CRANFIELD UNIVERSITY

ANTONIO D'AMMARO

COMPUTATIONAL DESIGN FOR MICRO FLUIDIC DEVICES USING LATTICE BOLTZMANN AND HEURISTIC OPTIMISATION ALGORITHMS

SCHOOL OF ENGINEERING
MSc Thermal Power

MSc THESIS

Academic Year: 2009 - 10

Supervisor: A. M. Savill

Co-Supervisor: T. Kipouros

September 2010

CRANFIELD UNIVERSITY

SCHOOL OF ENGINEERING
MSc Thermal Power

MSc THESIS

Academic Year 2009 - 2010

ANTONIO D'AMMARO

Computational Design for Micro Fluidic Devices Using Lattice Boltzmann and Heuristic Optimisation Algorithms

Supervisor: A. M. Savill

Co-Supervisor: T. Kipouros

September 2010

This thesis is submitted in partial fulfilment of the requirements for the degree of Master of Science

ABSTRACT

The study on micro devices is gaining importance in various fields from biological to engineering. The dimensions of these devices range from millimetres (*mm*) to micrometres (*μm*) and they work within the laminar flow regime due to their low Reynolds number. Although diffusivity dictates the mixing in such conditions, this work is based on the simulation of two non reacting iso-thermal and incompressible fluids for both streams, so the mixing is governed only by turbulence. A numerical study using the Lattice Boltzmann method (LBM) is carried out in order to examine the mixing in one configuration.

In the second part of the work an interface is developed between the LBM code and multi objective optimisation software in order to investigate new configurations which enhance the mixing. The objectives are to maximise the vorticity and minimise the pressure drop, which are conflicting between themselves. The tool that integrates the optimiser and the LBM code for simulating microreactor, can run on a multi level parallelisation, hence the time required for the whole simulation has been drastically reduced.

A preliminary optimisation is performed on a microreactor with multi holed baffle plate. Despite small number of iterations, three totally different configurations have been found: greatest vorticity, smallest pressure drop and a compromise. The first two configurations satisfy the expectations, whereas the compromise solution presents an innovative configuration. In fact, it has a low pressure drop and a high vorticity, which is achievable by having high Reynolds number and big diameters of the holes. Hence, this tool proves to be robust and efficient in the multi objective optimisation of microreactor

Keywords: Mixing, Fluid dynamics, Lattice Boltzmann Method, Multi Objective Tabu Search

ACKNOWLEDGEMENTS

I would like to thank my Supervisor Prof. Dr. Mark Savill for the possibility of developing this thesis. I would like to thank you my Advisor Dr. Timos Kipouros, who gave me plenty of suggestions and has always been capable to give good advices. Their support has been fundamental for the development of this thesis. I am grateful to the PhD Salwan David Saddawi, who carried out the CFX simulations and Prof. Lyazid Djenidi, who gave me the LBM code.

I appreciated and I give thanks to Dr. Les Oswald and Dr. Jonathan Haynes of the IT department in Cranfield. Their technical support played a key role in the development of the codes.

I am thankful to Politecnico di Bari, in particular Dr. Benedettini, for the opportunity they gave me to come to England and accomplish one year of double degree, it has been a wonderful experience.

I feel grateful to the www.lbmethod.org forum, for the precious advices during the development of my thesis, in particular the users Timm and Erlend_M.

I am grateful to my parents and brothers, who supported me, before in the studies in Italy and then for this year in England. Their precious suggestions, their constant and helpful presence, encouraged me to carry on, thanks for being always a reference point through - out my life.

Special thanks goes to Cristinica, who helped me so much during the thesis and more in my life in Cranfield. Her smile and her attentions, spurred me to work and relieved the pressure on my work.

Finally, a big thank you to all the persons I met here in Cranfield and made this year extraordinary. In particular, “*La Famiglia*”, this little planet we have built never made me feel alone and always surrounded by helpful friends, thank you so much.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATIONS

1 CHAPTER

Introduction

1.1 Overview and objectives

The technological progress in machining and the need of miniaturized device spurred on the research in the “micro” world field for biological or engineering applications. Microreactors play an important role in this progress; in particular they have applications in the combustion, chemical analysis or onboard applications. The smaller dimensions of these devices imply a low Reynolds number flow that is within laminar regime. The mixing, in such conditions, is mainly governed by diffusion of the species which curbs the speed of mixing. Despite its minor role, turbulence can increase the mixing, which can also be enhanced by using active or passive devices, such as moving parts or multi – holed baffle plate [1].

The passive mixers do not require energy input but need a more complex geometry because geometry enhances the mixing by increasing the turbulence. The mixer analysed in this work has been studied experimentally by Moghtaderi [2] and then analytically by Djenidi [3], which both showed that a multi – holed baffle plate can be a good option to increase the interfacial area between two fluids. The flows going out from the holes create some recirculation zones which promote the mixing.

This thesis presents a numerical study of the mixing within a micromixer device using the Lattice Boltzmann method (LBM). The code was tested by Djenidi [3] and it has been used in this work because its stability has been proved already. The code has been successfully validated with CFX and with [2], which work under similar conditions.

The geometrical and physical parameters of the micromixer as stated in [2] have been changed only for some different configurations, in order to study the behaviour of the device in diverse conditions. In fact, the work is at first stage,

and it was appropriate to investigate if new configurations could be able to give better mixing. In order to do that, the MOTS software developed by Jaeggi [4] was used. It is a multi objective optimiser which is based on local search method. Its robustness and reliability as optimiser have already been proved by Kipouros [5]. Thus, three design variables and two objectives were designed for the optimiser. The choice of the objective functions: vorticity and pressure drop, is suggested by the opposite behaviour of this two parameters and in fact a good mixing generally implies a big pressure drop. In order to find an optimum in this challenging investigation, the micromixer was simulated in different conditions to find a good compromise.

1.2 Structure of the thesis

The tasks presented above have been accomplished; the methodology and the results are presented in the following chapters. The second chapter addresses the microfluidic mixing and presents different types of micromixers. The chapter three and four are focused on the Lattice Gas Automata and the Lattice Boltzmann method respectively. The chapter five presents the code used for simulating the microreactor, explaining in detail the different subroutines. In addition it also explains how the code has been set in terms of parameters and evaluation of the objective functions; at the end is presented the validation of the code. The chapter six gives the theoretical background of the Multi Objective optimisation and a describes the software used. The chapter seven presents how the MOTS (Multi Objective Tabu Search) software has been set and how the optimisation simulation works on the high performance computer sited in Cranfield. The chapter 8 provides the results of the simulation and the three cases analysed: higher vorticity, lower pressure drop and a compromise solution. Conclusions and recommendations are provided in last chapter.

2 CHAPTER

Case Description

This chapter provides a description of the mixing characteristics in microfluidic devices. Later, some geometrical and physical values of the experiment test case, used to set up and validate the simulation are provided.

2.1 Mixing in microfluidics

The term microfluidics identifies all the devices whose dimensions are ranging from millimetres (*mm*) to micrometres (*μm*) and volumes of fluid which range from nanolitres (*nl*) to microliters (*μl*) [1]. The progress in the machining processes and the need of small devices, e.g. for biomedical processes or MEMS, resulted in an increase of demand and interest in this field [6]. The mixing might be an important process for the functioning of microfluidic devices, so that often micromixers are integrated in the systems.

The mixing between two components is based on two phenomena [1]:

- *molecular diffusion;*
- *bulk flow mixing due to turbulence*.

The former is obtained by reducing the length – scales, in this case the free mean path of the molecule is comparable with the characteristic length, hence two streams of different components mix. The latter is dominated by kinetic aspect and it involves the interspersion between two streams.

The mixers are divided into two main categories: active and passive mixers. The former requires an energy input in order to mix the flows, e.g. electro-hydrodynamic mixer, deeply treated in [7]. The latter, does not require a source of energy or moving parts, are easier to integrate and more stable in operation.

The mixing in microfluidic device mainly relies on the diffusion rate since the Reynolds number are low, which implies that the flow is mostly laminar and

there is not enough turbulence. However, if the two flows have the same composition or they present a very low coefficient of diffusion, the process is independent of diffusivity (e.g. [2] and this work). In these cases, despite the low velocities, it is possible to intensify the mixing by increasing the interfacial area between the two flows and/or a decrease of the mixing path [7]. This is achievable by adopting a series of schemes referred as streaming mixing (SM). They are well explained in Ehrfeld [8] and comprise, for instance, contact between two sub-streams in a T- or Y- shaped channel. However, as reported in Ottino [9]: “mixing in continuous 3D duct flows depends critically on cross – sectional area". In fact, the presence of components in the direction of the flow that stretches and folds the fluid over the cross section of the channel enhances the turbulence, e.g. a baffle plate.

2.2 Experimental case

The test case adopted for the validation of the code is described in Moghtaderi [2]. The latter presents a series of experiments, carried out on a microreactor model. It comprises a T junction where a multi holed baffle plate is located co-axially with another tube as represented in the Figure 2-1. The research of Moghtaderi is focused on the development of a miniaturized microcombustor. Hence the inner tube is adopted for the fuel flow, whereas through the baffle plate the oxidizer is introduced. Figure 2-2 reports an illustration of the model and its dimensions.

The experiments were conducted using the micro – PIV technique on a 1:1 replica of the actual reactor. In order to have a similarity between the experiments and the real reactor, the Reynolds numbers had to be equal (*Re=100*) in the two cases. It is calculated based on the outer diameter (*D=300 μm*) and the mean velocity over the full cross-sectional area, about *0.3 m/s*. In order to satisfy this constraint, the flow rate in the inner tube is set to *5%* of the flow in the outer tube, as in the real case. Hence, the equality of the ratio between density and viscosity for the experiment and real case, assures the equality of the Reynolds numbers:

$$(\rho/\mu)_{exp} = (\rho/\mu)_{real}$$

Since water is used as replica for the methane – oxygen mixture, by adding glycerine to water the above constraint has been satisfied [2], and the Reynolds numbers are equal.

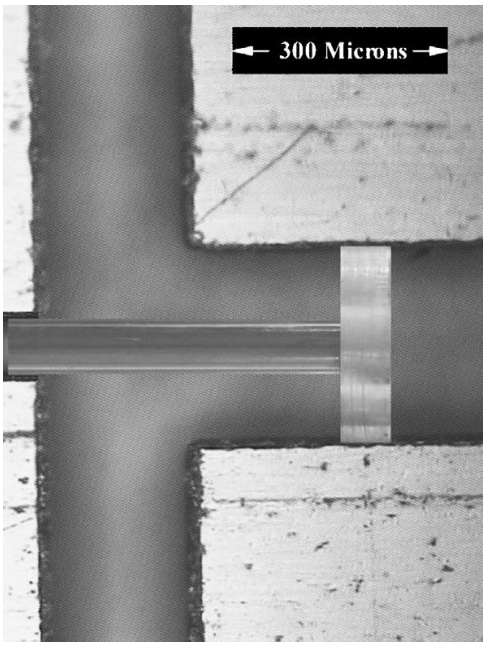


Figure 2-1 Microscopic top view of the baffle plate/inlet tube assembly [2]

The experiments have been made for two different values of Reynolds number (100 and 300) and different geometric configurations. However, this work will concentrate on the set – up described above and reported in Figure 2-2. The results of the experiments are reported in the section 5.19.

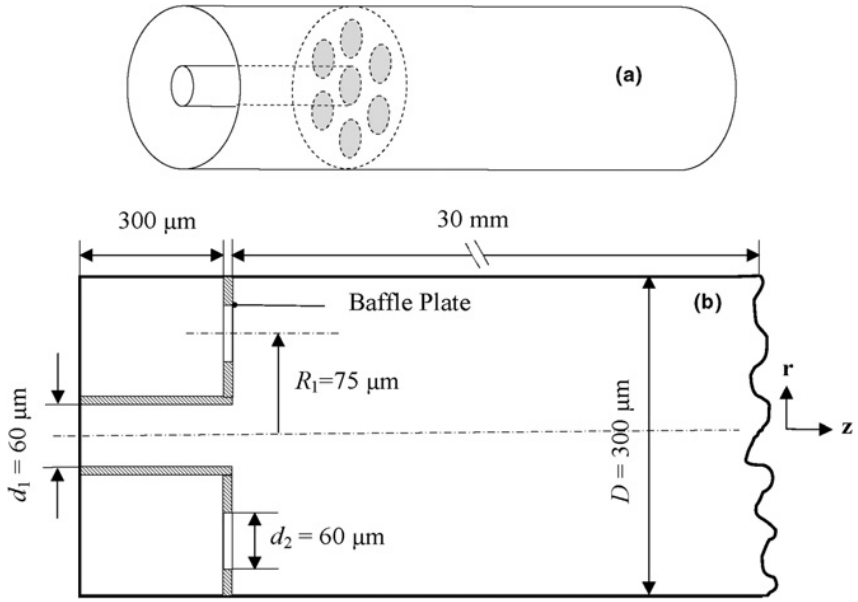


Figure 2-2 Schematic illustration (not to scale) of the microreactor model: (a) perspective view and (b) side view [2]

3 CHAPTER

Lattice Gas Automata

The first part of this chapter describes the microscopic behaviour of kinetic theory. The second part provides a description of Lattice Gas Automata (LGA) which is the predecessors of Lattice Boltzmann Method (LBM), in order to easily understand the LBM. In addition to the general equations which govern this method, a practical approach that describes the simplest LGA models: the HPP and FHP models are briefed.

3.1 The kinetic theory of gases

The kinetic theory is the branch of statistical physics which studies the microscopic behaviour of molecules, their interactions based upon the Newton's laws and their relaxation to thermodynamic equilibrium. The Boltzmann equation represents its cornerstone [10] and it was the first successful theory, capable to explain the macroscopic irreversibility through a statistical description of a system composed of a large number of particles.
Let us considering a small macroscopic volume of gas, e.g. $V=1\ cm^3$ which, according to the molecular theory of matter, contains about $N=10^{20}$ particles, each moving in an irregular way and colliding between themselves [11]. Their dynamics is governed by the Newton's equation and assuming the molecules as mass points, the equations which describe the motion are:

$$\begin{aligned} \dot{\boldsymbol{p}}_{\boldsymbol{i}} &= \boldsymbol{F}_{\boldsymbol{i}} \\ \dot{\boldsymbol{x}}_{\boldsymbol{i}} &= \frac{\boldsymbol{p}_{\boldsymbol{i}}}{m} \end{aligned} \qquad (3\text{-}1)$$

where $\boldsymbol{x_i}$, $\boldsymbol{p_i}$ are respectively the position vector and first linear momentum ($\boldsymbol{p_i}=m\boldsymbol{v_i}$) of the i-th particle (i=1,…,N) both functions of the time t; the dots denote the differentiation with respect to t. $\boldsymbol{F_i}$ denotes the force acting upon the

i-th particle divided by its mass as a result of the intermolecular interaction and external fields, e.g. an external force is the gravity.

The system of 6N first order differential equations, Eq. **(3-1)** in the 6N unknowns which are the components of the vectors: [$\boldsymbol{x_i(t),\ p_i(t)}$] (*i*=1,…,N) can be solved in time. The resolution of the system requires the knowledge of the *6*N initial conditions, which describe the initial state of the system:

$$\boldsymbol{x_i}(\mathbf{0}) = \boldsymbol{x_{0,i}} \quad ; \quad \dot{\boldsymbol{x}}_{\boldsymbol{i}}(\mathbf{0}) = \frac{\boldsymbol{p_i}(\mathbf{0})}{m} = \boldsymbol{v_{0,i}} \tag{3-2}$$

The solution of the Cauchy problem is unique and "yields a fully exhaustive knowledge of the state of the system". However, "this program is totally unviable and, fortunately, needless as well" [10] and the main reasons are:

1. The 6N initial conditions**:** [$\boldsymbol{x_{0,i},\ v_{0,i}}$] (i=1,…,N) are difficult to know at the same instant of time; imagine measuring simultaneously at *t=0* these values for 10^{20} particles.

2. Assuming that all the measures have been taken, they cannot be infinitely accurate; there will always be a truncation error. Assuming that the truncations errors are of the order of 10^{-16} (double precision number) which for a system of 10^{20} particles imply a huge quantity of data:

$$6\left[\frac{numbers}{molecule}\right] \cdot 10^{20}[molecules] = 6 \cdot 10^{20}[byte]$$

This quantity of data, for being read by the high performance computer presents in Cranfield University, which is 7.3 Teraflops, would require:

$$6 \cdot 10^{20}[byte] \cdot \frac{1}{7.3 \cdot 10^{12}}\left[\frac{sec}{byte}\right] = 2.61\ years$$

3. Even if we can measure all this information, with infinite precision it is almost impossible to solve them in a computer, for sensible time. The enormous number of particles: *N*, of the order of 10^{20}, is too big for handling in any existent computer.

Even if all the difficulties above mentioned and reported in [10; 11] are overcome, the information about the position and velocities of the particles does provide any information about the pressure, temperature and density, which are the properties more important, from an engineering point of view. “It appears wise therefore to approach the collective behaviour of our ensemble of molecules from a statistical point of view“ [10]. “Statistical Mechanics offers a statistical approach in which we represent a system by an ensemble of many copies” [12]. It means that, the work is not based upon certainties but upon probabilities. The particle will be defined not by its velocity and its position, but different probabilities of having the different position and velocities. This work will not go further on the description of the statistical mechanics, as further reading is suggested [11; 13].

3.2 Continuum approach and Kinetic theory

The fluid motion can be described by the Navier – Stokes equations and the kinetic theory. In the former, the fluid is viewed as continuous matter, and at each point of the volume it exists a value of pressure, density, velocity and temperature. The continuous fluid must obey the conservation laws of mass, conservation of the species, momentum and energy, which give rise to a set of partial differential equations. The solution of this set, gives a description in space and in time of the evolutions of the above mentioned quantities. The kinetic theory, as described in section 0, considers the fluid from the microscopic point of view, hence composed by a finite number of molecules, whose motion is governed by the dynamic’s laws. “It attempts to derive the macroscopic parameters and the behaviour of the fluid from the laws of mechanics and probability theory” [14]. In Appendix A.1 is reported the derivation of the Navier – Stokes equation of continuity from the LBM dynamics.

The hypothesis of continuum is a pillar for the Navier – Stokes equations, and when it is not verified, those equations cannot be applied to know the fluid behaviour. This depends on the Knudsen number (*Kn*), Figure 3-1. Call *L* the

characteristic macroscopic length (e.g. the diameter for a pipe) and l_{mfp} the mean free path of a molecule:

$$Kn = \frac{l_{mfp}}{L}$$

If *Kn* is higher than one, the resolution of the Navier – Stokes set of equations would not give a physical result. Whereas, for high values of this non-dimensional number the possibility of collisions between two molecules of the gas is negligible and is too dilute, such as in rarefied gases.

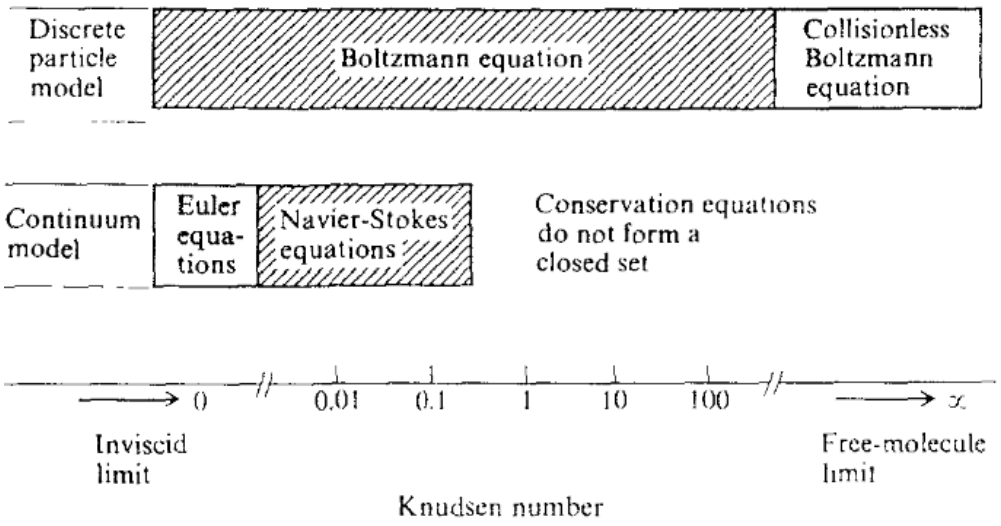


Figure 3-1 Regimes of applicability of various flow equations for low - density flows [15]

3.3 Cellular automata

The behaviour of a flow can be described statistically with the kinetic theory but before presenting the LBM, its harbingers, the Lattice cellular automaton models are described. A cellular automaton is an entity that occupies a position on a grid or a lattice, its behaviour is influenced by its and some neighbour states, and change in time.

3.3.1 HPP model

In 1973 Hardy, Pomeau and the Pazzis proposed the HPP model (from their surnames), for describing two – dimensional flows [14]. The model consisted in a square lattice, with lattice unit constant. The cellular automaton is a discrete particle, moving in discrete time and in the discrete space defined by the lattice. Each of them has unitary mass., four neighbours and four unitary possible velocities: $\boldsymbol{c_1}=(1,0)$, $\boldsymbol{c_2}=(0,1)$, $\boldsymbol{c_3}=(-1,0)$, $\boldsymbol{c_4}=(0,-1)$ as reported in Figure 3-2.

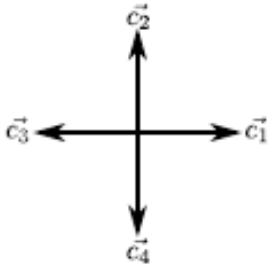


Figure 3-2 The velocity vectors of the HPP model [16]

At each time step, the particle moves forward in the direction it is pointing, as shown in Figure 3-3.

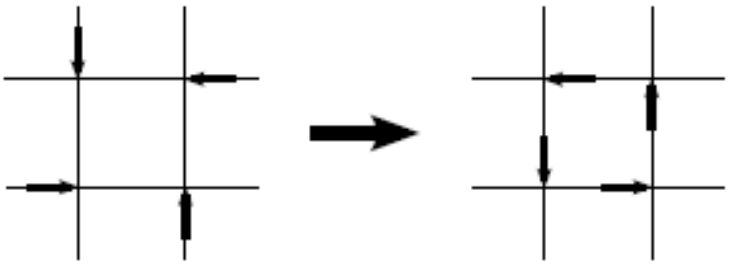


Figure 3-3 Four different particles moving along a HPP model after one time step [16]

It may happen that two different particles arrive in the same node, in that case, the collision occurs. A key concept throughout all the lattice models is that, in order to conserve the mass and momentum of the system, the number of particles (mass continuity) and the total velocity (momentum continuity) of all the particles in the node before the collision must be equal after the collision. An

head on collision is represented in Figure 3-4. When it happens the velocities of the two particles rotate of *π/2* (or *-π/2*). After the collisions, the particles move to the following node.

This model is deterministic, since the evolution is reversible. This is because there is only one type of collision, and there is only one possible result. This method presents numerous drawbacks; the main is the failure in achieving the rotational invariance, which means that its behaviours is anisotropic. This is not in accordance with the Navier –Stokes equations [16].

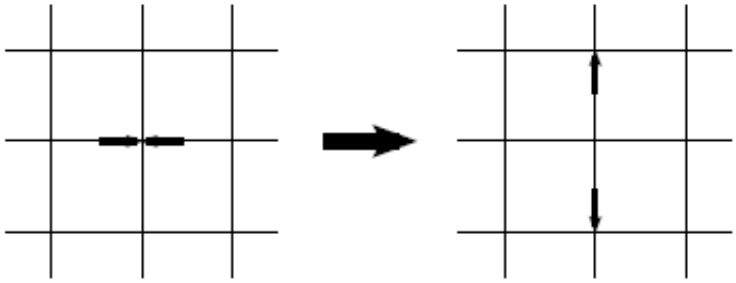


Figure 3-4 Head - on collision rule for the HPP model [16]

3.3.2 FHP model

The FHP model, overcomes the anisotropy by changing the shape of the lattice. It was introduced by Frisch, Hassalacher and Pomeau in 1986 [16]. The lattice is hexagonal, as shown in Figure 3-5. The velocity and the lattice units are no longer unitary, but: $\boldsymbol{c_1}=(1,0)$, $\boldsymbol{c_2}=(1/2,\sqrt{3}/2)$, $\boldsymbol{c_3}=(-1/2,\sqrt{3}/2)$, $\boldsymbol{c_4}=(-1,0)$, $\boldsymbol{c_5}=(-1/2,-\sqrt{3}/2)$, $\boldsymbol{c_6}=(1/2,-\sqrt{3}/2)$.

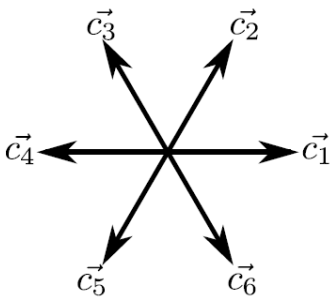


Figure 3-5 The velocity vectors of the FHP model [16]

As well as the HPP, the FHP model obeys to the conservation laws. This time the neighbours per node are six, and the lattice still lays on a two dimensional space. The collision rules are the same as before, but the result of the collisions is not unambiguous, hence the model is not deterministic. In Figure 3-6 are reported the collisions rules:

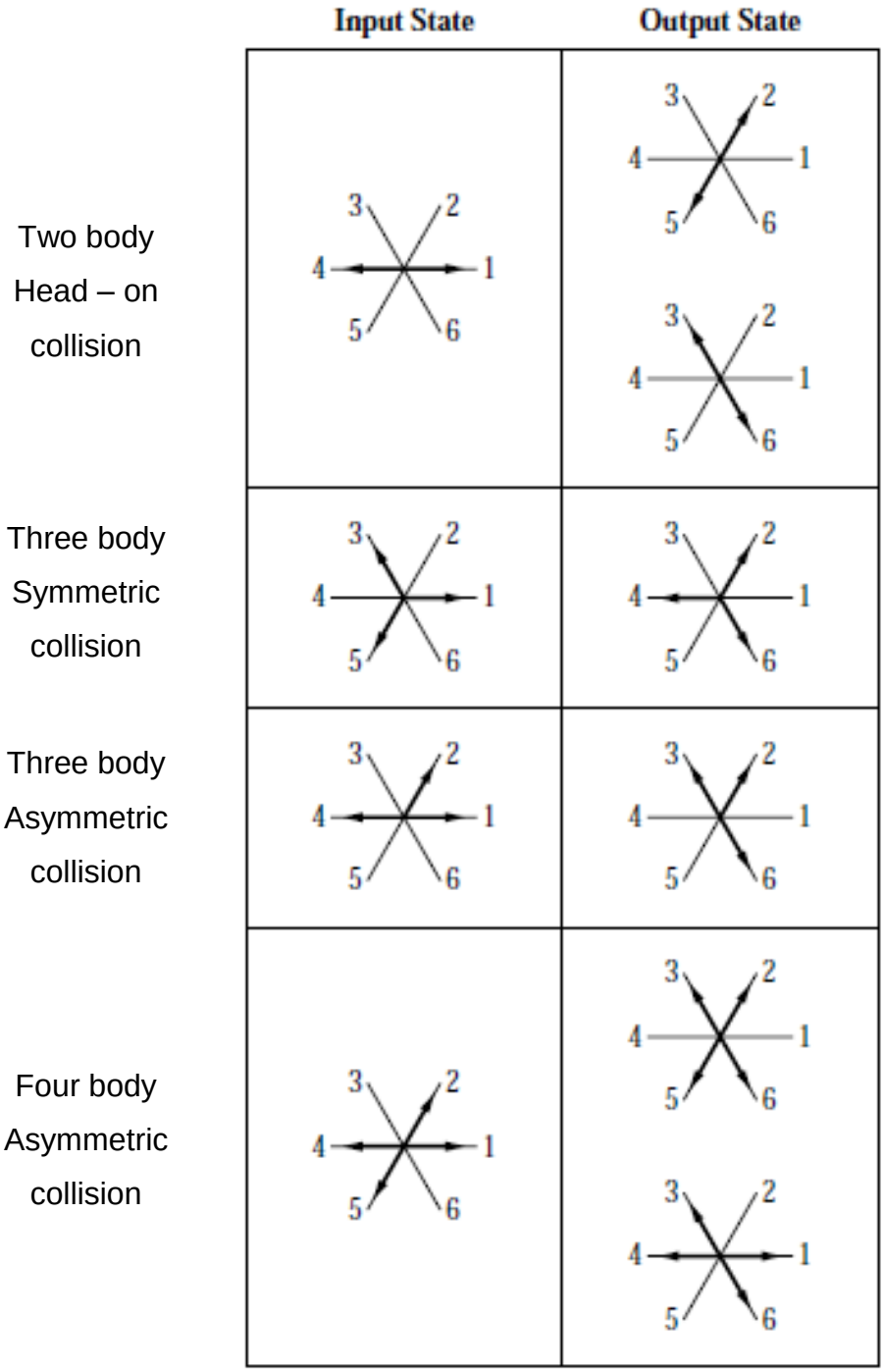


Figure 3-6 The collisions rules for the FHP [14]

The FHP presents the same advantages of the HPP, such as small storage space demands, highly parallelisation and exact dynamics. On the contrary, it does not present the anisotropy, Achilles' heel of the HPP, and it is consistent with the incompressible Navier – Stokes equations [16].

