# Physics of Dense Emulsions via High-Performance Fully Resolved Simulations

Ivan Girotto

Thursday 8th July, 2021

This work has been supported by the Eindhoven University of Technology, by the University of Modena and Reggio Emilia, and by the Abdus Salam, the International Centre for Theoretical Physics. Computer resources were made available by PRACE for the granted project (ID: 2018184340 & 2019204899) "TurEmu - The physics of (turbulent) emulsions", allocated at CINECA, Italy, and the Barcelona Supercomputing Centre (BSC), Spain, as well as by SURFsara, the Netherlands.

Eindhoven University of Technology
Department of Applied Physics
Research group: Turbulence and Vortex Dynamics
PO Box 513
5600 MB Eindhoven, The Netherlands


Cover design by Paul Verspaget
Printed by ADC Dereumaux, The Netherlands

A catalogue record is available from the Eindhoven University of Technology Library
ISBN: 978-90-386-5323-5

Physics of Dense Emulsions via High-Performance Fully Resolved Simulations / by Girotto Ivan. – Eindhoven: Technische Universiteit Eindhoven, 2021. – Proefschrift.

# Physics of Dense Emulsions via High-Performance Fully Resolved Simulations

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de
Technische Universiteit Eindhoven, op gezag van de
rector magnificus, prof.dr.ir. F.P.T. Baaijens, voor een
commissie aangewezen door het College voor
Promoties in het openbaar te verdedigen
op donderdag 8 juli 2021 om 11:00 uur

door

Ivan Girotto

geboren te Ferrara, Italië

Dit proefschrift is goedgekeurd door de promotoren en de samenstelling van de promotiecommissie is als volgt:

| | |
|---|---|
| voorzitter: | prof. dr. ir. G.M.W. Kroesen |
| 1e promotor: | prof. dr. F. Toschi |
| 2e promotor: | prof. dr. L. Pareschi (University of Ferrara) |
| copromotor: | dr. S.F. Schifano (University of Ferrara) |
| leden: | prof.dr. A. Soldati (Vienna University of Technology) |
| | prof.dr.ir. P.D. Anderson |
| | prof.dr. D. Bonn (Universiteit van Amsterdam) |

Het onderzoek of ontwerp dat in dit proefschrift wordt beschreven, is uitgevoerd in overeenstemming met de TU/e Gedragscode Wetenschapsbeoefening.

**UNIVERSITA' DEGLI STUDI**
**DI MODENA E REGGIO EMILIA**

**Dottorato di ricerca in Matematica**

Ciclo XXXIII

# Physics of Dense Emulsions via High-Performance Fully Resolved Simulations

Candidato: Girotto Ivan

Primo Relatore: Prof. Federico Toschi

Secondo Relatore: Prof. Lorenzo Pareschi

Correlatore: Prof. Sebastiano Fabio Schifano

Coordinatore del Corso di Dottorato:
Prof. Cristian Giardiná

# Contents


**1. Introduction** **1**
- 1.1. Background . . . 2
- 1.2. Objectives . . . 3
- 1.3. Outline . . . 4
- 1.4. Main contributions . . . 5

**2. Basics of emulsion physics** **7**
- 2.1. Fluids description . . . 7
  - 2.1.1. Multicomponent fluids modelling . . . 9
- 2.2. Short-range forces in multicomponent fluids . . . 12
  - 2.2.1. Surface tension . . . 12
  - 2.2.2. The role of the surfactant . . . 12
  - 2.2.3. Wetting and contact angle . . . 13
- 2.3. Turbulence . . . 14
  - 2.3.1. Turbulence phenomenology . . . 15
  - 2.3.2. Turbulence in multicomponent fluids . . . 18
- 2.4. Physics of dense emulsions . . . 19
  - 2.4.1. Yield stress, viscosity and jamming . . . 20

**3. Numerical modeling** **23**
- 3.1. The Boltzmann Equation . . . 23
- 3.2. Lattice Boltzmann Method and numerical modeling of multicomponent fluids . . . 25
- 3.3. Modeling of surface tension and disjoining pressure . . . 29
- 3.4. Modeling of the chaotic stirring . . . 31

**4. The making of dense emulsions** **35**
- 4.1. Building of an emulsion . . . 35
  - 4.1.1. Reaching highly packed emulsions . . . 36
  - 4.1.2. Emulsions classification . . . 38
  - 4.1.3. Interface fragmentation . . . 40
  - 4.1.4. Nucleation . . . 44
- 4.2. Effect of the forcing magnitude . . . 47

4.3. Jammed emulsions . . . . . 49
4.3.1. Stability of emulsions . . . . . 51
4.4. When things go wrong . . . . . 54

**5. Droplets dispersion in dense emulsions** **61**
5.1. Tracking of droplets in emulsions . . . . . 61
5.2. Analysis of droplet velocities and accelerations . . . . . 68
5.3. Analysis of droplet dispersions . . . . . 71
5.4. Analysis of droplet dynamics in dense emulsions . . . . . 76

**6. High-performance LBM** **79**
6.1. Optimization of LBM based applications . . . . . 79
6.1.1. Data layouts . . . . . 80
6.1.2. Vectorization . . . . . 80
6.1.3. Fused LBM . . . . . 83
6.1.4. LBE3D and OpenACC . . . . . 84
6.1.5. Optimization of LBM on ditributed memory . . . . . 85
6.2. Results . . . . . 87
6.2.1. The Poiseuille benchmarks on the SKL and the KNL processors . . . . . 88
6.2.2. The Poiseuille benchmarks on CPU sockets of the Intel processor family . . . . . 91
6.2.3. Poiseuille benchmarks on hybrid accelerated systems . . 92
6.2.4. The multicomponent LBM on distributed multi-GPUs . 96
6.3. Analysis of Energy Efficiency . . . . . 98

**7. Conclusions and remarks** **103**
7.1. Perspective on future works . . . . . 105

**A. Computer Architectures** **107**

**Bibliography** **111**

**List of publications** **121**

**Summary** **123**

**Acknowledgement** **125**

**Curriculum Vitae** **127**

# 1. Introduction

Emulsions are important ingredients of many foods and cosmetics and are present as well in several natural processes and, as such, these are common in our daily life. Emulsions are also interesting physical systems due to their extremely rich phenomenology. Binary emulsions are composed of two immiscible fluid components, one dispersed into the other, resulting in a mixture that can be stabilised thanks to the action of surfactants. The dispersed phase is generally in the form of droplets flowing in a continuous phase, where the ratio between the volume of the droplet phase and the volume occupied by the total multicomponent fluid, indicates the concentration of the emulsion.
From dilute to highly concentrated, emulsions, are characterised by a coupling between small-scale droplets and large-scale rheology critically determining the statistical properties of the resulting fluid mixture. Emulsions are commonly, but not only, obtained via a large-scale stirring, where the hydrodynamic stresses are responsible for breaking fluid interfaces, thus producing the droplet phase. At the same time, the presence and the concentration of the droplets influences the macroscopic flow behaviour. Stabilised concentrated emulsions can display a particularly rich dynamics, and nonlinear rheological properties, ranging from complex and non-Newtonian dynamics above the yield stress [1, 2, 3] (when these complex fluids flow) to the presence of localised topological rearrangement (such as plastic events below yield stress [4]).
In the Exascale computing era [5], numerical modelling achieved a relevant and complementary role to experiments for studying the dynamics of emulsions, yet still requiring a significant amount of computational resources. Indeed, in order to model emulsions in three-dimensions, even at modest space and time resolution, the computational cost is relevant, typically because of need to resolve the interface between the different fluid components. In this project, we developed and employed highly-optimised and largely-scalable computational code, based on the multi-component Lattice Boltzmann model (LBM) [6, 7, 8], to explore the physics of emulsions beyond what is currently experimentally possible.

## 1.1. Background

We aim at studying, via fully resolved numerical simulation, the dynamics of the formation of emulsion of two immiscible fluid components in presence of a surface stabiliser. To this end, we use the LBM method, as it was already demonstrated to be a flexible and promising approach that can go well beyond the Navier-Stokes equation (NSE) and can be employed in both laminar as well as turbulent flow conditions [9]. Significant progresses in the development of multi-phase and multi-component fluid solvers have been made in the last decades [10], enabling the possibility to numerically study the physics of complex flows [11, 12] and fluids. We build on previous works [13, 14, 15], which mainly focused on the dynamics of two fluid components, however in the absence of a stabilisation mechanism, stirred into a chaotic flow regime by means of a large-scale forcing, both for small [13, 14] and large volume fractions of the dispersed phase [15]. It must be stressed that these previous studies have not considered the presence of surfactants, or other interface stabilisers, as consequence, in those studies, when two droplets touch they immediately coalesce to form a larger droplet (with a volume that equals the sum of the droplets' individual volumes). In this work, a similar chaotic flow is introduced in order to break the interfaces between the two fluids, producing smaller and smaller droplets. These droplets, however, and at variance with previous studies [13], do not immediately coalesce when colliding with each other, due to the stabilising effect of surfactants. This simple microscopic force introduces a totally new physical phenomenology at the macroscopic scales: increasing the intensity of turbulence smaller droplets will be formed, while reducing the intensity of turbulence droplets may not (immediately) grow back to a larger size, like it was the case in [13]. This hysteresis is necessary if the emulsion has to stay stable when the stirring is stopped.
The LBM is widely used in computational fluid-dynamics to describe the behaviour of fluid flows, and is commonly applied in several scientific and engineering fields of research in order to accurately model single and multiphase flows also in presence of complex boundary conditions. Furthermore, applications based on LBM are also commonly employed in order to perform large-scale simulations to study the dynamics and the behaviour of fluid and gases, when large amount of computational resources are required. In several works [16, 17, 18], the performances in term of both computing and energy consumption of LBM applications have been studied. Extending those experiences, we have fully re-engineered the numerical implementation of the LBM [19], used in previous works on a broad range of flows and fluid conditions [14, 20, 21, 22, 13], developing a highly-optimised multicomponent LBM featuring an implementation

of large-scale chaotic forcing, surface tension and disjoining pressure. The new produced code is included in a software package, namely LBE3D, at which we will frequently refer for the rest of the thesis. With the growing complexity of high-end computer architectures the combination of multithreaded programming, vectorisation and the efficient utilisation of the different levels of the memory hierarchy are still considered to be the most promising solution to achieve high-performances on last generations of x86-64 based compute platforms. On the other hand more and more mature software environments are today available to aid scientific developers exploiting hybrid accelerated architectures, currently equipping the majority of the Tier-0 systems for HPC worldwide (see top500 list, [23]). Those are key aspects to consider when approaching the optimisation of numerical code on modern architectures for high-performance scientific computing, as well as the base of the work of code optimisation presented in this thesis.

## 1.2. Objectives

It is the objective of this project to build a new computational strategy, based on the multicomponent LBM, to explore the fundamental physics of emulsions, which is known to be strongly related to the accurate description of the internal structure of the multicomponent fluid. Because we limit at considering two immiscible fluids with the same physical properties, namely viscosity and density, these emulsions are characterised by the concentration of the dispersed phase(s), the degree of polydispersity of the droplet sizes distribution (DSD), the amplitude and shape of the hydrodynamic forcing applied at the large-scale to create the mixture, and the effect of the surfactant. To begin we mostly demonstrate that our numerical model, can build dense emulsions with different concentrations, stirring amplitude and resolutions, and we make a first qualitative analysis of the resulting emulsions morphology. Secondly, we designed and developed a new algorithm for tracking the droplets in the dispersed phase. Thanks to these tools we generated a unique database of droplets' Lagrangian trajectories during the full emulsification process, even in very dense emulsions. From the data obtained, we measure both the macroscopic hydrodynamic quantities as well as the small-scale morphology and the individual dynamics of single droplets, including an accurate tracking of breakup and coalescence events.
In summary, we developed and validated a series of tools in order to try to tackle the following fundamental outstanding questions regarding the physical behaviour of concentrated emulsions:

- How are multi-component fluids emulsions produced via chaotic large-scale stirring?
- How does the chaotic stirring and the droplets concentration influence droplets dynamics at the microscopic scale?
- How does the produced emulsion flow at the macroscopic scale, as a function of externally applied stresses?

Achieving those objectives via computer simulation requires highly expensive simulations in term of computer resources. We aim to introduce highly-optimised new memory-layout that can exploit vector instructions for the data movement from and to the main memory (DRAM), which is currently fundamental to maximise the throughput in term of memory bandwidth, on top of several optimisations to reduce towards its minimum the time to solution required to simulate the processes of emulsification on large-scale facilities for HPC. We aim to analyse in detail the performance of the newly optimised codes on both high-end CPU and CPU/GPU hybrid distributed HPC systems, scaling on a large number of processing units. Moreover, we have in view to introduce computational tools for massively parallel large-scale data analysis required to explore the produced statistics. Therefore, with this experience, we aim at contribute to the more general question: can large-scale computing simulations become an efficient and complementary tool to experiments for studying the details of emulsification processes?

## 1.3. Outline

This thesis is organised as follows. In Chapter 2 we briefly introduce the foundations and the most relevant scientific challenges relative to the modeling of multi-component turbulent emulsification processes. In Chapter 3 we discuss the numerical models implemented in the LBE3D and how our approach is capable of accurately describe the relevant physics of this problem. In Chapter 4 we present qualitative and quantitative results on the process of emulsions making via high-resolution computer simulations using the LBE3D code: from the initial nucleation or interface breakup, to the stirred stationary chaotic flow and to the final phase when the external forces are turned off and a stable emulsion is achieved. This chapter also includes qualitative results on the phenomena of phase inversion that can occur during the emulsification process. In Chapter 5 we present the innovative tracking algorithm used to follow the trajectories of droplets within a chaotic, possibly turbulent, three-dimensional multi-component emulsion flow. There, we also report results

relative to the dispersion of droplets. In Chapter 6 we present and discuss the code optimisation steps that we performed in order to enable the LBE3D on EU Tier-0 system for HPC in the pre-Exascale era, including a detailed analysis of performance and energy efficiency. In Chapter 7 conclusions are drawn and an outlook for future research is presented.

## 1.4. Main contributions

In this thesis we show how to build highly packed binary emulsions, with ($\sim 80\%$) volume fraction in the dispersed droplets phase, via computer simulations, in 3-dimensions, at high-resolution (up to $1024^3$) throughout millions of LB steps. For the first time ever, we show how the final emulsions remain stable for very long time (millions of LB steps) at rest, showing a finite yield stress, above which the emulsion flows like a viscous fluid and below which the emulsion behaves as an elastic solid. Our numerical code delivers about 85% memory bandwidth efficiency on fully memory bounded kernels exploiting vectorised data copies, and high compute performances exploiting data locality, on all other kernels. Moreover, it implements optimised data exchange between processes, including a multi-threaded packing/unpacking for the exchange of non-contiguous data, resulting an order of magnitude faster than comparable versions from previous works, and demonstrating performance portability on both CPU and CPU/GPU hybrid platforms. From a first analysis of droplets' trajectories we present droplets size distributions, including probability distribution functions of velocity and accelerations, on a $10^6$ number of samples (thousands of droplets for thousands of recorded steps), results of droplet dispersion in space and velocity considering thousands of trajectories, comparing results at different volume fractions, with both the forcing on and off. We show statistics of droplets dynamics on dense emulsion, at high-resolution and during chaotic motion, considering tens of thousands of detected events of breakups and coalescence.

# 2. Basics of emulsion physics

The phenomenology of (flowing) emulsions emerges from the interplay of several physical phenomena occurring at multiple length- and time-scales. From the mixing of immiscible fluids components, with the formation of droplets, to the complex relation between the micro-scale droplets morphology, and the macroscopic hydrodynamics, several are the physical components to take into account when modeling emulsions. This chapter provides an overview of the main physical ingredients beyond the physics of emulsions, including references to literature. We discuss how to describe fluids from the atomistic to the continuum level, the foundation of multicomponent fluids modeling, and the classical phenomenology of turbulence in single component Newtonian fluids. The chapter concludes focusing on the physics of dense stabilised emulsions and the morphological characteristics of those complex physical systems.

## 2.1. Fluids description

Considering a water flow between two plates as in Figure 2.1, and now considering of moving the upper plate while keeping the lower fixed, this provokes a strain rate in the system, deforming the water in the upper level with a certain angle in respect to the basement, indicated in Figure 2.1 as, $d\beta$. On the other hand, the surface contact between the plate and the water leads to a resistance by the fluid level, namely shear stress $\tau$. The viscosity is then defined as $\nu = \tau/\gamma$, where $\gamma = d\beta/dt$. The applied force generates a difference in velocity within the fluid, for instance the fluid closer to the plate is expected to move with an equal velocity of the plate, while the fluid near to the lower plate should be resting. This velocity variation along the distance $l$ between the two plates is referred also as velocity gradient $du/dl$. In other words, in simple Newtonian fluids the viscosity is given by the ratio between, the deformation induced by the forcing applied and the velocity gradient.
The separation of immiscible fluids is given in nature, by an attractive force between molecules of the same fluid, opposite to a repulsive potential separating molecules of the different components. Emulsions are composed by two or more immiscible fluids i.e., oil/water, like the systems we study in this thesis.

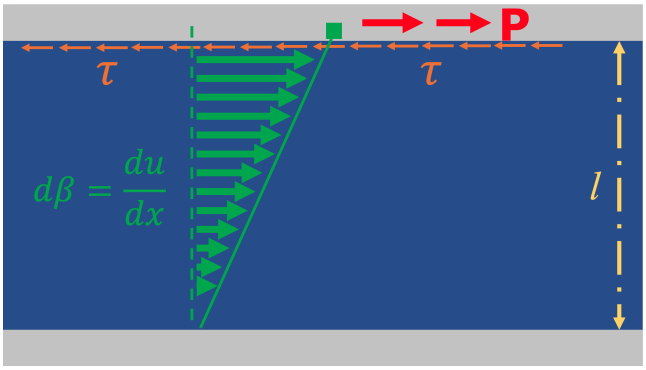



***Figure 2.1.:*** *Schematic description of the main physical properties of a two-dimensional section of a Newtonian fluid motion. They external (grey) boundaries represent two plates with a fluid (i.e., water) in between. The small green box on the above plate indicates the displacement of the top plate by applying an horizontal force. The picture schematically represents the velocity variation within the fluid, induced by the motion of the upper plate, along with the shear stress generated by the motion.*

Those chemical species do not mix as molecular polarity between the two component leads to phase separation. In a molecule of water, the potential charge (electronegativity) is given by the polar bonds between the atom of oxygen and the two atoms of hydrogen, making it a polar molecule, and therefore attracting others polarised molecules (hydrophilic). This also characterises water as a universal solvent as any hydrophilic component dissolve in water. On the other hand, the high negative charge of a water molecule prevents hydrophobic molecules to couple with it, like for the case of a fat/oil molecule, mainly composed by unpolarised bonding between carbons and hydrogens. Therefore, the molecules of each of the two separated components tend to stay each other, and in presence of gravity the less dense component will stay on the top of the heavier one as in Figure 2.2 (left). When fully separated, the two Newtonian fluids still behave as a Newtonian fluid as the influence of the surface between them may mostly be negligible. However, the physics of the emulsion becomes rapidly more complex whenever exposed to a chaotic motion, and generating droplets of the minority phase dispersed in the majority phase. The formation of droplets changes the properties of the fluid, due to the microscopic action of the forcing acting between the fluids surface, being one component dispersed into the other. This complexity rapidly increases when, as in the process of making mayonnaise at home, the component forming the dispersed phase is slowly, i.e. adiabatically added. In fact, the emulsification process of mayonnaise requires to slowly add the oil component while gently mixing

***Figure 2.2.:*** *From two simple fluids (left), to one complex fluid (right). Mayonnaise is commonly produced via the turbulent stirring of oil and water with addition of surfactants in order to stabilise the produced emulsion. In this thesis we describe how to go from the picture on the left to the picture on the right via computer simulation. We study the resulting phenomenology of emulsions like mayonnaise, including their micro- and macro-scale dynamics at rest and under flowing conditions*

the base, composed by egg yolk and water. The newly added oil generates either new droplets or being included in existing droplets that becomes bigger, forming a highly packed dispersed phase. The resulting emulsion, Figure 2.2 (right), shows non linear behaviour when subjected to hydrodynamic forces, due to the elasticity introduced by the surface tension and disjoining pressure, and droplet morphology. Therefore, the viscosity of dense emulsions does not linearly depend on the applied force, making the fluid non-Newtonian.

### 2.1.1. Multicomponent fluids modelling

There are several approaches to fluid modelling depending on the time- and length-scales considered. Every microscopic sample of liquid is composed by a huge number of molecules in constant (random) motion moving and rotating under the influence of hydrodynamic forces and thermal motions. In water, the molecular interaction is customarily modelled via the Lennard-Jones (LJ) potential [24, 25], which incorporates an attractive force that turns into a repulsive force when molecules are close to each other. The LJ potential is mathematically expressed as:

$$V_{LJ} = 4\epsilon \left[ \left( \frac{\sigma}{r} \right)^{12} - \left( \frac{\sigma}{r} \right)^{6} \right] \tag{2.1}$$

Where the long-range attractive tail has the form $-1/r^6$, the negative well has a depth $\epsilon$, and the steeply rising repulsive wall is located at distances $r \sim \sigma$. The full potential energy of the system is given by the sum of pairwise contributions from all molecules. However, in the case of multicomponent immiscible fluids, a broader range of forces must be considered for a better description of non-bonded interactions between the species, eg. in order to include hydrogen bonding. The modelling of this more complex interaction, at the atomistic scale, requires a more accurate description that better represents the total energy, if compared with the LJ potential. The OPLS force field [26], used for the simulation represented in Figure 2.3, along with many other force fields for long-range molecular interaction, are commonly available in standard packages for simulating molecular dynamics such as GROMACS [27] and LAMMPS [28]. At the molecular scale it is definitely possible to model

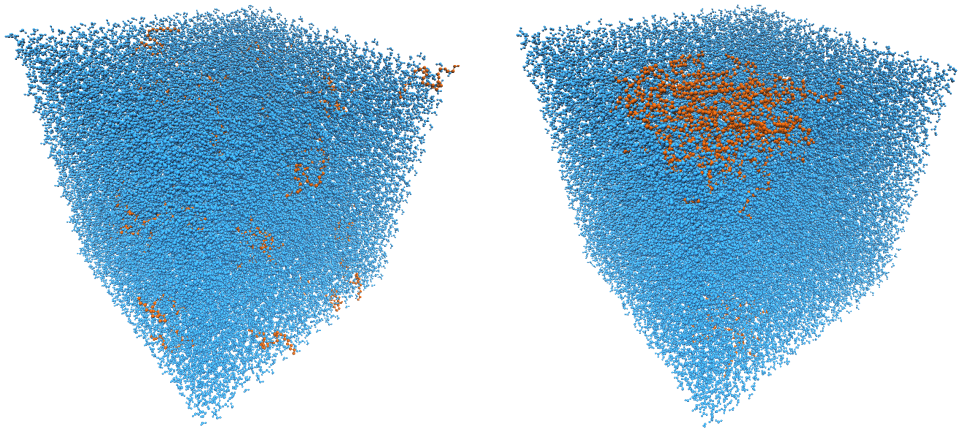


***Figure 2.3.:*** *Molecular dynamics simulation of a 10nm$^3$ box of water, including 50 C12E6 molecules randomly dispersed. From the initial configuration (left) we simulated 240ns of the evolution of an oil-water system on a single node of the Marconi-100 hosted at CINECA, achieving the final configuration on the right panel. As it can be observed, over time the initially dispersed oil molecules separate from water and assemble together forming a concentration of molecules of the same component. The simulation was performed with the GROMACS-2020 [29] code and the visualisation using the VMD [30, 31] tool. This way of modeling multicomponent immiscible fluids does not allow to computationally resolve hydrodynamics length- and time-scales.*

the interaction between different immiscible fluids, leading to the emergence of a surface between them, but modeling at this microscopic scale does not allow to answer simple questions such as: how does the resulting emulsion flow?
At the macroscopic hydrodynamic scale, fluids are considered as a continuum,

rather than as a collection of discrete molecules. This assumption at the basis of most analytical as well as computational fluid dynamics studies primarily rely on the Navier-Stokes equation (NSE), that describes the dynamics of a single component (incompressible, i.e. $\rho = 1$) viscous flow [32]:

$$\begin{aligned} \boldsymbol{\nabla} \cdot \boldsymbol{u} &= 0 \\ \partial_t \boldsymbol{u} + \boldsymbol{u} \cdot \boldsymbol{\nabla} \boldsymbol{u} &= \nu \nabla^2 \boldsymbol{u} - \boldsymbol{\nabla} p + \boldsymbol{F} \end{aligned} \tag{2.2}$$

The first equation represents the conservation of mass for an incompressible fluid, while the second describes the change of the fluid's momentum due to the pressure and dissipative forces, accordingly to the Newton's second law ($m\boldsymbol{a} = \boldsymbol{F}$), where $\boldsymbol{u}(\boldsymbol{x}, t)$ represents the velocity field of the fluid at location $\boldsymbol{x}$ and time $t$, $p$ is the pressure of the flow and $\nu$ represents the kinematic viscosity. The NSE describes how the velocity field changes in function of the viscosity of the fluid as well as the forcing applied to the fluid. The forces are given by the right hand side of the equation where the terms $\boldsymbol{\nabla} p$ and $\nu \nabla^2 \boldsymbol{u}$ describe the internal particle motion while $\boldsymbol{F}$ expresses the external forcing. The complexity of describing fluids using Navier-Stokes rapidly grows when considering multicomponent fluids. The equations for the conservation of species mass, momentum and energy in a multicomponent flow have been proposed in several forms [33, 34, 35, 36]. Here we report the set of macroscopic equations at the bases of the numerical modeling of multicomponent fluids, as implemented in the LBE3D code (see Chapter 3) and introduced in [37], and adopted for all computer simulations presented in this work, including the continuity equation of each component in separated form, in addition to an equation of motion describing the total momentum of the fluid. The equation for multiple fluid components extends the NSE (Eq. 2.2) by adding the two terms describing the fluid interaction at small-scale [38]:

$$\partial_t \rho_s + \partial_k (\rho_s u_k) = \partial_k J_{sk} \tag{2.3}$$

$$\begin{aligned} \partial_t (\rho u_i) + \partial_k (\rho u_i u_k) &= -\partial_k (c_s^2 \rho + \sigma_{ik}^{(visco)}) + \sum_s F_{sk} \\ &= -\partial_k (P_{ik} - \sigma_{ik}^{(visco)}) \end{aligned} \tag{2.4}$$

where $s$ describes the different fluid species, $\rho = \sum_s \rho_s$ is the total density, $u = \sum_s \rho_s u_s / \rho$ is the barycentric (total) fluid velocity, where $F_{sk}$ is the $k^{th}$ component of the force acting on specie $s$ and $\sigma_{ik}^{(visco)}(i, k = x, y, z)$ is the dissipative component of the momentum-flux tensor. The $J_{sk}$ term in Eq. 2.3 represents the attractive potential causing the phase separation of the two

component as given in [37]. In the following paragraph we describe the main small-scale potentials responsible for the interaction between the different fluid components, included into the $F_{sk}$ and the $J_{sk}$ terms.

## 2.2. Short-range forces in multicomponent fluids

Emulsion physics is critically characterised by the effect of microscopic short-range forces whose range and amplitude is non-trivial to determine [39]. Emulsion droplets are characterised by attraction between similar molecules and by repulsion forces due to the stabilisation mechanism (eg. presence of surfactant) necessary to stabilise the droplets against coalescence. In this paragraph we introduce the main short-range forces of emulsions.

### 2.2.1. Surface tension

Liquid molecules from same fluid tend to gather together because attracted to each other. This is the reason why a drop of oil in water assumes a perfect spherical shape, minimising the surface between the two fluids and inducing an overpressure between in the interior and the external of the drop. The presence of a surface tension gives rise to complex viscoelastic behaviour, even when each individual fluid component in the flow is itself a simple Newtonian fluid. In general, at mechanical equilibrium, the pressure on either side of fluid-fluid interfaces are different; the pressure inside being higher that outside. The pressure difference between the two component $(p_1 - p_2)$ satisfies the so-called Laplace equation [40]:

$$p_1 - p_2 = \gamma_{12} \left( \frac{1}{R_1} + \frac{1}{R_2} \right) \tag{2.5}$$

where $R_1$ and $R_2$ are the local curvature radii and $\gamma_{12}$ is the surface tension. The presence of a Laplace pressure is a consequence of the mismatch in the microscopic forces between molecules on either sides of the interface.

### 2.2.2. The role of the surfactant

Emulsion are composed of two immiscible fluids mixed together under the action of a (external) stirring force that working against the surface tension, breaks one into an other. In emulsions usually it is the minority fluid component that is dispersed, in the form of droplets into the majority phase. In absence of a stabilisation mechanism, when two droplets get in contact with each other, then

would rapidly coalesce, making emulsion unstable in nature. As consequence of successive coalescence, droplets will eventually join together to form a unique layer, once the emulsion is at rest for a sufficiently long time [41]. In other words, the fluid component will demix to completely separate as in Figure 2.2 (left). There are many cases where this process must be arrested or strongly slowed down. One common strategy is to introduce additional molecules that inhibit coalescence between droplets, forming emulsions that can remain stable for very long time. Those additional molecules, identified in chemistry as surface active agents or surfactants, are commonly derived from fat and characterised by an hydrophilic head and an hydrophobic long tail. Surfactant molecules are present in common emulsifiers utilised in food industry, such as egg yolks, mustard, milk and cream. When adding surfactant molecules, like lecithin, in water, the hydrophilic heads bonds with the water molecules of the emulsion while the hydrophobic tails bonds together. The surfactant molecules cluster in water, with an hydrophilic surface and an hydrophobic bulk. When adding instead molecules of surfactant to emulsions of oil and water, the effect of the surfactant is the same, but the surfactants hydrophobic tails now bonds with oil droplets, forming clusters with a core of oil droplets. This simple microscopic mechanism prevents oil droplets to coalesce, as the positive charge of the tail in the surfactant produces a repulsive forcing between the oil droplets, stabilising the emulsion.

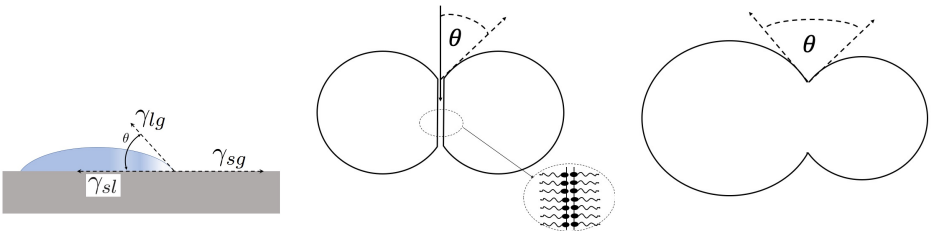
### 2.2.3. Wetting and contact angle

A liquid (water) droplet on a flat solid surface is a simple example of interaction between three phases: air, the solid substrate and the liquid. In absence of external forces, the intensity of the attractive force between the liquid particle defines how the liquid droplet will spread on the solid surface. Wetting is the description of this behaviour, characterised by the contact angle $\theta$ that forms between the liquid and solid surface (see Figure 2.4a: wetting behaviour (hydrophilic) if $\theta < 90°$, non-wetting behaviour (hydrophobic) if $\theta > 90°$. The contact angle results from the balance of forces is required in order to achieve mechanical stability at the contact line, where all three phases are in contact with each other [40, 42], (Fig. 2.4b). The balance of forces can be written as following Young's law:

$$\cos\theta = \frac{\gamma_{sg} - \gamma_{sl}}{\gamma_{lg}} \tag{2.6}$$

Here, $\gamma_{sl}$, $\gamma_{sg}$ and $\gamma_{lg}$ are, respectively, the solid-liquid, solid-gas and liquid-gas surface tensions. While the contact angle is commonly considered by the contact of different phases and defined with respect to the liquid phase, if considering

fluid mixtures there are different forms of contact angle that can be considered. In multicomponent emulsions, droplets form large contact angles as they adhere




**(a)** Contact angle **(b)** Contact angle and disjoining pressure **(c)** Contact angle in a dropelts' coalacensce

***Figure 2.4.:*** *The picture schematically reports different formations of contact angles. In 2.4a, the classical representation with a drop of liquid on a solid surface. In 2.4b and 2.4c, two different possible formation of contact angle in a multicomponent emulsion. In 2.4b, two droplets come to close contact but the presence of repulsive potential given by the surfactants avoids coalescence, creating a small contact angle ($< 90°$) between the contiguous film at the interface and the droplets surface. In 2.4c, a larger contact angle is created due to the coalescence of the two droplets.*

to one another 2.4b. However, in contrast to classical wetting, a thin liquid film of the matrix phases persists between the interfaces, stabilised by the surfactant molecules adsorbed at the interfaces. In fact, the surfactants favour the creation of this film between droplets of the same component, to arrest coalescence. This is modelled as disjoint pressure, opposed to the attractive forcing that favour the adhesion of droplets of the same component within the emulsion. In both cases those forces can be determined by the contact angle. In the case of adhesion during a droplets coalescence, Figure 2.4c, the contact angle is formed at the point of conjunction. While in the presence of surfactant, Figure 2.4c, the contact angle is formed between the film interface given by the disjoining potential of the surfactant, and the droplets of the same component characterised by attractive forcing. The different contact angle for the described cases is determined in[39].

## 2.3. Turbulence

Turbulent flows are largely present in nature as well as in our daily live, but expensive to model numerically and challenging to study experimentally too, also due to their chaotic nature, i.e. sensitivity to initial conditions, that calls for a statistical description [6]. In chaotic systems, it would requires an infinite

(numerical or experimental) accuracy to make deterministic predictions of the evolution of such flows, intrinsically unstable to perturbation, thereby making it impossible to answer simple questions like: what is the status of the flow at a given future time? An estimation of the probability that a turbulent flow finds itself in a given state can be given by making a statistical analysis on experimental data - and generally the more the data, the more precise the knowledge of the system. Another fundamental aspect of turbulence is the multi-scale and multi-time nature. When looking at thunderstorms we see clouds, which are composed by a large number of small droplets that collide with each other up to the point that, getting big and heavy enough, will cause a rainfall. This illustrates how small-scale behaviours, of droplets interacting under a chaotic flows, can influence the behaviour at the larger scale while, at the same time, the turbulence given by large-scale wind determines the behaviour of the droplets at the small-scales. Tackling those multi-scale physics problems is matter of research since decades.
In this work we introduce the concept of turbulence for two reasons: 1) the emulsions that we are going to study are characterised by an initial chaotic stirring aimed at breaking the interface between the fluids components thus generate the emulsion; 2) the numerical models that we developed and employed in this work are meant to contribute to the study of this complex flow physics. Basically, by tracking the droplets in a chaotic binary fluid we can provide a quantitative statistical measurement of observables that are key for studying the formation and flow of turbulent emulsions.

