Towards the Automatic Generation of Zonal Models from CFD Simulations

Thibault Jacques-Antoine Marzullo, M.Sc

Supervisor:

Dr. Rory Monaghan, Mechanical Engineering, National University of Ireland Galway

Co-supervisors:

Dr. Marcus, Keane, Civil Engineering, National University of Ireland, Galway

Dr. Marco Geron, School of Aerospace and Mechanical Engineering, Queen's University of Belfast

Submission date: April 2020

Table of Figures

Table of Tables

Abstract

This doctoral thesis presents a novel computational tool chain able to automatically extract zonal models from CFD simulations of the built environment in order to provide accurate and rapid predictions of indoor temperature distributions. This doctoral thesis introduces a method for automatically clustering the computational cells of CFD simulations into sub-zones that can be considered quasi isothermal and compute the mass and heat exchange between the sub-zones and the domain to extract a zonal model. The zonal model can then be solved for off-design boundary conditions at a fraction of the computational cost of CFD.

This doctoral research first developed three methods for clustering CFD cells and compared their ability to capture the temperature distributions of the computational domain. These methods are: (1) the Mean Values Segmentation, (2) the Classic Watershed, and (3) the Coarse Grid method. The research presented in this thesis shows that the Mean Values Segmentation method performs best with regards to computational power and pertinence of the extracted clusters.

Then, this doctoral thesis developed a method for generating zonal models from cell clusters and information extracted from the CFD simulation results such as boundary conditions. The method was applied to three case studies: an office space, a naturally ventilated meeting room, and an ideal ventilation case. This study has assessed the fidelity of the automatically generated zonal models when solved for off-design boundary conditions and has showed that the zonal models are able to predict temperature distributions with a weighted mean absolute error under 0.6 K when the temperature and mass flow rate at the boundaries are changed. The proposed methodology is able to capture and simulate local phenomena such as air stratification and thermal plumes with minimal error and is able to accommodate complex flow fields provided that the domain's temperature is not uniform.

Acknowledgements

I would like to thank Rory, Marcus and Marco for their constant support during the research that led to the redaction of this thesis. You have given me precious advices and guidance, and immense personal support when I needed it most. I would also like to thank all the research and technical staff of NUI Galway, particularly William Kelly who always helped me and never asked "why?", even if sometimes he should have. I would like to express my gratitude to the College of Engineering of Informatics of NUI Galway and HIT2GAP for funding this research, as well as the IRUSE and ThERME research groups for their data and expertise. I would also like to thank the members of my GRC for their advice and support, and my examiners for their advice.

Achieving this thesis would not have been possible without the precious support of my family, their constant encouragements and their unconditional trust. They pushed me when motivation failed me, they forgave my absence when my mind was busy elsewhere, and they sent parcels of comfort food when I was hungry. *Merci Maman, Jérôme, Eléonore, Albéric, Monique, Jacques, et les toutous.*

Words are not sufficient to express my gratitude and my unconditional love for Aurélie, who endured my rants, late nights, relocations, unfinished projects and Italian foul language for the last seven years. I would not have done half of what I did if she had not been there, constantly holding my hand and lending her shoulder. *Tu m'as rendu si heureux que je n'ai pas vu le temps passer, et je voudrais que l'on continue de ne pas le voir passer ensemble, pour toujours.*

I want to thank all my friends of course, with a special mention for Alessandro and Grazia.

« Ce qui est simple est toujours faux. Ce qui ne l'est pas est inutilisable. »

What is simple is always wrong. What is not is unusable.

Paul Valéry, Mauvaises pensées et autres, 1942

Chapter 1 Introduction

1.1. Chapter introduction

This chapter will provide an introduction on the motivations for this doctoral work, presenting the trends of global energy usage and urbanization as well as the place of the built environment in these global problematics, leading to the problem statement. The chapter will be concluded by the statement of the research question that this thesis aims at answering, the aims and objectives as well as an outline of the thesis.

1.2. Cities of the future

The industrial era and its economic boom saw the progressive urbanization of society, a shift of population from rural areas to urban areas where the industry, and therefore industrial labour, were centralized. Figure 1-1 shows the increase in population living in urbanized areas between 1960 and 2018. Urbanization is an ever-going phenomenon and it is estimated that by 2050 the world will host 3 billion urban dwellers [1].

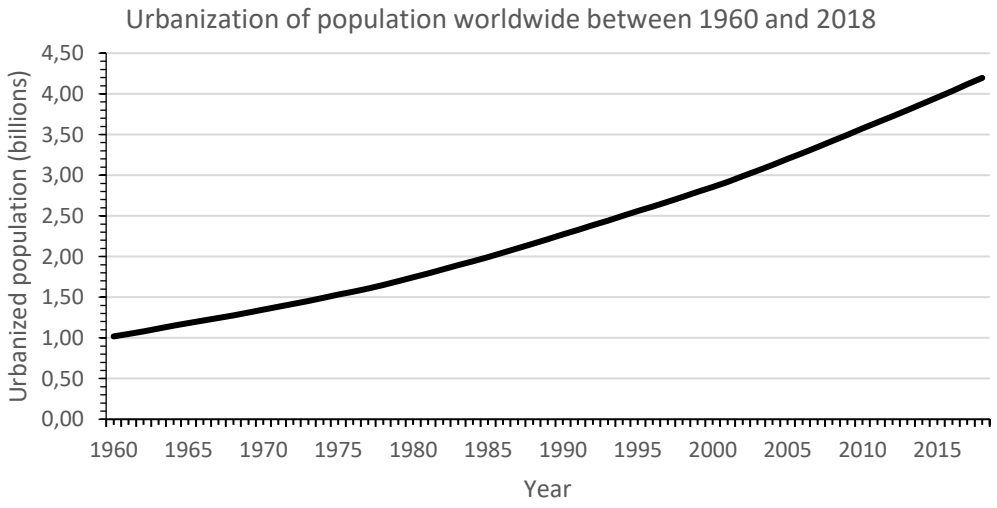


Figure 1-1: Urbanization of world population between 1960 and 2018[1]

With the phenomenon of urbanization and the increasing energy demand needed to sustain the modern comfort of life, the built environment became a major energy consumer and economic sector. Today it represents approximately 40% of global energy consumption, which is more than industry and transportation (Figure 1-2).

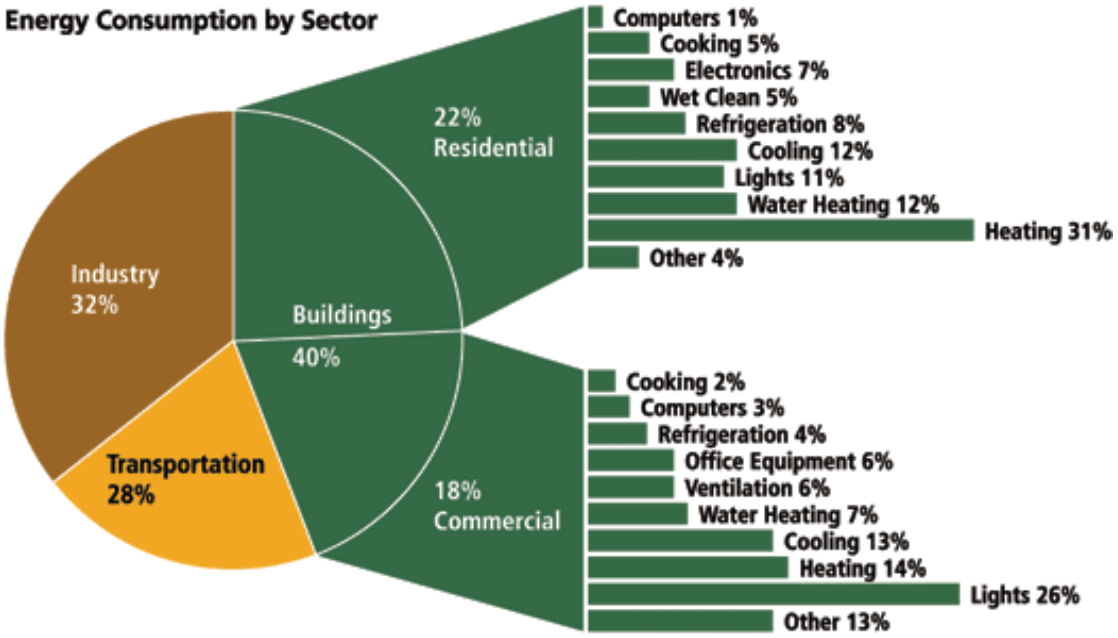


Figure 1-2: Global energy consumption by sector [2]

Following the global impulse towards the reduction of energy consumption, waste, and carbon emissions, urban areas are being redesigned to mitigate their energy usage and adapt to societal changes. Buildings, urban transportation and services are becoming more connected thanks to modern communication networks, and many aspects of urban life are being optimized with regards to energy consumption. These modern urban areas are often referred to as "smart cities", with the idea that they should be able to adapt to changing conditions in order to reduce waste and optimize the use of resources with regard to the immediate and predicted demands.

As a result, buildings too are experiencing a shift towards the adoption of Smart buildings or Net Zero Energy Buildings (NZEB). These buildings are designed to take advantage of envelope materials properties, environmental conditions, predictable energy demands, and computer models to optimize systems such that the energy input can be minimized without disruption or discomfort for the occupants. More specifically, Smart buildings are buildings equipped with sensing and actuating systems aimed at minimizing energy consumption through an optimal management of systems such as heating and cooling or

lighting. NZEB are buildings designed to produce as much energy as they consume, through the use of renewable energies such as geothermal and solar power and minimize the energy consumption through features that may also include those of Smart buildings.

1.3. Modern computing and the importance of indoor conditions

The advent of modern computing saw not only the increase of computational capabilities of electronic devices, but also their ubiquity in every aspect of modern life. In parallel, the evolution of communication networks allowed an increased interconnectivity of devices and led to the Internet of Things, paving the way for more complex and decentralized measurement and control systems. Modern computerized systems are used to gather data, communicate it and process it to extract information that can then be used to analyse trends, take informed decisions or provide services.

Like most other sectors, the built environment went through an important change in practices and methods related to the increasing availability and decreasing cost of connected devices and processing power. First it saw the development of computer models for thermal design of buildings, as early as 1963 [3]. This was followed by the increase of computational power and eventually the rise of more complex models, along with the development of embedded systems and cheaper sensors, which itself eventually led to the development of real-time Building Energy Management Systems (BEMS). Today the majority of building simulations focus on the optimization of energy usage, construction costs, operational costs and occupant comfort [4,5]. Recent studies focused on indoor airflow and thermal conditions and found that pollutant concentration and thermal distribution have a major impact on health, comfort ant productivity [6]. A poor indoor air quality can cause respiratory diseases, as well as the Sick Building Syndrome (SBS) which is a manifestation of illness symptoms linked to the amount of time an occupant spends in a building. Poor thermal distribution in a building can decrease productivity by 6 to 8% [7][8], as well as causing thermal discomfort. Designers mitigate these effects by optimizing the design of heating and cooling systems and the airflow in and around buildings. Unfortunately, the simulation of airflow and thermal distribution

for whole buildings is often an expensive and time-consuming process due to the computational power required to simulate airflow.

1.4. Model complexity

Despite the apparent availability of computational power, overly complex computer models require an extensive amount of time and expertise to provide accurate predictions. Therefore, the complexity of a model is often dictated by the intended application. Models aiming to simulate large systems such as whole buildings or districts will tend to ignore local physical phenomena to focus on large-scale effects such as global energy consumption or environmental factors. On the other hand, models simulating physics at a smaller scale such as a single room will include a more detailed description of the local physics.

With regards to complexity, one of the key assumptions is that of the uniformity of air properties [9]. Because of the large computational cost involved in simulating airflow, simplified models will tend to represent rooms or portions of buildings as instantly well-mixed volumes of air. This significantly simplifies the computation of a model's solution, introducing an acceptable degree of error but allowing the simulation of large systems over long periods of simulated time, or for rapid simulations. Unfortunately, the well-mixed assumption is not acceptable when simulating local phenomena such as thermal comfort or natural ventilation [10], therefore for these applications researchers often use more complex models such as computational fluid dynamics (CFD). These methods require extensive amounts of expertise and computational power but are predominant when simulating airflow inside and around buildings [11]. The well-mixed assumption is often a necessity rather than an intended feature, and intermediate methods such as zonal models have been developed to mitigate the situation. These intermediate methods usually imply developing a model from the ground up, or they imply extracting a simplified model from a more complex simulation. The resulting models are often rapid and accurate and allow the simulation of non-uniform airflows but they require extensive expertise for their development or extraction. Consequently, they do not have many practical applications despite their advantages [11].

1.5. Problem statement

Simplified approaches often assume that entire rooms or buildings are perfectly uniform, which does not correspond to reality nor permit the prediction of thermal comfort or natural ventilation. Complex models such as CFD are well established and are often used in the simulation of indoor airflow because they are able to provide accurate information. Unfortunately, they require extensive knowledge, user expertise, computational costs and validation efforts. Zonal approaches are often based on manual methods and user experience and can thus be expensive and time-consuming. Because of this major disadvantage, they are rarely used in practical applications.

1.6. Research question

Is it possible to leverage the accuracy of CFD to automatically generate zonal models that can provide accurate predictions of temperature distributions of indoor environments at a low computational cost?

1.7. Aims and objectives

The following aims and objectives have been defined to answer the research question.

Aims	**1.** ***To develop a method for reducing the complexity of CFD simulations***	**2.** ***To develop a toolchain for extracting zonal models from CFD simulations***	**3.** ***To develop a method for solving the zonal model for other boundary conditions***	**4.** ***To validate the suitability of the method for the built environment***
Objectives	**1.1** Define approaches for extracting thermal distributions from CFD simulations	**2.1** Develop an approach that automatically generates a zonal model from thermal distributions	**3.1** Develop a method for defining a new set of parameters for the zonal model	**4.1** Develop or replicate CFD simulations of cases commonly met in the built environment
	1.2 Ensure that the methods do not rely on user expertise	**2.2** Automatically extract boundary conditions from CFD for the zonal model	**3.2** Assess the error of the zonal model against CFD to define the usability envelope of the method	**4.2** Apply the developed method to the CFD simulations
	1.3 Assess the loss of accuracy for the candidate methods	**2.3** Assess the error between the zonal model and the CFD simulation	**3.3** Assess the flexibility of the method	**4.3** Identify strengths and shortcomings
	1.4 Assess the rapidity of the methods	**2.4** Assess the rapidity of the solver		

1.8. Thesis outline

The proposed doctoral thesis has been divided in 7 chapters:

Introduction: the current chapter, providing an overview of the research area and the research question, identifying the gap that this research proposes to bridge

Literature review: the state of the art will be presented with regards to energy modelling and the type of models used in the built environment, their applications along with their strengths and shortcomings. This chapter will narrow down the scope of the work to the generation of reduced-order models from CFD simulations.

Methodology: this chapter will present the proposed method for extracting zonal models from CFD simulations. It will give a detailed description of the data extraction process, the cell clustering methods developed during this doctoral work, the assumptions for the generation of a zonal model and finally the solution of the model.

Case studies: this chapter will present the three case studies that have been used to assess the performance of the proposed method. They include the CFD simulations of an office space, of a meeting room, as well as an ideal ventilation case.

Results: this chapter will present the core findings of the research, first with regards to data clustering, then the influence of the complexity of the zonal model on the final results, and finally on the flexibility of the zonal models generated using the proposed method.

Discussion: this chapter will give a critical discussion on the advantages of the method as well as the areas in which the performance should be improved.

Conclusion: this final chapter will summarize the research findings, and the knowledge gained during this doctoral work. It identifies the points on which the available literature was expanded and prospective areas of interest. It is concluded with proposals for future improvements or additions.

Chapter 2 Literature Review

2.1. Chapter Introduction

This chapter presents the review of the literature and the core findings of it. It has been organized into four sections. Section 2.2 focuses on sustainable building design: it presents the objectives of sustainable design and the main types of modelling approaches used in building simulation. Then, section 2.3 presents the two main approaches used to simulate airflow and thermal distributions in buildings, namely zonal models and CFD simulation. Section 2.4 presents the methods for extracting reduced-order models from high order CFD simulations. Finally, section 2.5 presents a conclusion on the current state of the research and identifies the problems that this doctoral thesis proposes to solve.

2.2. Sustainable building design

Buildings that are sustainable feature processes that are environmentally responsible and resource-efficient throughout a building's lifecycle. These processes include the design and construction at the early stages, but also the operation, maintenance and renovation of buildings. Finally, sustainable design considers the demolition and recycling of building materials. This review focuses on the design and operation of sustainable buildings. This section first presents the objectives commonly considered for energy and comfort optimization. Then it provides an overview of the types of computer models used to simulate and optimize the design of sustainable buildings.

2.2.1 Objectives of sustainable building design

If the only objective of sustainable buildings were to reduce the energy expense, the obvious solution would be to negate all energy consumption simply by disconnecting all systems from the power grid. Naturally this would render buildings unliveable, therefore designers and operators focus on other factors. By analysing some of the recent studies on building optimization, it is possible to identify several major trends. Evins [4] and Shaikh et al. [5] reviewed respectively 74 and 121 papers presenting studies on building optimization. Both found that a vast majority of studies place the energy consumption as the main objective for optimization methods. Evins found that energy consumption was followed by cost, which includes construction, operation and maintenance costs, on par with thermal and visual comfort. Shaikh et al. did not consider construction, operation and maintenance costs but they made a more thorough analysis on comfort. They found that after energy consumption, the most applied optimization objective was thermal comfort, followed by consumer preference, visual comfort, air quality and energy tariffs. Finally, both studies found that plug loads, lighting and humidity seem to be the least considered optimization objectives. Globally, energy consumption and thermal comfort stand out as the main objectives for the sustainable design of buildings.

2.2.2 Black, Grey and White box modelling

Computer models are used during the design as well as during the operation phases of a sustainable building. During the design phase, they are used to predict the effects of design choices on the building's performance and therefore help designers in reaching an optimal solution. Computer models can also be used to optimize existing systems, by providing energy-efficient control strategies and control parameters. The availability and accuracy of data, as well as the computational requirements inherent to each model usually restrict them to a certain domain of application. This section provides details on the different types of models for the built environment, which can be classified in three major categories: black box, white box and grey box.

Black box models, also called empirical models, are developed by analysing the parameters of a building during normal operating conditions and allow the characterization of the system without prior detailed knowledge of building materials,

sub-systems or environmental conditions. These are particularly useful when optimizing the performance of existing building stock. First, it is necessary to acquire data on the building's response to external inputs such as environmental conditions, energy demands or occupancy over a certain period. Then, statistical approaches such as linear regression [12] or artificial neural networks (ANN) [13] allow the correlation between inputs and response and allow the development of a model tailored to each individual building and its specifications. Black box models are generally fast to develop because they do not require a priori knowledge of the system, but they are particularly sensitive to the quality of the input data [14]. In fact, black box models are by definition generally reliable within the range of operating conditions they have been trained for but can fail if the conditions are off-design.

White box models, also called physics-based models, are built by defining mathematical equations that capture most of the physical interactions within the system and between the system and the environment. These equations are deducted directly from the laws of physics and engineering equations and can be simplified by the modelling expert if need be to characterize the simulated building. They can be used for whole building simulation or for the simulation of a limited system within a building, and their level of detail can vary greatly depending on their application. This approach, by its nature, allows the performance simulation of any building regardless of whether it has already been built and is therefore extensively used to test possible design choices and for optimization purposes. This allows designers to test various hypotheses and designs and make informed choices to improve the energy performance of a building project. Well established white box energy modelling tools such as EnergyPlus [23], ESP-r [24], and TRNSYS [25] are used in many studies aimed at analysing energy demand and test hypotheses for control strategies as presented in the review by Crawley et al. [15].

Grey box modelling combines the features of black box and white box modelling, by using empirical data to optimize a white box model. The white box model is developed using a simplified description of the system, and for which coefficients have been defined using rough estimates of the building's properties. The model is then optimized by comparing its output to empirical measurements taken in the building, which are used to correct the model's coefficients to match the outputs to the actual building operating conditions. A grey box method is described in Brastein et al. [16], which allowed the authors to estimate a building's thermal behaviour using a lumped capacitance model

coupled to empirical data. Similarly, Bacher and Madsen [17] applied a grey box model to a small research building, for which they first developed a lumped model of the building, and they then used empirical data to estimate parameters such as the thermal capacity and UA values of the building envelope.

The grey box approach aims to combine the advantages of both methods and mitigate their drawbacks. Simplified white box models offer lower computational costs and shorter development times, and errors introduced by the simplification are mitigated by tuning model parameters to match empirical data. The data gathering campaign can then be simplified since the model is supported by physical equations and not only by empirical data, also increasing the model's flexibility and accuracy when forecasting off-training conditions.

2.3. Simulating airflow in buildings

The optimal level of detail of a computer model should depend on its intended application and the extent of physical phenomena that it aims to capture. For example, in whole-building or yearly simulations the models tend to ignore the local physical phenomena, such as airflow in individual rooms, and use simplified representations instead in order to efficiently simulate a large-scale system. On the other hand, models aimed at studying smaller systems tend to reach a much higher level of detail, providing more accurate predictions of the cost of computational power. This is relevant for example when studying the influence of airflow on thermal comfort in a single room. This section will present the main levels of simplification and assumptions commonly used when simulating airflow, thermal distribution or pollutant transport in the built environment: (1) the zonal method, which divides large volumes into a set of perfectly-mixed sub-zones with simplified physics equations; and (2) Computational Fluid Dynamics (CFD) which simulates fluid flows with greater accuracy by solving a more extensive set of conservation equations including for example momentum and species transportation.

2.3.1 Zonal models

In building simulation, a zone is usually defined as an indoor space that typically consists of a room, a set of rooms or an entire building. For large-scale simulations such as whole-building or yearly simulations, zones are usually assumed instantly well-mixed and uniform. This allows a fast simulation of indoor conditions and is useful for energy systems modelling and optimisation. Unfortunately, the uniform assumption does not allow accurate predictions of air flows and temperature distributions and therefore cannot be used for accurate thermal comfort or pollutant dispersion simulations [18]. Zonal models address the issues inherent to uniform volume assumptions by dividing zones into smaller volumes called sub-zones. Sub-zones are aimed at capturing more finely local physics, thus allowing the characterization of phenomena like stratification, plumes or jets through a careful selection of volumes of interest. Figure 2-1 shows how a generic room can be divided into sub-zones that can capture temperature distributions and heat transfers. This allows the study of localised phenomena such as thermal comfort, energy consumption and air quality. Coupled with low computational cost, this provides rapid and reasonably accurate models of the built environment.

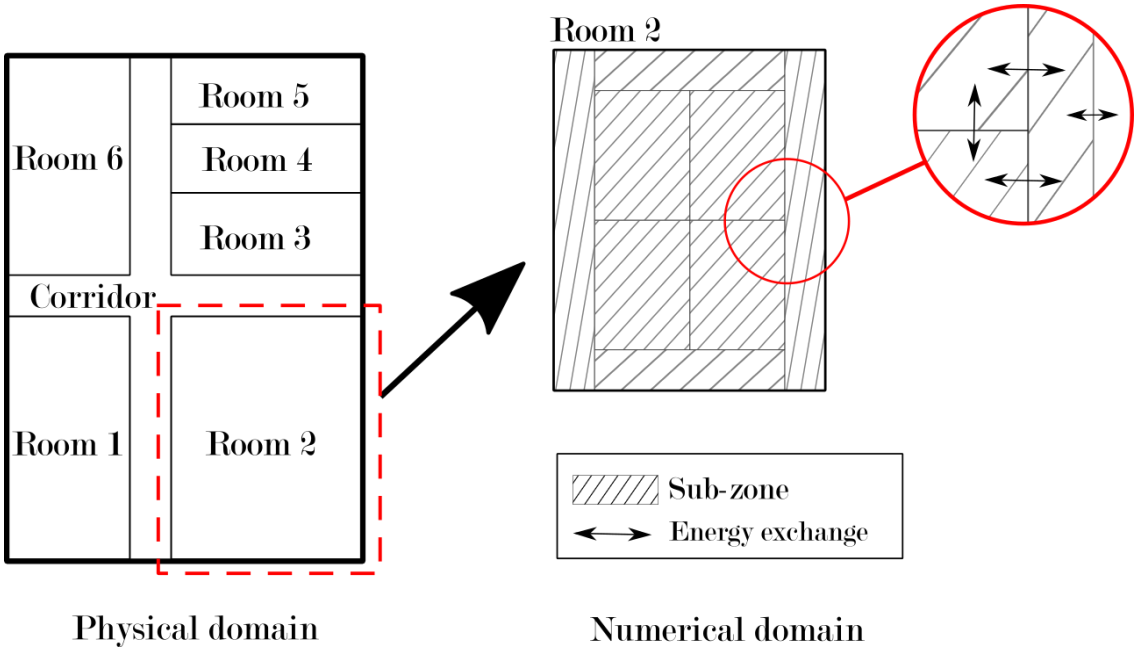


Figure 2-1: Simplified representation of the zonal model of a room divided into 8 sub-zones and the corresponding energy exchange paths

An early example of zonal model which uses mass and energy balance for each sub-zone and mass flows between sub-zones was developed by Wurtz [19]. The work studied the accuracy of the non-uniform temperature predictions when a heat source was present in the domain and found the results consistent with experimental data. This study also includes a combined natural and forced convection case which was developed by extracting data on mass flows from CFD results. Further work from Musy [20] showed that it is possible to automatically build zonal models by using Wurtz's model and reformulating it in smaller sets of equations from which it was possible to automatically extract a model given a certain partitioning of rooms inside a building. Musy was able to predict air movement, temperature distribution, and indoor air quality parameters in the whole building. Huang and Haghighat [21] developed a zonal model which has shown good agreement with CFD for airflow distribution, temperature distribution and contaminant pollution. The work features a mechanically ventilated room and integrates an air-jet model and a pollutant dispersion model to a 3-dimensional zonal model which was developed by partitioning a room into a coarse grid.

Other relevant work includes the study by Georges et al. [22] who developed a transient zonal model for the building energy simulation (BES) software IDA ICE [23]. They were able to validate a tool for modelling spaces heated by radiators and stoves. Their model was able to accurately predict the air stratification in the room, but it was unable to predict wall surface temperatures when the heat sources were near to the wall, due to the isothermal assumption for the walls used in many BES software packages. Their motivation was to provide a tool that would allow the prediction of thermal comfort and indoor air quality, which is impossible with common BES software that assume that zones are well-mixed volumes of air. Fang et al. [24] developed a zonal model for predicting indoor thermal distributions, in order to use it for simulating the performance of underfloor heating systems. Their work focused on airflow, and more particularly on developing an accurate air-jet submodel for the zonal model. They found good agreement between experimental data and their 160-sub-zone model's prediction. Wang et al. [25] developed a zonal model of a double-skin façade. The model would be used for dynamic simulations and whole-year simulations because of its computational efficiency. Their study found a good agreement between experimental and simulation data with a relative error below 8.3%. Song et al. [26] developed a zonal model for real-time predictions of temperature distribution in a mechanically ventilated room. They used the mean age of

air, predicted by a CFD simulation, as a criterion for defining the sub-zones of their model. They were able to validate the model against CFD and found that the definition of sub-zones based on the mean age of air provided more accurate results for the same number of sub-zones than the same model defined by hexahedral sub-zones (i.e. a coarse hexahedral mesh).

Table 2-1 below summarizes the zonal methods presented in this section.

Table 2-1 : Zonal models and their applications to the built environment

Authors	***Year***	***Method***	***Application***	***Notes***	***Ref.***
Wurtz	1995	Zonal model, own solver	Prediction of thermal comfort and pollutant transportation	Thesis, early work	[19]
Musy	1999	Zonal model, automatic method	Semi-automatic generation of zonal models	Thesis based on Wurtz's zonal model	[20]
Huang and Haghighat	2005	Zonal model with air-jet and pollutant dispersion models	Prediction of thermal distributions and airflow	Validated the model against CFD	[21]
Song et al.	2008	Zonal model	Prediction of temperature distributions	Validated the model against CFD	[26]
Fang et al.	2017	Zonal model with air-jet model	Prediction of temperature distributions	Validated against experimental data	[24]
Georges et al.	2019	Transient zonal model	Prediction of thermal comfort and thermal distribution	Developed in IDA ICE	[22]
Wang et al.	2019	Zonal model	Simulation of natural ventilation for double-skin façades	Yearly-simulation, validated against experimental data	[25]

The drawbacks of current zonal models are the difficulty in characterizing flow properties without additional data, such as CFD results, and the expertise needed to define pertinent sub-zones in order to efficiently capture most of the fluid domain's properties. Chen [11] underlines the fact that zonal models suffer from major overhead development times, which can sometimes be greater than for a CFD simulation.

2.3.2 Computational fluid dynamics

Computational fluid dynamics is a finite volume method for temperature and fluid flow analysis. Figure 2-2 shows how a fluid domain is subdivided into thousands or millions of finite volumes, called cells, which are used as control volumes for solving the Navier-Stokes equations iteratively. CFD can yield very accurate results and provide researchers with a deep insight on the properties of the fluid in the simulated domain. Chen [11] wrote a review on the tools used for studying ventilation performance in buildings. The study focused on research published during the previous year, in 2007, and found that over 70% of the publications used CFD for assessing ventilation performance. The author also noted that CFD was used for indoor air quality, natural ventilation and stratified ventilation cases since the other methods made assumptions on air mixing that prevented the prediction of these phenomena.

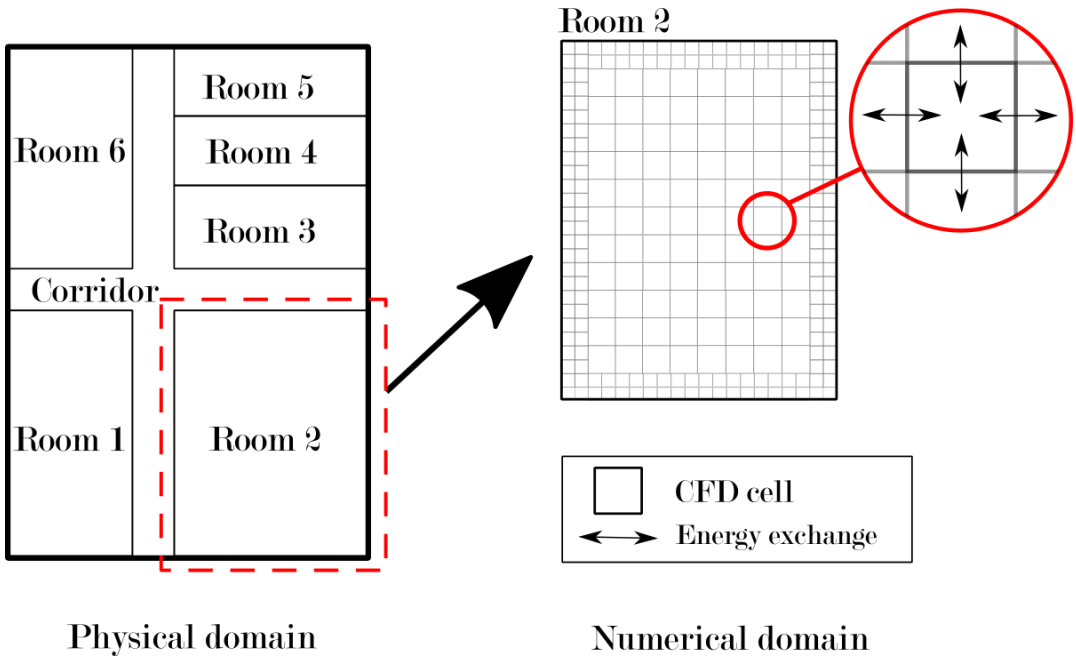


Figure 2-2: Simplified representation of the CFD finite volume method applied to a room, with the corresponding energy exchange paths

Lau and Chen [27] used CFD to simulate the airflow in a large workshop for assessing air quality when the air is supplied through floor-supply displacement ventilation compared to traditional mixing air supply. The CFD model was validated against experimental data. They found that while contaminant levels were lower in the case of floor-supply ventilation, the thermal comfort was lower than for mixing air supply due to high temperature stratification at occupant level. Liu and Mak [28] applied CFD to a natural ventilation case, by studying the performance of windcatchers as a means to provide natural ventilation for indoor spaces. They have validated the CFD model against published experimental data and found that windcatcher systems are very dependent on wind direction and speed, but they have a good efficiency as for most wind parameters the speed at the inlet in the room was equal to the incoming wind speed. They conclude that CFD is a suitable tool for studying natural ventilation cases. Zhao and Guan [29] used a CFD model for studying the indoor air quality when a room is equipped with a mixing ventilation system compared to a room equipped with personal ventilation systems. They validated the CFD model with regards to air velocity and temperature against experimental data. They found that while personal ventilation systems are seen as a good method for guaranteeing thermal comfort when several people are present in the same room, the air quality is lower.