The FHP model can be easily implemented, using Boolean number, according to the presence or not of a particle at site $\boldsymbol{x}$, at time t and velocity $\boldsymbol{c_i}$ (i=1,..,6). Using a 6 digit binary number, it is possible to describe the configuration in a node, and the collisions rules represented in Figure 3-6. For instance, a configuration with one particle with momenta: $\boldsymbol{c_1}$, $\boldsymbol{c_4}$ can be represented with the binary number: 001001. The collisions rules are reported in the table:

Table 3-1 The collision table for the 6 bit FHP model [12]

Input state	Output state
001001	010010 100100
001001	101010
001001	100110
001001	110110 101101

To describe the particle occupation at each lattice node ($\boldsymbol{x}$) at time t, a set of Boolean variables $n_\alpha(\boldsymbol{x},t)$, $(\alpha=1,\ldots,M)$ is defined, where M is the number of directions of the particle velocities at each node (6 in the 6 bit FHP model).

The FHP lattice gas automata, has been consequently built on a triangular lattice space, with unitary lattice constant and discrete time. The equation which describes the system is:

$$n_\alpha(\boldsymbol{x}+\boldsymbol{c_\alpha}\cdot\delta_t, t+\delta_t) - n_\alpha(\boldsymbol{x},t) = \Omega_\alpha[n(\boldsymbol{x},t)] \qquad (3\text{-}3)$$

Where Ω_α represents the collision operator, δ_t is the time step, usually set to unity as δ_x the lattice constant. The collision operator represents the creation or annihilation of $n_\alpha(\boldsymbol{x},t)$ due to collisions, which must respect the conservation of momentum and mass at each collision. A collision – less model has the right - hand side of Eq. (3-3) equal to zero.

The entire process in one time step for an LGA triangular lattice space is represented in Figure 3-7.

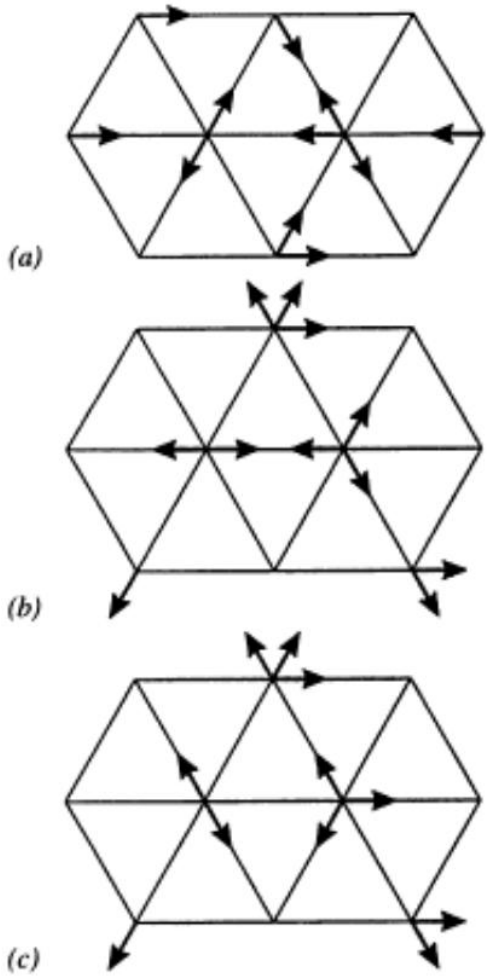


Figure 3-7 One time step in the evolution of a two-dimensional lattice gas. Each arrow represents a particle of unit mass moving in accordance with the arrow direction. (a) Initial condition, (b) Propagation to one lattice unit in the velocity direction, (c) Collision process [6]

Considering the system represented in Figure 3-7, (a) the system at the initial condition. The first step of the LGA is the streaming (b), when the particle moves to the nearest neighbourhood node along its velocity direction. The second step is the collision (c), the particles collide according to the laws before mentioned and change their direction.

The LGA presented in this chapter, present several defects. The most important is the noise of the simulations, because of the large fluctuations of n_α. In order to limit the noise, the LBE have been introduced in 1988 by Mcnamara and Zanetti [6]. The difference between these two models is that LBE do not use the Boolean particle occupation variables in Eq. (3-3). It is replaced by a single – particle distribution function f_α, which is the ensemble average of n_α, Eq. (3-4), and its values is between 0 and 1.

$$f_\alpha = \langle \mathrm{n}_\alpha \rangle \tag{3-4}$$

The mathematics behind this change is beyond of the scope of this work, and its demonstration and explanation can be found in [6]. Combining Eq. (3-3) - (3-4) it is possible to write:

$$f_\alpha(\boldsymbol{x} + \boldsymbol{c_\alpha} \cdot \delta_t, t + \delta_t) - f_\alpha(\boldsymbol{x}, t) = \Omega_\alpha[n(\boldsymbol{x}, t)] \tag{3-5}$$

This equation, obviously, presents all the simplifications made in deriving Eq. (3-3). Thus, in the next chapter will be presented how the Lattice Boltzmann method moved from microscopic scale, to mesoscopic scale, treating averages over many single particles.

4 CHAPTER

Lattice Boltzmann Method

This chapter provides the basics for understanding the Lattice Boltzmann method, its derivation, its weaknesses and strong points. The first part briefs the first order distribution function. The second part explains the important steps to take into account while building a LBM code.

4.1 First Order Distribution Function

Supposing now that it is possible to know the distribution function of the system presented in the paragraph 0, which means that the momentum (***p***) and the positions (***x***) of each particle of the volume are known, it would be described by $f^{(N)}(\boldsymbol{x},\boldsymbol{p},t)\mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{p}$. This gives the probability of finding *N* molecules within the range $\boldsymbol{x} \pm d\boldsymbol{x}$ and the momentum $\boldsymbol{p} \pm \mathrm{d}\boldsymbol{p}$. Using the statistical mechanics, it is possible to avoid such high resource demanding calculations.

Supposing to adopt a low order distribution function, e.g. *N=1*, called one-body kinetic level, “the distribution function: $f^{(1)}(\boldsymbol{x},\boldsymbol{p},t)$ gives the probability of finding a particular molecule with a given position and momentum; the positions and momenta of remaining *N*-1 molecules can remain unspecified because no experiment can distinguish between molecules, so the choice of which molecule does not matter” [12]. The reduction from $f^{(N)}$ to $f^{(1)}$ is adequate when the gas properties do not depend on the relative positions of the molecules which means long mean free path.

4.2 Streaming process

The function $f^{(1)}(\boldsymbol{x},\boldsymbol{p},t)\mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{p}$ gives the probability of finding one molecule at the instant *t* within the position $\boldsymbol{x} \pm d\boldsymbol{x}$ and the momentum $\boldsymbol{p} \pm \mathrm{d}\boldsymbol{p}$. Assume an external force ***F*** is acting in the system (e.g. the molecule is in a gravitational field), if collisions are neglected, the particle will stream in the direction it was

heading. Hence, the molecule that was at (***x***,***p***,*t*), in the next time step: *t*+d*t* will be:

t	$t + dt$
$\boldsymbol{x}$	$\boldsymbol{x} + (\boldsymbol{p}/m)\mathrm{d}t = \boldsymbol{x} + \boldsymbol{c}\mathrm{d}t = \boldsymbol{x} + (\mathrm{d}\boldsymbol{x}/\mathrm{d}t)\mathrm{d}t = \boldsymbol{x} + \mathrm{d}\boldsymbol{x}$
$\boldsymbol{p}$	$\boldsymbol{p} + \boldsymbol{F}\mathrm{d}t = \boldsymbol{p} + (\mathrm{d}\boldsymbol{p}/\mathrm{d}t)\mathrm{d}t = \boldsymbol{p} + \mathrm{d}\boldsymbol{p}$

$$f^{(1)}(\boldsymbol{x} + d\boldsymbol{x}, \boldsymbol{p} + d\boldsymbol{p}, t + dt)d\boldsymbol{x}d\boldsymbol{p} = f^{(1)}(\boldsymbol{x}, \boldsymbol{p}, t)d\boldsymbol{x}d\boldsymbol{p} \qquad (4\text{-}1)$$

Knowing the distribution at time *t* enables to know the particle distribution at time *t*+d*t*, applying Eq. (4-1).

4.3 **Collision process**

If the collisions are not neglected, the Eq. (4-1) needs to be changed. Boltzmann made stringent assumptions on the nature of the physical system by deriving his equation, which are dilute gas of point-like, structureless molecules interacting via a short range two body potential [10]. Hence, intermolecular interactions can be described assuming localized binary collisions and the molecules would spend most of the time on free trajectories. Some particles starting at (***x***,***p***,*t*) will not arrive the next time step at (***x***+d***x***, ***p***+d***p***,*t*+d*t)* because they have been collided out their path (loss). Whereas, others particles not starting from (***x***,***p***,*t*) will arrive at (***x***+d***x***, ***p***+d***p***,*t*+d*t)* because they have been collided into this path (gain) [16]. This situation is represented in Figure 4-1 where two molecules collide, the apex indicates the time step *t*+d*t.*

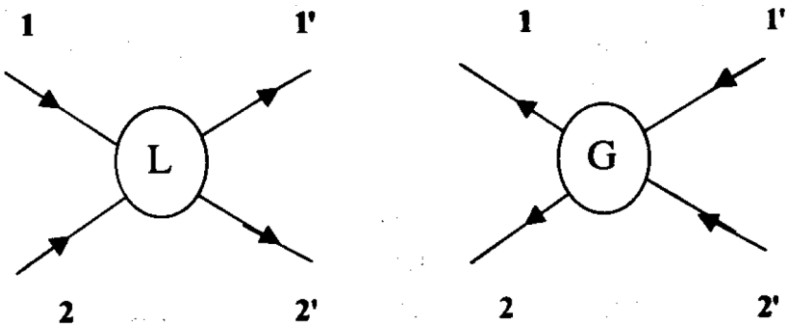


Figure 4-1 Symbolic diagram of molecules go out and into the path [10]

Call $\Gamma^{(-)}$d$\boldsymbol{x}$d$\boldsymbol{p}$dt equal to the number of molecules that do not arrive at the expected coordinates: ($\boldsymbol{x}$+d$\boldsymbol{x}$, $\boldsymbol{p}$+d$\boldsymbol{p}$,t+dt) due to collisions at time dt. Call $\Gamma^{(+)}$d$\boldsymbol{x}$d$\boldsymbol{p}$dt equal to the number of molecules not starting from ($\boldsymbol{x}$,$\boldsymbol{p}$) that arrive at the expected coordinates because of the collisions at time dt.

4.4 The Boltzmann equation

If the collisions terms are added to Eq. (4-1):

$$\begin{aligned} f^{(1)}(\boldsymbol{x}+d\boldsymbol{x},\boldsymbol{p}+d\boldsymbol{p},t+dt)d\boldsymbol{x}d\boldsymbol{p} \;&= \\ &= f^{(1)}(\boldsymbol{x},\boldsymbol{p},t)d\boldsymbol{x}d\boldsymbol{p} + \left[\Gamma^{(+)}-\Gamma^{(-)}\right]d\boldsymbol{x}d\boldsymbol{p}dt \end{aligned} \tag{4-2}$$

The first order terms of a Taylor series expansion of the left hand side of Eq. (4-2) (the dependence of $f^{(1)}$ from ($\boldsymbol{x}$,$\boldsymbol{p}$,t) is not explicitly written when it is obvious):

$$\begin{aligned} &f^{(1)}(\boldsymbol{x}+d\boldsymbol{x},\boldsymbol{p}+d\boldsymbol{p},t+dt)d\boldsymbol{x}d\boldsymbol{p} \;= \\ &\quad = f^{(1)}(\boldsymbol{x},\boldsymbol{p},t)d\boldsymbol{x}d\boldsymbol{p} + d\boldsymbol{x}\cdot\boldsymbol{\nabla}_{\boldsymbol{x}}f^{(1)} + d\boldsymbol{p}\cdot\boldsymbol{\nabla}_{\boldsymbol{p}}f^{(1)} + \left(\frac{\partial f^{(1)}}{\partial t}\right)dt + \cdots \end{aligned}$$

Thus, arranging Eq. (4-2) it is possible to write the **Boltzmann equation**:

$$\boldsymbol{v}\cdot\boldsymbol{\nabla}_{\boldsymbol{x}}f^{(1)} + \boldsymbol{F}\cdot\boldsymbol{\nabla}_{\boldsymbol{p}}f^{(1)} + \frac{\partial f^{(1)}}{\partial t} = \Gamma^{(+)}-\Gamma^{(-)} \tag{4-3}$$

Where the left-hand side represents the streaming motion of the molecules associated to the force field $\boldsymbol{F}$ whereas the right hand side is the effect of the two-body collisions taking molecules into or out the streaming path, according to the collisions rules. There will be neglected the apex (1) from now ahead for the distribution function.

4.5 LBE Framework

The lattice Boltzmann equation has been derived taking into account the possibility of the particles of moving freely in the space. Actually, discretising the spatial positions and the momenta that the molecules can have, and passing from a continuum change of time to discretised time steps vastly simplify the Boltzmann's resolution. As has been reported for the cellular automata models, the particle positions are limited to the nodes of the lattice and the momenta variations are limited as well, the particle mass is fixed and the velocity directions depend upon the model of node adopted. The latter can be in 2 dimensions (2D) or 3 dimensions (3D), and the directions can change from 8 to 27. At the moment square, hexagonal and triangular models have been presented, always in two dimensions. Paragraph 4.7 presents the D3Q19 which means three dimensions and 19 velocities, which has been chosen for the code implemented.

In order to simplify Eq. (4-2), assume that there are no external forces and the mass is normalised to 1,so that ***p=c***. Then, limit ***c*** to a set of vectors $\boldsymbol{c_\alpha}$ (***dx*** -> $\boldsymbol{\Delta x} = \boldsymbol{c_\alpha} \cdot \Delta t$) as seen in the case of HPP (Figure 3-2) or FHP (Figure 3-5), it is possible to write $f^{eq}(\boldsymbol{x},\boldsymbol{c},t)$ as $f_\alpha^{eq}(\boldsymbol{x},t)$. In addition, by discretising the time (dt -> Δt), it is possible to write $(\Gamma^{(+)}-\Gamma^{(-)})dt$ as $(\Gamma_\alpha{}^{(+)}(\boldsymbol{x},t)-\Gamma_\alpha{}^{(-)}(\boldsymbol{x},t))$. Hence the Eq. (4-2) can be rewritten, similarly to Eq. (3-3) as:

$$f_\alpha(\boldsymbol{x} + \boldsymbol{c_\alpha} \cdot \Delta t, t + \Delta t) - f_\alpha(\boldsymbol{x}, t) = \Gamma_\alpha^{(+)}(\boldsymbol{x}, t) - \Gamma_\alpha^{(-)}(\boldsymbol{x}, t) \qquad (4\text{-}4)$$

4.6 The BGK collision operator

The collision process is quite complicated and a deep analysis is beyond the scope of this thesis. There are different models well documented in [10] and among them the BGK is here presented. The latter is widely adopted due to its efficiency and simplicity in transcription as coding, the collision operator is given by:

$$\Omega_\alpha(\boldsymbol{x}, t) = \Gamma_\alpha^{(+)}(\boldsymbol{x}, t) - \Gamma_\alpha^{(-)}(\boldsymbol{x}, t) = -\frac{1}{\tau} \cdot \left[f_a\ (\boldsymbol{x}, t) - f_a^{eq}(\boldsymbol{x}, t)\right] \qquad (4\text{-}5)$$

This model was presented in 1954 by Bhatnagar, Gross and Krook [17] and subsequently simplified in 1992 by Quian, d'Humieres and Lallemand [18], which is the version here explained and adopted. The parameter τ is called relaxation time; f_α^{eq} is the equilibrium distribution of particles. The collision operator represents a relaxation of the distribution function f_α towards the equilibrium value f_α^{eq}.

The collision operator, as well as for the cellular automata Boolean operator, preserves the mass and the momentum, i.e.

$$\sum_\alpha \Omega_\alpha = 0$$
$$\sum_\alpha \boldsymbol{c_\alpha} \cdot \Omega_\alpha = 0 \tag{4-6}$$

Which implies that the equilibrium distribution function as well preserve the mass and momentum, i.e.

$$\rho = \sum_\alpha f_\alpha^{eq} = \sum_\alpha f_\alpha$$
$$\rho \cdot \boldsymbol{u} = \sum_\alpha \boldsymbol{c_\alpha} \cdot f_\alpha^{eq} = \sum_\alpha \boldsymbol{c_\alpha} \cdot f_\alpha \tag{4-7}$$

If the collision operator BGK is inserted in Eq.(4-4), the Lattice BGK (LBGK) model is obtained:

$$f_\alpha(\boldsymbol{x} + \boldsymbol{c_\alpha} \cdot \Delta t, t + \Delta t) - f_\alpha(\boldsymbol{x}, t) = -\frac{1}{\tau} \cdot \left[f_\alpha(\boldsymbol{x}, t) - f_a^{eq}(\boldsymbol{x}, t)\right] \tag{4-8}$$

This operator adopts a single relaxation time τ which is related to the kinematic viscosity of the simulated fluid by this equation:

$$\nu = c_s^2 \cdot \left(\tau - \frac{1}{2}\right) \tag{4-9}$$

where c_s^2 is the speed of sound, later introduced. The value of τ = 1 implies the complete relaxation, whereas for greater values of τ, the system is in subrelaxed

condition since the particle distribution is not completely relaxed to equilibrium. Finally, for τ smaller than 1, the system is called overelaxed, the particle distribution is moved beyond the equilibrium. For values of τ next to 0.5, there are numerical instabilities in the simulation. In fact, in the extreme case of τ = 0.5, there would be a viscosity equal to zero, hence no shear stress.

4.7 The lattice D3Q19

The Eq. (4-4) has been discretised along the node velocities of the lattice. There are different models of lattice, identified as DxQy where x indicates the number of dimensions (1, 2 or 3) and y the number of discrete velocities. The value of y is not arbitrary, since a lattice, for being used in LBGK, must satisfy some conditions which are reported in [19] and give the following set of equations for D3Q19:

$$\begin{aligned} w_0 + 6w_s + 12w_l &= 1 \\ 2w_s + 8w_l &= c_s^2 \\ 2w_s + 8w_l &= 3c_s^4 \\ 4w_l &= c_s^4 \end{aligned} \qquad (4\text{-}10)$$

Solving the system, the speed of sound $c_s^2=1/3$ and w_i are called the weights:

$$w_\alpha = \begin{cases} w_0=1/3 & \text{for } \alpha=0, \text{ the rest vector} \\ w_s=1/18 & \text{for } \alpha=1,..,6 \text{ the short vectors} \\ w_l=1/36 & \text{for } \alpha=7,..,18 \text{ the long vectors} \end{cases} \qquad (4\text{-}11)$$

The kinematic viscosity for this model is, according to Eq. (4-9):

$$\nu = \frac{1}{3} \cdot \left(\tau - \frac{1}{2}\right) \qquad (4\text{-}12)$$

And the Mach number is defined as:

$$Ma = \frac{|\dot{\boldsymbol{x}}|}{c_s} \qquad (4\text{-}13)$$

The D3Q19 is shown in Figure 4-2:

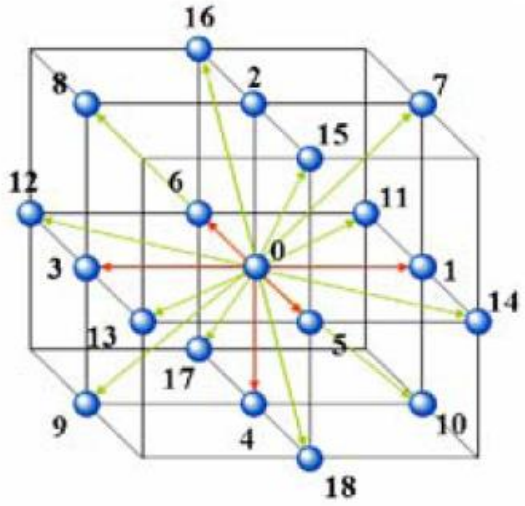


Figure 4-2 D3Q19 model [6]

as node velocity, the 19 discrete velocities are:

$$\boldsymbol{e}_{\alpha}= \begin{cases} (0,\,0,\,0) & \alpha=0 \\ (\pm1,\,0,\,0)\boldsymbol{c},\ (0,\,\pm1,\,0)\boldsymbol{c},\ (0,\,0,\,\pm1)\boldsymbol{c} & \alpha=1,..,6 \\ (\pm1,\,\pm1,\,0)\boldsymbol{c},\ (\pm1,\,0,\,\pm1)\boldsymbol{c},\ (0,\,\pm1,\,\pm1)\boldsymbol{c} & \alpha=7,..,18 \end{cases} \tag{4-14}$$

Assuming the lattice speed $\boldsymbol{c}=\Delta\boldsymbol{x}/\Delta t$ equal to one like the lattice constant and the time step.

According to Ali [6] D3Q19 is a good choice between D3Q15 which is prone to instability and D3Q27 is computationally expensive, which means it is a balance between reliability and efficiency.

4.8 The equilibrium distribution

The Eq. (4-8) presents one term f_{α}^{eq} called equilibrium distribution function which has not been defined. It derives from the Maxwell – Boltzmann velocity distribution according to a statistical mechanics approach. It is the probability of

finding particle in equilibrium. Its derivation is contained in Appendix A.2 and derived from [16], the result is reported here:

$$f_a^{eq}(\boldsymbol{x},t) = \rho \cdot w_\alpha \left[1 + \frac{\dot{\boldsymbol{x}} \cdot \boldsymbol{c_\alpha}}{\boldsymbol{c_s^2}} + \frac{(\dot{\boldsymbol{x}} \cdot \boldsymbol{c_\alpha})^2}{2 \cdot \boldsymbol{c_s^4}} - \frac{\dot{\boldsymbol{x}}}{2 \cdot \boldsymbol{c_s^2}}\right] \rightarrow$$
$$\rightarrow \rho \cdot w_\alpha \left[1 + 3 \cdot \dot{\boldsymbol{x}} \cdot \boldsymbol{c_\alpha} + \frac{9}{2} \cdot (\dot{\boldsymbol{x}} \cdot \boldsymbol{c_\alpha})^2 - \frac{3}{2} \cdot \dot{\boldsymbol{x}}\right] \quad (4\text{-}15)$$

It is easy to prove that Eq. (4-7) are verified, thus the equilibrium function as built, preserve the mass and the momentum.

4.9 Boundary conditions

The boundary conditions define the geometry of the system and the inlet/outflow conditions. There are different implementation proposed in [20] and "none of the methods is found to be throughout superior to the others. Instead, the choice of a best boundary condition depends on the flow geometry and on the desired trade – off between accuracy and stability".

4.9.1 Geometrical boundary conditions

The boundary conditions can be periodic or solid wall; the latter can be the boundaries of the control volume or obstacles. The periodic conditions are the easiest to implement, because the end of one edge is treated as the starting point of another edge. The distribution function coming out from one boundary, re – enters in the opposite boundary. Figure 4-3, is a simple two dimensional domain, with D2Q9 lattice model. If periodic conditions are applied between $\bar{x}$ and $\bar{N}_x$, they are formally written as:

$$f_\alpha(\bar{x}, y) = f_\alpha(\bar{N}_x, y) \qquad \alpha = 1{,}5{,}8{,}3{,}6{,}8 \quad \forall\, y$$

Taking into account that for the diagonal velocities represent two components, only the component in the x – direction is conserved between inlet and outlet.

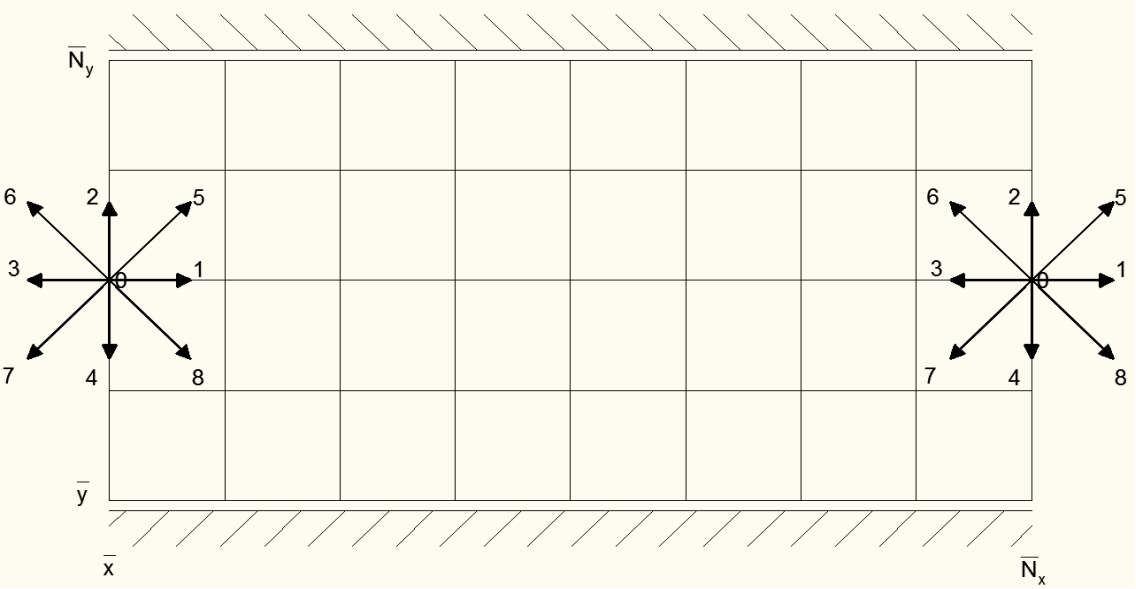


Figure 4-3 Sketch for a lattice nodes in periodic geometrical conditions

The bounceback boundaries are quite straightforward to implement according to Succi [10]. Unfortunately, despite the different variants they do not work perfectly, and the performances are satisfactory with $\tau \approx 1$ [20; 21]. The easiest way is to impose a complete change in direction if an obstacle/wall is found, it is called: fullway bounceback.