### 2.3.1. Turbulence phenomenology

Common to any irregular and chaotic flow motion, turbulence is a multi-scale problem [43, 44]. It is still considered one of the most interesting open problem in classical physics. Since the pioneering studies by Osborne Reynolds in the 1880s a large number of contributions have been given throughout the years within a wide span of disciplines. In [45], a detailed description of the main milestones of this journey is provided. At the beginning of the 19th century Taylor and Benard first realised in a laboratory that reducing the viscosity, $\nu$, of a fluid in a container, caused the transition from a regular flow to a more complex motion [44]. This variation inversely proportional to $\nu$ is related to the dimensionless Reynolds number, Re, that helps characterising how a flow qualitatively changes his behaviour: from a steady laminar flow, to a chaotic and unpredictable flow characterised by strong sensitivity to a tiny variation in its initial conditions. The Reynolds number effectively provides an estimate of the order of magnitude ratio between the inertial term $(\boldsymbol{u} \cdot \nabla \boldsymbol{u})$ and viscous

term ($\nu\nabla\boldsymbol{u}$) in the NSE (Eq. 2.2):

$$Re = \frac{LU}{\nu} \tag{2.7}$$

with $L$ and $U$ being the typical length and velocity scales of the flow, respectively [43], and $\nu$ being the kinematic viscosity of the fluid. At increasing of the Reynolds number the flow first becomes time dependent, showing periodic time behavior (i.e., via vortexes shielding), and later displays more and more frequencies up to the point of becoming totally chaotic. The NSE (Eq. 2.2) incorporates all complexity. At the increasing the Reynolds number the non-linearities of the NSE becomes dominant and flow problems cannot longer be solved analytically. Numerical approaches are, for large Reynolds numbers, fundamental tools to study the statistical properties of turbulent flows like eg. spectral properties and velocity structure function. Nevertheless, because of the high computational costs, search, for efficient methods to study the statistical properties of a turbulent flow is still somehow an open problem. Try to reduce

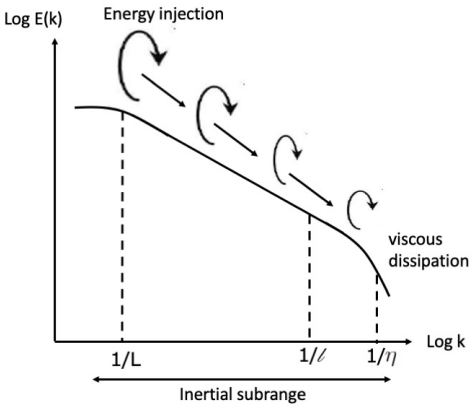



***Figure 2.5.:*** *Energy spectra with illustrated the idea of energy cascade. Indicated are also these three areas of energy scale: the integral range of scale where the force is injecting energy, the inertial range of scales and the dissipative range.*

the complexity of the multi-scale and multi-time physics turbulent models have been developed with the aim of reducing computational costs. These models however do not rest on solid theoretical grounds and cannot be employed when the focus is on the accurate resolution of the small-scale physics. From Leonardo's notes on turbulence (end of 15th century), to modern representation, turbulent flows are commonly sketched with a number of eddies randomly distributed from larger to smaller ones. In fact, turbulent flows typically show

a broad spectrum of eddy sizes, from large eddies present on the large-scale to small and small eddies dissipated at the smallest scale. Eddies in turbulent flows are distributed over a wide range of scales [46] and from their superposition one describes the flux of energy from the integral scale, where energy is injected, to the smallest dissipative scale, where kinetic energy is converted into heat due to viscous dissipation. Plotting the measured kinetic energy spectra as a function of eddies' size from large, $k = 1/L$, to small, $\eta$, illustrated the Richardson energy cascade, see Figure 2.5. The spectra can be used to illustrate the main range of scales in turbulence. The largest eddies are generated by the flow (integral or injection range, around $k = 1/L$), eddies destabilise and produce smaller eddies (inertial range of scales), this cascade process continues down to the small dissipative scale, $\eta$, where eddies are dissipated (dissipative or Kolmogorov scale). These three regions, are physically characterised by different properties. The integral scale shows non universal behaviours, dependent upon the forcing, here it is where Kinetic energy is injected into the system. The inertial range of scales is characterised by a flux from larger to small scale in a way that conserves energy that is largely universal, i.e. independent from large scale forcing and geometry. Finally, at the smallest, Kolmogorov scale, $\eta$, energy is dissipated into heat due to the viscous friction. By dimensional analysis it is possible to estimate, given the relevant time, $T$, size, $L$, and velocity, $U$, the Kinetic energy, $E$, as $U^2$ and the energy injection rate, $\epsilon$, as:

$$\epsilon_{in} \sim E/T \sim U^3/L \tag{2.8}$$

The physics of inertial range, as from Kolmogorov 1971 [47, 48, 49], states that energy is conserved as it flows downwards in scales in the inertial range. Additionally the average energy dissipation rate $\epsilon$ must equal the energy input $\epsilon_{in}$. With $\epsilon$ being a constant value, we readily obtain that:

$$\frac{\mu_e^3}{l} = \epsilon = \epsilon_{in} \tag{2.9}$$

which expresses the typical velocity $\mu_e$ of an eddy of size $l$ scales, with $l$ and $\epsilon$ as $\mu_e \sim (\epsilon l)^{1/3}$. The typical coherence time $\tau_e$ of an eddy of size $l$ (eddy turnover) is defined as $\tau_e \sim l/\mu_e \sim (l^2/\epsilon)^{1/3}$. While the smallest dissipative scale of turbulence, the Kolmogorov length scale $\eta$, where the viscosity becomes dominant on $E$, is defined as:

$$\eta \sim (U^3/\epsilon)^{1/4} \tag{2.10}$$

which allows to define the Taylor's scale Reynolds number for smaller scale as:

$$Re_\lambda = \left(\frac{L}{\lambda}\right)^{4/3} \tag{2.11}$$

where $\lambda$ is calculated with the $\mu_e'$ being the root mean square velocity as:

$$\lambda = \left(\frac{15\nu\mu_e'}{\epsilon}\right)^{1/2} \tag{2.12}$$

### 2.3.2. Turbulence in multicomponent fluids

The analytical description of fundamental properties, such as their effective viscosity $\nu_{eff}$, is largely unknown for turbulent multicomponent fluids, and mostly resting on phenomenological models or empirical collerations. Nevertheless, some exact results are available, as Einstein prediction for effective viscosity of dense emulsions. Things get much more complex when dealing with micro-scale deformation, size and dynamics of droplets, and the connection with macroscopic hydrodynamic quantities. Fluid dynamics phenomena, involving droplet dynamics, deformation, and breakup, have been the subject of many theoretical and experimental studies. Beyond their practical importance in a variety of applications [50], those quantities, describing the small-scale dynamics and morphology, are also relevant from the theoretical point of view, due to the complexity of the physics involved [51, 52, 53, 54]. At the smallest scale droplets deformation is given by the relative effect of viscous drag forces versus surface tension forces at the interface between two immiscible liquids. This deformation can be expressed as a function of the dimensionless capillary number $Ca$ defined as:

$$Ca = \frac{\nu\gamma D}{2\sigma} \tag{2.13}$$

where $\nu$ is the dynamic viscosity of the multicomponent fluid, $D$ the diametar of the initially undeformed spherical droplet, $\sigma$ the surface tension, and $\gamma$ the shear rate intensity [55]. When the droplet diameter is larger than the dissipative scale, under the action of the hydrodynamic turbulence droplets deform when $Ca > 1$, they break when larger than Hinze radius [56], and coalesce depends on density, flow and the external disjoining pressure. At this scale the level of deformation is governed by the ratio of the forces at the surface and the turbulence intensity or the Weber number, given as:

$$We = \frac{\rho^{(m)}\langle(\delta u_D)^2\rangle}{\sigma} \tag{2.14}$$

where $\rho^{(m)}$ is the density of the carrier medium fluid, $\delta u_D$ is the average velocity difference across the droplet, the angular bracket indicate spatial averaging, $D$ is the droplet diameter, and $\sigma$ is the surface tension. In the case of $We > 1$ the droplets break when the turbulent forces exceed the surface tension. Hinze [56] showed that using Kolmogorov theory for velocity differences $\langle(\delta u_D)^2\rangle\ \epsilon^{2/3}D^{2/3}$ where $\epsilon$ is the energy dissipation rate, the maximum droplet diameter that does not undergo breakup is given by:

$$D_{max} = 0.725\left(\frac{\rho^{(m)}}{\sigma}\right)^{-3/5}\epsilon^{-2/5} \tag{2.15}$$

In other words $D_{max}$ represents the maximum stable droplet size in a turbulent flow, beyond which droplets are subjects to breakup and coalescence events. Being $\epsilon$ a quantity depending upon the energy injection, it is expected that the variation of $\epsilon$ set the Hinze scale, while fixing the droplets size distribution with a peak around $D_{max}$. Combining this assumption with the turbulent phenomenology, there are different length scale governing the morphology of emulsions, around $D_{max}$:

$$L >> d >> \eta \tag{2.16}$$

with $L$ representing the hydrodynamic scale, characterised by the stirring, and $\eta$ representing the Kolmogorov scale, characterised by the viscosity $\nu$ and the dissipation rate $\epsilon$, where the range separation between the two extremes is given by the Reynolds number, Re=1.

## 2.4. Physics of dense emulsions

Emulsions exhibit different composition and statistical properties, from dilute to highly concentrated. The volume fraction is defined as the ratio between the volume occupied by the droplets in the dispersed phase and the total volume of the fluid bulk. Stabilised dense emulsions are characterised by highly packed droplets' in the small scale, causing droplets to deform from spherical shape to complicated geometrical structures. These emulsions do not longer shows characteristics of Newtonian fluids but rather properties typical of solids, displaying a rheology typical of soft-glassy materials. In this chapter we introduce dense emulsions providing relevant references at support of the presented analysis.

### 2.4.1. Yield stress, viscosity and jamming

A jammed system can resist finite small stresses without deforming in a reversible way, meaning that when the stress is removed, the system reverts back to the original shape. An unjammed system flows under any applied stresses, thus deforming irreversibly. Continuum modeling of structured fluids, in general, and emulsions in particular have revealed the importance of the so called non-local fluidity of the structured fluid, a dynamical order parameter somehow equivalent to an inverse viscosity [57]. The flow behaviour of very dense suspensions or emulsions can, if the flow is steady, be characterised by the so-called flow curve (see Figure 2.6), displaying the behaviour of the $\sigma(\dot{\gamma})$ shear stress, $\sigma$, as a function of the shear rate, $\dot{\gamma}$. A Newtonian fluid has the linear flow curve $\sigma = \eta\dot{\gamma}$ with $\eta$ being its dynamical viscosity. More generally $\eta$ is defined as the limit of the ratio $\sigma/\dot{\gamma}$ as $\dot{\gamma}$ tends to 0. Upward curvature of the flow curve is dubbed shear thickening, while a downward curvature shear thinning. A shear thinning behaviour is typically present when advection suppress fluctuations, causing the system to flow more easily at higher stresses. However, in some emulsions the opposite can occur, and this phenomena is named shear thickening behaviour [58]. If a yield stress $\sigma_Y$ is present the stress $\sigma$ tends to a finite limit $\sigma_Y$ as $\dot{\gamma}$ tends to 0, the viscosity $\eta$ then diverges. The

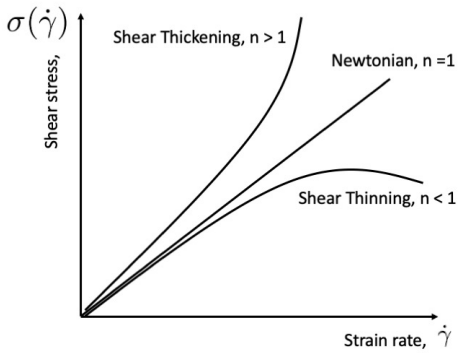



***Figure 2.6.:*** *The flow curve showing the relation between the applied shear rate, $\dot{\gamma}$, and the shear stress, $\sigma$. Dense emulsions commonly shows yield stress, and the shear stress, $\sigma$, tends to finite limit as $\dot{\gamma}$ tends to 0.*

rheology of soft glassy materials is often characterised by a nonlinear relation between the applied stress and the resulting strain. A popular expression of such non-linear behaviour is provided by the Herschel-Bulkley (HB) relation $\sigma = \sigma_Y + A\dot{\gamma}^{\beta}$, where $\sigma$ is the applied stress, $\dot{\gamma}$ the resulting shear (inverse

time) and $A$ a material constant [59, 60]. The HB relation is characterised by a non-zero yield stress, $\sigma_Y$ , below which no flow takes place, and by a scaling exponent $\beta \neq 1$. In [38] a first evidence that a mesoscopic LBM with competing short-range attractive and midrange repulsive interactions supports emergent Herschel-Bulkley (HB) rheology, i.e. a power-law dependence of the shear stress as a function of the strain rate, beyond a given yield stress threshold, following the "power-law fluid" for $A = 0$ [61].

# 3. Numerical modeling

This chapter introduces the numerical models, implemented in the LBE3D code, that we used in order to study chaotic emulsification via fully resolved computer simulations. To accurately model the physics of turbulent multicomponent emulsions we implement a numerical scheme based on the Lattice Boltzmann model (LBM) [19, 32], basically a discretisation of the continuum Boltzmann equation (BE) [62]. In particular, we implemented a force-based multicomponent solver following the formulation originally proposed by Shan and Chen in [6, 63] and later extended to incorporate double-belt coupling [37], thus allowing to model the presence of a disjoining pressure. First, we briefly present the BE from which the Lattice Boltzmann equation (LBE) is derived, second, we introduce the LBE multicomponent and then, the concepts of surface tension and disjoining pressure are presented. Finally, we describe how to model the turbulent stirring required to break the interface between the two components, fundamental to obtain a turbulent emulsion.

## 3.1. The Boltzmann Equation

In a diluted gas, investigating flows at the scale of eg. $1\text{cm}^3$ would require describing the dynamics of a number of particles of the order of the Avogrado number, $N_{av} \sim 10^{23}$, and is clearly impossible to tackle even with modern Exascale systems [32]. Additionally, when focusing on the meso- and macroscales, the description of the motion of each individual particle is no longer necessary because the macroscopic fields of interest, such as eg. densities, velocities, temperature, etc., can be computed through hydrodynamic quantities given by local averages over large number of molecules. The kinetic theory of gases [62] describes exactly how to perform this averaging, as well as how the hydrodynamic quantities evolve in time, via the introduction of the probability distribution function $f(\boldsymbol{x}, \boldsymbol{v}, t)$, a function of the position $\boldsymbol{x}$, velocity $\boldsymbol{v}$ and time $t$, giving the probability of finding a molecule close to the position $\boldsymbol{x}$, at time $t$, and with a velocity close to $\boldsymbol{v}$. From this mesoscopic quantity, $f$, it is then possible to define how hydrodynamics fields such as the density $\rho$ and the fluid velocity $\boldsymbol{u}$ can be derived as moments of $f(\boldsymbol{x}, \boldsymbol{v}, t)$. In particular,

the density of the fluid can be computed by first looking at the probability of finding a particle in a given position $\boldsymbol{x}$ with any velocity $\boldsymbol{v}$ and thus by finally integrating over the full velocity space:

$$\rho(\boldsymbol{x}, t) = \int f(\boldsymbol{x}, \boldsymbol{v}, t) d\boldsymbol{v} \tag{3.1}$$

Similarly the macroscopic flow momentum is computed as:

$$\rho(\boldsymbol{x}, t)\boldsymbol{u}(\boldsymbol{x}, t) = \int f(\boldsymbol{x}, \boldsymbol{v}, t)\boldsymbol{v} d\boldsymbol{v} \tag{3.2}$$

With the same approach other fields like stress tensor and energy can be derived [64]. To describe the evolution of the distribution function, $f(\boldsymbol{x}, \boldsymbol{v}, t)$, the Boltzmann Equation (BE) can be used:

$$\partial_t f + \boldsymbol{v} \cdot \boldsymbol{\nabla}_x f + \boldsymbol{a} \cdot \boldsymbol{\nabla}_v f = \Omega(f) \tag{3.3}$$

where the term $\partial_t$ is the partial derivative with respect to time of the probability function $f(\boldsymbol{x}, \boldsymbol{v}, t)$ at a given position in time, the term $\boldsymbol{v} \cdot \boldsymbol{\nabla}_x$ describes how $f(\boldsymbol{x}, \boldsymbol{v}, t)$ evolves in the presence of transport and the term $\boldsymbol{a} \cdot \boldsymbol{\nabla}_v$ describes the effect of the forcing ($\boldsymbol{a} = \boldsymbol{F}/m$). The collision operator $\Omega(f)$, on the r.h.s, describes all the interactions occurring between the particles. This equation cannot be solved until the collision operator is properly defined. The collision operator can be complicated to describe as it depends upon many parameters including the intermolecular force occurring between particles. One way to simplify the treatment of the collision operator, the BGK [65] approximation is usually applied. Such BGK collision operator corresponds to a first order Taylor expansion with respect to equilibrium and it can be expressed in the form of:

$$\Omega(f) = -\frac{1}{\tau}(f - f^{eq}) \tag{3.4}$$

where $\tau$ is the characteristic collision relaxation time which determines the viscosity of the fluid, while $f^{eq}$ is the function describing the local equilibrium. In the BGK collision operator, the local equilibrium is assumed to be described by the Maxwell-Boltzmann distribution which is given by:

$$f^{eq}(\boldsymbol{x}, \boldsymbol{v}, t) = \rho(\boldsymbol{x}, t) \cdot \left(\frac{m}{2\pi k_B T}\right)^{3/2} \cdot \left(-\frac{m|\boldsymbol{v} - \boldsymbol{u}(\boldsymbol{x}, t)|^2}{2k_B T}\right) \tag{3.5}$$

where $\rho(\boldsymbol{x}, t)$ is the hydrodynamic macroscopic density, and the velocities, $\boldsymbol{v}$, are normally distributed around the average local velocity $\boldsymbol{u}(\boldsymbol{x}, t)$. Collisions must

conserve the quantities of mass, momentum and energy, as initially assumed for the probability distribution function $f(\boldsymbol{x}, \boldsymbol{v}, t)$ in Eq. (3.1) and Eq. (3.2). The conservation constrains of mass and momentum as moments of the collision operator, are expressed as:

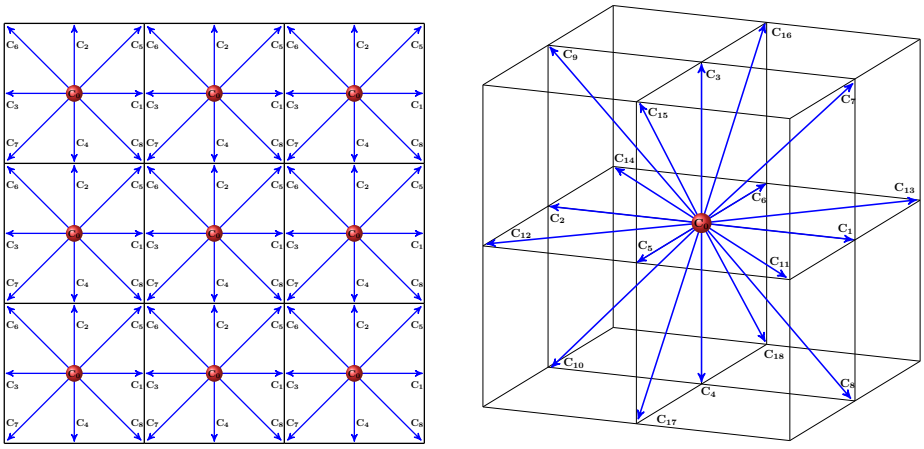
$$\rho = \int \Omega(f)^{eq} d^3\boldsymbol{v} = 0 \quad \text{and} \quad \rho\boldsymbol{u} = \int \Omega(f)^{eq} \boldsymbol{v} d^3\boldsymbol{v} = 0 \tag{3.6}$$

## 3.2. Lattice Boltzmann Method and numerical modeling of multicomponent fluids

Closing the gap between microscopic and macroscopic scale the LBM introduces a set of equations, directly derived from the kinetic theory of gases, describing the motion of particles at the mesoscopic scale. Although by switching from a description based on hydrodynamic fields, eg. Eq. 2.2, to the single particle distribution function reported in the section above, we shift from a 3-dimensional space problem, to a 6-dimensional space problem, plus time. This may not appear convenient to numerically simulate a single phase flow, however the LBM is highly suited for efficient implementation. This is partially due to the fact that a minimal set of velocities is required to concretely reproduce the evolution of hydrodynamic moments. The basic quantities of the LBM are the discrete particle distribution functions $f_i(\boldsymbol{x}, t)$, where the index $i$ indicates the discretised speeds, representing all possible direction of motion of the gas particle. In other words $f_i(\boldsymbol{x}, t)$ represents the density of particles with velocity $\boldsymbol{c}_i = (c_{ix}, c_{iy}, c_{iz})$ at position $\boldsymbol{x}$ and time $t$. Likewise the mass density $\rho$ and momentum density $\rho\boldsymbol{u}$ at $(\boldsymbol{x}, t)$ can be found through the weighted sums representing the lowest order moments of $f_i$:

$$\rho(\boldsymbol{x}, t) = \sum_i f_i(\boldsymbol{x}, t), \quad \rho\boldsymbol{u}(\boldsymbol{x}, t) = \sum_i \boldsymbol{c}_i f_i(\boldsymbol{x}, t) \tag{3.7}$$

The discretised function $f_i$ is defined in space, for every point $\boldsymbol{x}$ in the lattice, with spacing $\Delta x$, at certain discrete times $t$, separated by a time step $\Delta t$, conventionally set to 1. The space discretisation turns the continuum representation into a discrete one, in our case a 3-dimensional grid of points. We will name those as lattice sites in the rest of this thesis. The discrete velocities $\boldsymbol{c}_i$, representing each possible particle direction in the grid, are scaled in such a way that, at a time $\Delta t$, a particle moves from one grid cell to a neighbouring cell, assuming as customarily done, for the sake of simplicity, that $\Delta x = \Delta t = 1$. These velocity sets, named particle populations (or distributions) are usually



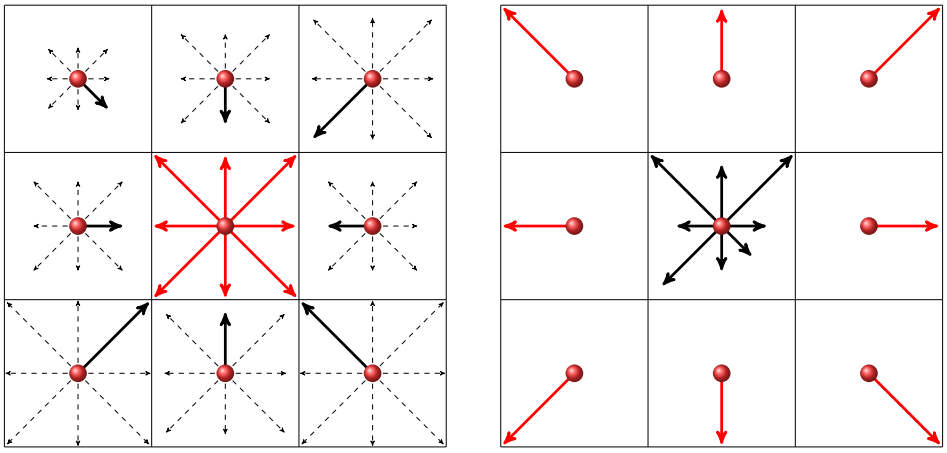

**(a)** A D2Q9 Grid of Latttices **(b)** The D3Q19 Lattice

***Figure 3.1.:*** *Graphical representation of two lattice schemes for 2-dimensional (3.1a) and 3-dimensional (3.1b) simulations. In 3.1b a D3Q19 lattice is represented, showing the velocity set of speeds $\boldsymbol{c}_i$, with $i = 0, \ldots, 18$, indicating all the potential direction of motions of a given particle. In 3.1a the representation of a discretised 3x3 grid of sites, in this case the D2Q9 lattice scheme is reported to simplify the representation. Each lattice shows a set of 9 kinetic velocities $\boldsymbol{c}_i$, with $i = 0 \ldots 8$*

denoted as D$d$Q$q$, where $d$ is the number of spatial dimensions and $q$ is the number of velocities. Figure 3.1 shows a graphical representation of two of those possible lattice schemes, for a 2-dimensional D2Q9 lattice (panel 3.1a), as well as for the D3Q19 scheme (panel 3.1b), that we implemented in the LBE3D and used for all simulations presented in this thesis. By discretising the BE in velocity space, physical space and time we obtain the LBE:

$$f_i(\boldsymbol{x} + \boldsymbol{c}_i \Delta t, t + \Delta t) = f_i(\boldsymbol{x}, t) + \Omega_i(\boldsymbol{x}, t) \tag{3.8}$$

This equation expresses the fact that $f_i(\boldsymbol{x}, t)$ particles move with velocity $\boldsymbol{c}_i$ from $\boldsymbol{x}$ to a neighbouring point $\boldsymbol{x} + \boldsymbol{c}_i \Delta t$ at the next time step $t + \Delta t$, therefore modelling the streaming of gas molecules long their characteristics. In the LBM this step is commonly named streaming step. Consequently, particles undergo a collision at each lattice node, this is represented by the $\Omega_i(\boldsymbol{x}, t)$ term in Eq. 3.8. The particles motion consequently generates a particles collision within each site. This collision step models the particle collisions by redistributing the particle populations $f_i$ on each site. A graphical representation of the LBM steps of streaming and collision is given in Figure 3.2.



**(a)** Streaming **(b)** Collision

***Figure 3.2.:*** *The two LBM steps of streaming and collision in a D2Q9 LBM scheme. In (a) the possible particle directions are highlighted in the centre lattice, surrounded by other lattices where the move of each single particle towards the centred lattice is displayed. All distributions are copied to the neighbour cells while a new set of distributions fills the cell. In (b) the diffusive transport is represented where the particles, first collide locally inside each grid cell and then relax towards the equilibrium, defining a new particles distribution function.*

While there are many different collision operators $\Omega_i$ available, the simplest one that can be used for the NS simulations is the discretised version of the already presented above BGK operator:

$$\Omega_i(f) = -\frac{f_i - f_i^{eq}}{\tau}\Delta t \tag{3.9}$$

The collision operator let the populations relax towards equilibrium $f_i^{eq}$ at a rate determined by the relaxation time $\tau$. The equilibrium is typically expressed via a polynomial approximation of the Maxwell-Boltzmann formulation and it is given by:

$$f_i^{eq}(\boldsymbol{x}, t) = w_i \rho \left(1 + \frac{\boldsymbol{u}\cdot\boldsymbol{c}_i}{c_s^2} + \frac{(\boldsymbol{u}\cdot\boldsymbol{c}_i)^2}{2c_s^4} - \frac{\boldsymbol{u}\cdot\boldsymbol{u}}{2c_s^2}\right) \tag{3.10}$$

where $w_i$ are the weights relative to the chosen velocity set. The equilibrium is such that its moments are the same, as those of $f_i$, implying $\sum_i f_i^{eq} = \sum_i f_i = \rho$ and $\sum_i \boldsymbol{c}_i f_i^{eq} = \sum_i \boldsymbol{c}_i f_i = \rho\boldsymbol{u}$. The equilibrium depends only on the local

density $\rho$ and fluid velocity $\boldsymbol{u}$. These are calculated from the local value of $f_i$ via Equation (3.7), with the fluid velocity found as $\boldsymbol{u}(\boldsymbol{x},t) = \rho\boldsymbol{u}(\boldsymbol{x},t)/\rho(\boldsymbol{x},t)$.

The discretised Boltzmann equation, together with the BGK collision operator reads:

$$f_i(\boldsymbol{x} + c_i\Delta t, t + \Delta t) - f_i(\boldsymbol{x},t) = -\frac{1}{\tau}[f_i(\boldsymbol{x},t) - f_i^{eq}(\rho,\boldsymbol{v})] \tag{3.11}$$

The equation is decomposed in two parts. The first part describes the collision operator (or relaxation) that in case of $\tau/\Delta t = 1$ becomes as:

$$f_i^*(\boldsymbol{x},t) = f_i^{eq}(\boldsymbol{x},t) \tag{3.12}$$

where $f_i^*$ represents the distribution function after collisions and $f_i^{eq}$ is found from $f_i$ through Equation (3.10). The second part is the streaming (or propagation),

$$f_i(\boldsymbol{x} + \boldsymbol{c}_i\Delta_t, t + \Delta_t) - f_i(\boldsymbol{x},t) = f_i^*(\boldsymbol{x},t) \tag{3.13}$$

The described classical form of the LBM in its simplest representation is a powerful tool for fluid simulation offering an alternative approach to study the physics of fluids via computer simulation. Indeed, the LBM has been used to perform simulations under a broad range of flow and fluid conditions such as, particle-laden flows, rarefied gases, suspensions in flow, etc. [20, 21, 22, 66]. Beyond the traditional problems of homogeneous hydrodynamics, Lattice Boltzmann models have proven suitable for the modeling of complex fluids. Indeed, one of the most popular applications of the LBM is simulating multicomponent flows, two (or more) different fluids which differ by their typical physical properties (density, viscosity, etc). In a multicomponent flow we have to account for the diffusion between these two components. In the model, each fluid, $\sigma = A$ and $B$, is described via its own discrete distribution function, $f_{\sigma a}(\boldsymbol{x}, \boldsymbol{c}_a; t)$, which represents a measure of the probability to find a particle of the fluid $\sigma$, at the position $\boldsymbol{x}$ and time $t$, streaming with the discrete velocity $\boldsymbol{c}_a$. Within the single component standard LBM only the nearest-neighbour (NN) is taken into consideration, as graphically represented in Figure 3.2. However, a larger range of interaction, indicated as second belt [67], has been proposed to model multicomponent and multiphase fluids with LBM. In replacement of the $i$ index used in other equations of this paragraph, the index $a$ is now used as it spans over the NN and next-to-nearest neighbour (NNN) in a Cartesian 3-D lattice grid (see next paragraph). The updating process for each distribution function $f_{\sigma a}$, at each time step, of unitary duration ($\Delta t = 1$) is represented by the classical LBE [68, 69]:

$$f_{\sigma a}(\boldsymbol{x}+\boldsymbol{c}_a;t+1)-f_{\sigma a}(\boldsymbol{x};t)=-\frac{f_{\sigma a}-f^{eq}_{\sigma a}}{\tau}+F_{\sigma a}(\boldsymbol{x};t) \tag{3.14}$$

which combines the effects of the free-streaming (l.h.s. of the above equation), the collision step, assumed for both fluids as a relaxation process towards the local equilibrium state with a characteristic time scale $\tau$ (first term at r.h.s.), and the interaction forces $f_{\sigma a}(\boldsymbol{x};t)$ (second term at r.h.s.). The local equilibrium fluid state, $f^{eq}_{\sigma a}(\boldsymbol{x},t)$, is evaluated via the common polynomial expansion of the Maxwell-Boltzmann distribution, valid in the limit of small fluid velocity:

$$f^{eq}_{\sigma a}(\boldsymbol{x},t)=w_a\rho_\sigma\left(1+\frac{\boldsymbol{u}\cdot\boldsymbol{c}_a}{\boldsymbol{c}_s^2}+\frac{(\boldsymbol{u}\cdot\boldsymbol{c}_a)^2}{2c_s^4}-\frac{\boldsymbol{u}\cdot\boldsymbol{u}}{2c_s^2}\right) \tag{3.15}$$

and constructed incorporating information on the local density of the fluid component $\sigma$, $\rho_\sigma$, and on the global velocity for the binary mixture, $\boldsymbol{u}$. These hydrodynamic quantities can be readily evaluated as $\rho_\sigma=\sum_a f_{\sigma a}$ and $\boldsymbol{u}=\sum_{\sigma a} f_{\sigma a}\boldsymbol{c}_a/\rho$ with $\rho=\sum_\sigma \rho_\sigma$. In the local equilibrium distributions, the coefficients $w_a$ represent a set of weights, specific for the chosen quadrature (D3Q19 for the model here employed), and $c_s=\frac{1}{\sqrt{3}}$ is a molecular characteristic velocity.

## 3.3. Modeling of surface tension and disjoining pressure

In order to study the dynamics of turbulent emulsions we implement in the LBE3D code a version of the LBM for non-ideal binary fluids able to combine a positive surface tension, responsible for the emergence of complex interfaces between the fluids, with a disjoining pressure able to prevent the interface coalescence, in absence of external stirring or high-packing pressure. In particular we implement the pseudo-potential approach introduced by Shan and Chen [6], combined with a simple phenomenological modeling of surfactant (i.e. surface stabilisation effects), via a second neighbour coupling in the Shan-Chen multicomponent force. This approach has been amply demonstrated to be accurate and physically quantitatively correct while retaining the conceptual simplicity and numerical efficiency of the local multicomponent force approach [70, 71, 4].

The non-ideal forces $f_{\sigma a}(\boldsymbol{x};t)$, which are modelled as in the Shan-Chen pseudo-potential formulation, incorporate several inter-particle forces. In particular, they feature a standard repulsive ($r$) force between the two components with magnitude $\mathcal{G}_{AB}$:

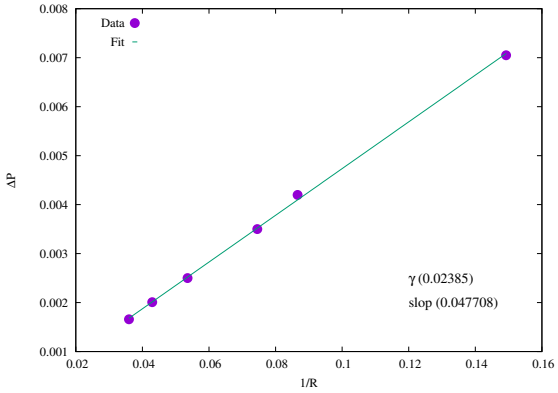
$$F^{(r)}_\sigma(\boldsymbol{x})=-\mathcal{G}_{AB}\varphi_\sigma(\boldsymbol{x};t)\sum_{a,\sigma'\neq\sigma} w_a\varphi_{\sigma'}(\boldsymbol{x};t)\boldsymbol{c}_a \tag{3.16}$$

where the pseudo-potential $\varphi_{\sigma'}(\boldsymbol{x};t)$ is set equal to the density, i.e. $\varphi_{\sigma'}(\boldsymbol{x};t) = \rho_{\sigma'}(\boldsymbol{x};t)$, and it is responsible for the phase separation. The introduction in the model of the potential between the components gives the opportunity to simulate the fluid separation, allowing to study complex phenomena such as spinodal decomposition of immiscible fluid mixture, formation and dynamics of droplets, multicomponent turbulence, etc. However, while the surface tension defines the separation between the fluid, it is not sufficient to maintain droplets of the same component separated due to the dominating attractive forcing between particles of the same component [14].

Therefore, we also implement a disjoining pressure [72] in order to model the repulsive force, acting as a stabilising agent (in this way mimicking the effect of a surfactant in emulsions), that instead stabilises the interaction between particles of the same component. Both fluid components A and B are subjected to competing interactions in order to induce frustration (f) forces and, therefore, preventing the occurrence of interface coalescence. In details, two different contributions to frustration are modelled as expressed via the following relation, being $\sigma$ one of the fluid components A and B:

$$F_{\sigma}^{(F)}(\boldsymbol{x}) = -\mathcal{G}_{\sigma\sigma,1}\psi_{\sigma}(\boldsymbol{x};t)\sum_{a\in NN} w_a\psi_{\sigma} - \mathcal{G}_{\sigma\sigma,2}\psi_{\sigma}(\boldsymbol{x};t)\sum_{a\in NNN} w_a\psi_{\sigma} \qquad (3.17)$$

where a short-range self-attraction force, involving only the NN sites, and governed by the magnitudes ($\mathcal{G}_{AA,1}, \mathcal{G}_{BB,1} < 0$; first term at r.h.s. of the above equation), is introduced along with a long-range self-repulsion force, involving only the NNN sites, and controlled by the magnitudes ($\mathcal{G}_{AA,2}, \mathcal{G}_{BB,2} > 0$; second term at r.h.s. of the above equation).
One can measure the surface tension at the surface of a sphere placed at the centre of a discretised grid in different dimensions, at different radius, by performing a standard Laplace test. The initial radius of the spheres $R$ and the size of the relative discretised grids $L$ are reported in Table 3.3b, together with the measured pressure jumps, $\Delta P$, computed as reported in Eq. 2.5. Data are plotted in Figure 3.3, showing the slope of the estimate the surface tension, $\gamma = 0.02378$. For all simulations presented in this project we use the same coupling parameters as from [37] to define the magnitudes of the micro-scale forcing at the surface. More in detail those are, the standard repulsive force, Eq. (3.16), the short-range self-attraction force and of long-range self-repulsion force, Eq. (3.17), between the two components, triggering the surface tension




| $R$ | $L$ | $\Delta P(10^{-}3)$ |
|---|---|---|
| 8 | 32 | 7.0 |
| 12 | 32 | 4.2 |
| 16 | 64 | 3.5 |
| 20 | 64 | 2.5 |
| 24 | 64 | 2.0 |
| 32 | 128 | 1.7 |

***Figure 3.3.:*** *The result of a Laplace test. The pressure jumps $\Delta P$ vs. the reverse of the droplet radius $1/R$. From the slope one can estimate the surface tension ($\gamma$) measured placing a sphere at the centre of a discretised grid. The table reports different droplets radius and different dimension of the discretised grid used for the measurement. The surface tension has been measured using the LBE3D and the parameters reported in Eq. (3.18), defined as standard set at $\rho_0 = 0.83$, indicating the density value above which non-ideal effects come into play for a given specie A or B. The relaxation time is fixed to $\tau = 1$. The surface tension is measured with the system at rest (no-forcing).*

and the disjoining pressure as described in Chapter 3:

$$\begin{aligned} \mathcal{G}_{AA,1} &= -9.0, \quad \mathcal{G}_{AA,2} = 8.1 \\ \mathcal{G}_{BB,1} &= -8.0, \quad \mathcal{G}_{BB,2} = 7.1 \\ \mathcal{G}_{AB} &= 0.405 \end{aligned} \tag{3.18}$$

defined as standard set at $\rho_0 = 0.83$, indicating the density value above which non-ideal effects come into play for a given specie A or B. The relaxation time is always fixed to $\tau = 1$. To be noted that the two opposite repulsion forces are able to introduce an effect localised at the interface between the two components such that the coarsening process in the system is dramatically slowed down. This feature is at the basis of the possibility to simulate droplets of one dispersed component into the other which, indeed, are effectively stabilised against coalescence, as displayed in Figure 3.4.

## 3.4. Modeling of the chaotic stirring

The applied external forcing used to generate a chaotic stirring to mix together the two immiscible fluids is modelled via a large-scale turbulent forcing as in [15, 13]. Due to the presence of this forcing, interfaces are broken as inertial

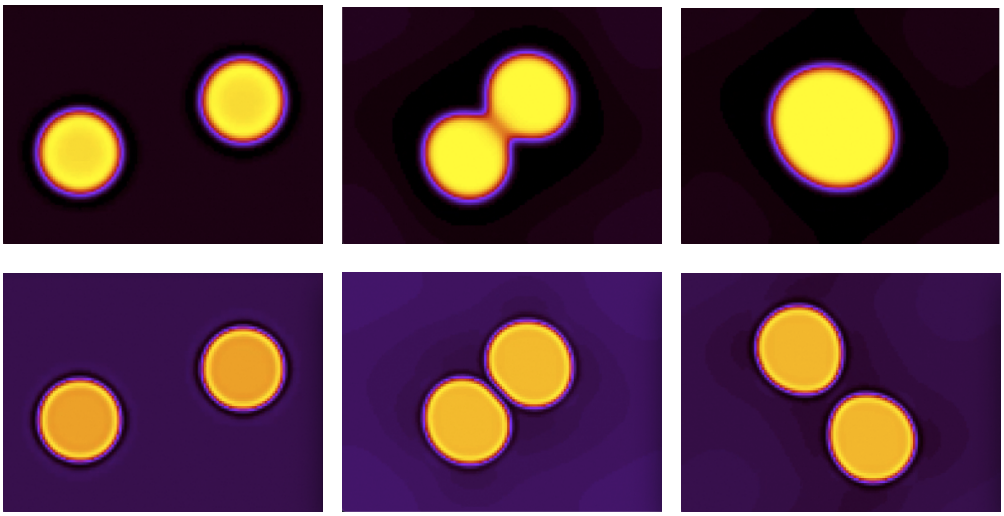


***Figure 3.4.:*** *The picture describes three different time snapshots of a 2-dimensional simulation performed to show the effect of the surface tension, and the combined implementation of both the surface tension and the disjoining pressure. In the upper part, starting from an initial condition of two droplets, the attracting potential clearly defines the separation between the two components (inside and outside the droplets' shape) but after few time steps the droplets of the same component merge together, despite the absence of external forcing. In the lower part, it is instead visible how the surface tension still act on long range interaction between the two droplets, that deforms when coming close to each other, inducing a repulsion such that those move away one from the other. A similar reppresentation was also given in [73].*

forces win over surface tension. When the flow intensity increases, smaller droplets are produced, when it decreases, larger droplets are produced. The hydrodynamic motion is generated by applying forcing at every point of the discretised grid, at every time step, modulated as a sum of sine waves with small wavenumbers. The sine waves phases are randomly evolved in time, by means of a stochastic process, to ensure a homogeneous and isotropic stirring. The forcing expression for the generic $i^{th}$ component is:

$$F_i^{\alpha}(\boldsymbol{x}, t) = A\rho^{\alpha} \sum_{j \neq i} \left[ sin(k_j x_j + \Phi_k^{(j)}(t)) \right] \tag{3.19}$$

where $i, j = 1, 2, 3$, $F^{\alpha}$ is the external force at $\boldsymbol{x}$ and time $t$, and $A$ controls the forcing amplitude, $k_j$ are the wave-vector components and the sum is limited to $k^2 = k_1^2 + k_2^2 + k_3^2 \leq 2$. The phases $\Phi_k^{(j)}$ are evolved in time according to independent Ornstein-Uhlenbeck processes with the same relaxation times

$T = u_{rms}/N$, where $N$ is the discretised box dimension and $\boldsymbol{u}_{rms}$ is chosen of the order of the typical values for the large-scale velocity. In Figure 3.5, the renderings show the effect of the forcing on the emulsification process, displaying the same emulsion with the hydrodinamic stirring on (left) and off (right).

