to Quantitative Evaluation”. In: *Remote Sensing* 16.1, p. 171.

Carlson, James A, Arthur Jaffe, and Andrew Wiles (2006). *The millennium prize problems*. American Mathematical Soc.

Carracedo, Juan C et al. (2022). “The 2021 eruption of the cumbre vieja volcanic ridge on la palma, canary islands”. In: *Geology Today* 38.3, pp. 94–107. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/gto.12388.

Castro, Jonathan M and Yves Feisel (2022). “Eruption of ultralow-viscosity basanite magma at Cumbre Vieja, La Palma, Canary Islands”. In: *Nature communications* 13.1, p. 3174. URL: https://www.nature.com/articles/s41467-022-30905-4.

Chang, Kao-Hua (2023). “A novel Eulerian SPH shallow water model for 2D overland flow simulations”. In: *Journal of Hydrology* 621, p. 129581. URL: https://www.sciencedirect.com/science/article/pii/S0022169423005231.

Chowdhary, KR1442 and KR Chowdhary (2020). “Natural language processing”. In: *Fundamentals of artificial intelligence*, pp. 603–649. URL: https://link.springer.com/chapter/10.1007/978-81-322-3972-7_19.

Chung, T J (2002). *Computational fluid dynamics*. Cambridge university press. URL: https://books.google.it/books?hl=it&lr=&id=yxOTSHy8NIwC&oi=fnd&pg=PR21&dq=computational+fluid+dynamics&ots=dJhkl08Syg&sig=vbiqAWu6Cf3Jqr04KGPVWT1ux8Q&redir_esc=y#v=onepage&q=computational%20fluid%20dynamics&f=false.

Cleary, Paul W and Joseph J Monaghan (1993). “Boundary interactions and transition to turbulence for standard CFD problems using SPH”. In: *Proc. of the 6th International Computational Techniques and Applications*, pp. 157–165. URL: https://books.google.it/books?hl=it&lr=&id=SQhqEAAAQBAJ&oi=fnd&pg=PA157&dq=boundary+layers+cfd+sph&ots=mJy4ObTvEn&sig=GZgovKmXTdNZjZmpO-jwfHy8JQ8&redir_esc=y#v=onepage&q=boundary%20layers%20cfd%20sph&f=false.

Clevert, Djork-Arné, Thomas Unterthiner, and Sepp Hochreiter (2015). “Fast and accurate deep network learning by exponential linear units (elus)”. In: *arXiv preprint arXiv:1511.07289*. URL: https://arxiv.org/abs/1511.07289.

Cole, Robert Hugh (1948). *Underwater Explosion*. Princeton Univ. Press. URL: https://www.science.org/doi/10.1126/science.108.2818.750.b.

Coppola, Diego et al. (2013). “Rheological control on the radiant density of active lava flows and domes”. In: *Journal of Volcanology and Geothermal Research* 249, pp. 39–48. URL: https://www.sciencedirect.com/science/article/pii/S0377027312002818.

Coppola, Diego et al. (2016). “Enhanced volcanic hot-spot detection using MODIS IR data: results from the MIROVA system”. In: *Geological Society, London, Special Publications* 426.1, pp. 181–205. URL: https://www.lyellcollection.org/doi/abs/10.1144/SP426.5.

Cordonnier, B, E Lev, and Fanny Garel (2016). “Benchmarking lava-flow models”. In: *Geological Society, London, Special Publications* 426.1, pp. 425–445. URL: https://www.lyellcollection.org/doi/abs/10.1144/sp426.7.

Corradino, Claudia et al. (2019a). “Mapping recent lava flows at Mount Etna using multispectral sentinel-2 images and machine learning techniques”. In: *Remote Sensing* 11.16, p. 1916. URL: https://www.mdpi.com/2072-4292/11/16/1916.

Corradino, Claudia et al. (2019b). “Smart decision support systems for volcanic applications”. In: *Energies* 12.7, p. 1216. URL: https://www.mdpi.com/1996-1073/12/7/1216.

Corradino, Claudia et al. (2021a). “Combining radar and optical satellite imagery with machine learning to map lava flows at Mount Etna and Fogo Island”. In: *Energies* 14.1, p. 197. URL: https://www.mdpi.com/1996-1073/14/1/197.

Corradino, Claudia et al. (2021b). “Towards an automatic generalized machine learning approach to map lava flows”. In: *2021 17th International Workshop on Cellular Nanoscale Networks and their Applications (CNNA)*. IEEE, pp. 1–4. URL: https://ieeexplore.ieee.org/abstract/document/9610813.

— (2022). “Data-Driven Random Forest Models for Detecting Volcanic Hot Spots in Sentinel-2 MSI Images”. In: *Remote Sensing* 14.17, p. 4370. URL: https://www.mdpi.com/2072-4292/14/17/4370.

Costa, Antonio and Giovanni Macedonio (2005). “Computational modeling of lava flows: A review”. In: *Special papers-Geological Society of America* 396, p. 209. URL: https://books.google.it/books?hl=it&lr=&id=efqyc-fh82YC&oi=fnd&pg=PA209&dq=A.+Costa+and+G.+Macedonio,+Computational+modelling+of+lava+flows:+a+review.+In:+M.+Manga,+G.+Ventura+(eds)+Kinematics+

and+Dynamics+of+Lava+Flows,+Geological+Society+of+America,+Special+Papers,+396,+209%E2%80%93218,+2005,+http://doi.org/10.1130/0-8137-2396-5.209&ots=owTgQifJZN&sig=Iz9COn75Gmkn75eneMg_5P8L_9M&redir_esc=y#v=onepage&q&f=false.

Crespo, AJC, M Gómez-Gesteira, Robert A Dalrymple, et al. (2007). "Boundary conditions generated by dynamic particles in SPH methods". In: *Cmc-Tech Science Press-* 5.3, p. 173. URL: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://cdn.techscience.cn/files/cmc/2007/v5n3/cmc.2007.005.173.pdf.