4.9.2 Inflow/Outflow boundary conditions

The inlet and the outlet of the flow domain have to be defined. This information may be given as velocity or pressure: Von Neuman and Dirichlet boundaries respectively. The former provides a vector with a given value of velocity. Whereas, the latter provides the density value, which for incompressible flow means giving the pressure (combining Eq. B-2 and Eq. B-3):

$$p = \rho \cdot c_s^2 \tag{4-16}$$

For the outflow, has been used the convective boundary condition which is optimal in case of unsteady flows [22; 23], and its formulation is:

$$\frac{\partial \boldsymbol{v}}{\partial t} + \bar{\boldsymbol{v}} \frac{\partial \boldsymbol{v}}{\partial n} = 0 \tag{4-17}$$

Where the $\bar{v}$ is the velocity in a location independent on the outflow surface and is chosen so that the overall conservation is maintained (i.e. it is the velocity required to make the outflow mass flux equal to the incoming mass flux), ***v*** is the velocity at the outlet, *t* the time and *n* is the perpendicular to the surface.

5 CHAPTER

Implementation of the code

The most important part of the work has been the understanding and modifying of the LBM code provided. This chapter will underline the way the code has been implemented, its subroutines and the flow diagram Figure 5-14. In addition, will be presented the conversion from lattice units to international system and how have been chosen some parameters for the simulation.

5.1 Introduction

The choice of using LBM in place of Navier – Stokes solver is suggested by many reasons:

1. It is easy to implement complex solid surfaces. This is the main reason because LBM are widespread in porous media works.
2. It is easy to create a parallelization of any LBM code. In fact it may be possible to simulate one node per processor, with no difficulty. It is possible because the LBM does not solve the Poisson equation for the pressure.
3. The Navier – Stokes equations are impossible to use when the Knudsen number is high (Figure 3-1). Even if this is not the case, it would be interesting to write the code for future work on smaller devices

The code provided has been successfully used by Djenidi research group for simulating both microchannels [24] and macroflows [25].

The code provided by Dr. Djenidi of the University of Newcastle (Australia) is a Lattice Boltzmann CFD three dimensional code for parallel computation, which has been written in different years (from 2003 to 2008) and is capable to simulate incompressible and isothermal fluid [3][26]. The geometry initially provided was different from the one reported in Figure 2-2 and the right geometry has been given separately, later. Some problems has been found in

the post – processing of the data, and they have been solved by the Tecplot © support. In addition some changes have been made to the code for making it more dynamic and user friendly.

The code is organised in subroutines, in order to be easier to read and understand. The following paragraphs will explain the code and the functions of the subroutines. It is important to underline the presence of two matrix: *node(α,x,y,z)* and *n_hlp(α,x,y,z)* which contain the velocities for each node *(x,y,z)* and discrete velocities (*α=0,…,18)* and are updated accordingly with the subroutine role.

5.2 Subroutine: read_reactor

The LBM requires the discretisation of the space. The dimensions of the original microreactor are reported in Figure 2-2, whereas in Figure 5-1 the dimensions in lattice units are reported. The total number of nodes is:

$$(620 + 60) \cdot 70 \cdot 70 = 3{,}332{,}000\ nodes \tag{5-1}$$

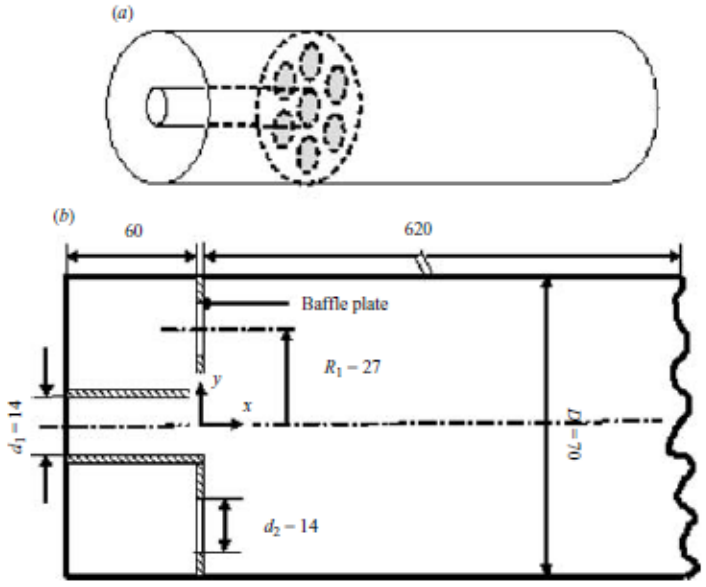


Figure 5-1 Schematic illustration (not to scale) of the microreactor model: (a) perspective view and (b) side view [lattice unit] [3]

The geometry is read one time at the beginning of the code, from the subroutine Read_Reactor. It is worth to mention that it is not possible to model a general curvilinear surface, thus it is used a stair – step approximation of the surface, in Figure 5-2 is represented the multi – holed baffle plate and the mesh.

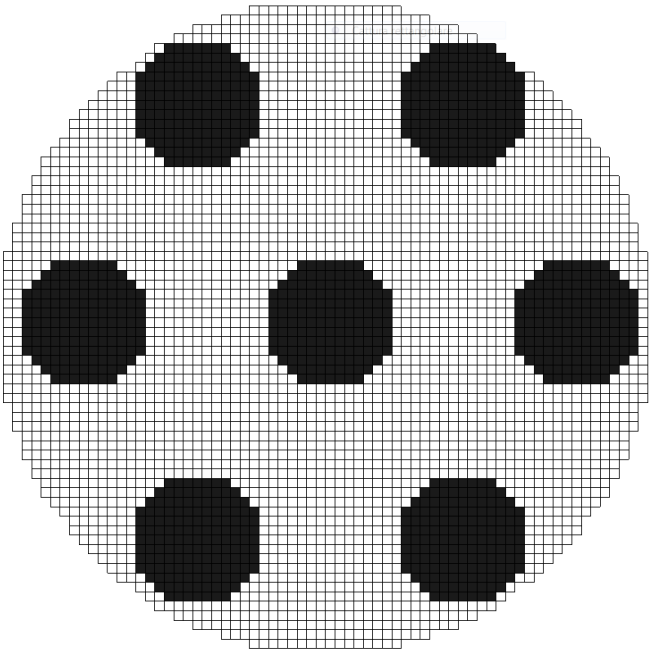


Figure 5-2 Computational domain, section y - z, x=60, multi holed baffle plate, (see Figure 5-1)

The annulus holes are equally spaced of an angle of 60°, their radius is equal to the radius of the inner tube *r_small=7 [lu]* and the the big diameter is *D_big=70 [lu]* whereas the distance between the big diameter and the centre of the annulus circles is *s=26 [lu]*.

5.3 Subroutine: init_density

This subroutine is called once and it is necessary in order to store in *node (α,x,y,z)* the product of density by the weights ($\rho \cdot w_\alpha$) presented in Eq. (4-11) and the same are subsequently copied in *n_hlp(α,x,y,z)*. The density is previously fixed to ρ*=1[mass/lu*^{3}*]*.

5.4 Subroutine: vector_ei

This subroutine is called once and it is necessary to store in the vector *e*(*α,3)* the directions reported in Eq. (4-14) for each discrete velocity in the three directions.

Now the loop with the discretised time starts and the maximum number of iterations can be decided. Djenidi [3] suggests using a number of iterations equal to 3,000 for reaching a steady – state solution, although 6,000 are used in order to remove any remaining effects of the transient regime. However, for this work a maximum of 4000 iterations have been adopted because during the optimisation, the number of iterations will play a key role in determining the velocity of the simulation, and no problem of stability have been detected with this number of iteration.

5.5 Subroutine: density_check

This subroutine is used to calculate the fluid density all over the domain. This is a good indicator of the stability of the code and in fact the value should be constant. If not it means that it has not been reached the steady state.

5.6 Subroutine: redistribute

This subroutine is used for two purposes:

1. *Initialisation of the solution*
2. *Inlet condition*

In Table 5-1 are listed the variables used for the initial conditions. The *mass_ratio* is defined in [2; 3] is the mass flow ratio between the inner tube and the annulus flow, *nu* is the kinematic viscosity, which is constant for all the cases, as will be explained in section 5.16.2. The Reynolds number is defined after the baffle plate:

$$Re = \frac{u\_avg \cdot D\_big}{nu} \rightarrow u\_avg = \frac{Re \cdot nu}{D\_big} = 0.025 \left[\frac{lu}{ts}\right] \tag{5-2}$$

Table 5-1 Redistribute initial conditions

Variable	Value
Reynolds number	*100*
R_big	*35* *[lu]*
r_small	*7* *[lu]*
mass_ratio	*0.05*
nu	*0.0175* *[lu²/ts]*

and for the continuity it is possible to write:

$$\left.\begin{array}{l} \dot{m}_{in} + \dot{m}_{ann} = \dot{m}_{out} \\ \dfrac{\dot{m}_{in}}{\dot{m}_{ann}} = 0.05 \end{array}\right\} \rightarrow 0.05 \cdot \dot{m}_{ann} + \dot{m}_{ann} = \dot{m}_{out} \rightarrow \dot{m}_{ann} = \frac{\dot{m}_{out}}{1.05}$$

$$\begin{array}{ll} \dot{m}_{in} = u\_xs \cdot \rho \cdot A\_in & A\_in = \pi \cdot (r\_small)^2 \\ \dot{m}_{ann} = u\_xa \cdot \rho \cdot A\_ann & A_{ann} = A\_out - A\_in \\ \dot{m}_{out} = u\_avg \cdot \rho \cdot A\_out & A\_out = \pi \cdot (R\_big)^2 \end{array} \tag{5-3}$$

$$u\_xs = 0.02976 \left[\frac{lu}{ts}\right];\ u\_xa = 0.02480 \left[\frac{lu}{ts}\right]$$

u_xs is the velocity, *A_in* the area and $\dot{m}_{in}$ the mass flow at the inner tube. *u_xa* is the velocity, *A_ann* the area and $\dot{m}_{ann}$ the mass flow at the annulus. *u_avg* is the velocity, *A_out* the area and $\dot{m}_{out}$ the mass flow at the outlet.

In the first iteration all the domain is initialised with the velocity *u_xa*, whereas the inner tube is initialised with *u_xs*. In the next iterations the conditions defined by *u_xa* and *u_xs* are applied only at the first node. These velocities are only in the axial direction of the tube. No conditions are applied for the other two components neither for the initialisation of the solution nor the inlet condition.

The density is calculated from *node(α,x,y,z)*, as sum of the values of node in a point (x,y,z) and (*α*=0,1,...,18). Later on, the equilibrium function Eq. (4-15) is calculated for each point and saved in *node(α,x,y,z)*. If a point is an obstacle, (both bounder or baffle plate) *node(α,x,y,z)* (*α* =0,...,18) is set to zero.

In comparison with the experiment, where a T junction is present as in Figure 2-1, the inlet conditions are not really well defined in the sense that the flow in the experiment presents even the radial components. However, this feature does not strongly influence the simulation as will be explained in section 5.19. In addition, this implementation on one hand does not allow having a direct control of the Reynolds number and takes quite some time to reach the steady state, on the other hand, it is simple and it works, as stated by Djenidi [26].

5.7 Subroutine: streaming

This subroutine is used to make the streaming process (section 4.2). The value of the velocity in the position 0 is just copied from *node(α,x,y,z)* to *n_hlp(α,x,y,z)*. Whereas for the other position (*α*=1,...,18) the values are copied from the node to its neighbour, Figure 3-7, according to the discrete velocities which is streaming. Again, the values are copied from *node(α,x,y,z)*, to *n_hlp(α,x,y,z).*

Consider, starting point (5,6,9), and using Figure 4-2 as reference for the discrete velocity direction:

discrete velocity 1 (*α*=1): *n_hlp(1,6,6,9)=node(1,5,6,9)*

discrete velocity 3 (*α*=3): *n_hlp(3,4,6,9)=node(3,5,6,9)*

discrete velocity 7 (*α*=7): *n_hlp(7,6,7,9)=node(7,5,6,9)* (5-4)

discrete velocity 9 (*α*=9): *n_hlp(9,4,5,9)=node(9,5,6,9)*

Obviously for the first and last node of a direction (*x* or *y* or *z* equal to 1 or their maximum), it is not possible to stream. For *x=1* it is defined the inlet condition whereas for *x=outlet*, it will be defined the convective boundary condition. The y

or z equal to 1 or their maximum are in the obstacle state (the y and z lattice width is not 70 but 73), which means that the velocity is zero, so there's no need to propagate any velocity there.

5.8 Subroutine: bounceback

This subroutine is used to consider the presence of the obstacles, and it uses a no-slip fullway bounceback boundary condition. In particular, if the node is an obstacle, the value reported in *n_hlp(α,x,y,z)* is rotated and copied in *node(α,x,y,z).*

Consider, the point (5,6,72), and using Figure 4-2 as reference for the discrete velocity direction:

discrete velocity 1 (*α*=1): *node(3,5,6,72)=n_hlp(1,5,6,72)*

discrete velocity 3 (*α*=3): *node(1,5,6,72)= n_hlp (3,5,6,72)*

discrete velocity 7 (*α*=7): *node(9,5,6,72)= n_hlp (7,5,6,72)*

discrete velocity 9 (*α*=9): *node(7,5,6,72)= n_hlp (9,5,6,72)* (5-5)

This process must preserve the mass and momentum conservation for each node. The velocity vector of all fluid densities are inverted, and the fluid densities are sent back to the node where they were located before the last propagation step, but with opposite velocity vector.

5.9 Subroutine: relaxation

This subroutine is used for the local propagation of the equilibrium function, paragraph 4.8, and within this subroutine is called the convective_BC subroutine. For each point (*x,y,z)* based on *n_hlp(α,x,y,z)* the density and the velocities *u_x*, *u_y* and *u_z* (taking into account the component of the discrete velocities, Figure 4-2) are computed. If the point is an obstacle the velocities are set to zero. It is now possible to compute the equilibrium function Eq. (4-15) and

the Eq. (4-8) is computed, in order to calculate the distribution:$f_\alpha(\boldsymbol{x} + \boldsymbol{c_\alpha} \cdot \Delta t, t + \Delta t)$. This value is stored in *node(α,x,y,z).*

Before calling the convective_BC, are defined the following vectors:

u_curr(y,z), v_curr(y,z), w_curr(y,z)	At boundary	
u_prev(y,z), v_ prev (y,z), w_ prev (y,z)	At boundary -1 in time	(5-6)
u_bdy1(y,z), v_ bdy1 (y,z), w_ bdy1 (y,z)	At boundary -1 in x direction	

which are saved when *x=outlet-1*, one node before the end of the domain. Those are calculated in the relaxation subroutine, and *i*_prev (*i*=u,v,w) are stored during the previous time step. If *time=1*, *i_prev*. are equal to *i_curr*.

5.10 Subroutine: convective_BC

This subroutine is used for defining the outlet conditions, based on Eq. (4-17). At first is computed the mean outflow velocity: U_c at *x=outlet* based on *u_curr* etc. The next step is updating the *u_curr* etc:

$$u\_curr(y,z) = \frac{(u\_prev(y,z) + U_c \cdot u\_bdy1(y,z))}{1 + U_c}$$
$$u\_prev(y,z) = u\_curr(y,z) \tag{5-7}$$

Similarly for the other two components.

This subroutine ends the calculations for the LBM code, the next subroutines are defined for the evaluations of the objectives and the production of the output.

5.11 Subroutine: write_tecplot

This subroutine is called only once, when the last iteration is reached. It evaluates all the velocities based on *node(α,x,y,z)* and save them into three vectors ***u****(x,y,z)*, ***v****(x,y,z)* and ***w****(x,y,z)*. Subsequently, based on the Eq. (4-16) it is possible to evaluate the pressure all over the domain. Finally the pressures

one node after the inlet (*p_in*) and one node before the outlet (*p_out*) are calculated. In addition, for each point of the domain, the components of the velocity vector and the pressure are written on a file, with binary encoding.

5.12 Subroutine: write_vorticity

This subroutine is called only once, when is reached the last iteration. It evaluates the vorticity vector according to the definition:

$$\omega\_x = \frac{\partial w}{\partial y} - \frac{\partial v}{\partial z} \tag{5-8}$$

$$\omega\_y = \frac{\partial u}{\partial z} - \frac{\partial w}{\partial x} \tag{5-9}$$

$$\omega\_z = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \tag{5-10}$$

The vorticity magnitude is given by:

$$\omega\_tot = \sqrt{\omega\_x^2 + \omega\_y^2 + \omega\_z^2} \tag{5-11}$$

The calculation of the vorticity has been delicate to incorporate, because different numerical schemes have been found for the numerical evaluation of the vorticity in the given velocity field. The lack of articles on this topic has moved to implement all of them in order to see the differences.

As clearly illustrated in the Figure 5-3, Figure 5-4, Figure 5-5, Figure 5-6, the behaviour of the methods is almost the same and among them has been preferred the circulation method because, despite the higher computational effort it proves to have a better behaviour in comparison with the third[27]. Whereas, the first one is too simple to have a good evaluation and the latter, is indicated for PIV measurements, further information are present in [28]. Hence, at the end of the subroutine all the vorticity magnitudes are summed up in order to evaluate the total vorticity of the microreactor In addition, the single components of the vorticity vector are written on a file with binary encoding.

5.12.1 Vorticity: definition

The definition of vorticity has been literally transposed in numerical calculations- For instance the Eq. (5-8) has been implemented as:

$$\omega\_x = (w(x, y+1, z) - w(x, y, z)) - (v(x, y, z+1) - v(x, y, z)) \quad (5\text{-}12)$$

The vorticity magnitude, Eq. (5-11), according to the vorticity definition is reported in Figure 5-3.

5.12.2 Vorticity: Circulation method

The Stokes theorem has been used, the method is called the circulation method since it calculates the circulation around a point. The implementation of Eq. (5-8) is:

$$\begin{aligned}\omega\_x = 1/8 \cdot [&(v(x, y-1, z-1) + 2 \cdot v(x, y, z-1) + v(x, y+1, z-1)) + \\ &+(w(x, y+1, z-1) + 2 \cdot w(x, y+1, z) + w(x, y+1, z+1)) - \\ &- (v(x, y+1, z+1) + 2 \cdot v(x, y, z+1) + v(x, y-1, z+1)) - \\ &-(w(x, y-1, z+1) + 2 \cdot w(x, y-1, z) + w(x, y-1, z-1))]\end{aligned} \quad (5\text{-}13)$$

The vorticity magnitude, Eq. (5-11), according to the circulation method is reported in Figure 5-4.

5.12.3 Vorticity: least – squares method

The least – squares method reduces the effect of fluctuations. It is indicated more for PIV measurements. For instance the Eq. (5-8) has been implemented as:

$$\begin{aligned}\omega\_x = 0.1 \cdot [&(2 \cdot w(x, y+2, z) + w(x, y+1, z) - w(x, y-1, z) - \\ &-2 \cdot w(x, y-2, z) - 2 \cdot v(x, y, z+2) - v(x, y, z+1) + v(x, y, z-1) + \\ &+2 \cdot v(x, y, z-2))]\end{aligned} \quad (5\text{-}14)$$

The vorticity magnitude, Eq. (5-11), according to the least squares method is reported in Figure 5-5.

5.12.4 Vorticity: Richardson extrapolation

The Richardson extrapolation has been designed in order to produce a smaller truncation error. The Eq. (5-8) has been implemented as:

$$\omega\_x = 1/12 \cdot [-w(x, y+2, z) + 8 \cdot w(x, y+1, z) - 8 \cdot w(x, y-1, z) + \\ +w(x, y-2, z) - v(x, y, z+2) + 8 \cdot v(x, y, z+1) - 8 \cdot v(x, y, z-1) + \\ +v(x, y, z-2)] \quad (5\text{-}15)$$

The vorticity magnitude, Eq. (5-11), according to the Richardson extrapolation method is reported in Figure 5-6.

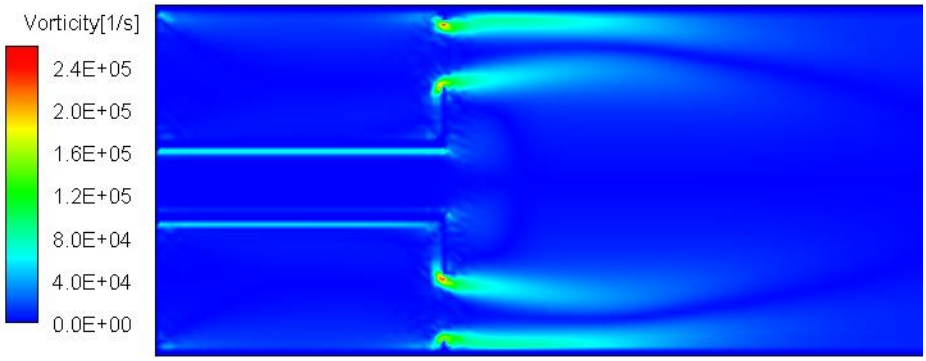


Figure 5-3 Vorticity Magnitude [1/s], definition

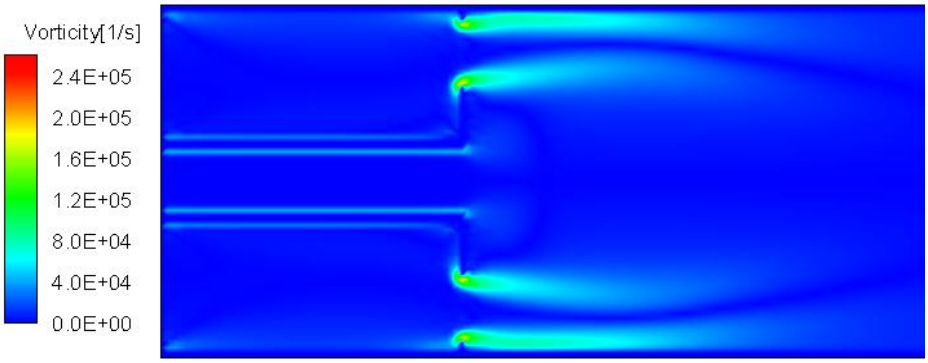


Figure 5-4 Vorticity Magnitude [1/s], Circulation method

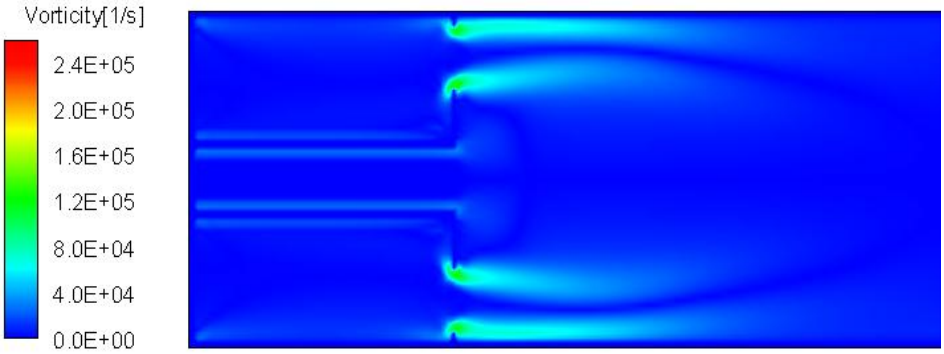


Figure 5-5 Vorticity Magnitude [1/s], Least - squares method

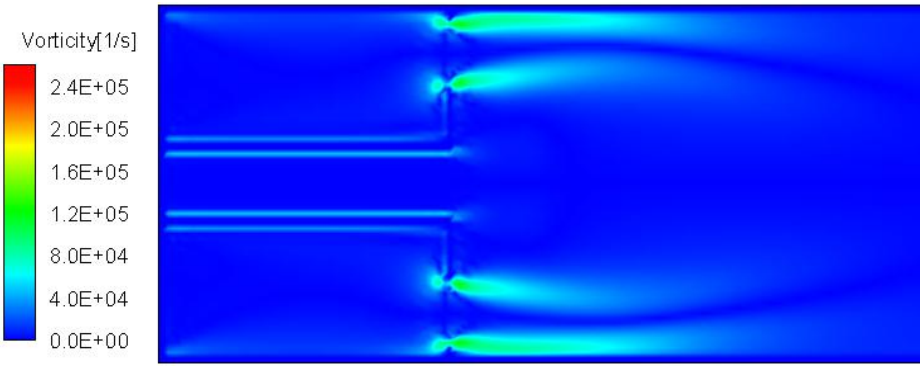


Figure 5-6 Vorticity Magnitude [1/s], Richardson extrapolation method

This LBM code as previously explained is written to be used with MPI (Message Passing Interface) protocol. This means that it is used in a multiprocessor environment. Hence, the results and the files are produced per processor. In order to gather the information between different processors are required MPI functions or FORTRAN subroutines. The latter, are used to merge the files written from the different processors.

5.13 Subroutine: assemble_tecplot

This subroutine is called only once, when is reached the last iteration. It creates a non-binary file containing 7 different columns: "*x, y, z, u, v, w, pressure*". It reads from the files created by the subroutine write_tecplot, paragraph 5.11. In addition the header has been created in order to be correctly read by the software Tecplot ©.

5.14 Subroutine: assemble_vortex

As for the previous subroutine, this is called only once, when is reached the last iteration. It creates a non-binary file containing 10 different columns: "*x, y, z, u, v, w, pressure, w_x, w_y, w_z*". It reads from the files created by the subroutine write_tecplot and write_vorticity, paragraph 5.11 and 5.12. In addition the header has been created in order to be read by the software Tecplot ©.

5.15 From SI to lattice units

This paragraph will describe a complicated argument for the LBM world. In fact, it was not easy to find the rules which explain exactly the conversion from SI to LB units. Some references have been used [29-31], which presents different approaches to the calculation. First of all, it is important to underline that the equations that we are solving, e.g. Eq. (4-8), are discretised in time and in space, this values are respectively Δt and Δx both equal to 1. The rationale behind the choosing of the lattice units is determined by two constraints: first, the lattice units should be consistent with the physical units (e.g. seconds with time step) and second, the parameters have to be chosen in respect to the

required accuracy. In fact changing the grid space (Δx) or the time step Δt provides different solutions. There is always a balance between stability/efficiency of the solution and higher accuracy, reached with smaller time step and grid space. Unfortunately, there is not a precise rule, but it is the experience which helps in the choice. The Table 5-2 indicates the relation between the LB units: "*lb*" and the physical units: "*phy*".