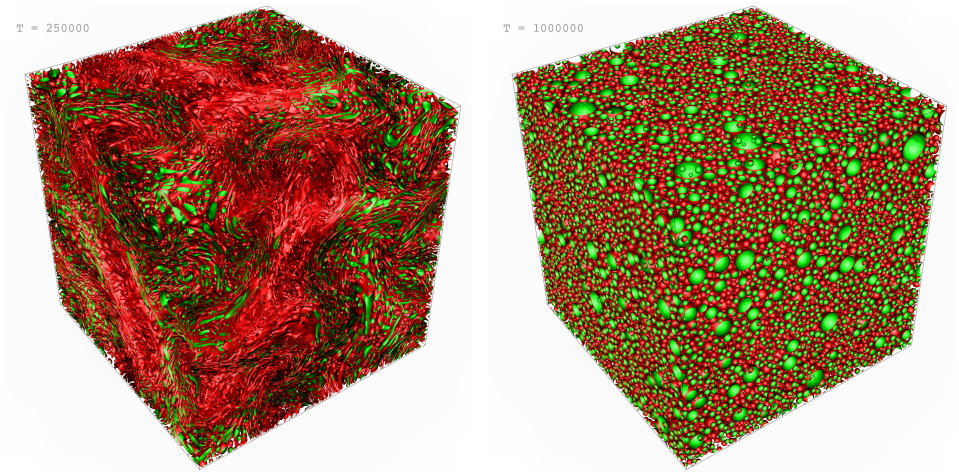



***Figure 3.5.:*** *Two different high-resolution images of a multicomponent flows in a 3-dimensional lattice with resolution* $1024^3$*, with the dispersed phase component being* 40% *of the total volume. The surface separating the two fluid component is visualised in such a way that the side of the surface facing fluid one and fluid two have two different colours, red and green respectively. Under the influence of a large-scale stirring (left), droplets in the minority phase constantly undergo coalescence and breakup, with the droplets shapes visibly non-spherical. After that the stirring is switched off (right) the emulsion achieves a metastable equilibrium. In this stationary non-jammed state droplets are visibly spherical.*

# 4. The making of dense emulsions

In this chapter we study the process of chaotic emulsification for stabilised binary mixtures of immiscible fluids via fully resolved computer simulations, using the mathematical and numerical models presented in chapter 3. [1]. We demonstrate that the slow (adiabatic) addition of one fluid component during the flowing state of the fluids allows to produce highly packed emulsion with a volume fraction ($\phi$) of the droplets phase close to 80%. We show how hydrodynamic stresses initially influence the droplets formation process, and later on, the flow and small scale morphology of the emulsion. An analysis on a number of simulations corresponding to different physical conditions is presented, reporting, e.g., global statistics such as the time evolution of the number and average size of droplets for different resolutions, initial conditions, volume fraction and forcing amplitude. Finally, we provide a description of the jammed resting state as well as what happens when things go wrong, namely the phenomena of catastrophic phase inversion.

## 4.1. Building of an emulsion

Dense emulsions are fragile systems characterised by the majority phase fluid component dispersed, in the form of droplets, in a matrix of the other fluid that makes a thin film around the droplets (see Figure 3.5b). Mayonnaise is an example of a dense emulsion, and it is commonly known that succeeding in making it at home, requires experienced hands to not curdle it. Similar problems arise when considering the making of dense emulsions via computer simulations. One faces two challenges here: first, the physical protocol to produce an emulsion requires the slow increase of the fluid component in the dispersed phase; second, the numerical model employed has never been used before for the study of chaotic emulsions in 3-dimensions. The choice of parameters and protocols needed to be explored almost from scratch. Indeed, here we simulated a number of emulsions analysing the phenomenology during the process

[1] This chapter closely follow the contents of Girotto et al., "Build up of yield stress fluids via chaotic emulsification", currently in preparation. The media content on the emulsification process was published in [74].

of emulsification as well as the emulsions morphology at different resolutions, varying the magnitude of the large-scale stirring force, increasing the volume fraction of the dispersed phase from low to high, and by varying the initial conditions. Most of the simulations studied are characterised by a common process in which the dispersed immiscible fluid component is slowly added in order to achieve the desired volume fraction in the multicomponent fluid.
To analyse the process of emulsification in detail, we introduced in the LBE3D a method to accurately and uniquely identify all droplets of the dispersed phase, see Chapter 5 for more details. Finally we compute the volume,$V_k$ of every single droplet, $k$, in the dispersed phase and define the volume fraction, $\phi$, of the dispersed phase as:

$$\phi = \frac{\sum_k V_k}{L^3} \tag{4.1}$$

where $L$ is the linear size (in lattice points) of the discretised grid, defining the resolution of a given simulation.

### 4.1.1. Reaching highly packed emulsions

As the main focus of this thesis is the analysis of dense stabilised emulsions, we describe here how, during the emulsification process, the volume fractions is increased in the dispersed phase. In a way, the in silico approach presented here is similar to the making of mayonnaise at home, a delicate process that does not always succeed where a key for success consists in both adding oil and mixing slowly. In the process of making emulsions via computer simulation, we control the forcing of the mixing with the parameter $A$ (Eq. 3.19), while the amount of the component adiabatically added to the emulsion is defined as follows. Given the total mass of the component $\alpha$ as $M_\alpha = \int \rho_\alpha(x) d^3x$, being $\alpha$ either the first (1) or the second (2) component, the total mass of the binary fluid is computed as $M = M_1 + M_2$. The dispersed component is uniformly added throughout the volume, incrementing the volume of existing droplets that subsequently break due to hydrodynamic stresses, thus resulting in the production of additional droplets. Of course, during this process, to maintaining incompressibility, an equivalent amount of the second component is removed from the volume. Therefore, every adiabatic addition results in a new state where the mass of the two components changes according to $M_1 = \alpha M_1$ and $M_2 = \beta M_2$, respectively. This process is repeated every a fixed number of LBM time steps during an initial injection phase, from the time step 0 to the time step 1332800 in the illustrated case in Figure 4.1. The two parameters $\alpha$ and $\beta$, regulating the ratio of addition and extraction of the components and the

resulting mass fraction, $\lambda$, are defined as:

$$\alpha = 1 + \delta \quad \beta = 1 - \left(\frac{M_1}{M_2}\right)\delta$$
$$\lambda = \frac{M_1}{M_1 + M_2} \tag{4.2}$$

with $\delta$ representing the relative amount of injected mass of the dispersed component (1) at every time step. During the injection phase, the total mass

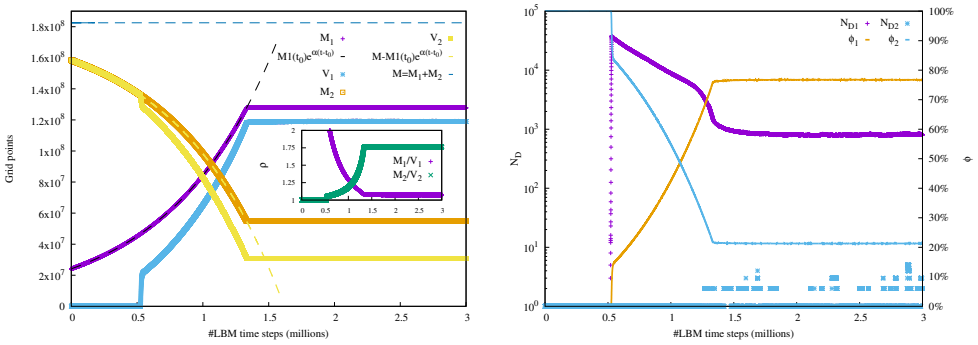



***Figure 4.1.:*** *Time evolution of the total mass, $M_1$ and $M_2$, along with the total volumes, $V_1$ and $V_2$, of the two components, (1) and (2), respectively, from the initial injection phase, to the final target volume fraction, $\phi$, for simulation $N512VF77$, see Table 4.2. Initially the volume fraction of the dispersed components is set to 0% (thus 100% of the volume is occupied by fluid (2)). (Left) Figure shows the (total) mass and volume of each of the two components varying as a function of time and as a consequence of the injection of component (1), together with the total mass of the multicomponent fluid, $M$. It is highlighted how, during the injection, the mass of the two components change exponentially following the analytical solution $M_1 = M_1 e^{\alpha(t-t_0)}$ and $M_2 = M - M_1 e^{\alpha(t-t_0)}$, derived from Eq. (4.2). The value of the average densities of the two components, defined as the total mass of the component divided by the total volume a specific component, is reported in the inset. The average density of the component (2) is initially equal to 1, being the only one present in the system. At the LBM time step $0.5M$ the droplets of the component (1) start to nucleate and therefore $V_1$ starts to be different from zero. (Right) Figure shows the detail of the variation of the volume fraction of the two components, $\phi_1$ and $\phi_2$, as a function of time, showing also the number of droplets for each component, $N_{D1}$ and $N_{D2}$, respectively. The volume fractions of the two component are indicated on the y2-axis.*

is preserved as the system gradually adjusts, according to the density value of the two components. In Figure 4.1 (left) we report the time evolution of the mass, $M$, and the volume, $V$, for both components, and for the whole time, for a typical simulation, namely $N512VF77$ (see Table 4.2). In the inset we report the average density values, on the full volume, of the two components, given as

$M_\alpha/V_\alpha$. The volume fraction of the dispersed component 1 is initially set to 0% (and 100% for component 2, respectively), with the LBM local density values equal to $\rho_1 = 0.18$ and $\rho_2 = 1.18$, on a discretised $512^3$ grid. The left panel of Figure 4.1 shows that the total mass $M = M_1 + M_2$ remains constant during the whole simulation, as it should, while the mass of the two components varies and balances accordingly. In fact, the (opposite) exponential behaviour, during the phase of injection, of the total mass of both the dispersed component, $M_1$, and of the contiguous component, $M_2$, can also be derived analytically from Eq. (4.2) as $M_1 = M_1 e^{\alpha(t-t_0)}$ and $M_2 = M - M_1 e^{\alpha(t-t_0)}$, with $\alpha$ being the parameter regulating the ratio of addition of the component (1), see Eq. (4.2). Notice that the volume of the two components tends to follow the total mass but with an increasing gap towards lower values. Given the fact that the LBM is a diffuse interface-based method and the consequent constrains in modeling a droplet surface (see Chapter 3), this has to be expected as the minority component is initially dispersed into the other but no droplets are formed yet, therefore introducing a bias resulting in a marked difference between mass, $M_1$, and volume, $V_1$, of the dispersed fluid. In Figure 4.1 (right) we report the time evolution of the volume fraction of the two components, together with the number of droplets. It clearly shows that initially there are no droplets, until they start to appear after about 0.5M LBM time steps. With the growing volume fraction of the dispersed field component (in the form of a dispersed droplets phase), also a very small number of droplets of the other component occasionally appears, this corresponds to the formation of a double emulsion [39].
In this thesis, two main approaches have been adopted to initialise the emulsion: either via interface fragmentation, starting from a flat separation of the two components, therefore mimicking the scenario depicted in Figure 4.2, or via nucleation, starting from only a continuum matrix phase (at 100%) and then adiabatically uniformly adding mass of the dispersed component, as in Figure 4.5. In the following we present and discuss results pertaining to a series of emulsions characterised by their initial conditions as well as their final volume fraction $\phi$, and describe how the morphology of the emulsions evolves over time, mainly in term of the number of droplets and their size.

### 4.1.2. Emulsions classification

In the following we present emulsions at different volume fraction, $\phi$, in order to highlight the most significant steps of the process of emulsification: from the formation of the droplets, either via nucleation or via interface fragmentation, to the phase of chaotic stirring, to the final jammed state once the large scale

stirring is turned off. To aid the reading we define a protocol that allows us to uniquely identify either a specific simulation, or a part of it, among the large number of simulations presented in this thesis. The basic identifier is composed by a minimum of there parameters:

INIT: the initial condition, with $I$ indicating interface fragmentation and $N$ indicating nucleation.

RES: the resolution expressed as the number of lattice points.

VF: the finally achieved volume fraction, $\phi$, of the dispersed phase over the total emulsion volume.

We attribute basic naming to all simulations presented in the following two chapters, according to these three parameters $[INIT][RES]V[VF]$. The following optional tags are added, in some cases, to the basic naming, either to uniquely identify a simulation or a specific section of it:

$[T|P|S|SO]$: the temporal intervals characteristic of the most relevant phases of the emulsification process: the initial transient when droplets are generated (T), the phase where the desired volume fraction is achieved (P), a stationary forced phase (S) and, finally, a stationary non-forced phase (SO).

$[FA]$: the constant amplitude prefactor of the applied forcing that multiplies $10^{-6}$.

$[FROM - TO]$: the temporal interval in a given simulation, or a single LBM time step, if FROM=TO.

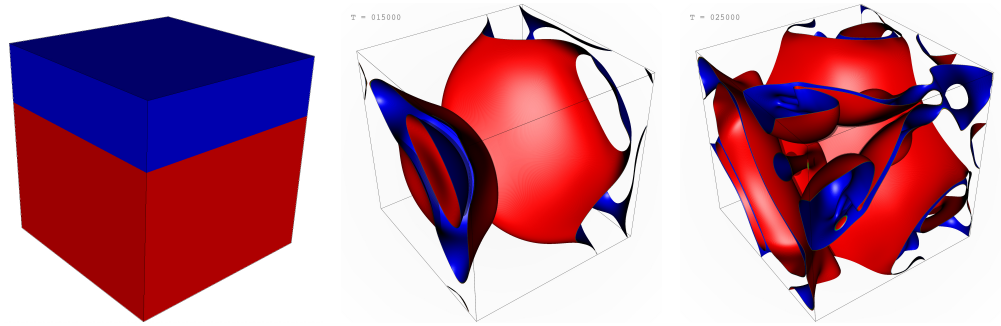
$[VF - FROM - TO]$: the considered interval of the volume fraction of the dispersed phase, or a fixed volume fraction if FROM=TO

We illustrate the use of this notation via few examples. A typical simulation runs for 1M LBM time steps, from an initial condition of a flat interface fragmentation at $512^3$ resolution with a constant volume fraction at 30% is denoted as $I512V30$. Supposing we want to specifically refer to the initial LBM time steps of the simulation, eg. from 0.01M to 0.1M, we refer to it as $I512V30\_T0.0.1-0.1$. Another simulation at $1024^3$ resolution runs for 3M LBM time steps with the dispersed component achieving a final 80% volume fraction

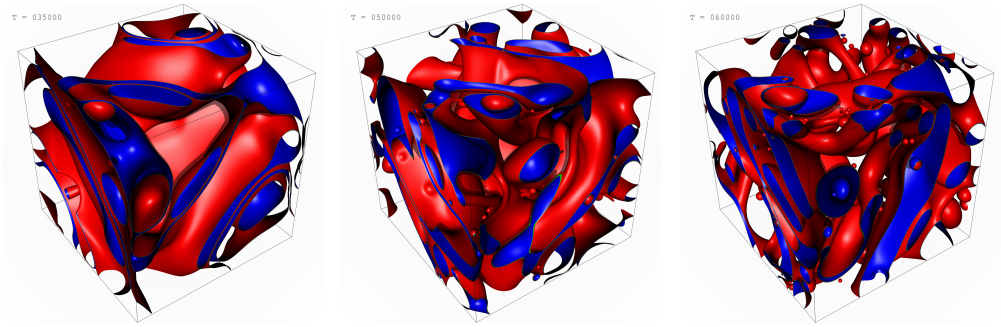
with a constant stirring magnitude of $1.25 \cdot 10^{-6}$. Supposing one wants to indicate a specific time frame of simulation, at the LBM time step 0.5M, when the volume fraction is still 50%, and one needs to identify this simulation between other similar simulations but with a different forcing amplitude. In this case one would attribute the following name: $I1024V80\_P0.5\_VF50\_FA1.25$. With the help of this convention it will be easy to distinguish against all different simulations.

### 4.1.3. Interface fragmentation

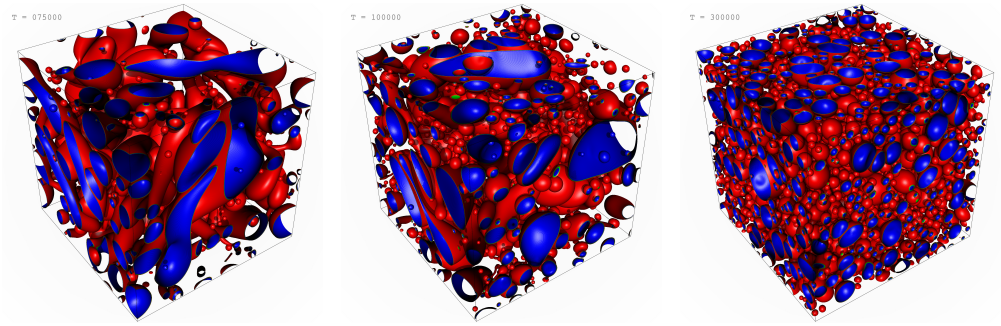
In the case of an emulsion generated from an already existing interface, we will refer to a fragmentation. In this case, the initially flat interface between the two immiscible fluids is broken by the large scale stirring that rapidly destroys the interface, forming a multitude of smaller and smaller droplets of the minority phase component. The reasons why the droplets produced are made of the minority phase component is easily explainable in terms of free energy arguments. Assuming that turbulent or chaotic stresses are responsible for breaking down fluid interfaces thereby producing droplets of a fixed typical size (determined by the surface tension and the intensity of the hydrodynamic stresses) one ends up with two possibilities, either having droplets of the minority phase dispersed in the majority phase or vice-versa. Clearly the first option has a smaller surface energy and it is therefore energetically favoured [75]. The breakup of the initial (flat) interface, in Figure 4.2 it is reported the case of a typical simulation, namely $I512VF38$ (see Table 4.1), takes several LBM time steps (approximately 0.3M LBM time steps for the considered case), and mostly depending on the intensity of the applied stirring. Due to the applied forcing the film interface is first stretched and then broken in large pieces that result in smaller droplets. Large droplets assume a deformed, stretched shape, as they deform under the effect of the forcing, till the moment they breakup into smaller, more spherical droplets. In Figure 4.3 we show the time evolution of the droplet formation for the set of simulations described in this section and reported in Table 4.1. From an initial condition corresponding to a flat interface separating the two fluids at 30% volume fraction, the hydrodynamic stirring is applied for 3M LBM time steps. For every simulation, Table 4.1 reports the time interval at which the desired volume fraction is reached, $T_{PO}$, the final volume fraction, $\phi$, and the final number of droplets, $N_D$. All simulations presented in this section were performed on a 3-dimensional discretised grid of lower resolution $512^3$, with the initial density of the fluid components set at $\rho_1 = 1.18$ and $\rho_2 = 0.18$, with $\rho = \rho_1 + \rho_2$. For all cases a constant forcing magnitude $A = 0.485 \cdot 10^{-6}$ was applied from the first time step throughout



**(a)** I512V38_P0_VF30 **(b)** I512V38_P0.015_VF30 **(c)** I512V38_P0.025_VF30



**(d)** I512V38_P0.035_VF30 **(e)** I512V38_P0.050_VF31 **(f)** I512V38_P0.065_VF31



**(g)** I512V38_P0.075_VF32 **(h)** I512V38_P0.1_VF34 **(i)** I512V38_P0.3_VF38

***Figure 4.2.:*** *Rendering of the most significant time steps during the initial interface fragmentation process for simulation $I512V38$ (see Table 4.1 for further details). The density field is visualised via a reciprocal type of isosurface, where the face of the isosurface facing fluid 1 is rendered in red and the one facing fluid 2 is rendered in blue. The figure shows the gradual formation of droplets due to the effect of hydrodynamic stirring, first deforming and then breaking, the interface between the two components. The evolution of the total number of droplets, as a function of time, for the reported temporal interval is reported in details in Figure 4.3(upper panel), from the LBM step 0 (a) to the LBM time step 0.3M (i).*

the entire simulation time corresponding to of 3M LBM time steps. Figure 4.3

| $Run_{id}$ | $L$ | $\delta$ | $\phi$ | $T_{PO}$ | $A \cdot 10^{-6}$ | $< N_D >$ |
|---|---|---|---|---|---|---|
| $I512V38$ | 512 | $1.25 \cdot 10^{-6}$ | 38% | 114900 | 0.485 | 3283 |
| $I512V49$ | 512 | $1.25 \cdot 10^{-6}$ | 49% | 289900 | 0.485 | 3061 |
| $I512V63$ | 512 | $1.25 \cdot 10^{-6}$ | 63% | 439900 | 0.485 | 2356 |
| $I512V70$ | 512 | $1.25 \cdot 10^{-6}$ | 70% | 504900 | 0.485 | 1573 |
| $I512V77$ | 512 | $1.25 \cdot 10^{-6}$ | 77% | 559900 | 0.485 | 800 |

***Table 4.1.:*** *Parameter description for all simulations displayed in Figure 4.3, showing in particular the injection rate, $\delta$, the magnitude of the forcing, $A$, and including the average number of droplets, $< N_D >$, during the stationary forced phase, after the injection has been stopped at a given LBM step, $T_{PO}$. The table reports values for different volume fractions, obtained using the LBE3D with the parameters reported in Eq. (3.18), defined as standard set at $\rho_0 = 0.83$, indicating the density value above which non-ideal effects come into play for a given specie A or B. The relaxation time is fixed to $\tau = 1$.*

(upper panel) reports the time evolution of the number of droplets, $N_D$, illustrating the characteristics evolution from the initial interface fragmentation up to when the emulsions are almost in steady state forced condition. There are three main phases deserving attention here, to which we will frequently refer to in the rest of this chapter. The first, $T_T$, characterises the time steps needed to form the first droplets. The second, $T_I$, characterises the phase showing a monotonic growth of the number of droplets, up to the the maximum point of fragmentation is reached. The third, $T_P$, characterises the phase where the number of droplets decreases monotonically due to the continuous injection of mass in the dispersed component and the consequent formation of larger droplets in the dispersed phase. We also define a fourth phase, $T_S$, as the time where the number of droplets in the system is mostly consolidated, with the emulsion in a stabilised motion (see the lower panel of Figure 4.3). As for all cases in Table 4.1 we consider $T_T$ up to 0.1M steps, $T_I$ from 0.1M and 0.3M LBM time steps, $T_S$ from 1.25M to 3M LBM time steps and $T_P$ strictly depending from the simulations, and generally related to the choice of $\delta$ (see Eq. (4.2)). The Figure 4.3 (upper panel) shows that at the end of $T_I$ the number of droplets begin to decrease for the cases at higher volume fractions (i.e. beyonds 50%), rapidly decreasing particularly for the cases at high volume fractions. The final morphology, i.e. at the LBM time step 3M, of the emulsions, for all simulations discussed in this section, is shown in Figure 4.4. It clearly visible the formation of larger droplets at increasing the volume fraction, as reported in column $N_D$ of Table 4.1. Droplets are present in larger number at lower volume fractions, displaying more spherical shape, while at higher volume fractions droplets show visible deformation, with marked deviation from sphericity,

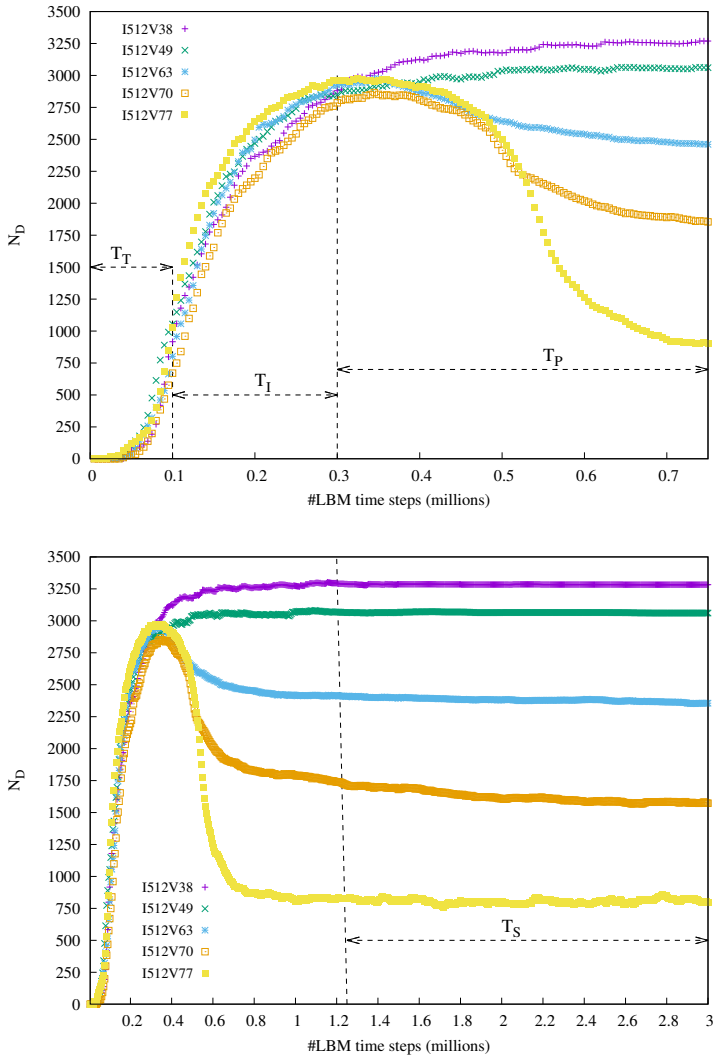
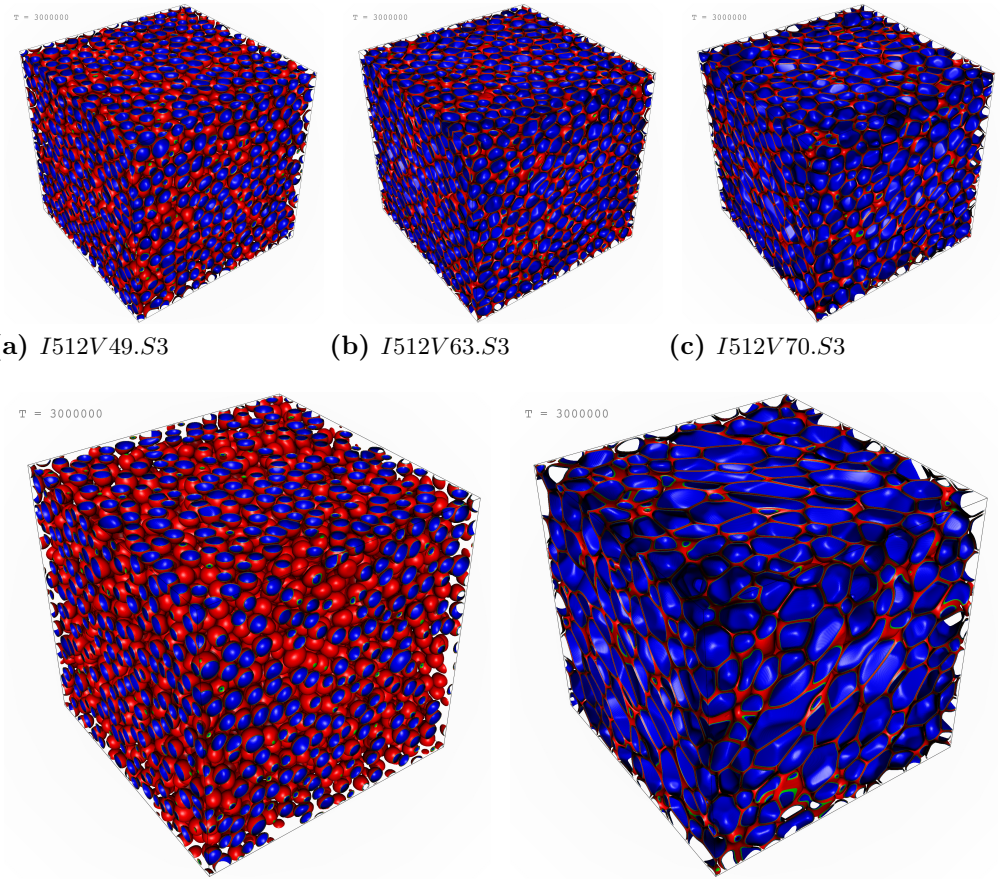



***Figure 4.3.:*** *The total number of droplets, $N_D$, are in function of time for different simulations corresponding to different volume fractions. All simulations are started from a flat interface between the two immiscible fluids at the volume fraction of 30%/70%. In the upper panel, the initial phase is highlighted while, in the lower panel, $N_D$, is plotted for the entire simulation. The final number of droplets achieved for the all cases is reported in column $N_D$ of the Table 4.1. Main temporal phases are reported among the two pictures: the time steps needed to form the first droplets, $T_T$; the phase showing a monotonic growth of the number of droplets, up to the the maximum point of fragmentation is reached, $T_I$; the phase where the number of droplets decreases monotonically due to the continuous injection of mass in the dispersed component and the consequent formation of larger droplets in the dispersed phase, $T_P$; and the time where the number of droplets in the system is mostly consolidated, with the emulsion in a stabilised motion, $T_S$.*

as a consequence of the chaotic stirring, as well as of the high packing ratio of the emulsion and the increased effective viscosity.




**(a)** $I512V49.S3$ **(b)** $I512V63.S3$ **(c)** $I512V70.S3$

**(d)** $I512V38.S3$ **(e)** $I512V77.S3$

***Figure 4.4.:*** *Rendering of the final frame, for all simulations reported in Table 4.1, with special focus on the first and the last. The picture shows the presence of less and larger droplets at increasing the volume fraction. Low volume fractions display spherical droplets while simulations at high-volume fraction are characterised by non-spherical and deformed droplets, evidence of internal frustration due to high-packing fractions.*

### 4.1.4. Nucleation

In the case of emulsion started from a 0% volume fraction the initial droplets form due to nucleation. Also in the case of nucleation we apply the same approach, injecting mass of the dispersed component into the emulsion, the only

difference being in the different initial condition where only the background fluid component is initially present, occupying all the volume, while the dispersed component is adiabatically added. This configuration is characterised by a different initial transient where a large number of small the droplets appears, all of a sudden. High-resolution renderings in Figure 4.5 show this process in details (see also [74] for the original published movie). Once the amount of the dispersed component increases above a critical threshold the first small droplets starts to nucleate, due to the parameter settings that favour phase separation. The high-quality rendering displays a large number of droplets generated in the short range of 0.01M LBM time steps. In Figure 4.6 (left), the time evolution for a set of simulation is reported. It reports $N_D$ in function of time from an initial condition of nucleation, at varying the volume fraction $\phi$ for all cases in Table 4.2. The inset zooms into the initial transient $T_T$ as defined in the previous section. It shows the rapid formation of the droplets after approximately 0.525M LBM time steps. From this point to the next 0.005M LBM time steps the process of nucleation forms about 37000 droplets, showing an exponential growth, with a shift in volume fraction from 0 to 8%.

| $Run_{id}$ | $L$ | $\delta$ | $\phi$ | $T_{PO}$ | $A \cdot 10^{-6}$ | $< N_d >$ |
|---|---|---|---|---|---|---|
| $N512V15$ | 512 | $1.25 \cdot 10^{-6}$ | 15% | 540400 | 0.485 | 7257 |
| $N512V20$ | 512 | $1.25 \cdot 10^{-6}$ | 20% | 654800 | 0.485 | 5859 |
| $N512V27$ | 512 | $1.25 \cdot 10^{-6}$ | 27% | 778200 | 0.485 | 5729 |
| $N512V34$ | 512 | $1.25 \cdot 10^{-6}$ | 34% | 885000 | 0.485 | 5807 |
| $N512V41$ | 512 | $1.25 \cdot 10^{-6}$ | 41% | 979200 | 0.485 | 5771 |
| $N512V48$ | 512 | $1.25 \cdot 10^{-6}$ | 48% | 1063600 | 0.485 | 5341 |
| $N512V55$ | 512 | $1.25 \cdot 10^{-6}$ | 55% | 1139800 | 0.485 | 4367 |
| $N512V62$ | 512 | $1.25 \cdot 10^{-6}$ | 62% | 1209400 | 0.485 | 3051 |
| $N512V70$ | 512 | $1.25 \cdot 10^{-6}$ | 70% | 1273400 | 0.485 | 1671 |
| $N512V77$ | 512 | $1.25 \cdot 10^{-6}$ | 77% | 1332800 | 0.485 | 806 |

***Table 4.2.:*** *Parameter description for all simulations diplayed in Figure 4.6, showing in particular the injection rate, $\delta$, the magnitude of the forcing, $A$, and including the average number of droplets, $< N_D >$, during the stationary forced phase, after the injection has been stopped at a given LBM step, $T_{PO}$. The table reports values for different volume fractions, obtained using the LBE3D with the parameters reported in Eq. (3.18), defined as standard set at $\rho_0 = 0.83$, indicating the density value above which non-ideal effects come into play for a given specie A or B. The relaxation time is fixed to $\tau = 1$.*

After the initial transient, during which the dispersed droplets are formed, the nucleation cases are mainly dominated in $T_P$ by a decreasing number of droplets, starting from a high number, finally stabilising when the desired volume fraction is achieved. However, this phase requires longer for lower volume fractions. For instance if looking at $T_P$ for the simulations $N512V15$ or $N512V20$, the number of droplets still tend to reduce while it is well stabilised for similar cases at higher volume fraction, i.e. $N512V70$ and $N512V77$. In Figure 4.6 (right), the average radius, $< R >$, for all presented simulations is

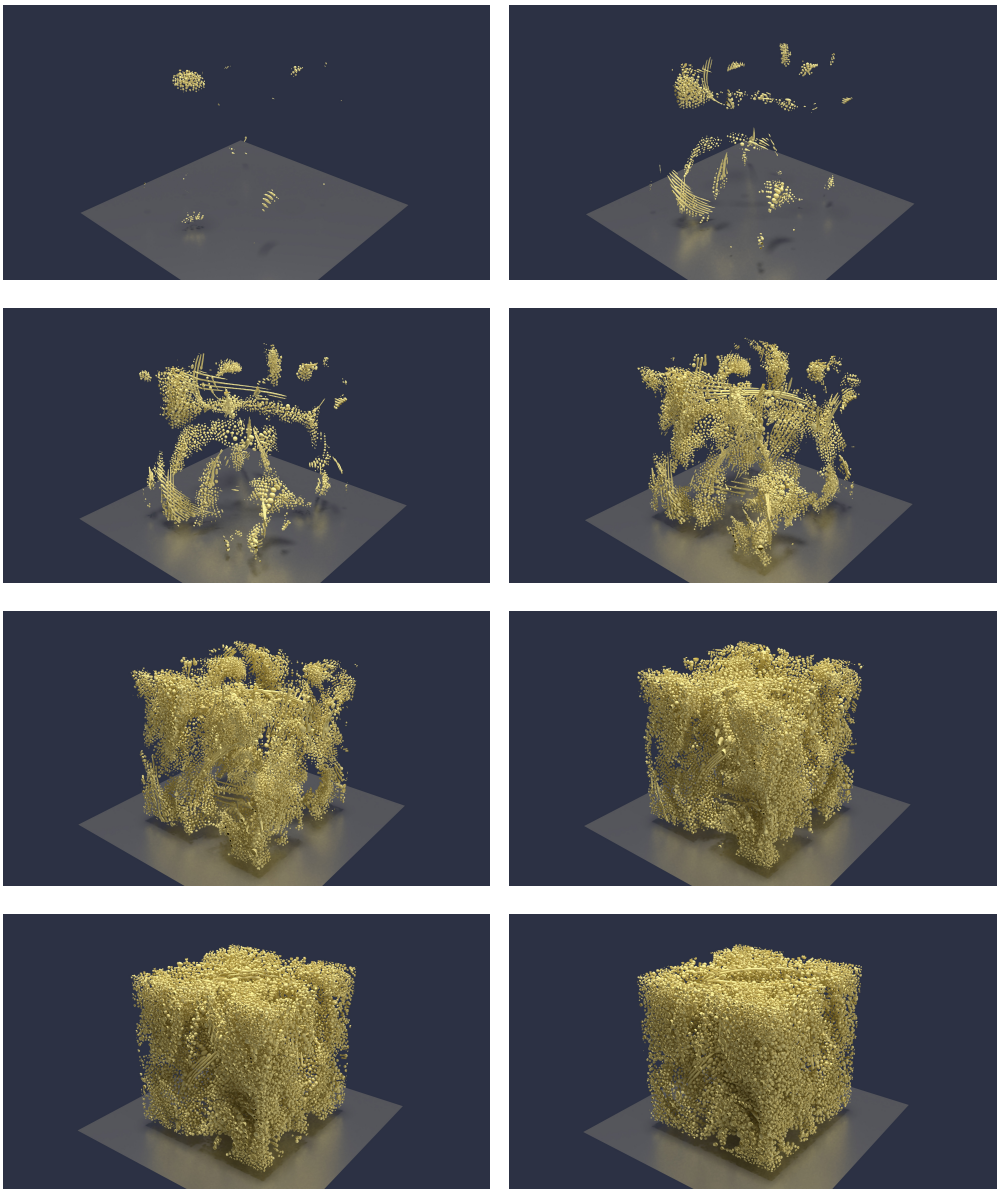


***Figure 4.5.:*** *The different phases during the of nucleation of emulsions starting from $\phi = 0\%$, from a simulation at $512^3$ resolution, using same parameters as for $N512V77$, see Table 4.2. The high quality renderings are reproduced from the published media content in [74] where the dynamics of the full simulation is visible.*

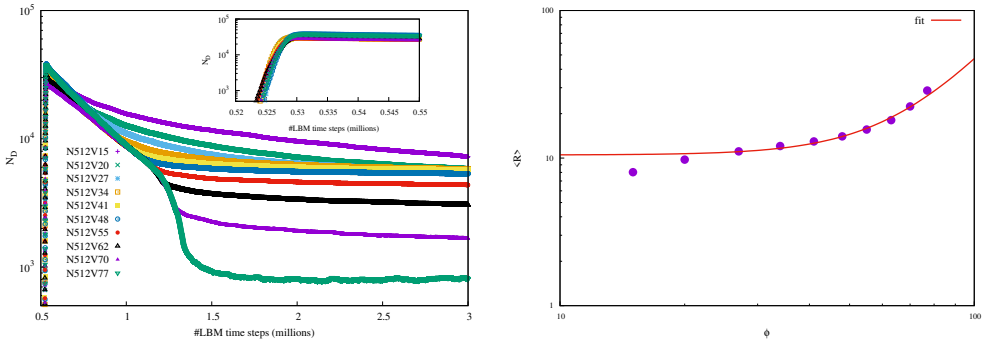



***Figure 4.6.:*** *(Left panel) The total number of droplets, $N_D$, as a function of time for different simulations corresponding to different volume fractions. All simulations are started from nucleation. The inset shows the same data but in log-linear scale to better illustrate the initial transient of droplets formation. The final number of droplets achieved for the all cases in 4.6 is reported in column $N_D$ of Table 4.2. (Right panel) The average droplet size, $< R >$ computed averaging over the $T_S$ phase and as a function of the volume fraction, $\phi$. The line is a fit with scaling $< R >= 1.4 \cdot 10^{-5}\phi^{3.2} + 10.5$.*

reported as a function of the volume fraction $\phi$. The radius of all droplets is averaged from the $T_S$ phase (from 1.5M to 3M LBM steps of simulation) over a number of samples of the order $\sim 10^7$.