Another approach to CFD modelling is to couple it with other models in order to provide more accurate boundary conditions parameters. For example, Du et al. [30] used CFD for studying the temperature distribution in a room in order to determine the optimal location for a HVAC temperature sensor. They used a nodal model to provide boundary conditions for CFD and determined the optimal location of the sensor with thermal comfort and energy usage as the objective functions. Zhai et al. [31] explored strategies to couple CFD with BES programs. They argue that CFD and BES provide complimentary information: the former provides detailed predictions on airflow and temperature distribution, while the latter provides predictions on thermal boundary conditions for the CFD. They realised two case studies in which they monitored heating and cooling loads for a given control strategy. They found that in both cases the detailed predictions provided by the coupled model, particularly on heat transfer coefficients at the walls, allowed a reduction in heating and cooling demand of at least 10% compared to predictions from the BES model only for the same control strategy.

Table 2-2 below presents recent studies that show the various applications of CFD to the built environment.

Table 2-2: Examples of the domains of application of CFD in the built environment

Authors	***Year***	***Method***	***Application***	***Notes***	***Ref.***
Lau and Chen	2007	RANS, RNG k-ε	Optimal placement of workshop ventilation	Experimentally validated	[27]
Lo, Banks and Novoselac	2013	RANS, 2-layer k-ε	Indoor airflow of wind-driven cross-ventilation, entire building	Experimentally validated. Steady-state and transient simulations.	[32]
Du et al.	2015	RANS, RNG k-ε	HVAC optimization with regards to thermal comfort	CFD-TRNSYS co-simulation	[30]
Zhang, Huang, Zhou	2015	RANS, 2-layer k-ε	Simulation of thermal comfort in a room when using various glazing and shading configurations.		[33]
Dong et al.	2017	RANS	Evaluation of different turbulence models in the simulation of indoor pollutant dispersion	Experimentally validated. Transient simulations. Standard k-ε, RNG k-ε, standard k-ω and SST k-ω	[34]
Zhao, Liu, Lai, Chen	2018	CFD-based adjoint method	Optimal design of air supply inlets in a room with regards to occupant thermal comfort	Identification of optimal number, size, location and parameters of air supply inlets	[35]
Cao et al.	2019	Semi-coupled CFD	Prediction of transient thermal responses of a room with HVAC	Fast CFD method	[36]
Kwok et al.	2019	RANS, SST k-ω	Prediction of temperature and pollutant distribution in an office building using a coupled approach.	Experimentally validated. Steady-state simulations.	[6]
Yang et al.	2020	RANS, RNG k-ε	Simulation of indoor ventilation in street canyons.	Experimentally validated. Steady state simulations.	[37]

The principal shortcomings of CFD are the high level of expertise and the long development and simulation time. In fact, results depend strongly on a good definition of boundary conditions and models included in the simulation, therefore they are particularly sensitive to the quality of data and the assumptions made on the boundaries or the models used to solve the CFD simulation. Consequently, the development of a CFD simulation requires a high level of user expertise and extensive amounts of time. The great numerical complexity renders computation power and time important factors to consider. This renders CFD impractical to whole building simulation, or when long periods of time need to be simulated.

2.3.3 Summary of building modelling methods

This section has introduced the three principal types of computer models, white box, black box and grey box and has then shown how the level of fidelity of computer models dictate their application to the built environment. The simplification usually applied to building simulations is that of the uniform and instantly well-mixed zone, which generally prevents the accurate prediction of airflow and thermal distributions. Zonal models tackle this problem by subdividing zones into smaller sub-zones and allow the prediction of temperature distribution within a zone. They are seen as a compromise between accuracy and computational power requirements, allowing real-time predictions of thermal distributions. However, they rarely provide satisfying predictions of airflow in a zone when used alone, and the development of a zonal model involves significant time and expertise. CFD provides accurate predictions on airflow properties and heat transfer coefficients and is extensively used in natural ventilation, indoor air quality and thermal comfort applications. The mathematical complexity of CFD renders it very expensive in terms of computational power, and the choice of parameters can have a substantially negative impact on the results accuracy. CFD therefore requires a high level of expertise and is time-consuming.

2.4. Reduction of CFD simulation complexity

Section 2.3.2 has shown the advantages of CFD and its domains of application, but also highlighted the high computational cost involved. To avoid this cost, researchers sometimes use other modelling methods such as zonal models, or couple CFD to these models as a means to provide accurate airflow predictions intermittently to increase the overall simulation accuracy.

Another approach to reducing the computational cost of CFD is to extract reduced-order models (ROMs) from it, or simplified models. Reduced-order models have the advantage of capturing most of CFD's accuracy at a much lower computational cost. This section presents the main approaches taken for model order reduction of CFD simulations of the built environment, divided into two main approaches: (1) numerical approaches, which extract a mathematical model from one or several CFD simulations, and (2) compartment models, which cluster CFD computational cells in order to reduce the overall complexity of the model.

2.4.1 Numerical methods for model order reduction

Reduced-order models aim to reduce the number of degrees of freedom of a CFD simulation, which is equal to the number of grid points times the number of state variables [38], by identifying a set of key parameters that are sufficient to model the physical phenomena of interest. A low order model is ideally capable of efficiently capturing a certain physical phenomenon but will not be able to capture the full extent of physical phenomena initially modelled in the high order model.

The literature review found that in the built environment one of the most common methods for reducing the order of CFD simulations is proper orthogonal decomposition (POD). The POD or Karhunen-Loève transform (KLT) uses a set of observations, a set of samples of a system's behaviour, to extract a selection of basic functions representative of the system's response in an optimal way [39]. In the built environment, research by Li

et al [40] applied POD to the prediction of temperature distributions in a room. Their objective was to provide accurate temperature distribution predictions for the optimal control of HVAC systems. They achieved this by generating a set of temperature distribution predictions with CFD, and later extracting a reduced order model using the POD method. They found that the reduced order model had good agreement with CFD, with an absolute error below 0.1 K at any point of the domain. The study, however, made several major assumptions: no heat sources were considered, the temperature of the boundaries was assumed constant and homogeneous, and the flow field was assumed constant. In fact, the authors assume that the flow field does not change when the reduced-order model is solved for different boundary conditions. Buoyancy effects induced by heat sources would negatively impact the accuracy of the reduced-order model and are therefore not considered by the authors. Similarly, Sempey et al. [41] applied POD to CFD simulations of an air-conditioned room in order to extract accurate and real-time reduced-order models. In order to mitigate the fixed flow field assumption, they modelled four cases corresponding to different inlet velocities in 2D CFD and used 30 to 34 snapshots depending on the case to extract a reduced-order model with POD. They found that the model had good accuracy compared to CFD, with a root mean squared error of under 0.66 K, including for cases where the reduced-order model was solved for inlet conditions that were not captured by the snapshots. They noticed however that the error was higher near the inlet air-jet and at the walls, suggesting that the reduced-order model still suffered from inaccuracies due to the fixed flow field assumption. When simulating the built environment, POD has proven to be a powerful method for predicting temperature distributions in real-time, with literature reporting good agreement between POD-extracted reduced-order models and CFD. The principal drawbacks of POD are the fact that it requires the simulation of a large number of high-order snapshots, and its inability to accommodate changes in the flow field. POD is also used in diverse fields such as image processing, signal analysis, chemical engineering, oceanography and aerospace [38].

Another method is fast fluid dynamics (FFD), which solves Navier-Stokes equations similarly to CFD, except for the advection term which is solved using a semi-Lagrangian approach [42]. This allows the solution of fluid dynamics simulations in real-time or faster than real-time. This method has been validated against experimental data for building simulation by Zuo and Chen [43], in three cases including one of cross-

ventilation in a room. Jin and Chen [44] applied FFD to whole-building simulation of buoyancy-driven airflow, in an effort to define a plume model that would accurately capture heat sources in the case of coarse-grid FFD. They have applied the method to several cases: a single room with a heated box, an occupied office, a system of two naturally ventilated rooms and a corridor, and a large atrium. They have compared the accuracy and computational cost of fine-grid FFD, coarse-grid FFD and coarse-grid FFD with a plume model. They have found that while coarse-grid FFD significantly reduced computing time at the cost of a larger error than FFD, coarse-grid FFD with plume model was as accurate as fine-grid FFD but faster to solve. Recent work by Tian et al. [45] has coupled FFD to a zonal model using the Modelica [46] programming language in order to obtain accurate predictions of thermal distributions within zones to optimize the control of HVAC systems. They have developed a dynamic simulation tool and applied it to a variable air volume system. They found that this method accurately represents the non-uniformity of momentum and temperature with a computational cost that would render it suitable for HVAC control applications. All the studies on FFD found during this literature review relied on solvers tailored to each case scenario, and extensive user expertise needed for setting the model's parameters.

POD and FFD have been both validated for use in building simulation, as they provide real-time predictions of temperature distributions which is an improvement over the well-mixed assumption found in zonal and nodal models. These advantages are mitigated by the fact that POD suffers from its dependency on many snapshots obtained from CFD simulations, while FFD still requires extensive user expertise.

2.4.2 Compartment models

Another approach for reducing the complexity of CFD simulations is the partitioning of CFD results into cell clusters that accurately represent a physical phenomenon of interest. This method has been studied in fields such as chemistry [47], hydraulics [48], but has seen very limited use in the built environment.

The method is based on clustering computational cells obtained from CFD results (Figure 2-3) in order to extract a simplified fluid network model that can subsequently be solved at a much lower computational cost. The clusters, or compartments, are assumed uniform and the interactions between compartments such as heat and mass exchange are used to

define a fluid network model. This results in a low order model that can capture non-uniformities in the fluid with good accuracy but at a much lower computational cost than CFD.

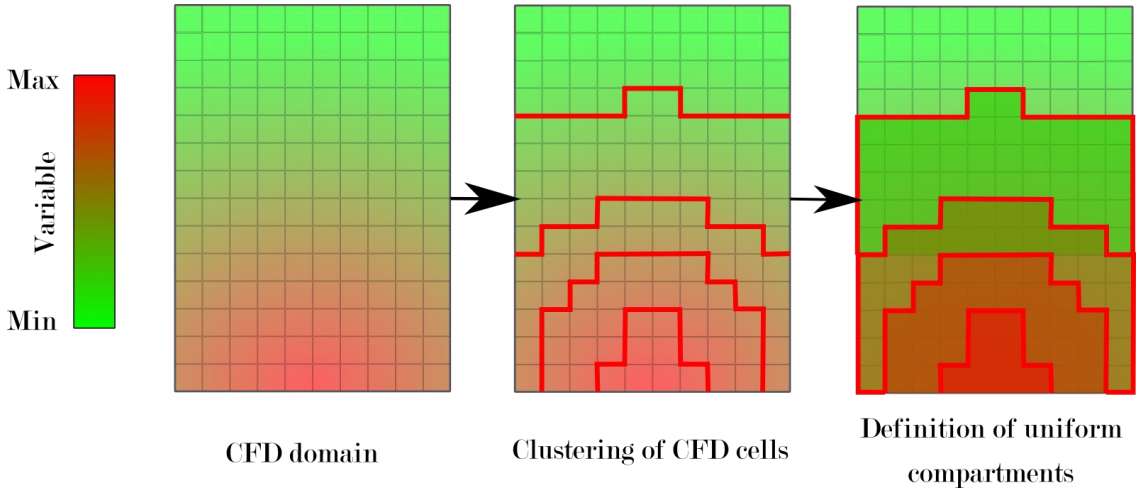


Figure 2-3: Simplified representation of the clustering of CFD cells into uniform compartments

In their recent paper, Tajsoleiman et al. [49] note that the accuracy of the resulting compartment model is heavily dependent on the definition of compartments. They also note that up to today, all the published work requires a manual definition of these compartments. In their work, they developed an automatic method for defining compartments based on pressure difference. They defined a range of pressure intervals and clustered the CFD cells depending on their pressure into a compartment. They applied this method to the case of a cylindrical stirring bioreactor and found that the resulting compartment model could provide results in good agreement with CFD in less than 2 seconds compared to the 2.5 hours it took to solve the corresponding CFD case. They also state that the model can be extrapolated to a range of operating conditions "as long as the flow regime does not change significantly". Another approach to a similar problem has been proposed by Weber et al. [47]. The compartment model of a multiphase loop reactor was designed and validated against validated CFD data. Their clustering method relied on a manually defined coarse grid, and the compartment model took 5.8 hours to yield a solution against 118 hours for CFD. In the field of hydraulics, Alvarado et al. [48] designed a compartment model to study the hydraulics of a wastewater stabilization pond. Their objective was to study the hydraulic behaviour coupled to a biokinetic reaction model, but CFD's computational cost was too high for practical use. They developed a compartment model based on validated CFD data where they manually defined a first set

of compartments based on "expert interpretation" of CFD results which were then divided into equal volumes. The resulting compartment model showed good agreement with experimental data at a computational cost that rendered the addition of a biokinetic reaction model acceptable.

No other published research that applied this method to the built environment has been found, except for the preliminary work done by Mullen et al. [50] which eventually led to the proposal of this doctoral thesis. Mullen et al. applied a clustering method based on temperature distributions to the CFD simulation of an office and found that the resulting reduced-order model could be solved at a lower computational cost and retain most of CFD's accuracy. They validated the model against CFD data and showed that the model could predict temperature distributions with an error of less than 10% when the temperature at the domain inlets were varied by ±10 K.

2.4.3 Summary of the methods for CFD order reduction

This section has shown the main approaches for reducing the complexity of CFD simulations in order to obtain rapid and accurate predictions of fluid flows in different applications. In the built environment, POD has been applied to several cases where the well-mixed assumption of zonal and nodal models did not permit prediction of airflow or temperature distributions within a zone. The published research shows that it is a valuable method for extracting a reduced-order model from CFD, but it is extracted from a sometimes-large number of CFD simulations, which renders it time-consuming. Furthermore, POD assumes a fixed flow field and has difficulties in retaining its accuracy when the flow properties change. FFD is another valid alternative to CFD for real-time applications, as it provides accurate and faster than real-time results for fluid dynamics simulations. While its results have been validated for simple cross-ventilation cases in the built environment, the method still relies on high user expertise and the FFD solvers are often developed case by case. Compartment models have been applied to other domains, such as bioreactors and wastewater hydraulics and have proven to be a viable alternative to CFD in applications where real-time results were needed. The method relies on clustering CFD cells into homogeneous volumes and defining a fluid network model based on the properties of each volume and the interactions between volumes. This

method however is still highly manual and relies on user expertise, with the exception of one work that proposed an automatic method based on pressure distribution.

2.5. Conclusions of the literature review

This chapter summarized the findings of the literature review made during this doctoral thesis. It first introduced the reader to the main objectives of building performance optimization: most studies aim at minimizing energy usage while maximizing occupant comfort, principally thermal. To achieve this, designers and operators use computer models to inform design choices and monitor systems operation.

The computer models used in building simulation are generally divided into three categories. White box models are developed by defining a mathematical representation of the physics of a building and its systems, which requires an extensive knowledge of the building and its systems but allows the simulation of hypothetical designs. Black box models are developed by fitting a mathematical function to actual measurements of a building's performance, or the response of a building to various inputs. These models do not require a detailed knowledge of a building's systems, and their fidelity relies on the quality of the data that can be gathered. They are often used for the performance optimization of existing buildings. Finally, grey box models take features from both black and white box modelling and offer a method for optimizing white box models with empirical data.

This chapter has also shown how building optimization requires certain assumptions on physics depending on the intended application of models. The well-mixed assumption does not allow the accurate description of airflow, useful when predicting occupant thermal comfort since it is dependent, among other factors, on temperature distributions. Researchers therefore rely on zonal models to assess fluid non-uniformities. Zonal models subdivide each zone into sub-zones and allow the characterization of temperature distributions at low computational cost. These models perform well and for a low computational cost, but they rely on extensive user expertise to define pertinent sub-zones that can capture the fluid physics accurately which eventually leads to long development times. Finally, CFD provides a detailed description of fluid physics with the use of a

finite-volume method. A zone is divided in hundreds of thousands or millions of control volumes, which allows the accurate representation of fluid flows. This makes CFD a widely used method for studying natural ventilation, thermal comfort and indoor air quality. However, CFD is sensitive to the assumptions made on boundary conditions and requires extensive computational power and user expertise. To mitigate this, researchers developed methods for extracting ROMs from CFD simulations. ROMs can capture a limited set of physical phenomena of interest and provide predictions with a fidelity comparable to CFD but at a fraction of the computational cost. Model order reduction methods applied to the built environment generally focus on POD and FFD. In POD, a set of CFD simulations is used as samples for extracting a mathematical model that give a simplified representation of the major flow features. In FFD, assumptions made on the Navier-Stokes equations allow a faster than real-time solution of fluid dynamics problems. Both methods have been validated for the built environment, but they require an additional level of user expertise. Finally, this literature review identified compartment models as a potentially automatable method for model order reduction. Compartment models are generated from CFD simulations by clustering cells that represent quasi-uniform regions of the domain. The clusters are then assumed uniform properties and are used to build a fluid network model representing the interactions between clusters, capturing mass and energy exchange. These models have shown good performance in other applications such as chemistry and hydraulics. They also show potential for method automation. An automated method for extracting zonal models that capture non-uniform temperature distributions would provide an advantage over the current highly manual methods used in the built environment.

Chapter 3 Methodology

3.1. Introduction

This chapter will present the proposed methodology for extracting zonal models from CFD simulations and solving them for different boundary condition parameters. First, a short overview of the method will be given. Then, an overview of the software packages used in this work will be presented. The CFD data extraction method will be introduced, with details on the available data and the steps taken to classify and record it. The methods used for clustering CFD cells will be presented, and the assumptions made on the resulting sub-zones will be explained along with the assumptions made on the energy exchange between sub-zones. Finally, the generation of the full zonal model will be explained as well as the method for assessing the results.

3.2. Method overview

The proposed method relies on the clustering of CFD cells to generate isothermal sub-zones that can later be considered uniform for the purpose of generating a simplified fluid network model. The fluid network model is generated using the uniform sub-zones as its nodes, and the energy exchange between nodes is computed from the mass and heat exchange between cell clusters from the CFD results. Figure 3-1 shows an idealized overview of the method: (a) the CFD simulation results are used to extract data on the properties of the fluid flow; (b) the CFD cells are clustered into isothermal sub-zones; (c) the sub-zones are then assumed uniform with regards to temperature, pressure and density; (d) boundary conditions are extracted from CFD as well as mass flow rates and heat exchange between sub-zones and to/from the boundaries to form a fluid network model; and finally (e) the model can be solved for different boundary conditions parameters in order to predict temperature distributions. The resulting simplified model

aims at providing predictions of temperature distributions for the indoor built environment.

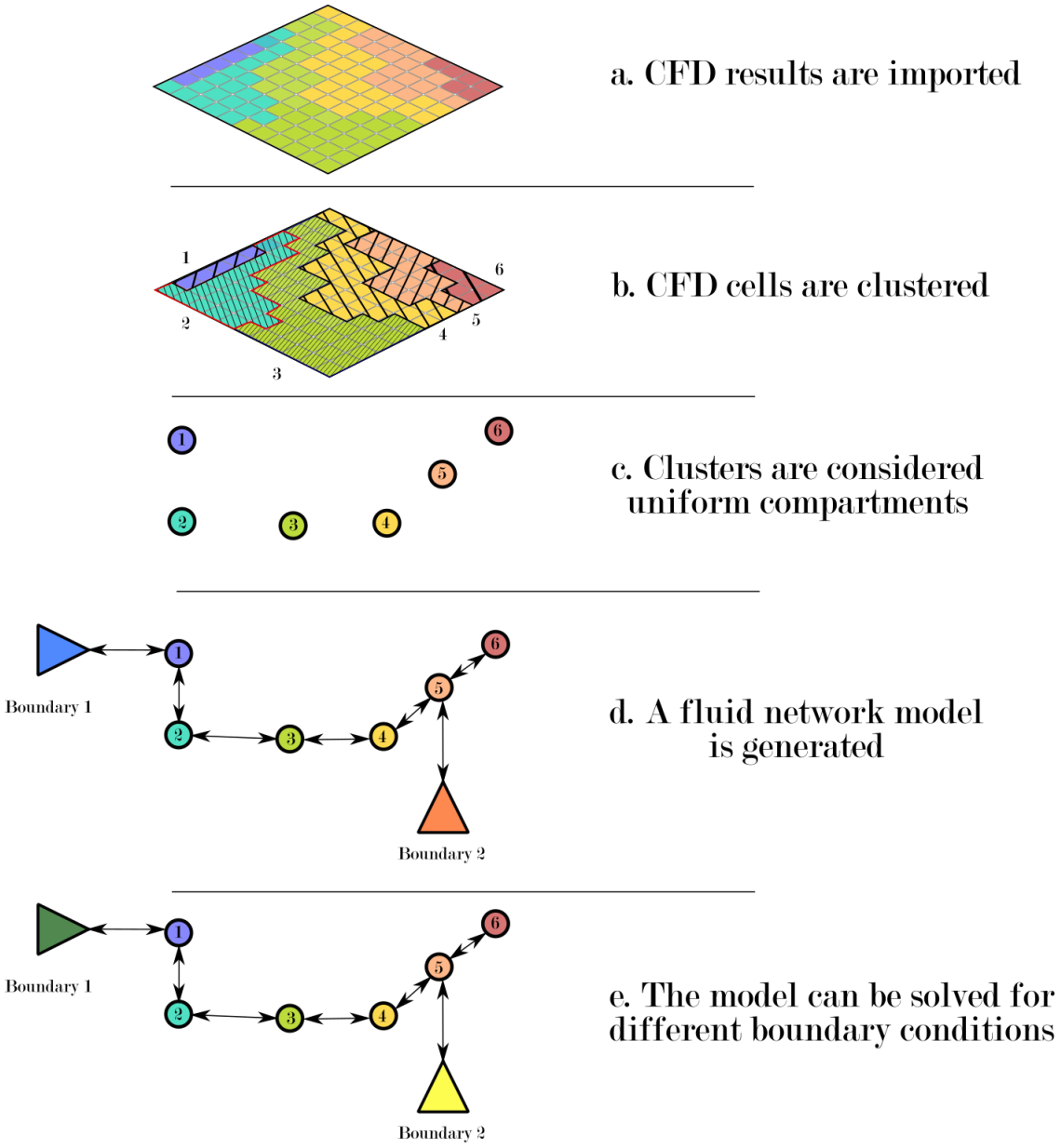


Figure 3-1: Overview of the proposed methodology. CFD data is imported (a) and used to define clusters (b) which are approximated as being uniform and used to define a fluid network model (d) that can be solved for different boundary conditions parameters (e)

3.3. Software packages

This section will present the different software packages that have been used for this research. PHOENICS CFD [51] was used for developing the CFD simulations, Sinda/FLUINT [52] was used for solving the zonal models, and the code presented in this work was developed in the Python programming language [53].

3.3.1 PHOENICS CFD

PHOENICS CFD is a CFD package published by CHAM. A feature of PHOENICS is the possibility to define immersed boundary conditions. The mesh is a cartesian structured mesh, which allows the definition of complex cases without the need to spend considerable time on meshing when coupled with immersed boundary conditions. The cartesian structured mesh also allows a more efficient ordering of data as will be explained in the following sections. PHOENICS CFD provides an interface for outputting the results in Tecplot format, which is compatible with many visualization software packages. Results files written in the Tecplot format can be output in either ASCII or binary format. ASCII files are written in characters readable by humans, which makes it a useful debugging tool, while binary format is not readable by humans but provides smaller files and renders read/write tasks more efficient computationally.

PHOENICS CFD has been used in a variety of studies, which include applications to the built environment. Cui and Zheng [54] used PHOENICS to study the effect of building greening on urban heat islands, and Zhang et al. [55] studied the impact of green roofs on building energy consumption. Guo et al. [56] used PHOENICS to study urban-scale natural ventilation. Dong and Liang [57] studied the dispersion of fire smoke in an atrium. PHOENICS provides features for customizing the type of energy sources and sinks through its text-based interface, as well as a user-friendly Graphical User Interface (GUI), which allows faster definition of cases where pre-defined objects can be used.

The quantities balanced in the PHOENICS solver are:

Scalars:

- Pressure
- Temperature
- Enthalpy
- Mass fractions
- Volume fractions
- Turbulence quantities
- Various potentials

Vectors:

- Velocities
- Radiation fluxes
- Displacements

The continuity equation (Equation 1), Navier-Stokes conservation of momentum (Equation 2) and the conservation of energy (Equation 3) can be written as:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho u) = 0 \qquad \textit{Equation 1}$$

Where ρ is the density, $\nabla \cdot$ the divergence and u the flow velocity;

$$\frac{\partial}{\partial t}(\rho u) + \nabla \cdot (\rho u \otimes u) = -\nabla \bar{p} + \mu \nabla^2 u + \frac{1}{3}\mu \nabla (\nabla \cdot u) + \rho g \qquad \textit{Equation 2}$$

Where $\otimes$ is the outer product, $\bar{p}$ the mechanical pressure, μ the shear viscosity coefficient, and g the gravitational force.

$$\frac{\partial}{\partial t}(\rho \varepsilon_0) + \nabla \cdot (\rho u \varepsilon_0) = \nabla \cdot \big((\tau - pI) \cdot u\big) + \rho g \cdot u + \nabla \cdot q + \nabla \cdot q_r \qquad \textit{Equation 3}$$

Where ε_0 is the total energy, τ the viscous stress tensor, I the unit tensor, q the heat-flux and q_r the radiative heat-flux.

The generalized form of the conservation equation solved in PHOENICS can be written as:

$$\frac{\partial(\rho\varphi)}{\partial t} + \nabla \cdot \left(\rho\vec{U}\varphi - \Gamma_\varphi \nabla\varphi\right) = S_\varphi \qquad \textit{Equation 4}$$

Where ρ is the density, φ the variable in question, $\vec{U}$ the velocity vector, Γ_φ the diffusing exchange coefficient of φ and S_φ the source term.

PHOENICS uses a finite volume formulation of the balance equation, obtained by integrating the differential equation over the cell volume. More details on the mathematical basis of PHOENICS are available in [58].

3.3.2 Python language

The Python language is an interpreted programming language often used in scientific applications because of its relative simplicity and the numerous scientific libraries available to the user. Interpreted languages are programming languages in which the instructions are executed directly from the source code, without needing to be compiled which is the translation of the source code to machine-language instructions. This allows for faster prototyping and testing of functions compared to compiled languages. As mentioned earlier, numerous scientific libraries are available to the user. This work used mainly one library, the NumPy library which is focused on numerical operations and data structuring. It was used because of the presence of useful basic functions such as *mean()*, which computes the mean value of an object, but also because it offers practical ways to initialize and modify n-dimensional arrays. Its functions have been used extensively in the code presented in this doctoral thesis.

3.3.3 Sinda/FLUINT solver

Sinda is a thermal analysis software package developed by C&R technologies. It is a package useful for compiling network models such as RC models applied to thermal problems. It was created in the 1960s and extensively used and developed. In the 1990s Sinda saw the extension of its features to fluid network models with a package named FLUINT. Sinda/FLUINT can be used either with a GUI or with input files written in a format similar to FORTRAN. This allows the automatic generation of an input file

directly from another code and the solution of the model by Sinda/FLUINT. This study uses Sinda/FLUINT because its features allow the definition of a fluid network model with heat and mass exchange, both in the fluid domain and with solid boundaries. Section 3.7.1 provides details on how Sinda/FLUINT's features are leveraged for the scope of this thesis, and sections 3.7.1.2 and 3.7.1.3 provide details on the equations solved by Sinda/FLUINT.

3.4. CFD data extraction

This section will present the first step toward generating a zonal model, which is the extraction of data from the CFD results.

3.4.1 CFD data structure

PHOENICS CFD uses a finite volume method, meaning that a fluid domain is discretized by subdividing it into smaller volumes. These elementary volumes are called cells, and their properties are assumed uniform within each cell. These cells are used to simulate the fluid flow, and the interactions of the fluid with solids. Typically, CFD codes define cells that can be tetrahedrons, pyramids, triangular prisms or hexahedrons, as shown in Figure 3-2.

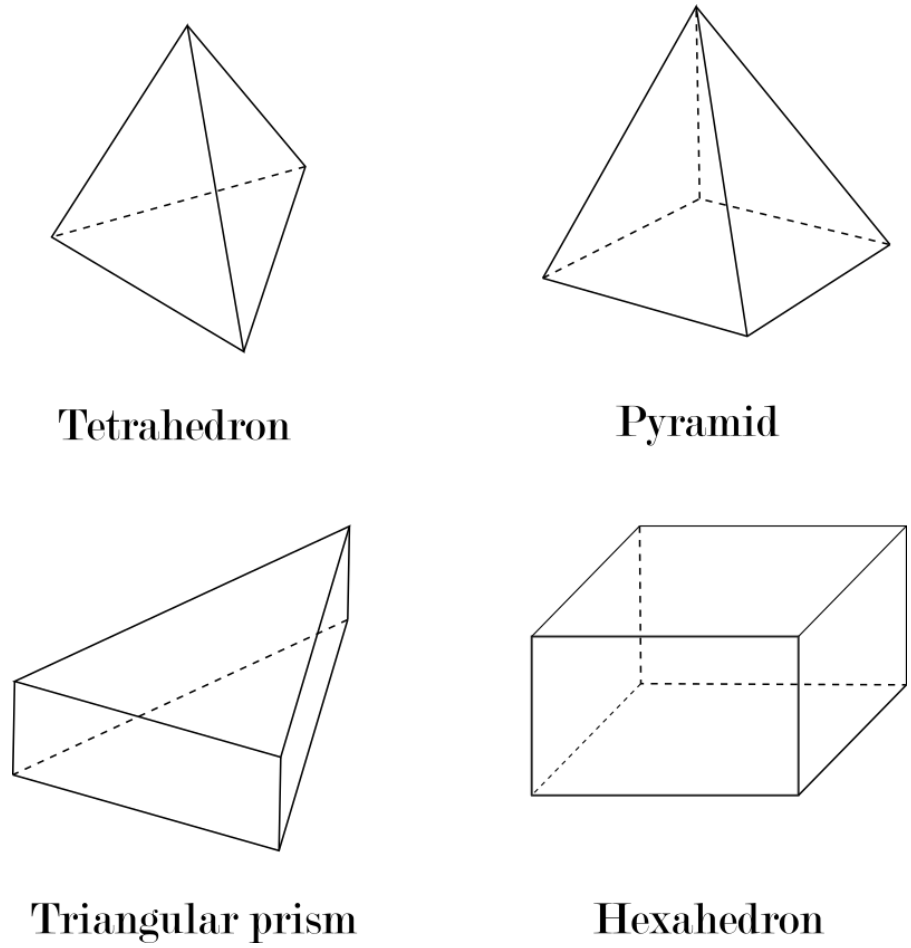


Figure 3-2: Usual geometries for CFD cells

The PHOENICS CFD package allows the definition of a cartesian structured mesh, meaning that all cells are hexahedrons and are arranged in a structured grid. This is an important factor as it means that each CFD cell can be indexed, therefore allowing their classification into a multidimensional structured array which is explained in detail in section 3.4.2.