Table 5-2 Conversion Factors

Length	$D\_big_{phy} = 300E-6m;\ D\_big_{lb} = 70\ lu \longrightarrow dx = \frac{300E-6}{70}\left[\frac{m}{lu}\right] = 4.286E-6\left[\frac{m}{lu}\right]$
Viscosity	$\nu_{phy} = 0.9546E-6\frac{m^2}{s};\ \nu_{lb} = 0.0175\frac{lu^2}{ts} \longrightarrow d\nu = \frac{0.9546E-6}{0.0175}\left[\frac{m^2}{s}\frac{ts}{lu^2}\right] = 54.55E-6\left[\frac{m^2}{s}\frac{ts}{lu^2}\right]$
Time:	$dt = \frac{dx^2}{d\nu} = \frac{(4.286E-6)^2}{54.55E-6}\left[\frac{s}{m^2}\frac{lu^2}{ts}\left(\frac{m}{lu}\right)^2\right] = 0.3367E-6\left[\frac{s}{ts}\right]$
Velocity:	$d\nu = \frac{dx}{dt} = \frac{4.286E-6}{0.3367E-6}\left[\frac{m}{lu}\frac{ts}{s}\right] = 12.73\left[\frac{m}{lu}\frac{ts}{s}\right]$
Density:	$\rho_{phy} = 1002\frac{kg}{m^3};\ \rho_{lb} = 1\frac{mass}{lu^3} \longrightarrow d\rho = \frac{\rho_{phy}}{\rho_{lb}} = \frac{1002}{1}\left[\frac{kg}{m^3}\frac{lu^3}{mass}\right]$
Mass:	$dm = d\rho \cdot dx^3 = 1002 \cdot (4.286E-6)^3\left[\frac{kg}{m^3}\frac{lu^3}{mass}\left(\frac{m}{lu}\right)^3\right] = 7.886E-14\left[\frac{kg}{mass}\right]$
Pressure:	$dp = d\rho \cdot d\nu^2 = 1002 \cdot 12.73^2\left[\frac{kg}{m^3}\frac{lu^3}{mass}\left(\frac{m}{lu}\frac{ts}{s}\right)^2\right] = 162.4E3\left[\frac{Pa}{pres}\right]$

This table allows making a fast change from lattice units to SI units and vice versa. It is important to underline that the pressures are defined compared to a reference pressure, which means that if it is required to calculate the pressure in a point, the reference pressure has to be defined first. On the contrary, if the difference of pressure is required, it can be computed with no pressure reference definition.

5.16 Preliminary calculations

In order to set up the design point, some calculations needed to be done. Some of them have been presented in Eq. (5-2), (5-3).and Table 5-2; others will be introduced in this paragraph.

5.16.1 Density

The fluid used in the experiment is a mixture of water and glycerine ($2\%_{wt}$) [2]. Using the tables in [32] it is possible to know the density at 25°C: *1001.8 kg/m^3*. Whereas, the density in lattice Boltzmann units has been arbitrarily set to: *1 mass/lu^3*.

5.16.2 Dynamic and kinematic viscosities

As for the previous case, using the tables in [33] it is possible to evaluate the dynamic viscosity at 25°C.

Table 5-3 Dynamic Viscosity

Dynamic Viscosity [centipoise]				
		Temperature [°C]		
		20	**25**	30
$\%_{wt}$ of glycerine	0	1.005	0.90285	0.8007
	2	1.066	**0.95628**	0.84656
	10	1.31	1.17	1.03

Using a linear interpolation the dynamic viscosity of the mixture is found to be *0.95628 centiPoise*.

The kinematic viscosity is calculated as:

$$\nu = \frac{\mu}{\rho} = \frac{0.95628E-3}{1001.8} = 0.9546\frac{m^2}{s} \tag{5-16}$$

The viscosity in lattice Boltzmann units has been fixed to 0.0175 [lu^2/ts], for two reasons:

1. *This value has been used previously in the code provided*
2. *The viscosity is directly related to the relaxation time Eq.* (4-9) (4-12)

In particular, the second reason requires more detailed information. In fact, using a constant relaxation time for all the simulations is not really a good choice. In fact, the value of the relaxation time strongly influences the error in the LBM, [21; 34]and different results are proposed in papers:

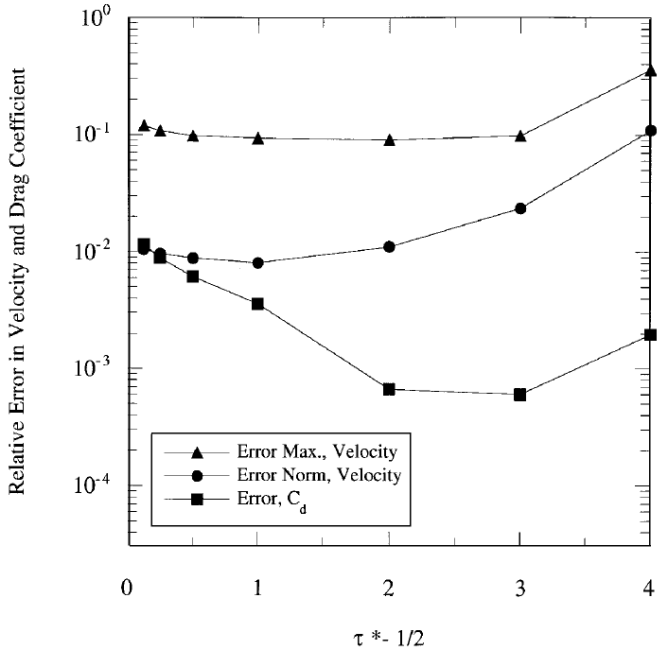


Figure 5-7 Comparison of relative errors in flow about octagonal cylinders as a function of relaxation time with Re=10 and 170x10 grid [21]

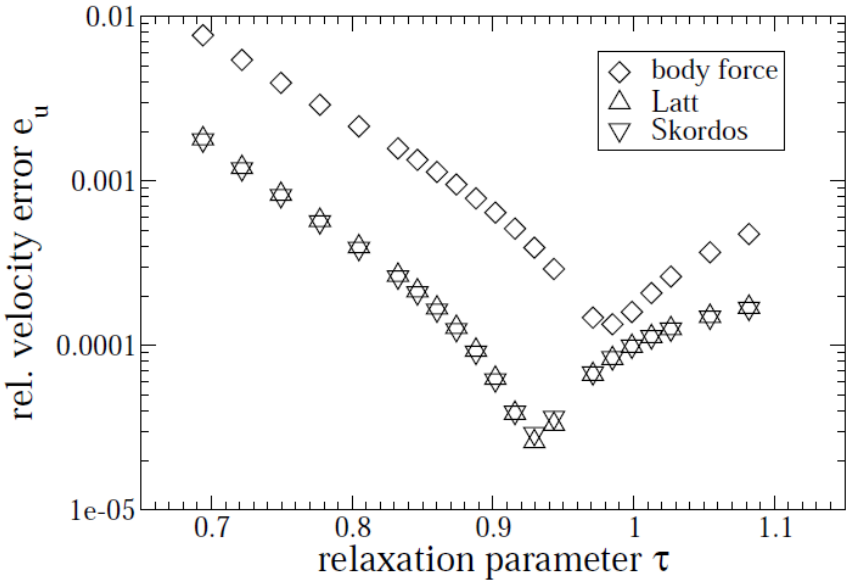


Figure 5-8 Behaviour of the relative velocity error for Re=10 [34]

In the paper [34] a study carries out a comparison with different Reynolds number and it emerges that the higher the Reynolds number the higher is τ in order to have the minimum error, but the values is always included between 0.9 and 1 (Reynolds between 1 and 100). Different results are given in [21], hence a parametric study would have been more interesting in order to capture the different behaviour of the code with the change of Reynolds number. However, the value of τ for all the simulations is (Eq. (4-12)):

$$\tau = \frac{6\nu + 1}{2} = 0.5525 \tag{5-17}$$

The value is next to 0.5, hence it is in an unstable zone, but since in the code provided [26] there were no proofs of unsuitability it has been kept for this work. In addition, this value of viscosity used for the Reynolds number brings to an average velocity for the LB simulation of *0.025 lu/ts* $(Re \cdot \nu / D = 0.025 lu/ts)$ which is a good value to use in the simulation. In fact, this velocity keeps the maximum Mach number lower than 0.1 (Eq. (4-13)).

5.17 Grid Resolution and Time Resolution

The grid resolution and the time resolution have been calculated in Table 5-2, using the same of the code provided. It is important to underline that the conversion factors for the grid and time resolution play a big role in terms of stability and efficiency of the simulation. In fact, smaller resolution brings better accuracy but requires more time in order to reach the stable solution. It is a compromise to find in order to get good results in reasonable time.

5.17.1 Choice of the time step

The time step *dt* of a LB simulation is intimately related to the Mach number, and cannot be chosen freely [35]. In order to simulate an incompressible fluid, the Mach number and consequently the time step must be chosen very small and this neglects the compressibility effects. Hence, as Figure 5-9 shows, in order to achieve a better accuracy, it is required a small time step (e.g. $dt=10^{-6}$, as presented in Table 5-2).

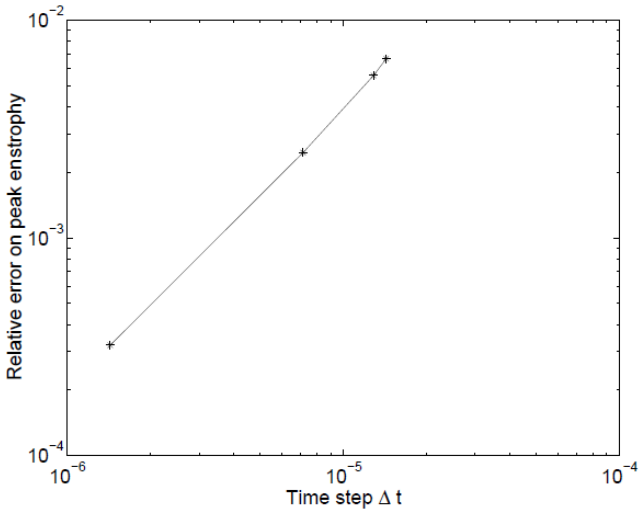


Figure 5-9 Relative error on the value of enstrophy at the first enstrophy peak related to the time step [35]

5.17.2 Choice of the space step

The space step is a parameter which, as the time step, may control the accuracy and efficiency of the simulation. It is suggested to adopt values of these steps such as $\delta_t \sim \delta_x^2$ [30]where $\delta_t = \frac{1}{number\ of\ iterations}$ and $\delta_x = \frac{1}{characteristic\ length} = 0.0143$, hence:

$$Number\ of\ iterations \sim \frac{1}{\delta_x^2} = 4900 \tag{5-18}$$

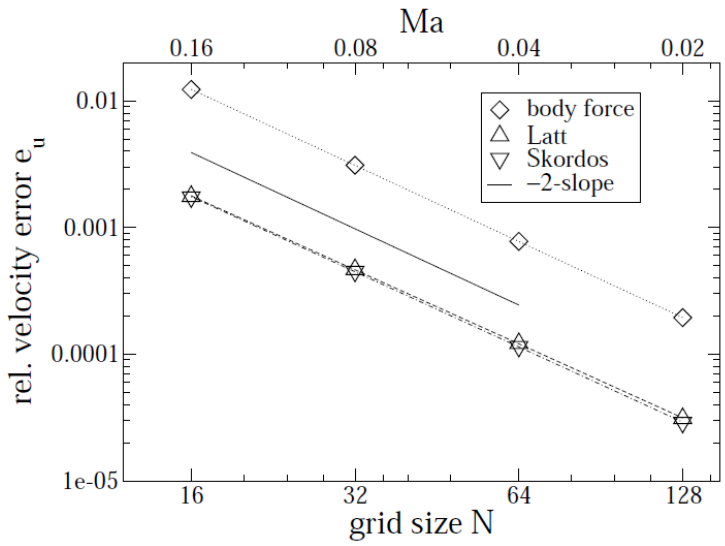


Figure 5-10 Velocity errors as function of the system size N and Mach [34]

As expected, higher grid size implies better accuracy. In particular it should be noted that the slope of the convergence is very close to *-2* as in the time step case. It shows that the LBM are second order accurate with respect to the velocity [34].

5.18 Mass increase

The LBM simulations are often accompanied by a phenomenon known as "*mass leakage*" [36] which implies that the mass of the system is not conserved exactly. In fact, there will be mass loss/gain during the beginning of the simulation. This leakage is bond to the accuracy of the simulation. In fact, for

high Mach number it is experienced higher fluctuations in the mass of the system, and more time steps will be necessary in order to reach the steady state. This behaviour is influenced even by the relaxation time, the grid and time resolution. The reason of this performance are in the interchange of mass between fluid nodes and boundary nodes, a good demonstration is contained in [36]. Modified boundary conditions can remove this drawback, but sometimes even if the mass is conserved at the walls locally, still the total mass of the system changes with the iteration step [12]. In this case, the inlet and outlet velocity conditions may play the key role. “Due to the equation of state of lattice fluid Eq. (4-16), a pressure gradient is equivalent to a density gradient, leading to a larger inlet than outlet flux, and mass is accumulating in the system” [34]. There are different models presented in [34], [36] and [37], in order to deepen this argument.

5.19 Validation of the code

After the presentation of the code and its most important parameters, it is important to make the validation of the code. For this purpose both the experiments and the solution of the same problem with Navier – Stokes equations using commercial software: Ansys CFX, carried out by PhD Salwan Saddawi were used. Though few changes has been made to the code because of the lack of some information. In addition, some changes have been made in order to take more familiarity with the code itself, hence was important to make sure that the changes did not change the behaviour of the code.

Figure 5-11 and Figure 5-12 show the ensemble of the first part of the reactor with coloured vectors according to the velocity magnitude for both the code and the CFX simulation. Figure 5-13 shows the velocity field in the same section for the code simulation together with the CFX, Djenidi [3] simulations and the experiment [2].

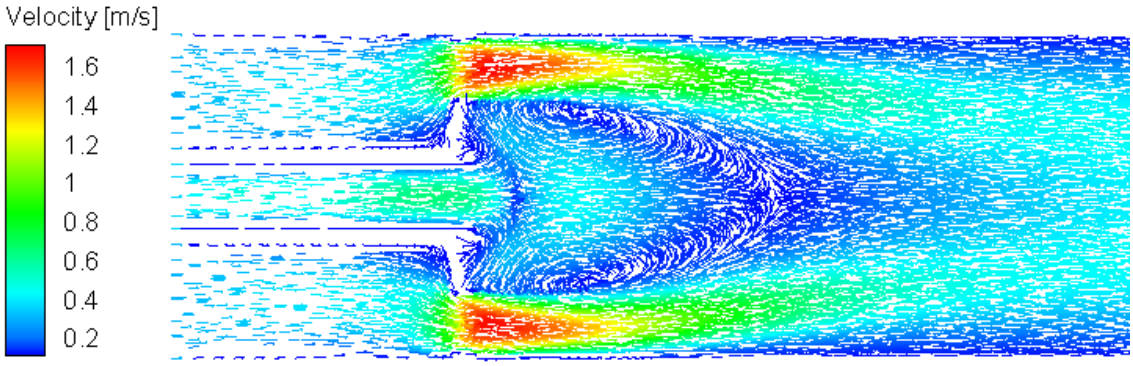


Figure 5-11 Numerical simulation CFX Salwan

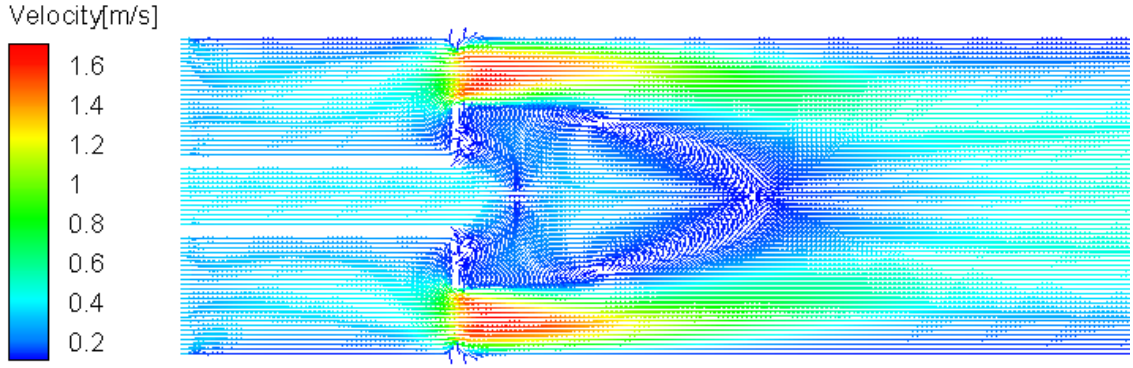
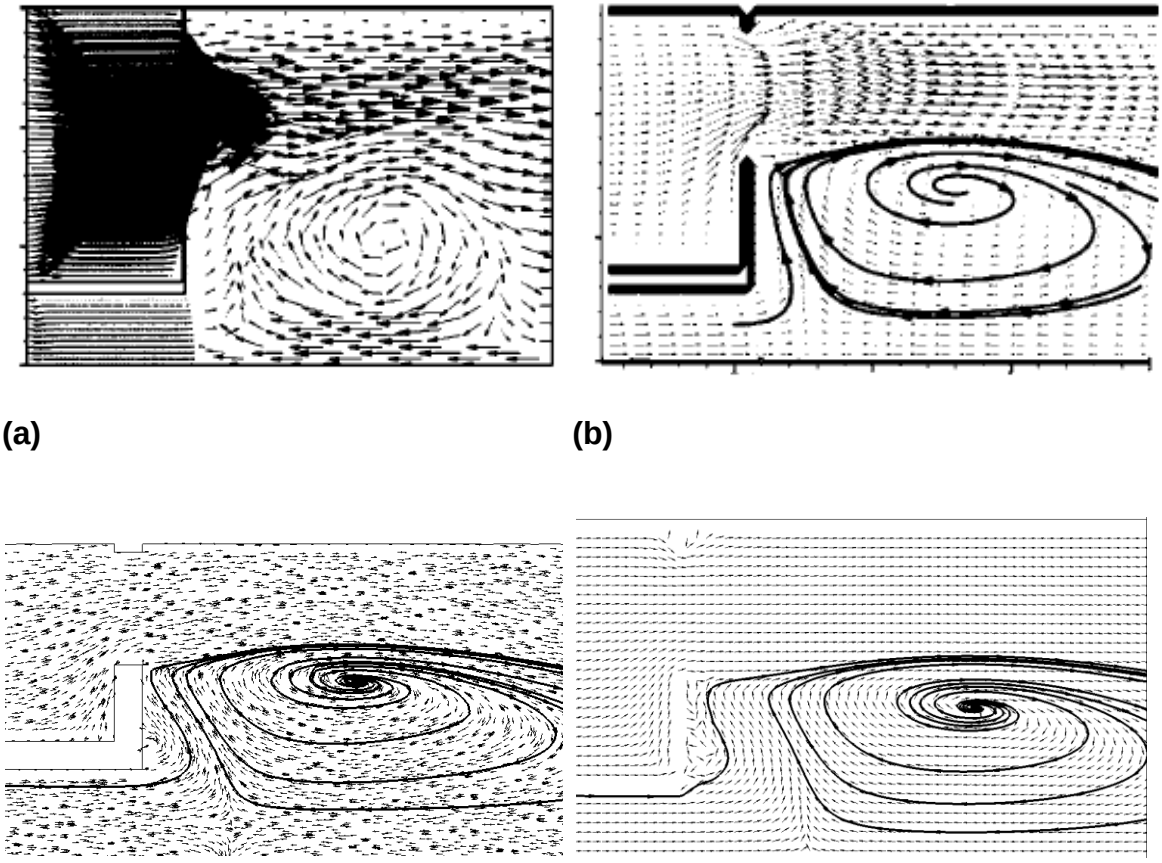


Figure 5-12 Numerical simulation code LBM

From the pictures of the reactor Figure 5-11and Figure 5-12 or the particular next to the multi – holed baffle plate, Figure 5-13, it is possible to see that the values of velocity are similar and the shape from both the vectors and the streamlines is almost equal. The differences are unavoidable because, the CFX and the experiments consider the holes not like a stair step approximation. In addition the latter is affected by errors and all the simulations, as well, are not completely reliable, because it depends on the mesh, the model adopted etc. Finally, some of the data used in the simulation have not been given and some assumptions have been made (e.g. the outlet velocity is $\approx 0.3\, m/s$ [2]). However for the purposes set, the similarity between the different models and the previous work with this code [3] are sufficient to say that the code developed behaves in a correct way and it is ready to be used in an optimisation cycle.



(a) **(b)**

(c) **(d)**

Figure 5-13 Comparison between (a) Measurements [2], (b) numerical simulations LBM [3], (c) numerical simulations CFX Salwan, (d) numerical simulations LBM. The simulations of the velocity field are taken in the same section with the multi – holed baffle plate, and some streamlines are shown

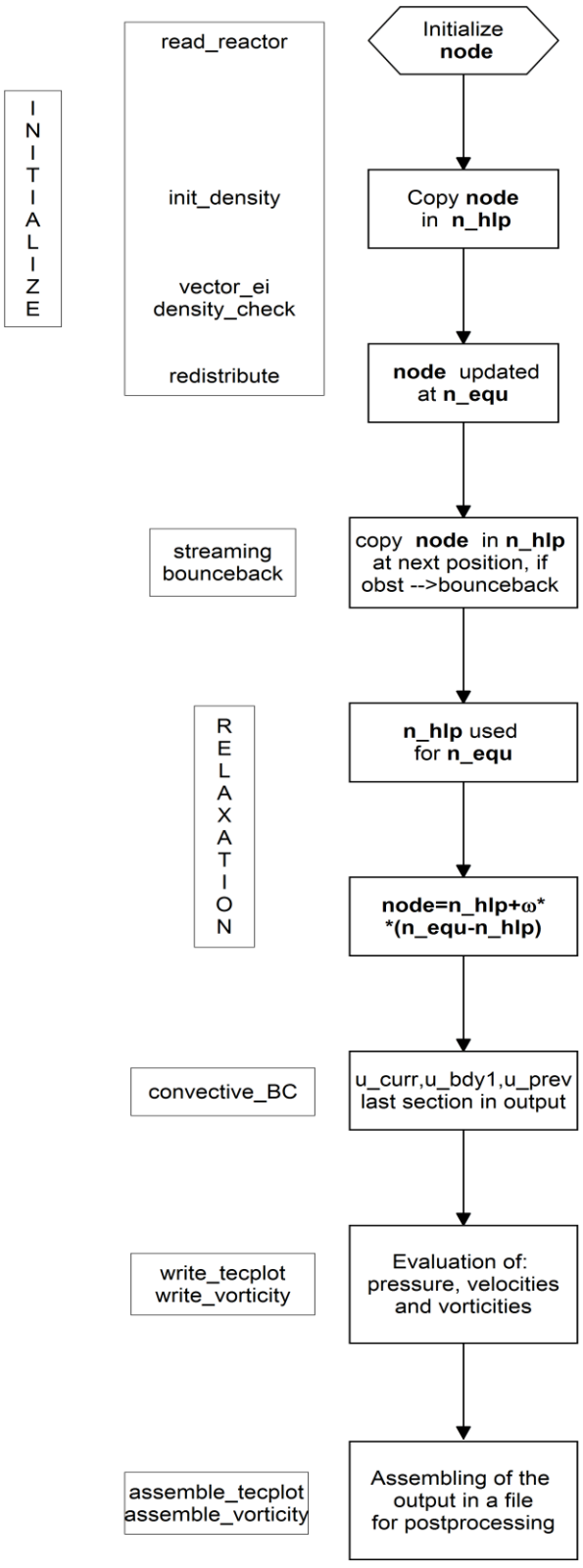


Figure 5-14 Flow diagram of LBM code

6 CHAPTER

Multi-objective optimisation

This chapter introduces to the optimisation theory and it is focused on the Tabu Search algorithm. Finally is presented the cycle of the multi objective optimisation, which has been used in the multi objective optimisation, main objective of this work.

6.1 Introduction

The optimisation process is nowadays widely used in order to find the best possible solution to a problem, given a set of constraints. This solution is called the global optimum and most of the time it represents an innovative design in comparison with the standard. Unfortunately, most of the time when an optimisation needs to be devised, the model of the problem do not present only one objective, but more than one. In fact, the real – world problems depend upon different objectives, most of the time in conflict between them, e.g. increasing the vorticity in the microreactor implies an increase of the pressure drop across the device itself. Hence, it is obvious that the optimisation process can be distinct in:

- *Single – objective optimisation*
- *Multi – objective optimisation*

The multi – objective are more used nowadays and presents more complexity than the single objective. In fact, if the single objective is most of the time a non linear problem [38] the multi – objective will provide not the global optimum but a plethora of different solutions and may be highly non linear. Each of the solution will accentuate some features and penalise others, or they can be good compromises among the objectives. Thus, the role of the engineer is to discern the best according to other requirements, e.g. cost, and the surrounding environment.

At first, some algorithms tried to consider the multi – objective nature of some problems by giving to each objective a “weight” and evaluate the total feasibility of a design by “summing” the objectives. Unfortunately, the *composite objective method*, that is its name, requires a precise definition of the weights, and it introduces pre – conceptions to the design, despite its ease in implementing and solving. Hence, “performing real multi – objective optimisation and finding a Pareto front is the only effective way to find a set of designs satisfying performance criteria in an industrial context” [5].

6.2 Overview on the optimisation algorithms

In the last three decades, particularly the last decade, a wealth of multi – objective optimisers have been developed and some problems have been solved by using these optimisers [39], but they need to be improved, as there are many unresolved problems [40]. In addition, it has been revealed [41] that the performances of the evolutionary algorithms, which are the more studied nowadays, are overshadowed by simulated annealing or tabu search. The latter, despite the higher complexity due to the local search methods, later explained, are most suitable in some problems, such as the one faced in this work.

Mathematically an optimisation problem can be expressed as:

$$\begin{cases} \text{Minimize} & \bar{f}_b(\bar{x}), & b = 1, \dots, B & \text{(6-1)} \\ \text{Constraints on:} & \begin{cases} \text{Objective functions} & \bar{g}_j(\bar{x}) \geq 0 & j = 1, \dots, J \\ & \bar{h}_k(\bar{x}) = 0 & k = 1, \dots, K \end{cases} & & \text{(6-2)} \\ & \text{Design vector} \quad \bar{x}_i^{(L)} \leq \bar{x}_i \leq \bar{x}_i^{(U)} & i = 1, \dots, n & \text{(6-3)} \end{cases}$$

Where $\bar{x} = [x_1, \dots, x_n]^T$ is defined as the vector containing *n* design variables which are subjected to the constraints reported in Eq. (6-3). There are *B* objective functions $\bar{f}(\bar{x}) = [f_1(\bar{x}), \dots, f_B(\bar{x})]^T$, subjected to two sets of

constraints: $\bar{g}(\bar{x}) = [g_1(\bar{x}), \dots, g_J(\bar{x})]^T$ and $\bar{h}(\bar{x}) = [h_1(\bar{x}), \dots, h_K(\bar{x})]^T$. The aim of the optimisation is to minimize the objective functions: $\bar{f}(\bar{x})$ (Eq. (6-1)), and in the same time satisfy the constraints defined in Eq. (6-2) and (6-3) for the set of design variables chosen.

Assume that $\mathbf{x_1}$ and $\mathbf{x_2}$ are two different design vector of the mathematical set defined by Eq. (6-3); $\mathbf{x_1}$ will be better than $\mathbf{x_2}$ if $f(\mathbf{x_1}) < f(\mathbf{x_2})$, the minimisation of the function is the aim. In this case, there is one optimum solution, since there is only one function. Opposed to this evaluation, there is the multi – objective optimisation, where there are more objective functions to evaluate and the aim is to minimize all of them. Hence, as reported in Eq. (6-1), the minimisation of $f_1(\bar{x}), \dots, f_B(\bar{x})$, with $B>2$, does not provide a unique solution, but different solutions. One design vector $\boldsymbol{x_1}$, is said to be dominated by another solution $\boldsymbol{x_2}$, if $f(\mathbf{x_2})$ is at least small as $f(\mathbf{x_1})$ which means: $\bar{f}_b(\bar{x}_2) \leq \bar{f}_b(\bar{x}_1)\ \forall b = 1, \dots, B$ and $\bar{f}_b(\bar{x}_2) < \bar{f}_b(\bar{x}_1)$ for at least one of the objectives $b \in \{1, \dots, B\}$. If does not exist any design vector $\boldsymbol{x_2}$, with the characteristics just defined, $\boldsymbol{x_1}$ is said to be non dominated. A non dominated design vector can "be Pareto optimal if it is not dominated by another solution in the feasible solution space" [42]. Joining all the Pareto optimal solutions give a curve called Pareto optimal front. It represents the boundary beyond which, any improvements of the overall performance can be reached only by the degradation of any of the objective functions. However, the solution of the system represented by Eq. (6-1), (6-2), and (6-3), not always give the optimum practical solution. Hence, a Decision Maker (DM) is required in order to find one best solution among the set of Pareto – optimal found. Thus, the multi – objective optimisation require both an optimiser and a decision maker in order to find the final solution. The Decision Makers present in literature are:

- *A posteriori*: this DM selects the final solution from the Pareto – optimal set once all the possible alternatives have been presented. This approach presents some difficulties when there is a high number of objective functions.