## 4.2. Effect of the forcing magnitude

In this section we present results of the effect of the forcing magnitude $A$ on the process of emulsification of dense emulsions. We present data and results for two sets of simulations characterised by different initial conditions. Table 4.3 reports all data for these two sets, with the label $I$ indicating a simulation in the set initiated with interface fragmentation and $N$ indicating simulations in the set initiated with nucleation. Every set includes five simulations where the volume fraction $\phi$ is increased at the high value of 77% for all cases, on a $512^3$ grid. We focus on a range of magnitudes in the interval around the typical forcing $A = 0.485 \cdot 10^{-6}$ utilised for the simulations previously presented at the same resolution. In particular, we present results of emulsifications obtained with a set of five forcing levels from $A = 0.405 \cdot 10^{-6}$ to $A = 0.505 \cdot 10^{-6}$. For all cases a constant forcing is applied throughout an overall simulation time of 3M LBM time steps, with a constantly applied injecting rate of $\delta = 1.25 \cdot 10^{-6}$, up to when a level of 77% volume fraction in the dispersed phase is achieved. As mentioned in the previous section, the final value of the number of $N_D$

remains comparable for cases with different initial conditions. However, there is a substantial variation in $N_D$ at varying the applied forcing magnitude $A$. In Figure 4.7, we show the variation of $N_D$ as a function of time, for different

| $Run_{id}$ | $L$ | $\delta$ | $\phi$ | $T_{PO}$ | $A \cdot 10^{-6}$ | $< N_d >$ |
|---|---|---|---|---|---|---|
| $N512V77\_FA0.405$ | 512 | $1.25 \cdot 10^{-6}$ | 77% | 1332800 | 0.405 | 986 |
| $I512V77\_FA0.405$ | 512 | $1.25 \cdot 10^{-6}$ | 77% | 559900 | 0.405 | 967 |
| $N512V77\_FA0.425$ | 512 | $1.25 \cdot 10^{-6}$ | 77% | 1332800 | 0.425 | 929 |
| $I512V77\_FA0.425$ | 512 | $1.25 \cdot 10^{-6}$ | 77% | 559900 | 0.425 | 911 |
| $N512V77\_FA0.455$ | 512 | $1.25 \cdot 10^{-6}$ | 77% | 1332800 | 0.455 | 856 |
| $I512V77\_FA0.455$ | 512 | $1.25 \cdot 10^{-6}$ | 77% | 559900 | 0.455 | 861 |
| $N512V77\_FA0.485$ | 512 | $1.25 \cdot 10^{-6}$ | 77% | 1332800 | 0.485 | 806 |
| $I512V77\_FA0.485$ | 512 | $1.25 \cdot 10^{-6}$ | 77% | 559900 | 0.485 | 800 |
| $N512V77\_FA0.505$ | 512 | $1.25 \cdot 10^{-6}$ | 77% | 1332800 | 0.505 | 503 |
| $I512V77\_FA0.505$ | 512 | $1.25 \cdot 10^{-6}$ | 77% | 559900 | 0.505 | 761 |

***Table 4.3.:*** *Parameter description for all simulations displayed in Figure 4.7b, showing in particular the injection rate, $\delta$, the magnitude of the forcing, $A$, and including the average number of droplets, $< N_D >$, during the stationary forced phase, after the injection has been stopped at a given LBM step, $T_{PO}$. The table reports values for different volume fractions, obtained using the LBE3D with the parameters reported in Eq. (3.18), defined as standard set at $\rho_0 = 0.83$, indicating the density value above which non-ideal effects come into play for a given specie A or B. The relaxation time is fixed to $\tau = 1$.*

forcing magnitudes $A$, and for the considered two sets of simulations. Looking

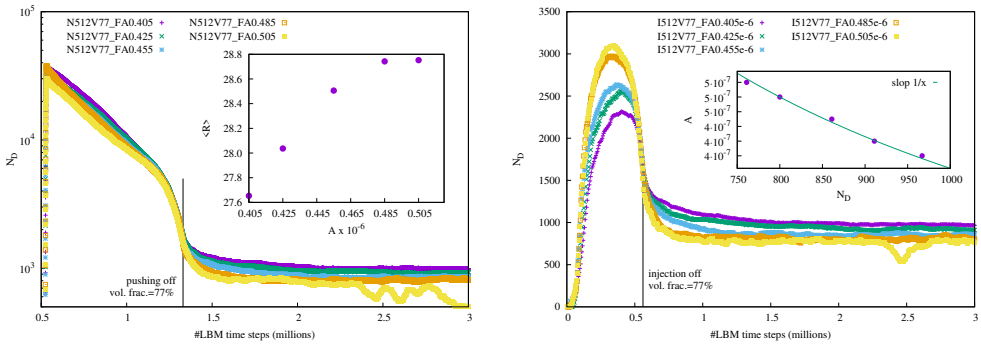



***Figure 4.7.:*** *Total number of droplets, $N_D$, as a function of time for all simulations performed achieving 77% volume fraction in the dispersed phase, at varying the forcing amplitude. The figures show the case of emulsions started via nucleation (left figure) and via interface (right figure). The inset in the figure on the left reports the average droplet radii, $< R >$, as a function of the applied forcing magnitude, A, while the inset in the figure on the right shows how the increasing of the forcing lowers the value of $N_D$.*

at the value of $N_D$ beyond 1.5M LBM time steps, the Figure 4.7 shows a similar behaviour for both sets despite the different initial conditions. This similarity is also due to the high volume fraction, influencing the emulsion composition due to the high-packing. A different behaviour is displayed only for the cases with

$A = 0.505 \cdot 10^{-6}$, clearly presenting some degree of instability, especially in the case of nucleation ($N512V77\_FA0.505$). This effect is further discussed in the next section. Notice, that both $N_D$ and $< R >$ are influenced by the intensity of the hydrodynamic stresses with opposite behaviour at increasing the volume fraction. At low volume fraction a higher forcing produces a larger number of droplets, $N_D$, with smaller average radii $< R >$, as reported in several works [76], and also visible during the initial phase, $T_I$, in Figure 4.7(right), when the volume fraction of the dispersed phase is in the range between 30% and 60%. On the other hand, the insets in Figure 4.7(left) shows an opposite trend for highly packed emulsions, $\phi = 77\%$. In details, the inset in Figure 4.7(right) displays how the value of $N_D$ is reduced at increasing the forcing magnitude showing a trend of approximately consistent with $(1/ < R >)$, when high volume fraction is achieved and the simulation is in forced stationary state. The higher forcing increases the number of droplets while breaking the interface up to a volume fraction of approximately 60%. After this threshold, the higher the forcing, the smaller is the resulting number of droplets in the emulsion. On the other hand, the inset of Figure 4.7(left) displays the average droplets radius, $< R >$, as a function of the applied forcing amplitude. Due to the final instability of $N512V77\_FA0.505$, for all simulations $< R >$ is given by the average value of the radius of all existing droplets for 0.5M LBM time steps, precisely from 1.75M to 2.25M, over a number of samples of the order $\sim 10^6$. The inset displays how the value of $< R >$ increases as a function of the stirring magnitude. In summary, we can say that in the processes of emulsification of dense emulsions, when a high volume fraction of the dispersed phase is achieved, the higher the magnitude of the turbulent forcing, the smaller the number of droplets and the larger the average droplets' size of the resulting emulsion.

## 4.3. Jammed emulsions

In this section we describe a set of three simulations achieving high volume fraction at $\phi = 77\%$ at different resolutions, namely $256^3$, $512^3$ and $1024^3$. The data of all these simulations are reported in Table 4.4 and every simulation is performed with the highest stable forcing magnitude $A$, approximately and empirically found for every resolution by trial and error. In particular, we tested several forcing magnitudes and registered the highest forcing, giving stabilised forced emulsions at high volume fraction, probably too high for the $256^3$ case as it presents more instability during $T_S$, see Figure 4.8. The Figure shows the time evolution of $N_D$ for all resolutions, in relation to the increasing of the volume fraction. Every emulsions is generated from interface fragmentation,

| $Run_{id}$ | $L$ | $\delta$ | $\phi$ | $T_{PO}$ | $A \cdot 10^{-6}$ | $< N_d >$ |
|---|---|---|---|---|---|---|
| $I256V77$ | 256 | $1.25 \cdot 10^{-6}$ | 77% | 560000 | 1.05 | 112 |
| $I512V77$ | 512 | $1.25 \cdot 10^{-6}$ | 77% | 560000 | 0.485 | 857 |
| $I1024V77$ | 1024 | $1.25 \cdot 10^{-6}$ | 77% | 560000 | 0.305 | 5615 |

***Table 4.4.:*** *Parameter description for all simulations displayed in Figure 4.8, showing in particular the injection rate, $\delta$, the magnitude of the forcing, $A$, and including the average number of droplets, $< N_D >$, during the stationary forced phase, after the injection has been stopped at a given LBM step, $T_{PO}$. The table reports values for different volume fractions, obtained using the LBE3D with the parameters reported in Eq. (3.18), defined as standard set at $\rho_0 = 0.83$, indicating the density value above which non-ideal effects come into play for a given specie A or B. The relaxation time is fixed to $\tau = 1$.*

with an initial volume fraction ratio of 30% between the two components as in Figure 4.2a. The three simulations are all constantly forced for the first 1.2M LBM time steps, for a total of simulation of 2M LBM time steps, from 1.2M to 2M LBM time steps the system is not forced. We define the time of a stationary non-forced phase of a given simulation as $T_{SO}$. Figure 4.8 shows how all simulation have similar behaviour in term of the time evolution of $N_D$. When the forcing is turned off the system takes approximately 0.05M LBM time steps to properly stabilise and the value of $N_D$ remaining almost constant at the transition of the two phases: with forcing and with no forcing. The inset of the Figure 4.8 displays how the value of $N_D$ registered at the end of the simulation scales with a trend power-law, as expected. In Figure 4.9 we show the rendering of the final morphology for all cases in Table 4.4, at the end of $T_{SO}$. The high-packing presents non-spherical droplets showing high internal pressure, along with (poly-)dispersity. We measure the droplet size distribution (DSD) at different intervals of the simulation $I1024V77$, showing the temporal evolution of the microscopic composition in the different phases of the process of emulsification. Figures 4.10a and 4.10b display the DSD of the typical phases of simulation with interface fragmentation $T_T$ and $T_I$, respectively. Due to the interface fragmentation the DSD shows the presence of large droplets together with a relevant number of smaller droplets. The distribution seems to be in qualitative agreement with a -10/3 power-law scaling, also reported in several works on turbulent emulsions [77, 78], possibly first introduced in [79], on the basis of experimental observation of bubble entrainment in the breaking of marine waves. At increasing the number of LBM time steps of the simulation, the extreme part of the plot tail, towards large droplets radii, tends to the average diameter, possibly creating a double slope where the upper part fits the -2/3 power-law while it get much steeper, approximately -9 for large droplets. In the latest part when high volume fraction is achieved, the two slops tend to uniform due to the increasing presence of larger droplets. At the highest volume

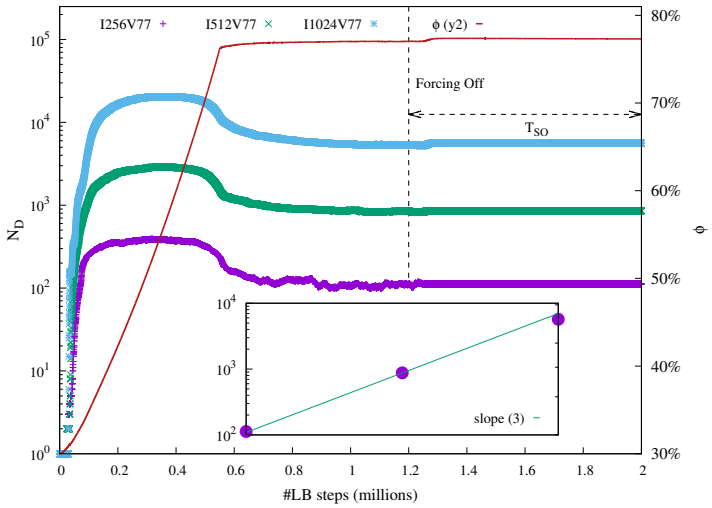


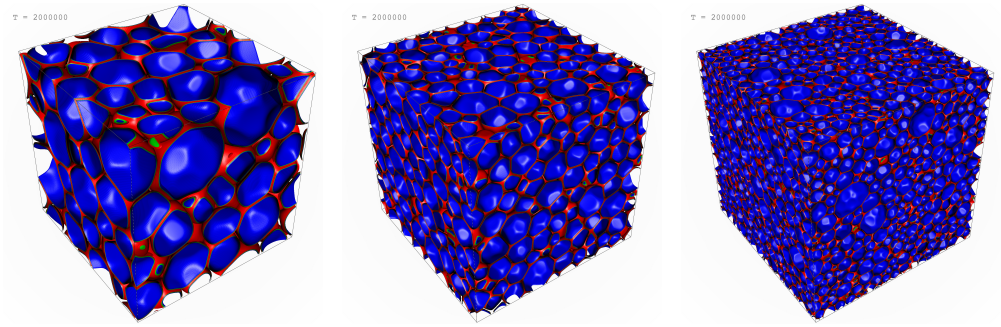

***Figure 4.8.:*** *The value of the number of droplets, $N_D$, as a function of time for different resolutions , namely $256^3$, $512^3$ and $1024^3$. In the inset the scaling of the final number of droplets present at the end of the simulations. Being the forcing magnitude not the same for all cases, the final number of droplets slightly differs, however it is visible how at increasing resolution the number of droplets follows the expected power-law 3 scaling.*

fraction, beyond 70%, the distribution shows a bimodal shape with a secondary peak towards 1/10 of the average diameter, evidence of a relevant presence of small droplets. This effect is attributed to the frequent fragmentation of large droplets during the chaotic motion as the bimodal distribution disappears when the forcing is finally turned off.

### 4.3.1. Stability of emulsions

We discussed how the small-scale morphology, and particularly the droplet number and size, is effected by large-scale quantities such as the volume fraction and the stirring amplitude. Here, we analyse how the small-scale morphology impacts on the fluid flow. To this end, we perform rheological test applying to the emulsion at rest an external periodic forcing in the x-direction of the form $F_x(y) = A_K sin(k_f y)$, with a wave number $k_f = \frac{2\pi}{L}$, as reported in [38] for 2-dimensional jammed systems. The forcing amplitude $A_K$ produces a sinusoidal Kolmogorov flow of the kind $U_x \propto \sin(2\pi y/L)$. In Figure 4.11a we show the actual space-time averaged velocity profile:

$$U_x(y) = \frac{1}{T}\int_0^T u_x(x,y,z,t)dxdzdt, \qquad T > 1 \tag{4.3}$$



**(a)** $I256V77\_SO2\_FA1.05$ **(b)** $I512V77\_SO2\_FA0.485$ **(c)** $I1024V77\_SO2\_FA0.305$

***Figure 4.9.:*** *Via turbulence stirring we could produce 77% volume fraction emulsions at different numerical resultions, from left to right, $256^3$, $512^3$ and $1024^3$. In the picture we compare the final configurations at high volume fraction, showing high packing and non-spherical droplets in the final non-forced steady state stage of the emulsions.*

as for different forcing intensity $A_K$, ranging from $0.5 \cdot 10^{-6}$ to $2.25 \cdot 10^{-6}$, applied to the same initial configuration, $I512V77\_SO2M\_FA0.485$, at rest. The Figure 4.11a shows a flattening of the velocity profile in the central region is clearly observed, for all values of $A_K \leq 1.75 \cdot 10^{-6}$, evidence of non-Newtonian behaviour [80]. Here the forcing is applied for 1M LBM time steps and data averaged in time, $T$, as indicated in Eq. 4.3. On the same configurations we compute the value of the shear as the spatial derivative of the time averaged velocity fields, $\gamma(y) = \partial_y U(y)$, and the value of the stress induced from Eq. (4.3) as $\sigma = \int_0^y F_{xz}(y')dy'$. In Figure 4.11b we show a scatter plot of the stress, $\sigma$ as a function of the applied shear, $\gamma$, showing that the jammed emulsion system flows, $\gamma \neq 0$ only above a critical threshold (yield stress, $\sigma_Y$) of the order of $\sigma_Y \sim 4 \cdot 10^{-5}$, with all various cases showing similar curves, but with longer tails at higher forcing. To confirm the evidence of the presence of yield stress at the highest volume fractions we define an effective global shear as follows:

$$\dot{\gamma}_{\text{eff}} \equiv \frac{\langle \sigma(y)\dot{\gamma}(y) \rangle}{\langle \sigma^2(y) \rangle^{1/2}} \tag{4.4}$$

where the average is meant to be taken over $y$. Analogously, the effective shear stress will be $\sigma_{\text{eff}} = \langle \sigma^2 \rangle^{1/2} = \left(\frac{L}{2\pi}\right) \frac{A_K}{\sqrt{2}}$. So, in essence, from every simulation, with a certain forcing amplitude, $A_K$, we are able to extract a couple of values $\dot{\gamma}_{\text{eff}}$ and $\sigma_{\text{eff}}$. The resulting flow curves are shown in Figure 4.12 for both low and high volume fractions. The Figure 4.12 shows how $\sigma_{\text{eff}}$ tends to 0 as $\dot{\gamma}_{\text{eff}}$ for lower volume fraction, $\phi$, while it reports a finite value of $\sigma_Y$ for the

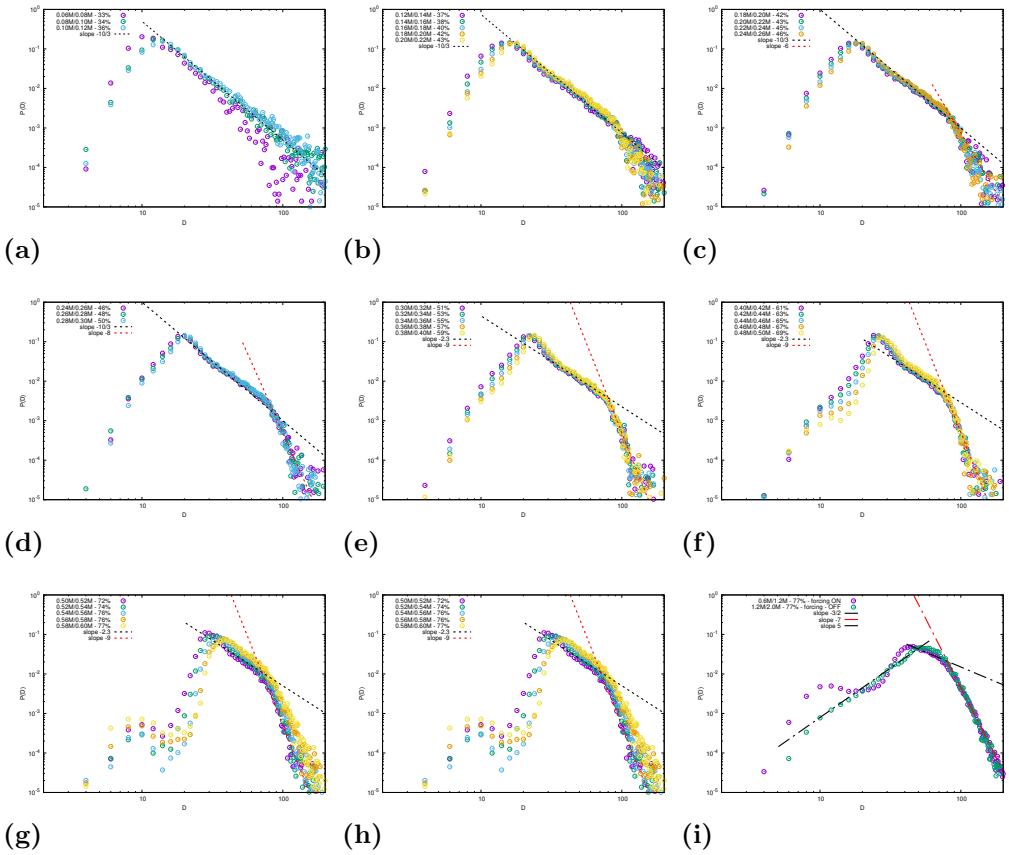
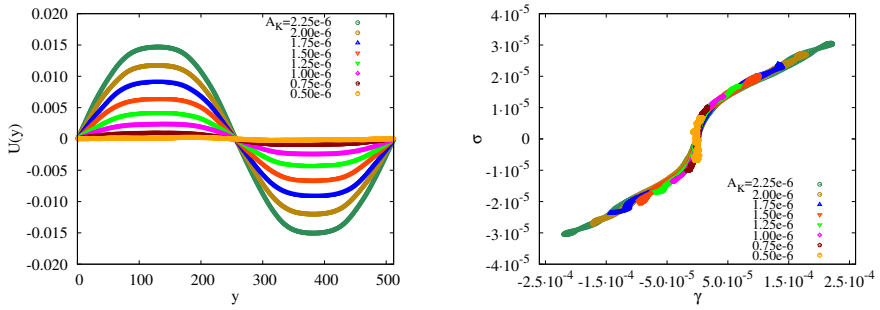



***Figure 4.10.:*** *The Figures shows in log-log scale the temporal evolution of the droplets size distribution (DSD) for the simulation $I1024V77$. The DSD is the PDF of the effective droplet diameter (i.e. the diameter of a sphere with the same volume), and normalised to have area one (as for probability distribution function). In particular, the plots report, for every dataset, the time interval range in millions LBM time steps, and the volume fraction of the dispersed phase. During the initial transient of interface fragmentation the DSD displays a typical slope close to -10/3 which gradually convert towards a much steeper slope as the stirring completes the process of interface fragmentation. At the highest volume fraction, e.g. beyond 70%, the DSD assume a bimodal shape, with a secondary peak at smaller radii evidence of the increasing presence of small droplets. This effect tends to vanish when the forcing is eventually switched off.*




**(a)** Velocity profile **(b)** Yield stress

***Figure 4.11.:*** *a) Average (over time and space) velocity profile for different values of the external sinusoidal forcing applied to an emulsion in a jammed state as obtained after a chaotic large-scale forcing is applied. b) Average stress as a function of the observed shear when an external periodic sinusoidal forcing is applied. The effect of the amplitude of the sinusoidal forcing, A, is shown and a clear non-Newtonian rheological behaviour is found.*

highest volume fraction for $\dot{\gamma}_{\text{eff}} \to 0$, fitting the Herschel-Bulkley relation, see Chapter 2 with the exponent $\beta \sim 0.5$, also reported by [38], showing a power-law dependence of the shear stress as a function of strain rate, only beyond a given yield stress threshold.

## 4.4. When things go wrong

In the emulsification process presented, two main parameters control the making of a stabilised dense emulsions: the forcing magnitude $A$ (Eq. 3.19) and the amount of the dispersed component injected $\delta$ (Eq. 4.2). Both parameters impact on the input energy when achieving higher volume fractions, with the forcing magnitude influencing the behaviour also when the desired volume fraction is reached and the emulsion is flowing in a stable chaotic motion. We empirically found a proper setting for all presented resolutions but experienced several cases, that lead to instabilities or even catastrophic effects. By fixing $\delta = 1.25 \cdot 10^{-6}$ for all simulations we could minimise this instability.
The catastrophic event of phase inversion is when the dispersed phase, being the major component reverses, becoming the contiguous phase, surrounding droplets of the fluid in the minority phase: i.e., from an emulsion of oil in water to an emulsion of water in oil as for our cases. In other words, when a phenomena of phase inversion occurs, the component previously in the form of dispersed droplets gradually becomes the contiguous film at the surface and vice-versa. The Figure 4.13 presents the renderings of three different temporal

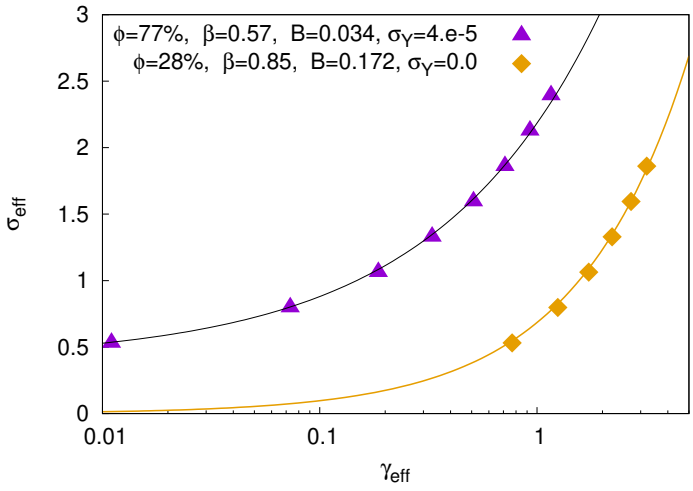
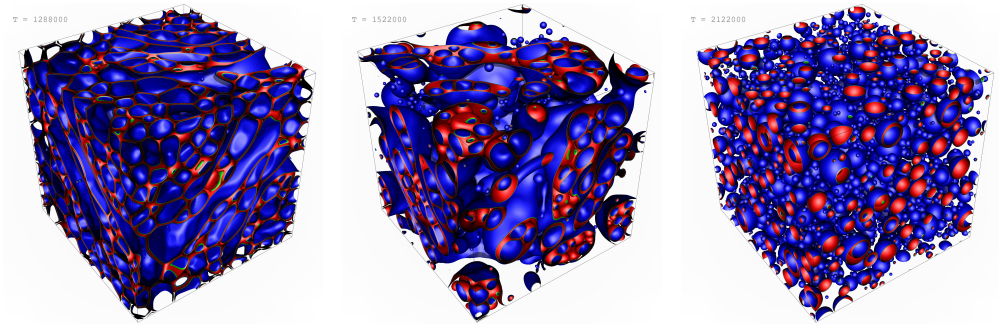



***Figure 4.12.:*** *The stress $\sigma_{eff}$ in function of $\dot{\gamma}_{eff}$ as result of a rheological Kolmogorov tests at different forcing amplitudes, for different volume fractions. The points can be fitted via the Herschel-Bulkley relation $\sigma_{eff} = \sigma_Y + B\dot{\gamma}_{eff}^{\beta}$ for different values of $\beta$ as reported in the caption, showing the presence of a finite yield stress for the case corresponsing to the highest volume-fraction. For $\phi = 77\%$ the fit provides a value of $\beta \sim 0.5$, consistent with what reported in [38].*

steps of a sequence of inversion. Here, a highly dense emulsion exposed at a too high forcing, gradually leads to the inversion: first to an unstable composition, including creations of double emulsions [81], then resulting to a phase inversion. From the rendering, the inversion is also visible by the swap of the positions of colours representing the two components, respectively. In Figure 4.13a the internal component of the droplet is the blue, with the red colour being the external component. Nevertheless, Figure 4.13c shows the colours fully reversed 4.13c, while Figure 4.13b displays the process of transition where the two colours are both concurrently present internally and externally to the droplets. Between Figures 4.13a and Figure 4.13c, it is notable also the difference in the droplet size distribution. The inverted system in Figure 4.13c presents droplets in the minority phase (approximately 20% volume fraction) with smaller volumes, as well as more spherical shape, if compared with Figure 4.13c. In Figure 4.14 we show the process of phase inversion, providing a qualitative description of the internal morphology of the emulsion. We plot the time evolution of the number of droplets together with the volume fraction of the two components, for two simulations. The value of $N_{D1}$ and $N_{D2}$ are used to indicate the number of droplets of the two components, while $\phi_1$ and



**(a)** **(b)** **(c)**

***Figure 4.13.:*** *The renderings capture three different moments of the phenomena of catastrophic phase inversion occurring during the making of a highly packed dense emulsion. On the left the dense emulsion with 77% volume fraction is exposed to a chaotic stirring and is already displaying the formation of large bubbles that gradually coalesce and include smaller bubble (double emulsion) up to the point displayed in the central picture where the phase inversion is about to happen. This frame displays really irregular structure, including formation of a double emulsions. On the right we show the final stage of the simulation when the forcing is turned off. The colours underline the phase inversion too, with blue colour being internal to the bubbles on the left and vice-versa on the right. The resulting system has 22% of water dispersed in a contiguous film of oil.*

$\phi_2$ are used to indicate the volume fractions, respectively. The simulations are characterised by having the same settings but varying the forcing amplitude by a factor 2. The parameters set of the simulation reported in Figure 4.14a is the same for case a) in Table 4.4, while a forcing magnitude $A = 2.05 \cdot 10^{-6}$ is used for the simulation in Figure 4.14b. Here we run only 0.6M LBM time steps with a constant forcing on a discretised $256^3$ grid, and starting from a flat separation of the two component at 30% of volume fraction. In stable emulsions only $N_{D1}$ is expected to increase while $N_{D2}$ is expected to be at a constant value of 1, see Figure 4.14a, commonly to a lower value if compared with the number of droplets of the dispersed component initially forming the dispersed phase. In Figure 4.14b a sample of phase inversion is shown. Due to the higher forcing, the interface is quickly fragmented and the number of droplets rapidly increase to a much higher value, if compared with the case in Figure 4.14a. However, already at volume fraction of 35%, the number of droplets irreversibly start to decrease, with the simultaneous increasing of the number of droplets in the second component, when a volume fraction of around 70% is reached. From this stage onwards the droplets of the second component rapidly increase completing the inversion in less than 0.1M LBM time steps. The two renderings

in Figure 4.14c and Figure 4.14d report the states of the two simulations at the LBM time step 0.54M, for a visual analysis of the morphology of the two emulsions for the two discussed cases. Therefore, the formation of droplets in

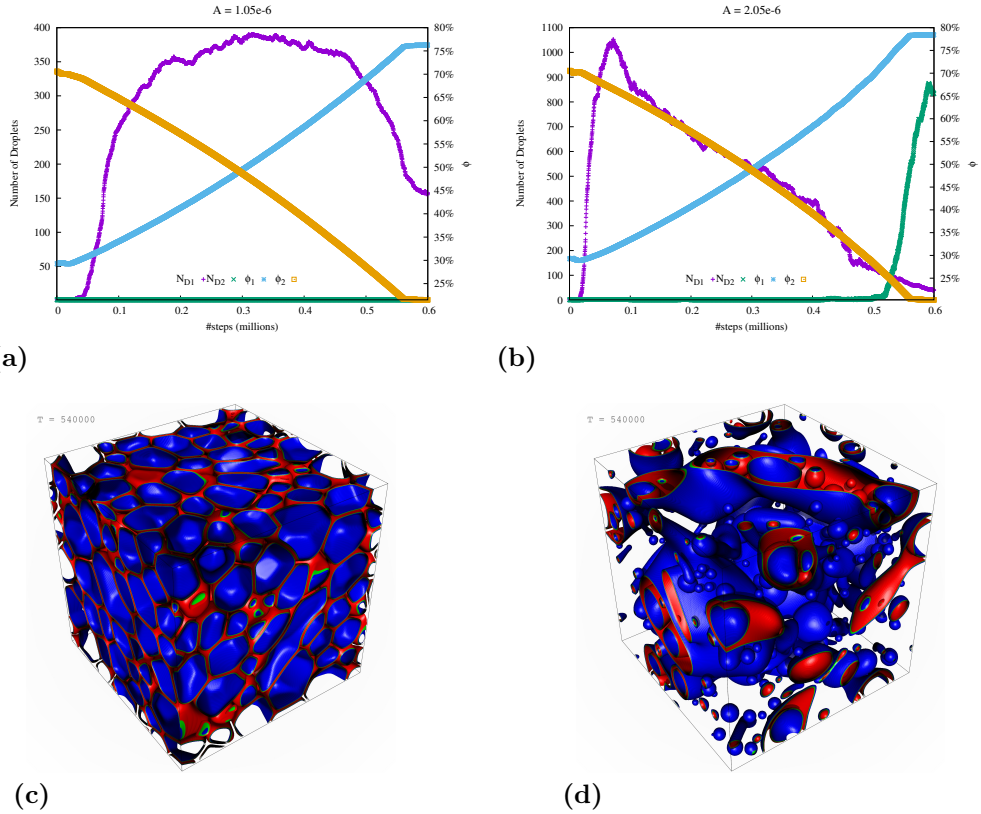
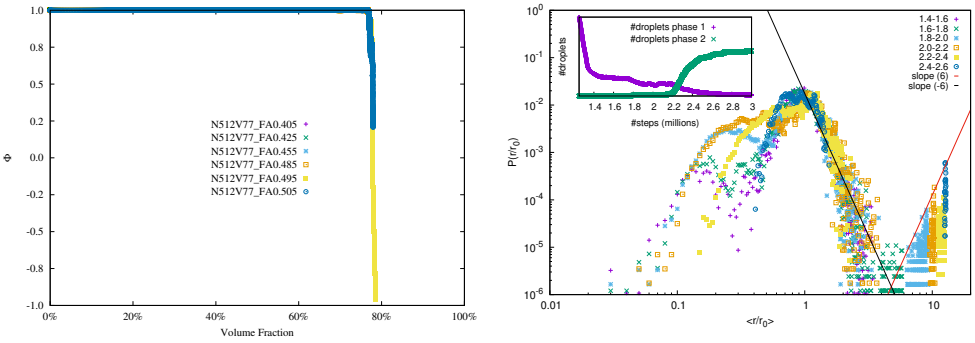



***Figure 4.14.:*** *Qualitative description of a phase inversion happening due to a too intense forcing magnitude. A stable simulation (left) displays a value of $N_{D1}$ showing that the component initially in minority phase, turn in the majority phase without causing instability as the value of $N_{D2}$ remains close to 1. A unstable simulation (right) displays the value of $N_{D2}$ rapidly rises during the catastrophic phase inversion. The two rendering in 4.14c and 4.14d show the status of the emulsion at the 0.54M LBM time step.*

the second component is evidence of some instability. To provide a qualitative analysis of phase inversion we define the quantity $\Phi$ as:

$$\Phi^{(t)} = \frac{N_1^{(t)} - N_2^{(t)}}{N_1^{(t)} + N_2^{(t)} - 2} \tag{4.5}$$

where $N_1^{(t)}$ and $N_2^{(t)}$ are the number of droplets in the two components, respectively. When $N_1^{(t)} > N_2^{(t)}$, the value of $\Phi$ indicates stability for $\Phi = 1$, while it indicates instability with $0 < \Phi \leq 1$, as this is given by the formation of droplets of the component initially in the form of a contiguous film, or phase inversion $-1 \leq \Phi \leq 0$. In Figure 4.15 we plot the evolution of $\Phi$ for all simulations in Figure 4.7a, adding to the set, one more case with $A = 0.495 \cdot 10^{-6}$ that rapidly leads to phase inversion, namely $N512V77\_FA0.495$. Surprisingly, this case displays phase inversion earlier than any other. From our experience, for the same setting used for simulations in Figure 4.7a, we expect phenomenas of instability with a random probability in any range of forcing amplitudes $0.485 \cdot 10^{-6} \leq A \leq 0.505 \cdot 10^{-6}$, with almost 100% probability of phase inversion above this threshold and the same level of probability to obtain stable highly-dense emulsions below this threshold. In Figure 4.15b we show the small




**(a)** **(b)**

***Figure 4.15.:*** *(Left panel) The quantitative for phase inversion, $\Phi$, defined as Eq. (4.5), at increasing the forcing amplitude for all simulation in Figure 4.7a, adding to the latest set the case $N512V77\_FA0.495$, leading to phase inversion. (Right panel) Detailed (DSD) for distinct time intervals of a phase inversion. Initially the emulsions shows a bimodal distribution similar, to the one reported in Figure 4.10i, typical of highly packed emulsion in forced steady state. When the emulsion is almost reversed the dual peak of the bimodal distribution merge together, while the tail of large droplets separates from the distribution creating a gap between average droplets sizes and droplets about a factor 10 larger. This is the beginning of the inversion that finally leads to the presence of only large droplets, that will become a unique contiguous film of fluid. The inset of Figure 4.15b shows the temporal evolution of the number of droplets, $N_D$, of the two fluid components.*

scale morphology (DSD) during the phenomena of phase inversion. Initially the emulsions shows a bimodal distribution similar, to the one reported in Figure 4.10i, typical of highly packed emulsion in forced steady state. While the emulsions composition tends to reverse, we notice a constant increase of

the number of small droplets, likely in the from of multiple emulsion, together with the appearing of larger droplets. When the emulsion is almost reversed the dual peak of the bimodal distribution merge together, while the tail of large droplets separates from the distribution creating a gap between average droplets sizes and droplets about a factor 10 bigger. This is the beginning of the inversion that finally leads to the presence of only large droplets, that will become a unique contiguous film of fluid. The inset of Figure 4.15b shows the temporal evolution of the number of droplets, $N_D$, of the two fluid components. However, even finding a consistent couple of the injection rate, $\delta$, and of the forcing amplitude, $A$, the highest volume fraction reachable in the dispersed phase remains limited, beyond which it is hard to achieve stable emulsions. As for our simulation we have defined this limit at 77% volume fraction in the dispersed phase. We performed a set of simulations, with similar configuration of the set presented in Table 4.1, to achieve this statement. Figure 4.16 shows (upper panel) the time evolution of the number of droplets for several level of volume fraction, with two of the six simulation in the set, displaying instability ($\Phi < 1$) and phase inversion. In the same figure (lower panel) we show the evolution of $\Phi$ while reaching high volume fractions, with several attempts to extend the volume fraction beyond 77% leading to phase inversion. This set of simulations is generated from an initial interface between the components with a 40% volume fraction. We report values of $\Phi < 1$ for any volume fraction $\phi \geqslant 80\%$. After these initial experiments we considered $\phi = 77\%$ as the maximum volume fraction beyond which the emulsion would not stabilise in the long term, and lead to phase inversion.