Dar, Zulkeefal, Joan Baiges, and Ramon Codina (2023). "Artificial neural network based correction for reduced order models in computational fluid mechanics". In: *Computer Methods in Applied Mechanics and Engineering* 415, p. 116232. URL: https://www.sciencedirect.com/science/article/abs/pii/S0045782523003560.

Dekker, Anthony H (1994). "Kohonen neural networks for optimal colour quantization". In: *Network: Computation in Neural Systems* 5.3, p. 351. URL: https://iopscience.iop.org/article/10.1088/0954-898X/5/3/003/meta.

Del Negro, Ciro, Annalisa Cappello, and Gaetana Ganci (2016). "Quantifying lava flow hazards in response to effusive eruption". In: *Bulletin* 128.5-6, pp. 752–763. URL: https://pubs.geoscienceworld.org/gsa/gsabulletin/article/128/5-6/752/126246/Quantifying-lava-flow-hazards-in-response-to.

Del Negro, Ciro et al. (2004). "Volcanomagnetic changes accompanying the onset of the 2002–2003 eruption of Mt. Etna (Italy)". In: *Earth and Planetary Science Letters* 229.1-2, pp. 1–14. URL: https://www.sciencedirect.com/science/article/abs/pii/S0012821X04006533.

Del Negro, Ciro et al. (2020). "Living at the edge of an active volcano: Risk from lava flows on Mt. Etna". In: *Bulletin* 132.7-8, pp. 1615–1625. URL: https://pubs.geoscienceworld.org/gsa/gsabulletin/article-abstract/132/7-8/1615/575769/Living-at-the-edge-of-an-active-volcano-Risk-from.

Deng, Jia et al. (2009). "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee, pp. 248–255. URL: https://ieeexplore.ieee.org/abstract/document/5206848.

Donea, Jean et al. (2004). "Arbitrary L agrangian–E ulerian Methods". In: *Encyclopedia of computational mechanics*.

Duguay, JM, RWJ Lacey, and J Gaucher (2017). "A case study of a pool and weir fishway modeled with OpenFOAM and FLOW-3D". In: *Ecological Engineering* 103, pp. 31–42. URL: https://www.sciencedirect.com/science/article/pii/S0925857417300502.

Dym, Clive (2004). *Principles of mathematical modeling*. Elsevier. URL: https://shop.elsevier.com/books/principles-of-mathematical-modeling/dym/978-0-12-226551-8?country=IT&format=print&utm_source=google_ads&utm_medium=paid_search&utm_campaign=itpmax&gclid=CjwKCAjw1t2pBhAFEiwA_-A-NPRYh1TtKduEaHiLJwn3RPpZZCCtbvd-YknF2i-vQzUbLb3AL4KSbBoC4jQQAvD_BwE&gclsrc=aw.ds.

Edwards, Dilwyn and Mike Hamson (1989). "Guide to mathematical modelling". In: *(No Title)*. URL: https://cir.nii.ac.jp/crid/1130282271326282368.

Fernández-Nieto, Enrique D et al. (2016). "A multilayer shallow model for dry granular flows with the-rheology: application to granular collapse on erodible beds". In: *Journal of Fluid Mechanics* 798, pp. 643–681. URL: https://www.cambridge.org/core/journals/journal-of-fluid-mechanics/article/abs/multilayer-shallow-model-for-dry-granular-flows-with-the-itmuirheology-application-to-granular-collapse-on-erodible-beds/8BC4D9B4095D9A7E9E45AACBE9928CEC.

Ferrand, Martin et al. (2013). “Unified semi-analytical wall boundary conditions for inviscid, laminar or turbulent flows in the meshless SPH method”. In: *International Journal for Numerical Methods in Fluids* 71.4, pp. 446–472. URL: https://onlinelibrary.wiley.com/doi/full/10.1002/fld.3666.

Ferrand, Martin et al. (2017). “Unsteady open boundaries for SPH using semi-analytical conditions and Riemann solver in 2D”. In: *Computer Physics Communications* 210, pp. 29–44. URL: https://www.sciencedirect.com/science/article/pii/S0010465516302806.

Fowler, Andrew Cadle (1997). *Mathematical models in the applied sciences*. Vol. 17. Cambridge University Press. URL: https://books.google.it/books?hl=it&lr=&id=2KeYPU78AsMC&oi=fnd&pg=PR7&dq=mathematical+modeling&ots=lUQFfsCR9K&sig=ZaTxealyMrOd1ND45k-_nDn6o1I&redir_esc=y#v=onepage&q=mathematical%20modeling&f=false.

Ganci, G et al. (2012). “A year of lava fountaining at Etna: volumes from SEVIRI”. In: *Geophysical Research Letters* 39.6. URL: https://agupubs.onlinelibrary.wiley.com/doi/full/10.1029/2012GL051026.

Genzano, Nicola, Nicola Pergola, and Francesco Marchese (2020). “A Google Earth Engine tool to investigate, map and monitor volcanic thermal anomalies at global scale by means of mid-high spatial resolution satellite data”. In: *Remote Sensing* 12.19, p. 3232. URL: https://www.mdpi.com/2072-4292/12/19/3232.

Ghahramani, Zoubin (2015). “Probabilistic machine learning and artificial intelligence”. In: *Nature* 521.7553, pp. 452–459. URL: https://www.nature.com/articles/nature14541.

Giacomelli, Lisetta and Roberto Scandone (2004). *Vulcanologia: principi fisici e metodi d'indagine*. Liguori Editore.

Gingold, Robert A and Joseph J Monaghan (1977). “Smoothed particle hydrodynamics: theory and application to non-spherical stars”. In: *Monthly notices of the royal astronomical society* 181.3, pp. 375–389. URL: https://academic.oup.com/mnras/article/181/3/375/988212.