- *A priori*: this DM selects the final solution from the Pareto – optimal set by giving a numerical value to the objective functions and picking up the minimum. This approach presents the same drawbacks of the composite objective method, since it is moving a multi – objective to a single objective, hence the scalar constant gives pre – conceptions.
- *Interaction methods*: this DM is a compromise between the previous two models, since it chooses the best solution at some intervals during the process. In addition, it presents higher level of confidence upon the final solution [43].

As example, it is reported in Figure 6-1 the optimisation search pattern and the Pareto optimal set for the blade of an axial compressor [5].

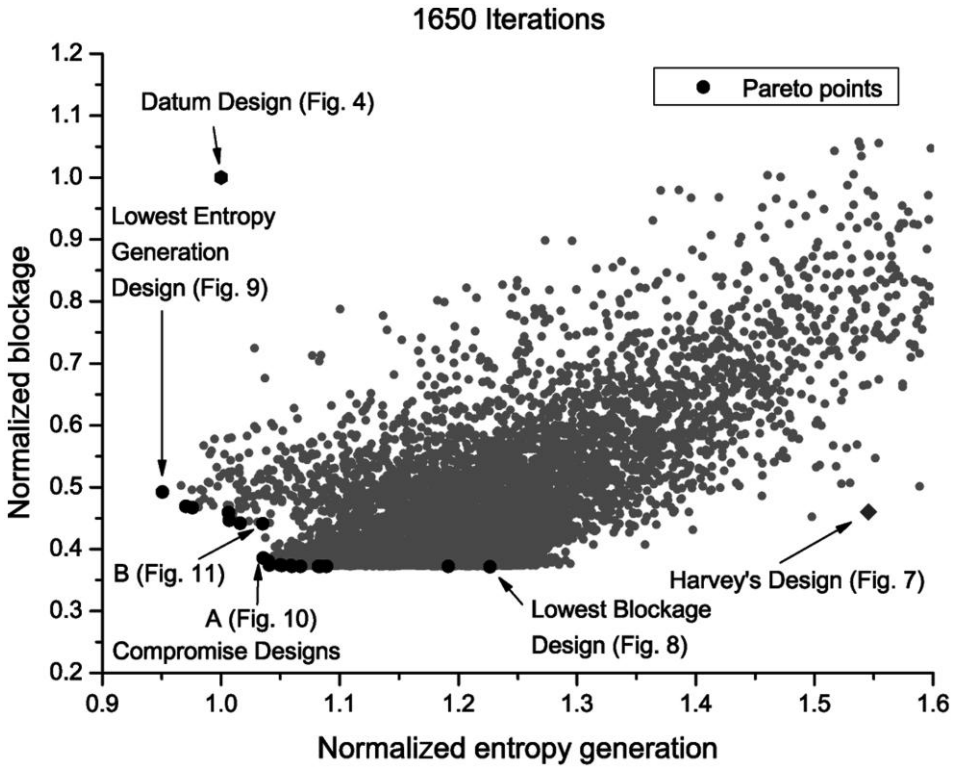


Figure 6-1 Optimisation search pattern and the Pareto – front [5].

There are different heuristic methods that can be adopted for the simulation: genetic algorithm simulated annealing and tabu search. The latter has been chosen to perform the optimisation in this work, and will be described in the next paragraph.

6.3 Multi Objective Tabu Search (MOTS)

The Tabu Search (TS) is a local search method, which uses an adaptive memory which avoids exploring unproductive design space, hence works in an efficient way. There are two types of memory in the TS which are: explicit and attributive. The former stores the Pareto – optimal solutions which have been found along the simulation, in addition it has an extension in order to record some attractive but unexplored design vectors in the design space. The latter is used during the search process for guiding. It records information about the solution characteristics which changes when moving from one solution to the new explored.

The TS local search method is based on the Hooke and Jeeves local search algorithm [4]. At each iteration, the optimiser creates $2 \cdot n_{var}$ points using a step δ_i (which can be hardcoded) in summing $x_i + \delta_i$ and subtracting$x_i - \delta_i$. If the constraints, Eq. (6-3), are not violated, the objective functions are evaluated for the new points created and the best among them is used as base for the new search. The points are classified in order to optimally choose the new points to use, as base for performing the evaluation. They are classified according to the domination respect to the current point, those are the categories:

Table 6-1 Point classification [5]

Objective functions	Type A	Type B	Type C
Decreased	Yes	Yes	No
Unchanged	Possible	Possible	Possible
Increased	No	Yes	Yes

The type A are all the non – dominating points, whereas the type C are the dominating points. If there is an only type A, it is automatically chosen as base point for the new search, otherwise the type A points are sorted in order to remove, eventually, dominated points, and in the set one point is randomly

selected among the remaining. If no type A point is present, the procedure above explained is applied to the type B points. Finally, if there are not type B points, a dominated solution is chosen randomly among the type C points. Moreover, if there type A points that have not been previously selected as next base point, they are stored in an Intensification Memory, which is called later in the search procedure. The TS algorithm is made up of three principal stages, each of them utilizes a peculiar memory.

- The short term memory (STM) records the recently visited points and creates a list, called Tabu list. It indicates design vectors that will be not revisited. The number of the recent points visited depends upon the number of objective functions and design variables.
- The medium term memory (MTM) records the optimal or near – optimal points which are later used for performing the Search Intensification (SI) strategy. As the words suggest, this process is used when it is required a more detailed analysis of a region previously explored. The intensification process starts after a certain number of local search iterations have been found unfruitful. The region that seemed attractive is explored and in a multi – objective optimisation the SI selects casually a new design vector from the Pareto front.
- The long term memory stores the over visited areas while the SI makes the intensification in one region of the domain space. The storage is made by the Search Diversification (SD), which divides the domain in sub domains (the number depends upon the objective functions and design variables). During the SD process the search is moved randomly to a sub domain which has been not extensively visited. This choice is based upon the visited index, which is equal to the number of visited points in a sub domain.

Figure 6-2 presents a concise example of the different memories within the TABU search algorithm.

In case of many unfruitful design vectors, the algorithm performs the Step Size Reduction (SSR). This strategy makes an intensive search next to the current optimal solution. The functions size of a step each design variable and the

search is applied on a random point from the MTM. When the SSR is performed, the counter of the other strategies (SD, SI and SSR) is reset to zero.

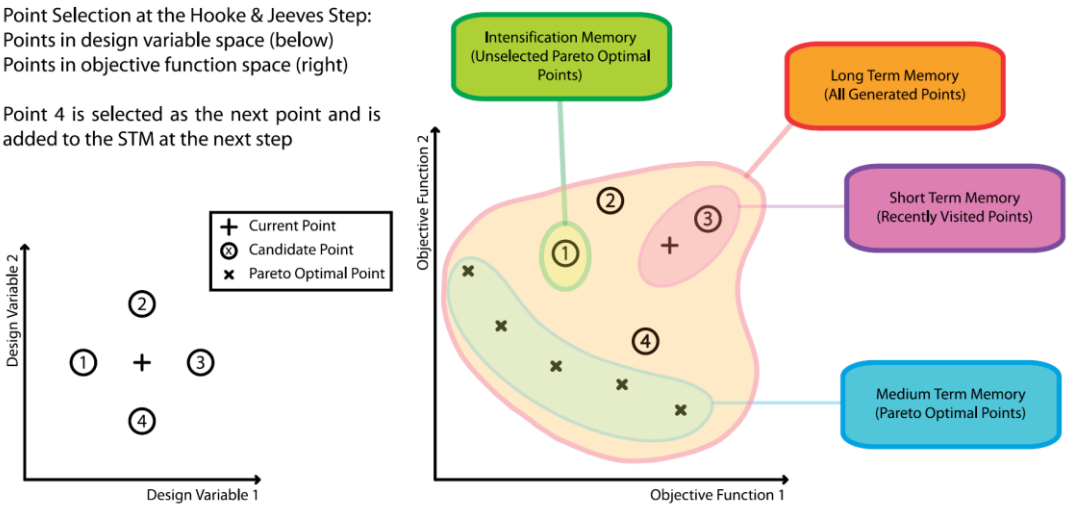


Figure 6-2 Point selection for the Hooke &Jeves move and Tabu search memories [4]

The Tabu Search algorithm has been chosen among others multi objective optimisers because it presents an intelligent approach to the optimisation process. Many works have been successfully carried out with this method [5; 44]. Finally, it presented equal or better performance in comparison with its "competitor" [4; 39].

6.4 Software description

The MOTS presented in the previous paragraph, has been implemented as C++ code, [4], adapting a single – objective Tabu Search developed by Connor and Tilley and cited in [4]. A flow diagram of the MOTS software is reported in Figure 6-3. From the latter it is possible to note how the algorithm is based on the i_{local} which is the local counter and is reset each time the MTM is successfully updated. The different schemes previously introduced (SI, SD, SSR) are called when i_{local} reaches values, specified by the user, which imply the selection of a random point from the MTM. In this case, the optimisation

process is ended since a stopping criteria has been found. The latter can be the maximum number of evaluations or the time available for the task.

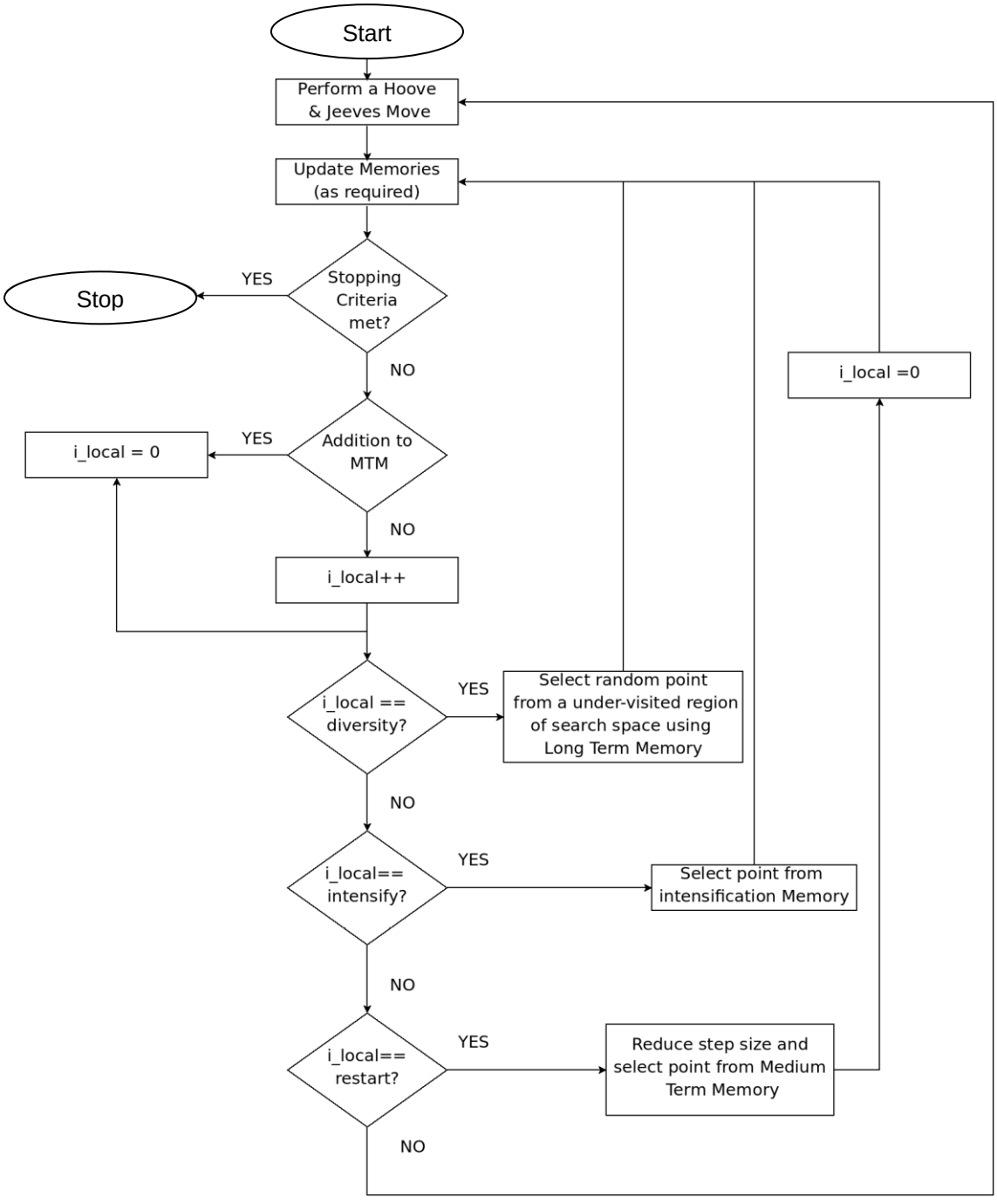


Figure 6-3 Flow diagram of the Multi Objective Tabu Search algorithm [4]

The constraints are hardcoded, and are quite straightforward to implement, in particular they belong to two different types:

- *Binary constraint*: is performed a check if the new geometry created by the code (the new point of the design space) violates any design vector constraints (Eq. (6-3)). If it does, is subsequently stored in the Tabu list so that it will be not picked up again. In this case, no evaluation is performed, since the results would be of no use;
- *Penalty functions:* after an evaluation of the design vector has been made, the quantification of the output is used in order to suggest to the code where it has to investigate to find new design vectors. It is possible to define these constraints as "soft" which means that even if the values are far from the suggested values, it is always possible to investigate those areas. This is impossible in the case of binary constraints violation.

The MOTS software is built in master and slave configuration, ready to be used in a parallel architecture. The master is the core of the optimizer whereas the slaves run the evaluations of the objective functions for a particular design vector which has been given by the master. The MOTS software has been used in cohesion with the LBM code in the previous chapter introduced, hence a LBM interface has been implemented within the MOTS code, which allows the two codes to exchange information simultaneously. The LBM code works in a parallel environment as well, but, as well as for the MOTS software, both can run from a single processor to an high number of processors. This feature makes this combination of tools really versatile and multi level parallelisation.

7 CHAPTER

Software facilities

This chapter is involved in the description of the optimisation cycle, the exchange of the information between the MOTS and the LBM software. At the end is presented the high performance computer used for the simulations.

7.1 Optimisation cycle

The MOTS and the LBM codes need to exchange information, which are the design vector and the objective functions, in order to make working the optimisation cycle. Figure 7-1 Illustrates how the optimisation cycle works, and highlights its phases.

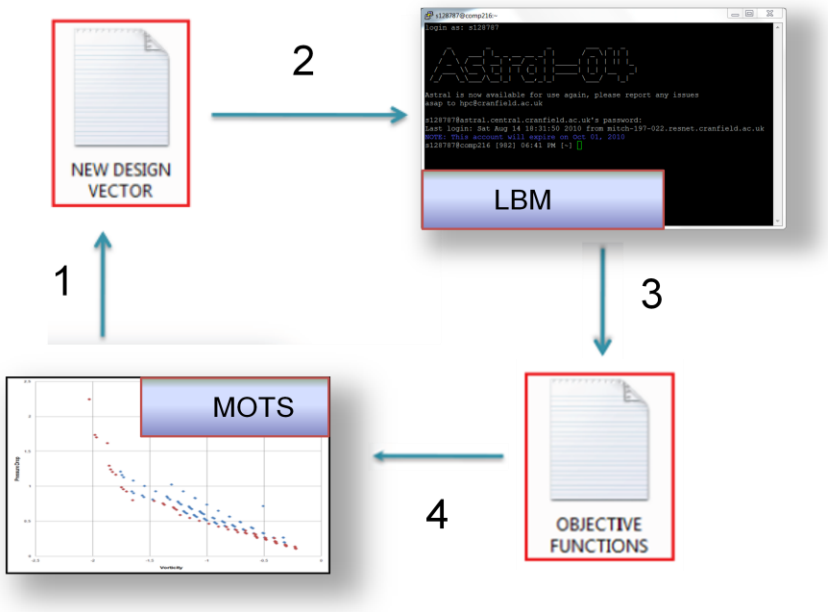


Figure 7-1 Flow chart of the optimisation cycle

The first step is made by the MOTS (Multi Objective Tabu Search) software; the master starts and initialises the solution to the Design Point, which corresponds to the condition of the paper [2]. Each slave controls the design vector and, if it does not violate the constraint here defined, it writes a *.txt* file which contains:

the design vector and the ID number of the processor of the slave itself, according to the rules presented in the previous chapter. This file is called "*input//proc//*", where *//proc//* is the ID number. This suffix is important when more than one slave is used, because in that case each slave produces different inputs, so it is required to differentiate them.

After the design vector is created, the slave calls the LBM code. This is made by accessing in the shell, using the *system()* command. The LBM code is called so that it that takes as input a file whose name is exactly the one created by the slave: "*input//proc//*".

The parameters are read from the LBM code which can run the simulation. Since it is not important to create the velocity, pressure and vorticity fields, but only the objective functions are required, the subroutines *write_tecplot*, *write_vorticity, assemble_tecplot*, and *assemble_vortex,* are not called. At the end of the simulation the vorticity in the microreactor and the pressure drop are written in a file called: "*output//proc//*" analogously to the previous defined in the first step. This file is a *.txt* and contains the objective functions value.

The last step is made by the slave again, which reads the file in output from the LBM code and compare the objective function values with the values of the "*design point*". So, the mots_slave sends this information to the mots_master, which decides which region in the design space needs to be explored, according the three strategies explained in the previous chapter.

7.1.1 Design Vector

The design vector is written by the mots_slave in the *input//proc//* file and must respect the constraints (better defined in the next paragraph). In Figure 7-2, are reported the geometrical variables identification:

- *r_small:* the radius of the inner tube, equal to the radius of the holes of the multi holed baffle plate
- *s:* is the space between the centre of the inner tube and the centre of one of the holes of the baffle plate

- *Reynolds:* define the Reynolds number:

$$Re = \frac{u\_avg \cdot D\_big}{\nu}$$

which directly affects the *u_avg* since the *D_big* is constant and the same is for ν, the kinematic viscosity. According to the equations presented in Eq. (5-3), this implies a direct change for both the *u_xa* and *u_xs* for respectively the annulus and inner tube.

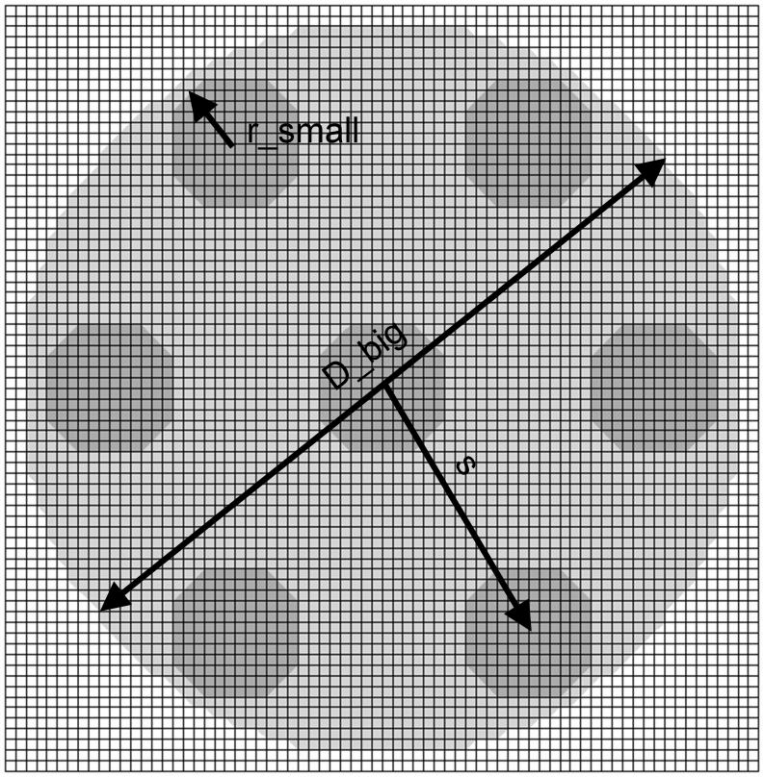


Figure 7-2 Sketch of the multi - holed baffle plate and indication of the geometrical design variables

One example of the *input//proc//* file content is here below reported

Table 7-1 Example of Design Vector (input file)

r_small	s	Reynolds	Processor
7.02867	15.7791	16.2722	9468

Where the header in bold has been written in order to understand the meaning of the numbers. The file is stored in the path where the MOTS software is running:

/lustre/scratch/s128787/mots_test.half/input%s

Where %s indicates the number of processor in C++ code, analogously to the *//proc//* of Fortran.

7.1.2 Constraints handling

The mots_master defines the range and the initial step of variability for each design vector, so it can define the analysis area, they are summarised in Table 7-2.

Table 7-2 Constraints variability and step definition

Parameter	Min	Max	Step
r_small	5	11	1
S	11	29	1
Reynolds	1	200	10

The actual operation of feasibility of the design vector is made by the mots_slave, because the value of *s* are related to the *r_small*, hence is defined a range of values for *s* as *r_small* changes:

$$s_{min} = 2 \cdot r\_small + 1$$
$$s_{max} = D\_big - r\_small - 1 \quad (7\text{-}1)$$

The geometrical constraints are made in order to keep the outer diameter (*D_big*) constant and avoid unphysical configurations, e.g. the borders of the circles touch the outer diameter. If one design configuration do not respect the constraints defined in Table 7-2 and Eq. (7-1), it is discarded and the design vector is not created.

7.1.3 Objective function

The LBM code is called from C++ from the shell, by the command:

system(mpirun -np 4 /lustre/scratch/s128787/mots_test.half/a.out < input%s)

which runs the LBM multiprocessor code (MPI) before compiled and calls *a.out*, using *input%s* as input file to the code. The LBM code produces the output file:

open(process,file='/lustre/scratch/s128787/mots_test.half/output'//proc//'')

and here writes the *vor* and *pr_diff*,

Table 7-3 Example of Objective Function (output file)

vorticity	Pressure difference
147.07	3.0102

The file is stored in the path where the MOTS software is running: This file is picked up from the mots_slave, and compared with the values of the "design point" (*vor_0, pr_diff_0*). Since the minimisation of the objective function is the aim, in case one objective needs to be maximised, a minus is put in front of it, so the comparison is made as:

$$new\_cost[0] = -vor/vor\_0$$
$$new\_cost[1] = -pr\_diff/pr\_diff\_0 \quad (7\text{-}2)$$

The vector *new_cost* is sent to the mots_master, which decides where to move for the next evaluation and provides a new design vector.

7.2 Process Parallelisation

One simulation of the LBM code requires almost four hours, with one processor, A reasonable amount of evaluations for the MOTS optimisation could be 2000, which implies more than 11 months for having good results. In order to reduce this time, the MOTS – LBM integrated system has been designed both in parallel environment.

- The MOTS software is provided already available for the parallelisation, by choosing the number of slave which perform the optimisation
- The LBM presents domain decomposition, in the sense that the domain is divided in sub – domains (equal to the number of processors required) and each of those is executed from different processor. This requires to implement in the code the MPI functions to send/receive information between different processors

This allows having a multi – level parallelisation tool, and reduces drastically the running time of the optimisation. In Figure 7-3 is reported the parallelisation scheme of the optimisation tool and in the next paragraph is well explained how the jobs are executed.

7.3 Astral High Performance computing

The computer used for performing the simulation is sited in Cranfield University, and it is a high performance cluster. The grid is made up of 214 nodes; each of them presents two Intel 5160 Xeon 3 Ghz dual core processors which imply a total of 4 processors per node and 856 processors for the cluster. Each node presents 8 GB of shared memory and each dual core cpu's has 4 MB of cache shared between the cores [44]. Since each node presents 4 processors, the LBM has been chosen to be limited to 4 processors whereas the number of slaves has been set to 3 as explained in section 6.3 and it depends on the number of design variables and of objective functions.

After the compiling phase for the LBM and MOTS software, the link between the codes is made via the *Makefile* and after the optimisation job is sent to the grid. The latter operation is made by the submission of the SCRIPT_ASTRAL_MPI (Appendix A.3). In Figure 7-3 the parallelisation scheme of the optimisation as it runs on Astral is depicted. As the job starts, the mots_master begins on one node and then it calls the three slaves, two of them in different nodes, and one on the same node. The reason of this decision is that the computational resources used by the mots_master are low if compared to the slave or to the LBM. Hence, reserving one node only for the master would imply and underloaded node and a waste of the system resources. Once the slaves are running, they will call the LBM code (giving the input file) and the second level of parallelisation is reached. The LBM will provide at the end of the simulation an output file to the slave which will give this information to the master that will generate the new design vector so that the slave process can start again.

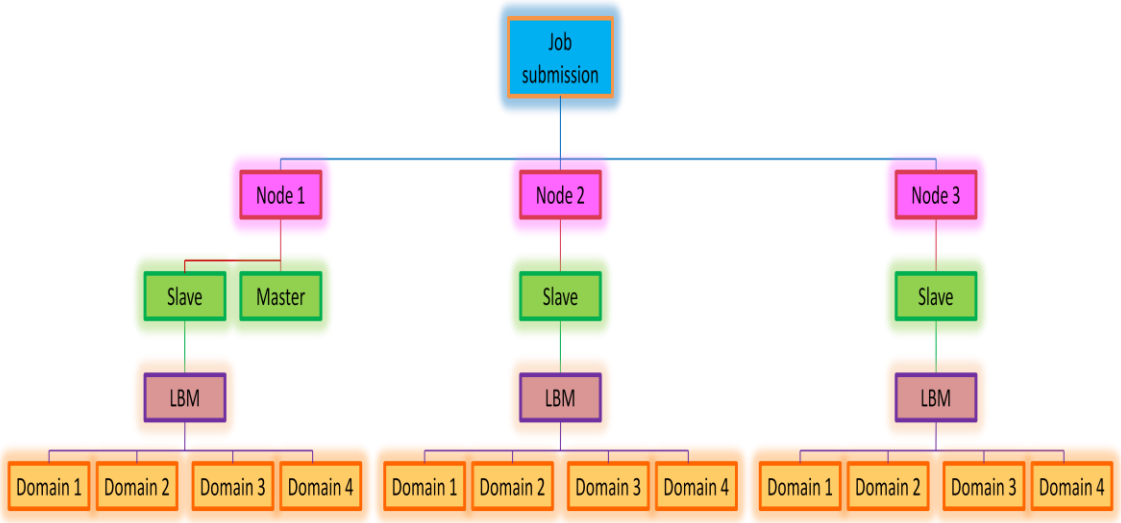


Figure 7-3 Parallelisation scheme of the optimisation tool (MOTS-LBM) for Astral

8 CHAPTER

Results and discussion

This chapter presents the results obtained from the preliminary multi objective optimisation. The first section gives the Pareto front and defines the parameters used in the simulation. Subsequently, three design vectors are presented: for highest vorticity, lowest pressure drop and a compromise solution. The last section presents some overall comparison between the different solutions.