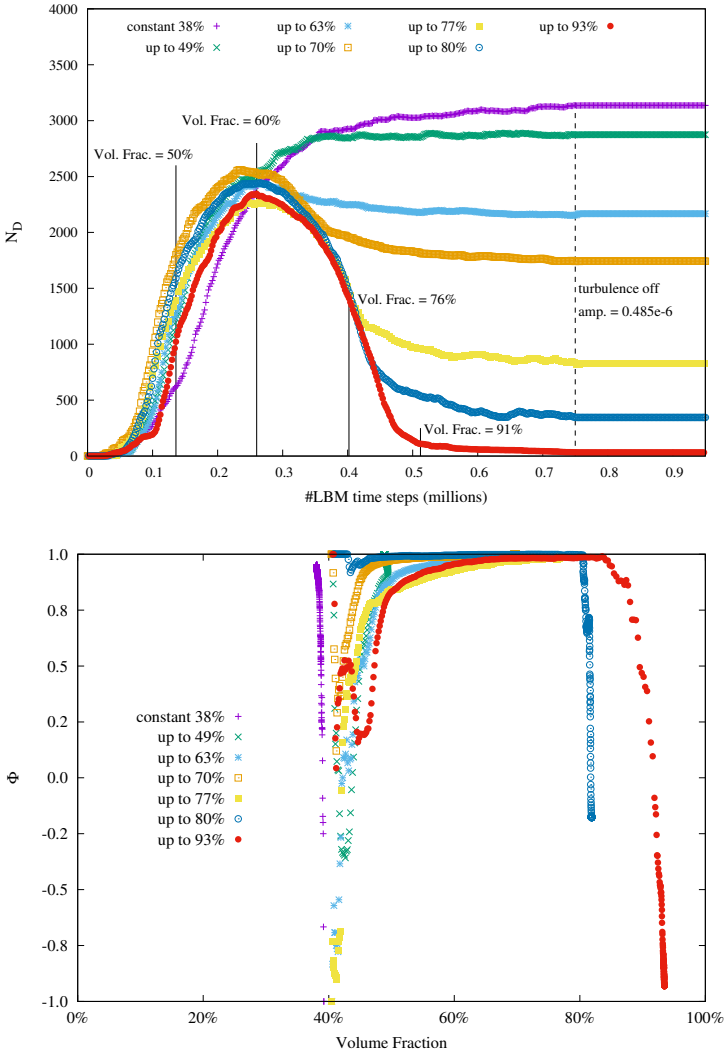



***Figure 4.16.:*** *On the upper panel, the number of droplets, $N_D$ as a function of time for different emulsification processes at varying the volume fraction of the dispersed phase. On the lower panel, $\Phi$, as described in Eq. 4.5, at the increasing of the volume fraction for all simulations displayed on the upper panel. The figure shows how any tentative beyond 77% leads to phase inversion.*

# 5. Droplets dispersion in dense emulsions

In this chapter we describe a novel computational approach for tracking droplets in dense emulsions [1]. This tool allows us to identify every single droplet present in a dense emulsion, either flowing in chaotic motion or when almost at rest in a non-forced state. This tool is extended with a tracking algorithm to reconstruct droplets' Lagrangian trajectories during the full process of emulsification. Thanks to this tool we can analyse the absolute dispersion of the droplets which are here quantified and discussed in term of the PDF of droplets acceleration and velocities, and Lagrangian structures functions of droplets dispersion. In additional to the tracking, our tool allows to accurately identify breakup and coalescence events of which we provide a preliminary analysis.

## 5.1. Tracking of droplets in emulsions

The tracking algorithm described in this section relies on the fact that the two components of a binary emulsion are represented by a spatial variation of the density values, with one of the components in the form of dispersed droplets surrounded by a continuous film of the other. Therefore, while we apply this tracking algorithm to the output of an LBM simulation, this could find a more general application to other numerical approaches used to study emulsions via computer simulations and based on Eulerian density representation of the emulsion. The algorithm can be seen as composed of two parts: first we identify and extract every single droplet dispersed in the contiguous phase, and second, we track their motion in space and time. More specifically, based on percolation theory, the Hoshen-Kopelman algorithm [82] is used to identify individual droplets, defined as a closed cluster of density values above a given threshold $(\rho_1 - \rho_2)/2$. We implement the cluster multiple labelling technique for 3-dimensional systems to select every droplet $k$ in the dispersed phase,

[1] This chapter closely follow the contents of Girotto et al. "Droplets dispersion in dense emulsions via fully resolved simulations" in preparation and the content of Girotto et al. "Droplets tracking in dense emulsions" submitted abstract to the 30th International Conference on Discrete Simulation of Fluid Dynamics.

as presented in [83]. In particular, the algorithm processes the density field throughout multiple times, labelling all grid points included in a given droplet $k$ of volume, $V_k$, with unique integer, from 1 to $N_D$, where $N_D$ indicates the total number of droplets present in the emulsion. In other words, the output of the algorithm is a copy of the density field, where every density values included in a position within a droplet volume is replaced by a unique integer identifier of the droplets, while it is set to zero otherwise. Accessing this new data structure, including the droplets labelling, it is now simpler to identify closed droplets in a 3-dimensional density field, even in the case of a field representing a dense emulsion. In [74] we first demonstrated the effectiveness of the clustering algorithm, showing that we can fully decompose the emulsion in the set of individual droplets present in the process of emulsification. Consequently, this constitutes a powerful tool allowing to extract the physical information relative to every single droplet such as its position, size and shape. We report here, in Figure 5.1, only few relevant snapshots of the full movie included in [74]. Being

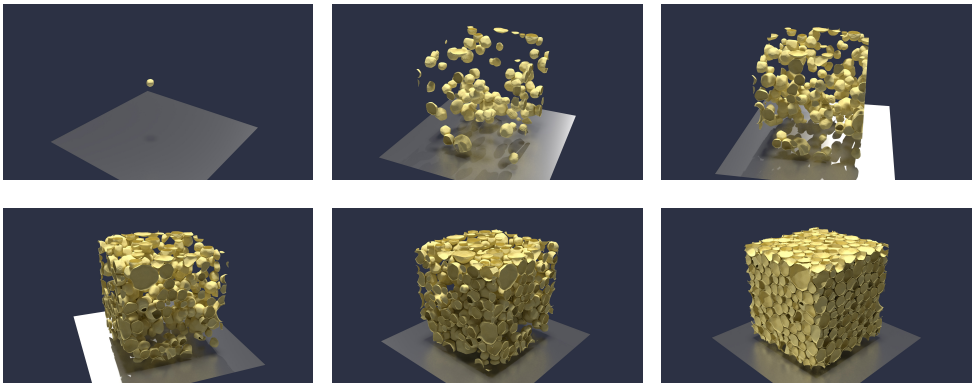


***Figure 5.1.:*** *Representation of the effectiveness of the cluster multiple labelling technique applied on a $512^3$ lattice. After having fully decomposed the dispersed phase (up to a single droplet level, top-left) we recompose it, showing the emulsion morphology is fully re-built consistently (bottom-right). The Figure shows only few relevant renderings in temporal sequence, while the full video is included in [74].*

able to identify the single droplets at different time steps, introduced in [84], does not solve the problem of reconstructing the Lagrangian trajectories. The clustering algorithm distinguishes droplets in the emulsion at the different time steps, by assigning a unique identifier to every single droplet with no relation between the different time steps. This means that same droplet labelled as 203 in the LBM time step 100 could be labelled by the algorithm as 27 at the LBM time step 101. Therefore, given all droplets at all time steps, we still need to

solve the tracking problem.
We developed, therefore, a new algorithm to track droplets in dense emulsions across different time steps, inspired by similar approaches used in experiments (namely, particle tracking velocimetry, PTV). The approach is to perform density snapshots of the density field and apply the clustering algorithm to the 3-dimensional system, for all different and consecutive steps. The two density snapshots are then virtually overlapped defining, for every droplet, $k$, its overlap with a droplet, $n$, at the successive time step. Therefore, the accuracy in the reconstruction of a droplet trajectory between to different successive temporal snapshots, therefore, also depends upon the time distance between snapshots. On the other hand, the clustering algorithm has a relevant computational cost, requiring a non-trivial processing of the density field, on a distributed 3-dimensional grid of processes, contrasting with the fact that the higher the frequency at which the clustering is performed (ideally at every LBM time step), the higher is the accuracy of droplets' tracking. We stress, one more, that our aim is to be able to simulate processes of emulsification for millions of LBM time steps and, therefore, finding a trade-off between the computational cost of the numerical model and the computational cost of the droplet tracking algorithm is essential.
We now describe how the tracking is concretely performed. Let us suppose in the domain at time $t_1$ there are $N_1$ droplets and at (an immediately later dump) $t_2$ there are $N_2$ droplets. We first round the background continuum density field to a 0 value (i.e. in all lattice points where there are no droplets), and the lattice sites inside the droplets are assigned a density volume of 1. We then define $\rho_k(\boldsymbol{x}, t)$ as a "density" field that is equal to 1 only in the lattice sites, $x$, that are inside the droplet, $k$, at time, $t$, and equal to zero everywhere else. The initial state representing the single droplet $k_1$ at time $t_1$ is denoted in the ket notation (reminiscent of quantum mechanics states) as $|k_1, t_1\rangle$ and the final state corresponding to a droplet $k_2$ at time $t_2$ is represented by the following bra notation: $\langle k_2, t_2|$. Both bra and ket are conveniently normalised by the square-root of the volumes of droplets $k_1$ and $k_2$, respectively: $\sqrt{V_1}$ and $\sqrt{V_2}$. We want to define a transition (or better re-identification) probability in order to track droplets in time, including the possibility to identify coalescence and breakup events. The transition probability is given by the following bra-ket

expression:

$$P_{k_1 \to k_2} = \langle k_2, t_2 | k_1, t_1 \rangle = \frac{1}{V} \int \rho_{k_2}(\boldsymbol{x}, t_2) \rho_{k_1}(\boldsymbol{x}, t_1) d^3x$$
$$with \tag{5.1}$$
$$V = \sqrt{V_1 \cdot V_2}$$

This transaction probability is equal to 1 in the case droplets $k_1$ and $k_2$ perfectly overlapped and it is zero if they do not overlap at all. A high $P$ value gives us, therefore, the confidence in having re-identified the same droplet at two different time steps. What happens if a droplet is not deforming and just translating with uniform velocity $\boldsymbol{v}$? We expect that the probability will decrease due to an imperfect overlap between the droplet $k$ at time $t$ and the same droplet $k$, shifted by $\boldsymbol{v} \cdot \delta t$, at time $t + \delta t$. Therefore, we expect that the maximal correlation will occur for:

$$\langle shift(k, \boldsymbol{v} \cdot dt), t + \delta t | k, t \rangle = \frac{1}{V} \int \rho_k(\boldsymbol{x} - \boldsymbol{v} \cdot dt, t + \delta t) \rho_k(\boldsymbol{x}, t) d^3x \tag{5.2}$$

Of course the amount of this effect is proportional to $\delta t$. In order to (partially) reduce the effect due to the translation of the droplet we implement a sort of Kalman filtering, evaluating the overlap against the final derivatives at the same $t + \delta t$ shifted backwards by $\boldsymbol{v} \cdot \delta t$. During the simulation we save on disk, for each time step $t_1$, a 2-dimensional probability matrix of size $(N_1, N_2)$. We define this matrix as the $OverlapMatrix$, constructed as described in Algorithm 1, for droplets distributed among different physical processors. The information included in the $OverlapMatrix$ files are the base to reconstruct the Lagrangian trajectories, even for long simulations. For a given LBM time step $t$, of a given simulation, we read a pair of $OverlapMatrix$ files, corresponding to $t$ and $t + 1$, to reconstruct the droplet trajectories. We start from an initial set of droplets identifiers (IDs) ranging from 1 to $N_1$, created at the initial step of the tracking, by incrementally assigning a new ID for every newly found droplet, throughout the LBM time steps. A newly occurring droplet is expected to be generated by an event of either breakup or coalescence, therefore this allows us to detect those events. More precisely, starting from time step $t_1$ we first check if and which of the $N_1$ droplets still exist at time $t_2$. When a droplet is re-identified (this is assessed based on the probability $P_{k1 \to k2}$ being higher than some $x|$ threshold), it is "consolidated", and its Lagrangian trajectory is incrementally stored along with relevant droplet properties such as its centre of mass (CoM), volume, and velocity. For the droplets not "consolidated", we seek

**Algorithm 1:** $OverlapMatrix$[t, t+1]. A pseudocode illustrating how the overlap matrix is computed. Starting from the continuous density field, $c(\boldsymbol{x}, t)$, we define $N$ fields, $c_k(\boldsymbol{x}, t)$, such that $c_k(\boldsymbol{x}, t) = 1$ inside the droplet $k$ and 0 outside, and where N is the total number of droplet at the time step $t$. The algorithm computes the normalised scalar product $\langle k_2, t_2 | k_1, t_1 \rangle$, normalised by the product of the square roots of the volumes of the droplets $k_1$ and $k_2$ and, additionally, by considering the spatial shift given by the Kalman filtering, in order to improve the accuracy for larger intervals between the time steps.

```
for all k, 1 → N do
    origin = c[ t, k ]
    destination = c[ t+1, k ]
    ovelapmatrix[ origin, destination ] += 1
end for
```

first if the sum of the probability relative to a pair of initial non-consolidated droplets existing at time $t_1$ towards the same single droplet existing at time $t_2$ is above a fixed threshold. All droplets at time $t_2$ satisfying this condition will be labelled as a new droplet, generated by the coalescence of two droplets existing at time $t_1$. Complementarily, for all droplets that are not consolidated at $t_1$ we seek among the droplets existing at time $t_2$, if there exists a pair of droplets such that the probability towards these two final droplets is above a fixed threshold. All droplets existing at time $t_2$ satisfying this condition will be labeled as a new droplets, generated by the breakup event of a droplet existing at time $t_1$. With this algorithm we can reconstruct, both the trajectories of all droplets as well as binary coalescence and breakup events in dense emulsions. The main limitation of the presented algorithm is that it misses events involving more than 3 droplets. In fact only binary events are detected, such as, e.g., when a droplet breaks into two droplets or when two droplets coalesce into a single droplet. In Figure 5.2 we report results on the analysis of the accuracy of the presented algorithm showing the details of the tracking for a reference simulation, $I512V77$ Table 4.4, but constantly forced for 3M LBM time steps. The tracking is performed on the base of dumps performed with a frequency of 100 LBM time steps. This frequency of dumping was chosen in order to maintain a balance of about 50% between the computational cost of the tracking (including the I/O of the $OverlapMatrix$) and the computational cost of running the simulation of the model. In the inserts of Figure 5.2 we

**Algorithm 2:** The Lagrangian tracking algorithm

```
for all k_1 droplets in N_1 do
    for all k_2 droplets in N_2 do
        if P_{k_1→k_2} > Threshold then
            {the droplet k_1 at the time t_1 is the droplet called k_2 at the time t_2}
            (k_2, t_2) ← (k_1, t_1)
        end if
        if P_{k_1→k_2} < Threshold then
            for all k_α droplets in N_1 do
                if V_{k_1} + V_{k_α} == V_{k_2} then
                    {droplets k_1 and k_α at the time t_1 coalesced to form droplet k_2 at the time t_2}
                    (k_2, t_2) ← (k_1, k_α, t_1)
                end if
            end for
            for all k_α droplets in N_2 do
                if V_{k_2} + V_{k_α} == V_{k_1} then
                    {droplets k_1 at the time t_1 broke into to (main) parts, namely droplets k_2 and k_α
                    at time t_2}
                    (k_2, k_α, t_2) ← (k_1, t_1)
                end if
            end for
        end if
    end for
end for
```

show the cumulative number of registered events, together with the number of lost events (non-registered). The result shows how the number of lost events increases during the phases with more intense dynamics, i.e. during the phase of fragmentation of the initial interface. From a maximum of $\sim 5$ lost events per time step during, $T_S$, the number goes up to 8 for the initial transient, $T_T$, of interface fragmentation. To quantify the accuracy of the algorithm we define the statistical error for a given time interval of a simulation, during tracking, as $\epsilon_t = E_l/(E_r \cdot LBsteps)$. Where $E_r$ is the total number of breakup and coalescence events registered by the tracking algorithm and $E_l$ is the total

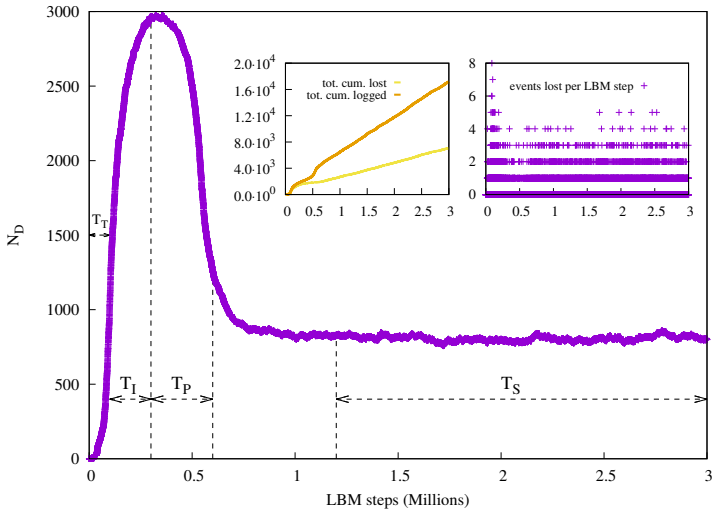



***Figure 5.2.:*** *(Main panel) The time evolution of the total number of droplets, $N_D$, for a reference simulation, $I512V77$ reported in Figure 4.3. As it can be seen the initial phase is characterised by the explosive production of a large number of droplets. Later on coalescences reduce the number of droplet to a lower value. The number of droplets oscillates around a stationary value from $t \sim 0.75M$ LBM time steps onwards. In the insert we show the cumulative number of lost events together with the cumulative number of total events (left) and the number of registered lost event per time step. The tracking is performed every 100 LBM time steps, giving roughly a $50 \div 50\%$ balance in term of computational costs between the diagnostic and the numerical model.*

estimate number of events that, for some reason, have not been detected by the tracking algorithm. We define afs $N_{D_{start}}$ the number of droplets present in the emulsion at the start of the tracking and as $N_{D_{stop}}$ the number of droplet present at the end of the tracking, indicating the registered breakup as $N_b$ and the number of coalescence as $N_c$. From those quantities we estimate $E_l = |N_c + N_b - N_{D_{start}} - N_{D_{stop}}|$. In Table 5.1 we report the estimated error of the tracking algorithm for the case in Figure 5.2, including details of the different time intervals. In particular, we show all time intervals $T_T$, $T_I$, $T_P$ and $T_S$ reported in figure Figure 4.3. We find an $\epsilon_t = 9.7 \cdot 10^{-5}$ for the case in Figure 5.2 when looking at the whole simulation. However, the reported error highlights the limit of our algorithm, which identifies only binary events, more likely to happen in a case of coalescence than in the case of breakups. The highest error is registered for the phase of interface fragmentation where the dynamics is fully breakup driven ($T_T$ and $T_I$) and it is much lower for $T_P$, where mostly coalescence events are registered due to the constant increase of the volume fraction, during the slow injection of the dispersed fluid component.

| $Run_{id}$ | $t_{start}$ | $t_{stop}$ | $N_b$ | $N_c$ | $N_{D_{start}}$ | $N_{D_{stop}}$ | $E_l$ | $\epsilon_t$ |
|---|---|---|---|---|---|---|---|---|
| $I512V77$ | $0M$ | $0.1M$ | 541 | 4 | 1 | 1027 | 489 | $0.9 \cdot 10^{-4}$ |
| $I512V77$ | $0.1M$ | $0.3M$ | 1219 | 239 | 1027 | 2962 | 955 | $0.3 \cdot 10^{-4}$ |
| $I512V77$ | $0.3M$ | $0.6M$ | 291 | 1901 | 2962 | 1260 | 92 | $1.4 \cdot 10^{-5}$ |
| $I512V77$ | $1.2M$ | $3M$ | 4702 | 4976 | 828 | 799 | 245 | $1.4 \cdot 10^{-6}$ |
| $I512V77$ | $0M$ | $3M$ | 8167 | 9015 | 1 | 799 | 50 | $9.7 \cdot 10^{-8}$ |

***Table 5.1.:*** *Table reporting the error estimate for the tracking algorithm for the reference simulations in Figure 4.3. Columns indicate: the time step when the tracking was started and stopped, $t_{start}$ and $t_{stop}$, the number of detected breakup and coalescence events, $N_b$ and $N_c$, respectively, and the number of droplets present in the emulsion at ($t_{start}$), $N_{D_{start}}$, and the final time of the tracking ($t_{stop}$), $N_{D_{stop}}$. From those parameters we compute the number of totally lost event, $E_l$, and the error of the tracking algorithm $\epsilon_t$.*

## 5.2. Analysis of droplet velocities and accelerations

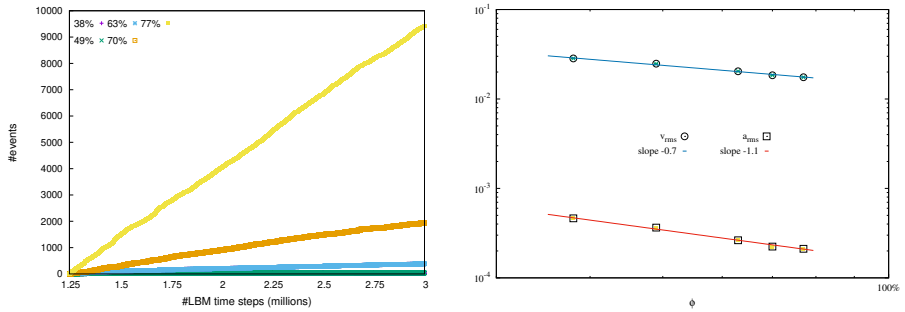
Here we discuss the results obtained by analysing the trajectories of droplets as obtained by the droplet tracking algorithm on the set of simulations reported in Figure 4.3, with particular focus on the stationary phase $T_S$, reported in Figure 5.2. We extend Table 4.1 reporting, in Table 5.2, also the average root mean square of droplets accelerations, $< a_{rms} >$, and the velocities, $< v_{rms} >$, the average droplet size, $< R >$, and also the standard deviation of the droplet size, $< R_\sigma >$. The reported numbers are computed by averaging the values for all droplets existing in the $T_S$ time interval and by statistically averaging throughout all time steps. At the increasing of the volume fraction, accelerations and velocities decrease, showing higher effective viscosity, while at the same time, the $< R_\sigma >$ confirms an increase polydispersity, together with higher droplets' deformation, at the increasing of the volume fraction, as also reported in 4.10. Figure 5.3b, in particular, shows that at increasing of the volume fraction the $< a_{rms} >$ decreases with a slope $-0.7$, while the acceleration presents a decreasing slop of $-1.1$. We estimated the statistical error by dividing the $T_S$ interval in 5 equal sized intervals and computing the variance between the estimate in the difference intervals. This resulted in an error of only about 1% due to high-number of statistical sampling ($\sim 10^9$) during the stationary forced process of emulsification. As already discussed, the tracking algorithm detects events of breakup and coalescence, identifying as "new" droplets, those that are formed by the dynamic of the emulsification process. In Figure 5.3 we show a cumulative statistic of the number of registered events, counting both breakup

| $Run_{id}$ | $< N_D >$ | $N_{D_r}$ | $< v_{rms} >$ | $< a_{rms} >$ | $< R >$ | $< R_{sigma} >$ |
|---|---|---|---|---|---|---|
| $I512V38$ | 3283 | 3332 | 0.028 | 0.00046 | 14.9 | 2.2 |
| $I512V49$ | 3061 | 3127 | 0.025 | 0.00036 | 17.1 | 2.4 |
| $I512V63$ | 2356 | 3055 | 0.020 | 0.00026 | 20.2 | 2.9 |
| $I512V70$ | 1573 | 5056 | 0.018 | 0.00022 | 23.7 | 4.0 |
| $I512V77$ | 800 | 18381 | 0.017 | 0.00021 | 28.9 | 8.7 |

***Table 5.2.:*** *Parameter description for all simulations discussed in this section and initially introduced in Paragraph 4.1.3. Columns indicate, the simulations identifier, $Run_{id}$, the average number of droplets in the emulsion, $< N_D >$, the total number of registered droplets by the tracking algorithm, $N_{D_r}$, the average root mean square of acceleration and velocity, considering all registered droplets, $N_{D_r}$, at all time steps, the average droplet size, $< R >$, and the standard deviation of the droplets sizes, $< R_\sigma >$.*

and coalescence for the various cases. Figure 5.3a reports that at higher volume fraction a much bigger number of events is detected, evidence of the presence of relevant dynamics when compared with cases at lower volume fractions, this, despite the fact that the same constant forcing amplitude was used for all cases. This confirms how the low forcing needed to achieve high volume fractions is not strong enough to provoke significant breakup/coalescence dynamics at lower volume fractions. On the other hand, the reported statistics shows strong breakup/coalescence dynamics at higher volume fraction, the main focus of our tools and analysis. We indicate with, $N_{D_r}$, in Table 5.2, the number of tracked droplets for the various cases, showing that despite the number of $< N_D >$ is lower for high volume fraction the value of $N_{D_r}$ is much higher as result of the droplets dynamics. In Figure 5.4 we report the results for the PDFs of the acceleration and velocity considering all droplets, $N_{D_r}$, for all reported cases in Table 5.2. All PDFs are normalised to have area one while values on the x-axis are normalised by the values of $< v_{rms} >$ and $< a_{rms} >$, reported in Table 5.2, respectively. The velocity distribution shows an interesting three-modal distribution that can be fitting via the superposition of three Gaussian curves. The acceleration distribution of the Lagrangian trajectories is instead strongly non-Gaussian, showing for higher volume fraction, clearly visible large tails, typical of intermittent process, as e.g. the dynamics of particles in turbulence as reported by experiments [85] as well as by direct numerical simulations in fully developed turbulence [86]. The acceleration PDF data can be fit by a the stretched exponential distribution over a wide:

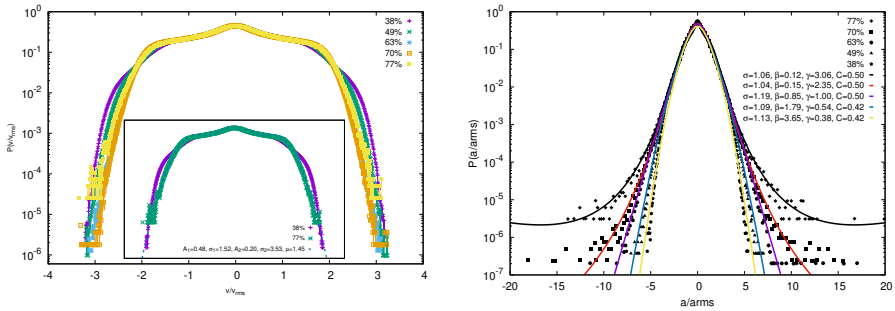
$$P(x) = C \cdot \exp\left(-\frac{x^2}{(1 + |x\beta/\sigma|^\gamma) \cdot \sigma^2}\right) \tag{5.3}$$




**(a)** Registered events vs $\phi$ **(b)** $a_{rms}$ and $v_{rms}$ vs $\phi$

***Figure 5.3.:*** *(Left panel) 5.3 reports in linear scale the cumulative number of breakup and coalescence events registered by the tracking algorithm as a function of time. The events are registered from the beginning to the end of the $T_S$ time interval, as reported in the lower panel of Figure 4.3, and for all simulations in Table 4.1, characterised by constant forcing, A, as well as constant injection rate, $\delta$, but different final volume fraction $\phi$. It is apparent how at low volume fractions only rare events are only registered, indicating that the hydrodynamic stirring amplitude was relatively low, but high enough to display considerable breakup/coalescence dynamics at higher volume fractions (without occurring in phenomena of catastrophic phase inversion).*

provided that the different parameters are adjusted in relation to the volume fraction, as reported in Figure 5.4b. The value of $\gamma$ monotonically decreases going from high to low volume fractions ranging from about $\gamma = 3.19$ to a value close to 0, (typical of Gaussian distributions). Despite the low level of chaotic motion in our system, necessary in order to achieve high volume fraction with the physical parameters modelled by our system (surface tension and disjoining pressure), the acceleration distribution show stretched tails, somehow reminiscent of what measured for low volume fraction droplet dispersion in fully developed turbulence. We attribute this behaviour to the presence of a large number of small droplets, typically present in forced highly packed emulsions. In support of this hypothesis we plot a 2D PDFs of velocities and accelerations, in Figure 5.5 in function of the droplets' radius and comparing the highest and the lowest volume fractions. The PDF of velocity seems to be rather uncorrelated from the droplet radius showing, in both cases, similar shapes but larger for the high volume fraction where higher (poly-)dispersity is present. For both the volume fractions, the velocity peaks match with the shape centre, which roughly coincides with the average radius. The accelerations PDFs instead shows a different shape for the two volume fractions showing, for the case at the highest volume fraction, a more diagonal orientation towards the bottom right, showing that larger values of acceleration correlate with smaller values



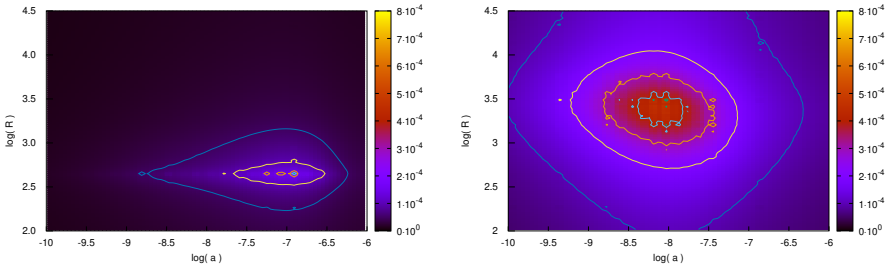

**(a)** $PDF(v/v_{rms})$ **(b)** $PDF(a/a_{rms})$

***Figure 5.4.:*** *The PDFs of the velocity (left panel) and acceleration (right panel), for all droplets and different volume fractions of the emulsification processes, in Table 4.1. (Left panel) the data are fitted via the superposition of three Gaussian of the form* $A_1 \exp\left[-\sigma_1\left(\frac{v}{v_{rms}}\right)^2\right] + A_2 \exp\left[-\sigma_2\left(\frac{v}{v_{rms}} - \mu\right)^2\right] + A_2 \exp\left[-\sigma_1\left(\frac{v}{v_{rms}} + \mu\right)^2\right]$. *(Right panel) the PDF of accelerations is fitted via the stretched exponential distributions* $P(x)$*, see Eq. (5.3). For high volume fraction the PDF shows wide tails,* $\lambda > 1$*, typical of PDF of accelerations of particles in turbulent emulsions. The PDF are reported in linear-log scale and normalised to area one*

for the radius. Therefore, confirming that the large tails of the acceleration PDFs are influenced by the presence of small droplets. It is interesting to show the PDF of acceleration and velocity of the droplets "at rest". Indeed, the emulsion is never completely at rest and keeps slightly moving, particularly at highest volume fraction where diffusion and droplets elasticity favour the presence of plastic events, local topological rearrangement of few droplets. In Figure 5.6 we show the data for the PDFs of the acceleration and velocity for several simulation at rest varying the volume fraction of the droplets in the dispersed phase of the multicomponent fluid.