3.4.1.1 Fluid data

Each CFD cell stores the properties of the fluid contained within it, and these quantities are available to the user when the simulation is complete. Some quantities are stored at the centre of the cell such as the pressure, and others such as velocities are stored at the vertices (or nodes) of the cells. Consequently, the data is available in two separate arrays: cell-centred and node-centred. In PHOENICS CFD the user must define which properties

are presented in the results file, and Table 3-1 summarizes the properties that are needed by the algorithm for clustering cells and generating a zonal model.

The *X*, *Y* and *Z* coordinates are used to process interface areas between cells and between cells and boundaries (sections 3.6.2 and 3.6.3). The density *DEN1* and pressure *P1* are used to process the uniform properties of sub-zones (section 3.5.4), as well as the temperature *TEM1* which is also used in the cell-clustering process (section 3.5). *PRPS* is used to differentiate between cells that contain a fluid and cells that contain a solid (section 3.4.2). The heat transfer coefficient *HTCO* and mass flow rates *CNH1*, *CNE1* and *CNN1* are used to process the mass and heat exchange between sub-zones and between sub-zones and boundaries (sections 3.6.2 and 3.6.3). The mass flow rates at the High, East and North face of each cell are stored at the cell centre and the mass flow rates at the Low, West and South faces are deducted from the values of the previous cells in the I, J and K direction. PHOENICS uses an hybrid upwind discretization scheme of the convection equation described in [59].

Table 3-1: Variables extracted from the CFD results, the dataset they correspond to, and their usage in the proposed work

Variable	***Description***	***Unit***	***Storage***	***Usage***
X, Y, Z	Coordinates	m	Node	Model generation
DEN1	Density	$kg.m^{-3}$	Node	Model generation
CNH1	High face mass flow rate	$kg.s^{-1}$	Cell-centre	Model generation
CNE1	East face mass flow rate	$kg.s^{-1}$	Cell-centre	Model generation
CNN1	North face mass flow rate	$kg.s^{-1}$	Cell-centre	Model generation
P1	Pressure	Pa	Cell-centre	Model generation
TEM1	Temperature	K	Node	Zoning/Model generation
PRPS	Material type	N/A	Cell-centre	Zoning
HTCO	Heat transfer coefficient (turbulent flow)	$W.m^{-2}$	Cell-centre	Model generation
R1	Mass flow rate through domain inlets and outlets	$kg.s^{-1}$	CFD results output	Model generation

3.4.1.2 Boundary conditions

PHOENICS CFD outputs another set of data, the “result” file. It contains information from the solver, and for the scope of this work it is used to extract information on the boundary conditions. It contains the name and parameters for each boundary, except for geometrical information such as their position and dimensions.

PHOENICS defines the *"COVAL"* solver statement which provides a unique format for defining all the boundary conditions of the simulation. *"COVAL"* stands for *CO*efficient and *VAL*ue, and depending on the type of boundary condition, both will represent different variables. The *CO* and *VAL* variables are used in this work to extract surface

heat transfer coefficients and temperatures for linear heat sources. Next is the *R1* variable which corresponds to the mass flow rate of inlets and outlets. Finally, a name is assigned to each boundary condition and is used in this work for identification purposes. Figure 3-3 shows an extract of the "result" file where the solver command used to set a surface heat flux has been highlighted.

```
  Parent VR object for this patch is: NWINDOW1
PATCH(OC7-H ,WEST , 4, 4, 8, 26, 60, 72, 1, 1)
COVAL(OC7-H ,TEM1,2.3 ,20.82 )

  Parent VR object for this patch is: NWIND_2
PATCH(OC8-H ,WEST , 4, 4, 29, 55, 60, 72, 1, 1)
COVAL(OC8-H ,TEM1,2.3 ,20.82 )

  Parent VR object for this patch is: NWIND_3
PATCH(OC9-H ,WEST , 4, 4, 57, 82, 60, 72, 1, 1)
COVAL(OC9-H ,TEM1,2.3 ,20.82 )
```

Figure 3-3: extract from the "result" file containing the PHOENICS solver commands, automatically generated by the GUI tool or manually input by the user. The command for defining a surface heat transfer is highlighted, in this case the "NWINDOW1" boundary is set to have a heat transfer coefficient (CO) of 2.3 W.m^{-2}.K^{-1}and a temperature (VAL) of 20.82 °C

3.4.1.3 Domain objects

The last file required in this work is the geometry file. It contains information on the position and size of each object, as well as their type. All the objects of the domain are represented in this file, identified by a unique name and a list of node coordinates used to locate the object in the domain.

3.4.2 Data extraction and classification:

This section describes the functions used for reading the files, extracting and organizing the data. Currently the code interprets files written in ASCII format or text format. This is for debugging purposes in order to offer a data format readable by humans, but there is no technical limitation to using the memory-efficient binary file format.

3.4.2.1 Fluid data

Fluid data is input in the Tecplot data format, which is a format shared by numerous CFD and data visualization packages. The code reads the input file line by line, parses each line, and organizes the data into NumPy arrays for later access.

The Tecplot format packages data and is organized as follows. First, a header comprising a few lines contains information on the data array. It includes the dimensions of the array in the orthonormal *I*, *J* and *K* directions (corresponding to *X*, *Y* and *Z*), the list of variables, whether the array is cell or node centred, and additional information that is required by the Tecplot format but not used in this work. Finally, the data array contains the values of the variables for each node in the array. Figure 3-4 shows an example of CFD results in the Tecplot format.

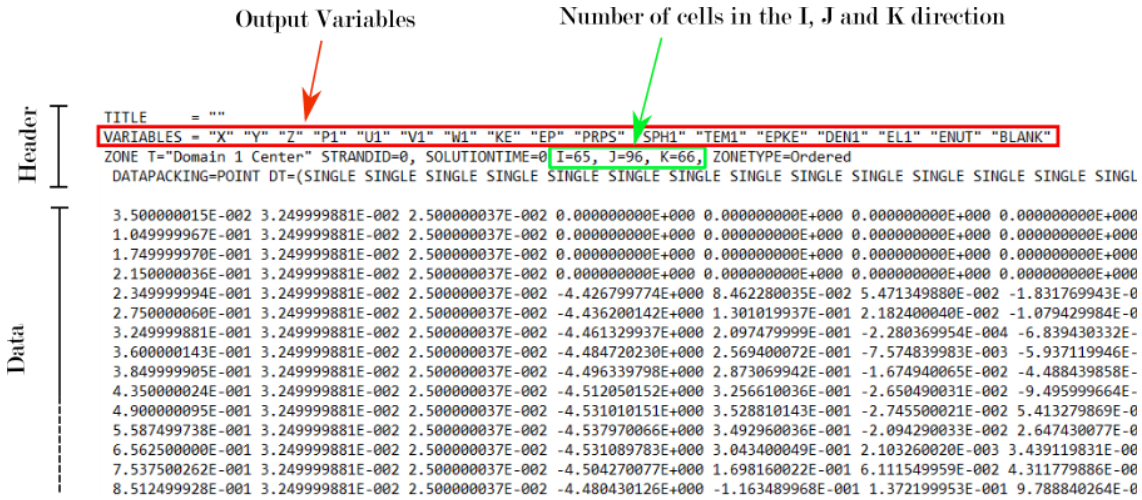


Figure 3-4: Extract of CFD results in the Tecplot format

Then, the header and data array are repeated for the cell-centred data. The data is presented as a 1-dimensional array of $I \times J \times K \times N_{variables}$ numbers that need to be reorganized in a 4-dimensional array containing the flow data. The first step is to identify in the header the values for *I*, *J* and *K*. Then, the list of variables is extracted and checked for two reasons: the code checks that all the required variables are present, and it counts the total number of variables $N_{variables}$. With this information available, the next step is to create a NumPy array of dimensions $I \times J \times K \times 5$ containing the node-centred data. In fact, it is an array of $I \times J \times K$ cells containing each a vector of length 5 in the form $\{X, Y, Z, DEN1, TEM1\}$ as illustrated in Figure 3-5.

The same process is repeated to generate the cell-centred array. Since cell-centred data is used extensively throughout all the steps of the method presented in this work, the array sees the addition of new variables used during cell clustering and for data output. Consequently, the array is of dimensions $I \times J \times K \times 15$ in order to record 15 variables for each CFD cell. The variables are X, Y, Z, $Zone\ ID$, $Zone\ type$, $TEM1$, $Zone\ temperature$, $Error$, $DEN1$, $CNN1$, $CNH1$, $CNE1$, $P1$, $Volume$, $Boundary$ as seen in Figure 3-5.

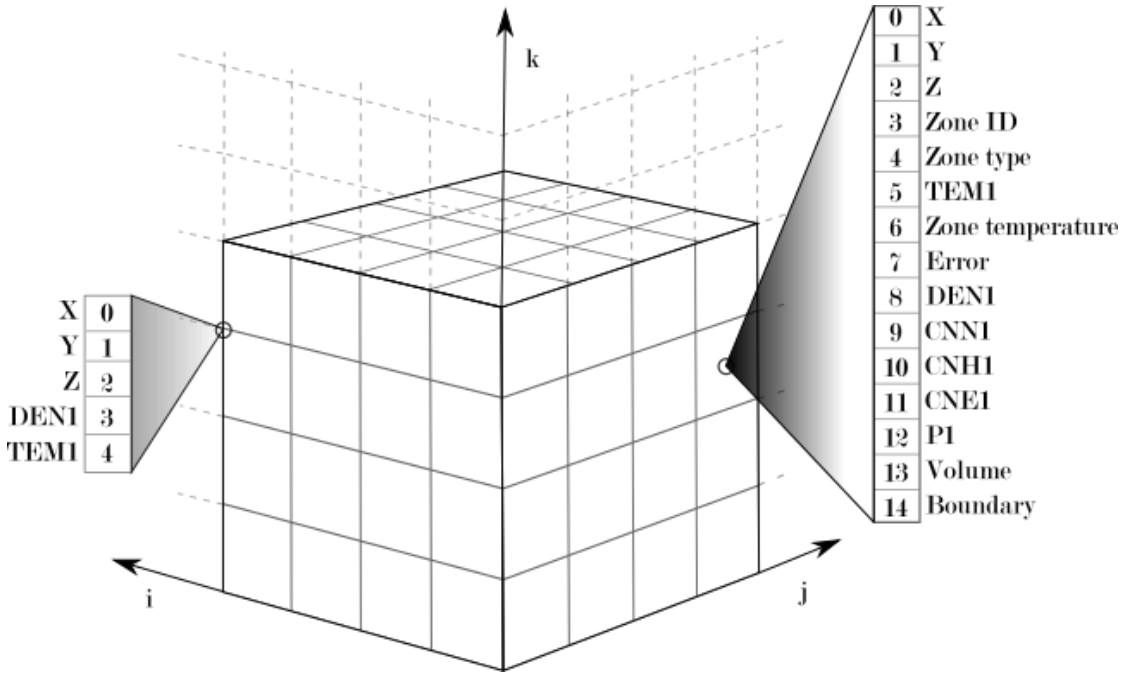


Figure 3-5: Representation of the data space used in this work. Two datasets are created: a node-centred and a cell-centred. Each 3-dimensional node and cell contains a vector representing its properties

The values for *X, Y, Z, DEN1*, *TEM1*, *CNN1*, *CNH1*, *CNE1* and *P1* are taken from the CFD result file while the other values in the vector are initialized to -1 and then changed to the appropriate value in the clustering and model generation steps, because the code only uses positive integers to identify zone types, sub-zones and boundaries.

Finally, a $I \times J \times K$ Boolean array is created to flag solid cells. The solid cells flag Boolean array is generated with the information provided by the *PRPS* variable and contains 0 for a fluid cell and 1 for a solid, as a consequence to the immersed boundary condition approach used in PHOENICS. This is used for later excluding any solids from cell clustering, as PHOENICS outputs the temperature of a solid cell to 0 (Kelvin or Celsius) which would cause problems during cell clustering as explained in the following

sections. As an outcome to this first step, the flow data has been organized into two 4-dimensional arrays containing the node-centred data and the cell-centred data and an additional array used to flag solids. It is a more intuitive way to access and visualize the data than the original 1-dimensional list of values.

3.4.2.2 Boundary conditions

The next step in the CFD data extraction process is the identification of boundary conditions. The *"result"* file contains this information, in the form of commands for the CFD solver and outputs from the simulation. The commands can either be input manually by the user or automatically generated when using PHOENICS' GUI utility. Most of the information is extracted from solver commands: the code searches lines that start with the keyword *PATCH* followed by the unique name of the boundary and then a keyword identifying the type of boundary condition. In the case of a linear heat source for example, the keyword will be *TEM1* (an example is shown in Figure 3-6). The next two numbers in the line are the *CO* and *VAL* values. As stated earlier, in the case of a linear heat source these values correspond respectively to the heat transfer coefficient and temperature of a surface, and they are recorded in a Python dictionary. If the flow is turbulent, PHOENICS stores the heat transfer coefficient in the HTCO output variable. If such is the case, the HTCO value will be used instead in the zonal model.

```
  Parent VR object for this patch is: SWALL_2
PATCH(OB1B ,EWALL , 86, 86, 3, 16, 51, 78, 1, 1)
COVAL(OB1B ,V1 , GRND2 ,0. )
COVAL(OB1B ,W1 , GRND2 ,0. )
COVAL(OB1B ,KE , GRND2 , GRND2 )
COVAL(OB1B ,EP , GRND2 , GRND2 )
COVAL(OB1B ,TEM1, GRND2 ,23.639999 )
```

Figure 3-6: Solver command setting the properties of each boundary

VAL on the other hand can be directly extracted from another section of the *"result"* file, the "Nett sources" section (Figure 3-7). In this section the energy sources of the domain boundaries are printed after the solution to the CFD model is available. Among other results for energy sources, the values for surface heat fluxes and the mass flow rates at the inlets and outlets are present. Again, the code searches for the keyword "Nett source of:" which is followed by a keyword identifying the type of source. If the keyword is

TEM1 it means that the line contains the temperature of a surface, and if it is *R1* it means that the line contains the mass flow rate at an inlet or outlet. After the source type keyword, the unique name of the boundary is written and recorded by the code, and it is followed by the actual value for that source: temperature or mass flow rate. Other types of sources such as turbulent kinetic energy are present but they are not used in this work.

```
Nett source of R1   at patch named: OB21     (EASTINLET   ) = 1.566548E-01
Nett source of R1   at patch named: OB24     (NORTHOUTLET ) =-1.566500E-01
pos. sum=0.156655 neg. sum=-0.15665
nett sum=4.813075E-06

Nett source of KE   at patch named: OB21     (EASTINLET   ) = 2.093300E-04
Nett source of KE   at patch named: OB24     (NORTHOUTLET ) =-2.201188E-04
Nett source of KE   at patch named: OB1      (SWALL       ) = 3.469573E-08
```

Figure 3-7: Output of the nett sources of energy at the domain boundaries in the result file

At this stage, information on inlets and outlets has been extracted and at least part of the information on linear heat sources has been extracted. PHOENICS records the *HTCO* value when the flow is turbulent, when this happens the *HTCO* value is used instead of the boundary's *CO* value for the heat transfer coefficient. This method is repeatable for quadratic heat sources as they are recorded in a similar way in the "result" file.

3.4.2.3 Domain objects

PHOENICS CFD records the shape and position of 2D and 3D objects in the domain in a file called "geometry". Objects regroup items such as boundary conditions (inlets, outlets, heat sources) and solids that interfere with the fluid flow. Extracting the shapes of the objects is important for defining the boundary conditions for the zonal model: it is necessary for processing exchange areas for heat flows and to process mass flow rates at the inlets and outlets.

For each object, the geometry file (shown in Figure 3-8) contains its unique name, the type of object (listed in Table 3-2) and a list of node coordinates used to characterize the shape and position of the object in the 3D domain. The code extracts the unique name of the object and links it to the list of objects extracted at the previous step. Then the type of object is recorded, as well as the list of node coordinates.

```
TITLE="Geometry data file for: CS04"
VARIABLES = "X", "Y", "Z"
ZONE T="SWALL", ZONETYPE=FEQUADRILATERAL, DATAPACKING=POINT, N=    24, E=     6
AUXDATA OBJTYPE="PLATE"
  6.15600E+00  1.60000E-01  0.00000E+00
  6.15600E+00  1.60000E-01  0.00000E+00
  6.15600E+00  1.60000E-01  3.47000E+00
  6.15600E+00  1.60000E-01  3.47000E+00
  6.15600E+00  0.00000E+00  0.00000E+00
  6.15600E+00  1.60000E-01  0.00000E+00
  6.15600E+00  1.60000E-01  3.47000E+00
  6.15600E+00  0.00000E+00  3.47000E+00
  6.15600E+00  0.00000E+00  0.00000E+00
  6.15600E+00  0.00000E+00  0.00000E+00
  6.15600E+00  0.00000E+00  3.47000E+00
  6.15600E+00  0.00000E+00  3.47000E+00
  6.15600E+00  1.60000E-01  0.00000E+00
  6.15600E+00  0.00000E+00  0.00000E+00
  6.15600E+00  0.00000E+00  3.47000E+00
  6.15600E+00  1.60000E-01  3.47000E+00
  6.15600E+00  0.00000E+00  3.47000E+00
  6.15600E+00  0.00000E+00  3.47000E+00
  6.15600E+00  1.60000E-01  3.47000E+00
  6.15600E+00  1.60000E-01  3.47000E+00
  6.15600E+00  1.60000E-01  0.00000E+00
  6.15600E+00  1.60000E-01  0.00000E+00
  6.15600E+00  0.00000E+00  0.00000E+00
  6.15600E+00  0.00000E+00  0.00000E+00
```

Figure 3-8: Extract of the PHOENICS geometry file. It presents the coordinates (in metres) of the object's vertices

The node coordinates must be indexed first. The code uses the node-centred array defined in Section 3.4.2 to compare the coordinates extracted from the geometry file to the coordinates present in the node-centred array. As a result, each object is now characterized by a list of node indices rather than coordinates. Depending on the type of object, it can be defined by as little as 4 nodes or by the full list of nodes contained by the object. After extracting the list of node indices for each object, the code detects whether the object is 2D or 3D: it searches for the maximum and minimum indices in each direction *I*, *J* and *K*. If any direction has a minimum index equal to the maximum index, it signifies that the object is 2D. Otherwise, it is 3D.

Table 3-2: List of object types used by PHOENICS CFD, and whether the current version of the python code is capable of extracting them. Note: object types used only for plotting results are not included. Source: PHOENICS user manual [60]

Object Type	***Brief Description***	***Implemented in the method***
Pressure Relief	single cell fixed pressure point	Y
Outlet	2D, fixed pressure	Y
Inlet	2D, fixed mass source	Y
Wind_Profile	2D, fixed mass source following atmospheric boundary layer	N
Plate	2D, zero thickness obstacle to flow. May be porous	Y
Thin Plate	2D, nominal thickness for heat transfer	Y
Fan	2D, fixed velocity	N
Blockage	3D, solid or fluid. Can apply heat and momentum sources	Y
Angled-in	3D, fixed mass source on surface of underlying BLOCKAGE object	Y
PCB	3D, solid or fluid with non-isotropic thermal conductivity	N
ROTOR	3D, rotating co-ordinate zone in cylindrical-polar grid	N
Wind	3D, whole domain, applies wind profiles at domain boundaries	N
Sun	3D, whole domain, applies solar radiation heat load within domain	Y
Angled-out	3D, fixed pressure on surface of underlying BLOCKAGE object	Y
Foliage	3D, represents effects of vegetation	N
User Defined	2D or 3D, for setting user-defined sources (PATCH/COVAL)	Y
Celltype	2D or 3D, for setting user-defined sources (cannot affect grid)	N
Assembly	2D or 3D container object for multi-component object	N

3.4.2.3.1 2D objects

When an object is identified as 2-dimensional, the code generates a temporary list of node indices comprising all the nodes belonging to the object. To do so, the gap between the minimum and maximum indices of the object's nodes is filled. Figure 3-9 shows the three-step process: (1) the four nodes of the object are extracted, (2) all the nodes in between the initial nodes are added to the object, and (3) the cell faces contained by the nodes are added to the object.

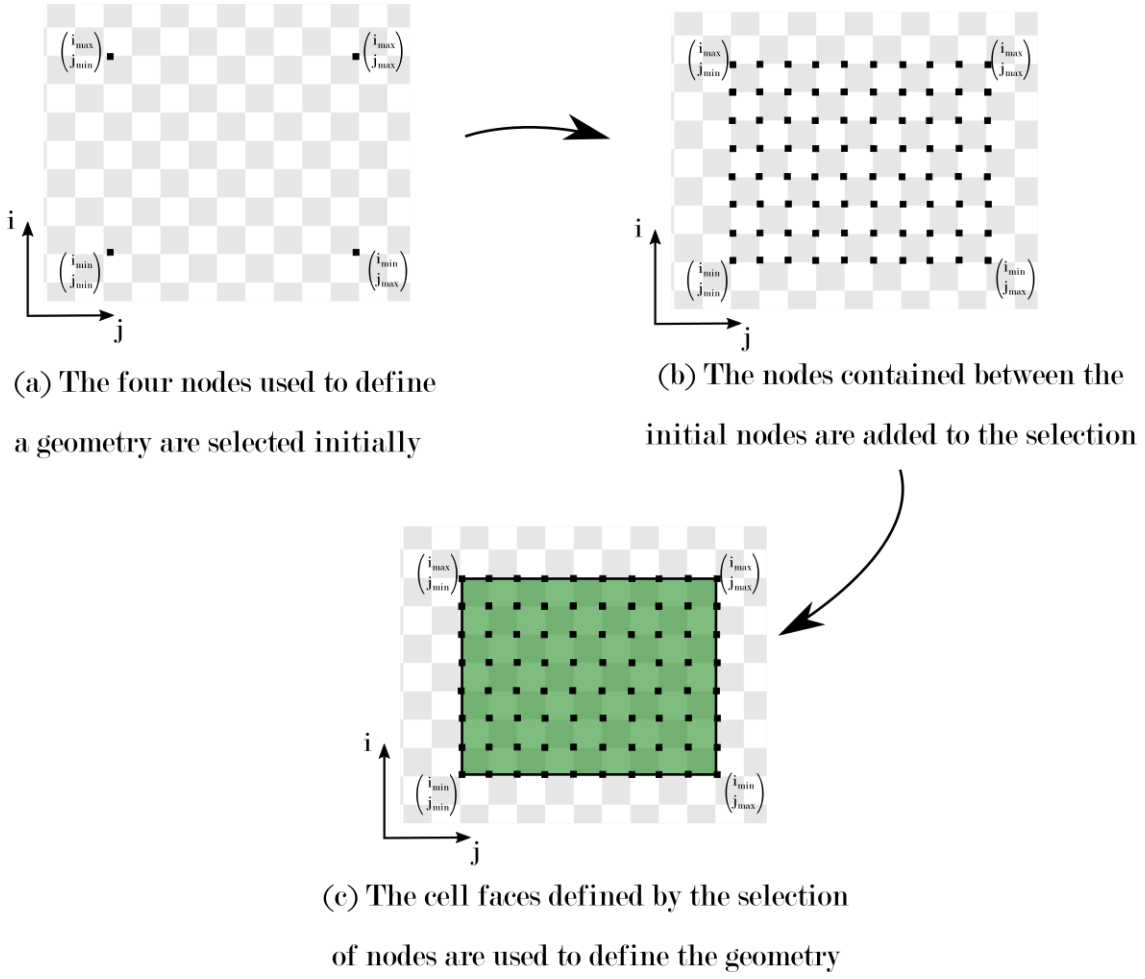


Figure 3-9: 3 step procedure for importing 2D geometries: the 4 vertices are extracted from the geometry file, the nodes in the gap are added to the geometry, and finally the cells contained within the nodes are considered as being part of the geometry

When all 2-dimentional objects have been extracted, the code compares the temporary lists of indices to detect any intersection between 2D objects. If there is an intersection, as shown in Figure 3-10, the code updates the list of nodes of each object to guarantee that objects do not overlap.

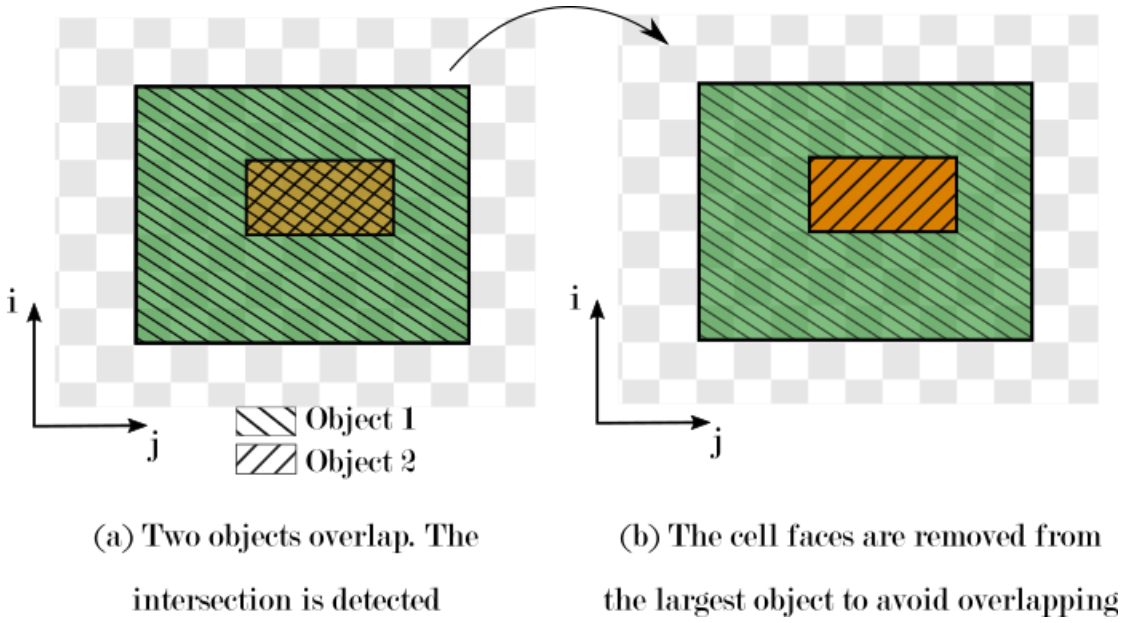


Figure 3-10: Illustration of the step resolving geometry intersection issues

Finally, the code updates the cell-centred array to identify cells that have a face in contact with a boundary. A unique identifier is defined for each object and the value of the *Boundary* variable in the cell-centred array is updated with the object identifier (shown in Figure 3-5).

3.4.2.3.2 3D objects

Currently the code is able to correctly identify hexahedral objects and *ANGLED-IN/ANGLED-OUT* objects as defined in Table 3-2. In PHOENICS CFD, *ANGLED* objects are inlets or outlets that are not strictly in the XY, XZ or YZ plane. PHOENICS defines an angled inlet or outlet as the intersection between an *ANGLED* object and a *BLOCKAGE* object as shown in Figure 3-11.

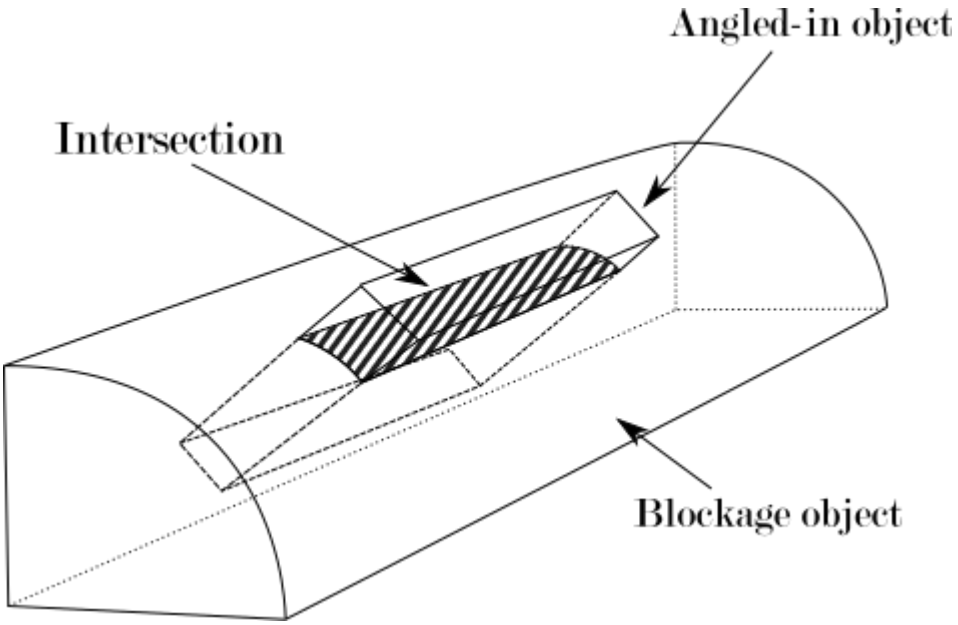


Figure 3-11: The angled inlet defined as the intersection between an ANGLED-IN and BLOCKAGE object in PHOENICS CFD

As in the 2D case, the code first extracts a temporary list of node indices. This time since the objects are 3-dimensional, the list is translated directly into a list of cell indices. Each object is characterized by a name, its type (*BLOCKAGE, ANGLED-IN or ANGLED-OUT*) and the list of cells belonging to the object. After completing the temporary list of objects, the code searches for objects of type *ANGLED-IN* or *ANGLED-OUT*. If it finds any, it will search for intersections with objects of type *BLOCKAGE*. When an intersection is found, the list of objects is updated: the *BLOCKAGE* and *ANGLED* object are deleted and replaced by the new object defined by their intersection. Similarly to the 2-dimensional case, each object is assigned a unique identifier and the cell-centred array is updated by changing the $Boundary$ variable of cells belonging to an object. As stated earlier, currently only hexahedral objects can be correctly identified. Further work is required to identify other 3-dimensional objects, in order of complexity this would mean first objects that can be constructed by extrusion of any 2D object and then the more complex case of polyhedrons.