Giordano, Daniele and D Dingwell (2003). “Viscosity of hydrous Etna basalt: implications for Plinian-style basaltic eruptions”. In: *Bulletin of Volcanology* 65, pp. 8–14. URL: https://link.springer.com/article/10.1007/s00445-002-0233-2.

Goodfellow, Ian (2016). “Nips 2016 tutorial: Generative adversarial networks”. In: *arXiv preprint arXiv:1701.00160*. URL: https://arxiv.org/abs/1701.00160.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep learning*. MIT press. URL: https://books.google.it/books?hl=it&lr=&id=omivDQAAQBAJ&oi=fnd&pg=PR5&dq=goodfellow+deep+learning&ots=MNV-gsnGUY&sig=Khu3IM_Mwfa4DYpnfX-JKurVh64&redir_esc=y#v=onepage&q=goodfellow%20deep%20learning&f=false.

Goodfellow, Ian et al. (2014). “Generative adversarial nets”. In: *Advances in neural information processing systems* 27. URL: https://proceedings.neurips.cc/paper_files/paper/2014/hash/5ca3e9b122f61f8f06494c97b1afccf3-Abstract.html.

Gorelick, Noel et al. (2017). “Google Earth Engine: Planetary-scale geospatial analysis for everyone”. In: *Remote sensing of Environment* 202, pp. 18–27. URL: https://www.sciencedirect.com/science/article/pii/S0034425717302900.

Graw, JH, WT Wood, and BJ Phrampus (2021). “Predicting global marine sediment density using the random forest regressor machine learning algorithm”. In: *Journal of Geophysical Research: Solid Earth* 126.1, e2020JB020135. URL: https://agupubs.onlinelibrary.wiley.com/doi/full/10.1029/2020JB020135.

Greiner, Walter, Joachim A Maruhn, et al. (1996). *Nuclear models*. Springer. URL: https://link.springer.com/book/10.1007/978-3-642-60970-1.

Groeneveld, DC et al. (2007). “The 2006 CHF look-up table”. In: *Nuclear engineering and design* 237.15-17, pp. 1909–1922. URL: https://www.sciencedirect.com/science/article/abs/pii/S0029549307002002.

Harris, Andrew (2013). *Thermal remote sensing of active volcanoes: a user's manual*. Cambridge university press. URL: https://books.google.it/books?hl=it&lr=&id=xY4oYzbHOooC&oi=fnd&pg=PR9&dq=Thermal+Remote+Sensing+of+Active+Volcanoes+A+User%27s+Manual+Andrew+Harris&ots=THS4BlGObc&sig=oh5BYRTjR4AYSDTvR4HBSsDvIsO&redir_esc=y#v=onepage&q=Thermal%20Remote%20Sensing%20of%20Active%20Volcanoes%20A%20User's%20Manual%20Andrew%20Harris&f=false.

Harris, Andrew J and S Rowland (2001). “FLOWGO: a kinematic thermo-rheological model for lava flowing in a channel”. In: *Bulletin of Volcanology* 63, pp. 20–44. URL: https://link.springer.com/article/10.1007/s004450000120.

Head, Elisabet M, Ann L Maclean, and Simon A Carn (2013). “Mapping lava flows from Nyamuragira volcano (1967–2011) with satellite data and automated classification methods”. In: *Geomatics, Natural Hazards and Risk* 4.2, pp. 119–144. URL: https://www.tandfonline.com/doi/full/10.1080/19475705.2012.680503.

Heap, MJ et al. (2014). “Microstructural controls on the physical and mechanical properties of edifice-forming andesites at Volcán de Colima, Mexico”. In: *Journal of Geophysical Research: Solid Earth* 119.4, pp. 2925–2963. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1002/2013jb010521.

Hecht-Nielsen, Robert (1992). “Theory of the backpropagation neural network”. In: *Neural networks for perception*. Elsevier, pp. 65–93. URL: https://www.sciencedirect.com/science/article/abs/pii/B9780127412528500108.

Hérault, Alexis (2008). “Création d'un système d'information pour la gestion des risques volcaniques”. PhD thesis. Paris Est. URL: https://www.theses.fr/2008PEST0229.

Hérault, Alexis et al. (2011). “Numerical simulation of lava flow using a GPU SPH model”. In: *Annals of Geophysics*. URL: https://www.earth-prints.org/handle/2122/7604.

Hirt, CW (1974). “An arbitrary Lagrangean-Eulerian computing method for all flow speeds”. In: *J. Comput. Phys.* 14, p. 203.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). “Long short-term memory”. In: *Neural computation* 9.8, pp. 1735–1780. URL: https://ieeexplore.ieee.org/abstract/document/6795963.

Hornik, Kurt, Maxwell Stinchcombe, and Halbert White (1989). “Multilayer feedforward networks are universal approximators”. In: *Neural networks* 2.5, pp. 359–366. URL: https://www.sciencedirect.com/science/article/abs/pii/0893608089900208.

Hush, Don R and Bill G Horne (1993). “Progress in supervised neural networks”. In: *IEEE signal processing magazine* 10.1, pp. 8–39. URL: https://ieeexplore.ieee.org/abstract/document/180705.

Iandola, Forrest N et al. (2016). “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size”. In: *arXiv preprint arXiv:1602.07360*. URL: https://arxiv.org/abs/1602.07360.

James, Gareth et al. (2013). *An introduction to statistical learning*. Vol. 112. Springer. URL: https://link.springer.com/book/10.1007/978-1-0716-1418-1.

Jasak, Hrvoje, Aleksandar Jemcov, Zeljko Tukovic, et al. (2007). “OpenFOAM: A C++ library for complex physics simulations”. In: *International workshop on coupled methods in numerical dynamics*. Vol. 1000, pp. 1–20.

Kasim, Muhammad Firmansyah et al. (2021). “Building high accuracy emulators for scientific simulations with deep neural architecture search”. In: *Machine Learning: Science and Technology* 3.1, p. 015013. URL: https://iopscience.iop.org/article/10.1088/2632-2153/ac3ffa/meta.