## 5.3. Analysis of droplet dispersions

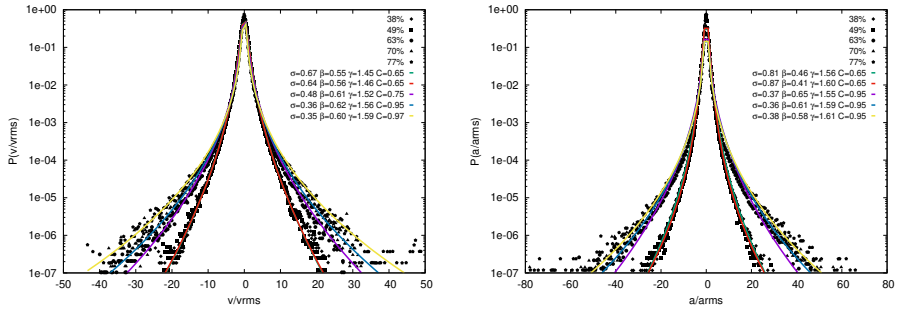
In order to statistically characterise droplets dynamics, we analyse the droplets trajectories, for a large number of droplets, we present measurements relative to space and velocity correlations. In particular, we define the two structure functions of order $n$, $R_n$ for the positions and $S_n$ for the velocity, to provide a quantitative measurement of the droplets dispersion in space and velocity,




**(a)** $I512V38$ **(b)** $I512V77$

**(c)** $I512V38$ **(d)** $I512V77$

***Figure 5.5.:*** *(Top row) A 2-dimensional PDF for all droplets accelerations as a function of the droplets' radius size for both low (left panels), and high (right panels) volume fractions. In the case of low volume fraction the peak of the acceleration is in the centre of the distribution, corresponding to $< R >$. At high-volume fraction the PDF presents a diagonal shape showing a peak of the acceleration towards lower radius sizes. (Bottom row) Shows a 2-dimensional PDF for all droplets velocities as a function of the droplets' radius for both low (left panels), and high (right panels) volume fractions.*



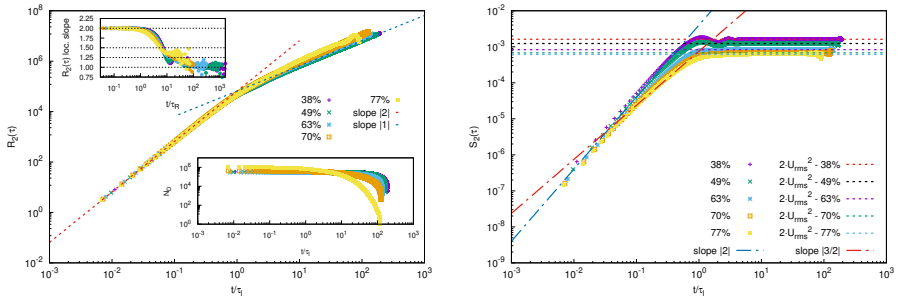

**(a)** $PDF(v/v_{rms})$ **(b)** $PDF(a/a_{rms})$

***Figure 5.6.:*** *The PDF of the velocity and acceleration for all droplets in the considered time frame of the emulsification process. The PDF are reported in linear-log scale and normalised such as to have area one. The PDFs show for all volume fractions, and for both acceleration and velocities, wide tails, corresponding to values of ($\lambda > 1$), showing that the emulsion presents a non-trivial dynamics even, "at rest", increasing at the increasing of the volume fraction.*

respectively:

$$\begin{aligned} {\delta_k}^2 &= (k(t+\tau) - k(t))^2 \\ R_n &= \langle ({\delta_x}^2 + {\delta_y}^2 + {\delta_z}^2)^{(n/2)} \rangle \\ {\delta_{v_k}}^2 &= (v_k(t+\tau) - v_k(t))^2 \\ S_n &= \langle ({\delta_{v_x}}^2 + {\delta_{v_y}}^2 + {\delta_{v_z}}^2)^{(n/2)} \rangle \end{aligned} \tag{5.4}$$

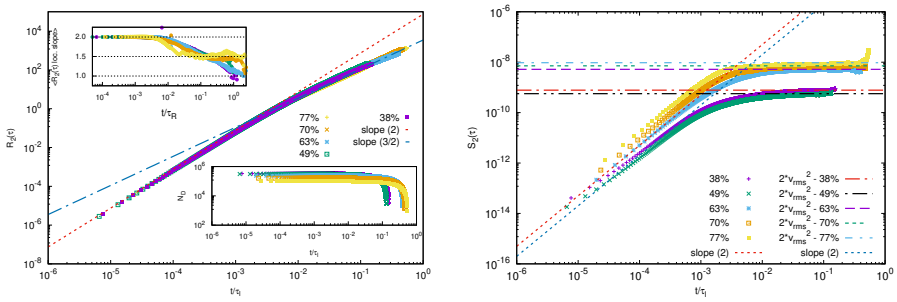
where $\delta_k$ is the distance covered by a droplet along the direction $k = (x, y, z)$ at between the time $t$ and the time $t+\tau$. The quantity $\delta_{v_k}$ is the variation of a droplet velocity along the direction $k$ between the time $t$ and at the time $t+\tau$. Figure 5.7 reports the structure functions $R_n$ and $S_n$, for the order $n = 2$. Here we analyse the trajectories obtained from tracking droplets on some of the simulations introduced in Chapter 4, varying the volume fraction, the forcing amplitude and with the hydrodynamic forcing turned off. Figure 5.7 reports the behaviour of $R_2$ (left panel) and $S_2$ (right panel) for emulsions in forced steady state at different volume fractions, see Table 4.1. Values are reported in logarithmic scale with the x-axis normalised by $\tau_l = L/2U_{rms}$, where $L$ is the size of the simulation domain in lattice points and $U_{rms}$, is defined as the root mean square of the hydrodynamic fluid velocity. In Figure 5.7a the slopes show a cross over between an initial ballistic motion (corresponding to slope 2) and a diffusive behaviour (corresponding to slope 1) with $t/\tau_l = 1$. Notice that the latter slope tends to assume slightly higher values for higher




**(a)** $R_2(\tau)$ **(b)** $S_2(\tau)$

***Figure 5.7.:*** *In the Figure the measured $R_2$ (5.7a) and $S_2$ (5.7b) for emulsions in forced steady state at different volume fractions. Values are reported in logarithmic scale with the x-axis normalised by $\tau_l = L/2U_{rms}$. In Figure 5.7a the slopes shows a cross over between ballistic motion (slope (2)), and diffusion (slope (1)) with $t/\tau_l = 1$. In Figure 5.7b the initial slope (2) displays an initial exponential grow, turning in a constant value when $S_2(\tau) = 2{U_{rms}}^2$ for all cases. The top-left insert in 5.7a plot the local slope defined as $d\ln(R_2(\tau))/d\ln(\tau)$ The bottom-right insert in 5.7a shows the number of droplets counted in the statistic for the various cases.*

volume fractions, displaying how the high-packing effects somehow influences the diffusive behaviour, something not present in the lower volume fractions where a clean diffusive behaviour is visible. The statistics considers all droplets, $N_{D_r}$, taken at different intervals of $T_S$. In particular, we break the motion of every droplets in 100 time intervals, performing an ensemble averaging of the absolute dispersion for all droplets in all intervals. The insert, bottom-right corner, in Figure 5.7a shows, in log-log scale, the number of droplets counted for every considered time interval, such that a number of samples of the order $10^6$ are considered in the analysis. As expected, the case at the highest volume fraction shows that the initial highest statistics that, more rapidly, drop to lower values due to the strong dynamics and the fact that droplets are likely to occur in the event of a breakup or an event of coalescence with other droplets. The insert, top-left corner, in Figure 5.7a shows the local slope of structure function $R_2$, defined as $d\ln(R_2(\tau))/d\ln(\tau)$, and reported in log-log scale with the x-axis normalised by $\tau_R = R/2U_{rms}$, where $R$ is the droplets' average size. The local slope seems to confirm how higher volume fractions present different characteristics, in particular lower volume fraction reaching in the local slope at 1, while higher volume fraction stop earlier, approximately at the value of 1.25. In Figure 5.7b the initial slope 2 displays an initial power-law growth, turning in a constant value when $S_2(\tau) \simeq 2{U_{rms}}^2$, for all cases. Of particular




**(a)** $R_2(\tau)$ **(b)** $S_2(\tau)$

***Figure 5.8.:*** *(Left panel) $R_2$ and (right panel) $S_2$ for emulsions in non-forced steady state at different volume fractions. Plots are in log-log scale with the x-axis normalised by $\tau_l = L/2U_{rms}$. In (left panel) the slopes show a cross over between ballistic motion (slope 2), and a slope close to 3/2 when the statistics flats down. In the right panel the initial slope 2 displays turns in a constant value when $S_2(\tau) \simeq 2{U_{rms}}^2$ for all cases. The top-left insert in the left panel shows the local slope defined as $d\ln(R_2(\tau))/d\ln(\tau)$ The bottom-right insert in the left panel shows the number of droplets counted in the statistic for the various cases as a function of $t$.*

interest is to note the gap between high and lower volume fractions in the average velocity, showing how, at a constant magnitude of the hydrodynamic stirring $A = 0.485 \cdot 10^{-6}$ (see Table 4.1), emulsions at lower volume fraction present higher hydrodynamic velocities, $U$.

In Figure 5.8 the same structure functions of order 2, $R_2$ and $S_2$, are reported for the non-forced cases. These data are obtained restarting the simulations reported in Table 4.1 for additional 4M LBM time steps, where only the latest 2M LBM time steps are considered. This is to make sure that inertial effect from previously forced flow is entirely dissipated. In this case we find a fully reversed situation if compared with the forced case. In fact, at rest, the higher volume fraction presents longer dispersion, Figure 5.8a, and higher velocities, Figure 5.8b, with respect the simulations at lower volume fraction. In Figure 5.8a the slopes show a cross over between the initial ballistic motion (slope 2) and the not yet fully developed diffusive behaviour, showing a slope 1.5 from approximately $t/\tau_l = 2 \cdot 10^{-3}$ onwards. In Figure 5.7b the initial slope 2 displays an initial power-law growth, turning in a constant value when $S_2(\tau) \simeq 2{U_{rms}}^2$ for all cases, as for the forced cases but, as expected presenting an hydrodynamic velocity $U$ order of magnitudes lower. The insert shows the number of considered samples which, in this case, presents a similar trend for all cases, due to the fact that for all cases the number of droplets, $N_d$, remains

constant due to the low dynamics as the emulsions are considered at rest. The insert, upper-left corner, reports the local slope in log-log scale with the x-axis normalised by $\tau_r = R/2U_{rms}$, where $R$ is the droplets' average size. The local slope of the structure function, $d\ln(R_2(\tau))/d\ln(\tau)$, shows lower volume fraction would reach the diffusive behaviour leading to 1 while higher volume fraction seems to stop at a value of 1.5.

## 5.4. Analysis of droplet dynamics in dense emulsions

We provide here a quantitative description on the droplets dynamics in a stirred dense emulsion in a stationary forced state, $T_S$, for 0.7M LBM time steps, for the simulation starting from interface fragmentation, and at $1024^3$ resolution. In particular, we restarted $I1024V77$ for 1M LBM time step, considering only the latest 0.7M LBM time steps in order to make sure the chaotic stirring is fully developed. During the temporal evolution we registered a total number of 31153 events, divided in, $N_b = 14101$ breakups, and $N_c = 17052$ coalescence events, with an estimated $\epsilon_t = 1.4 \cdot 10^{-5}$. Despite the intense dynamic, the number of droplets in the stationary forced state remains mostly constant: the number of droplets in the emulsion at the beginning of the tracking was $N_{D_{start}} = 5176$, and number of droplets at the end of the tracking was $N_{D_{stop}} = 5178$. In Figure 5.9 (upper panel) we show the droplet size distribution for the droplets present in the emulsion, $10^8$ number of samples, together with the DSD of the droplets involved in an event of breakup or an event of coalescence, $10^5$ samples. In particular we count the sizes of the two daughters droplets generated in a breakup event, for a larger mother droplet breaking in two (or more) pieces, and the sizes of the two daughters droplets that coalescing forms a larger mother droplet. The DSDs are displayed as the probability distribution function (normalised to area 1) of the droplet size (expressed as the diameter of a sphere having the same volume of the droplet). Notice that the two distributions present a similar bimodal shape, with the DSD of events shifted towards larger radii, showing that it is unlikely for smaller droplets occur in an event of breakup or coalescence. In Figure 5.9 (lower panel) we normalise the DSDs of events by the DSD of the emulsion. It confirms an increase of the probability of occurring an event for larger droplets following a power-law of 5.25, for droplets size near the average size $< R >$.

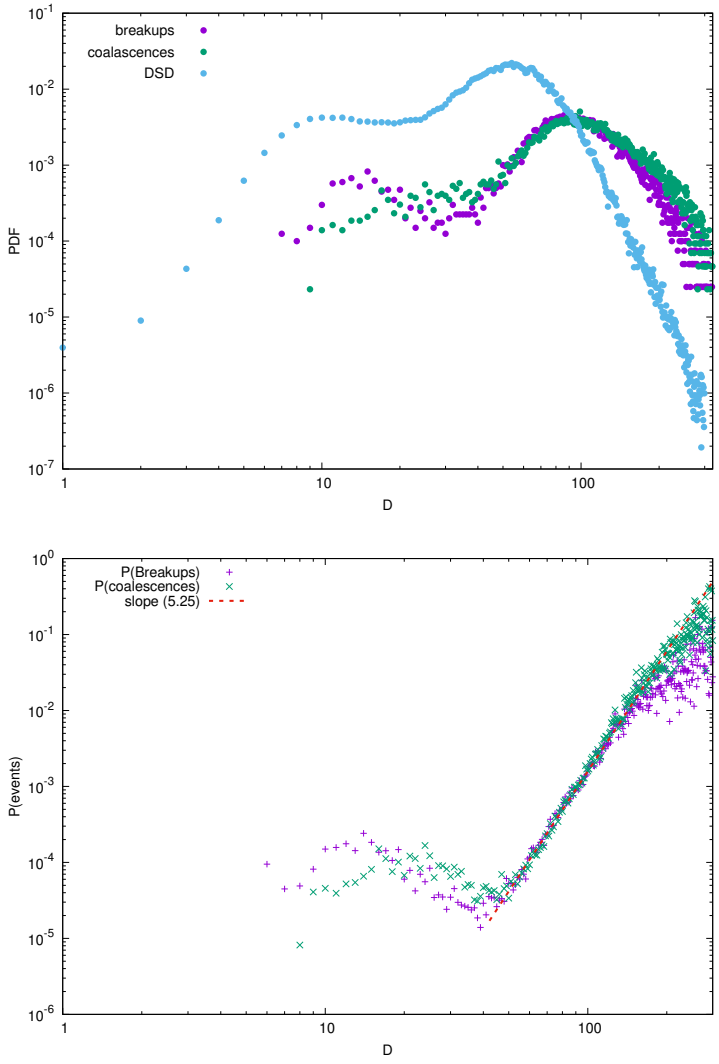



***Figure 5.9.:*** *(Top panel) The droplet size distribution (DSD) for the droplets present in the emulsion considered in this Paragraph, together with the DSDs of the droplets involved in an event of breakup or coalescence, respectively. (Bottom panel) Reports on the y-axis the probability that an event of breakup or coalescence occurs for a droplet of a given diameter. It displays a rapid increase of the probability for larger droplets, showing a power-law with a slope close to* 5.25.

# 6. High-performance LBM

In this chapter, we present the enabling of the LBE3D, code to study the physics of dense emulsions, on current and latest of world-class facilities for HPC. [1] We assess the impact on computing performances of several optimization steps to maximize, the overhead of internode communication, the number of flops and the memory throughput, including an analysis of costs and power consumption. We present results on both Intel processors for high-end computing, as well as on hybrid platforms equipped with NVIDIA accelerators, of the latest generations. In these analysis we focus on the archiectures equipping the three supercomputers target of this project. Those are the Marconi-KNL partition, hosted at CINECA, the MareNostrum4, hosted at the Barcelona Supercomputing Center (BSC) and the hybrid CPU/GPU Marconi100 at CINECA, a reppresentative target computer platform for further projects.

## 6.1. Optimization of LBM based applications

The LBM is based on the synthetic dynamics of populations arranged at the edges of a discrete lattice. It is discrete in time, space and momenta, offering a large amount of easily identifiable parallelism while making it an ideal tool for investigating performances of modern systems for high-performance computing [90, 91, 92]. One of the most appealing feature of the LBM, for what concerns high-performance computing, is the fact that it is an explicit and local scheme. This allow relatively straightforward and efficient parallelization on massively parallel machines, due to the data-locality of the algorithm. At each time step, populations are first moved from lattice-site to lattice-site applying the *propagate* operator, and then are modified through a collisional operator changing their values according to the local equilibrium condition. The computing pattern for the collision (*collide*) and the consequent streaming (*propagate*), introduced in Section 3.2 (Figure 3.2), are renown main bottlenecks for the LB class of applications.

[1] This chapter closely follow the contents of [87, 88, 89] and of Aliei et al., "An OpenACC based multicomponent Lattice-Boltzmann solver on GPUs ", currently in preparation.

### 6.1.1. Data layouts

Many stencil based applications, including LBM, are commonly implemented using either Array of Structures (AoS) or Structure of Arrays (SoA) data layouts. In the AoS approach, originally used in the LBE3D code too, population elements of each lattice site are stored contiguously in memory. Therefore, the AoS structure is constructed as a 3-dimensional lattice where each element is composed by $N$ population values (NPOP). On the other hand, LB based applications adopting the SoA schema, allocate the 3-dimensional lattice as a collection of NPOP arrays each storing for every site a single element population value. However, none of those two data layouts have been demonstrated to be ideal for LB based applications since, while the AoS delivers better performances for the *collide* kernel, it lacks in memory bandwidth if compared with SoA when considering the *propagate* kernel. Alternate data layouts aim to improve the overall performances of LB based applications, and in Figure 6.1 we show a graphical representation for a sample lattice of 4 x 8 using two populations per site (memory addresses increase left-to-right top-to-bottom). The two new layouts called CSoA and CAoSoA are seen as an extension of the SoA where $VL$ lattice-site data at distance $L/VL$ ($L$ dimension of major order store) are clustered in consecutive elements for each population array, with $VL$ equal to the number of double precision elements that can be stored in the vector register available on the given architecture. The newly designed layouts for high-performance LBM based codes allow to store data properly aligned in memory and aiding the compiler in the process of auto-vectorization of the steps required to compute the LBM main loop. This efficient memory access allows to vectorize the steps of *propagate* kernel. In particular, the CAoSoA structure is a mix between CSoA and AoS, and allows to exploit the benefit of the $VL$ clusterization of lattice sites element as introduced by the CSoA schema but with the benefit of higher locality in regards to the populations. For this reason, this layout may deliver better overall performances for the *collide* kernel. In Figure 6.1, each dark grey-box is a cluster with $VL$=2 for both the CSoA and the CAoSoA data layouts.

### 6.1.2. Vectorization

In the AoS layout, the population data of each lattice site is stored in memory in consecutive locations, and then each lattice site is stored one after the other. Conversely, in the SoA scheme, populations with the same index of all sites are stored at consecutive addresses as an array, and then arrays of different population are stored one after the other. The first scheme in general optimizes cache

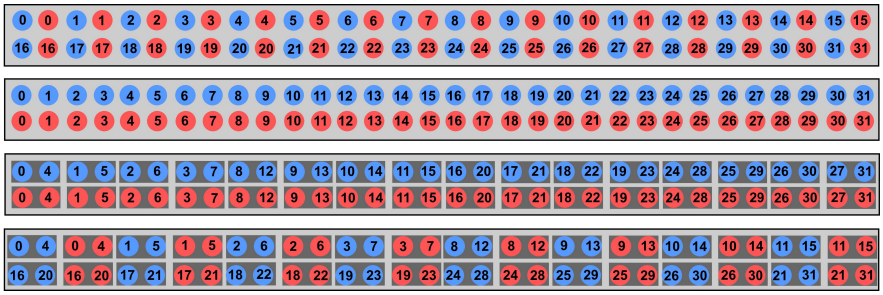
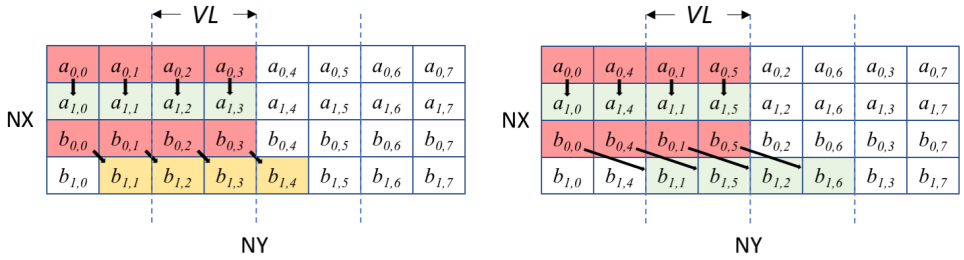


***Figure 6.1.:*** *Top to bottom, AoS, SoA, CSoA and CAoSoA data memory layouts for a 4 x 8 lattice with two populations (red and blue) per site. This picture was originaly proposed in [17] and reported here for courtesy of the authors.*

accesses and data re-use, while the latter fits better data-parallelism processing appropriate for latest processors based on large vector SIMD data-paths. The CSoA is a direct extension of SoA, where each population array is composed of clusters of $VL$ data elements at distance $L/VL$, being $L$ the size of the lattice, and $VL$ the size of SIMD registers of CPUs. Cluster are then stored in memory as vectors at properly aligned addresses, and then processed using SIMD operations to exploit vectorization. A graphical representation of the improvement given by the CSoA structure, in regards to the SoA, is shown in Figure 6.2. In particular, the Figure 6.2a represents a two-dimensional lattice $(2 \times 8)$ of two populations stored in row-major order, using the SoA layout and Figure 6.2b the CSoA layout, respectively. For simplicity the pictures show the data shift in place, however in the case of LBE3D the *propagate* kernel works on two lattices of equal dimensions. In the SoA classical scheme, site elements of different populations are contiguously placed in memory and all data elements of a single population move in the same direction (as described by a given LB scheme). In Figure 6.2a we represent the data shift of the two populations ($a$ and $b$), with the population ($a$) moving along the x-coordinate and ($b$) moving along both coordinates. Supposing each lattice element is stored in double precision and the memory alignment as well as vector registers length being 128bits, when using SoA every data stream along the most nested coordinate (NY, in the described case) is misaligned in memory. In fact, when the data move along the x-direction both the operation of reading and writing are aligned in memory and can be vectorized while, if considering the case of the population ($b$), only the reading is aligned in memory but the operation of writing would result misaligned. On the other hand, in the case of using the CSoA data layout, the data shift in any direction consists in moving clusters of VL data being aligned in memory where a vectorized

data shift can be always applied. Therefore, the CSoA effective improvement with respect to the SoA layout depends by the utilized LB scheme as each scheme describes a different number of shifts along the most nested direction. In the case of the LBE3D, implementing the D3Q19 scheme, only 9 of the 19 directions stream along the z-direction and therefore the improvement is only related to 50% of the data shift happening on a single LB loop (population 0 does not move). However, applications are usually composed of many kernels,




**(a)** Memory data shift for SoA data layout **(b)** Memory data shift for CSoA data layout

***Figure 6.2.:*** *The Figure shows the data shift happening for the propagate kernel on a* $2 \times 8$ *two-dimensional lattice with two populations (a and b), using the SoA (a) and the CSoA (b) schemas, respectively. Data are considered stored in row-major order and the indexes indicating the site position in the two-dimensional lattice. In the picture the arrows describe the direction on which each given population moves. Data elements representing a specific data movement have been coloured: red for elements with correct memory alignment for the operation of reading, green for elements with a correct memory alignment for the operation of writing and finally, yellow for elements with a memory misalignment for the operation of writing. While in (a) only 50% of the operations of writing are aligned in memory, in (b) 100% of the copies of element required by the propagate kernel will result aligned in memory.*

and each may have different computing requirements. This is for example the case of LBM, where the *propagate* kernel performs only memory operations, while the *collide* kernel is more compute intensive. In the other direction, the CAoSoA layout mixes the vectorization features of the CSoA with the locality of AoS. In practice, it stores at contiguous memory locations a cluster of $VL$ populations of same index of different sites, and then all clusters of population for that $VL$ sites are stored in memory one after the other. The rationale behind this scheme is that each block can be moved and processed using the SIMD instructions of the processor, keeping cluster corresponding to different populations indexes relatively close in memory to enhance locality of memory accesses. This choice potentially retains the benefits of the AoS and CSoA layouts, and therefore provides a compromise between the requirements of both *propagate* and *collide* kernels. Achieved vectorization by the fused and

***Table 6.1.:*** *VPU Usage measured by the Intel® VTune for fused LBE3D*

| *data layouts* | *Vector VPU Intensity* |
|---|---|
| AoS | 20% |
| SoA | 20% |
| CSoA | 100% |
| CAoSoA | 100% |

fully optimized version (see next section) of the LBE3D is shown in Table 6.1 where we report the Vector Processing Unit (VPU) activity as the measured ratio of the two Intel VTune's counters UOPS_RETIRED.SCALAR_SIMD and UOPS_RETIRED.PACKED_SIMD. The final value is given by the ratio between the number of vector operations the core performed (PACKAED_SIMD) to the sum of all operations (SCALAR_SIMD and PACKED_SIMD) as properly described in [93].

### 6.1.3. Fused LBM

A further otimization step introduced in the LBE3D is the tentative to exploit data locality, by taking advantage that all hydrodynamic quantities can be computed at any given time from the discrete particle distribution function $f_i(\boldsymbol{x}, t)$, stored in the lattice schema (see 3.2). The Equation 3.8 generically describes the Boltzmann equation in a form where it is simple to identify the two operators of propagation and collision: left- and right-hand side of the equal, respectively. A naive implementation of the equation would require to apply first the *propagate* kernel, then all discretized hydrodynamic quantities needed for computing the equilibrium function $f^{eq}$, and finally the collision operator. To improve this approach, we have implemented two separate versions of the code where the *collide* and *propagate* kernels have been fused in a single step, as illustrated in the Algorithm description 3. We refer with CF the classical scheme implemented in LBM applications, where the *propagate* and *collide* kernels are kept separated, and the density is stored on a separate structure, and with FF the fully fused version where the two kernels are joined together and the density is computed as temporary when needed. As in the FF version all local quantities are stored locally, in vectors of $VL$ dimension, the number of vectorized instructions within the FF kernel goes to from 20% to 100% as we measured runtime using the Intel VTune and described in Table 6.1.

**Algorithm 3:** LBM fused loop optimization schema

The LBM main loop implemented by simply coding the LB equation as shown in Eq. 3.8

```
for all time steps do
    Set boundary conditions
    for all lattice sites do
        Propagate
    end for
    for all lattice sites do
        Hydrovar
    end for
    for all lattice sites do
        Equili
    end for
    for all lattice sites do
        Collis
    end for
end for
```

The collisional operator (*collide*) is computed with only one iteration over the lattice sites, named CF version

```
for all time steps do
    Set boundary conditions
    for all lattice sites do
        Propagate
    end for
    for all lattice sites do
        COLLIDE_FUSED
    end for
end for
```

The *propagate* and the *collide* kernels are fused together. All intermediates quantities are only temporarily computed, named FF version

```
for all time steps do
    Set boundary conditions
    for all lattice sites do
        FULLY_FUSED
    end for
end for
```

Loop compression and better data locality!!

### 6.1.4. LBE3D and OpenACC

The fast growing popularity of accelerated computing is coming along with the rising of programming paradigms that are meant to simplify the effort of porting complex codes on accelerated distributed systems. Indeed, for many years a major obstacle has been that only highly optimized approaches, based on specific languages, were demonstrating to achieve decent computing performances. Scientific developers were recluctant to include those languages in complex codes, requiring relevant effort first, in the phase of coding, and then for the subsequent maintance, at the price of no-portability. OpenACC [94] is a GPU-based programming model that is emerging as the standard in directive-based accelerator programming. It offers the oppotunity, such as others directive-

based paradigms (i.e. OpenMP [95]), of incremental development, avoiding intrusive additional coding, while being portable across different architectures and compilers, including hybdrid accelerated platforms, with a tradeoff between performance and portability. Although OpenACC is desirable for its simplicity and unified interface but compared to, e.g. CUDA, it has a rather significant performance drop-off, for a study of performance portability of OpenACC compared to CUDA for LBM applications, see [18]. In C, the LBE3D programming language, these directives take the form of a $\#pragma \quad acc \quad kernels$. Similarly to other pragma based paradigms the main development task is to define code sections, that in this case are offloaded to the accelerator, while managing the data movement between CPU and GPU. On of the main advantages is also the fact that recent developments avoid to force usage of pointers to identify the different memory levels. Today, this extends efficiently also to the interGPUs communication, thanks to the latest PCI-X and NVLINK technology generations, allowing fast data transfer between distributed GPUs. However, high-end scientific simulations also requires high-performance I/O. In our OpenACC based implementation of the LBE3D we perform those operations on the CPUs. A schematic algorithm of how the main LBE3D loop as implemented with OpenACC, in its non-fused version (see Algorithm 3), is shown in Algorithm description 4.

### 6.1.5. Optimization of LBM on ditributed memory

For simulations at high-resolutions, we have been scaling the LBM up to a 1024 compute nodes of the Marconi-KNL partition, hosted at CINECA, for an equivalent of +70K distributed processing units interconnected via the Intel Omnipath network. At this scaling the data exchange required for updating the boundary conditions becomes the determining factors for high-performance and efficiency. The argument becomes even more relevant for the multi-GPUs version. We have first implemented an hybrid MPI+OpenMP version for the LBE3D to reduce the number of MPI processes involved in the data exchanges, favouring the runtime configuration of running 1 MPI process per CPU socket, while using multi-threading to exploit the remaining cores available. The hybrid MPI+Open approach offers more flexibility at runtime for finding the best configuration accordingly to the characteristics of the computer architecture. For instance, in the case of the multi-GPU version, the number of MPI processes per node is instead related to the number of GPUs per node as every MPI processes is meant to address a single accelerator. We implemented an optimized version of the exhange of the boundary conditions, which reduces the amount of data while taking advantage of the hybrid approach MPI+OpenMP

**Algorithm 4:** Schematics of the LBE3D Algorithm

```
Initialization;
Move lattice data to device;
for i ← 1 to NUM_STEPS do
    Set Boundary Condition;
    Stream populations for all lattice sites;
    Compute Hydrodynamical Quantities;
    Compute Equilibrium State;
    Collide populations for all lattice sites;
    if i%NUM_DIAG = 0 then
        Update lattice data on the host;
        Compute Hydrodynamical Quantities;
        Do Statistical Analysis ;
        Do Diagnostics ;
        Dump Data ;
    end
    Swap lattice pointers ;
end
```

approach. As represented in Figure 3.2a, in the streaming kernel every lattice point is updated with the contribution from neighbor points. At the boundaries of the local data domain of every MPI process, this translates in the need to update the neighbor elements as stored on a different physical memory. The contribution from neighbor processes is then stored in *halos*, data used during the *streaming* kernel. We implement a process 3-dimensional grid distribution on a data domain in 3-dimensions and therefore at the time of exchanging the boundaries elements will not be contiguous in memory as an operation of data exchange using MPI would require (commonly MPI datatypes are used to define view of non-contiguous data). We implemented operations of data packing and unpacking of non-contiguous data into buffers of contiguous memory, for transfering only contiguous memory areas, while performing of packing and packing locally to every process exploiting multi-threading to speed up. This offers a significant boost in the communication, especially when considering the distributed multi-GPUs version. Another common approach is to transfer between processes buffers of size $N \cdot NPOP$, where $N$ is the size of the boundary side of the discretized grid along a given dimension and $NPOP$ is the number of population per every lattice point. We have optimized this approach by packing for sending and receiving, only the populations needed on given directions, as

schematically reppresented in Figure 6.3. This significantly reduces the number of elements being communicated and therefore speeding up the overall communication. On Marconi100, hosted at CINECA, we use CUDA-aware implementation of spectrum-mpi for GPU-to-GPU communication. Spectrum-mpi, which is an IBM quality version of MPI implementation based on OpenMPI. For that we use the $\#pargma \quad acc \quad host\_data \quad use\_device(field)$ which tells the compiler to map the GPU memory to host, so that it can be used in MPI calls.

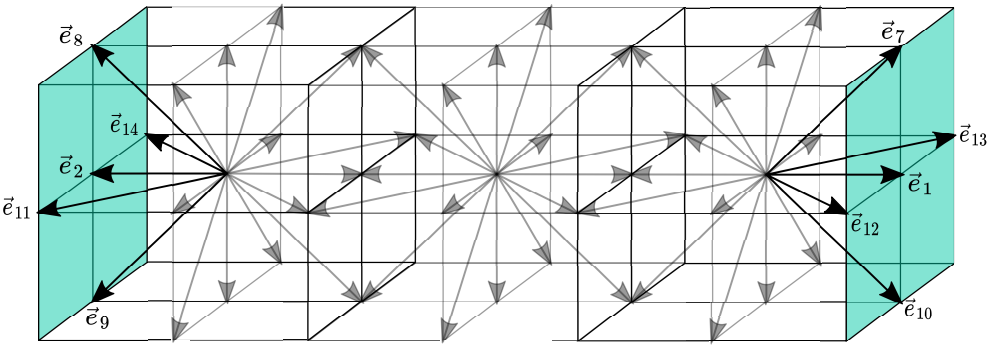



***Figure 6.3.:*** *Schematics exchange of optimized periodic boundary conditions in a LB computational kernel. Populations that needs to be exchanged at the one of the $x$ surfaces are highlighted. We use a packing scheme to reduce the amount of data exchanged in operations of boundaries exhange between processes.*

## 6.2. Results

In this Section we present the performance analysis of the LBE3D application on different architectures. The first analysis, aimed to select the most efficient data-layouts on a 3-dimensional system, is based on a semplified suite of representative benchmarks. We show the results of the described optimizations on the *propagate* and *collide*, including the fused versions, on different platforms for high-end computing. Then, we discuss the performance in term of time-to-solution for the multicomponent code, comparing the latest results obtained on Marconi100 using the OpenACC version, with the results obtained during the PRACE project production on the MareNostrum.

### 6.2.1. The Poiseuille benchmarks on the SKL and the KNL processors

To measure the performance improvement of the proposed optimizations, we first implemented a suite of benchmarks including several versions of the LBE3D, by varying the data-layouts. All presented performance measurements are referred to a simple channel flow set-up as shown in Figure 6.4. A fluid between two parallel solid walls is put into motion by a homogeneous body force, and no-slip boundary conditions are applied at these walls, while periodic boundary conditions are applied along the two other directions. We first

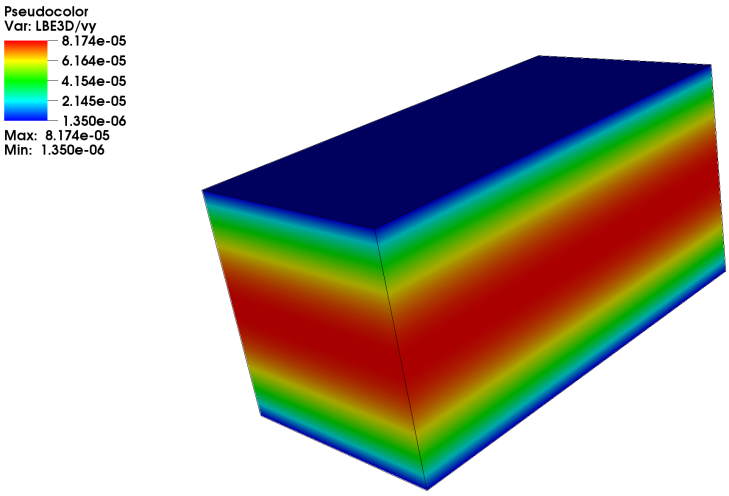



***Figure 6.4.:*** *Velocity field along the forcing direction. Snapshot taken once the flow field reached the final steady state.*

present performance for the *propagate* and *collide* kernels, on the KNL processor. As mentioned earlier, LBM are characterised by a phase of propagate, where populations move from lattice-site to lattice-site, describing the flows momentum. Kernels implementing this phase are, in particular, memory-bound because the movement is practically implemented as the copy of a single data (double precision floating point number) from one location to another location in memory. Therefore, a key aspect to achieve maximum speed is to describe this operation with a memory access pattern that can exploit the maximum memory bandwidth. Clustered data layouts allow to vectorize such operation, whether data are aligned in memory, as the compiler replaces the scalar operation of copy with a copy operation on registers capable of multiple elements of the same kind (vector registers). In Figure 6.5a, we report the measured

performances of memory bandwidth obtained by the *propagate* kernel on a KNL node configured in Flat mode. The CSoA version allows to achieve almost 350 GB/s memory with a significant improvement in performance if compared with the canonical AoS or SoA approaches. We can confirm that also for the cases of 3-dimensional lattices the CSoA version allows for the *propagate* kernel to achieve the highest value of memory bandwidth if compared with other analysed data layouts. It is relevant to underline how in most cases the memory bandwidth saturates at 64 threads (a single hardware thread per core). The

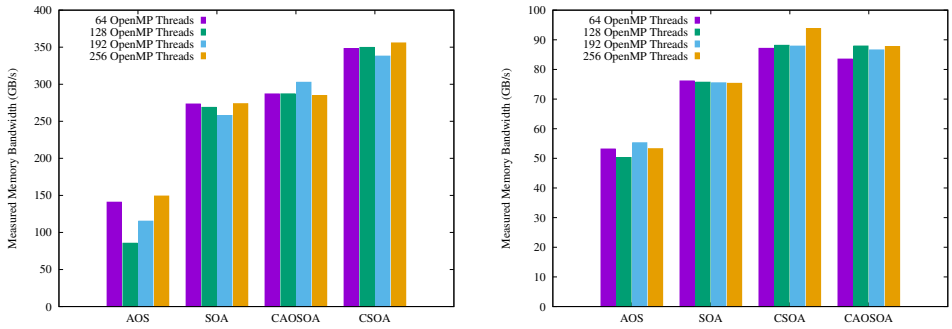



***Figure 6.5.:*** *Measured memory bandwidth for the propagate kernel using different data layouts: AoS, SoA, CSoA and CAoSA. The KNL model Intel(R) Xeon Phi(TM) CPU 7230 @ 1.30GHz is configured in flat mode 6.6a and in cache mode 6.6b. The LBE3D is benchmarked on a $256^3$ lattice.*

measured memory bandwidth drops if considering larger lattices with a data domain unable to fit in the 16 GB/s of the MCDRAM. In Figure 6.5b, data are obtained with the KNL configured in Cache mode and the real peak performance achieved by the CSoA data layout is of about 80 GB/s, with a factor of 4x reduction if compared with the memory bandwidth obtained when data fit into the MCDRAM. Moreover, the intensive use of the DRAM memory squeezes the performance gap among the various data layouts such that CSoA becomes only 10% better of the SoA and comparable with the CAoSoA versions. The other important phase of LBM is the collision phase. It is implemented as the *collide* kernel which is generally considered compute bounded because all discretised quantities such as forces, velocities and densities are per site computed with high data locality. For most of those quantities, the lattice elements are read from the input lattice and written to the output lattice contiguously, but for the computation of the density, and velocity, which requires accessing all population elements for each site. While this is a cache friendly operation for the AoS scheme, due to data locality (all populations elements are stored

contiguously), it is not for data layouts where per-site population are scattered in memory, such as SoA and CSoA. However, as per the CSoA version the speedup achieved by vectorized operations on clustered data helps to overrun this problem. The mixed schema of the CAoSoA is expected to take advantage of accessing contiguous population elements while working on clustered data. In Figure 6.6a the measured value of peak performance is reported. In flat mode for a medium size grid of 256x256x128 we measure a peak performance of about 700 GFLOP/s corresponding to around the 25% of the nominal peak: approximately half of the value reported on the Nov 2018 list of the TOP500 for KNL based architectures running the High Performance Computing Linpack Benchmark. CSoA and CAoSoA are confirmed to outperform canonical AoS and SoA data layouts by a factor between x2 to x3. However, when increasing the lattice size while switching to the Cache mode configuration the performance gap between the various data layouts drops significantly, also in the case of the *collide* kernel. As we report in Figure 6.6b, when intensively using DRAM the real performance peak is reduced by a average factor x2 if compared with the same kernel running on KNL in Flat mode, and the gaps between the different data layouts drops between x1.5 to x2.5.