3.4.3 Summary of CFD data extraction

This section presented the type of data that is extracted from CFD results, and the details on how the code extracts and organizes this data. Three types of information are extracted from the CFD results: (1) the fluid flow properties, organized in a node-centred array and a cell-centred array; (2) the boundary conditions, recorded as a Python dictionary for later usage during the parametrization of the zonal model; and (3) the boundary geometry information, used to detect which nodes and cells are in contact with a boundary. The next section will present the method for clustering cells into sub-zones of uniform properties.

3.5. Sub-zone Generation

The method evaluated by this doctoral thesis is the clustering of CFD computational cells into sub-zones, which can be assumed to have uniform properties, to capture most of the flow features while providing a model that can be solved for a lower computational cost than CFD. Sub-zone generation ignores CFD cells that do not contain a fluid, therefore at each step the code verifies that the cells it iterates through have not been flagged as solid in the array defined in section 3.4.2.1.

This work has developed and compared three algorithms for clustering cells: (1) Mean Values Segmentation (MVS), created within the scope of this work; (2) Coarse Grid, (CG), in which the CFD mesh is interpolated to a very coarse cartesian structured mesh; (3) Classic Watershed (CW), which is a method used in image processing for image segmentation. This section will present the three algorithms and the assumptions made on cell clusters in order to define uniform sub-zones to be used in the zonal model.

3.5.1 Mean Values Segmentation

In the MVS algorithm the clustering of CFD cells into sub-zones is done in 3 steps, described in this section: (1) the definition of zone types, or the criteria for including a cell in a certain cluster; (2) cell clustering, or the method for efficiently searching for cells

of identical zone type and assigning them to a cluster; and (3) the conversion of clusters into sub-zones of uniform properties.

3.5.1.1 Zone types

This section will define two notions: the zone type and zone criteria. In order to define clusters, a rule for inclusion of cells into clusters must be defined. This is the role of the zone type: to define an interval in which a parameter can fit, or not. Zone criteria are the parameters for which zone types are defined: they can be for example temperature, velocity magnitude, pressure or density. This work used a single zone criterion at a time, and the results have been presented for temperature as the zone criterion. Therefore, in this section the description of the method will be done with temperature as a zone criterion, but the algorithm is applicable to other parameters as well. The usage of multiple zone criteria has not been developed in this work. In order to define zone types, MVS proceeds iteratively (Figure 3-12). First, the entire domain is scanned and the mean value for the zone criterion is computed, in this work it is temperature. Then, the mean temperature is used to split the cells in two groups: cells which temperature is lower or equal to the mean temperature, and cells which temperature is higher than the mean temperature. The operation is repeated, the mean temperature of both groups is computed, and they are split each into two new groups.

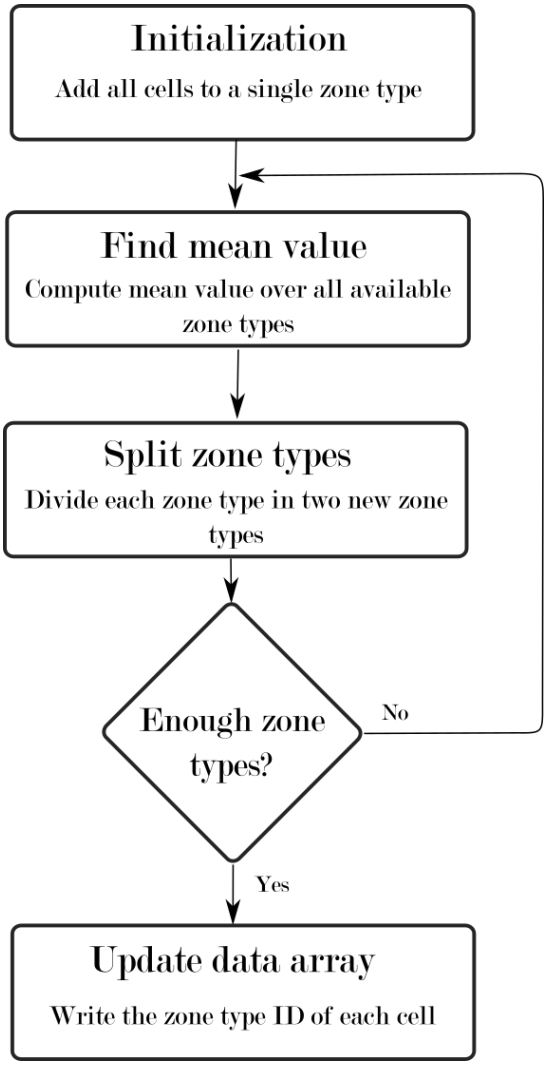


Figure 3-12: Flowchart of the MVS method

This will be repeated until the number of zone types defined by the user has been generated. This process is shown in Figure 3-13. At each step, the list of cells belonging to a zone type is split into two new lists that are used to compute the mean temperature of each zone type. The lists contain a set of data vectors containing the I, J and K coordinates of each cell as well as its temperature.

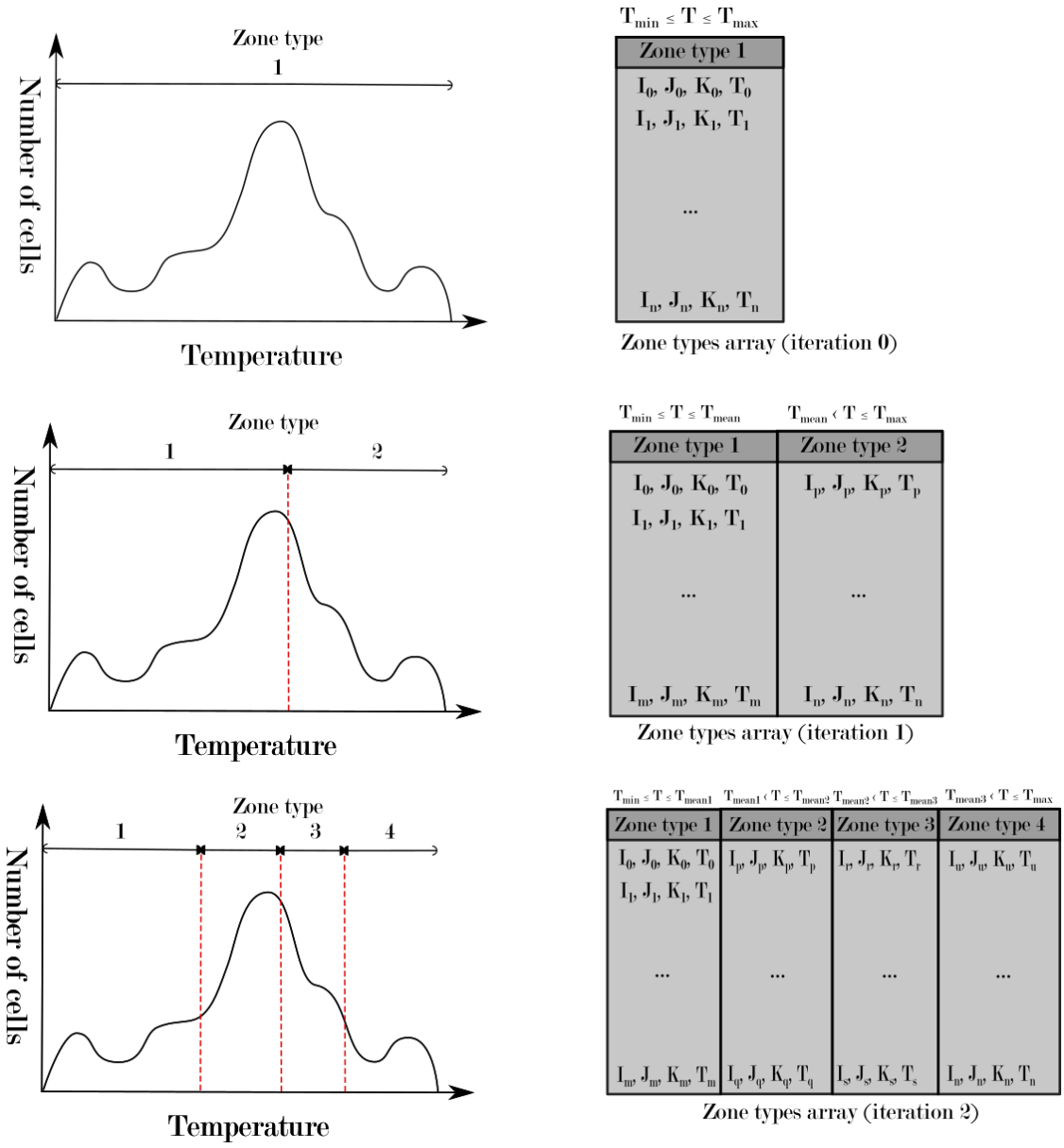


Figure 3-13: Representation of the iterative assignment of zone-types. On the left, a representation of cell number versus temperature, and on the right the zone types array containing the I, J, K coordinates and the temperature of each cell. At the n-th step, the cells are classified into 2^n groups, which in the n+1-th step are divided into 2^{n+1} groups depending on their mean temperature

The number of zone types corresponds to the minimum number of clusters that should be created: several clusters can belong to the same zone type, and a zone type has necessarily at least one cell belonging to it.

With regards to the code, the first step is to create an array containing all the cells and their temperature. The array contains the *I*, *J* and *K* indices of each cell along with the corresponding temperature. The mean value for temperatures in the array is computed, and the array is split into two columns, containing each the list of cells belonging to each

zone type. The operation is repeated, and at the end of the process the array will contain as many columns as there are zone types.

As a final step, the code updates the cell-centred array with the *zone type ID* of each cell. This information will be used in the next step of the MVS algorithm, cell clustering.

3.5.1.2 Cell clustering

In the previous step, zone types have been defined and the cell-centred array has been updated with this information, but no clusters have been created yet. A cluster is defined as a group of computational cells that are contiguous and share the same zone type. To create these clusters, the algorithm proceeds in the following way: (1) the starting cell does not belong to a cluster yet, so a new cluster is created; (2) all the cells surrounding the initial cell and which do not belong to a cluster yet are scanned, if they share the same zone type as the initial cell they are added to the cluster; (3) the algorithm proceeds to the next cell of the cluster and repeats the operation; (4) when no more cells can be added to the cluster any more, a new cluster is created and populated with the next available cell; (5) clustering continues until all the cells in the domain have been assigned to a cluster. Figure 3-14 shows a workflow of the clustering method. This section presents the computational details of the clustering method.

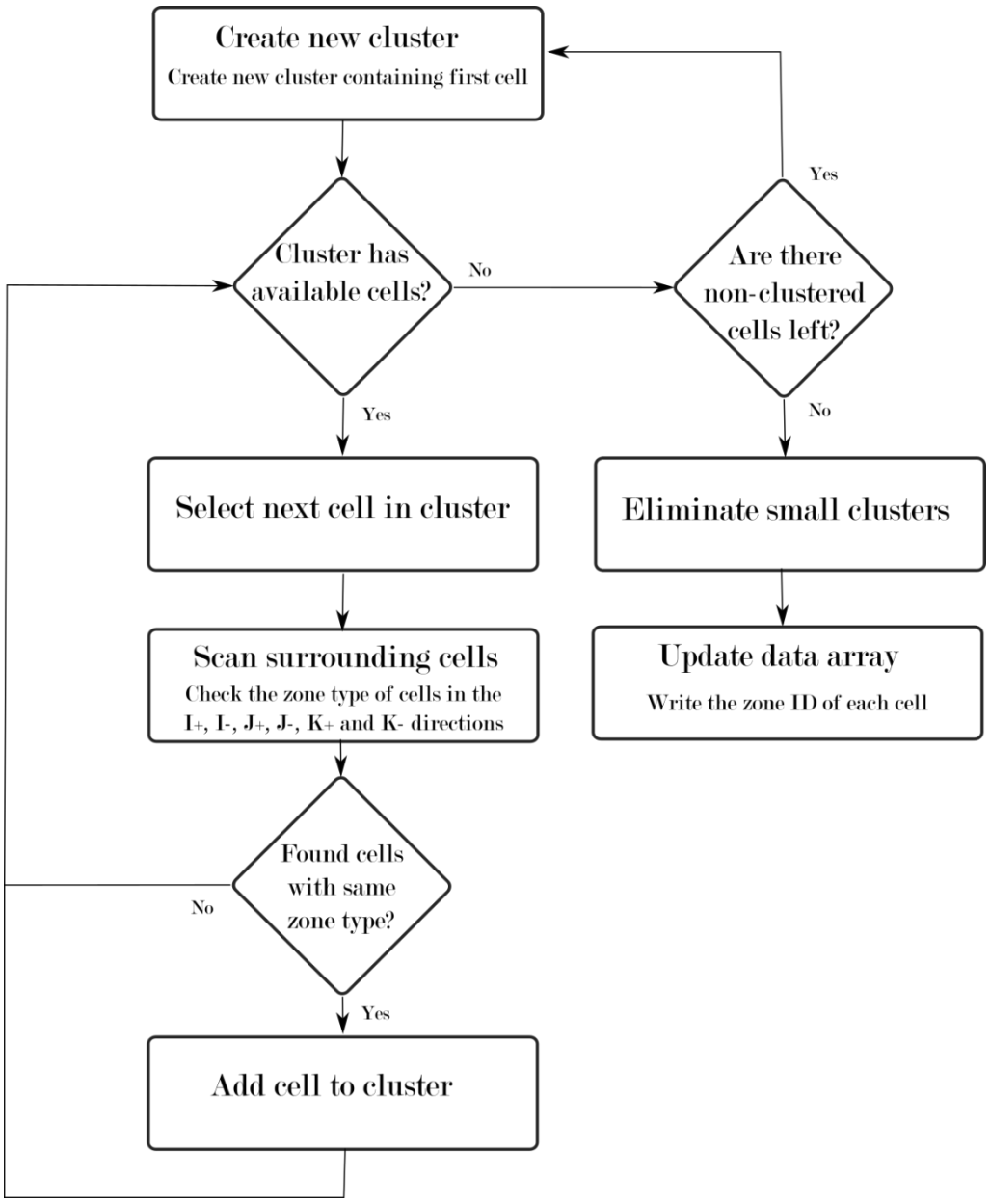


Figure 3-14: Flowchart of the clustering method

First, a new empty array is created to keep track of the clusters and the cells that belong to each of them. The code starts at the cell with index (I=0, J=0, K=0). It creates a new cluster ID and a new column in the clusters array, and updates the cell-centred array by changing the *"zone"* variable of cell (0, 0, 0) with the new cluster ID. Then, the code searches for other cells in 6 directions corresponding to the 6 faces of the cell (I+, I-, J+, J-, K+, K-). An index check prevents the code from trying to access invalid indices (*index* < 0 or *index* > index_{max}). If one of the 6 cells belong to the same zone type and if it does not belong to a cluster yet, it is added to the cluster, the cluster array is updated with the new cell, and the cell's *"zone"* variable is updated in the cell-centred array. When the 6

surrounding cells have been checked, the algorithm passes to the next cell of the cluster array and scans its 6 surrounding cells. As long as the algorithm finds new cells of identical zone type, it will keep adding cells to the cluster array. When it reaches the last cell of the cluster array and does not find new suitable cells, it considers the cluster as complete and creates a new one. A new column is added to the cluster array, and the next available cell is added to that new cluster. A simplified 2-dimensional representation of the clustering method is shown in Figure 3-15.

Due to the large number of cells in a CFD simulation (hundreds of thousands, if not millions), it is frequent that this method produces a number of large clusters representing the main features of the temperature distribution as well as many small clusters containing in the order of 100 cells each. These very small clusters are artefacts of the clustering method since clustering considers position and zone type, but zone types are defined regardless of the position of a cell. These small clusters do not contribute to the accuracy of the final zonal model, but they do add a significant computational cost in all the subsequent steps. A post-processing step is necessary to eliminate very small clusters which consist of a negligible amount of cells, which is done iteratively. After all the cells in the domain have been assigned to a cluster the mean temperature, pressure and density of each cluster are computed and recorded for later usage during model parametrization (Section 3.6). During this process, the cell-centred array is updated to include the cluster ID of each cell.

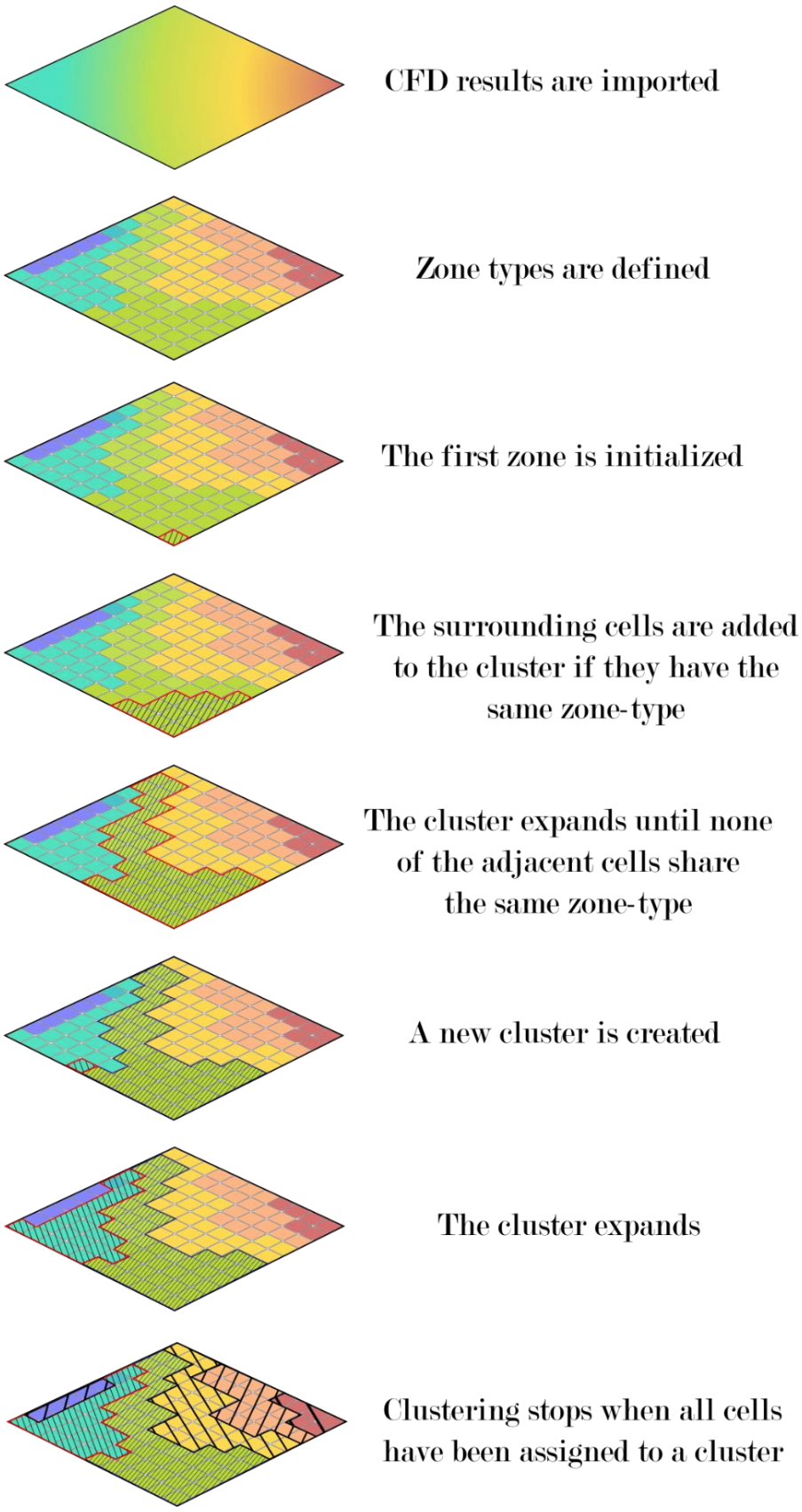


Figure 3-15: Graphical step by step representation of the MVS cell clustering

3.5.2 Coarse Grid

The second method developed while investigating cell clustering is the interpolation of the initial CFD mesh to a coarse mesh. Since PHOENICS uses a cartesian structured mesh, the definition of a coarse mesh is straightforward: (1) the user defines the number of desired cells in the coarse grid in the *I*, *J* and *K* directions; (2) CFD cells are assigned to a cluster depending solely on their coordinates. Figure 3-16 shows a simplified 2D representation of the clustering process.

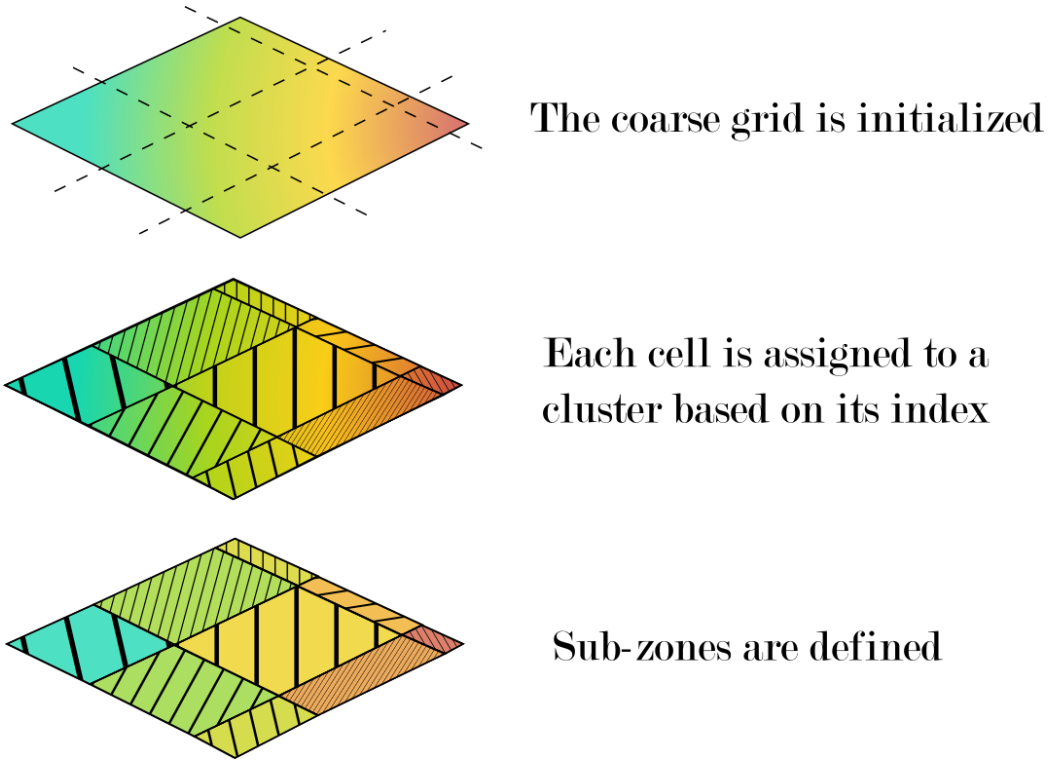


Figure 3-16: Graphical step by step representation of the coarse grid clustering

When the number of cells has been defined for the coarse grid, the code computes the exact coordinates in metres of each plane used to divide the domain in a coarser grid. Then, the plane's coordinates are matched to the actual CFD grid so that a CFD cell cannot be cut in two. After this step, the code iterates through each cell of the CFD domain and compares its coordinates to the ones defined for dividing the domain. Depending on their coordinates, the cells are immediately assigned to a cluster. Finally, the mean values

of temperature, pressure and density are computed for each cluster and recorded for later usage during model parametrization.

3.5.3 Classic Watershed

The Classic Watershed (CW) is a method used in image processing for image segmentation initially presented by Soille and Vincent [61]. It has been adapted by the author to ROM generation because its principle is very similar to cluster generation. In fact, the CW algorithm is mainly used for grayscale image segmentation, but its fundamental basis can be used for any n-dimensional dataset.

The CW algorithm comprises three steps shown in Figure 3-17. First, the local minima and maxima are detected and used as initial seeds for the cell clusters. Then, the "water level" or tolerance to include a neighbouring cell to a cluster is iteratively raised. In this work, the tolerance was the ΔT between the seed and other cells' temperature. When all the cells have iteratively been added to an existing cluster, "watersheds" are raised i.e. the clusters are considered complete.

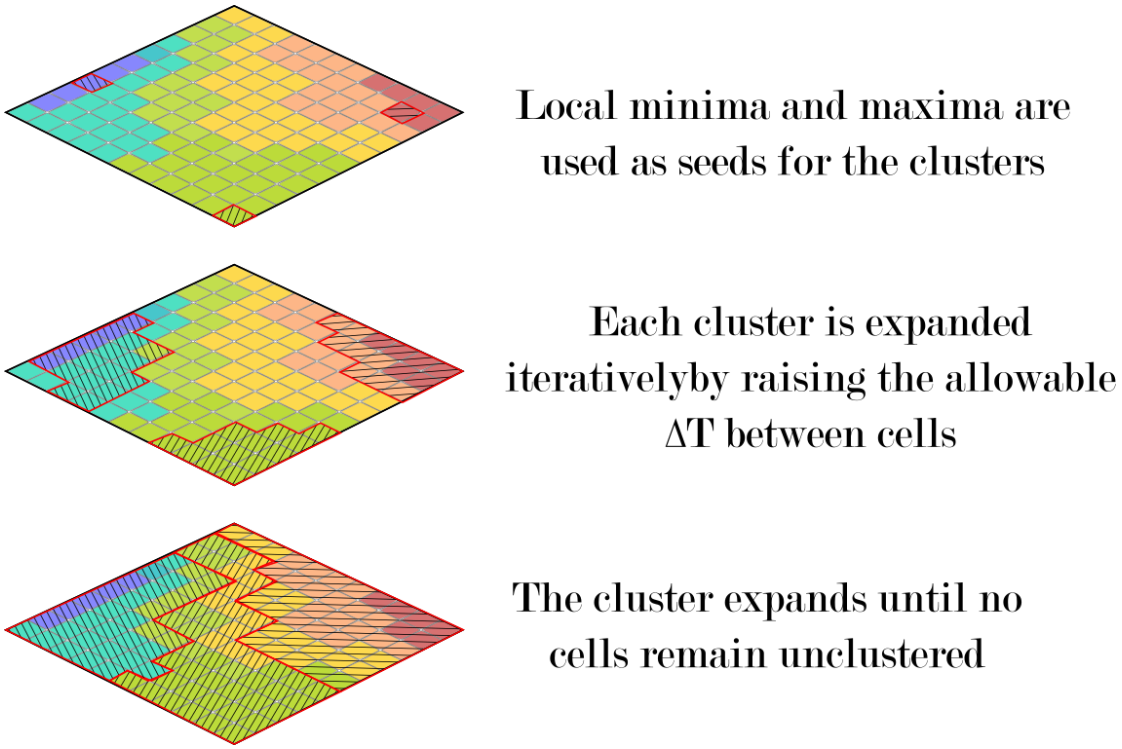


Figure 3-17: Graphical step by step representation of the classic watershed cell clustering

3.5.4 Sub-zone assumptions

Once cell clusters have been formed, they are considered as being a sub-zone, or a portion of the fluid domain which is considered uniform for the purpose of the subsequent calculations on the simplified model. Sub-zones have therefore uniform temperature, pressure and density. These values are calculated by computing the mean value of each parameter on the cells composing the sub-zone. By selecting cells that have similar properties, clustering them into sub-zones and assuming their uniformity allows a significant reduction of the computational power needed to solve the zonal model, while ensuring that the thermal distribution of the domain is represented with a good accuracy.

3.5.5 Summary of sub-zone generation

This section has introduced the three clustering methods developed for this work. The first method is the MVS clustering algorithm, which defines a set of temperature intervals from the iterative division of the dataset along the mean value of the temperature. The second method is a coarse grid division which corresponds to the method used for many zonal and compartment models as shown in 2.4.2. Finally, the CW algorithm was adapted from the eponymous image segmentation method and relies on the selection of initial seeds for sub-zones and the iterative raise of the tolerance of each sub-zone to the inclusion of new cells with a different temperature. A comparison of the level of error against CFD and the time taken to generate a zonal model with each method is shown in Section 5.2.

3.6. Model parametrization

This section will introduce the steps taken to generate a zonal model based on the information extracted from the clustering steps seen in Section 3.5. The physical interactions considered in the model are boundary conditions for inlets and outlets (mass flow rates and temperatures), heat transfers at domain boundaries, and mass exchange between sub-zones.

3.6.1 Introduction

After clustering computational cells together and considering them as sub-zones of the zonal model, it is necessary to compute the interactions between sub-zones and between the sub-zones and the various boundary conditions detected during the import of CFD data.

As a reminder, at this stage the available information is: (1) the list of boundary conditions, their type and their position (Section 3.4.2) and (2) the cell-centred array containing information on the cell clusters (Section 3.5).

Two new 2D arrays will be generated at this step: one keeping track of mass flow rates in and out of each sub-zone (called *intMFR* for "Interface Mass Flow Rates") and one keeping track of areas of exchange (*intAreas* for "Interface Areas"). Both arrays have dimensions $A \times B$ with A the number of sub-zones and B the number of sub-zones plus the number of boundaries (shown in Figure 3-18).

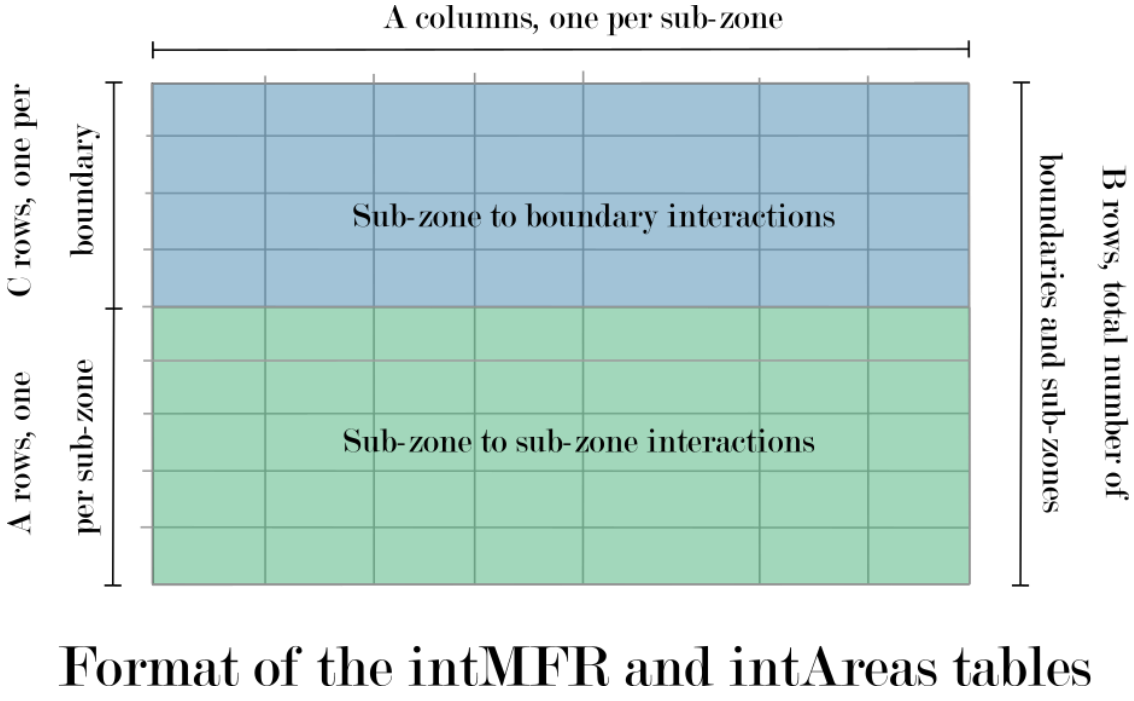


Figure 3-18: Format of the intMFR and intAreas tables.