Kelfoun, Karim and Silvia Vallejo Vargas (2016). “VolcFlow capabilities and potential development for the simulation of lava flows”. In: *Geological Society, London, Special Publications* 426.1, pp. 337–343. URL: https://www.lyellcollection.org/doi/full/10.1144/SP426.8.

Kim, Byungsoo et al. (2019). “Deep fluids: A generative network for parameterized fluid simulations”. In: *Computer graphics forum*. Vol. 38. 2. Wiley Online Library, pp. 59–70. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13619.

Kochkov, Dmitrii et al. (2021). “Machine learning–accelerated computational fluid dynamics”. In: *Proceedings of the National Academy of Sciences* 118.21, e2101784118. URL: https://www.pnas.org/doi/abs/10.1073/pnas.2101784118.

Kramer, Mark A (1991). “Nonlinear principal component analysis using autoassociative neural networks”. In: *AIChE journal* 37.2, pp. 233–243.

— (1992). “Autoassociative neural networks”. In: *Computers & chemical engineering* 16.4, pp. 313–328. URL: https://www.sciencedirect.com/science/article/abs/pii/009813549280051A.

Ladickỳ, L’ubor et al. (2015). “Data-driven fluid simulations using regression forests”. In: *ACM Transactions on Graphics (TOG)* 34.6, pp. 1–9. URL: https://dl.acm.org/doi/abs/10.1145/2816795.2818129.

Lamb, Horace (1994[1932]). *Hydrodynamics (6th ed.)* Cambridge university press. URL: https://books.google.com.ag/books?id=2grlAQAACAAJ&printsec=copyright#v=onepage&q&f=false.

Lassila, Toni et al. (2014). “Model order reduction in fluid dynamics: challenges and perspectives”. In: *Reduced Order Methods for modeling and computational reduction*, pp. 235–273. URL: https://link.springer.com/chapter/10.1007/978-3-319-02090-7_9.

Le Bas, MJ (2000). “IUGS reclassification of the high-Mg and picritic volcanic rocks”. In: *Journal of Petrology* 41.10, pp. 1467–1470. URL: https://academic.oup.com/petrology/article/41/10/1467/1412827.

Legay, Antoine, Jack Chessa, and Ted Belytschko (2006). “An Eulerian–Lagrangian method for fluid–structure interaction based on level sets”. In: *Computer Methods in Applied Mechanics and Engineering* 195.17-18, pp. 2070–2087. URL: https://www.sciencedirect.com/science/article/pii/S0045782505001982.

Lemarié-Rieusset, Pierre Gilles (2018). *The Navier-Stokes problem in the 21st century*. CRC press. URL: https://books.google.it/books?hl=it&lr=&id=bSCyDwAAQBAJ&oi=fnd&pg=PP1&dq=Millennium+Problems+2000+navier+stokes&ots=lpNRZY9GA9&sig=jD9WsY83Su0EWWVxejqzK9SPaLM&redir_esc=y#v=onepage&q=Millennium%20Problems%202000%20navier%20stokes&f=false.

Li, Long et al. (2015). “Impact of environmental factors on the spectral characteristics of lava surfaces: Field spectrometry of basaltic lava flows on Tenerife, Canary Islands, Spain”. In: *Remote Sensing* 7.12, pp. 16986–17012. URL: https://www.mdpi.com/2072-4292/7/12/16986.

Li, Xin et al. (2023). “Big Data in Earth system science and progress towards a digital twin”. In: *Nature Reviews Earth & Environment* 4, pp. 1–14. URL: https://www.nature.com/articles/s43017-023-00409-w.

Li, Zewen et al. (2021). “A survey of convolutional neural networks: analysis, applications, and prospects”. In: *IEEE transactions on neural networks and learning systems*. URL: https://ieeexplore.ieee.org/abstract/document/9451544.

Li, Zijie and Amir Barati Farimani (2022). “Graph neural network-accelerated Lagrangian fluid simulation”. In: *Computers & Graphics* 103, pp. 201–211. URL: https://www.sciencedirect.com/science/article/pii/S0097849322000206.

Lin, Tianyang et al. (2022). “A survey of transformers”. In: *AI Open*. URL: https://www.sciencedirect.com/science/article/pii/S2666651022000146.

Lin, Weiwei et al. (2017). “An ensemble random forest algorithm for insurance big data analysis”. In: *Ieee access* 5, pp. 16568–16575. URL: https://ieeexplore.ieee.org/abstract/document/8007210.

Lind, Steven J and PK Stansby (2016). “High-order Eulerian incompressible smoothed particle hydrodynamics with transition to Lagrangian free-surface motion”. In: *Journal of Computational Physics* 326, pp. 290–311. URL: https://www.sciencedirect.com/science/article/pii/S0021999116304041.

Liu, Gui-Rong and Moubin B Liu (2003). *Smoothed particle hydrodynamics: a meshfree particle method*. World scientific. URL: https://books.google.it/books?hl=it&lr=&id=_cwFMmEQvZQC&oi=fnd&pg=PR7&dq=liu+and+liu+Smoothed+Particle+Hydrodynamics+A+Meshfree+Particle+Method&ots=QFzACl SUaU&sig=eaRNmMzg0BsTwFlTgJKO_Zu-xtI&redir_esc=y#v=onepage&q=liu%20and%20liu%20Smoothed%20Particle%20Hydrodynamics%20A%20Meshfree%20Particle%20Method&f=false.

Lucy, Leon B (1977). “A numerical approach to the testing of the fission hypothesis”. In: *Astronomical Journal, vol. 82, Dec. 1977, p. 1013-1024.* 82, pp. 1013–1024. URL: https://adsabs.harvard.edu/full/1977AJ.....82.1013L7.