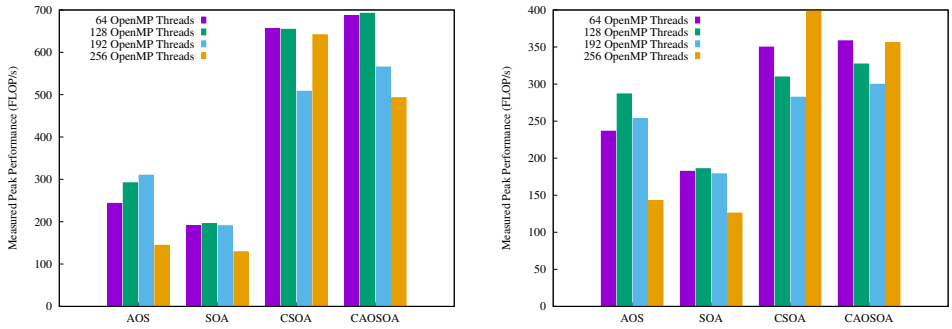
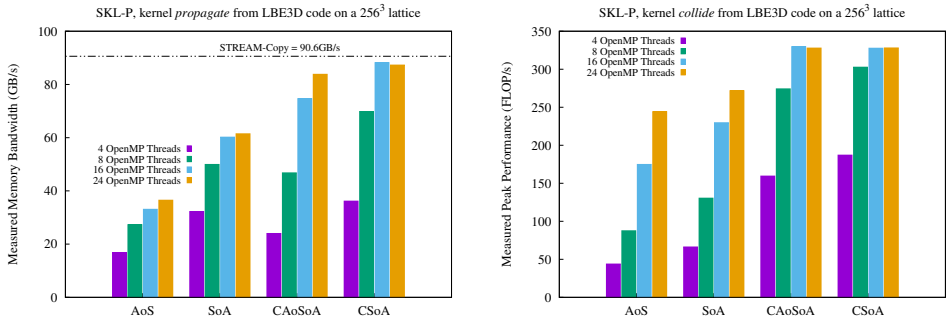



***Figure 6.6.:*** *Measured real peak of performances for the collide kernel using different data layouts: AoS, SoA, CSoA and CAoSA. The KNL model Intel(R) Xeon Phi(TM) CPU 7230 @ 1.30GHz is configured in flat mode 6.6a and in cache mode 6.6b. The LBE3D is benchmarked on a* $256^3$ *lattice.*

The performance analysis reported in Figure 6.7a shows the measured memory bandwidth of the *propagate* kernel, while scaling the number of threads within a single SKL socket. Also the tests on the SKL-P confirms as for the KNL processor, how clustered data layouts CSoA and CASoA deliver good results in term of memory bandwidth on the SKL processor too. Peak of the measured memory bandwidth reaches almost 90GB/s which is about 70% of




**(a)** Measured Memory Bandwidth **(b)** Measured Peak Performance

***Figure 6.7.:*** *6.7a measured memory bandwidth for the propagate kernel, 6.7b the measured peak performance for the collide kernel, for the SKL-P processor. The performance measurement is reported using a single MPI process on the socket and a number of threads equal to the maximum number of cores available on the given CPU model. Results are from a simulation with $256^3$ lattice sites representing a real workload.*

the nominal peak performance (128GB/s) and close to the results achieved using the STREAM-Copy benchmark. However, the thread scaling shows a saturation of the memory bandwidth already at 16 threads per socket despite the 24-cores available on socket. On the SKL-P processor the CSoA layout, implemented in the LBE3D, delivers more than 2 times higher performance if compared with the AoS implementation and it is 50% better if compared with the canonical SoA layout. The same performance saturation, Figure 6.7b, is reported for the *collide* kernel where we report the number of FLOP/s, despite as mentioned in Section 6.1 the *collide* kernel is expected to be more computing intensive. It is interesting to notice how good thread scaling is given within the socket by the AoS structure, which is overall less efficient if compared with other data layouts, confirming how scaling should not be considered a metric for analyzing high-performance computing applications unless coupled together with efficiency.

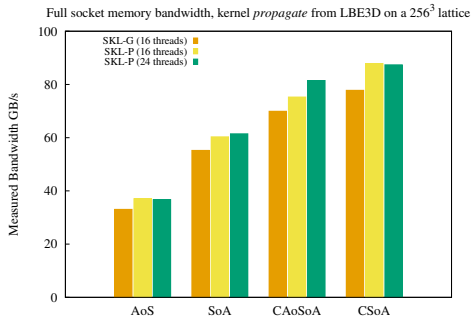
### 6.2.2. The Poiseuille benchmarks on CPU sockets of the Intel processor family

In Figure 6.8a we compare performances of the LBE3D on both the SKL-P and the SKL-G processors. The latest being expected to be a less performing processor as equipped with approximately 30% lower number of cores per socket if compared with the SKL-P. Our results show that if considering the most optimized version of the LBE3D, CSoA, there is only a really small

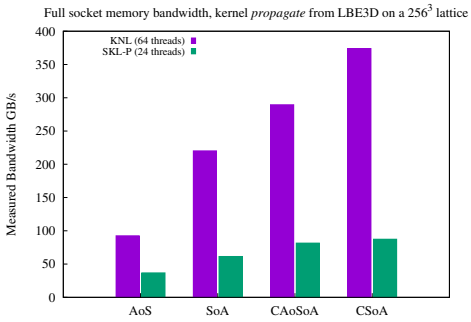
performance gap between the two processors despite the SKL-P costing about 2.5 times more than the SKL-G. Indeed, the recommended market price for those two processors [96] is approximately €4700 for the SKL-P version and approximately €1900 for SKL-G. The relevance of the memory bandwidth limit of the LBE3D application on the SKL processor is also underlined in the Figure 6.8b where we show that the performance gap in term of memory bandwidth between the two processors defines a similar performance trend when looking at the entire LBM main loop, in the most optimized version of the LBE3D (FF). The Figure 6.8 generally shows the impact of the memory bandwidth capability available on socket on the LBE3D. Indeed, by looking at the comparison between the SKL-P and the KNL processors a factor of about 4 times in the performance per socket is measured, which is similar to the difference in term of memory bandwidth between the two systems, as for the KNL we consider the MCDRAM only (flat mode). The Figure 6.8d reports the performance gap between the SKL and the KNL processor for the same simulation while considering the highly-optimized version (FF) of the LBE3D. For the KNL processor, installed in our lab, the recommended market price is approximately €1900 [96]. We have also analyzed the performance of the main kernels, *propagate* and *collide*, of the LBE3D application on various models of the Intel processor family. In particular we have analyzed the last four generations, from the Westmere to the Skylake, including the KNL many-core system. In Figure 6.9 we present the results obtained by the *propagate* kernel in 10 years of processors manufacturing by using one thread per core while filling all the cores available on the various sockets. The analysis confirms how LBE3D is capable, when using the clustered data layout, to exploit the whole memory bandwidth available on the various systems, see Figure 6.9a. The memory per core would remain almost constant over the years, if considering the canonical layouts AoS and SoA, but the clustered structure shows an increase of the memory bandwidth per core. On the other hand, the Figure 6.9b presents the data for the *collide* kernel on several platforms, showing a higher value of FLOP/s per core for the SKL-G processor but again, showing how the KNL processor offers the best performances.

### 6.2.3. Poiseuille benchmarks on hybrid accelerated systems

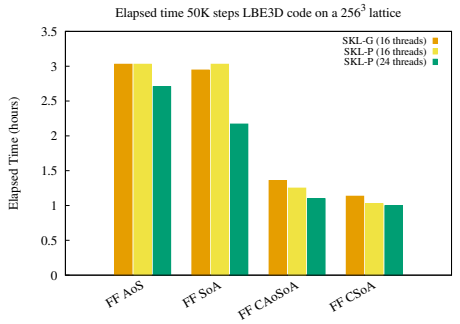
In this section we present the performances of the Poiseuille benchmarks on a socket of Marconi100, considered as 8-cores of the Power9 CPU, and one V100 NVIDIA GPU. In Figure 6.10 we report the results of the OpenACC porting of the compute intensive kernel of the numerical code for modeling the Poiseuille flow in the fully fused ($FF$) version which shows how the clustered



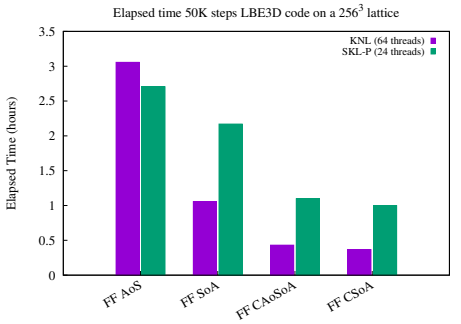

**(a)** Measured Memory Bandwidth




**(b)** Measured Memory Bandwidth



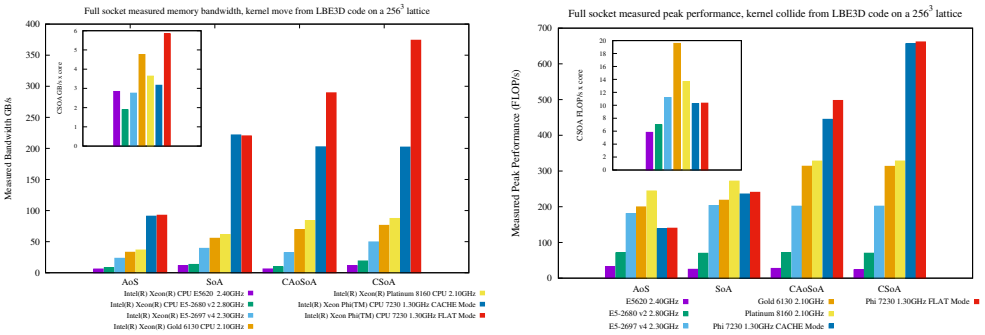

**(c)** Time for LBM loop




**(d)** Time for LBM loop

***Figure 6.8.:*** *We report compute performance value of the optimized data layouts measured on both the SKL and the KNL processors. Figure6.8a reports the data related to the memory bandwidth measure with the propagate kernel on the SKL Gold 6230 and the SKL Platinum 8160. To emphasise the level of memory bandwidth saturation for the SKL Platinum 8160 we also report the numbers obtained using only 16 threads of the 24 available on socket. A comparison of the performance of memory bandwidth between the SKL and the KNL processor is reported in 6.8b. Figure 6.8c and 6.8d compare the performance, in terms of time-to-solution (full LB loop), of the three processors for the most optimized version of the LBE3D.*

data-layouts, the best on regular x86-64 based CPUs, do not perform well on GPU, where instead the legacy data-layout, Structure of Arrays (SoA) delivers the best performance. Single GPU results was gathered using the NVIDIA V100 nodes installed at CINECA Marconi100 partition and the CPU results are for the SKL-P partition installed at Barcelona Supercomputing Center and Intel Knights Landing cluster at CINECA-KNL partition. The results shows how the clustered data structures deliver high-performance on x86-64 based CPU [88], while delivering low performance on less cache effient




**(a)** Measured Memory Bandwidth **(b)** Measured Peak Performance

***Figure 6.9.:*** *We report in 6.9a the measured memory bandwidth for the propagate kernel while in 6.9b the measured peak performance for the collide kernel. In this case, we compare several models representing 10 years of CPU manufacturing from the Intel product family, including the KNL many-core system. The performance measurement is reported using a single MPI process on the socket while scaling the number of threads up to the maximum number of core available on the given CPU socket, and it is performed on a lattice size of $256^3$ representing a real workload.*

architectures such as GPUs. Moreover, the OpenACC programming paradigm is meant to convert existing code into GPU codes without major effort from the developers but still not enough mature to handle efficiently optimize complex structures, such as the case of the clustered data-layouts CSoA and CAoSoA. As the OpenACC translates the C code of the LBE3D in GPU code, we expect the clustered version are also penalized because implemented on a more complex data structure. However, as for a first analisys of the LBE3D on hybrid accelerated platforms we consider those first results quite promising. Figure 6.11 shows the speedup we achieved comparing best GPU version, considered as the $FF\_SoA$, with the best CPU version, considered as the $FF\_CSoA$. We compare 1 GPU Tesla V100, with 1 GPU Tesla P100 (oldest generation) with single socket CPUs of different Intel processors. We see that moving from V100 to P100 we gain a factor x2 in speed and about x2.5 if compared with the KNL processor, in its best configuaration, therefore where the data fits the on-chip memory. The V100 outperforms Intel x86-64 CPUs for high-end computing by several factors. In particular we underline the  x8 factor speedup if comparared with the SKL-P socket, equipping the compute nodes of MareNostrum4.

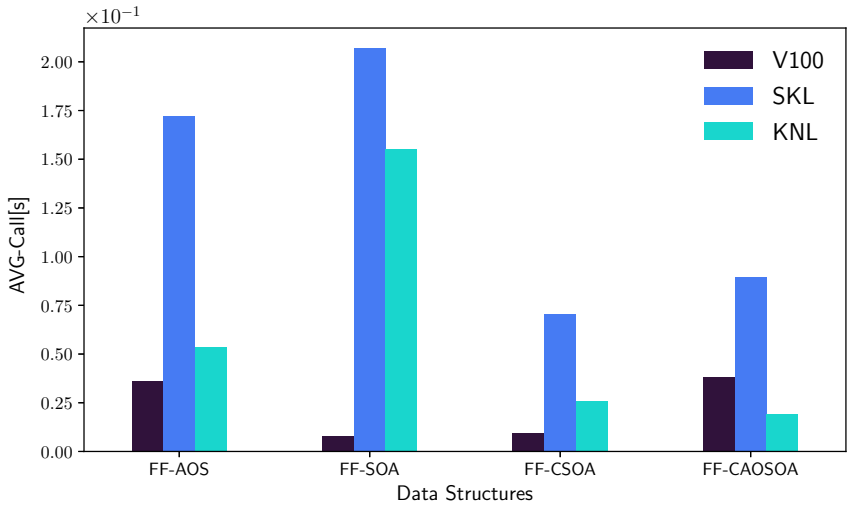



***Figure 6.10.:*** *Average call per iteration for the Poiseuille benchmark for different data structures, with FF standing for fully fused version of the kernels. It is evident that SoA has the least average time per iteration. Note that on CPU, CSoA data structures has the best performance, while on GPU the legacy SoA data structure has the least average time per iteration.*

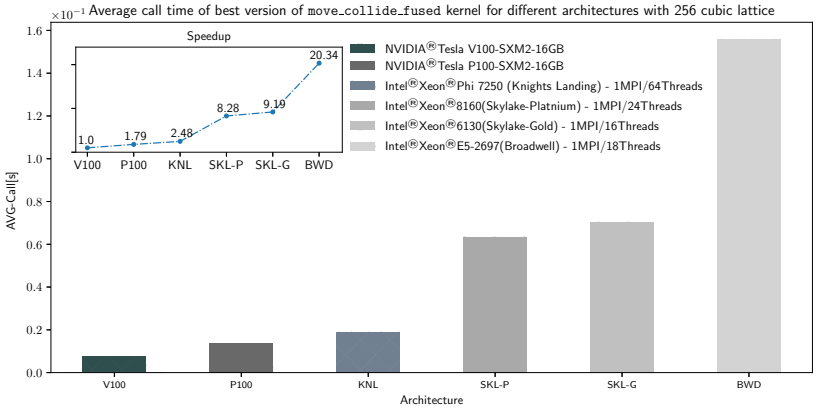



***Figure 6.11.:*** *Speedup and Average call time for different architectures, with 256 cubic lattice. For CPU versions we have used the fully fused CSoA kernel data and for GPU the fully fused version of SoA has been used.*

**Algorithm 5:** Schematic algorithm of the multicomponent solver with OpenACC porting on the GPU.

```
Initialize ρ, u, f_i, and f_i^eq for each component ;
Create and update the lattice and the hydrodynamical variables on
GPU using OpenACC pragmas;
for i ← 0 to NUM_STEPS do
    Set periodic boundary condition for the first component lattice
     with optimized kernels for PBC;
    Fused propagation and hydrodynamical variables calculations;
    Perform boundary condition exchange for the lattice, without
     populations;
    Evaluate velocities;
    Calculate pseudo potential with the first component;
    Perform double-belt periodic boundary condition exchange;
    if shell forcing then
        Do an optimized fused multiphase double belt shell force;
    end
    else
        Do an optimized fused multiphase double belt without shell
         forcing;
    end
    Calculate pseudo potential with the second component;
    Perform boundary condition exchange for the lattice, without
     populations;
    if IO step then
        update the lattice and the hydrodynamical variables on the
        host using OpenACC pragmas;
        Compute hydrodynamical quantities;
        Do statistical analysis;
        Do diagnostics;
        Dump data;
    end
    Swap lattice pointers for each component;
end
```

### 6.2.4. The multicomponent LBM on distributed multi-GPUs

On the base of the results obtained with the SoA version of the LBE3D on GPUs we implemented a distributed multi-GPUs version of the multicomponent code

as presented in 3. In Algorithm 5 we present a schematic representation of how the implementation of the multicomponent LBM results in the OpenACC version, using the SoA data-layout. The multicomponent code is expected to scale on multi-GPUs as we target simulations at high-resolutions. Therefore, the porting of the multicomponent includes also the exchange of the boundaries, which is implemented with packing and unpacking via OpenACC. All operation of diagnostic and I/O are then left to the backward CPU platform, with the user of the application responsible to find a good trade-off between compute time on the GPU and the CPU time required to runtime analysis of the data and I/O operations. In addition we implement a fused version of the *propagate* kernel and for the calculation of the density, which is performed directly without restoring on the main memory the streamed lattice populations. While this would be irrelevant on CPU it becomes important on the GPUs where the data locality plays a significant role. Every compute node of the Marconi100 hosts 4 accelerators with a significant amount of compute capability per node. This translates in the need to fill as much as possible every node, to keep the GPUs always working at full speed. In Figure 6.12 we show the strong scaling with

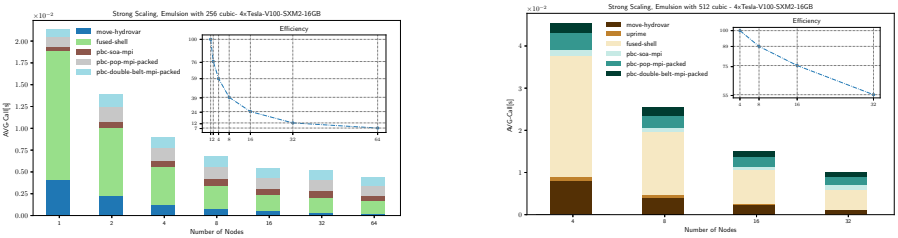



***Figure 6.12.:*** *Strong scaling for 256 and 512 cubic lattice multicomponent LBM.*

256 and 512 cubic lattice. While the 256 case offers a limited strong scaling already beyond a single node, for the bigger case we see a scaling with 75% of efficiency up to 16 nodes (64 GPUs). As limited in resources we do not provide results of the strong scaling for the highest resolution too. However, we had the chance to measure the weak scaling of the multicomponent LBE3D up to 128 compute nodes, for the highest resolution of 1024-dimension. We can show how our implementation of the LBE3D shows a weak scaling efficiency of 90% on a 4PFlop/s partition of Marconi100, if considering the 256-dimension case on 1 node, the 512-dimension case on 8 nodes and the 1024-dimension case on 128 nodes. With those numbers we expect to target emulsions at the resolution of either 2048-dimension or even 4096-dimension in the next projects.

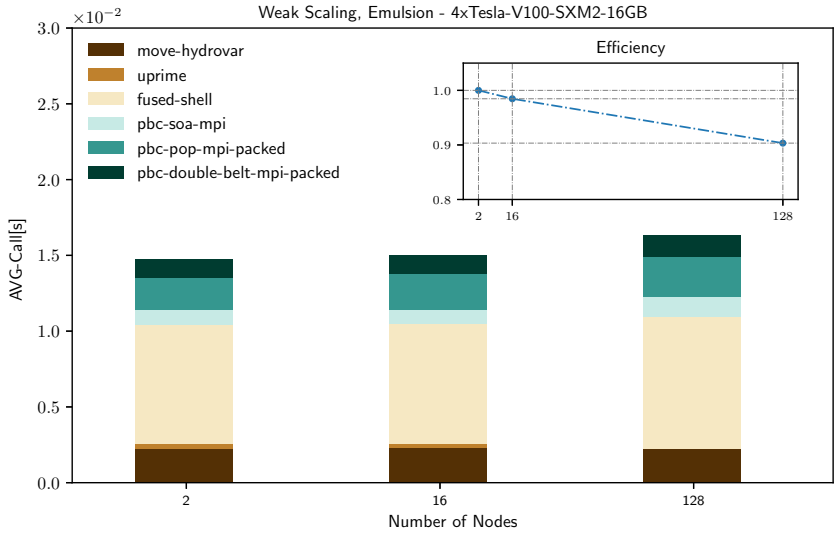



***Figure 6.13.:*** *Weak scaling for multicomponent with 2, 16, and 128 nodes with 256, 512, and 1024 cubic lattices.*

The positive feedbacks on the LBE3D porting on multi-GPUs platforms is also consolidated if comparing results between Marconi100 and MareNostrum. In Figure 6.14 we report the average call time we registered in production, while performing the presented emulsions on MareNostrum4, comparing with the results obtained on Marconi100. In particular we compare how many nodes of MareNostrum versus Marconi100 are needed to perform the same simulation at the resolution of 512-dimension and 1024-dimension. The results show how this ration is measured to be approximatly 8. In other words, we estimate that x8 nodes of MareNostrum are equivalent to one node of Marconi100 as for our multicomponent code. Nominally a node of Marconi100, about 32TFLOP/s in double precision, corresponds to approximately 10 nodes of MareNostrum4, each approximately 3.2TFLOP/s in double precision. However, we are comparing an highly optimized CPU version with a first acceptable implementation for hybrid accelerated platforms.

## 6.3. Analysis of Energy Efficiency

We now consider energy efficiency for the LBE3D across the multiple data layouts presented. We use data from the Running Average Power Limit (RAPL)

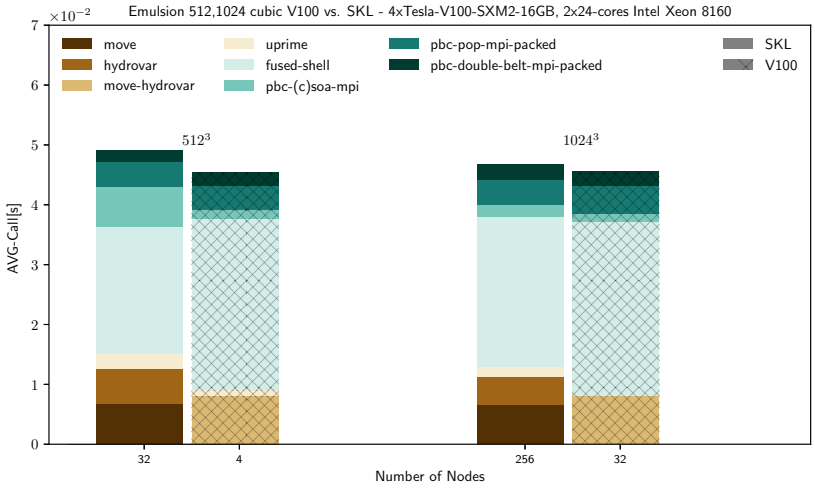



***Figure 6.14.:*** *Comparison between V100 and SKL nodes of CINECA. The first group is with 512 cubic and the second group is with 1024 cubic lattices.*

register counters available in the KNL, read through the custom library developed in [97]. In Figure 6.15 we show the measured values of energy consumption (Joule) for the LBE3D application respectively for the processor and the off-chip DRAM memory.

The DRAM energy consumption is lower as the KNL is configured in Flat mode and during the simulation data are all stored into the MCDRAM. Indeed, energy consumption for the DRAM memory only registers the value in the state of idle, while for the MCDRAM is accounted within the CPU (on cheap memory).

What is relevant to notice is that despite the CSoA and CAoSoA data layouts are expected to stress the CPU system more than the AoS and SoA (higher utilizzation of the VPU), we can assume the absorbed power remains approximately constant when considering different data layouts. Indeed, it is evident how the energy consumption remains mostly proportional to the time-to-solution, Figures 6.15: usage of the CSoA data layout brings a factor from x2 to x3 advantage both in term of time-to-solution, and energy to solution.

As for the KNL processor, we measured the energy efficiency on the SKL processor, on a real workload and analyzing all the discussed data layouts implemented in the LBE3D code. In particular, we measure the energy consumption of the processor package and of the DRAM memory system, using the RAPL PACKAGE_ENERGY and DRAM_ENERGY hardware counters.The PAPI

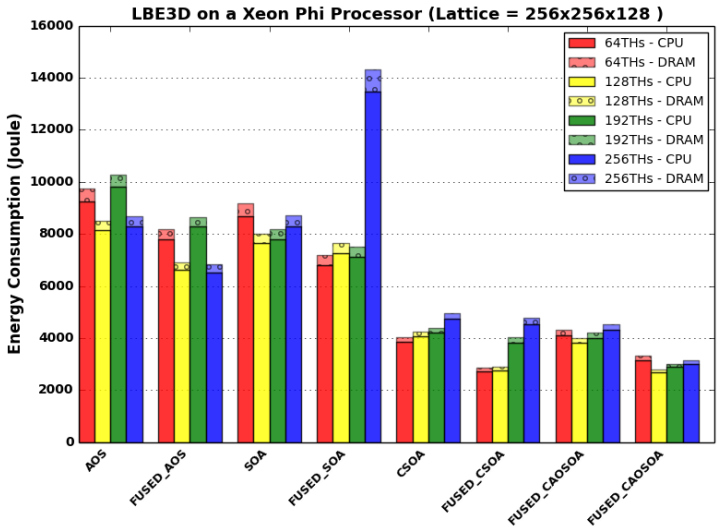
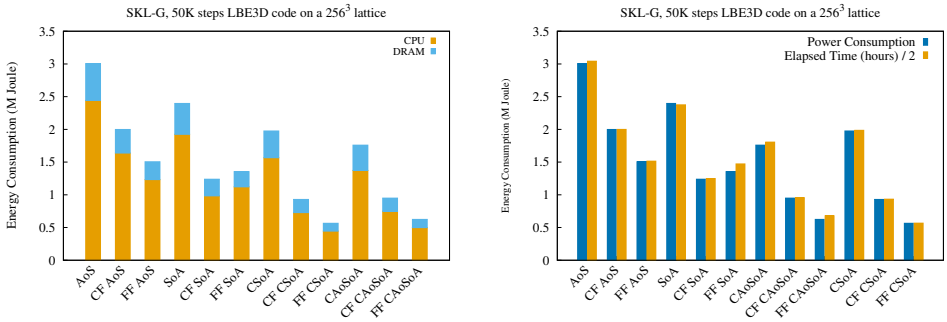



***Figure 6.15.:*** *Energy consumption profiling of the LBE3D application.*

[98] library is considered an established practice for accessing the counters, as already validated in other studies [99, 100]. On top of the PAPI library, we have developed a custom library [97] to manage power/energy data acquisition from hardware registers by simply literally adding in the LBE3D three lines of code for the initialization of the runtime environment, the start and the stop of the measurement.

The library allows to read at runtime the register counters available in the SKL processor. In Figure 6.16a, we show in details the measured values of energy consumption (MJoule) for the LBE3D application, respectively, for the CPU and the DRAM memory. In the Figure 6.16b, we report on the same scale the total (CPU and the DRAM memory together) energy consumption compared with the elapsed time of the simulation. It is relevant to notice that, despite the CSoA and CAoSoA data layouts are expected to stress the CPU system more than the AoS and SoA (higher utilisation of the VPU), we can assume that the absorbed power remains approximately constant when considering different data layouts. This is confirmed also by the data reported in Figure 6.17c where we compare the average power absorbed by the LBE3D in the FF version, comparing the SKL and the KNL processor. Therefore, the energy consumption of the LBE3D application on real workloads remains mostly proportional to the time-to-solution and, how reported in Figure 6.16b, there is almost a constant ratio of one million Joules of energy consumed every




**(a)** Details of energy consumption on the SKL-G processor

**(b)** Energy consumption and time on the SKL-G processor

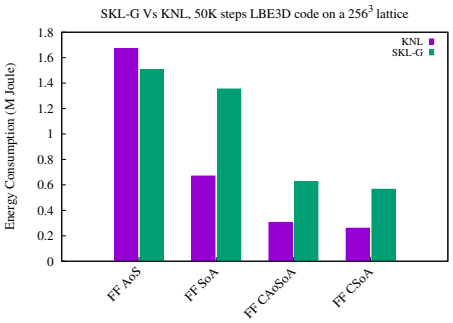
***Figure 6.16.:*** *Measure of energy consumption for the LB main loop on SKL-G processor. In (a) the measured energy efficiency is reported for 50K time steps of the LB main loop on a $256^3$ lattice measuring all considered data layout.*

2 hours, if considering a single SKL-G socket.

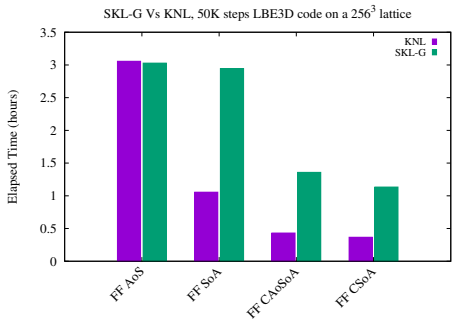
Finally, we consider the energy efficiency when running the same simulation on the SKL-G and on the KNL processor, Figure 6.17. It is evident how the optimized clustered structures result to be from 2 times to 2.5 times more efficient, considering the two analyzed architectures singularly, if comparared with the canonical SoA data layout but even more if considering the AoS. It is also interesting to see how the KNL is the most efficient architecture for running LBM based application implementing the proposed clustered layouts considering power consumption, Figure 6.17a, as well as time-to-solution, Figure 6.17b. Indeed, the memory bandwidth limit of the application remains the main bottleneck and therefore, the KNL delivering a really high memory bandwidth when using efficiently the memory on chip, it gives the best performances. Despite the SKL-G processor is 15-25% more efficient if looking at the average power drain, Figure 6.17c, it results twice less efficient in energy-to-solution. It confirms how lower average power drain is not much significant to improve efficiency of high-performance computing applications if not integrated over the application execution time (or *time-to-solution*, $T_s$) to obtain the application consumed energy (or *energy-to-solution*, $E_s$):

$$E_s = T_s \times P_{avg}$$

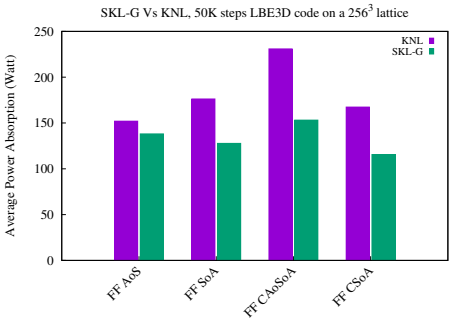
thus an increase in $T_s$ may lead to an $E_s$ increase, despite a lower average power drain $P_{avg}$. The performance gap between the SKL-G and the KNL



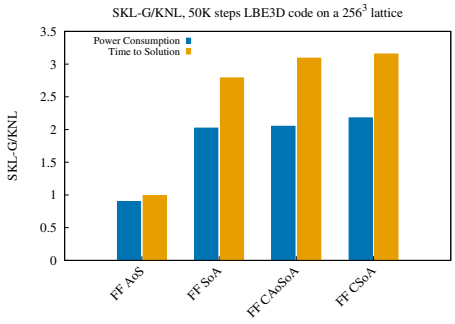

**(a)** Energy consumption: SKL-G Vs KNL




**(b)** Time-to-solution: SKL-G Vs KNL




**(c)** Power absorption: SKL-G Vs KNL




**(d)** Performance ratio: SKL-G/KNL

***Figure 6.17.:*** *The Figure reports an overview of the energy efficiency, when comparing the SKL-G and the KNL processors on the same simulation using the LBE3D. In all cases only the the fully fused version of the LBE3D is reported, to compare the two architectures in respect to the most optimized configuration. In (a) we compare the energy consumption, while in (b) we provide a view of the time-to-solution. In (c) we report the average power absorption and finally, in (d), we summarize the performance gap between the two architectures.*

¡

processors is also summarized in Figure 6.17d, where we report the ratio, in both terms of power consumption and time-to-solution, while comparing the two architectures on the same workload.

# 7. Conclusions and remarks

In this research we engineered and developed, a series of highly-optimised computational tools capable to reproduce, via fully resolved numerical simulations, the process of emulsification at high-resolution in 3-dimensions. To the best of our knowledge this is the first time ever that a numerical code could reproduce the dynamic of the formation of a yield stress fluid, from stirring a multicomponent fluid. Our numerical code, based on the multicomponent LBM include, surface tension and disjoining pressure, between the two immiscible fluid components and thanks to a large scale, hydrodynamic forcing, has been employed to simulate the process of chaotic emulsification at the mesoscopic level. We simulated the process of emulsions formation as well as their statistically stationary dynamics, for millions of LBM time steps, for a large number of emulsions, at varying the resolution, the magnitude of the hydrodynamic stirring and the volume fraction of the dispersed phase. In case of highly concentrated emulsions (volume fraction of the dispersed phase $\sim 80\%$) we demonstrated how this process resulted in fluids with a finite yield stress. The qualitative interpretation of the dynamics is also supported by high-resolution renderings of the morphology of these complex systems, clearly displaying relevant features associated with internal frustration of high-packing ratios, such as complex non-spherical droplets shapes.
We designed and developed a novel and efficient computational approach for accurately tracking droplets in dense emulsions, as a possible solution in order to extend the capability to study Lagrangian dynamics of the LBM, that intrinsically describe droplet at an Eulerian level. The high Lagrangian tracking accuracy of our approach ($\epsilon \sim 10^{-5}$) makes it a unique tool capable to provide information on emulsion complementary and difficult to measure in experiments. We analyse the trajectories of individual droplet, allowing to described the processes of emulsification at the mesoscopic level, showing relevant statistics droplet morphology, including droplet size distribution, PDF of droplets accelerations and velocities, as well as droplets dispersion. All these statistical quantities have been measured with high statistical accuracy, corresponding to samples between $10^6$ and $10^8$. Results highlights that the initial interface fragmentation shows a droplet size distribution (DSD) compatible with a power-law with slope close to $-10/3$, commonly associated to droplets

distribution in turbulent flows. At a constant magnitude of the hydrodynamic chaotic stirring, highly-packed emulsions, if compared with dilute emulsions, present an increase in the droplet average radii, a reduction in the number of droplets and a lower hydrodynamic velocity, due to an increase of the effective viscosity. We observed that, in highly concentrated emulsions, the rapid reduction of the number of droplets, along with the concurrent increase of the average radii, is a signal of approaching the threshold leading to occurrence of catastrophic phase inversion. Our analysis also shows that the DSD of very dense emulsions present a bimodal distributions when flowing under chaotic flow, displaying a secondary peak about one order of magnitude below the mean peak for the droplets radii, $< R >$, and one peak approximately in correspondence, evidence of the relevant presence of small droplets. As consequence of this characteristics morphology of dense emulsions, we report PDF for droplets accelerations with fat tails that we could fit with stretched exponential.
All the computational tools introduced in this thesis, come with a non-trivial computational cost, particularly when dealing with high-resolution simulations, as well as when dealing with the tracking of a very large number of droplets, for simulations millions of LBM time steps long. We tackled this problem by introducing highly-optimized data-layouts suitable for the classic LBM algorithm, that exploits the computational capabilities of modern generation CPUs with vectorial sets of instructions as well as of hybrid CPU/GPU system. In particular, the data organisation follows a clustered Structure of Arrays (CSoA) layout that allow to achieves $\sim 85\%$ of the nominal memory bandwidth on the recent KNL and Skylakes Intel CPUs, and $\sim 25\%$ computational performance, when considering the multicomponent LBM algorithm being mostly memory bounded. We developed a version of the LBM multicomponent, including MPI+OpenMP+OpenACC capable run efficiently on hybrid distributed CPU/GPU Tier-0 systems for high-performence scientific computing, exploiting, instead, the Structure of Arrays (SoA) layout. The latest version of the multicomponent LBM code presents a fair performance portability between distributed CPU systems and distributed CPU/GPU hybrid system. In term of time to solution, scaling on a large number of nodes, we measured 1 compute node of the Marconi-100 hosted at CINECA being comparable with 8 compute nodes of the MareNostrum4, hosted at Barcelona Supercomputing Centre. We presented a detailed performance analysis of those data-layouts on a number of different CPUs and GPU models, providing an insight on the development of LBM-based codes on modern and future valuable HPC architectures.

## 7.1. Perspective on future works

This work demonstrated that fully resolved numerical models based on the LBM allows to perform numerical studies that are key in order to connect the microscopic dynamics with the large-scale rheology and vice-versa. The methodology presented in this thesis can provide an essential contribution in order to considerably advance the level of knowledge on emulsification processes. Additionally, we demonstrated that this type of fully resolved numerical models is within reach of current computer capabilities. In this thesis we limited ourself to preliminarly studies of the dynamics of multicomponent fluids, where the two immiscible components are characterised by the same values of density and viscosity. Our numerical studies allow investigating small-scale morphology of emulsions under flow, opening the possibility to develop effective CFD models based, for example, on population balance. It is worth noticing that more recent multicomponent LBM, e.g. the color-gradient multicomponent LBM [101], promise to overcome the limitations, e.g. on the density and the viscosity ratios, intrinsic of the Shan-Chen model (Chapter 3) [102, 103, 104]. Results of this project promise to disclose new opportunities on several research fields such as emulsion physics, droplets and particles dynamics and, more generally, of the phenomenology of soft-glassy systems. Moreover, this works demonstrated how the presented numerical models can efficiently scale on Tier-0 system for HPC, exploiting current generation of multi-GPUs distributed system. This promises that this method will be efficiently deployed on future architectures as well. The large statistical database produced within this work has only partially been explored, leaving room for numerous future investigations aimed at answering the many open questions.

# A. Computer Architectures

In here, we briefly describe the computer architectures used to produce the results presented in Section 6.2. We first present an overview of all the Intel processors employed in the performance measurement on CPU platforms. Then we focus on the KNL, on the SKL processor and on the NVIDIA GPU V100. Those are the reference architectures for this analysys as featuring the world-class facilities we deployed this project on, the MareNostrum4 and the Marconi-KNL partition, as well as the possible target platform for next projects, such as the Marconi-100, at CINECA. A summary of all the architectures is given in Table A.1 to facilitate the reading. There, the column "Short name" indicates how we will refer to a specific processor model for the rest of the paper, avoiding to mention the entire processor name and model repeatedly.