3.6.2 Mass flow rates and interface areas

In order to compute the mass exchange between sub-zones, the mass flow rates at the cell faces composing the interface between two sub-zones are summed to define a total mass flow rate between two sub-zones at their interface as seen in Figure 3-19.

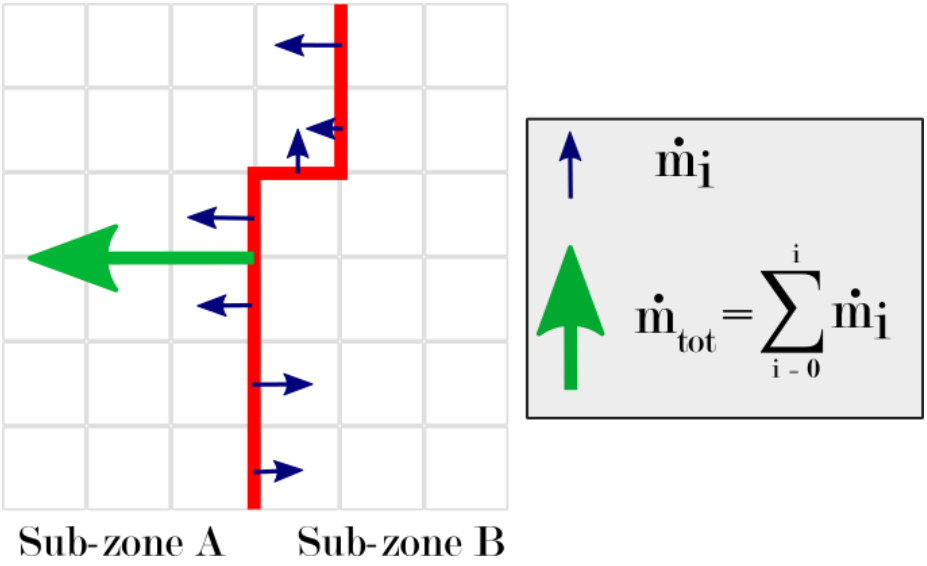


Figure 3-19: Interface between sub-zones. The total mass flow rate between sub-zones is computed from the unitary mass flow rates of the cells composing the interface

This section of the code processes the interface areas at the same time, which will be used in two calculations: the contact area between two sub-zones as well as the contact area between a sub-zone and the domain boundaries.

The code starts by iterating over all the cells of the CFD domain. First, each cell is checked to ensure that it is a fluid cell by reading the corresponding value in the solids flag array. If it is indeed a fluid cell, the code will check turn by turn all 6 cells in the *I+*, *I-*, *J+*, *J-*, *K+* and *K-* directions and verify two values in the cell-centred array: the "zone" value and the "boundary" value. Three outcomes are then possible:

- If the cells have a different "zone" value (i.e. they do not belong to the same sub-zone), the *intMFR* table is updated: depending on the orientation of the face between the two cells (North, East or High) the cell value for *CNN*, *CNE* or *CNH* in $kg.s^{-1}$ is added to the current value for the total mass flow rate between the two sub-zones. If these sub-zones are number *A* and *B*, the value is contained in the *intMFR* table in column *A* and row *B+number of boundaries*.

The area in squared metres of the contact face is computed as well and added to the current value for contact area between the two sub-zones in the *intAreas* table in the same manner as for the *intMFR* table. The code then repeats the process for the rest of the neighbouring cells.

- If the cell is marked as belonging to a boundary, the contact area between the cell and the boundary is computed and is added to the value of the corresponding cell in the *intAreas* array. Additionally, the *HTCO* value in the cell-centred array and temperature values in the node-centred array are recorded for later usage, in case the flow over the surface is turbulent as described in Section 3.4.2.
- If none of the above apply, the code moves on to the next cell

After all the cells have been verified, the *intMFR* and *intAreas* tables will contain information respectively on the mass flow rates and on the interface area between sub-zones and between sub-zones and boundaries.

3.6.3 Boundary conditions

Three types of boundary conditions are considered in the model: mass exchange through inlets and outlets, heat exchange through the walls, and heat input from other constant heat sources. In PHOENICS CFD the inlets and outlets are defined by a surface over which the mass flow rate is distributed uniformly. Therefore, the zonal model also assumes that the mass is exchanged uniformly over the inlet/outlet surface.
As a result, if one sub-zone only is in contact with an inlet/outlet then the entirety of the mass will be exchanged with that sub-zone, but if several are in contact with the boundary then the mass flow rate in each sub-zone will be calculated as follows:

$$\dot{m}_{boundary \rightarrow sub-zone} = \frac{A_{boundary- \, sub-zone}}{A_{boundary}} \times \dot{m}_{boundary \rightarrow domain} \qquad \textit{Equation 5}$$

With $\dot{m}_{boundary \rightarrow sub-zone}$ the mass flow rate from the boundary to a sub-zone, $A_{boundary/sub-zone}$ and $A_{boundary}$ the interface area between the boundary and the sub-zone and the total area of the boundary respectively, and $\dot{m}_{boundary \rightarrow domain}$ the total mass flow rate from the boundary to the domain. When summed together, all the mass

flow rates from the boundary to each sub-zone in contact with it will equal to the total mass flow rate between the boundary and the domain (Figure 3-20).

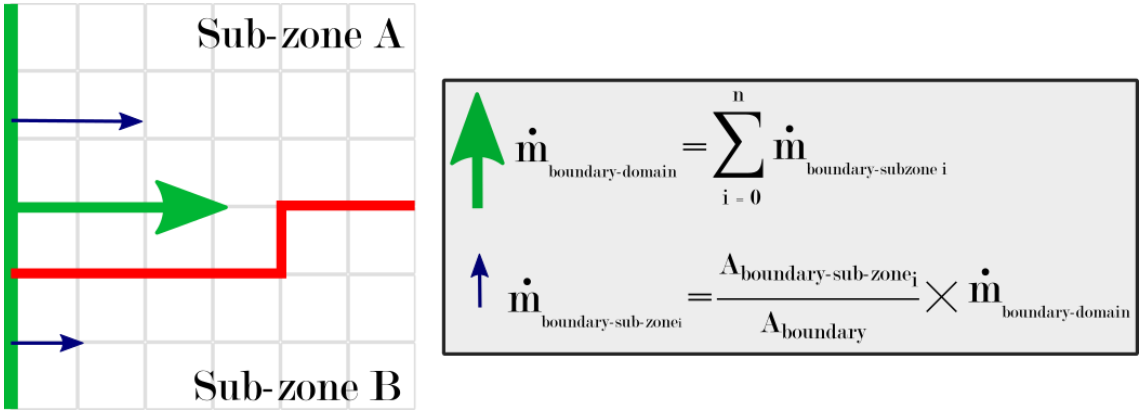


Figure 3-20: If a boundary has an interface with more than one sub-zone, the total mass flow rate from the boundary is passed to each sub-zone proportionally to the surface of each interface

Heat exchange between the walls (or any non-adiabatic surface) and the domain is treated by processing the corresponding *UA* value. The *UA* value corresponds to the overall heat transfer coefficient times the exchange area. The overall heat transfer coefficient is directly extracted from the CFD results as explained in Section 3.4.2, and the exchange area has been extracted in the previous step as explained in Section 3.6.2. When the *HTCO* value is available, then the flow is turbulent and the *HTCO* value available in the cell-centred array is used instead. The mean *HTCO* value is computed from the *HTCO* values of the cells at the interface between the sub-zone and the boundary, and it is used as the *U* value.

Finally, other heat inputs such as heat from people or from computers are treated as constant heat sources. The key assumption on heat sources is that the heat input is distributed uniformly over the sub-zones in contact with the heat source. The heat input into each sub-zone is computed in a similar fashion as for the mass flow rates, as follows:

$$\dot{Q}_{boundary \rightarrow sub-zone} = \frac{A_{boundary-\ sub-zone}}{A_{boundary}} \times \dot{Q}_{boundary \rightarrow domain} \qquad \textit{Equation 6}$$

With $\dot{Q}_{boundary \rightarrow sub-zone}$ the heat transfer rate from the boundary to a sub-zone, $A_{boundary/sub-zone}$ and $A_{boundary}$ the interface area between the boundary and the sub-zone and the total area of the boundary respectively, and $\dot{Q}_{boundary \rightarrow domain}$ the total heat rate from the boundary to the domain. When summed together, all the heat rates from the

boundary to each sub-zone in contact with it will equal to the total heat rate between the boundary and the domain (Figure 3-21).

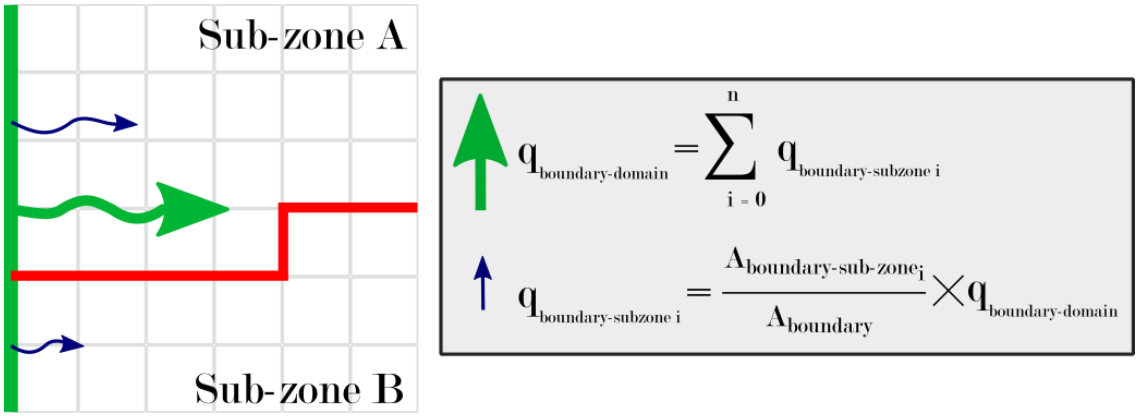


Figure 3-21: If a boundary has an interface with more than one sub-zone, the total heat exchange from the boundary is passed to each sub-zone proportionally to the surface of each interface

3.6.4 Summary of the extraction of boundary conditions

This section has shown how heat and mass exchange between sub-zones and between sub-zones and the domain are extracted and compiled for parametrizing the zonal model. Mass flow rates between sub-zones are extracted directly from CFD data and are summed up to compute the total mass flow rate between two sub-zones. The parameters for heat exchange are extracted from the CFD results file, and when they are associated to the values for interface areas they allow the computation of heat exchange between the boundaries and sub-zones. At this stage, the available information on the model is:

- The temperature, pressure, density and volume of each sub-zone
- The mass flow rates in and out of each sub-zone
- The interface area between sub-zones and between sub-zones and the boundaries
- The heat transfer coefficients between boundaries and sub-zones
- Mass flow rates between inlets/outlets and sub-zones
- Heat exchange between constant heat sources and sub-zones

The next section will explain how this data is compiled into a zonal model.

3.7. Fluid network model

The previous sections have shown how sub-zones were extracted from CFD simulations along with data on the heat and mass exchange between sub-zones and between sub-zones and the domain. At this stage, all the data necessary to compile the zonal model is available and the last step is to generate an input file for the solver. This section will present the features available in Sinda/FLUINT to build a fluid network model, and the steps taken to compile the available data into a format that is readable by Sinda/FLUINT.

3.7.1 Sinda/FLUINT solver

Sinda/FLUINT has been introduced in Section 3.3.3. This section will present more specifically the features available to the user to build and solve a steady-state fluid network model (Figure 3-22). Sinda/FLUINT models are built on two main elements: the fluid submodel and the thermal submodel. Fluid submodels include all the features of the simulated fluid system: the definition of each sub-zone and the energy and mass exchange between sub-zones. The thermal model incorporates the features for thermal exchange. For the scope of this work, the thermal submodel is used to represent the thermal boundaries of the domain and the heat exchange between the boundaries and the fluid domain.

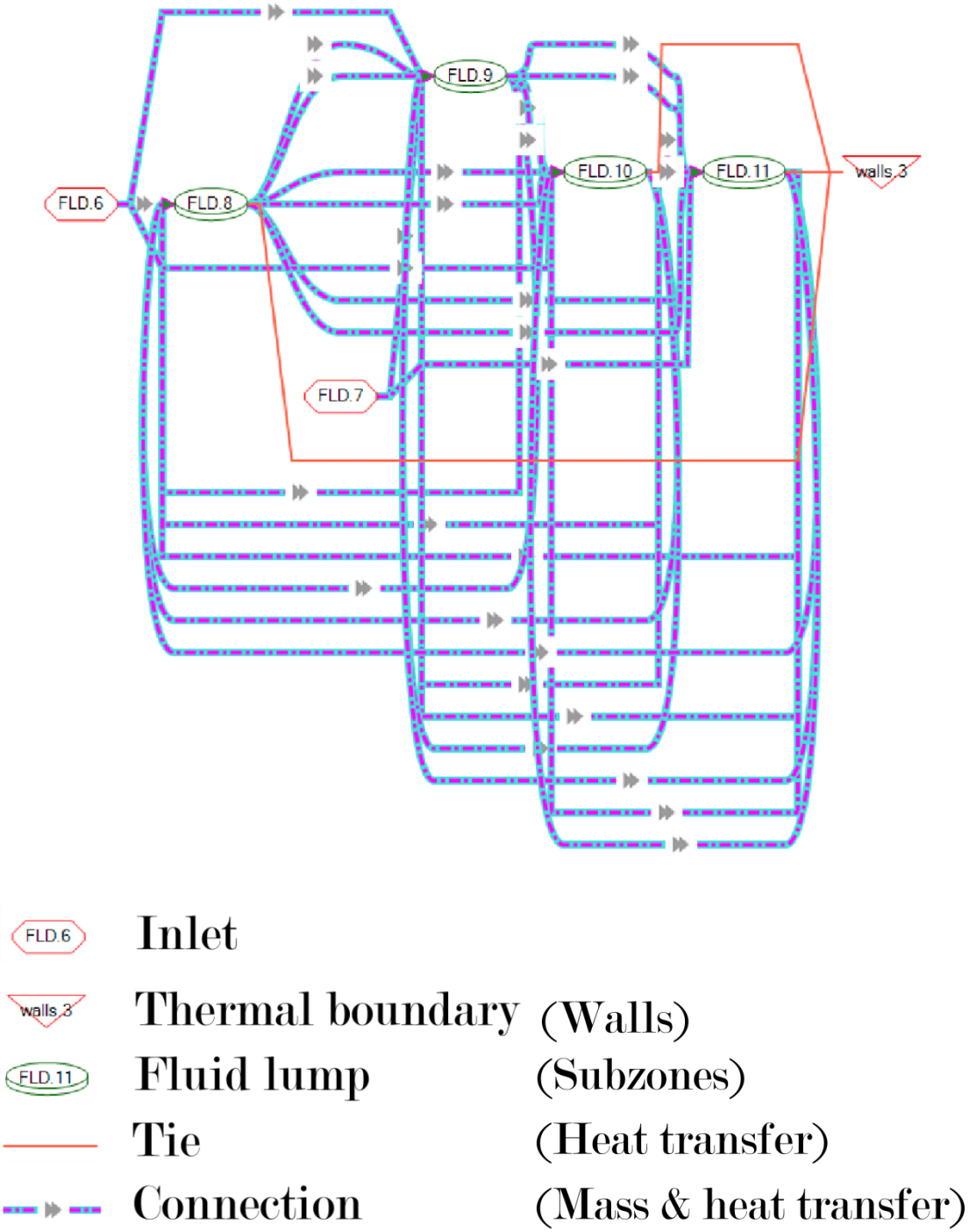


Figure 3-22: 4-sub-zones fluid network model in Sinda/FLUINT's GUI tool Sinaps, with the fluid lumps representing the uniform sub-zones, 1 inlet and 1 outlet, 1 thermal boundary and the corresponding ties (thermal to fluid heat exchange) and connections (fluid to fluid mass and heat exchange).

3.7.1.1 Thermal submodel

The thermal submodel defines the boundaries of the model that exchange heat with the domain's fluid through convection. The boundaries include walls, windows and other non-adiabatic objects in the domain. They are assumed to have a constant and uniform temperature identical to the properties of the corresponding boundaries in the CFD simulation.

In order to define the thermal submodel, it is necessary to describe the properties of each of its components. Each boundary must be defined by a unique number for identification, and its temperature. Each of these new elements is from this point on called a *node* and represents only thermal elements. First, a header for the thermal submodel must be declared. It is written as follows:

HEADER NODE DATA, **NAME**

With *NAME* the name of the thermal submodel.

A node is input using the following format:

#N, T, C

With *#N* being an integer used to identify the node, preceded by the sign – to identify it as a thermal node; *T* the temperature of the node, and *C* the thermal capacitance of the node. By defining a capacitance of 0 $J.K^{-1}$ the solver assumes that the node has an infinite capacitance, thus a constant temperature throughout the simulation.

An example of node declaration is given in Figure 3-23 below.

```
HEADER NODE DATA,walls
C DOM_YMIN_W
        -1,301.150000,0.0
C DOM_YMAX_W
        -2,301.150000,0.0
C DOM_ZMIN_W
        -3,301.150000,0.0
C DOM_ZMAX_W
        -4,301.150000,0.0
C XMIN
        -5,301.150000,0.0
C PLAT6
        -6,301.150000,0.0
```

Figure 3-23: Example of a thermal model declaration in Sinda/FLUINT. This thermal submodel consists of 6 nodes, the lines starting with C indicate a comment used to identify the nodes by name for debugging

3.7.1.2 Fluid submodel

The fluid submodel is composed of two core elements: *tanks* and *plena*. Tanks are finite volumes of fluid, while plena are infinite volumes of fluid. Tanks are used in this study to represent the sub-zones extracted from the CFD simulations. They have several properties: volume, density, pressure and temperature. Plena are used in this study to represent inlets and outlets; they share the same properties as tanks except for volume which is considered infinite.

First, a header declaring the fluid submodel is necessary. It is written in the following format:

HEADER FLOW DATA, **FLD**, FID = **FLDID**

With *FLD* the name of the fluid submodel and *FLDID* the fluid identifier, which in this study is air as an ideal gas (ID = 8729).

In order to declare a tank, the following format is used:

LU TANK, **#N**, **TL**, **VOL** [optional: **QL**, **PL**, **DL**, …]

With *#N* being an integer identifying the tank, *TL* the tank temperature, *VOL* the volume, *PL* the pressure, *DL* the density, and *QL* an additional heat input in the tank.

Similarly, plena are declared as follows:

LU PLEN, **#N**, **TL** [optional: **PL**, **DL**, …]

With the same input data as for tanks, except for the volume which is considered infinite. An example of tank and plena declaration is given in Figure 3-24.

```
HEADER FLOW DATA,fld,FID=8729
LU TANK,8
        TL = 301.15
        PL = 101325.0
        VOL = 64.000000
        QL = 0.0
C  Inlet: INLE7
LU PLEN,6
        TL = 301.150000
        PL = 101325.0
```

Figure 3-24: Example of a fluid submodel declaration in Sinda/FLUINT. This fluid submodel consists of a tank and a plenum

In Sinda/FLUINT, conservation of energy and mass of is shown in Equation 7 and Equation 8 below:

$$\sum_i e_i \times \dot{m}_i = \frac{dm}{dt} \qquad \textit{Equation 7}$$

$$\sum_i e_i \times h_i \times \dot{m}_i + \dot{Q} - p \times (\dot{v} + \frac{dp}{dt} \times V) = \frac{du}{dt} \qquad \textit{Equation 8}$$

With e_i the direction of the flow (± 1), $\dot{m}_i$ the mass flow rate, m the mass of the tank, h_i the donor enthalpy, $\dot{Q}$ the heat transfer rate, p the pressure, $\dot{v}$ the volume rate of change, V the volume, and u the internal energy. The volume V and heat transfer rate $\dot{Q}$ are set by the Python code as the total volume of a zone and the heat transfer rate $\dot{Q}$ in or out of the lump (for example, a heat source) as seen in Equation 6, while the pressure p is initialized

as the mean pressure in a zone computed from CFD. The other variables are computed by the solver.

3.7.1.3 Connections

Connections are used to define the interactions between fluid lumps, they are therefore used in this study to parametrize the mass exchange between sub-zones and between inlets/outlets and sub-zones. They are characterized by two main parameters: the mass flow rate and area. These two parameters are taken from the previous step described in section 3.6.2. The mass flow rate in a connection is prescribed and constant. A connection is declared as follows:

PA CONN, **#N**, **L1**, **L2**, **FR**, **DEV**, **AF**, **smfr**

With *#N* the unique identifier of the connection, *L1* and *L2* the lumps (tank or plenum) it connects, *FR* the initial mass flow rate, *DEV* the type of device (in this work, *MFRSET* for fixed mass flow rate), *AF* the area, and *smfr* the prescribed mass flow rate. An example is given in Figure 3-25.

Connections are governed by Equation 9 (mass balance) and Equation 10 (energy balance).

$$\sum_i e_i \dot{m}_i = 0 \qquad \textit{Equation 9}$$

$$\sum_i e_i \, h_i \dot{m}_i = 0 \qquad \textit{Equation 10}$$

Where i indicates the upstream or downstream end of the connection, e_i is the flow rate rectifier (+1 if upstream, -1 if downstream), $\dot{m}_i$ is the mass flow rate and h_i is the donor enthalpy.

Ties are used to define heat exchange between thermal nodes and fluid lumps. The solver requires the input of the overall heat transfer coefficient times the exchange area, or *UA* value, which is computed as described in section 3.6.3. The assumption used in this study is that a tie has a constant *UA* value, which is used to compute the heat exchange between a wall and a fluid lump as described by Equation 11. The constant UA assumption is due

to a limitation of the Sinda/FLUINT solver which offers only a limited number of options for heat transfer in the case of natural convection, a variable UA is only available for forced convection.

$$\dot{Q}_{tie} = UA_{tie} \times (T_{node} - T_{lump}) \qquad \textit{Equation 11}$$

Where $\dot{Q}_{tie}$ is the heat transfer rate between the thermal node and the fluid lump through a tie, UA_{tie} the overall heat transfer coefficient times the area of exchange between the node and the lump, T_{node} the temperature of the node and T_{lump} the temperature of the lump. The solver assumes that the wall has a uniform temperature T_{node} and the fluid's bulk temperature is that of the lump T_{lump}.

In Sinda/FLUINT, a tie is declared as follows:

T HTU, **#N**, **L1**, **N1**, **UA**

With *#N* the tie identifier, *L1* and *N1* the lump and node it connects respectively, and *UA* the fixed *UA* value for the heat exchange. An example of tie declaration is given in Figure 3-25.

```
PA CONN,2,7,8,
        FR = -0.12
        DEV=MFRSET
        AF = 3.139048
        smfr = -0.12
T HTU,1,8,walls.1,
        UA = 1.505
```

Figure 3-25: Example of connection and tie declaration in Sinda/FLUINT

3.7.2 Input file

Sinda/FLUINT's input file structure is described in this section. First, the thermal nodes are declared along with their properties, then the fluid lumps, and finally the connections and ties. As a last step, the parameters for the solver must be specified.

To generate the Sinda/FLUINT input file, the code starts by initializing a blank text file. It writes the compulsory header, in which the only variable is the model name. It declares

the header for the thermal submodel and proceeds to write the thermal nodes. It first assigns a unique number to a thermal boundary which corresponds to the identifier defined in section 3.4.2.3. Depending on the information extracted in previous steps, it includes the temperature directly extracted from the results file or it includes the temperature processed from the mean temperature of the CFD nodes at the boundary. It repeats this process until all thermal boundaries have been set and records the identifier of each boundary for later use when defining ties.

Then, the code declares the fluid submodel header and proceeds in a similar manner on declaring all the fluid lumps. The code first assigns a unique identifier to a lump equal to the unique ID assigned during the clustering process, and defines its parameters depending on the sub-zone properties computed earlier as described in section 3.5.4. It includes the temperature of the fluid lump, its volume, and finally its density and pressure. The code repeats the operation for all the sub-zones defined earlier.

Then, the code declares the connections header and iterates through the *intMFR* table to write each connection between fluid lumps. It iterates through the table because as stated earlier, each cell records the value of mass flow rate between sub-zone number (row - geometries) and sub-zone number (column). If the cell of the *intMFR* table has a non-zero value, it signifies that there is a mass exchange between the two fluid lumps. A unique identifier must be defined for each connection, along with the identifiers of the two fluid lumps linked by that connection, which are taken from the row and column indices of the cell containing the mass flow rate value. Finally, the connection is characterized by a mass flow rate and the area of the connection. This information is taken respectively from the *intMFR* and *intAreas* arrays populated earlier.

Finally, the fluid network model is completed by declaring all the ties between thermal boundaries and fluid lumps. This step is similar to the precedent: the code iterates through the *intAreas* array except it is limited to rows 0 through (*number of boundaries*) to detect only the cells representing a connection between a boundary and a sub-zone. If that cell has a non-zero surface and a null mass flow rate, it signifies that the thermal boundary has a tie with the fluid sub-zone. Therefore, the code declares a new tie between boundary (row) and sub-zone (column). It assigns the tie a unique id and a *UA* parameter taken from the data extracted in section 3.6.3.

3.7.3 Summary of fluid network model generation

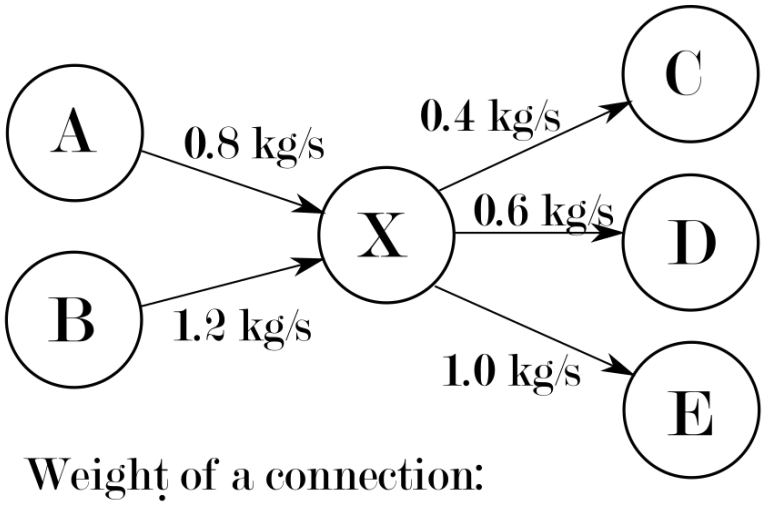
This section has described how the model parameters extracted from the CFD can be translated into a zonal model, in the form of a fluid network model solvable by Sinda/FLUINT. The sub-zones are translated into fluid lumps, with finite volume and initial temperature; inlets and outlets are translated into fluid lumps of infinite volume and prescribed temperature; thermal boundaries are translated into thermal nodes with fixed temperature; and finally, the heat and mass exchanges are translated respectively into ties with fixed *UA* and connections with fixed mass flow rates. Once the model has been defined, the Sinda/FLUINT solver outputs a result file displaying the steady-state temperature of each fluid lump and thermal node.

3.8. Changing the model parameters

When the mass flow rates at the domain inlets and outlets are changed, the reduced-order model becomes unbalanced because at this stage the mass exchanges between sub-zones are identical to those of the original case, and changing the mass flow rates only at the inlets and outlets unbalances the model. Before solving the newly defined system it is necessary to rebalance the system with regards to mass exchange. Since there is no information on the flow shape in the new configuration, the assumption made during rebalancing is that the mass exchange between sub-zones remains proportional to the original one.

In order to recalculate the mass exchange while maintaining the same proportions as in the original model, a simple algorithm has been written to iteratively balance the mass exchanges between sub-zones. First, a map of the current mass exchanges between sub-zones is generated. It records the proportion of mass coming in each sub-zone that flows in each of the next sub-zones as a weight factor. Figure 3-26 illustrates the process of defining the weights for each mass exchange between sub-zones. It illustrates an example where a sub-zone is connected to 2 sub-zones upstream and 3 downstream. In that example, the mass exchange map will record that 20% of the mass flows into sub-zone C

from sub-zone X, 30% into D and 50% into zone E, and that 40% of mass is coming from sub-zone A into X and 60% from sub-zone B.



Weight of a connection:

$$W_{xy} = \frac{\dot{m}_{xy}}{\dot{m}_{tot(in/out)}}$$

In this example:

Connection	A-X	B-X	X-C	X-D	X-E
Weight	0.4	0.6	0.2	0.3	0.5

Figure 3-26: Example of weight assignment for a sub-zone X connected to 6 other sub-zones. Each weight is computed as the ratio between the mass flow rate at the connection and the total mass flow rate entering or exiting the sub-zone

Then, the algorithm updates the mass flow rate at the inlets and outlets with the new values requested by the user. Finally, the mass exchange is balanced iteratively. To achieve this, at each step the algorithm will update the mass entering each sub-zone, compute the mass imbalance, compute the new mass exiting each sub-zone and updates the *IntMFR* table accordingly. Equation 12 shows how new upstream mass flow rates are computed from the downstream mass flow rate and the connection weights defined earlier. Equation 13 shows how the mass imbalance is computed for each sub-zone.

$$\dot{m}_{out,i} = w_i \times \sum_{j=0}^{n} \dot{m}_{in,j} \qquad \textit{Equation 12}$$

where $\dot{m}_{out,i}$ is the upstream mass flow rate through connection i, w_i is the weight of connection i, and $\dot{m}_{in,j}$ is the downstream mass flow rate through connection j.

$$\varepsilon_i = \sum_{j=0}^{n} \dot{m}_j \qquad \text{Equation 13}$$

where ε_i is the mass imbalance of sub-zone i and $\dot{m}_j$ is the mass flow rate through connection j.

One execution of this routine is not sufficient to obtain a balanced model, so at this stage the algorithm computes the total mass imbalance over all the sub-zones and will repeat the mass-balancing step until the total imbalance has been reduced to under 0.1 g/s. Figure 3-28 shows a flowchart of the mass rebalancing algorithm, and Figure 3-27 shows the rebalancing of a simplified network where sub-zones are arranged in series. The sub-zone generation method presented in this thesis typically generates sub-zones that are connected in a more complex arrangement.