Macedonio, Giovanni, Antonio Costa, and Antonella Longo (2005). “A computer model for volcanic ash fallout and assessment of subsequent hazard”. In: *Computers & Geosciences* 31.7, pp. 837–845. URL: https://www.sciencedirect.com/science/article/pii/S0098300405000269.

Macia Lang, Fabricio et al. (2011). “Benefits of using a Wendland kernel for free-surface flows”. In.

Magnall, Nathan et al. (2019). “The origin and evolution of breakouts in a cooling-limited rhyolite lava flow”. In: *GSA Bulletin* 131.1-2, pp. 137–154. URL: https://pubs.geoscienceworld.org/gsa/gsabulletin/article/131/1-2/137/546407/The-origin-and-evolution-of-breakouts-in-a-cooling.

Malik, Anurag et al. (2021). “Prediction of multi-scalar standardized precipitation index by using artificial intelligence and regression models”. In: *Climate* 9.2, p. 28. URL: https://www.mdpi.com/2225-1154/9/2/28.

Marchese, Francesco et al. (2019). “A multi-channel algorithm for mapping volcanic thermal anomalies by means of Sentinel-2 MSI and Landsat-8 OLI data”. In: *Remote Sensing* 11.23, p. 2876. URL: https://www.mdpi.com/2072-4292/11/23/2876.

Mavriplis, Dimitri J (1996). “Mesh generation and adaptivity for complex geometries and flows”. In: *Handbook of computational fluid mechanics*. Elsevier, pp. 417–459. URL: https://www.sciencedirect.com/science/article/abs/pii/B9780125530101500086.

McCormack, P (2012). *Physical fluid dynamics*. Elsevier. URL: https://books.google.it/books?hl=it&lr=&id=VlPPybdRX9sC&oi=fnd&pg=PP1&dq=McCormack,

+P.+(2012).+Physical+fluid+dynamics.+Elsevier&ots=gomiRroCiK&sig=JrCBx2MH4t1vg78Pr6WLnyCGFb4&redir_esc=y#v=onepage&q=McCormack%2C%20P.%20(2012).%20Physical%20fluid%20dynamics.%20Elsevier&f=false.

Medsker, Larry R and LC Jain (2001). “Recurrent neural networks”. In: *Design and Applications* 5.64-67, p. 2. URL: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://d1wqtxts1xzle7.cloudfront.net/31279335/___Recurrent_Neural_Networks_Design_And_Applicatio%28BookFi.org%29-libre.pdf?1390940527=&response-content-disposition=inline%3B+filename%3DEffects_of_Liquid_and_Encapsulated_Lacti.pdf&Expires=1700328090&Signature=D7yDV94plQkMc03kRVmzky~QQCn95DAs
BCQCU5q8sOxI~cnOkMVO4tFpYqvmlESxEMcOgAQ1HMBl73J2~pjxP-b1dpgmf5aBvuKpY1CMCJQAbMul7eugyiI
rswopHOw16zNClVJ6j3NbeP5xm6~7cZfoouKjk8VVrBnBrJmvnMWrx3Iyze5M5pJVKvzOiVxzVm3teWc4P9CphY
RDkdJtttWg__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA.

Meerschaert, Mark (2013). *Mathematical modeling*. Academic press. URL: https://books.google.it/books?hl=it&lr=&id=cI03SqP4vVgC&oi=fnd&pg=PP1&dq=mathematical+modeling&ots=DSflcli04k&sig=_7GSoS7ApqaVs9vPVl3WN6kg7Z4&redir_esc=y#v=onepage&q=mathematical%20modeling&f=false.

Melnikova, Valeriia G, Andrey S Epikhin, and Matvey V Kraposhin (2021). “The eulerian–Lagrangian approach for the numerical investigation of an acoustic field generated by a high-speed gas-droplet flow”. In: *Fluids* 6.8, p. 274. URL: https://www.mdpi.com/2311-5521/6/8/274.

Menze, Bjoern H et al. (2009). “A comparison of random forest and its Gini importance with standard chemometric methods for the feature selection and classification of spectral data”. In: *BMC bioinformatics* 10, pp. 1–16. URL: https://link.springer.com/article/10.1186/1471-2105-10-213.

Mohaghegh, Shahab D et al. (2022). “Application of artificial intelligence to computational fluid dynamics”. In: *Advances in Subsurface Data Analytics*. Elsevier, pp. 281–352. URL: https://www.sciencedirect.com/science/article/abs/pii/B9780128222959000017.

Molteni, Diego and Andrea Colagrossi (2009). “A simple procedure to improve the pressure evaluation in hydrodynamic context using the SPH”. In: *Computer Physics Communications* 180.6, pp. 861–872. URL: https://www.sciencedirect.com/science/article/abs/pii/S0010465508004219.

Monaghan, Joe J (1992). “Smoothed particle hydrodynamics”. In: *Annual review of astronomy and astrophysics* 30.1, pp. 543–574. URL: https://www.annualreviews.org/doi/abs/10.1146/annurev.aa.30.090192.002551?journalCode=astro.

— (2005). “Smoothed particle hydrodynamics”. In: *Reports on progress in physics* 68.8, p. 1703. URL: https://iopscience.iop.org/article/10.1088/0034-4885/68/8/R01/meta.

Morita, Yuki et al. (2022). “Applying Bayesian optimization with Gaussian process regression to computational fluid dynamics problems”. In: *Journal of Computational Physics* 449, p. 110788. URL: https://www.sciencedirect.com/science/article/pii/S0021999121006835.

Nasar, Abouzied MA et al. (2021). “High-order velocity and pressure wall boundary conditions in Eulerian incompressible SPH”. In: *Journal of Computational Physics* 434, p. 109793. URL: https://www.sciencedirect.com/science/article/pii/S0021999120305672.

Noble, William S (2006). “What is a support vector machine?” In: *Nature biotechnology* 24.12, pp. 1565–1567. URL: https://www.nature.com/articles/nbt1206-1565.