***Table A.1.:*** *The table reports a summary of all computer architectures of the Intel Processors Family mentioned in this section. In all cases the version 2018 of the Intel Compiler has been used to compile the LBE3D application but for the ICTP cluster where the version 2017 is available.*

| Host | Proc. Name | Proc. Model | #Cores | Compiler Flags | Short name | VL |
|---|---|---|---|---|---|---|
| ICTP | Westmere | E5620 @ 2.40GHz | 4 | -sse4 | WEP | 1 |
| ICTP | Ivy Bridge | E5-2680 v2 @ 2.80GHz | 10 | -xavx | IB | 2 |
| CINECA | Broadwell | E5-2697 v4 @ 2.30GHz | 18 | -xavx2 | BWD | 4 |
| CINECA | Knights Landing | Phi7250 @ 1.40GHz | 68 | -xMIC-AVX512 | KNL | 8 |
| INFN | Skylake | Gold 6130 @ 2.10GHz | 16 | -xCORE-AVX512 | SKL-G | 8 |
| INFN | Knights Landing | Phi7230 @ 1.30GHz | 64 | -xMIC-AVX512 | KNL | 8 |
| BSC | Skylake | Platinum 8160 @ 2.10GHz | 24 | -xCORE-AVX512 | SKL-P | 8 |

The SKL is a member of the last generation of the 14nm Intel Xeon Processor Scalable Family. It is available on market in various versions categorized as Bronze, Silver, Gold and Platinum, on the base of different architectural aspects. Indeed, the metal indicates a growing computational capability from low performance consumer hardware to processors for high-performance computing. We are interested in measuring the performance of the LBE3D on the SKL-P processor as we are committed to a large number of simulations on MareNostrum at BSC. However, for the analysis on the energy consumption of the LBE3D on the SKL processor we refer to the SKL-G processor available in our lab, at the University of Ferrara, because the measurements require *root* privileges. We also highlight performances of both the SKL-P and the

SKL-G processors as some interesting results came up while looking at the performance numbers obtained on the two architectures. The processors have comparable characteristics in terms of memory bandwidth being both equipped with 6 Double Data Rate fourth-generation (DDR4) channels of synchronous Dynamic Random-Access Memory (DRAM), but delivering a significant difference in peak performance since the SKL-P hosts 24-cores while the SKL-G version comes with 16-cores. From the x86 based many-cores series of the Intel processor family, the KNL is a processor model designed for high-end computing. It is equipped with 6 DDR4 channels of DRAM, with a nominal bandwidth of 115.2 GB/s and four high-speed memory banks based on the Multi-Channel DRAM (MCDRAM) that provides 16 GB of memory, capable to deliver an aggregate bandwidth of more than 450 GB/s when accessing the on-package high-bandwidth memory. The processor can be configured in three different modes: Flat, Cache and Hybrid mode. Each mode is characterised by how the program execution accesses to the MCDRAM at runtime as, to achieve high-performance, it is fundamental to exploit the MCDRAM. The Flat mode defines the whole MCDRAM as addressable memory allowing explicit data allocation, whereas Cache mode uses the MCDRAM as a last-level cache. In a previous work [89] we analyzed in detail how the LBE3D, being a massively scalable code, can well exploit the MCDRAM memory if keeping the amount of memory per node around the 16GB while scaling up the number of nodes for large problems. For the analysis presented in this paper we focus on the KNL configured in Flat mode because it maximizes the usage of the MCDRAM at runtime. The Marconi-A2 partition (recently out of production) was based on the KNL processor 7250. However, for the analysis we refer to the 64-cores KNL processor 7230 available in our lab, because of the same reason we mentioned above for the SKL processor. The Intel library implementing the Message Passing Interface (MPI) was used together with the Intel compiler to compile and build the LBE3D application which requires the MPI even if running on single process. The Table A.1 reports the Intel compiler version used to build the application on the various system. The same level of optimization (-O3) was also used in all cases on top of the specified flags for auto-vectorization. The "VL" column of the table shows the vector length ($VL$) used to compile the LBE3D, accordingly to the instruction set supported by a given CPU model, as it is described in the following Section. The LBE3D is a highly scalable application that implements a distributed memory approach for inter-node communication while multi-threading via OpenMP is implemented to exploit the intra-node compute capability by scaling with the number of threads. The threads affinity is set at run time with the Intel environment

variables “KMP_AFFINITY=compact” and “I_MPI_PIN_DOMAIN=socket”. All presented results are related to a single socket execution therefore using a single MPI processes and a given number of threads accordingly to the number of cores available. The same approach was used for all benchmarks. GPU architecture differs significantly from that of CPUs. NVIDIA GPUs have hundreds of cores that can process thousands of software threads simultaneously. GPUs are today connected to the hosting CPU platform through a high-speed I/O bus, typically PCI-X or NVLINK, and with their own device memory (Tesla V100 comes in 16 and 32GB, configurations). For most applications, implementing efficient data transfer between the hosting the devices and the GPUs is still a key aspect for high-performance. The V100 memory supports very high data bandwidth up to 900 GB/s and approximately between, delivering a peak performance in double precision of approximately 7.8 TFLOP/s. In this section we take into consideration two accelerated hybrid platforms, one hosted at the ICTP and one composing the compute nodes of Marconi-100. The fist, to which we attribute the name *GPU*
*_ICTP* is a single compute node equipped with x2 NVIDIA Quadro GP100 and 2x10 cores Intel Broadwell CPU E5-2640v4 at 2.40GHz. The second is the Marconi-100 at CINECA where each compute node is equipped with 2x16 cores IBM POWER9 AC922 at 3.1 GHz, and 4 x NVIDIA Volta V100 GPUs, Nvlink 2.0, 16GB. The Marconi-100 is composed by 980 compute nodes interconnected with Mellanox Infiniband EDR DragonFly+ for an overall peak performance of 32 PFlop/s.

# Bibliography


[1] Peter Sollich. Rheological constitutive equation for a model of soft glassy materials. *Phys. Rev. E*, 58:738–759, Jul 1998.

[2] P. Hébraud and F. Lequeux. Mode-coupling theory for the pasty rheology of soft glassy materials. *Phys. Rev. Lett.*, 81:2934–2937, Oct 1998.

[3] Michel Cloitre, Régis Borrega, and Ludwik Leibler. Rheological aging and rejuvenation in microgel pastes. *Phys. Rev. Lett.*, 85:4819–4822, Nov 2000.

[4] Roberto Benzi, Pinaki Kumar, Federico Toschi, and Jeannot Trampert. Earthquake statistics and plastic events in soft-glassy materials. *Geophysical Journal International*, 207(3):1667–1674, 09 2016.

[5] Fabrizio Gagliardi, Miquel Moreto, Mauro Olivier, and Mateo Valero. The international race towards exascale in europe. *CCF Transactions on High Performance Computing*, 1:3–13, 2019.

[6] Xiaowen Shan and Hudong Chen. Lattice boltzmann model for simulating flows with multiple phases and components. *Phys. Rev. E*, 47:1815–1819, Mar 1993.

[7] Xiaoyi He, Shiyi Chen, and Raoyang Zhang. A lattice boltzmann scheme for incompressible multiphase flow and its application in simulation of rayleigh–taylor instability. *Journal of Computational Physics*, 152(2):642 – 663, 1999.

[8] S. Succi. *The Lattice Boltzmann Equation for Fluid Dynamics and Beyond (Numerical Mathematics and Scientific Computation)*. Numerical mathematics and scientific computation. Oxford University Press, 1 edition, August 2001.

[9] Xiaowen Shan, Xue Feng Yuan, and Hudong Chen. Kinetic theory representation of hydrodynamics: A way beyond the navier-stokes equation. *Journal of Fluid Mechanics*, 550:413–441, mar 2006.

[10] Junfeng Zhang. Lattice boltzmann method for microfluidics: models and applications. *Microfluidics and Nanofluidics*, 10(1):1–28, Jan 2011.

[11] S. Arcidiacono, I. V. Karlin, J. Mantzaras, and C. E. Frouzakis. Lattice boltzmann model for the simulation of multicomponent mixtures. *Phys. Rev. E*, 76:046703, Oct 2007.

[12] Ciro Semprebon, Timm Krüger, and Halim Kusumaatmaja. Ternary free-energy lattice boltzmann model with tunable surface tensions and contact angles. *Phys. Rev. E*, 93:033305, Mar 2016.

[13] Prasad Perlekar, Luca Biferale, Mauro Sbragaglia, Sudhir Srivastava, and Federico Toschi. Droplet size distribution in homogeneous isotropic turbulence. *Physics of Fluids*, 24(6):065101, 2012.

[14] Prasad Perlekar, Roberto Benzi, Herman J. H. Clercx, David R. Nelson, and Federico Toschi. Spinodal decomposition in homogeneous and isotropic turbulence. *Phys. Rev. Lett.*, 112:014502, Jan 2014.

[15] Luca Biferale, Prasad Perlekar, Mauro Sbragaglia, Sudhir Srivastava, and Federico Toschi. A lattice boltzmann method for turbulent emulsions. *Journal of Physics: Conference Series*, 318(5):052017, dec 2011.

[16] Enrico Calore, Nicola Demo, Sebastiano Fabio Schifano, and Raffaele Tripiccione. Experience on Vectorizing Lattice Boltzmann Kernels for Multi- and Many-Core Architectures. In *Parallel Processing and Applied Mathematics: 11th International Conference, PPAM 2015, Krakow, Poland, September 6-9, 2015. Revised Selected Papers, Part I*, Lecture Notes in Computer Science, pages 53–62. Springer International Publishing, Cham, 2016.

[17] Enrico Calore, Alessandro Gabbana, Sebastiano Fabio Schifano, and Raffaele Tripiccione. Early experience on using knights landing processors for lattice boltzmann applications. In *Parallel Processing and Applied Mathematics: 12th International Conference, PPAM 2017, Lublin, Poland, September 10-13, 2017*, volume 1077 of *Lecture Notes in Computer Science*, pages 1–12, 2018.

[18] Enrico Calore, Alessandro Gabbana, Jiri Kraus, Sebastiano Fabio Schifano, and Raffaele Tripiccione. Performance and portability of accelerated lattice boltzmann applications with openacc. 28(12):3485–3502, August 2016.

[19] R. Benzi, S. Succi, and M. Vergassola. The lattice boltzmann equation: theory and applications. *Physics Reports*, 222(3):145 – 197, 1992.

[20] A. Scagliarini, H. Einarsson, A. Gylfason, and F. Toschi. Law of the wall in an unstably stratified turbulent channel flow. *Journal of Fluid Mechanics*, 781:R5, 2015. doi: 10.1017/jfm.2015.498.

[21] G. Di Staso, S. Srivastava, E. Arlemark, H.J.H. Clercx, and F. Toschi. Hybrid lattice Boltzmann-direct simulation Monte Carlo approach for flows in three-dimensional geometries. *Computers & Fluids*, 2018.

[22] A. Gupta, H.J.H. Clercx, and F. Toschi. Simulation of finite-size particles in turbulent flows using the lattice Boltzmann method. *Communications in Computational Physics*, 23(3):665–684, 2018.

[23] Jack Dongarra and Piotr Luszczek. *TOP500*, pages 2055–2057. Springer US, Boston, MA, 2011.

[24] J. E. Jones and Sydney Chapman. On the determination of molecular fields.—i. from the variation of the viscosity of a gas with temperature. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, 106(738):441–462, 1924.

[25] J. E. Jones and Sydney Chapman. On the determination of molecular fields. —ii. from the equation of state of a gas. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, 106(738):463–477, 1924.

[26] William L. Jorgensen, David S. Maxwell, and Julian Tirado-Rives. Development and testing of the opls all-atom force field on conformational energetics and properties of organic liquids. *Journal of the American Chemical Society*, 118(45):11225–11236, 1996.

[27] Mark James Abraham, Teemu Murtola, Roland Schulz, Szilárd Páll, Jeremy C. Smith, Berk Hess, and Erik Lindahl. GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. *SoftwareX*, 1:19–25, September 2015.

[28] Steve Plimpton. Fast parallel algorithms for short-range molecular dynamics. *Journal of Computational Physics*, 117(1):1 – 19, 1995.

[29] Szilárd Páll, Artem Zhmurov, Paul Bauer, Mark Abraham, Magnus Lundborg, Alan Gray, Berk Hess, and Erik Lindahl. Heterogeneous parallelization and acceleration of molecular dynamics simulations in gromacs. *The Journal of Chemical Physics*, 153(13):134110, 2020.

[30] William Humphrey, Andrew Dalke, and Klaus Schulten. VMD – Visual Molecular Dynamics. *Journal of Molecular Graphics*, 14:33–38, 1996.

[31] John Stone, Justin Gullingsrud, Paul Grayson, and Klaus Schulten. A system for interactive molecular dynamics simulation. In John F. Hughes and Carlo H. Séquin, editors, *2001 ACM Symposium on Interactive 3D Graphics*, pages 191–194, New York, 2001. ACM SIGGRAPH.

[32] S. Succi. *The Lattice Boltzmann Equation: For Complex States of Flowing Matter*. Numerical Mathematics and Scientific Computation Series. Oxford University Press, 2018.

[33] S.R. de Groot and P. Mazur. *Non-equilibrium Thermodynamics*. Dover Books on Physics. Dover Publications, 1984.

[34] D.E. Rosner. *Transport Processes in Chemically Reacting Flow Systems*. Dover Civil and Mechanical Engineering. Dover Publications, 2012.

[35] V. Giovangigli. *Multicomponent Flow Modeling*. Modeling and Simulation in Science, Engineering and Technology. Birkhäuser Boston, 2012.

[36] R.J. Kee, M.E. Coltrin, and P. Glarborg. *Chemically Reacting Flow: Theory and Practice*. Wiley, 2005.

[37] R. Benzi, M. Sbragaglia, S. Succi, M. Bernaschi, and S. Chibbaro. Mesoscopic lattice boltzmann modeling of soft-glassy systems: Theory and simulations. *The Journal of Chemical Physics*, 131(10):104903, 2009.

[38] R. Benzi, M. Bernaschi, M. Sbragaglia, and S. Succi. Herschel-bulkley rheology from lattice kinetic theory of soft glassy materials. *EPL (Europhysics Letters)*, 91(1):14003, jul 2010.

[39] F. Leal-Calderon, V. Schmitt, and J. Bibette. *Emulsion Science: Basic Principles*. Springer New York, 2007.

[40] P.G. de Gennes, F. Brochard-Wyart, and D. Quere. *Capillarity and Wetting Phenomena: Drops, Bubbles, Pearls, Waves*. Springer New York, 2003.

[41] Akira Onuki. *Phase Transition Dynamics*. Cambridge University Press, 2002.

[42] J.S. Rowlinson and B. Widom. *Molecular Theory of Capillarity*. Dover books on chemistry. Dover Publications, 2002.

[43] U. Frisch. *Turbulence: The Legacy of A. N. Kolmogorov*. Cambridge University Press, 1995.

[44] P. Davidson. *Turbulence: An Introduction for Scientists and Engineers*. OUP Oxford, 2015.

[45] P.A. Davidson, Y. Kaneda, K. Moffatt, and K.R. Sreenivasan. *A Voyage Through Turbulence*. Cambridge University Press, 2011.

[46] Lewis Fry Richardson and Gilbert Thomas Walker. Atmospheric diffusion shown on a distance-neighbour graph. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, 110(756):709–737, 1926.

[47] Andrei Nikolaevich Kolmogorov, V. Levin, Julian Charles Roland Hunt, Owen Martin Phillips, and David Williams. Dissipation of energy in the locally isotropic turbulence. *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, 434(1890):15–17, 1991.

[48] A. N. KOLMOGOROV. On degeneration (decay) of isotropic turbulence in an incompressible viscous liquid. *Dokl. Akad. Nauk SSSR*, 31:538–540, 1941.

[49] A N Kolmogorov. LOCAL STRUCTURE OF TURBULENCE IN AN INCOMPRESSIBLE VISCOUS FLUID AT VERY HIGH REYNOLDS NUMBERS. *Soviet Physics Uspekhi*, 10(6):734–746, jun 1968.

[50] Raymond W. Flumerfelt. Drop breakup in simple shear fields of viscoleastic fluids. *Industrial & Engineering Chemistry Fundamentals*, 11(3):312–318, 1972.

[51] Ivan Fortelný and Josef Jůza. Description of the droplet size evolution in flowing immiscible polymer blends. *Polymers*, 11, April 2019.

[52] Francesco Greco. Drop deformation for non-newtonian fluids in slow flows. *Journal of Non-Newtonian Fluid Mechanics*, 107(1):111 – 131, 2002.

[53] S. Guido and F. Greco. Dynamics of a liquid drop in a flowing immiscible liquid. In *DYNAMICS OF A LIQUID DROP IN A FLOWING IMMISCIBLE LIQUID*, 2004.

[54] P.J.A. Janssen and P.D. Anderson. A boundary-integral model for drop deformation between two parallel plates with non-unit viscosity ratio drops. *Journal of Computational Physics*, 227(20):8807 – 8819, 2008.

[55] Geoffrey Ingram Taylor. The viscosity of a fluid containing small drops of another fluid. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, 138(834):41–48, 1932.

[56] J. O. Hinze. Fundamentals of the hydrodynamic mechanism of splitting in dispersion processes. *AIChE Journal*, 1(3):289–295, 1955.

[57] Marco E. Rosti, Zhouyang Ge, Suhas S. Jain, Michael S. Dodd, and Luca Brandt. Droplets in homogeneous shear turbulence. *Journal of Fluid Mechanics*, 876:962–984, 2019.

[58] C. Miguel and M. Rubi. *Jamming, Yielding, and Irreversible Deformation in Condensed Matter*. Lecture Notes in Physics. Springer Berlin Heidelberg, 2006.

[59] E. Dickinson. *An Introduction to Food Colloids*. An Introduction to Food Colloids. Oxford University Press, 1992.

[60] Peter Sollich, Fran çois Lequeux, Pascal Hébraud, and Michael E. Cates. Rheology of soft glassy materials. *Phys. Rev. Lett.*, 78:2020–2023, Mar 1997.

[61] H.A. Barnes, K.W.H.A.B. John Fletcher Hutton, J.F. Hutton, and K. Walters. *An Introduction to Rheology*. Rheology Series. Elsevier Science, 1989.

[62] K. Huang. *Statistical Mechanics*. Wiley, 1987.

[63] Xiaowen Shan and Hudong Chen. Simulation of nonideal gases and liquid-gas phase transitions by the lattice boltzmann equation. *Phys. Rev. E*, 49:2941–2948, Apr 1994.

[64] T. Krüger, H. Kusumaatmaja, A. Kuzmin, O. Shardt, G. Silva, and E.M. Viggen. *The Lattice Boltzmann Method: Principles and Practice*. Graduate Texts in Physics. Springer International Publishing, 2016.

[65] C. Cercignani. *The Boltzmann Equation and Its Applications*. Applied Mathematical Sciences. Springer New York, 1987.

[66] Florian Janoschek, Federico Toschi, and Jens Harting. Simulations of blood flow in plain cylindrical and constricted vessels with single cell resolution. *Macromolecular Theory and Simulations*, 20(7):562–570, 2011.

[67] Giacomo Falcucci, Stefano Ubertini, and Sauro Succi. Lattice boltzmann simulations of phase-separating flows at large density ratios: the case of doubly-attractive pseudo-potentials. *Soft Matter*, 6:4357–4365, 2010.

[68] S. Chibbaro, G. Falcucci, G. Chiatti, H. Chen, X. Shan, and S. Succi. Lattice boltzmann models for nonideal fluids with arrested phase-separation. *Phys. Rev. E*, 77:036705, Mar 2008.

[69] Xiaowen Shan. Pressure tensor calculation in a class of nonideal gas lattice boltzmann models. *Phys. Rev. E*, 77:066702, Jun 2008.

[70] R. Benzi, M. Sbragaglia, P. Perlekar, M. Bernaschi, S. Succi, and F. Toschi. Direct evidence of plastic events and dynamic heterogeneities in soft-glasses. *Soft Matter*, 10:4615–4624, 2014.

[71] R. Benzi, M. Sbragaglia, A. Scagliarini, P. Perlekar, M. Bernaschi, S. Succi, and F. Toschi. Internal dynamics and activated processes in soft-glassy materials. *Soft Matter*, 11:1271–1280, 2015.

[72] M. Sbragaglia, R. Benzi, M. Bernaschi, and S. Succi. The emergence of supramolecular forces from lattice kinetic models of non-ideal fluids: applications to the rheology of soft glassy materials. *Soft Matter*, 8:10773–10782, 2012.

[73] Linlin Fei, Andrea Scagliarini, Andrea Montessori, Marco Lauricella, Sauro Succi, and Kai H. Luo. Mesoscopic model for soft flowing systems with tunable viscosity ratio. *Phys. Rev. Fluids*, 3:104304, Oct 2018.

[74] Ivan Girotto, Karun Datadien, Gianluca Di Staso, and Federico Toschi. The chaotic life of mayonnaise, November 2019.

[75] G.E.J. Vaessen and H.N. Stein. The applicability of catastrophe theory to emulsion phase inversion. *Journal of Colloid and Interface Science*, 176(2):378 – 387, 1995.

[76] Siddhartha Mukherjee, Arman Safdari, Orest Shardt, Saša Kenjereš, and Harry E. A. Van den Akker. Droplet–turbulence interactions and quasi-equilibrium dynamics in turbulent emulsions. *Journal of Fluid Mechanics*, 878:221–276, 2019.

[77] R. Skartlien, E. Sollum, and H. Schumann. Droplet size distributions in turbulent emulsions: Breakup criteria and surfactant effects from direct numerical simulations. *The Journal of Chemical Physics*, 139(17):174901, 2013.

[78] Giovanni Soligo, Alessio Roccon, and Alfredo Soldati. Breakage, coalescence and size distribution of surfactant-laden droplets in turbulent flow. *Journal of Fluid Mechanics*, 881:244–282, 2019.

[79] Grant B. Deane and M. Dale Stokes. Scale dependence of bubble creation mechanisms in breaking waves. *Nature*, 418:839–844, 2002.

[80] Gijs Katgert, Andrzej Latka, Matthias E. Möbius, and Martin van Hecke. Flow in linearly sheared two-dimensional foams: From bubble to bulk scale. *Phys. Rev. E*, 79:066318, Jun 2009.

[81] Subeen Kim, KyuHan Kim, and Siyoung Q. Choi. Controllable one-step double emulsion formation via phase inversion. *Soft Matter*, 14:1094–1099, 2018.

[82] J. Hoshen and R. Kopelman. Percolation and cluster distribution. i. cluster multiple labeling technique and critical concentration algorithm. *Phys. Rev. B*, 14:3438–3445, Oct 1976.

[83] S. Frijters, T. Krüger, and J. Harting. Parallelised hoshen–kopelman algorithm for lattice-boltzmann simulations. *Computer Physics Communications*, 189:92 – 98, 2015.

[84] Pinaki Kumar, Roberto Benzi, Jeannot Trampert, and Federico Toschi. A multi-component lattice boltzmann approach to study the causality of plastic events. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 378(2175):20190403, 2020.

[85] A. La Porta, Greg A. Voth, Alice M. Crawford, Jim Alexander, and Eberhard Bodenschatz. Fluid particle accelerations in fully developed turbulence. *Nature*, 409:1017–1019, 2001.

[86] Prakash Vedula and P. K. Yeung. Similarity scaling of acceleration and pressure statistics in numerical simulations of isotropic turbulence. *Physics of Fluids*, 11(5):1208–1220, 1999.

[87] Ivan Girotto, Sebastiano Fabio Schifano, Enrico Calore, Gianluca Di Staso, and Federico Toschi. Performance and energy assessment of a lattice boltzmann method based application on the skylake processor. *Computation*, 8(2), 2020.

[88] Ivan Girotto, Sebastiano Fabio Schifano, Enrico Calore, Gianluca Di Staso, and Federico Toschi. Performance optimization of d3q19 lattice boltzmann kernels on intel knl. In *INFOCOMP 2018: The Eighth International Conference on Advanced Communications and Computation*, pages 31–36, 2018.

[89] Ivan Girotto, Sebastiano Fabio Schifano, Enrico Calore, Gianluca Di Staso, and Federico Toschi. Computational performances and energy efficiency assessment for a lattice boltzmann method on intel knl. Advances in Parallel Computing, 2019.

[90] S. Williams, J. Carter, L. Oliker, J. Shalf, and K. Yelick. Optimization of a Lattice Boltzmann computation on stateof-the-art multicore platforms. *Journal of Parallel and Distributed Computing*, 69:762–777, 2009.

[91] Samuel Williams, Jonathan Carter, Leonid Oliker, John Shalf, and Katherine A. Yelick. Lattice Boltzmann simulation optimization on leading multicore platforms. *2008 IEEE International Symposium on Parallel and Distributed Processing*, pages 1–14, 2008.

[92] Massimo Bernaschi, Massimiliano Fatica, Simone Melchionna, Sauro Succi, and Efthimios Kaxiras. A flexible high-performance lattice Boltzmann gpu code for the simulations of fluid flows in complex geometries. *Concurrency Computat.: Pract. Exper.*, pages 22: 1–14, 2009.

[93] James Jeffers, James Reinders, and Avinash Sodani. *Intel Xeon Phi Processor High Performance Programming.* Morgan Kaufmann, June 2016. ISBN: 978-0-12-809194-4.

[94] R. Farber. *Parallel Programming with OpenACC.* Elsevier Science, 2016.

[95] Leonardo Dagum and Ramesh Menon. Openmp: an industry standard api for shared-memory programming. *Computational Science & Engineering, IEEE*, 5(1):46–55, 1998.

[96] Feb 2020.

[97] Enrico Calore, Alessandro Gabbana, Sebastiano Fabio Schifano, and Raffaele Tripiccione. Evaluation of dvfs techniques on modern hpc processors and accelerators for energy-aware applications. *Concurrency and Computation: Practice and Experience*, 29(12):1–19, 2017.

[98] V.M. Weaver, M. Johnson, K. Kasichayanula, J. Ralph, P. Luszczek, D. Terpstra, and S. Moore. Measuring energy and power with PAPI. In *Parallel Processing Workshops (ICPPW), 2012 41st International Conference on*.

[99] D. Hackenberg, R. Schone, T. Ilsche, D. Molka, J. Schuchart, and R. Geyer. An energy efficiency feature survey of the intel haswell processor. In *Parallel and Distributed Processing Symposium Workshop (IPDPSW), 2015 IEEE International*, pages 896–904, May 2015.

[100] Spencer Desrochers, Chad Paradis, and Vincent M. Weaver. A validation of dram rapl power measurements. In *Proceedings of the Second International Symposium on Memory Systems*, MEMSYS '16, pages 455–470, 2016.

[101] Sébastien Leclaire, Marcelo Reggio, and Jean-Yves Trépanier. Numerical evaluation of two recoloring operators for an immiscible two-phase flow lattice boltzmann model. *Applied Mathematical Modelling*, 36(5):2237–2252, 2012.

[102] Andrea Montessori, Adriano Tiribocchi, Marco Lauricella, Fabio Bonaccorso, and Sauro Succi. Mesoscale modelling of droplets' self-assembly in microfluidic channels. *Soft Matter*, 17:2374–2383, 2021.

[103] A. Tiribocchi, A. Montessori, F. Bonaccorso, M. Lauricella, and S. Succi. Shear dynamics of polydisperse double emulsions. *Physics of Fluids*, 33(4):047105, 2021.

[104] Andrea Montessori, Marco Lauricella, Elad Stolovicki, David A. Weitz, and Sauro Succi. Jetting to dripping transition: Critical aspect ratio in step emulsifiers. *Physics of Fluids*, 31(2):021703, 2019.

# List of publications

## Refereed journal articles

- Ivan Girotto, Sebastiano Fabio Schifano, Enrico Calore, Gianluca Di Staso, and Federico Toschi, "Computational performances and energy efficiency assessment for a lattice Boltzmann method on Intel KNL". Advances in Parallel Computing, 2019.

- Ivan Girotto, Sebastiano Fabio Schifano, Enrico Calore, Gianluca Di Staso, and Federico Toschi, "Performance and energy assessment of a lattice Boltzmann method based application on the Skylake processor. Computation, 8(2), 2020.

## Refereed multimedia content

- Ivan Girotto, Karun Datadien, Gianluca Di Staso, and Federico Toschi, "The chaotic life of mayonnaise". Gallery of Fluid, 72th Annual Meeting of the APS Division of Fluid Dynamics, November 2019

## In preparation

- Girotto et al., "Build-up of yield stress fluids via chaotic emulsification", currently in preparation.

- Girotto et al., "Droplets dispersion in dense emulsions via fully resolved simulations" in preparation.

- Girotto et al., "Droplets tracking in dense emulsions" submitted abstract to the 30th International Conference on Discrete Simulation of Fluid Dynamics

- Aliei et al., "An OpenACC based multicomponent Lattice-Boltzmann solver on GPUs"

## Non-refereed journal articles

- Ivan Girotto, Sebastiano Fabio Schifano, Enrico Calore, Gianluca Di Staso, and Federico Toschi, "Performance optimization of D3Q19 lattice Boltzmann kernels on Intel KNL". In INFOCOMP 2018: The Eighth International Conference on Advanced Communications and Computation, pages 31-36, 2018.

## Presentations at international conferences

- Ivan Girotto, Roberto Benzi, Karun Datadien, Gianluca Di staso, Prasad Perlekar, Jean-Paul van Woensel, Federico Toschi, "Dynamics of droplets in dense emulsions", 73rd Annual Meeting of the APS Division of Fluid Dynamics, Volume 65, Number 13, November 2020

- I. Girotto, K. Datadien, S. Aliei, G. Di Staso, S. F. Schifano, R. Benzi & F. Toschi, "The Chaotic Life of Mayonnaise", EuroHPC Summit Week 2021, Scientific and Industrial Conference (PRACEdays21), March 2021

- Ivan Girotto, Sebastiano Fabio Schifano and Federico Toschi, "Optimization of D3Q19 Lattice Boltzmann Kernels for Recent Multi- and Many-cores Intel Based Systems", Intel eXtreme Performance Users Group (IXPUG), CINECA, Bologna, March 2018

- Ivan Girotto, "The making of a turbulent emulsions", Physics@Veldhoven, Jan 2020, Veldhoven

## Master thesis from this work

- Michele Scotto di Petra, "Towards population balance modeling of drop breakup and coalescence in emulsions", M.Sc. in Chemical Engineering, University of Naples Federico II.

- Jean-Paul van Woensel, "A study on the properties of droplets in dense emulsion simulations", M.Sc. in Applied Physics, Eindhoven University of Technology.

- Saeid Aliei, "Porting of the LBE3D to GPU with OpenACC", Master in High Performance Computing, SISSA/ICTP joint programm.

# Summary

## Physics of Dense Emulsions via High-Performance Fully Resolved Simulations

Emulsions are important ingredients of industrial processes for many foods and cosmetics and interesting physical systems as well. In particular, stabilised concentrated emulsions can display remarkable rheological properties, ranging from complex and non-Newtonian flow dynamics, above the yield stress (when these complex fluids flow), to the presence of localised topological rearrangement (such as plastic events) below the yield stress, typical of soft-glassy materials. In addition to the complexity to describe these physical analytically, dense emulsions are also extremely challenging to be studied experimentally, due to the high concentration that prevents to accurately follow the dynamics of all fluid components, making it difficult to uncover their dynamics and thus their rich physical phenomenology.

In the pre-Exascale computing era, numerical modelling achieved a relevant and complementary role to experiments for studying the dynamics of emulsions, yet still requiring a significant amount of computational resources. The Lattice Boltzmann model (LBM) is widely used in computational fluid-dynamics to describe the behavior of fluid flows, and is commonly applied in several scientific and engineering fields of research in order to accurately model single and multiphase flows also in presence of complex boundary conditions. In this thesis, we re-engineered the numerical implementation of the LBM code used in previous works, developing a highly-optimized multicomponent LBM featuring an implementation of large-scale chaotic forcing, surface tension and disjoining pressure, to explore the physics of emulsions beyond what is currently experimentally possible.

In the first part of this thesis, we introduce the foundations and the most relevant scientific challenges relative to the modeling of multi-component turbulent emulsification processes and the implemented numerical code to accurately simulate the physics of dense emulsions. In Chapter 4 we describe in detail the process of emulsification via high-resolution computer simulations: from the initial formation of the fluid mixture, to the stirred stationary chaotic flow and to the final phase when the external forces are turned off and a stable

emulsion is achieved. We present qualitative and quantitative results on the morphology of the emulsions from dilute to highly concentrated, obtained at different resolutions, and with different magnitude of the hydrodynamic stirring. This chapter also includes qualitative results on the phenomena of phase inversion that can occur during the emulsification process.
From dilute to highly concentrated, emulsions are characterised by a coupling between small-scale droplets and large-scale rheology critically determining the statistical properties of the resulting fluid mixture. In the second part of this thesis, we introduce the innovative tracking algorithm used to follow the trajectories of droplets within a chaotic, possibly turbulent, three-dimensional multi-component emulsion flow. Thanks to these tools we obtain a unique database of droplets' Lagrangian trajectories during the full emulsification process, even in very dense emulsions. In Chapter 5 we present statistic results, on a large number of samples, on both the macroscopic hydrodynamic quantities as well as the small-scale morphology and the individual dynamics of single droplets, including an accurate tracking of breakup and coalescence events.
In Chapter 6 we present and discuss in detail the code optimisation steps that we performed in order to enable the computational tools aimed to study the physics of dense emulsions on EU Tier-0 system for HPC in the pre-Exascale era, including a detailed analysis of performance and energy efficiency. We describe the effort of the OpenACC based porting of the multicomponent LBM code on EU Tier-0 multi-GPUs distributed systems, such as Marconi-100, hosted at CINECA, showing performance portability on hybrid CPU/GPU architectures. We conclude presenting the conclusion of this work and an outlook for future research.

# Acknowledgement

This thesis work is fully dedicated to the people that have been negatively affected by me having undertaken this challenge in an advanced stage of life when it gets more complicated and decisions impact others too. It is dedicated to Carla, my lovemate of many years, for remaining at my side despite all the difficulties, to my parents, and to my grandmas, Ines Lissandrin and Laura Pizzirani, who both passed away in the last year, not leaving me the chance to properly thank them for having grown me safely and extraordinarily loved.
This work is the result of effort by many people, and I want to take the occasion to thank them all. First, I would like to thank Prof. Federico Toschi and Prof. Fabio S. Schifano for having led and co-supervised the research project, providing effective advice and valuable insight throughout the PhD program. Second, I would like to thank the scientific advisors of this work: Prof. Roberto Benzi, Dr. Andrea Scagliarini, Dr. Prasad Perlekar, Dr. Enrico Calore and Dr. Gianluca Di Staso, for the useful discussions along with their contribution to the manuscripts. Third, I would like to thank everybody that practically contributed to this thesis: Saeid Aliei, for his contribution to the GPU porting of the multicomponent LBM code; Jean-Paul van Woensel and Michele Scotto di Perta, for their contribution in the analysis of the Lagrangian trajectories of the droplets; Karun Datadien, for his contribution to set the initial parameters and for the high-quality graphics (also included in the cover of this thesis); Dr. Muhammad Nawaz Qaisrani, for his contribution to setting up the calculation for the MD simulation; and Dr. Alessandro Corbetta, for his initial contribution to the tracking algorithm.
Finally, I would like to thank the institutions that have made all this possible. First, the Abdus Salam International Centre for Theoretical Physics (ICTP), which I have been serving for the past 9 years, and specifically Prof. Fernando Quevedo, former ICTP Director, Prof. Sandro Scandolo, ICTP research director, and Uli Singe, ICTP senior operations, finance and IT officer, for having offered me the opportunity to do this PhD while employed at the centre, as well as for the funding. Second, I would like to thank the Eindhoven University of Technology and the University of Modena and Reggio Emilia for having hosted my PhD program. To conclude, I would like to thank PRACE and its hosting centres, CINECA and the Barcelona Supercomputing Centre (BSC), for having

provided the computer resources that formed the foundation of this research project.

# Curriculum Vitae

Ivan Girotto was born on 25-12-1980 in Ferrara, Italy. In 2006, he obtained his Master in Computer Science at University of Ferrara, Italy. From 2016 he started a joint PhD project between the Technological University of Eindhoven, the Netherlands, and the University of Modena and Reggio Emilia, Italy, of which the results are presented in this dissertation. Since 2012 he is a professional civil servant of the United Nations Educational, Scientific and Cultural Organization (UNESCO), serving the Organization as specialist for high-performance computing (HPC) at the Abdus Salam, International Center for Theoretical Physics (ICTP). After his M.Sc., he dedicated its professional career to the field of high-end scientific computing, working at International centers for HPC since earlier than its graduation. From 2005 to 2009 he has been member of the supercomputing group at CINECA, Italy, and from 2009 to April 2012 working as computational scientists at the at the ICHEC, Ireland, with the main goal of enabling large scale scientific simulations for both academic and public research as well as industry (i.e. ENI, BMW Oracle Racing, Tullow Oil, Paddy Power, etc...). He is pioneer of GPU computing applied in science with its contribution to the porting of the Quantum ESPRESSO package on hybrid GPU distributed platforms. Today at ICTP, Ivan Girotto is co-director and lecturer in number of international workshops and advanced schools for higher education as well as other programs for training young scientists on parallel programming for scientific computing and HPC (i.e., the joint SISSA/ICTP Master program in HPC, the master in Data Science and Scientific Computing at the University of Trieste). His main activity at the ICTP remains to conduit and support the world-wide community to efficient research production on HPC facilities. Since 2015, Ivan Girotto is also among the Principal Investigators of the European Centre of Excellence MaX - Materials design at the Exascale.