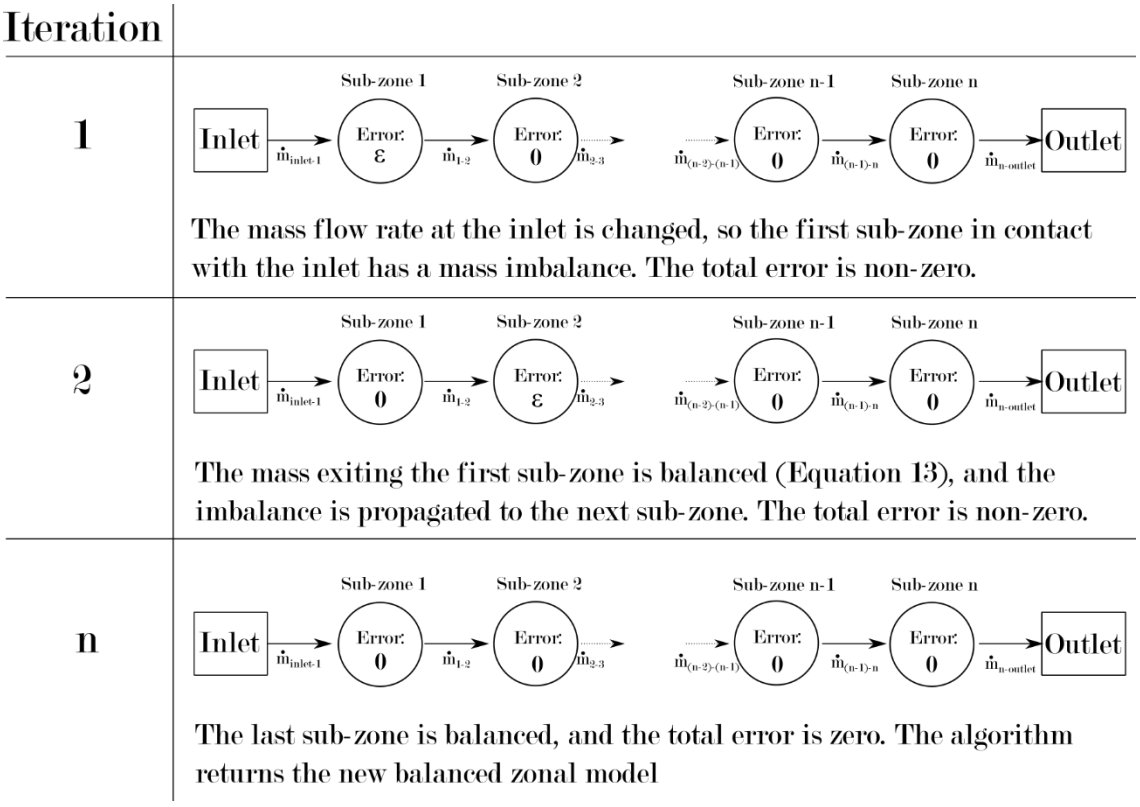


Figure 3-27: representation of the mass rebalancing algorithm on a simplified network of sub-zones in series

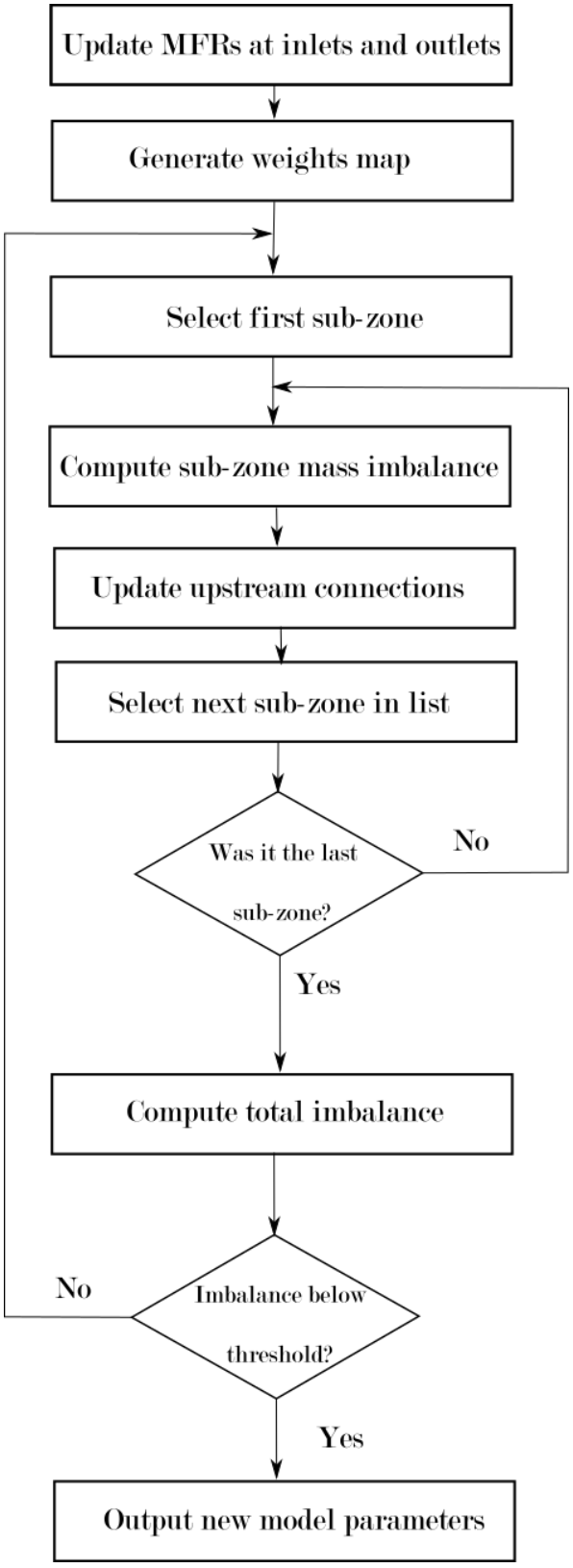


Figure 3-28: Flowchart of the iterative mass rebalancing algorithm when the mass flow rates at the inlets or outlets of the model are modified

3.9. Results output

Once the model has been solved, Sinda/FLUINT outputs a steady-state prediction of the temperature of each lump. This information is then mapped back to the original CFD domain to obtain a prediction of temperature distributions. Additionally, this step is used to quantify the temperature difference between the zonal model's predictions and the CFD predictions. This section will therefore describe both the error quantification and the transposition of the results to the original CFD domain.

3.9.1 Error quantification

In this study, the error between the temperature prediction of the zonal models and that of the corresponding CFD simulations has been quantified using the *Weighted Mean Absolute Error (WMAE)*. This method offers a useful information on the accuracy of the method relative to the CFD predictions, while not needing a reference temperature like the *Weighted Mean Average Percentage Error* which must be defined case by case. The error is weighted to account for varying cell volumes, so that the error is proportional to the volume the cell occupies in the domain. The calculation for the WMAE is shown in Equation 14:

$$WMAE = \sum_{i=0}^{N} \frac{|T_i - T_{CFDi}|}{V_{domain}} \times V_i \qquad \textit{Equation 14}$$

With T_i the temperature of cell i, computed when solving the zonal model, T_{CFDi} the temperature of cell i computed with CFD, V_i the volume of cell i and V_{domain} the total volume of the domain.

3.9.2 Remapping of sub-zone properties to the CFD domain

Since Sinda/FLUINT solves the model as a fluid network, it outputs a set of properties for each sub-zone. In order to obtain a 3-dimensional information on the temperature distribution, the new value of temperature in each sub-zone is mapped to the cells of the original CFD domain.

Once the results have been obtained after solving the Sinda/FLUINT model, the code will initialize a blank file used to store the results of the zonal model solution. The results file is written in the *Tecplot* format and therefore respects the formatting described in section 3.4.1: first, a header defines the number of cells in each dimension and the name of the variables available for each cell. In this case, the code takes the domain dimensions from the size of the cell-centred data array, and the variables recorded in the file are 6, for visualization purposes: the *X*, *Y* and *Z* coordinates of each cell, *T* the cell temperature computed from the zonal model, *TCFD* the original CFD temperature of the cell and *AEI* the absolute error of temperature for each cell. By doing so, it is possible to visualize the temperature distributions obtained from solving the zonal model but also information on the local temperature error for debugging purposes.

3.10. Chapter summary

This chapter has given details on the methodology developed in this doctoral thesis. First, it introduced an overview of the method and described the software used for CFD simulations, for the zonal model extraction, and finally for the model's solution. Then it described the steps taken to extract the necessary information on fluid properties and boundary conditions from the CFD results, or how the fluid data could be extracted directly from CFD while the objects of the domain and their respective boundary conditions must be extracted from a file destined to be used by the CFD solver. Subsequently, it introduced the three clustering methods that have been developed for clustering CFD computational cells together in order to define sub-zones for the zonal model and proposed a method to extract quasi isothermal volumes of fluid from a 3-

dimensional domain and therefore define a small number of sub-zones. Then it described the steps taken to translate the information extracted from CFD and from the clustering step into a zonal model, or how the sub-zones have been generated from the clusters and how the boundaries between sub-zones and between sub-zones and the domain have been characterized. Finally, it showed how the fluid network model was input to the solver through a text-based input file. Additionally, the method for quantifying the error induced by this method has been defined, through the use of WMAE. A flowchart of the method is shown in Figure 3-29. It highlights the main steps from the CFD data extraction to the solution of a zonal model. The next chapter will present the case studies to which the proposed method has been applied in order to assess its performance in extracting zonal models from CFD simulations of the built environment.

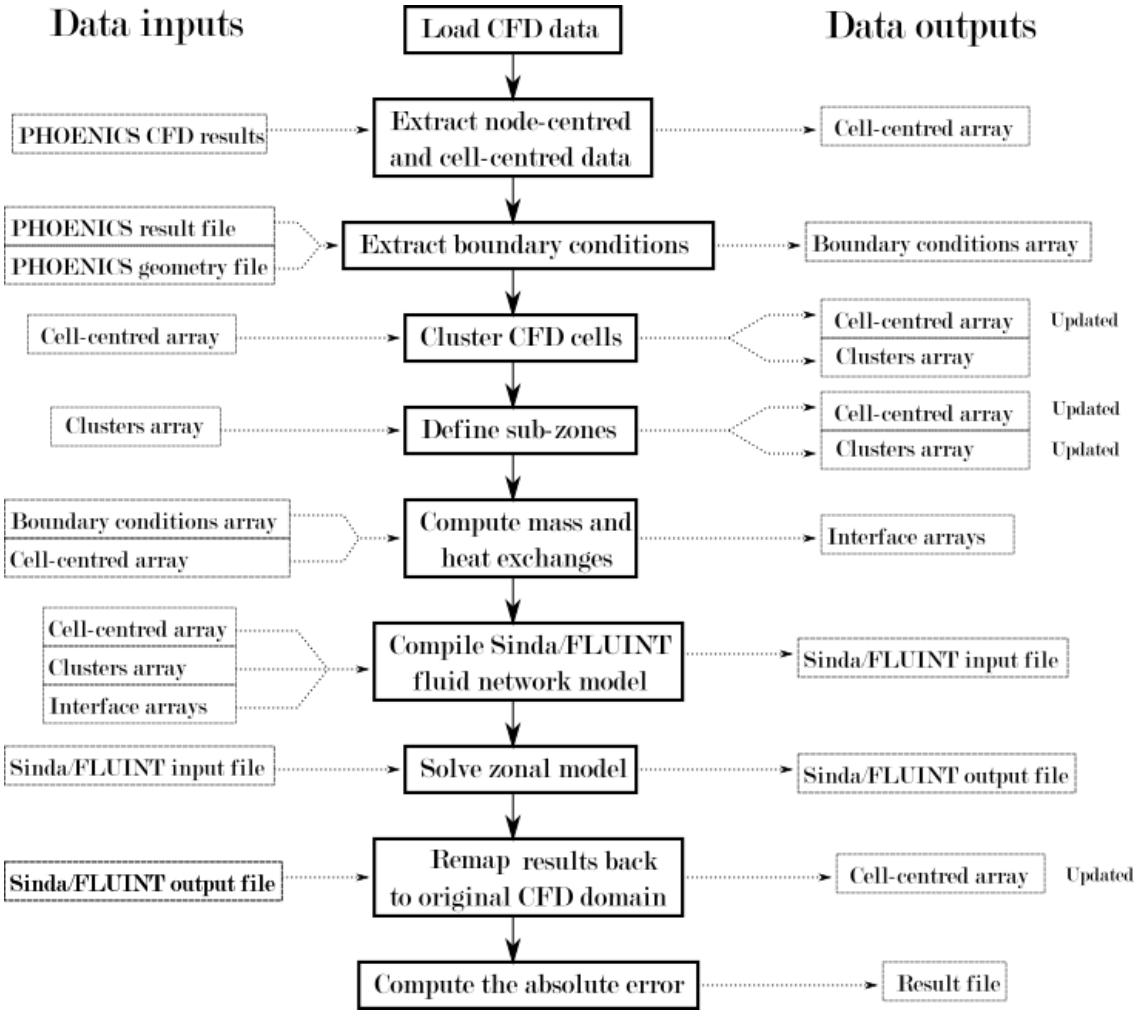


Figure 3-29: Detailed workflow of the proposed method with the corresponding data inputs and outputs at each step

Chapter 4 Case Studies Presentation

4.1. Introduction

This chapter will present the three CFD cases used to benchmark the performance of the method developed in this work. Two case studies represent existing rooms in two Irish research institutes, and one ideal ventilation scenario taken from a CFD benchmark case.

Each case study will be described with the corresponding simulation parameters, CFD solution time and research interest. All cases have been developed in the PHOENICS CFD package and solved on the same machine, a desktop computer with an Intel Core i7-4790 processor and 8GB of RAM.

4.2. Case study 1: office space

The first case study is the one of an office space in the Environmental Research Institute (ERI) of University College Cork, in Ireland. A picture of the office is shown in Figure 4-1. The office measures 5.2m by 5.6m by 2.9m and its furniture is composed of five desks and chairs, and shelf units.

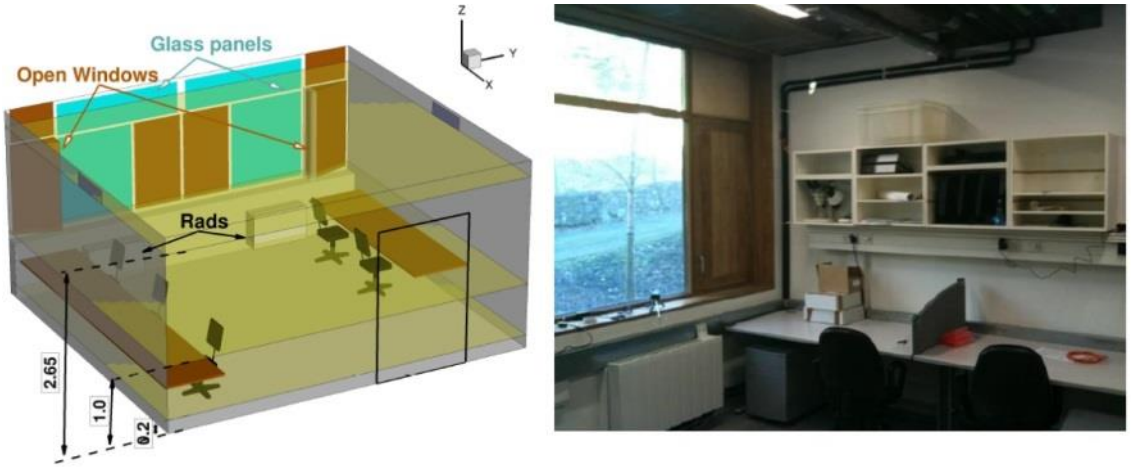


Figure 4-1: Office LG04 in the ERI of UCC, used for the case study 1

The office is heated by two fan convectors on the north wall, which is glazed and features two windows which can be open or closed. Additionally, two air vents on the east and west walls are present, and the south wall has a door that can be open or close. The office has been modelled in PHOENICS by Mullen et al. [50] in steady-state using a regular structured grid with 115x147x93 cells (totalling 1,572,165 cells) shown in Figure 4-2. Turbulence was modelled using the Reynolds Average Navier-Stokes (RANS) approach coupled with the Re-Normalisation Group (RNG) k-ε turbulence model. Air was modelled as an incompressible ideal gas with buoyancy driven by density difference. The ceiling, floor, east and west walls were assumed to have a constant temperature of respectively 23.2 °C, 18 °C, and both east and west walls 20 °C. The windows had a temperature of 9.35 °C and 8.3 °C for the East and West windows respectively. All other objects were considered adiabatic. There are no other heat sources in the office, such as computers or people. The CFD simulations have been validated with experimental data and previously published [50], so further CFD validation will not be presented in this thesis. The CFD domain energy sources are listed in Table 4-1. In the case used for this study, the windows and door are closed, and the two convectors heat up the air to 45 °C at their outlets.

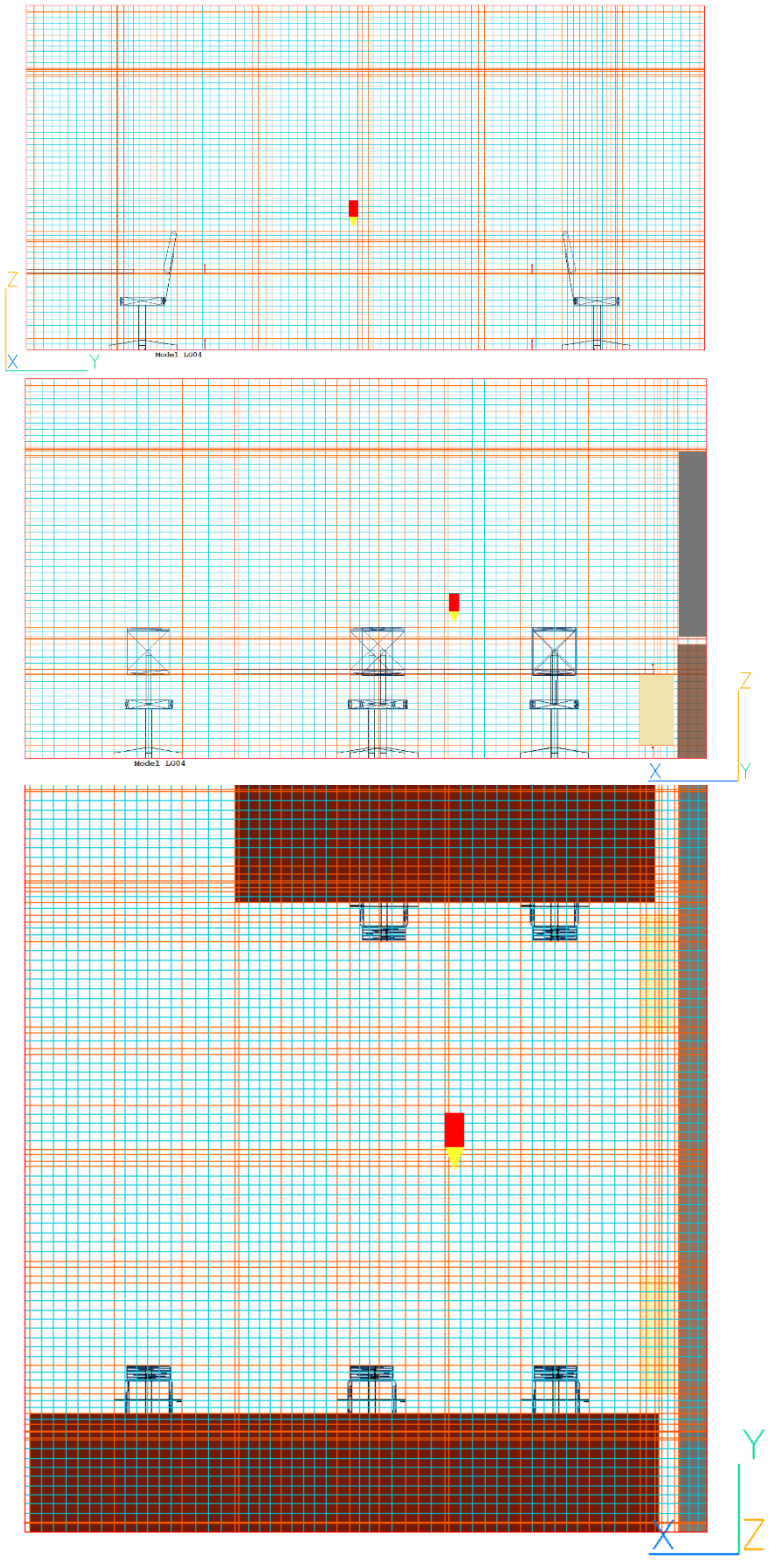


Figure 4-2: +X, +Y and -Z views of the mesh used in Case Study 1. Orange lines represent region boundaries, used in PHOENICS CFD for generating the mesh around domain features.

Table 4-1: Parameters of the boundary conditions of case study 1

Boundary	***Comment***	***Type***	***Base value***
Convectors			
East convector		2D inlet	$T_{Econvector}$ = 45 °C $\dot{m}_{Econvector}$ = 0.048 kg.s^{-1}
West convector		2D inlet	$T_{Wconvector}$ = 45 °C $\dot{m}_{Wconvector}$ = 0.048 kg.s^{-1}
Walls			
Ceiling		2D plate, constant $T_{ceiling}$	$T_{ceiling}$ =23.2 °C
Floor		2D plate, constant T_{floor}	T_{floor} =18 °C
East wall		2D plate, constant T_{Ewall}	T_{Ewall} =20 °C
West wall		2D plate, constant T_{Wwall}	T_{Wwall} =20 °C
Windows			
East window	Closed	Angled-in	$T_{Ewindow}$ = 9.35 °C $\dot{m}_{Ewindow}$ = 0 kg.s^{-1}
West window	Closed	Angled-in	$T_{Wwindow}$ = 8.3 °C $\dot{m}_{Wwindow}$ = 0 kg.s^{-1}
Other openings			
East vent		Opening	T_{Event} = 20.2 °C P_{Event} = 1.013 x10^{5} Pa
West vent		Opening	T_{Wvent} = 20.7 °C P_{Wvent} = 1.013 x10^{5} Pa
Door	Closed	2D plate	$\dot{m}_{door}$ = 0 kg.s^{-1}

The case has been solved for different parameters of temperature and mass flow rates at the convectors inlets. The convector temperature has been varied from 35 °C to 60 °C by increments of 5 °C, and the convector mass flow rates have been varied from 0.0528 kg.s^{-1} to 0.192 kg.s^{-1}. The interactions between temperature and velocity field are ignored, the convectors being in a regime of forced convection. A summary of the simulation parameters is shown in Table 4-2.

Table 4-2: List of parameters used for the variations of case study 1

Variable	***Cases***	***Values***
Temperature ($T_{convector}$)	*T1*	35 °C
	T2	40 °C
	Base	45 °C
	T3	50 °C (West) & 45 °C (East)
	T4	50 °C
	T5	55 °C
	T6	60 °C
Mass flow rates ($\dot{m}_{convectors}$)	*M1*	0.0528 $kg.s^{-1}$
	M2	0.0672 $kg.s^{-1}$
	Base	0.096 $kg.s^{-1}$
	M3	0.1248 $kg.s^{-1}$
	M4	0.1392 $kg.s^{-1}$
	M5	0.1536 $kg.s^{-1}$
	M6	0.192 $kg.s^{-1}$

Each of the cases took approximately 7.5 to 8 hours to solve. Temperature plots of the CFD simulations are shown in Figure 4-3 and Figure 4-4. The slice is a XZ slice taken at Y = 1.17 m, since this slice lies in the centre of the west convector and clearly shows the temperature stratification in the domain as well as the heat plume above the convector.

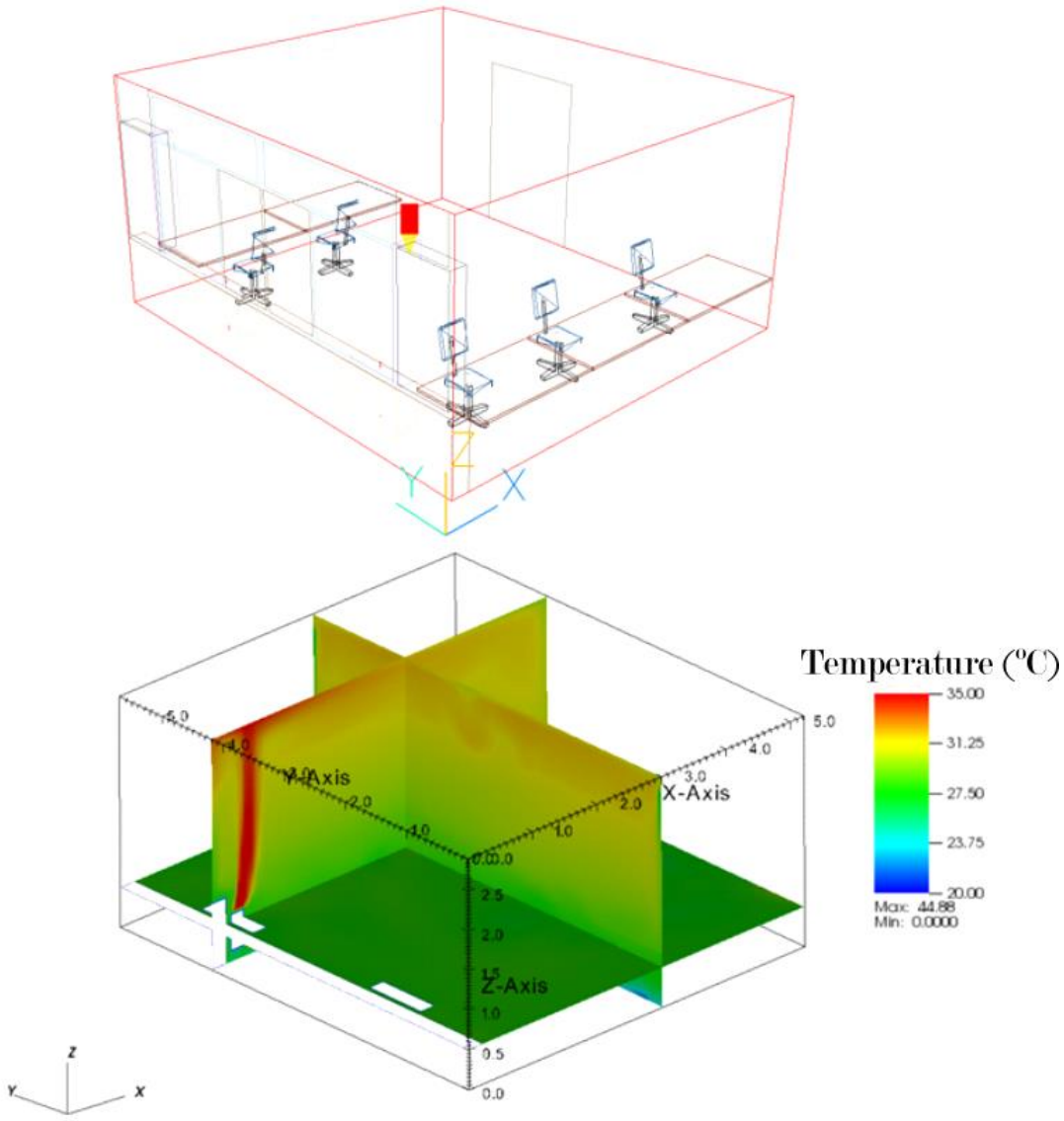


Figure 4-3: Numerical domain (top) and three-slice view (bottom) of the CFD temperature distribution in the domain

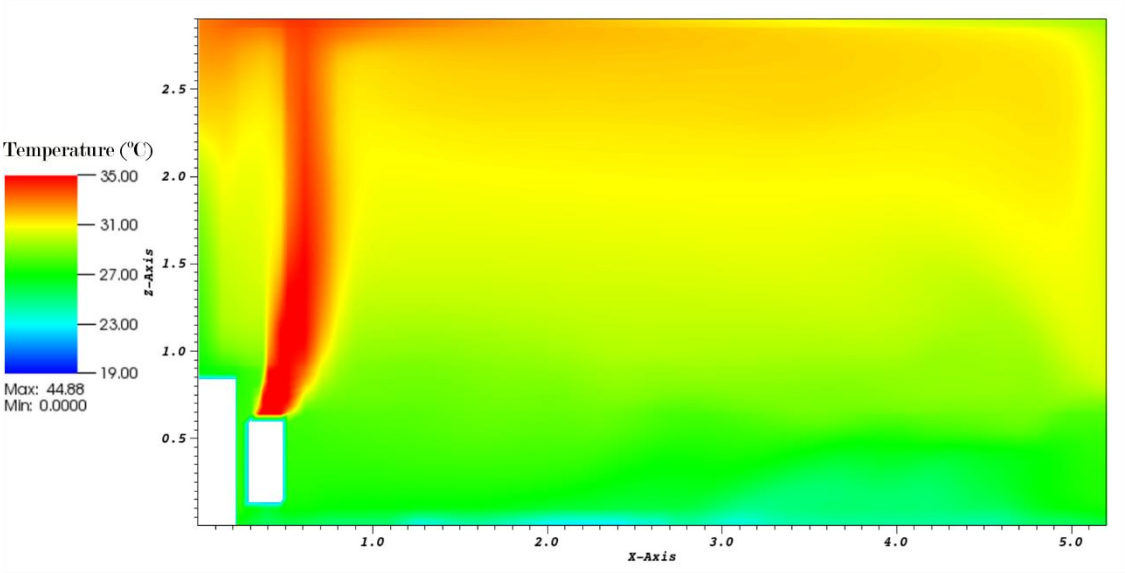


Figure 4-4: Temperature plot, XZ slice of the domain at Y = 1.17m. It shows the heat plume above the convector and the air stratification in the room

This case is interesting for this research because the air in the office is stratified, which is a case that nodal models cannot capture, but can be captured by zonal models. There are no major drafts in this case, which allows a simpler calculation of thermal distributions

4.3. Case study 2: Meeting room

The second test case considered in this study is a meeting room of the Alice Perry Engineering Building of the National University of Ireland, Galway. The meeting room is a highly glazed, naturally ventilated space shown in Figure 4-5. Its dimensions are 4.90 m by 5.89 m by 3.47 m, and it is located in the corner of the building.

Figure 4-5: photograph of the meeting room considered for the second case study

The North and East walls are highly glazed, and on each of the walls one of the bottom windows is opened. Consequently, there is an air inlet on the east wall and an air outlet on the north wall. The furniture of the room comprises of a large desk and 12 chairs. The other objects are a soundproofing panel above the desk, and a concrete column in the North-East corner of the room. Finally, two laptops delivering 130 W of heat each and two people delivering 60 $W.m^{-2}$ of heat each are placed at the northern end of the desk. Simulations were developed in Ansys Fluent by Hajdukiewicz et al. [62] and validated experimentally. For the scope of this study, the CFD study has been replicated in PHOENICS CFD (Figure 4-6) in steady-state. The RNG- k-ε turbulence model was selected, and the air was modelled as an incompressible gas. Buoyancy was computed using density difference.