Oger, Guillaume et al. (2007). “An improved SPH method: Towards higher order convergence”. In: *Journal of Computational Physics* 225.2, pp. 1472–1492. URL: https://www.sciencedirect.com/science/article/pii/S0021999107000630.

Pedersen, GBM (2016). “Semi-automatic classification of glaciovolcanic landforms: An object-based mapping approach based on geomorphometry”. In: *Journal of Volcanology and Geothermal Research* 311, pp. 29–40. URL: https://www.sciencedirect.com/science/article/pii/S0377027316000056.

Pelleg, Dan, Andrew W Moore, et al. (2000). “X-means: Extending k-means with efficient estimation of the number of clusters.” In: *Icml*. Vol. 1, pp. 727–734.

Petrovsky, Ivan Georgievich (2012). *Lectures on partial differential equations*. Courier Corporation. URL: https://books.google.it/books?hl=it&lr=&id=5x-6ntujRZ0C&oi=fnd&pg=PP1&dq=+Petrovsky,+I.+G.+(1991)+%5B1954%5D.+Lectures+on+Partial+Differential+Equations.+Translated+by+Shenitzer,+A.+(Dover+ed.).+New+York:+Interscience.+ISBN+0-486-66902-5&ots=R80xUyxR6j&sig=g_GQFTXS-9C41uQJauFbUxJ9Nj4&redir_esc=y#v=onepage&q&f=false.

Planck, Max (1901). “On the law of the energy distribution in the normal spectrum”. In: *Ann. Phys* 4.553, pp. 1–11.

Price, Daniel J (2008). “Modelling discontinuities and Kelvin–Helmholtz instabilities in SPH”. In: *Journal of Computational Physics* 227.24, pp. 10040–10057. URL: https://www.sciencedirect.com/science/article/pii/S0021999108004270.

Proietti, Cristina et al. (2009). “A quantitative approach for evaluating lava flow simulation reliability: LavaSIM code applied to the 2001 Etna eruption”. In: *Geochemistry, Geophysics, Geosystems* 10.9. URL: https://agupubs.onlinelibrary.wiley.com/doi/full/10.1029/2009GC002426.

Raissi, Maziar, Paris Perdikaris, and George E Karniadakis (2019). “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations”. In: *Journal of Computational physics* 378, pp. 686–707.

Rakhsha, Milad, Christopher E Kees, and Dan Negrut (2021). “Lagrangian vs. Eulerian: An analysis of two solution methods for free-surface flows and fluid solid interaction problems”. In: *Fluids* 6.12, p. 460. URL: https://www.mdpi.com/2311-5521/6/12/460.

Rogers, Benedict D et al. (2020). “DualSPHysics: an open-source code for engineering purposes”. In: *2020 SPHERIC Harbin International Workshop*, pp. 1–7. URL: https://upcommons.upc.edu/handle/2117/182712.

Rogic, Nikola et al. (2022). “The impact of dynamic emissivity–temperature trends on spaceborne data: Applications to the 2001 mount Etna eruption”. In: *Remote Sensing* 14.7, p. 1641. URL: https://www.mdpi.com/2072-4292/14/7/1641.

Rongo, Rocco et al. (2016). “SCIARA: cellular automata lava flow modelling and applications in hazard prediction and mitigation”. In: *Geological Society, London, Special Publications* 426.1, pp. 345–356. URL: https://www.lyellcollection.org/doi/full/10.1144/SP426.22.

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). “U-net: Convolutional networks for biomedical image segmentation”. In: *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer, pp. 234–241. URL: https://link.springer.com/chapter/10.1007/978-3-319-24574-4_28.

Sadourny, Robert (1975). “The dynamics of finite-difference models of the shallow-water equations”. In: *Journal of Atmospheric Sciences* 32.4, pp. 680–689. URL: https://journals.ametsoc.org/view/journals/atsc/32/4/1520-0469_1975_032_0680_tdofdm_2_0_co_2.xml.

Saikali, Elie et al. (2020). "Accuracy improvements for single precision implementations of the SPH method". In: *International Journal of Computational Fluid Dynamics* 34.10, pp. 774–787. URL: https://www.tandfonline.com/doi/abs/10.1080/10618562.2020.1836357.

Sanchez-Gonzalez, Alvaro et al. (2020). "Learning to simulate complex physics with graph networks". In: *International conference on machine learning*. PMLR, pp. 8459–8468. URL: https://proceedings.mlr.press/v119/sanchez-gonzalez20a.html.

Saramito, Pierre (2022). "Efficient C++ finite element computing with Rheolef". In.

Savov, Ivan P, James F Luhr, and Carlos Navarro-Ochoa (2008). "Petrology and geochemistry of lava and ash erupted from Volcán Colima, Mexico, during 1998–2005". In: *Journal of Volcanology and Geothermal Research* 174.4, pp. 241–256. URL: https://www.sciencedirect.com/science/article/abs/pii/S0377027308000851.

Schiff, Joel L (2011). *Cellular automata: a discrete view of the world*. John Wiley & Sons. URL: https://books.google.it/books?hl=it&lr=&id=uXJC2C2sRbIC&oi=fnd&pg=PR7&dq=Schiff,+Joel+L.+(2011).+Cellular+Automata:+A+Discrete+View+of+the+World.+Wiley+%26+Sons,+Inc.+p.+40.+ISBN+9781118030639&ots=56zJWlyvxp&sig=g0YsKmSYao1W36oj10KHd08yszM&redir_esc=y#v=onepage&q=Schiff%2C%20Joel%20L.%20(2011).%20Cellular%20Automata%3A%20A%20Discrete%20View%20of%20the%20World.%20Wiley%20%26%20Sons%2C%20Inc.%20p.%2040.%20ISBN%209781118030639&f=false.