8.1 The Pareto Front

The optimisation process is computationally expensive but the tool is designed to produce output solutions as the evaluations are made. Despite the fact that optimisation is still running, it is possible to draw the Pareto front of the simulation at 75 MOTS evaluations and 276 LBM evaluations. In addition, it is also possible to know how many times the optimiser provided a design vector which was not accepted by the mots_slave. It emerged that almost 25% of the design vectors did not satisfy the condition of spacing: Eq. (7-1), and hence are not feasible.

In Figure 8-1 the Pareto front and the search pattern of the optimiser is presented. The x – axis is shows the minimisation of the vorticity and the y – axis shows the minimisation of the pressure drop. The starting point (-1,1) corresponds to the design point. The more the design is far from this point the more promising is the design. According to the location of the point picked from Figure 8-1, the design will enhance the vorticity or will decrease the pressure drop. As expected, the behaviour of the objective functions is proportional, which means increase in vorticity will tend to increase the pressure drop. Plenty of design vectors that enhances the mixing have been found but with more than the double pressure drop of the design vector. Similarly, design vectors which drastically reduce the pressure drop have been found, but with low vorticity, which implies not a good mixing. The next paragraphs will show these configurations.

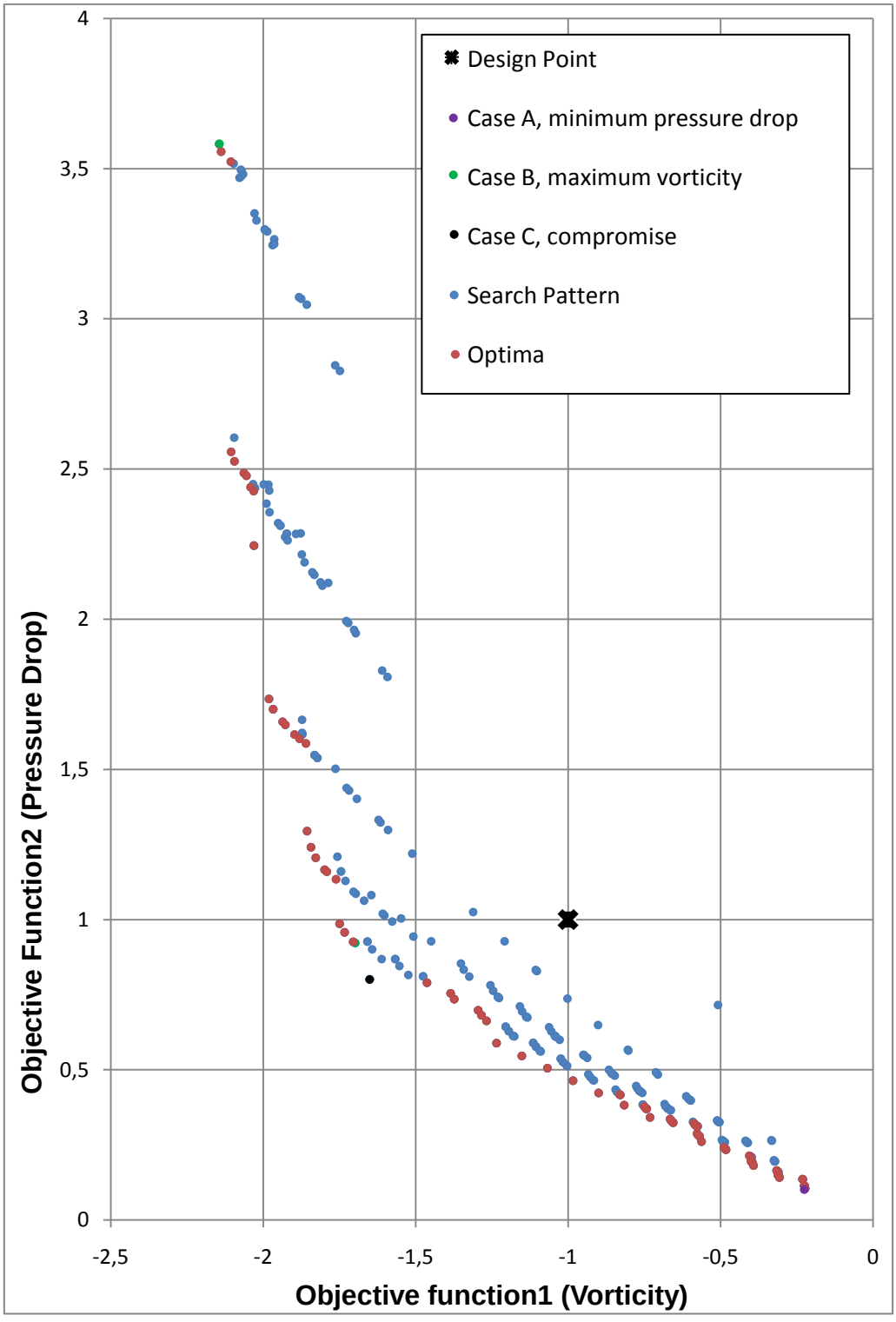


Figure 8-1 Search pattern and Pareto front

8.2 Case A

The case A is for a microreactor with minimum pressure drop. This is achievable with slow flow, which means low Reynolds number, big radius and the spacing as small as possible. The expectations are confirmed by the design reported in Figure 8-2, with Reynolds number equal to 25.73, which is the lowest at the moment used by the optimiser. The spacing is equal to the minimum allowed, whereas the radius might increase by a unit, and arrive to 11. Obviously the design presents less vorticity than the design point vector, but this decrease is almost linearly proportional to the pressure drop. It is interesting to note that there are lot of dominated solutions next to the design picked, which means that eventually the optimiser could move further.

8.3 Case B

The case B is for a microreactor with maximum vorticity. This is achievable with fast flow, since more vortical structures can be made. The better way to achieve such design is by having high Reynolds number, small radius and the spacing as large as possible. The expectations are confirmed in the design reported in Figure 8-3, with Reynolds number equal to 195.7, which is very close to the maximum set at 200. The small radius is 6, and it is next to the minimum, set to 5, whereas the spacing is in the middle between the maximum and the minimum allowed. The latter feature is predictable, since in this manner two vortex structures area created, one between the inner tube and the annulus circles, the second in the annulus. The latter vortex is not responsible of any enhancement of the mixing between the two flows, because it is mostly composed by the fluid coming out from the baffle plate and the inner flow does not participate. The pressure drop in this design is more when compared to the design point, almost three times whereas the vorticity is roughly doubled. The design vector is in a zone that has just been started to be investigated, but has almost reached the limit.

8.4 Case C

The case C is for a microreactor that is a compromise between the above two cases. This case is very new and hence is very interesting. The Reynolds number is equal to the case B, the radius is higher than the case A and the spacing is an average between the previous cases. Hence, instead of having all the parameters which are an average between the extreme cases, the compromise presents a high Reynolds number which is, counterbalanced by the increase of the radius. Figure 8-4 reports the multi – holed baffle geometry.

8.5 Discussion

Despite the relatively low number of iterations, the optimisation produces valuable results and hence it can be used for complex in-depth researches. The two extreme cases, in fact, showed design vectors as expected. It has been noted that the spacing values (21 to 24) does not much affect the three different cases. On the contrary, high values of Reynolds number and small radius results in higher pressure drop and vorticity.

Figure 8-5 and Figure 8-6 represent the velocity contours in the three cases. Since the range of the three cases are very different, it was required to plot them in three different ranges as shown in Figure 8-5. However, it could be seen that the compromise case does not present vortex structure, despite the high Reynolds number. This is due to the high radius which makes the microreactor as an open channel and seems to have no multi holed baffle plate. One interesting feature is that the velocity in the Case B after the vortical structures is almost equal to the velocity of the compromise. This is illustrated in Figure 8-6. Though cases B and C have the same Reynolds number and different radius, but this characteristics brings to the conclusion that are not required small holes in the baffle for obtaining high values of velocity, avoiding the drawbacks in pressure drop of such a design.

Figure 8-7 and Figure 8-8 represent the pressure contours of the three cases. In order to better understand the values throughout the reactor, the range are

different for the three cases, though the values are comparable. It should be noted that case A and case C present a almost equal range, which implies, though the Reynolds number of case C is higher than case A, the high value of radius in case C is able to minimise the pressure drop along the reactor by not over-accelerating the flow locally. The latter is almost concentrated in the baffle plate as shown in **Errore. L'origine riferimento non è stata trovata.**, because there is a sudden acceleration of flow that results in high pressure drop. This is well shown in **Errore. L'origine riferimento non è stata trovata.**, where the relative pressure distributions along the channel are compared. Apart from Case A which presents high radius and low Reynolds number, all the others have a consistent pressure drop along the baffle plate. In the compromise case, the pressure drop is well revealed, since the pressure decrease almost linearly along the microreactor. In fact, according Darcy equation high velocity implies high pressure drop.

Figure 8-9 and Figure 8-10 show the vorticity contours, in which the wide range of difference among the three cases did not allow to plot them consistently with the same scale. The higher vorticity is in the vicinity of the baffle and it is related to the high change in velocity here experienced by the flow. However, for case B and case C, the velocity profile reaches the Poiseuille flow late compared to the case A. Thus high vorticity are experienced for longer distances after the baffle. The compromise, as explained before, behaves almost like an open channel. However the high Reynolds number allows having a high vorticity despite the small change in velocity due to the baffle. This is well shown in Figure 8-14, where the integration of the vorticity on the surface perpendicular to the axial direction along the channel for the three different cases is presented. The Figure 8-15 illustrates a comparison of the three different cases and the design point. It shows that, despite the similar shape of the circulation, they present completely different values. In particular Case A and Case B have one order of magnitude of difference. For saving computational resources, the vorticity can be evaluated within the first half of the microreactor since the most of the vorticity is concentrated in this zone.

Figure 8-11 presents the velocity contours and the streamlines. The latter are a family of curves that are instantaneously tangent to the velocity vector of the flow, hence shows the direction of the particle of fluid in time. These are important in the detection of the vortical structures. As expected, the case A does not present any vortical structures and so is the case C, for the reasons abundantly above explained. Whereas, the case B, has many vortical structures, two small vortices next to the inner tube outlet and a huge vortical structure where the flow from the inner tube reaches the flow from the annulus. The former penetrates in the microreactor for some lattice units after the baffle and then impacts with the high velocity flow from the annulus. This combination creates a vortex which has a width nearly equal to half of the diameter of the microreactor.

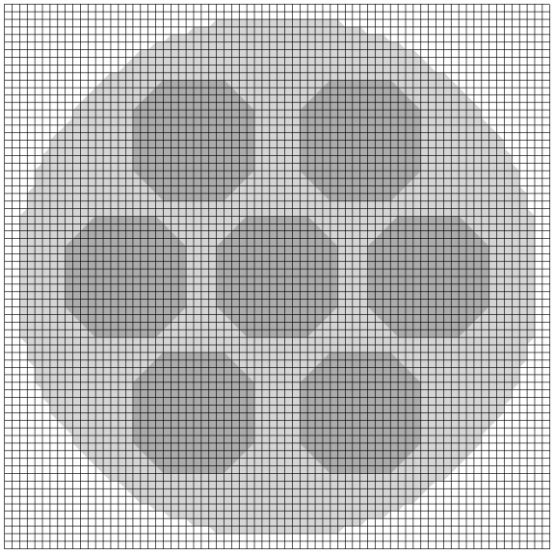


Figure 8-2 Case A

Lowest pressure drop

r_small:9.7

spacing:21

Reynolds: 25.73

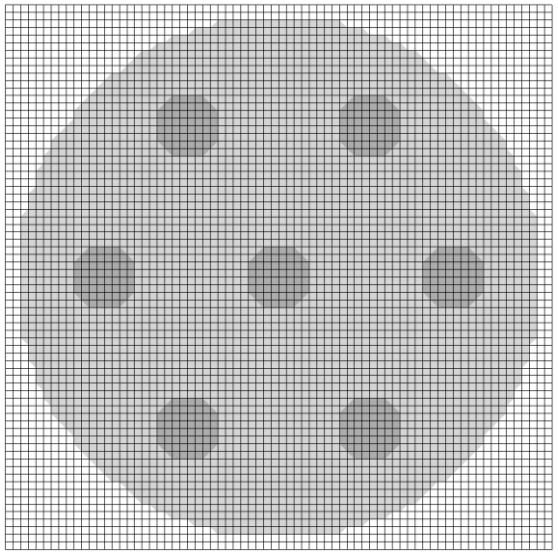


Figure 8-3 Case B

Highest vorticity

r_small:5.7

spacing:24

Reynolds: 195.7

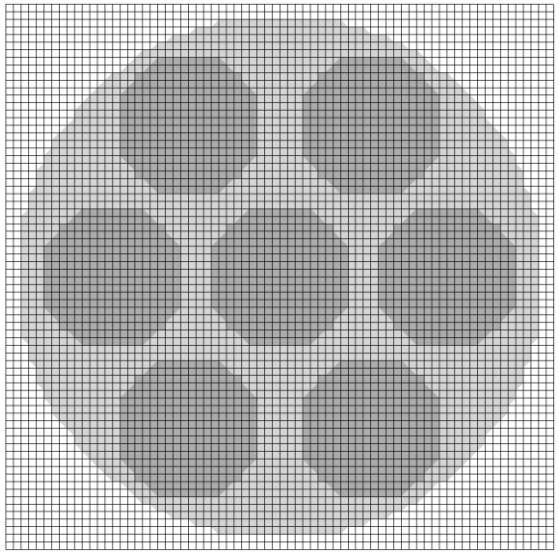


Figure 8-4 Case C

Compromise

r_small:11

spacing:23

Reynolds: 195.7

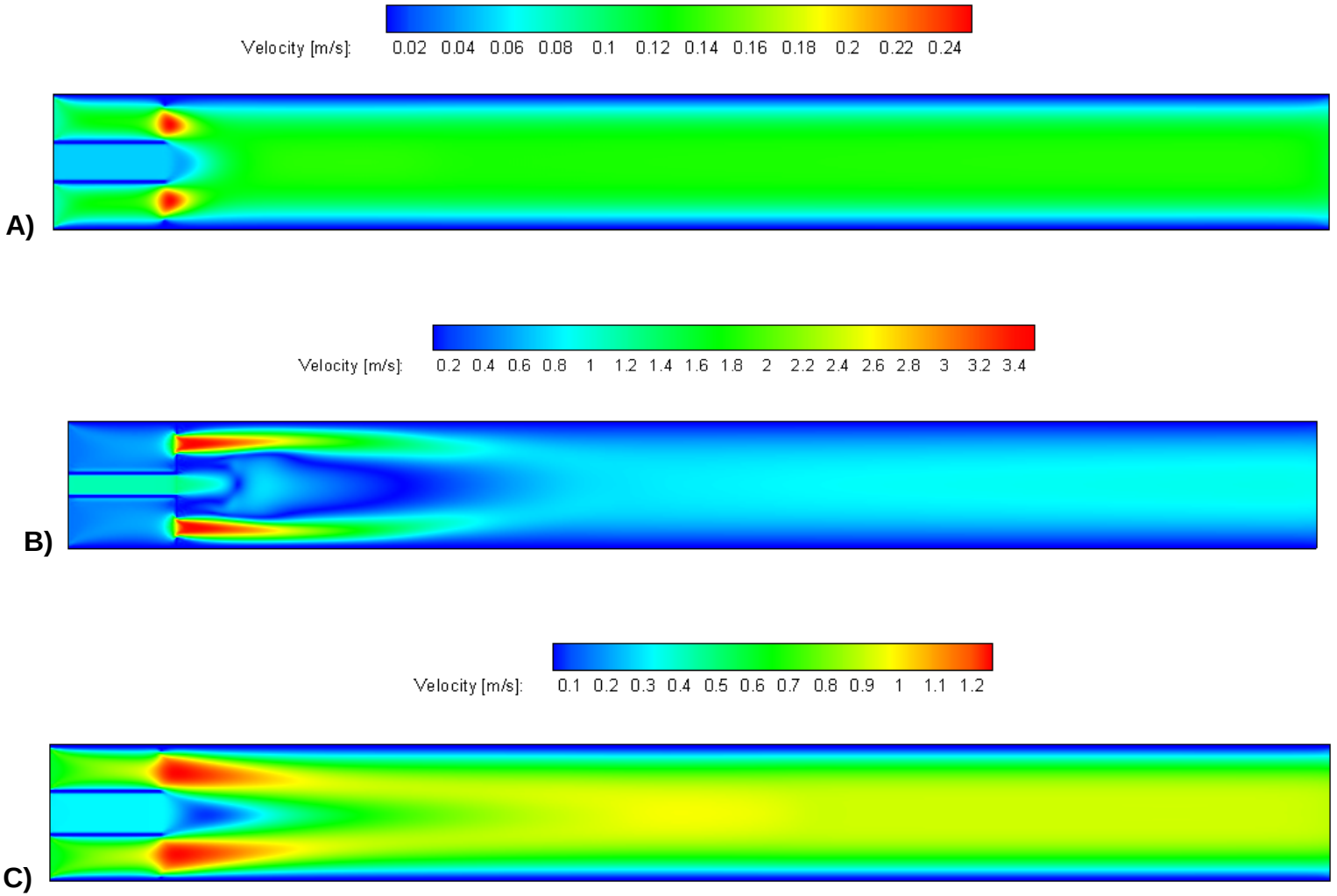


Figure 8-5 Velocity contours in [m/s] in the mid plane along the microreactor for the three different cases

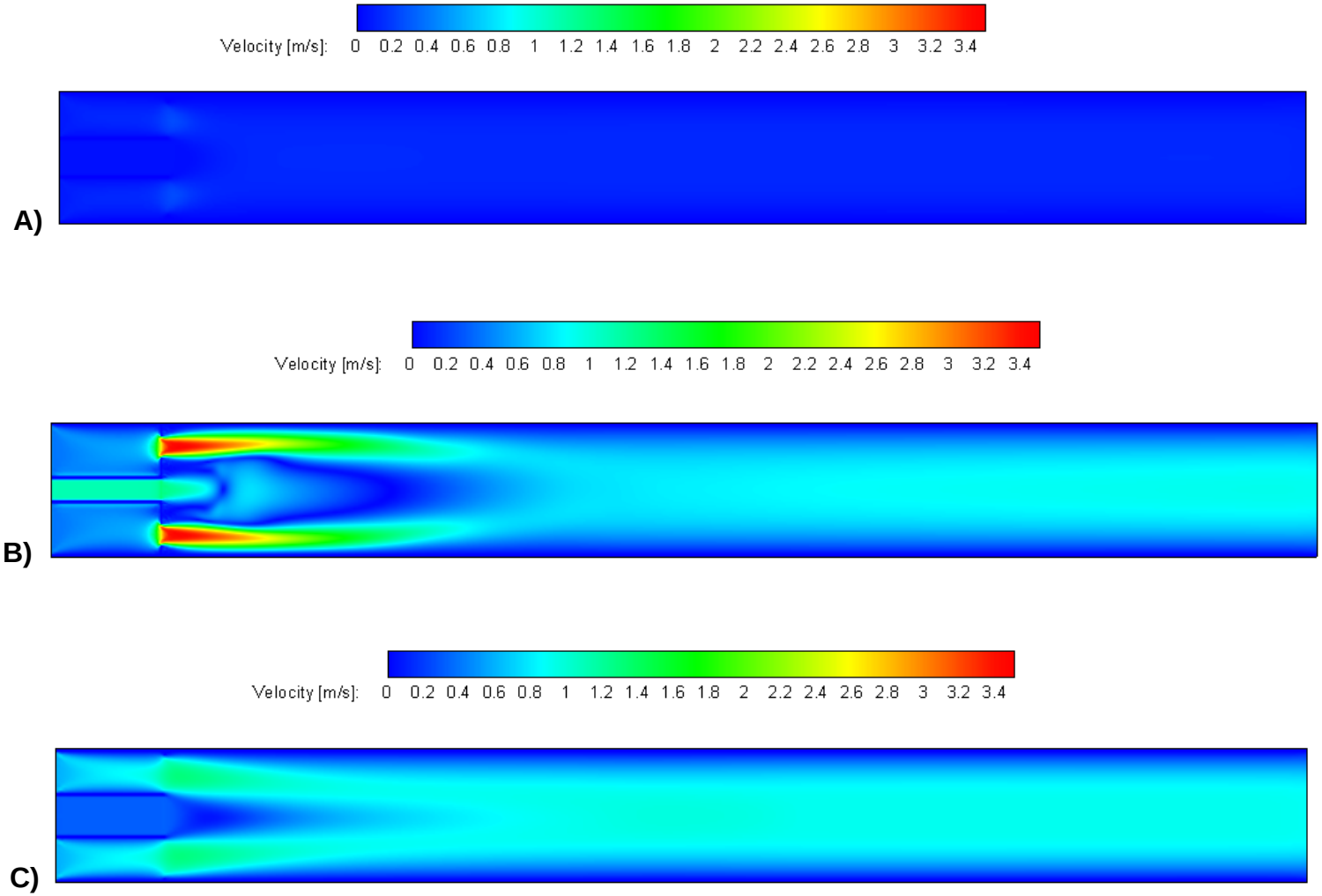


Figure 8-6 Velocity contours in [m/s] in the mid plane along the microreactor for the three different cases (same scale)

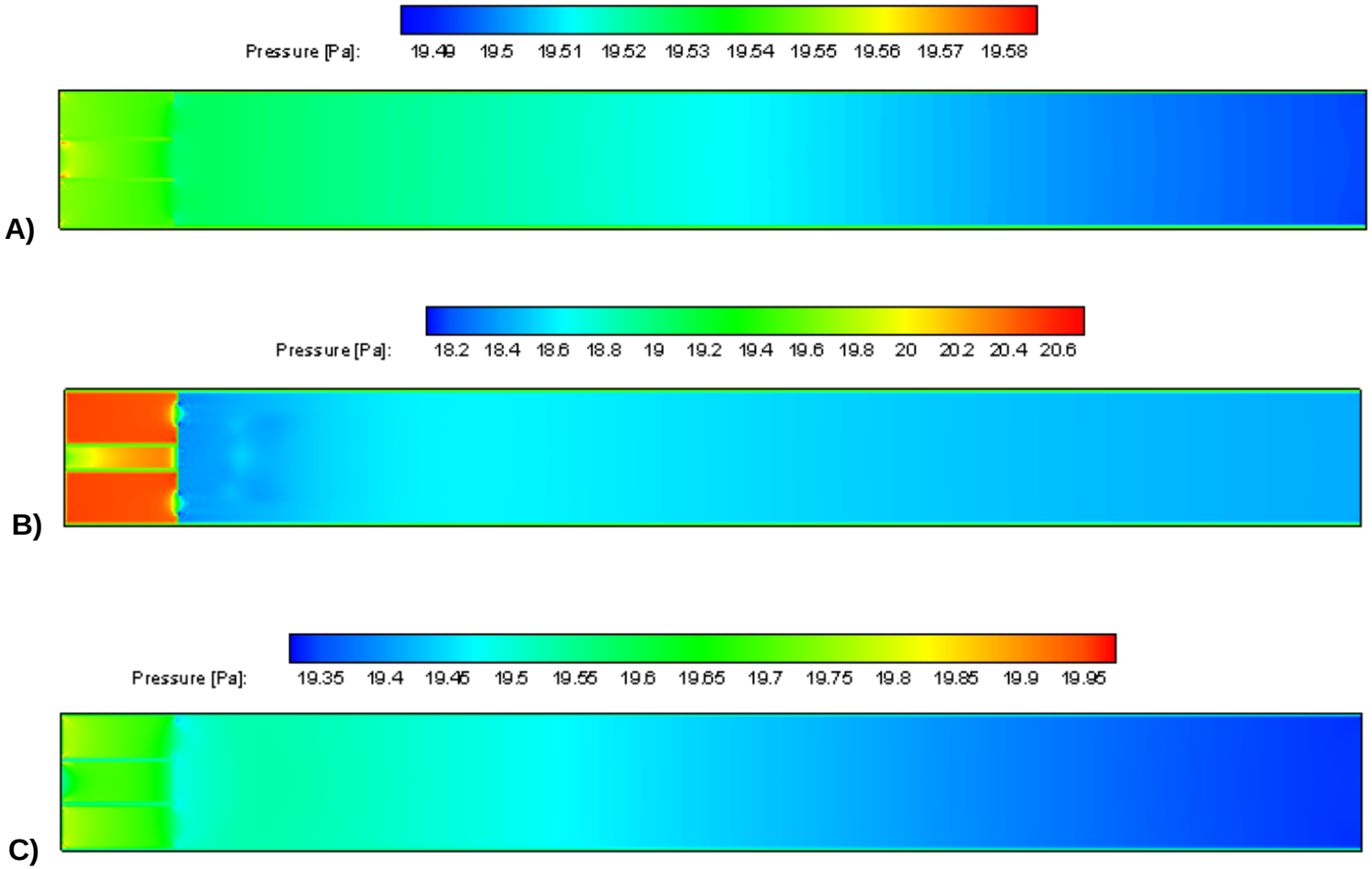


Figure 8-7 Relative pressure contours in [Pa] in the mid plane along the microreactor for the three different cases (Eq. (4-16)); (p_{ref}=101325 [Pa])

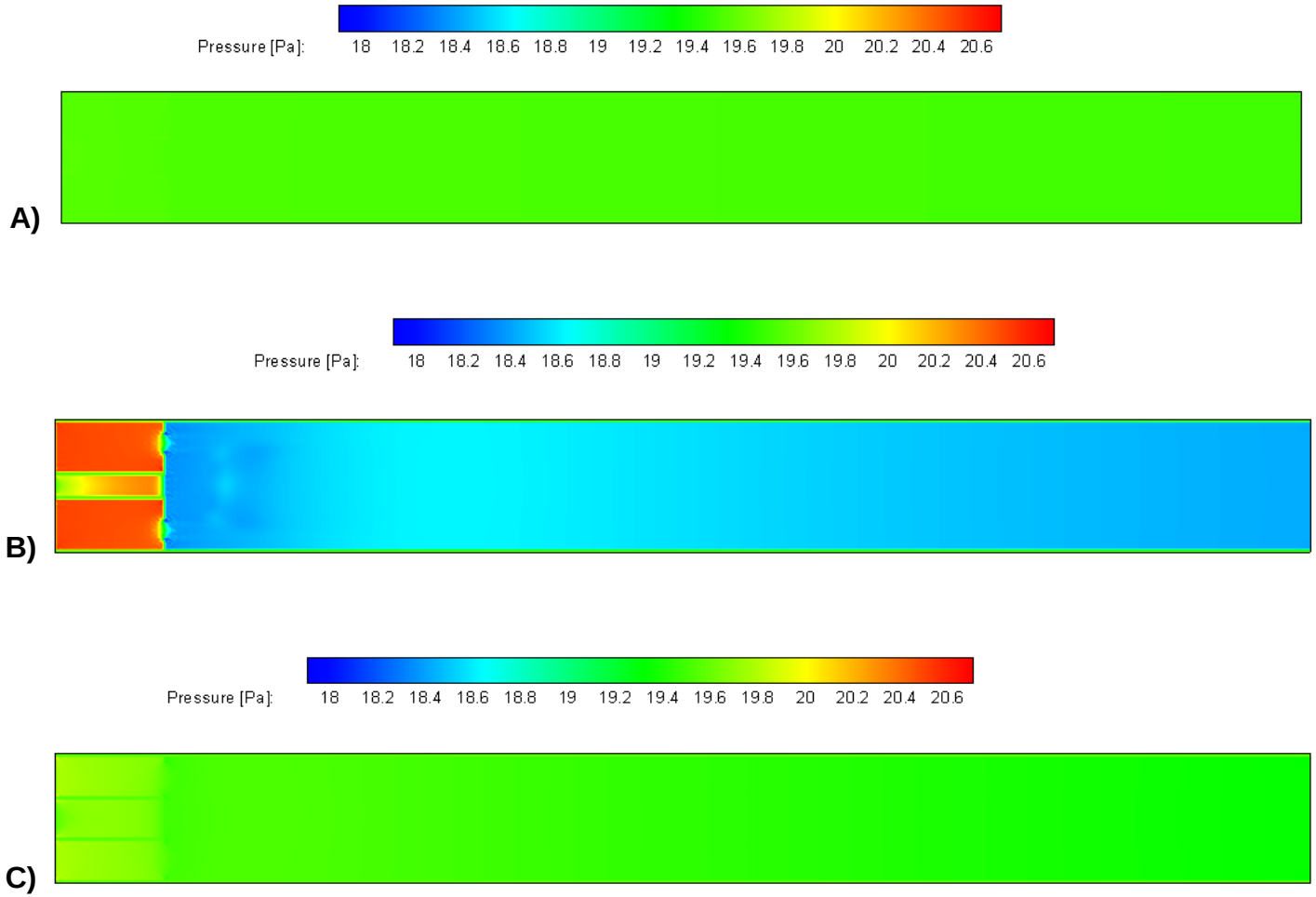


Figure 8-8 Relative pressure contours in [Pa] in the mid plane along the microreactor for the three different cases (same scale) (Eq. (4-16)); (p_{ref}=101325 [Pa])