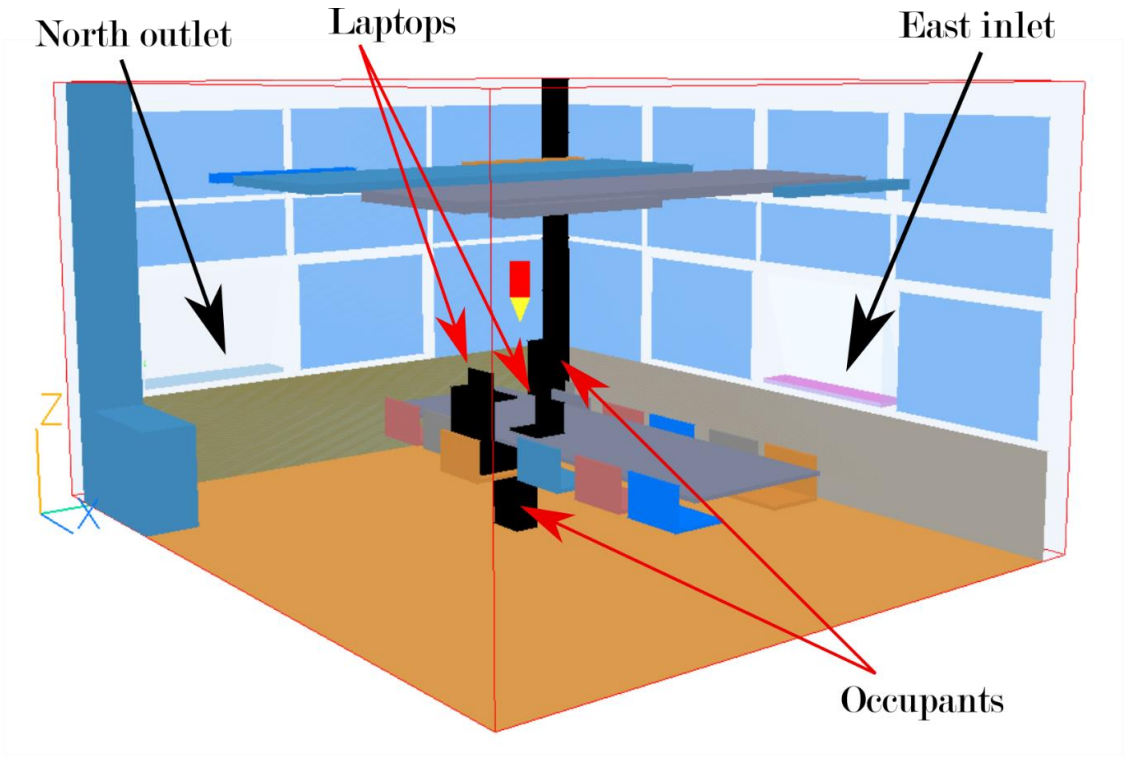


Figure 4-6: CFD model of the meeting room of case study 2.

The mesh was a cartesian structured grid of 120 by 100 by 70 cells (totalling 840,000 cells) shown in Figure 4-7. A grid independence study is presented in Figure 4-8, and has been carried out with three meshes: a coarse mesh of 120 by 100 by 70 or 840,000 cells, a medium mesh of 120 by 100 by 100 or 1,200,000 cells, and a fine mesh of 130 by 120 by 120 or 1,872,000 cells. The locations of pole 1 through 4 are presented in Figure 4-9, taken from Hajdukiewicz et al. [62]. For poles 1 through 3, the temperature variation between grids is less than 0.4 K, while pole 4 exhibits larger variations. This is due to the complex flow at the location of pole 4, which is in the draft between the east and north windows as well as close to two heat sources (one occupant and one laptop). The $y+$ values are between 30 and 60 at the first cell centres in contact with the wall, defined using a log-law wall function. For a high Reynolds number model such as RNG κ-ε, values between 30 and 300 guarantee that the wall function is valid and that the region near the wall is sufficiently resolved [63] when the analysis focuses on the mixing in the domain rather than the forces applied to the wall. The outside temperature was set to 20.82 °C, and the east side inlet passes 0.184 kg.s^{-1} of air at 20.82 °C to the domain. The outlet on the north side is modelled as an outlet with a constant 101,325 Pa pressure and a temperature of 20.82 °C used in case of backflow.

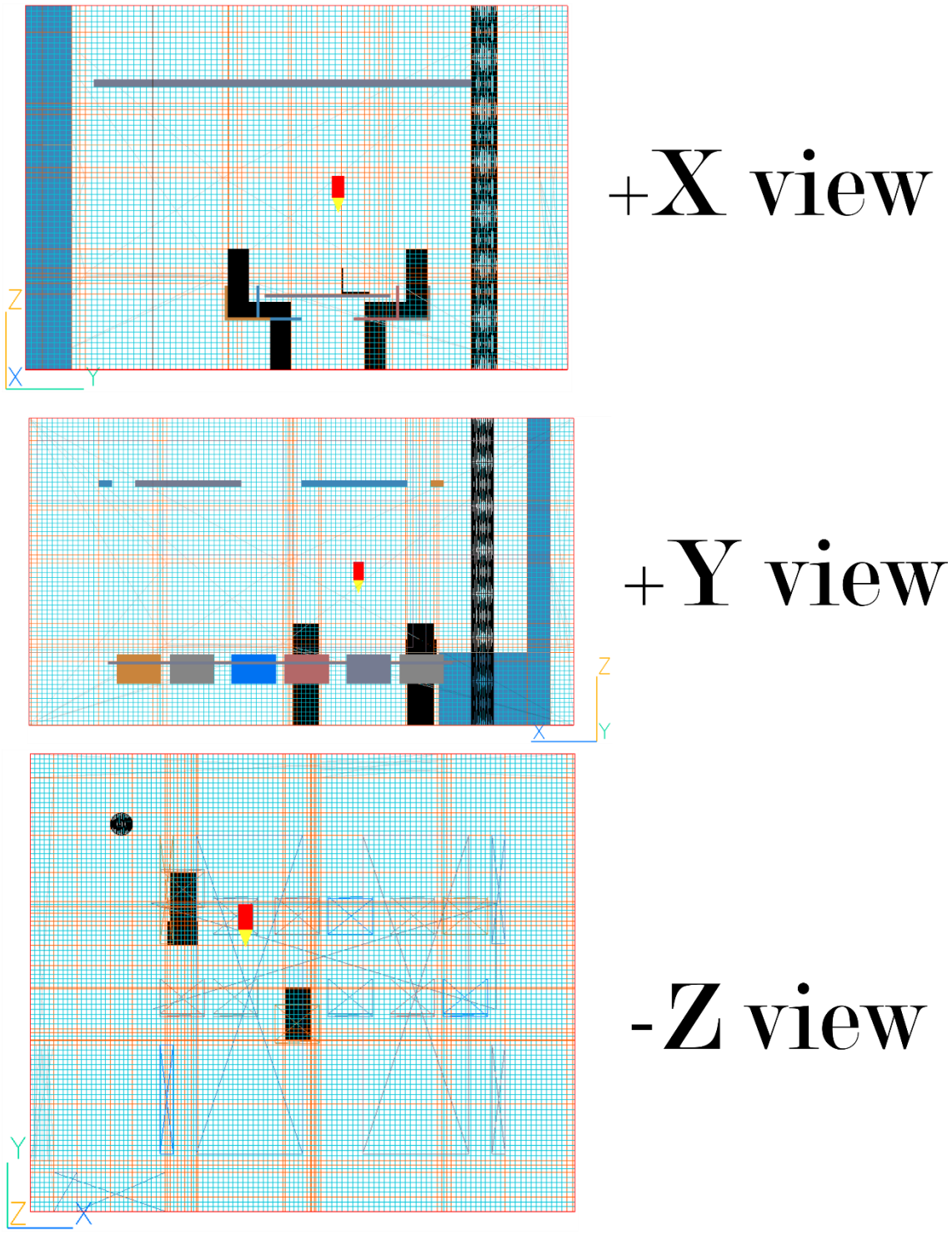


Figure 4-7: +X, +Y and -Z views of the mesh used in Case Study 2

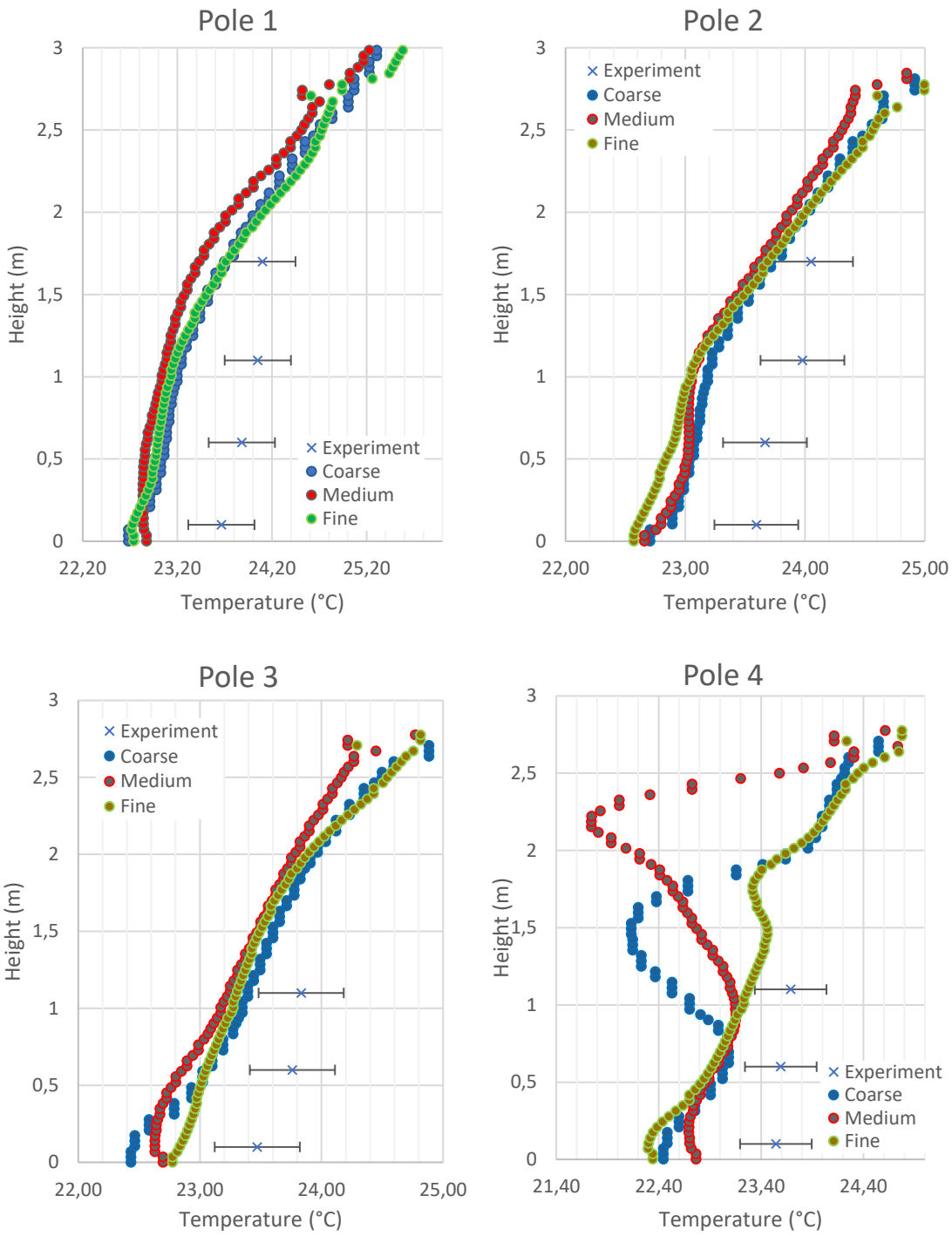


Figure 4-8: Grid independence study with a coarse (840,000 cells), medium (1,200,00 cells) and fine (1,872,000 cells) mesh for case study 2. The error bars show the sensor's uncertainty in measurement. Sensor measurements are taken from Hajdukiewicz et al. [62]

The windows have been modelled as a surface with a constant temperature of 20.82 °C, and the built-in IMMERSOL radiation model of PHOENICS was used to simulate radiative heat transfer. The non-glazed portion of the North and East walls have a fixed temperature of 22.6 °C and the South and West walls have a fixed temperature of 23.64

°C and 24.45 °C respectively. Finally, the floor and ceiling are set to a temperature of 23.5 °C and 23 °C respectively. All the windows have a heat transfer coefficient of 2.3 $W.m^{-2}$. The desk, chairs and soundproofing panel are fully immersed in the fluid domain and considered adiabatic. A summary of the boundary conditions set for this case is available in Table 4-3.

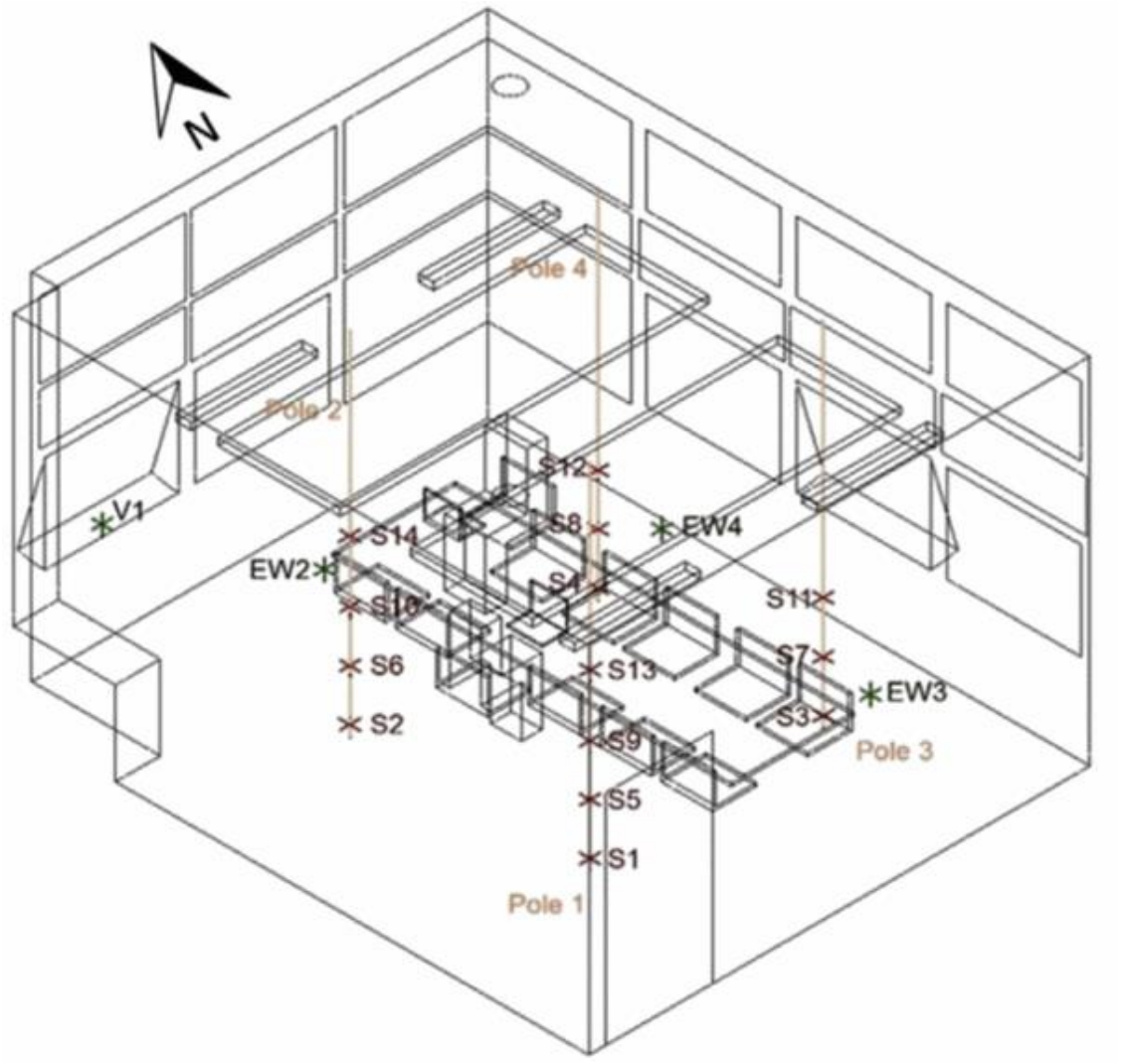


Figure 4-9: View of the computational domain taken from Hajdukiewicz et al. showing the position of the four poles, comprising temperature sensors S1 through S14 used for validation.

Table 4-3: List of boundary conditions of case study 2

Boundary	***Type***	***Value***
Window bottom gap	Angled-in	$\dot{m}_{inlet}$ = 0.184 kg.s^{-1} T_{inlet} = 20.82 °C
Window outlet	Outlet	
Door	2D plate, adiabatic	
Double glazed windows	2D plate, constant heat flux	h_{window} = 2.30 W.m^{-2}.K^{-1} T_{window} = 20.82 °C
South wall	2D plate, constant temperature	T_{Swall} = 23.64 °C
West wall	2D plate, constant temperature	T_{Wwall} = 24.45 °C
North wall	2D plate, constant temperature	T_{Nwall} = 22.6 °C
East wall	2D plate, constant temperature	T_{Ewall} = 22.6 °C
Column	3D object, constant temperature	T_{column} = 22.8 °C
Ceiling	2D plate, constant temperature	$T_{ceiling}$ = 23.0 °C
Floor	2D plate, constant temperature	T_{floor} = 23.5 °C
Occupants	3D objects, constant heat source	$q_{occupants}$ = 60 W.m^{-2}
Laptops	3D objects, constant heat source	$Q_{laptops}$ = 131 W
Table, chairs, soundproofing panel, lamps	3D objects, Adiabatic	

Each case took approximately 12 to 13 hours to solve. The case was solved for different values of mass flow rate and temperature at the inlet as summarized in Table 4-4. The mass flow rate was varied from 0.121 $kg.s^{-1}$ to 0.225 $kg.s^{-1}$, while the temperature was varied between 14.82 °C and 26.82 °C.

Table 4-4: List of case variations for case study 2

Variable	***Cases***	***Values***
Temperature (T_{inlet})	*T1*	14.82 °C
	T2	17.82 °C
	Base	20.82 °C
	T3	23.82 °C
	T4	26.82 °C
Mass flow rates ($\dot{m}_{inlet}$)	*M1*	0.09828 $kg.s^{-1}$
	M2	0. 1310 $kg.s^{-1}$
	Base	0.1638 $kg.s^{-1}$
	M3	0.1965 $kg.s^{-1}$
	M4	0.2293 $kg.s^{-1}$

Figure 4-10 shows a three-slice view of the domain, the XY slice is taken at Z = 1.1 m, the XZ slice is taken at Y =2.9 m and the YZ slice is taken at X = 1.5 m. This case highlights the air draft from the East window as well as the heat plumes from one occupant and one laptop.

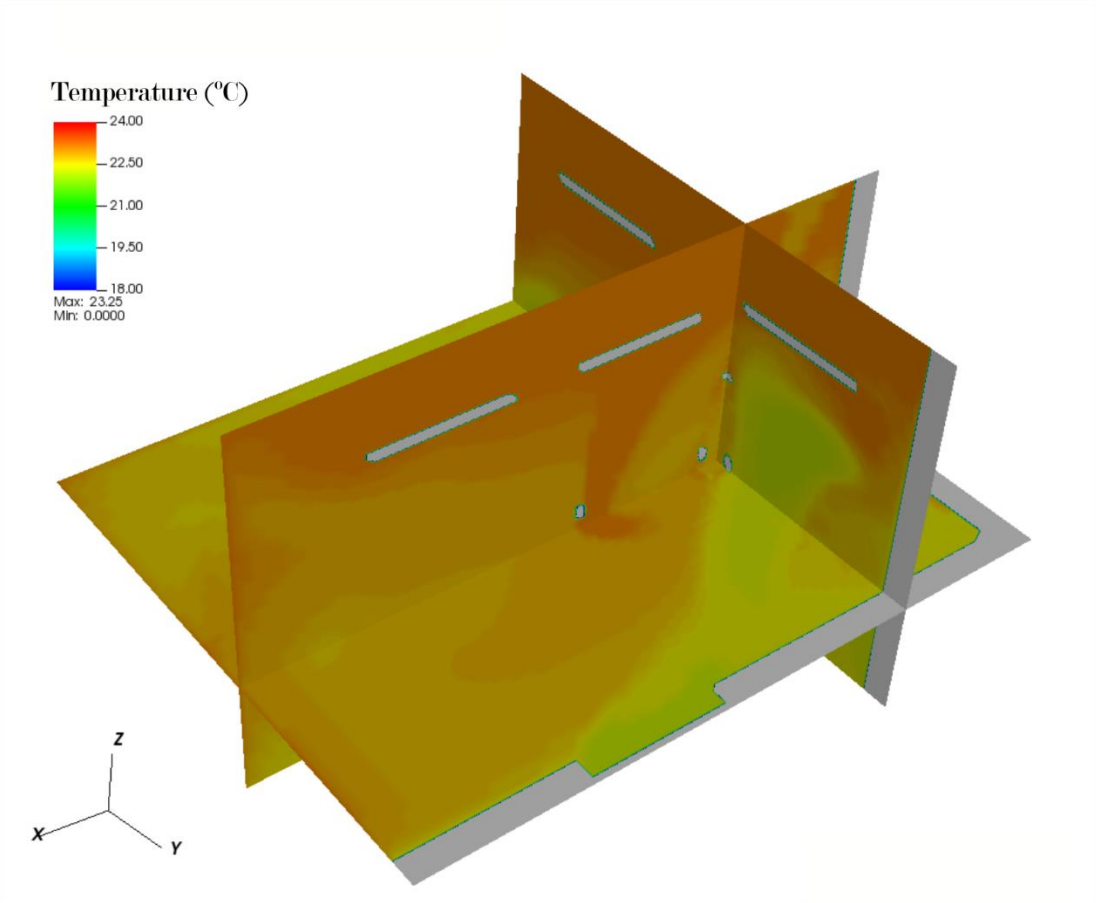


Figure 4-10: Three-slice view of the temperature plot of the meeting room's CFD simulation, highlighting the air draft from the window as well as the heat plumes above the occupants and laptops. The white spots on the heat plumes are due to errors in the output file used for plotting.

This case presents an interest because it features a naturally ventilated room, which is an architectural choice often used in NZEBs, and consequently it features important drafts in the fluid domain that can influence the performance of the proposed method. Since clustering is done on the temperature variable, variable flow fields are expected to challenge the accuracy of the simulated thermal distribution. Furthermore, this case has much lower temperature gradients than the first case study and the air is reasonably mixed, which is also expected to be a challenge for the method.

4.4. Case study 3: Ideal ventilation case

The last case is taken from Cao and Meyers [64] and is a simple ventilation CFD case. It is a cubic room of 4 m by 4 m by 4 m metres in dimension, and it features an inlet at the top of a wall and an outlet at the bottom of the opposite wall. A schematic of the room is shown in Figure 4-11 and the CFD model is shown in Figure 4-12.

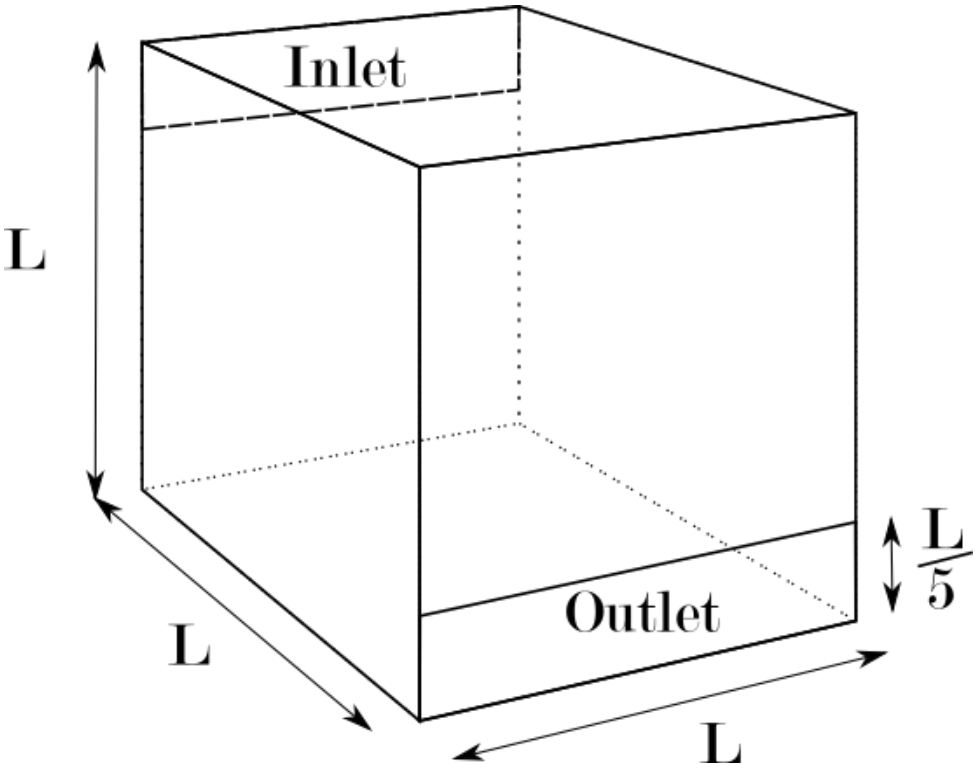


Figure 4-11: representation of the room used for case study 3

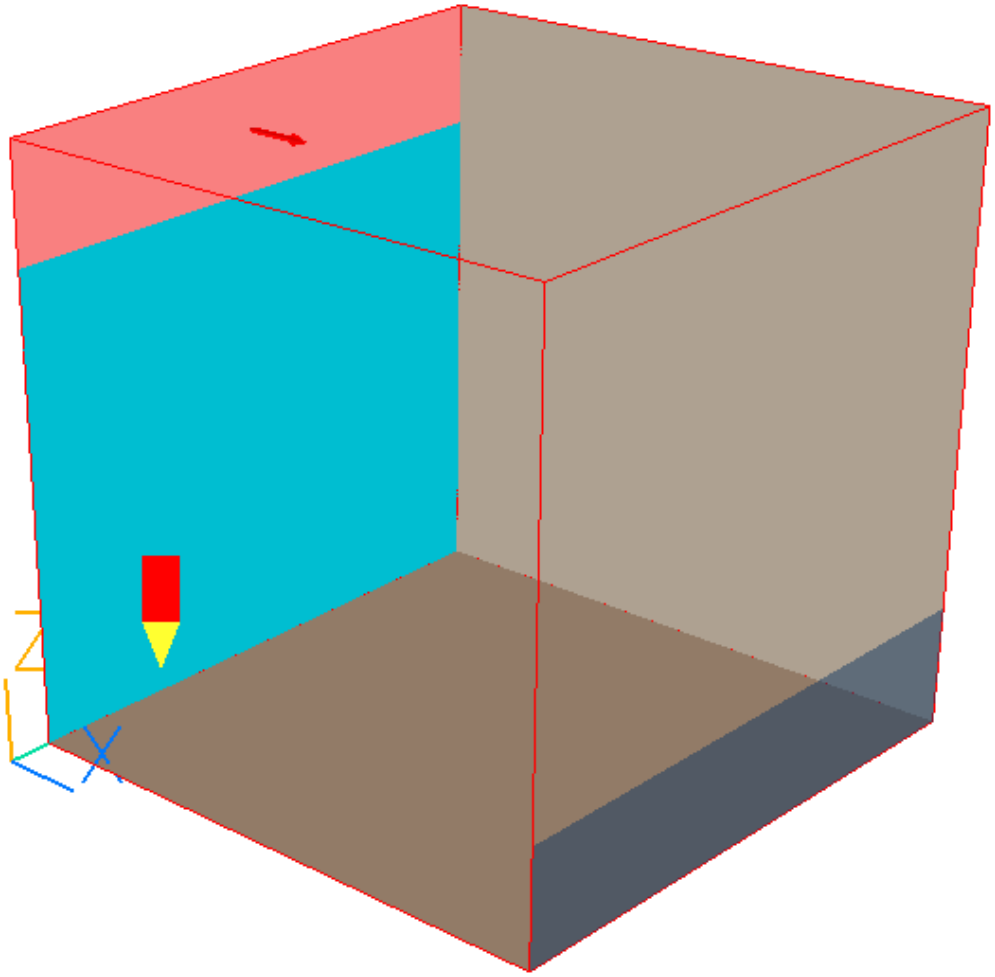


Figure 4-12: CFD model of the ideal ventilation case. It features an inlet and an outlet on two opposing walls

The case has been developed in PHOENICS CFD in steady-state by following the guidelines of Cao and Meyers where if the length of one side of the room is L, the inlet and outlet should be L by L/5 metres in size. Cao and Meyers use the data from Smith and Smith [65] who state that the typical values for mass flow rates at the inlets for a laboratory room of 64 m^3 are usually between 0.12 and 0.25 kg/s therefore 7 simulations have been run for mass flow rates from 0.12 $kg.s^{-1}$ to 0.25 $kg.s^{-1}$ at a temperature of 24 °C, and 5 further simulations with inlet temperatures ranging from 21 °C to 33 °C with a mass flow rate of 0.12 $kg.s^{-1}$. The domain was subdivided in a cartesian structured grid of 27 by 27 by 52 cells (totalling 37,908 cells) shown in Figure 4-13. A grid independence study was realized with three meshes: a coarse mesh with 27 by 27 by 52 or 37,908 cells, a medium mesh with 30 by 30 by 58 or 52,200 cells, and a fine mesh of 35 by 35 by 63

or 77,175 cells for which the results are shown in Figure 4-14 at a vertical pole located at X=2m and Y=2m, or the centre of the domain.

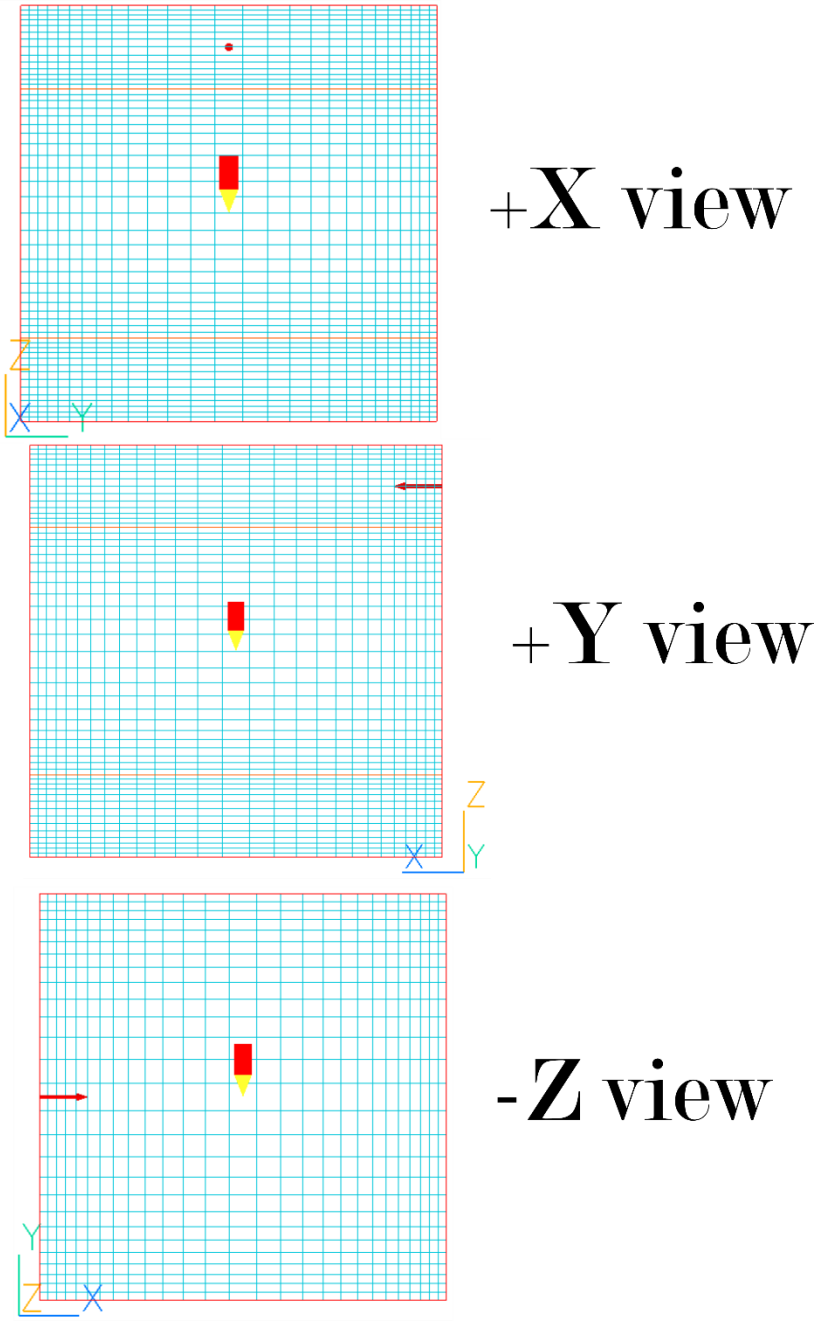


Figure 4-13: +X, +Y and -Z views of the mesh used in Case Study 3

The study shows that there is good agreement between the results obtained with the coarse and fine grid, the temperature variation between h = 3 m and h = 4 m being low with a maximal value of 0.25 K. As in Cao and Meyers [64], the centre of the first cell near the wall is at less than 0.5×10^{-3} L and the $y+$ value is comprised between 0.2 and 9. The air was modelled as an incompressible gas. Low-Reynolds Chen-Kim-k-ε was used as the

turbulence model, and the buoyancy was set to be computed on density differences, as in Cao and Meyers. The outlet has been modelled as a pressure relief outlet at a pressure of 101325 Pa. The walls and ceiling were considered adiabatic, while the floor had a temperature of 18 °C, with a heat transfer coefficient of 5 $W.m^{-2}.K^{-1}$. The reason for this parameter is that otherwise, the temperature in the domain would be perfectly uniform and the method would be inapplicable. The boundary conditions for this case have been summarized in Table 4-5 and the various parameters for the 11 cases have been summarized in Table 4-6. This case, due to its relative simplicity, took 5 to 7 minutes to solve.