Schlichting, Hermann and Klaus Gersten (2016). *Boundary-layer theory*. springer. URL: https://books.google.it/books?hl=it&lr=&id=bOUyDQAAQBAJ&oi=fnd&pg=PR7&dq=Schlichting,+Hermann%3B+Gersten,+Klaus+(2017).+%222.1+Boundary%E2%80%93Layer+Concept%22.+Boundary-Layer+theory+(Ninth+ed.).+Berlin+Heidelberg:+Springer.&ots=tr9bJrGSmM&sig=4x7fHWa-I53e0omQrPazunN9rCA&redir_esc=y#v=onepage&q&f=false.

Scifoni, S et al. (2010). "Mitigation of lava flow invasion hazard through optimized barrier configuration aided by numerical simulation: The case of the 2001 Etna eruption". In: *Journal of Volcanology and Geothermal Research* 192.1-2, pp. 16–26. URL: https://www.sciencedirect.com/science/article/pii/S0377027310000363.

Shadloo, M Safdari, G Oger, and David Le Touzé (2016). "Smoothed particle hydrodynamics method for fluid flows, towards industrial applications: Motivations, current state, and challenges". In: *Computers & Fluids* 136, pp. 11–34. URL: https://www.sciencedirect.com/science/article/pii/S0045793016301773.

Shaik, Anjaneyulu Babu and Sujatha Srinivasan (2019). "A brief survey on random forest ensembles in classification model". In: *International Conference on Innovative Computing and Communications: Proceedings of ICICC 2018, Volume 2*. Springer, pp. 253–260. URL: https://link.springer.com/chapter/10.1007/978-981-13-2354-6_27.

Sigurdsson, Haraldur et al. (1999). *Encyclopedia of volcanoes*. Academic press. URL: https://books.google.it/books?hl=it&lr=&id=02xTBVaPn50C&oi=fnd&pg=PP1&dq=+The+encyclopedia+of+volcanoes&ots=7NLQvnCKe4&sig=nzKn4vczgwy1kouy0farPWeHiUs&redir_esc=y#v=onepage&q=The%20encyclopedia%20of%20volcanoes&f=false.

Sofos, Filippos et al. (2022). "Current trends in fluid research in the era of artificial intelligence: A review". In: *Fluids* 7.3, p. 116. URL: https://www.mdpi.com/2311-5521/7/3/116.

Spinetti, Claudia et al. (2009). "Spectral properties of volcanic materials from hyperspectral field and satellite data compared with LiDAR data at Mt. Etna". In: *International journal of applied earth observation and geoinformation* 11.2, pp. 142–

155. URL: https://www.sciencedirect.com/science/article/abs/pii/S0303243409000038.

Temam, Roger (2001). *Navier-Stokes equations: theory and numerical analysis*. Vol. 343. American Mathematical Soc. URL: https://books.google.it/books?hl=it&lr=&id=CKYJBAAAQBAJ&oi=fnd&pg=PR9&dq=navier+stokes+equations&ots=0rzJPPfXLl&sig=k7ND4fIxuCywJoGDaNBVFMgTwHs&redir_esc=y#v=onepage&q=navier%20stokes%20equations&f=false.

Tobias Pfaff Meire Fortunato, Alvaro Sanchez-Gonzalez Peter W. Battaglia (2021). "Learning mesh-based simulation with graph networks". In: *ArXiv*. URL: https://arxiv.org/abs/2010.03409.

Torricelli, Evangelista (1644). *Opera geometrica Evangelistae Torricellii*. Florentiae: Masse and de Landis. URL: http://eudml.org/doc/203881.

Torrisi, Federica et al. (2022a). "Characterization of Volcanic Cloud Components Using Machine Learning Techniques and SEVIRI Infrared Images". In: *Sensors* 22.20, p. 7712. URL: https://www.mdpi.com/1424-8220/22/20/7712.

Torrisi, Federica et al. (2022b). "The FastVRP automatic platform for the thermal monitoring of volcanic activity using VIIRS and SLSTR sensors: FastFRP to monitor volcanic radiative power". In: *Annals of Geophysics*. URL: https://www.earth-prints.org/handle/2122/16394.

Ummenhofer, Benjamin et al. (2019). "Lagrangian fluid simulation with continuous convolutions". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=B1lDoJSYDH.

Vacondio, Renato et al. (2021). "Grand challenges for Smoothed Particle Hydrodynamics numerical schemes". In: *Computational Particle Mechanics* 8, pp. 575–588. URL: https://link.springer.com/article/10.1007/s40571-020-00354-1.

Vaswani, Ashish et al. (2017). "Attention is all you need". In: *Advances in neural information processing systems* 30. URL: https://proceedings.neurips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Velten, Kai (2009). *Mathematical modeling and simulation: introduction for scientists and engineers*. John Wiley & Sons. URL: https://books.google.it/books?hl=it&lr=&id=_B1fbVnM5uQC&oi=fnd&pg=PR5&dq=Mathematical+Modeling+and+Simulation+Introduction+for+Scientists+and+Engineers&ots=AxORH_lRw0&sig=t8cjmv3ADeKjm9DYeBg9FM6_tzw&redir_esc=y#v=onepage&q=Mathematical%20Modeling%20and%20Simulation%20Introduction%20for%20Scientists%20and%20Engineers&f=false.

Vicari, Annamaria et al. (2011). "LAV@ HAZARD: a web-GIS interface for volcanic hazard assessment". In: *Annals of Geophysics*. URL: https://www.earth-prints.org/handle/2122/7593.

Vila, Jean Paul (1999). "On particle weighted methods and smooth particle hydrodynamics". In: *Mathematical models and methods in applied sciences* 9.02, pp. 161–209. URL: https://www.worldscientific.com/doi/abs/10.1142/S0218202599000117.

Vinuesa, Ricardo and Steven L Brunton (2022). "Enhancing computational fluid dynamics with machine learning". In: *Nature Computational Science* 2.6, pp. 358–366. URL: https://www.nature.com/articles/s43588-022-00264-7.