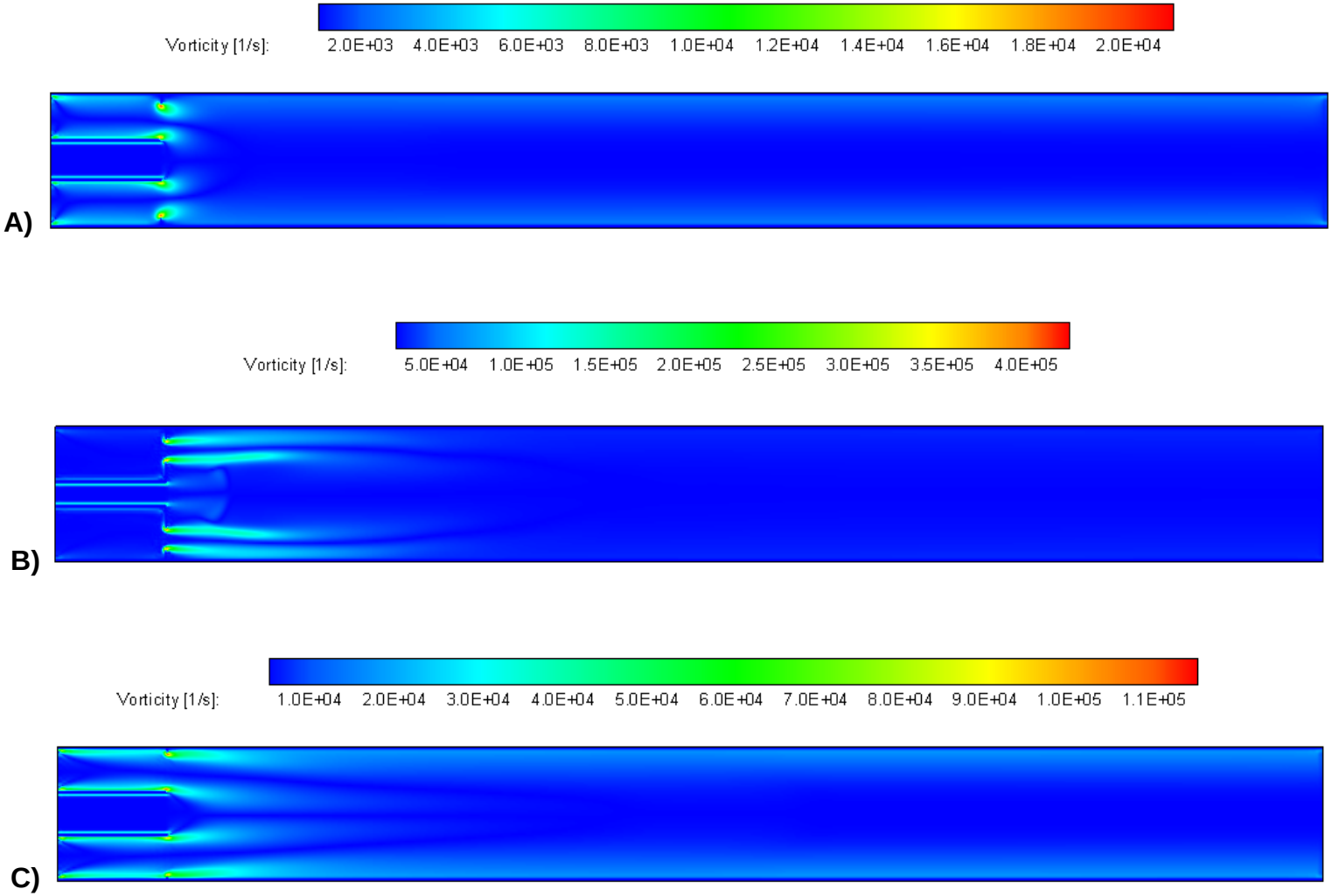


Figure 8-9 Vorticity contours in [1/s] in the mid plane along the microreactor for the three different cases

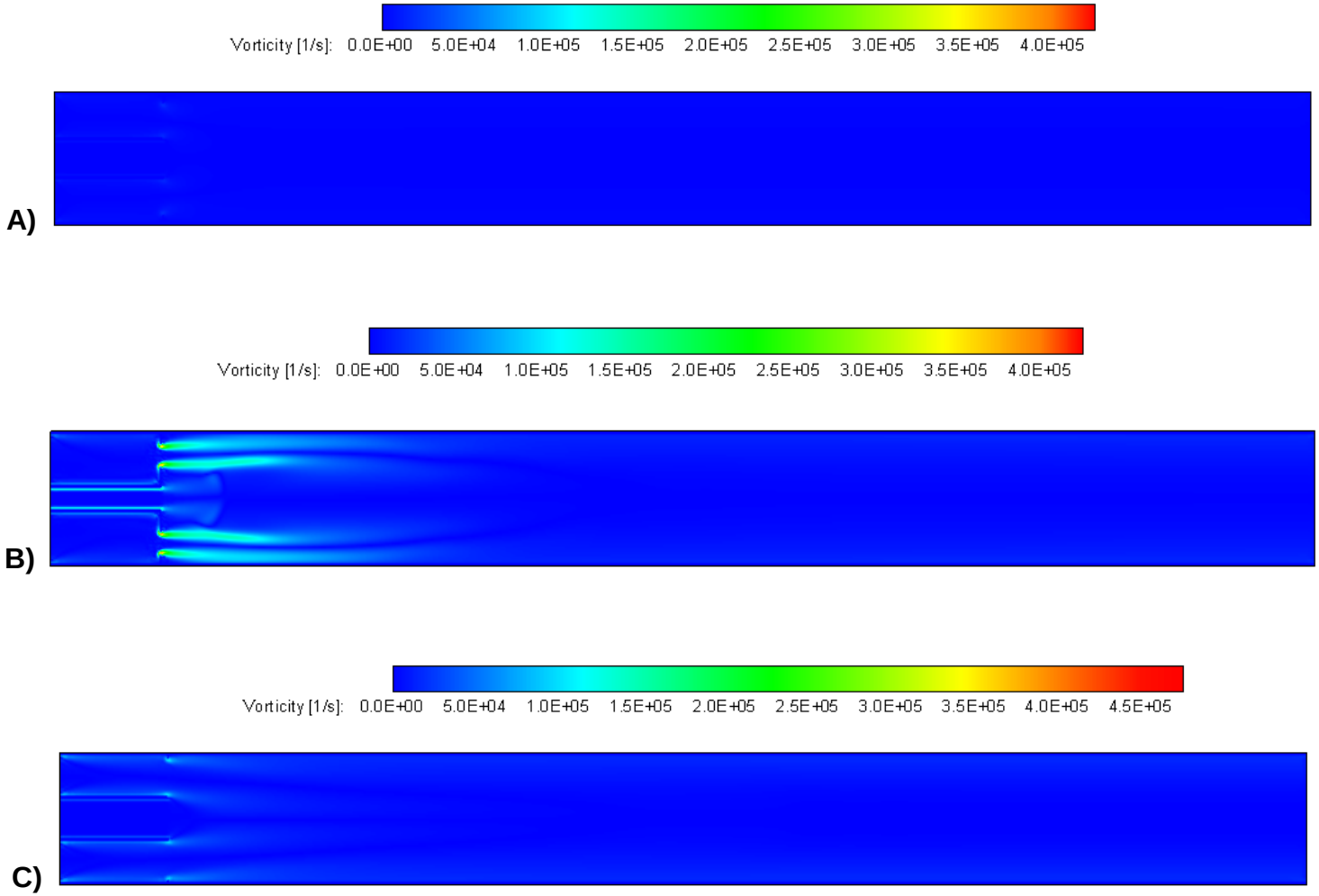


Figure 8-10 Vorticity contours in [1/s] in the mid plane along the microreactor for the three different cases (same scale)

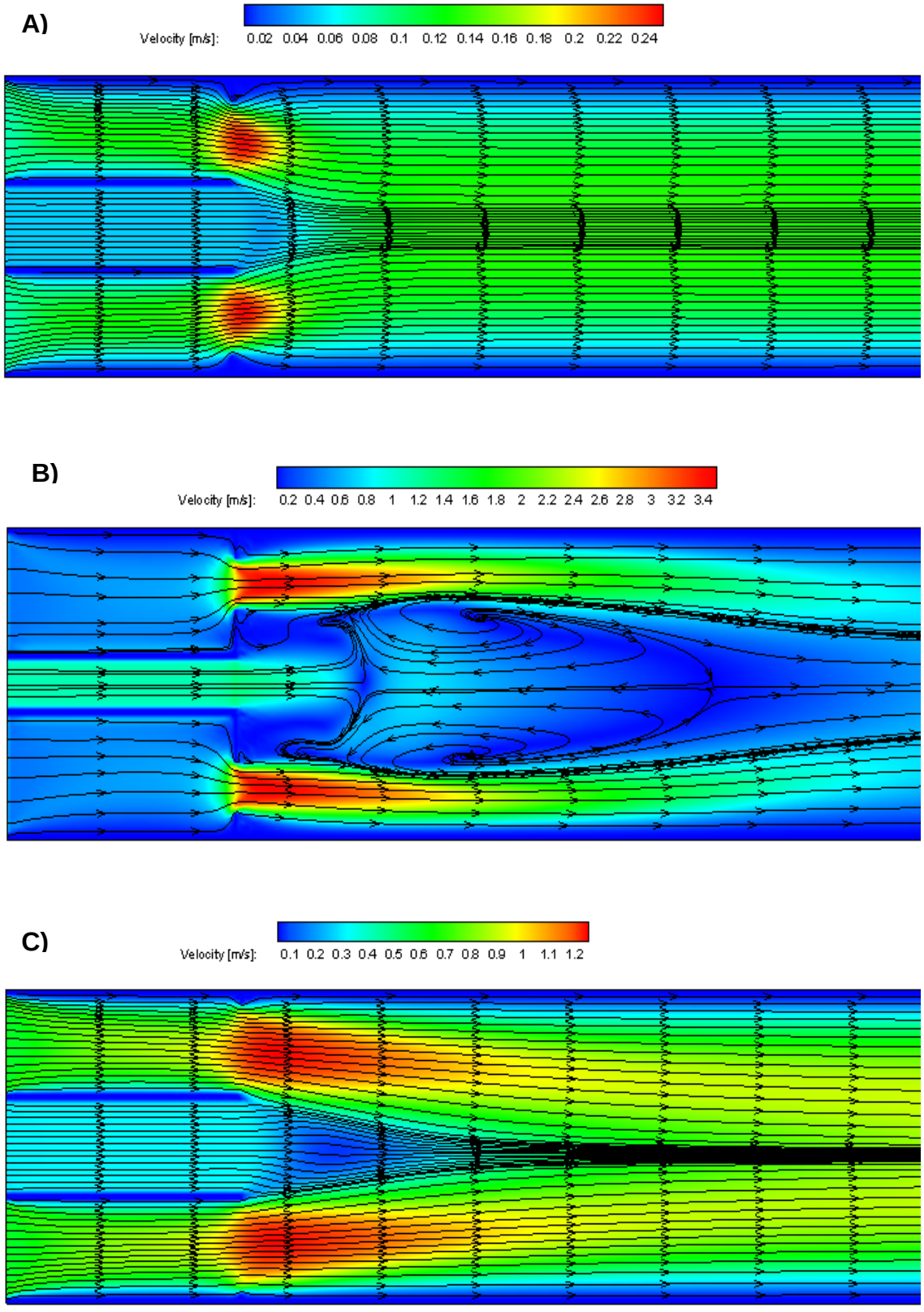


Figure 8-11 Streamlines and velocity contours in [m/s] for the first part of the microreactor

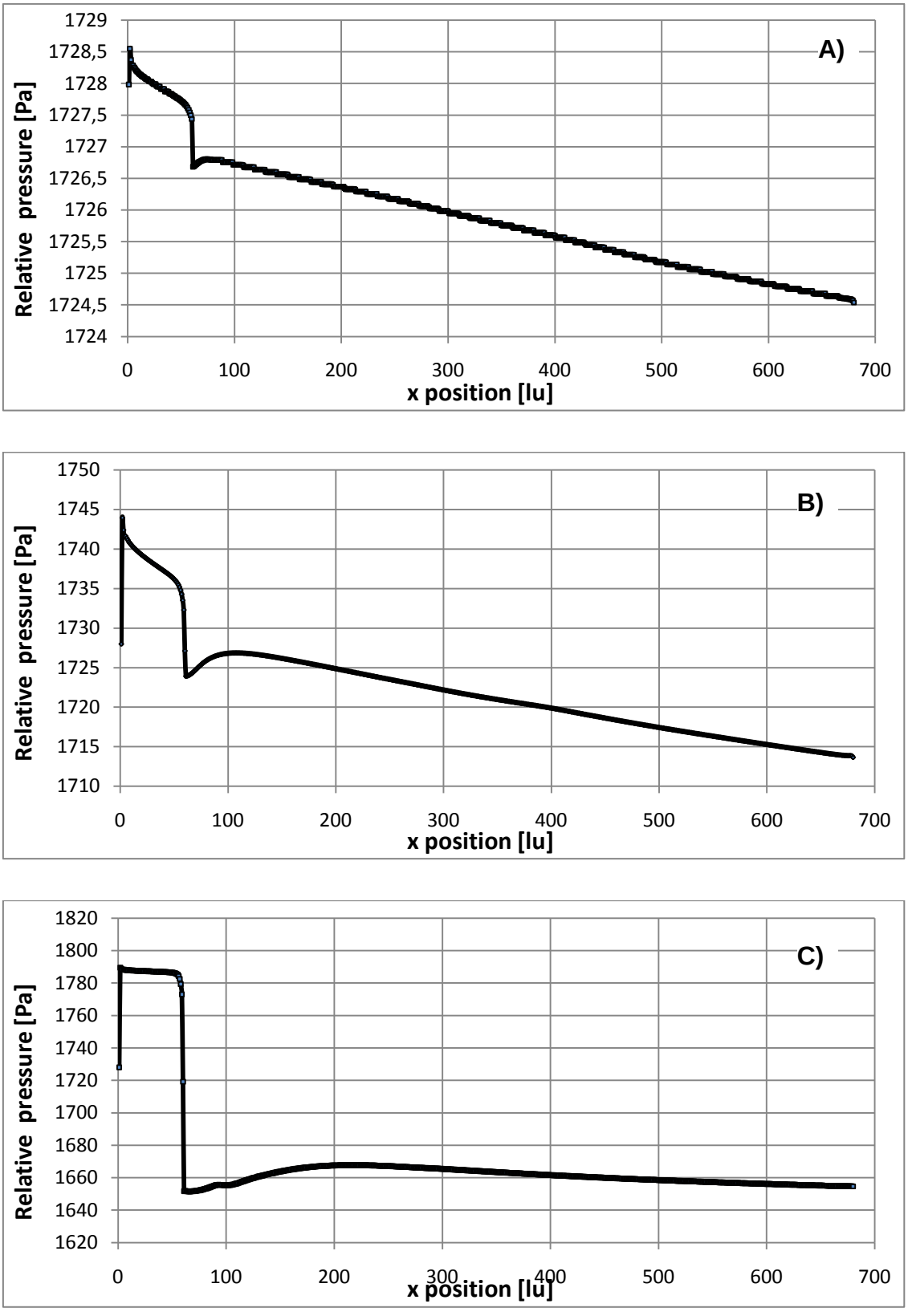


Figure 8-12 Relative pressure distributions along the channel for the three cases (p_{ref}=101325 [Pa])

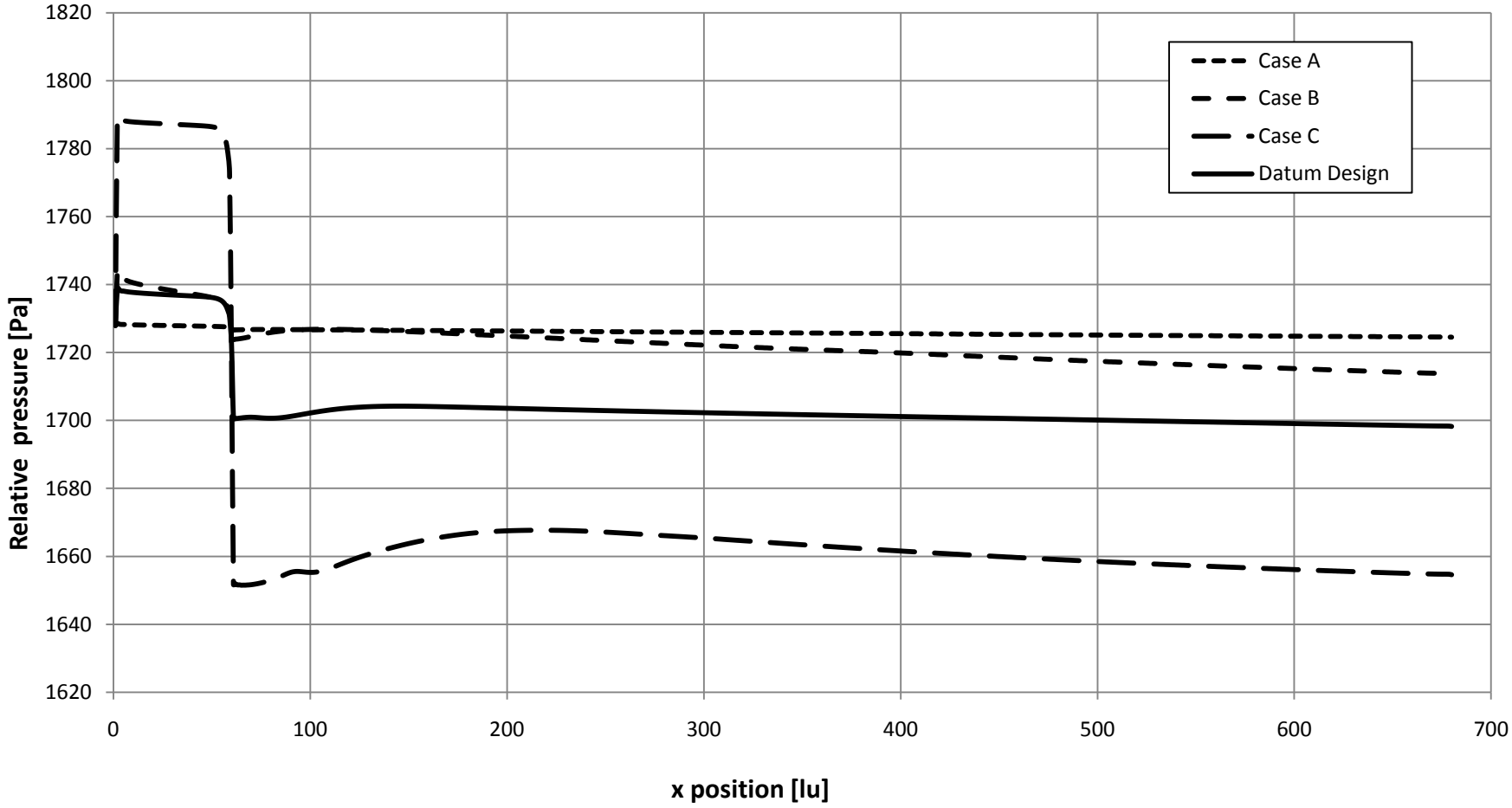


Figure 8-13 Comparison of the relative pressure distributions along the channel for the three cases and the design point (p_{ref}=101325 [Pa])

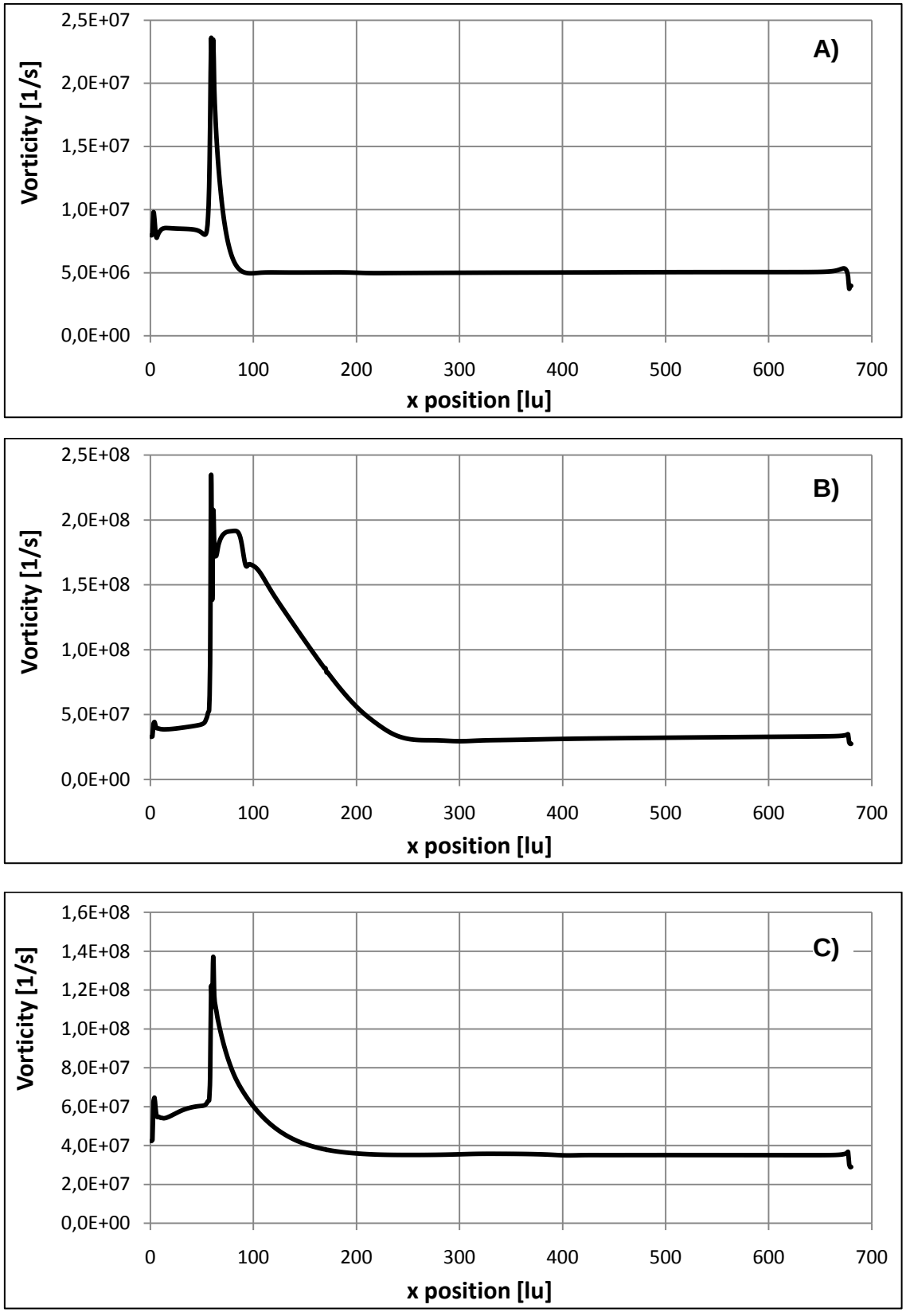


Figure 8-14 Circulation along the channel for the three cases

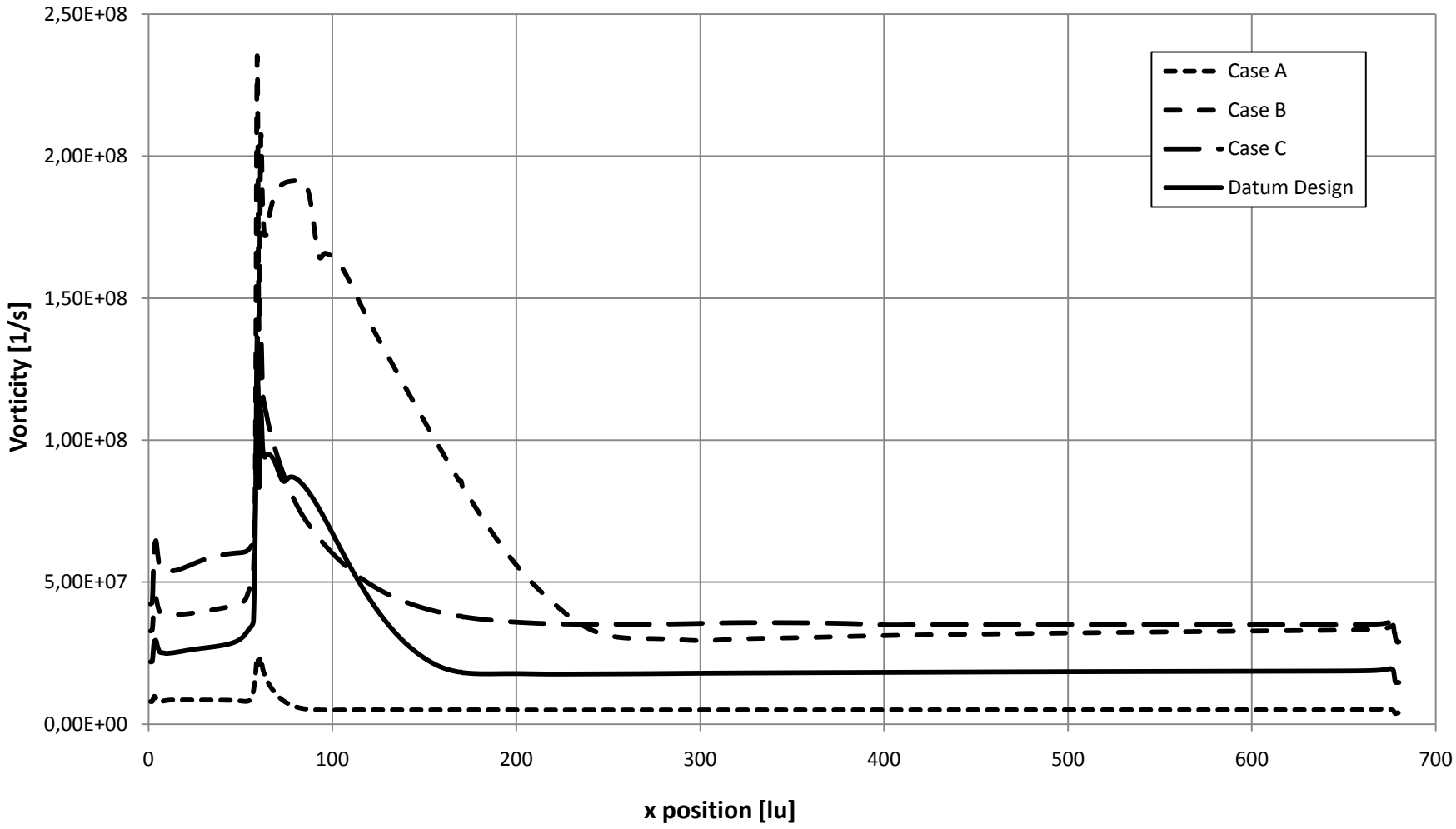


Figure 8-15 Comparison of the circulation along the channel for the three cases and the design point

9 CHAPTER

Conclusions and recommendations

9.1 Conclusions

A lattice Boltzmann code simulating the laminar flow of water in a microreactor with a multi-holed baffle plate has been developed and is validated with experimental and numerical results. The code is later integrated into a multiobjective optimiser, and a comprehensive and robust tool has been developed. The software has been executed in a parallel environment at Cranfield University and the data collected are presented in the work.

The LBM code has been developed by Djenidi [3], and it simulates isothermal non viscous flow based on the lattice Boltzmann method, an innovative and promising fluid flow method that can be an alternative to the conventional Navier Stokes equations. The C++ code developed by Jaeggi [4] is a multi objective optimiser based on local search method. The code is composed of a *master* that calls as many *slaves* as required. Both codes required few changes to accommodate for the current case study. The two codes interface to each other by the use of files. The tool created by the two codes together runs on a multi level parallelisation, because the LBM presents a domain subdivision and three slaves (three processors) are used for the optimisation.

The tool so built has been used for performing an optimisation design of the microreactor reported in [2]. The Reynolds number, the radius and the distance between the holes have been set as design variables. Whereas, the objective functions are pressure drop and vorticity that indicates the mixing quality. Since the two metrics are proportional, increase in the vorticity tends to increase the pressure drop. Hence, the best way to find innovative configurations is by performing an optimisation. The tool made up of MOTS and LBM showed good robustness and efficiency. In fact, despite the low number of iterations performed, a good solution as compromise between the pressure drop and the mixing has been found.

The main task of this work has been accomplished; the results obtained so far are promising for future studies in this field, with more detail and with different geometries.

9.2 Further work

The tool developed and implemented in this work, proves to be robust and reliable. Due to the concentration in the development of the code, the optimisation could only run for a relatively small time in order to give new configurations. In addition, the LBM code itself, can be improved, from a computational point of view and by adding new features. The suggestions for further work are:

- The LBM code needs a better implementation for some part of subroutines, which are called more than once and have been repeated;
- The convergence can be enhanced by changing the redistribute function and relate the initialisation to the design vector. In fact, the cases with low Reynolds number, when the change in velocity was smaller have presented a faster convergence;
- The implementation of a two phase flow in the code, in order to evaluate the importance of the diffusion process within the mixing;
- The code presents already the capability to make calculations with the temperature. This feature has not been used, but it could be interesting to investigate this feature;
- The addition of a new design variable in the optimiser: the diameter of the inner tube. The change of this parameter may give interesting results, because in this manner it could be possible to vary the inner tube velocity, as its influence on creating vortical structure have already been shown.
- Implement new geometries in order to investigate new configurations.

REFERENCES

[1] Ottino, J.M. and Wiggins, S., (2004), *Introduction: mixing in microfluidics*, The Royal Society, London, pp. 923-927

[2] Moghtaderi, B., Shames, I. and Djenidi, L. (2006), "Microfluidic characteristics of a multi-holed baffle plate micro-reactor", *International Journal of Heat and Fluid Flow,* vol. 27, no. 6, pp. 1069-1077.

[3] DJENIDI, L. and MOGHTADERI, B. (2006), "Numerical investigation of laminar mixing in a coaxial microreactor", *Journal of Fluid Mechanics,* vol. 568, no. -1, pp. 223-243.

[4] Jaeggi, D. M., Parks, G. T., Kipouros, T. and Clarkson, P. J. (2008), "The development of a multi-objective Tabu Search algorithm for continuous optimisation problems", *European Journal of Operational Research,* vol. 185, no. 3, pp. 1192-1212.

[5] Kipouros, T., Jaeggi, D. M., Dawes, W. N. and Parks, G. T. (2008), "Biobjective Design Optimization for Axial Compressors Using Tabu Search", *AIAA JOURNAL,* vol. 46, no. 3.

[6] Ali, M.A. and Djenidi, L., (2009), *Lattice Boltzmann Simulation of Pulsed Jet in T-Shaped Micromixer*, pp. 701-711

[7] Nguyen, N. and Wereley, S. T. (2006), *Fundamentals and applications of microfluidics,* 2nd ed, Artech House, Boston, Mass. ; London, pp 357-389.

[8] Ehrfeld, W. (2000), *Microreactors : new technology for modern chemistry,* Wiley-VCH, Weinheim ; Chichester, pp. 41-83.

[9] Ottino, J. M. and Wiggins, S. (2004), "Foundations of chaotic mixing", pp. 937-939.

[10] Succi, S. (2001), *The Lattice Boltzmann Equation for fluid dynamics and beyond,* Clarendon, Oxford, pp. 1-16; pp. 65-73;

[11] Cercignani, C. (1975), *Theory and application of the Boltzmann equation,* Scottish Academic Press; Distributed by Chatto and Windus, Edinburgh; London, pp.1-12.

[12] Sukop, M. C. (2006), *Lattice Boltzmann modeling : an introduction for geoscientists and engineers,* Springer, Berlin ; London, pp. 1-66.

[13] Gurney, R. W. (1949), *Introduction to Statistical Mechanics,* McGraw Hill Book Co, New York.

[14] Luo, L. S. (1993), *Lattice-gas automata and lattice Boltzmann equations for two-dimensional hydrodynamics* (unpublished PhD thesis), Georgia Institute of Technology, Georgia, pp. 1-139

[15] Anderson, J. D. (2006), *Hypersonic and high-temperature gas dynamics,* 2nd ed, American Institute of Aeronautics and Astronautics, Reston, Va, pp. 20-22; pp.449-573.

[16] Viggen, E. M. (2009), *The Lattice Bolzmann Method with Applications in Acoustics*Department of Physics, NTNU, Trondheim, pp. 1-83

[17] Bhatnagar, P. L., Gross, E. P. and Krook, M. (1954), "A Model for Collision Processes in Gases. I. Small Amplitude Processes in Charged and Neutral One-Component Systems", *Physical Review,* vol. 94, no. 3, pp. 511.

[18] Qian et al, Y.H., (1992), *Lattice BGK Models for Navier-Stokes Equation*, pp. 479-484

[19] Wolf-Gladrow, D. A. (2000), *Lattice-gas cellular automata and lattice Boltzmann models : an introduction,* Springer, Berlin ; London, 13-213

[20] Latt, J., Chopard, B., Malaspinas, O., Deville, M. and Michler, A. (2008), "Straight velocity boundaries in the lattice Boltzmann method", *Physical Review E,* vol. 77, no. 5, pp. 056703.

[21] Gallivan, M. A., Noble, D. R., Georgiadis, J. G. and Buckius, R. O. (1997), "AN EVALUATION OF THE BOUNCE-BACK BOUNDARY CONDITION FOR LATTICE BOLTZMANN SIMULATIONS", *International Journal for Numerical Methods in Fluids,* vol. 25, no. 3, pp. 249-263.

[22] Ferziger, J. H. (2002), *Computational methods for fluid dynamics,* 3rd, rev. ed, Springer, Berlin ; London, pp. 181-183 ; pp. 273

[23] Hasan, N., Anwer, S. F. and Sanghi, S. (2005), "On the outflow boundary condition for external incompressible flows: A new approach", *Journal of Computational Physics,* vol. 206, no. 2, pp. 661-683.

[24] AGRAWAL, A., DJENIDI, L. and ANTONIA, R. A. (2005), "Simulation of gas flow in microchannels with a sudden expansion or contraction", *Journal of Fluid Mechanics,* vol. 530, no. -1, pp. 135-144.

[25] DJENIDI, L. (2006), "Lattice-Boltzmann simulation of grid-generated turbulence", *Journal of Fluid Mechanics,* vol. 552, no. -1, pp. 13-35.

[26] DJENIDI, L., (2008), *LBM 3 - D Code*, NewCastle Australia.

[27] Abrahamson, S. and Lonnes, S. (1995), "Uncertainty in calculating vorticity from 2D velocity fields using circulation and least-squares approaches", *Experiments in Fluids,* vol. 20, no. 1, pp. 10-20.

[28] Raffel, M. (2007), *Particle image velocimetry : a practical guide,* 2nd ed, Springer, Heidelberg ; New York, pp. 190-208.

[29] Viggen, E. M. (2010), "The lattice Boltzmann method in acoustics", *33rd Scandinavian Symposium on Physical Acoustics,* 2010, Trondheim, NTNU, pp. 1-5.

[30] Latt, J. (2008), Choice of units in lattice Boltzmann simulations, , LBMethod.org, pp. 1-3.

[31] Krüger, T., (2010), *Lattice units calculations, problem with density*, http://www.lbmethod.org/forum/read.php?3,2685,2711.

[32] DOW (2010)*, Density (1% to 10% Glycerine)*, available at: http://www.dow.com/glycerine/resources/table4_010.htm (accessed 08/07/2010).

[33] DOW (2010)*, Viscosity of Aqueous Gliceryne solutions*, available at: http://www.dow.com/glycerine/resources/table18.htm (accessed 08/07/2010).

[34] Krüger, T., Varnik, F. and Raabe, D. (2009), "Shear stress in lattice Boltzmann simulations", *Physical Review E,* vol. 79, no. 4, pp. 046704.

[35] Latt, J. (2007), *Hydrodynamic Limit of Lattice Boltzmann Equations* (Docteur ès sciences thesis), Université de Genève, Genève, pp. 1-72.

[36] Yuan, P. (2005), *THERMAL LATTICE BOLTZMANN TWO-PHASE FLOW MODEL FOR FLUID DYNAMICS* (Doctor of Philosophy thesis), University of Pittsburgh, Pittsburgh, 1-54.

[37] Hollis, A., Halliday, I. and Car, C. M. (2006), "Enhanced, mass-conserving closure scheme for lattice Boltzmann equation hydrodynamics", *JOURNAL OF PHYSICS A: MATHEMATICAL AND GENERAL,* , no. 39, pp. 10589.

[38] Coello. (CINVESTAV-IPN), (2005), *Twenty Years of Evolutionary Multi-Objective Optimization: A Historical View of the Field* Mexico, pp. 1-2

[39] Ho, S. L., Shiyou Yang, Wong, H. C. and Guangzheng Ni (2003), "A simulated annealing algorithm for multiobjective optimizations of electromagnetic devices", *Magnetics, IEEE Transactions on,* vol. 39, no. 3, pp. 1285-1288.

[40] Zitzler, E. and Deb, K. (2000), "Comparison of multiobjective evolutionary algorithms: Empirical results", *Evolutionary Computation,* vol. 8, pp. 173.

[41] Knowless, J. D. and Come, D. W. (2000), "Approximating the nondominated front using the Pareto archived evolution strategy", *Evolutionary Computation,* vol. 8, pp. 149.

[42] Kulturel-Konak, S., Smith, A. E. and Norman, B. A. (2006), "Multi-objective tabu search using a multinomial probability mass function", *European Journal of Operational Research,* vol. 169, no. 3, pp. 918-931.

[43] Kipouros, T. (2003), Multiobjective Aerodynamic Design Optimization, 1, Cambridge University, Cambridge, pp. 31-52.

[44] Trapani, G. (2009), *Multi-Objective Optimization of 2D High-Lift Airfoil Configurations using Tabu Search* (unpublished Master of Science thesis), Cranfield University, Cranfield, pp. 79-87.

APPENDICES

A.1 Derivation of the Navier – Stokes continuity equation from the LBM dynamics

The fluid dynamics, can be most of the times solve by both the Navier – Stokes equations and the lattice Boltzmann equation. This appendix shows how to derive the continuity equation of the former from the latter.

Consider the Eq. (4-8) which is the LBGK model, and perform a Taylor series expansion in time and space at the second order, using small parameters $\delta_x = \delta_t$:

$$\delta t \frac{\partial f_\alpha}{\partial t} + \partial t e_{\alpha k} + \frac{\partial f_\alpha}{\partial x_k} + \frac{(\delta t)^2}{2}\left[\frac{\partial^2 f_\alpha}{\partial t^2} + 2e_{\alpha k}\frac{\partial^2 f_\alpha}{\partial t \partial x_k} + e_{\alpha k}e_{\alpha n}\frac{\partial^2 f_\alpha}{\partial x_h \partial x_n}\right] + \frac{1}{\tau}\left(f_\alpha - f_\alpha^{(eq)}\right) = 0 \tag{A-1}$$

The Chapman – Enskog expansion is used:

$$\frac{\partial}{\partial t} = \varepsilon\frac{\partial}{\partial t_1} + \varepsilon^2\frac{\partial}{\partial t_2}$$
$$\frac{\partial}{\partial x} = \varepsilon\frac{\partial}{\partial x_1} \tag{A-2}$$

It is a standard multi – scale expansion, and ε is a small parameter ($\varepsilon \ll 1$) which means that the convection time scale, represented by t_1, is much smaller than the diffusion time scale t_2. [36] Hence the expansion of the particle distribution function can be written:

$$f_\alpha = f_\alpha^{(0)} + \varepsilon f_\alpha^{(1)} + \varepsilon^2 f_\alpha^{(2)} + \mathcal{O}(\varepsilon)^3 \tag{A-3}$$

Combining the three equations above defined, the following equations are obtained:

$$\varepsilon^2\left\{\left[\delta t\frac{\partial {f_\alpha}^{(1)}}{\partial t_1} + \frac{\partial {f_\alpha}^{(0)}}{\partial t_2} + e_{\alpha k}\frac{\partial {f_\alpha}^{(0)}}{\partial x_{1k}}\right] + \frac{(\delta t)^2}{2}\left[\frac{\partial^2 {f_\alpha}^{(0)}}{\partial {t_1}^2} + 2e_{ak}\frac{\partial^2 {f_\alpha}^{(0)}}{\partial t_1 \partial x_{1k}} + \right. \tag{A-4}$$

$$+e_{\alpha k}e_{\alpha n}\frac{\partial^2 f_\alpha{}^{(0)}}{\partial x_{1k}\partial x_{1n}}\Bigg]+\frac{1}{\tau}f_\alpha{}^{(2)}\Bigg\}+\varepsilon\left[\delta t\frac{\partial f_\alpha{}^{(0)}}{\partial t_1}+\delta t e_{\alpha k}\frac{\partial f_\alpha{}^{(0)}}{\partial x_{1k}}+\frac{1}{\tau}f_\alpha{}^{(1)}\right]=$$

$$=-\frac{1}{\tau}\left(f_\alpha-f_\alpha^{(eq)}\right)$$

In order to understand which terms are more influent, they can be collected in order of ε:

$\mathcal{O}(\varepsilon^0)$ $$f_\alpha{}^{(0)}-f_\alpha^{(eq)} \tag{A-5}$$

$\mathcal{O}(\varepsilon^1)$ $$f_\alpha{}^{(1)}=-\tau\delta t\left[\frac{\partial f_\alpha{}^{(0)}}{\partial t_1}+e_{\alpha k}\frac{\partial f_\alpha{}^{(0)}}{\partial x_{1k}}\right] \tag{A-6}$$

$\mathcal{O}(\varepsilon^2)$ $$f_\alpha{}^{(2)}=-\tau\delta t\left[\frac{\partial f_\alpha{}^{(1)}}{\partial t_1}+\frac{\partial f_\alpha{}^{(0)}}{\partial t_2}e_{\alpha k}\frac{\partial f_\alpha{}^{(0)}}{\partial x_{1k}}\right]- -\tau\frac{(\delta t)^2}{2}\left[\frac{\partial^2 f_\alpha{}^{(0)}}{\partial t_1}+2e_{\alpha k}\frac{\partial^2 f_\alpha{}^{(0)}}{\partial t_1 x_{1k}}+e_{\alpha k}e_{\alpha n}\frac{\partial^2 f_\alpha{}^{(0)}}{\partial x_{1k}\partial x_{1n}}\right] \tag{A-7}$$

Combining Eq (A-6) and Eq (A-7) it is possible to write:

$\mathcal{O}(\varepsilon^2)$ $$f_\alpha{}^{(2)}=-\tau\delta t\frac{\partial f_\alpha{}^{(0)}}{\partial t_2}+\left(\frac{1}{2}-\tau\right)\delta t\left[\frac{\partial f_\alpha{}^{(1)}}{\partial t_1}+e_{\alpha k}\frac{\partial f_\alpha{}^{(1)}}{\partial x_{1k}}\right] \tag{A-8}$$

Whereas, combining Eq. (A-5) and Eq. (A-3), the particle distribution function is:

$$f_\alpha=f_\alpha^{(eq)}+\varepsilon f_\alpha^{(1)}+\varepsilon^2 f_\alpha^{(2)}+\mathcal{O}(\varepsilon)^3 \tag{A-9}$$

This equilibrium distribution function, as well as the particle distribution function, satisfies the Eq. (4-7), and it leads to the equations:

$$\sum_\alpha f_\alpha^{(k)}=0 \qquad \sum_\alpha \boldsymbol{e}_\alpha f_\alpha^{(k)}=0\ (k=1,2) \tag{A-10}$$

Multiplying by e_{an} all the components of Eq. (A-6) and Eq. (A-8), summing for all values of α and using the constraints defined by the Eq. (4-7) and Eq. (A-10), the subsequent equations are obtained:

$\mathcal{O}(\varepsilon^1)$ $$\frac{\partial\rho}{\partial t_1}+\frac{\partial\rho u_k}{\partial \mathrm{x_{1k}}}=0 \tag{A-11}$$

$$\mathcal{O}(\varepsilon^1) \qquad \frac{\partial \rho u_n}{\partial t_1} + \frac{\partial}{\partial \mathrm{x_{1k}}} \sum_{\alpha} e_{\alpha k} e_{\alpha n} \frac{\partial {f_\alpha}^{(eq)}}{\partial x_{1k}} = 0 \qquad \text{(A-12)}$$

$$\mathcal{O}(\varepsilon^2) \qquad \frac{\partial \rho}{\partial t_2} = 0 \qquad \text{(A-13)}$$

$$\mathcal{O}(\varepsilon^2) \qquad \frac{\partial \rho u_n}{\partial t_1} + \left(1 - \frac{1}{2\tau}\right) \frac{\partial}{\partial \mathrm{x_{1k}}} \sum_{\alpha} e_{\alpha k} e_{\alpha n} \frac{\partial {f_\alpha}^{(eq)}}{\partial x_{1k}} = 0 \qquad \text{(A-14)}$$

The continuity equations of the Navier Stokes equation is recoverable to the second order of ε, combining Eq. (A-12) and Eq. (A-14):

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_k}{\partial \mathrm{x_k}} = 0 \qquad \text{(A-15)}$$

The momentum continuity can be recovered from these equations as well, but its derivation is more complex and it is not beyond the scope of this work. However, for further information, it is suggested to read the [19; 36].

A.2 Derivation of the equilibrium distribution function

The equilibrium distribution function, .Eq. (4-15), is directly derived from the Maxwell – Boltzmann velocity distribution, adopting a statistical mechanics approach. In fact, it represents, as reported in paragraph 4.8, the probability distribution of particles in a gas in equilibrium. This equation, for two dimensional case is [16]:

$$\boldsymbol{f_B}(\boldsymbol{v}) = \left(\frac{m}{2\pi k_B T}\right) exp\left[-\frac{m\boldsymbol{v}^2}{2\pi k_B T}\right] \tag{B-1}$$

And defining the ideal gas law for a control volume:

$$p = \frac{N k_B T}{V} \tag{B-2}$$

And the evaluation of the pressure for an isothermal and ideal gas:

$$\boldsymbol{c_s^2} = \frac{k_B T}{m} \tag{B-3}$$

Where the meaning of the symbols are:

k_B: the Boltzmann constant;
T: the gas temperature;
N: number of molecules;
V: volume of the box
$\boldsymbol{c_s}$: speed of sound in the lattice;
m: mass of the particle

Combining the equations above reported and rewriting $\boldsymbol{v}$ as $\boldsymbol{c_\alpha} - \boldsymbol{u}$, where $\bar{u}$ is the mean velocity and $\boldsymbol{c_\alpha}$ is the deviation from this value:

$$\boldsymbol{f_B}(\boldsymbol{c_\alpha}) \propto exp\left[-\frac{(\boldsymbol{c_\alpha} - \boldsymbol{u})^2}{\boldsymbol{c_s^2}}\right] = exp\left[-\frac{\boldsymbol{c_\alpha}^2}{2\boldsymbol{c_s^2}}\right] exp\left[-\frac{\boldsymbol{u}^2 - 2\boldsymbol{u}\cdot\boldsymbol{c_\alpha}}{2\boldsymbol{c_s^2}}\right] \tag{B-4}$$

The latter equation can be rewritten taking into account the Taylor expansion for $e^x = 1 + x/1! + x^2/2! + x^3/3! + \cdots$, which leads to:

$$exp\left[-\frac{\boldsymbol{u}^2-2\boldsymbol{u}\cdot\boldsymbol{c_\alpha}}{2\boldsymbol{c_s^2}}\right]\approx 1-\frac{\boldsymbol{u}^2-2\boldsymbol{u}\cdot\boldsymbol{c_\alpha}}{2\boldsymbol{c_s^2}}+\frac{(\boldsymbol{u}^2-2\boldsymbol{u}\cdot\boldsymbol{c_\alpha})^2}{8\boldsymbol{c_s^4}} \tag{B-5}$$

The second term of the left side of this equation is equal to:

$$(\boldsymbol{u}^2-2\boldsymbol{u}\cdot\boldsymbol{c_\alpha})^2=\underbrace{\boldsymbol{u^4}}_{\mathcal{O}(\mathrm{u}^4)}-\underbrace{4\boldsymbol{u^2}(\boldsymbol{u}\cdot\boldsymbol{c_\alpha})}_{\boldsymbol{\mathcal{O}}(\mathbf{u^3})}+4(\boldsymbol{u}\cdot\boldsymbol{c_\alpha})^2=4(\boldsymbol{u}\cdot\boldsymbol{c_\alpha})^2+\mathcal{O}(\boldsymbol{u^3}) \tag{B-6}$$

And combining the last two equations:

$$exp\left[-\frac{\boldsymbol{u}^2-2\boldsymbol{u}\cdot\boldsymbol{c_\alpha}}{2\boldsymbol{c_s^2}}\right]\approx 1+\frac{\boldsymbol{u}\cdot\boldsymbol{c_\alpha}}{\boldsymbol{c_s^2}}+\frac{(\boldsymbol{u}\cdot\boldsymbol{c_\alpha})^2}{2\boldsymbol{c_s^4}}-\frac{\boldsymbol{u}}{2\boldsymbol{c_s^2}}+\mathcal{O}(\boldsymbol{u^3}) \tag{B-7}$$

It is possible to show that

$$f_\alpha^{(0)}=Kw_\alpha\left[1+\frac{\boldsymbol{u}\cdot\boldsymbol{c_\alpha}}{\boldsymbol{c_s^2}}+\frac{(\boldsymbol{u}\cdot\boldsymbol{c_\alpha})^2}{2\boldsymbol{c_s^4}}-\frac{\boldsymbol{u}}{2\boldsymbol{c_s^2}}\right] \tag{B-8}$$

Is a suitable choice for the equilibrium distribution function, which means that it obeys to the conditions reported in Eq. (4-7), in addition it will be found the of the value of constant *K*. The coefficient *t*, is called the weight and it has been introduced in the paragraph 4.7.

$$\rho\sum_i f_\alpha^{(0)}=K\sum_i w_\alpha\left[1+\frac{\boldsymbol{u}\cdot\boldsymbol{c_\alpha}}{\boldsymbol{c_s^2}}+\frac{(\boldsymbol{u}\cdot\boldsymbol{c_\alpha})^2}{2\boldsymbol{c_s^4}}-\frac{\boldsymbol{u}}{2\boldsymbol{c_s^2}}\right] \tag{B-9}$$

And writing the dot – products of the previous equation with the index notation:

$$\begin{aligned}\rho &= K\sum_i \mathrm{w}_\alpha\left[1+\frac{u_\alpha c_{i\alpha}}{c_s^2}+\frac{u_\alpha u_\beta c_\alpha c_\beta}{2c_s^4}-\frac{u_\alpha u_\alpha}{2c_s^2}\right]\\ &= K\sum_i w_\alpha+\frac{Ku_\alpha}{c_s^2}+\sum_I w_\alpha c_{i\alpha}+\frac{Ku_\alpha u_\beta}{2c_s^4}\sum_i w_\alpha c_{i\alpha}c_{i\beta}-\frac{Ku_\alpha u_\alpha}{2c_s^2}\sum_i w_\alpha\end{aligned} \tag{B-10}$$

And applying the condition for the lattice isotropy, used in Eq. (4-10), the previous equation is:

$$\rho = K + \frac{Ku_\alpha u_\alpha}{2c_s^2} - \frac{Ku_\alpha u_\alpha}{2c_s^2} = K \tag{B-11}$$

Combining the value of *K* found with Eq.(B-8), brings to

$$f_\alpha^{(0)} = \rho \mathrm{w}_\alpha \left[1 + \frac{\boldsymbol{u} \cdot \boldsymbol{c_\alpha}}{\boldsymbol{c_s^2}} + \frac{(\boldsymbol{u} \cdot \boldsymbol{c_\alpha})^2}{2\boldsymbol{c_s^4}} - \frac{\boldsymbol{u}}{2\boldsymbol{c_s^2}}\right] \tag{B-12}$$

The demonstration that Eq. (B-8) respects the second condition reported in Eq. (4-7) is not here reported, but it is possible to find in [16].

A.3 SCRIPT_ASTRAL_MPI

```
#BSUB -n4
#BSUB -ext "SLURM[nodes=3]"
#BSUB -N
#BSUB -k chkfiles
#BSUB -u a.dammaro@cranfield.ac.uk
#BSUB -q extralong

##ssh_create_shared_keys

TMPFILE=/lustre/scratch/s128787/mots_test.half/appfile-hp

LIST="$LSB_HOSTS"
echo $LIST

NODE=$(echo $LSB_HOSTS |cut -d " " -f1)
echo "-h $NODE -np 1 /lustre/scratch/s128787/ mots_test.half /mots_master"
>> $TMPFILE
NODE=$(echo $LSB_HOSTS |cut -d " " -f2)
echo "-h $NODE -np 1 /lustre/scratch/s128787/ mots_test.half /mots_slave" >>
$TMPFILE
NODE=$(echo $LSB_HOSTS |cut -d " " -f3)
echo "-h $NODE -np 1 /lustre/scratch/s128787/ mots_test.half /mots_slave" >>
$TMPFILE
NODE=$(echo $LSB_HOSTS |cut -d " " -f4)
echo "-h $NODE -np 1 /lustre/scratch/s128787/ mots_test.half /mots_slave" >>
$TMPFILE


mpirun -f /lustre/scratch/s128787/ mots_test.half /appfile-hp
```