Wang, Sun-Chong and Sun-Chong Wang (2003). "Artificial neural network". In: *Interdisciplinary computing in java programming*, pp. 81–100. URL: https://link.springer.com/chapter/10.1007/978-1-4615-0377-4_5.

Wendland, Holger (1995). "Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree". In: *Advances in computational Mathematics* 4, pp. 389–396. URL: https://link.springer.com/article/10.1007/BF02123482.

Wiering, Marco A and Martijn Van Otterlo (2012). "Reinforcement learning". In: *Adaptation, learning, and optimization* 12.3, p. 729. URL: https://link.springer.com/book/10.1007/978-3-642-27645-3.

Winston, Patrick Henry (1984). *Artificial intelligence*. Addison-Wesley Longman Publishing Co., Inc. URL: https://dl.acm.org/doi/abs/10.5555/30424.

Wooster, Martin J et al. (2005). "Retrieval of biomass combustion rates and totals from fire radiative power observations: FRP derivation and calibration relationships between biomass consumption and fire radiative energy release". In: *Journal of Geophysical Research: Atmospheres* 110.D24. URL: https://agupubs.onlinelibrary.wiley.com/doi/full/10.1029/2005JD006318.

Wooster, MJ, B Zhukov, and D Oertel (2003). "Fire radiative energy for quantitative study of biomass burning: Derivation from the BIRD experimental satellite and comparison to MODIS fire products". In: *Remote Sensing of Environment* 86.1, pp. 83–107. URL: https://www.sciencedirect.com/science/article/abs/pii/S0034425703000701.

Xu, Feiyu et al. (2019). "Explainable AI: A brief survey on history, research areas, approaches and challenges". In: *Natural Language Processing and Chinese Computing: 8th CCF International Conference, NLPCC 2019, Dunhuang, China, October 9–14, 2019, Proceedings, Part II 8*. Springer, pp. 563–574. URL: https://link.springer.com/chapter/10.1007/978-3-030-32236-6_51.

Xu, Rui and Donald Wunsch (2005). "Survey of clustering algorithms". In: *IEEE Transactions on neural networks* 16.3, pp. 645–678. URL: https://ieeexplore.ieee.org/abstract/document/1427769.

Yang, Qiang et al. (2020). *Transfer learning*. Cambridge University Press. URL: https://books.google.it/books?hl=it&lr=&id=dG_IDwAAQBAJ&oi=fnd&pg=PR9&dq=Yang,+Q.%3B+Zhang,+Y.%3B+Dai,W.%3B+Pan,+S.J.+Transfer+Learning%3B+Cambridge+University+Press:+Cambridge,+UK,+2020&ots=Ud01chPQxQ&sig=_idFkInkjWP_FzKCqryFVEnA-Bs&redir_esc=y#v=onepage&q&f=false.

Young, Philippa and Geoff Wadge (1990). "FLOWFRONT: simulation of a lava flow". In: *Computers & Geosciences* 16.8, pp. 1171–1191. URL: https://www.sciencedirect.com/science/article/pii/009830049090055X.

Zago, V et al. (2023a). "Characterization and modeling of greenwater overtopping of a sea-level deck". In: *Ocean Engineering* 275, p. 114131. URL: https://www.sciencedirect.com/science/article/pii/S0029801823005152.

Zago, Vito (2019). "Smoothed Particle Hydrodynamics method and flow dynamics: the case of lava numerical modeling and simulation". In: URL: http://dspace.unict.it/handle/10761/4145.

Zago, Vito et al. (2017). "Simulating complex fluids with smoothed particle hydrodynamics". In: *Annals of Geophysics*. URL: https://www.earth-prints.org/handle/2122/11282.

Zago, Vito et al. (2018). "Semi-implicit 3D SPH on GPU for lava flows". In: *Journal of computational physics* 375, pp. 854–870. URL: https://www.sciencedirect.com/science/article/abs/pii/S002199911830593X.

Zago, Vito et al. (2019). "Preliminary validation of lava benchmark tests on the GPUSPH particle engine". In: *Annals of Geophysics* 62(2). URL: https://www.earth-prints.org/handle/2122/12526.

Zago, Vito et al. (2021). “Overcoming excessive numerical dissipation in SPH modeling of water waves”. In: *Coastal Engineering* 170, p. 104018. URL: https://www.sciencedirect.com/science/article/abs/pii/S0378383921001666.

Zago, Vito et al. (2023b). *On Artificial Intelligence-based emulators of physical models to forecast the evolution of lava flows*. Tech. rep. Copernicus Meetings. URL: https://meetingorganizer.copernicus.org/EGU23/EGU23-16305.html.

Zerrouki, Nabil and Djamel Bouchaffra (2014). “Pixel-based or object-based: Which approach is more appropriate for remote sensing image classification?” In: *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, pp. 864–869. URL: https://ieeexplore.ieee.org/abstract/document/6974020.

Zhao, Jingbo et al. (2012). “Color segmentation on FPGA using minimum distance classifier for automatic road sign detection”. In: *2012 IEEE International Conference on Imaging Systems and Techniques Proceedings*. IEEE, pp. 516–521. URL: https://ieeexplore.ieee.org/abstract/document/6295528.

Zhu, H, YD Kim, and D De Kee (2005). “Non-Newtonian fluids with a yield stress”. In: *Journal of Non-Newtonian Fluid Mechanics* 129.3, pp. 177–181.

Zierenberg, Robert A et al. (2021). “Active Basalt Alteration at Supercritical Conditions in a Seawater-Recharged Hydrothermal System: IDDP-2 Drill Hole, Reykjanes, Iceland”. In: *Geochemistry, Geophysics, Geosystems* 22.11, e2021GC009747. URL: https://agupubs.onlinelibrary.wiley.com/doi/full/10.1029/2021GC009747.