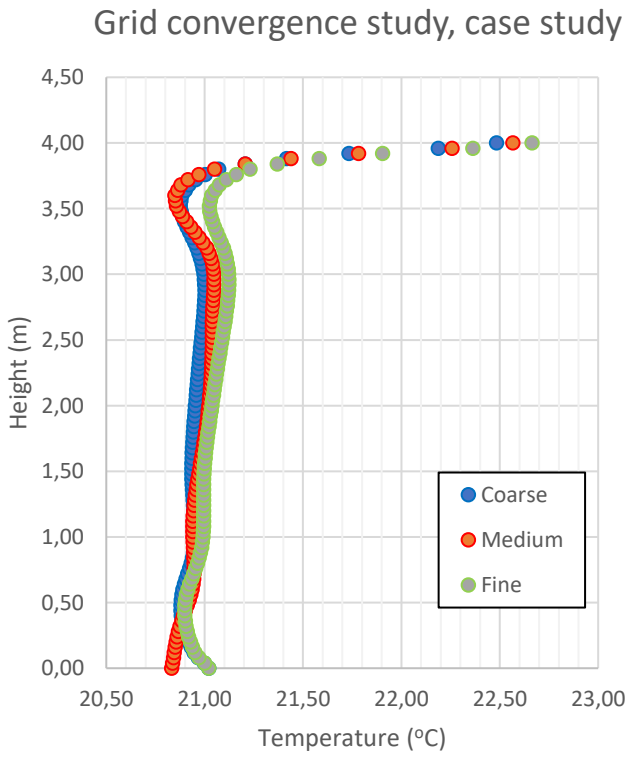


Figure 4-14: Grid independence study for case study 3, carried with three grids: a coarse (37,908 cells), medium (52,200 cells) and fine (77,175 cells) mesh. The temperature is plotted on a vertical pole located at X=2m and Y=2m

Table 4-5: boundary conditions for case study 3

Boundary	***Type***	***Value***
Inlet	2D Inlet	T_{inlet} = 24 °C $\dot{m}_{inlet}$ = 0.12 kg.s^{-1}
Outlet	2D outlet	P_{out} = 101325 Pa
Floor	2D plate	h_{floor} = 5 W.m^{-2}.K^{-}1 T_{floor} = 18 °C
Ceiling	2D plate	adiabatic
Walls	2D plate	adiabatic

Table 4-6: List of case variations for case study 3

Variable	***Cases***	***Values***
Temperature (T_{inlet})	*T1*	21 °C
	Base	24 °C
	T3	27 °C
	T4	30 °C
	T5	33 °C
Mass flow rates ($\dot{m}_{inlet}$)	*M1*	0.12 kg.s^{-1}
	M2	0.14 kg.s^{-1}
	M3	0.16 kg.s^{-1}
	Base	0.18 kg.s^{-1}
	M4	0.20 kg.s^{-1}
	M5	0.22 kg.s^{-1}
	M6	0.25 kg.s^{-1}

Figure 4-15 shows a XZ slice of the domain taken at its centre in the Y direction, since the domain is symmetric along that axis. It clearly shows the air circulating in the room in a clockwise direction from the plot's point of view and exiting the room at the lower outlet, in accordance with the flow patterns described by Cao and Meyers [64].

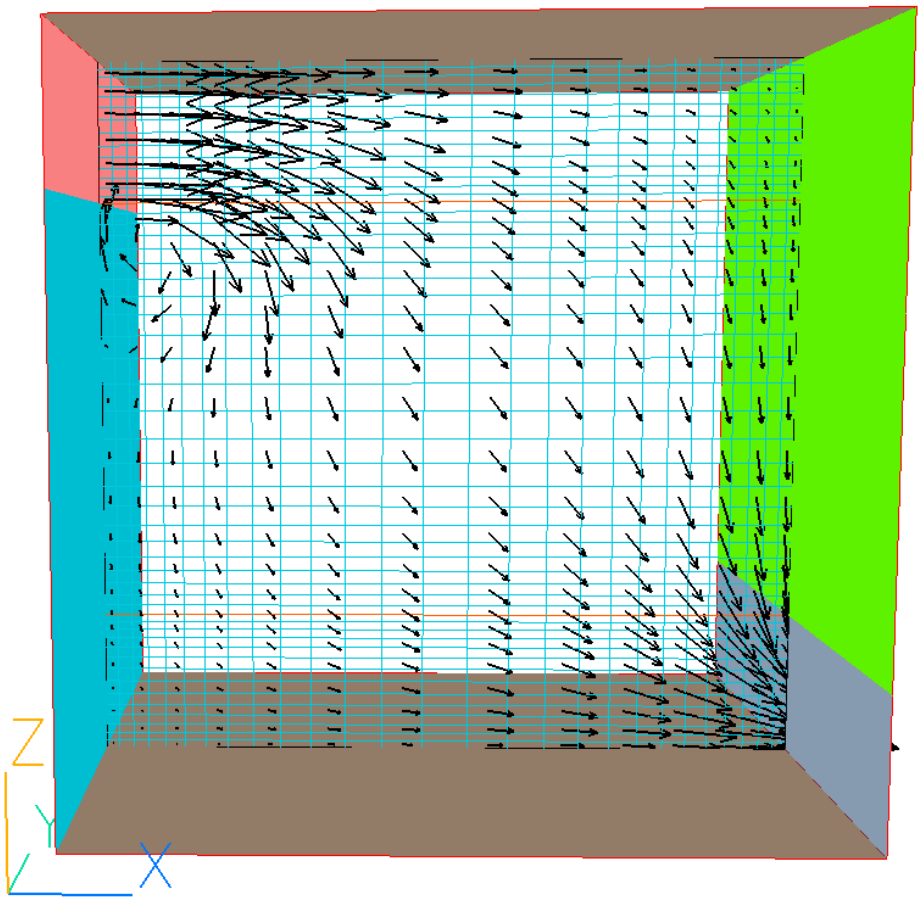


Figure 4-15: XZ slice taken at Y = 2m that shows the inlet at the top left corner and a clockwise flow in the room, with $\dot{m}_{inlet} = 0.12\ kg.s^{-1}$.

The interest of this case is to benchmark the performance of the method in a ventilated environment with a simplified geometry. This allows to pinpoint code-related errors rather than the influence of simulation factors due to a complex simulation such as the one in case study 2 with multiple boundaries and a complex flow shape. Additionally, this case allows a finer study of the performance of the method around air inlets.

4.5. Summary of the Case Studies Presentation

This chapter has presented the three case studies used to benchmark the performance of the method proposed in this doctoral thesis. Two cases were taken from validated studies of real building offices: the first one is a small office space with a notable effect of air stratification, and the second is a larger highly-glazed and naturally-ventilated meeting room featuring two occupants and their laptops as well as a draft in the domain. The last case is a simple cubic room with air mixing and was used for benchmarking the method in a simple ventilation case. These cases represent a range of potential applications for the method and they include features that provide a benchmark for the method in scenarios with air stratification and natural ventilation. The scope is to select a range of cases that allow an assessment of the strengths and shortcomings of the proposed method, and the results of the analysis in these scenarios are presented in the next chapter.

Chapter 5 Results

5.1. Introduction

In order to assess the performance and validity of the proposed method, zonal models have been extracted from the CFD cases presented in Chapter 4 . They have been subsequently solved and the results are presented in this chapter. Firstly, this work has studied the performance of the three clustering algorithms presented in 3.5. The accuracy and time taken to generate the zonal model have been compared, and the results are presented in Section 5.2. Then, this work studied the influence of the number of sub-zones on the error and time to solution of the generated zonal model. Zonal models with 2 to 153 sub-zones have been generated for the three case studies and their level of error compared to CFD is presented in Section 5.3. Finally, this study assessed the performance of zonal models generated using this methodology by measuring the error in temperature distribution predictions when the parameters of temperature and mass flow rate at the domain boundaries were changed. This was done to assess the flexibility of the zonal models and the overall validity of the method. The results are presented in Section 5.4.

5.2. Comparison of clustering methods

The first study concerns the performance of the three clustering methods presented in 3.5. The methods are Mean Values Segmentation (MVS), Coarse Grid (CG) and Classic Watershed (CW). The results of the comparison have been published before [66]. In this study, the accuracy of the clustering methods has been assessed as follows: for each clustering method, clusters were extracted from case study 1, and the accuracy of the resulting temperature distribution was measured using WMAE. Additionally, the time taken to generate each zonal model was measured. This section presents the time to generation of the three algorithms in order to compare their computational performance, a study of the time to solution of the extracted zonal models is shown in the next section.

The three methods were benchmarked against case study 1, in the base configuration where the convectors output air at a temperature of 45 °C and a mass flow rate of 0.048 $kg.s^{-1}$ each which was validated by Mullen et al. [50]. The windows and door are closed, and two air vents are located in the domain. Zonal models were generated for sub-zone numbers ranging between 2 and 35, and the time taken by the Python code from the start to the point where clusters were generated has been timed using the time() function of Python.

The results presented in Figure 5-1 show the WMAE of each method versus the number of sub-zones. These results show that MVS has a WMAE that converges around 0.10 K for 24 sub-zones, while CG's WMAE has not fully converged yet at 27 sub-zones and still presents a level of error higher than MVS. Finally, CW has shown the largest level of error of the three methods, around 1.15 K of WMAE for 26 sub-zones. The grids used for CG are defined in Table 5-1.

Table 5-1: Grid dimensions for the zonal models extracted using CG

	Cells in dimension		
Sub-zones	***I***	***J***	***K***
2	1	1	2
3	1	1	3
8	2	2	2
12	2	2	3
16	2	2	4
27	3	3	3

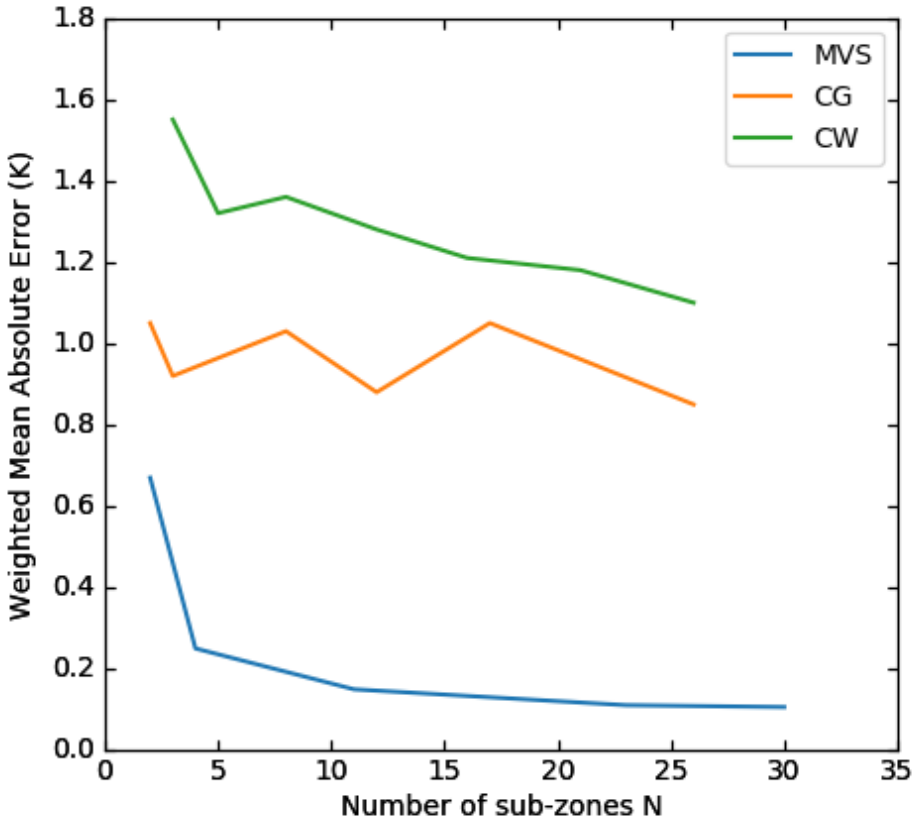


Figure 5-1: Comparison of the MVS, CG and CW methods with regards to WMAE in K versus the number of sub- zones N

Figure 5-2 shows a comparison of the temperature distribution predictions of the solved zonal models for the three methods and the error against CFD. The 2D YZ slice is taken at X=1.17m and shows the heat plume above the convector and temperature stratification in the office. The error plots show clearly that MVS was able to capture both the stratification in the office as well as the large temperature gradients around the convector's heat plume. On the other hand, CW has shown difficulties in capturing both the stratification and the large temperature gradients. The sub-zones are initialized correctly near the temperature extrema such as around the convector, as well as in volumes with small temperature gradients. However, after expanding each sub-zone iteratively the sub-zones fail at capturing temperature distributions as detailed in Figure 5-5. Finally, the CG method did not capture flow properties at all, since the method is based on grid coarsening for performance rather than capturing flow properties, which did not prove satisfactory enough as shown by the WMAE results presented above.

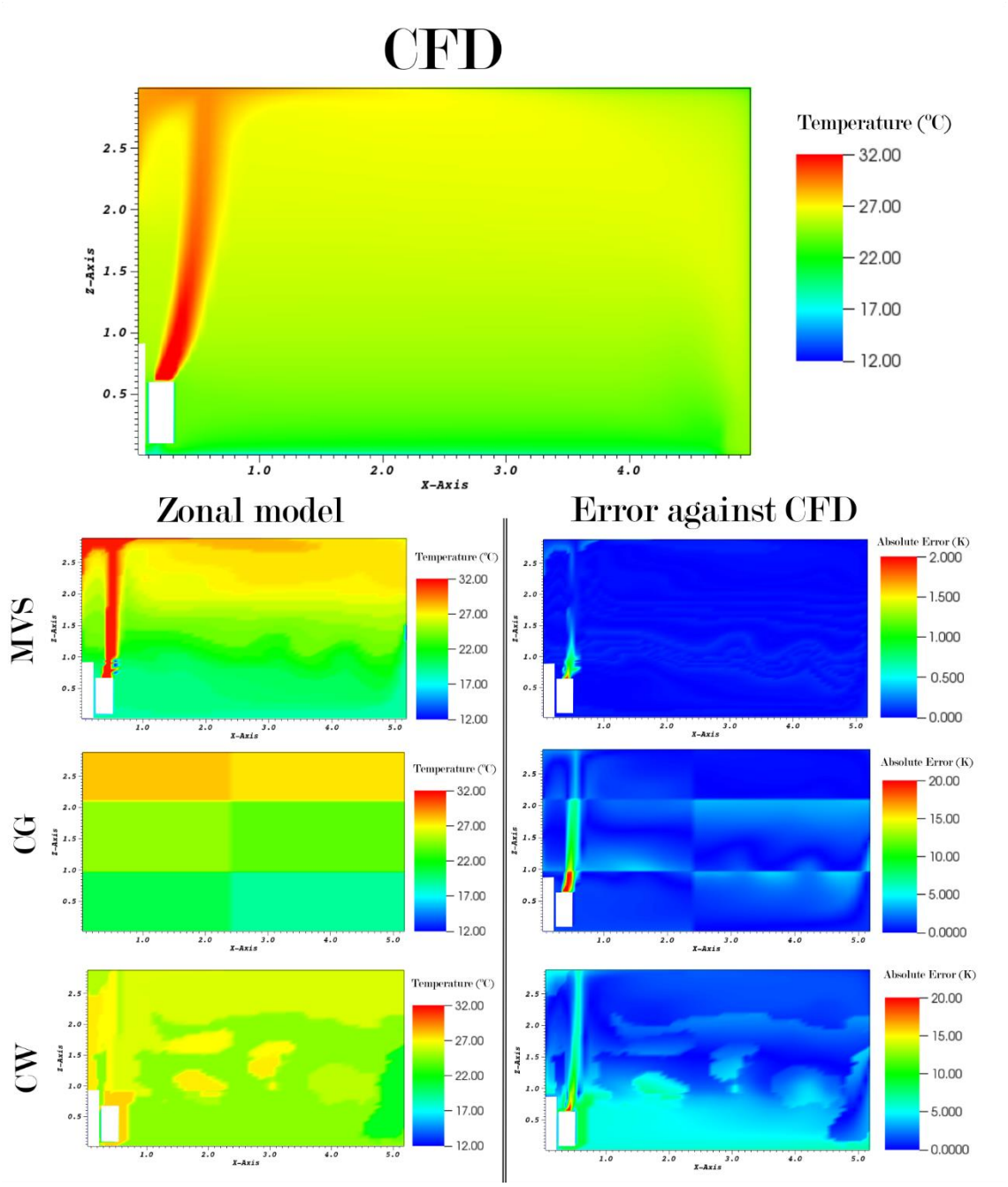


Figure 5-2: Comparison of the temperature distributions predicted by zonal models generated using the MVS, CG and CW clustering methods. The absolute error is measured in K and against CFD.

Figure 5-3 shows a detailed view of the plume above the convector of case study 1, with the peak error of the zonal model's temperature prediction in the first cells above the convector. The error in temperature prediction is limited to 1 K in the rest of the heat plume and becomes negligible 1 meter above the convector. Figure 5-4 also highlights an error below 0.2 K at the interface between sub-zones capturing air stratification. There

are a limited number of cells that are not recorded correctly and as such appear to have a temperature of 0 °C in the output file.

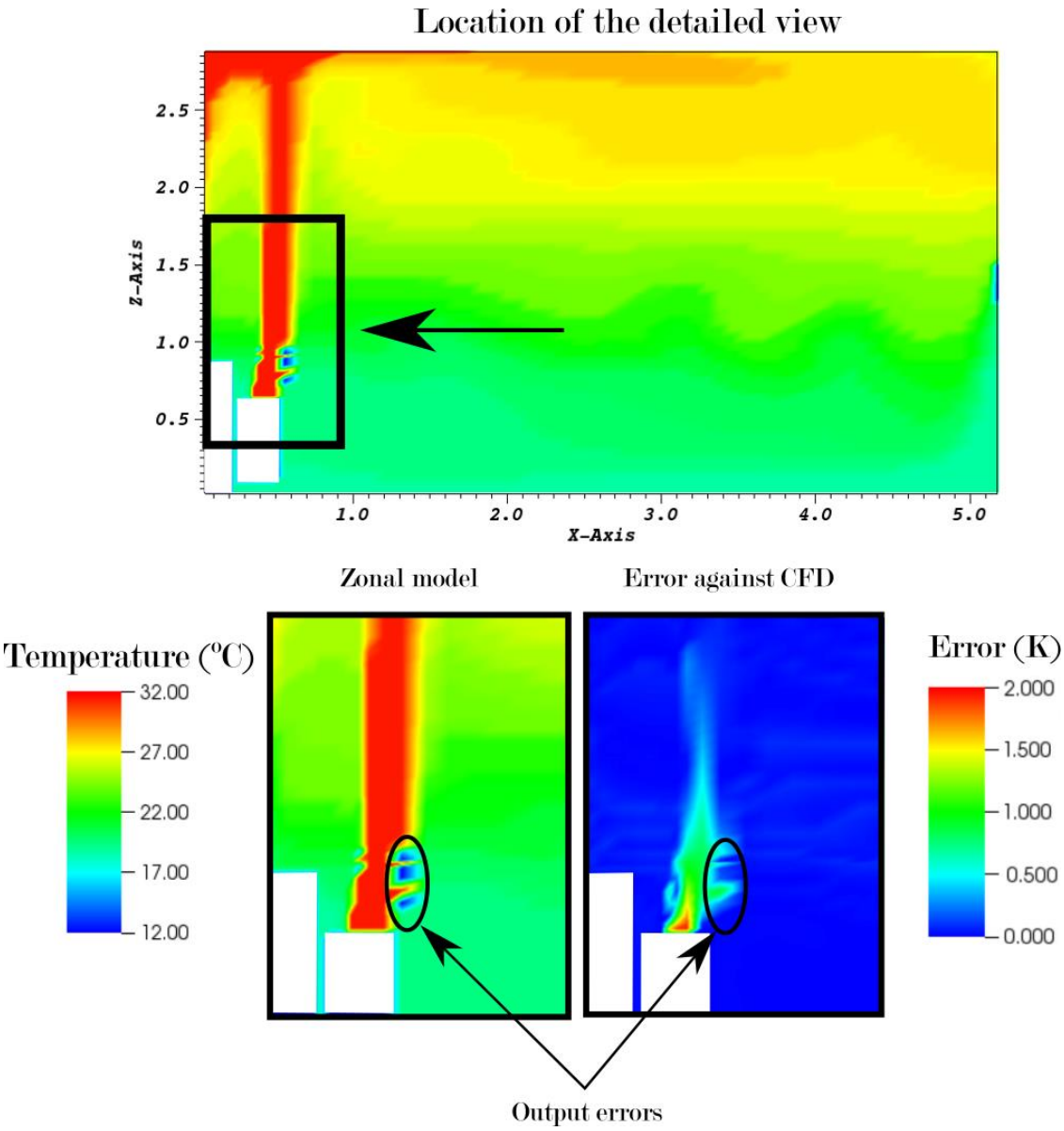


Figure 5-3: detailed view of the heat plume above the convector of case study 1, with the error of the MVS-generated zonal model's temperature prediction against CFD. The peak error in this case is situated in a limited volume directly above the convector.

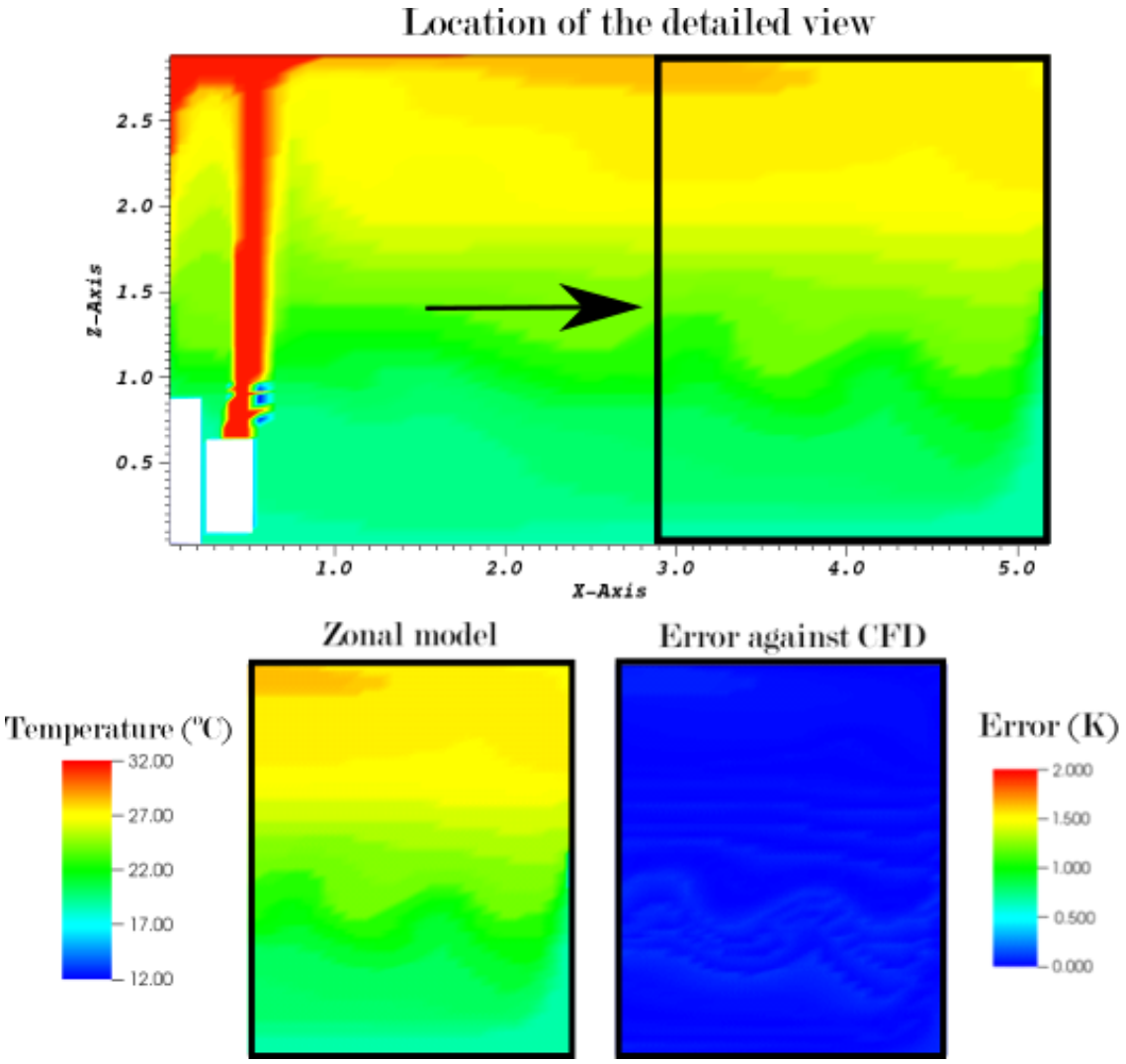


Figure 5-4: detailed view of the volume of stratified air of case study 1, with the error of the MVS-generated zonal model's temperature prediction against CFD

The CW method relies on seeds to initially define sub-zones which are then iteratively expanded to include surrounding cells that fall within a certain temperature range. The method increases the allowable temperature range of each sub-zone until all cells of the domain have been assigned to a sub-zone. As a result, if a quasi-isothermal volume of the domain were not captured by a seed in the initial step, it would be wrongfully included in another sub-zone as seen around the convector in Figure 5-5. The figure shows two examples of sub-zones that expanded excessively: the first one is located directly above the convector and comes from a sub-zone initialized at the base of the heat plume, which expanded to include all the cells surrounding the convector; the second comes from a sub-zone initialized on the heat plume which expanded to include most of the cells between the plume and the wall. This suggests that the method could be improved in two ways:

(1) by allowing the creation of new sub-zones during the sub-zone generation step, and (2) by studying more robust methods for efficiently selecting seeds in a large dataset.

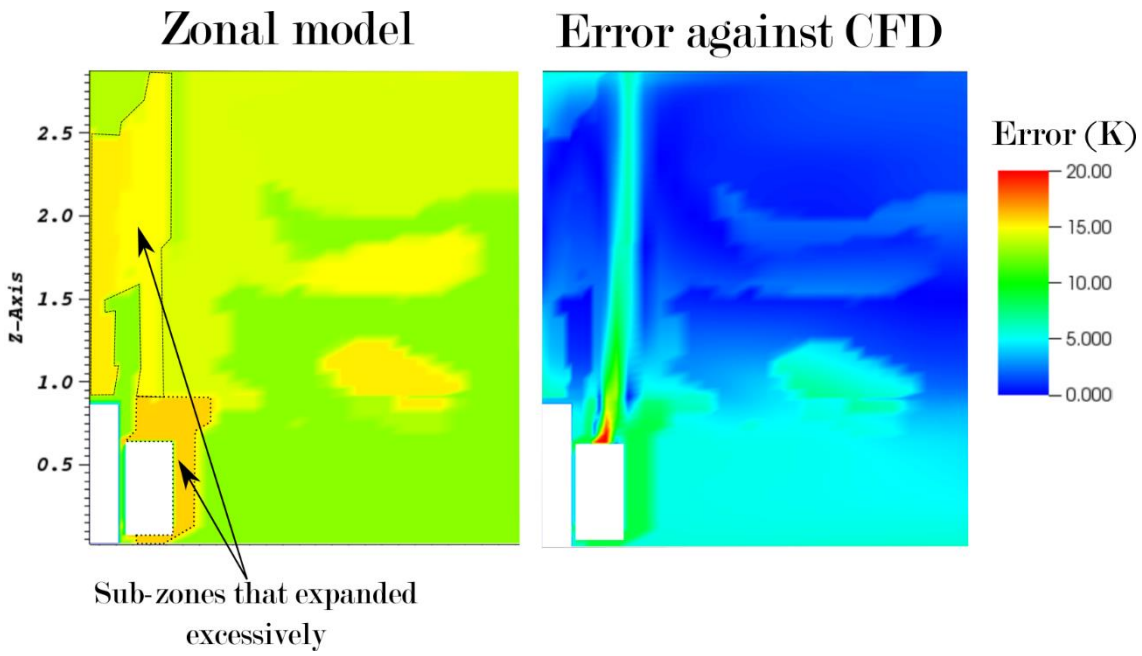


Figure 5-5: detailed view of the heat plume above the convector of case study 1, with the error of the CW-generated zonal model's temperature prediction against CFD. The highlighted sub-zones are an example of sub-zones that expanded excessively, causing large errors in the resulting temperature distributions.

The second step of this assessment considers the time taken from CFD data extraction to sub-zones generation, in order to compare the three methods on their computational cost. The results presented in Figure 5-6 show that of the three methods, CG (left axis) is the most rapid as it was expected due to its computational simplicity, generating sub-zones in around 17 seconds quite constantly regardless of the number of sub-zones. MVS (left axis) is the second fastest, generating sub-zones in approximately 26 to 29 seconds for respectively 2 to 29 sub-zones, while CW (right axis) is the least rapid method with sub-zones generated in approximately 600 to 800 seconds. The CW method has shown to be computationally inefficient, because of a time-consuming search for seeds and for the number of iterations needed to populate the clusters, placing its time to generation an order of magnitude higher than the CG and MVS methods. The difficulty resides in correctly locating local maxima and minima in a 4-dimensional dataset, which is a time-consuming task.

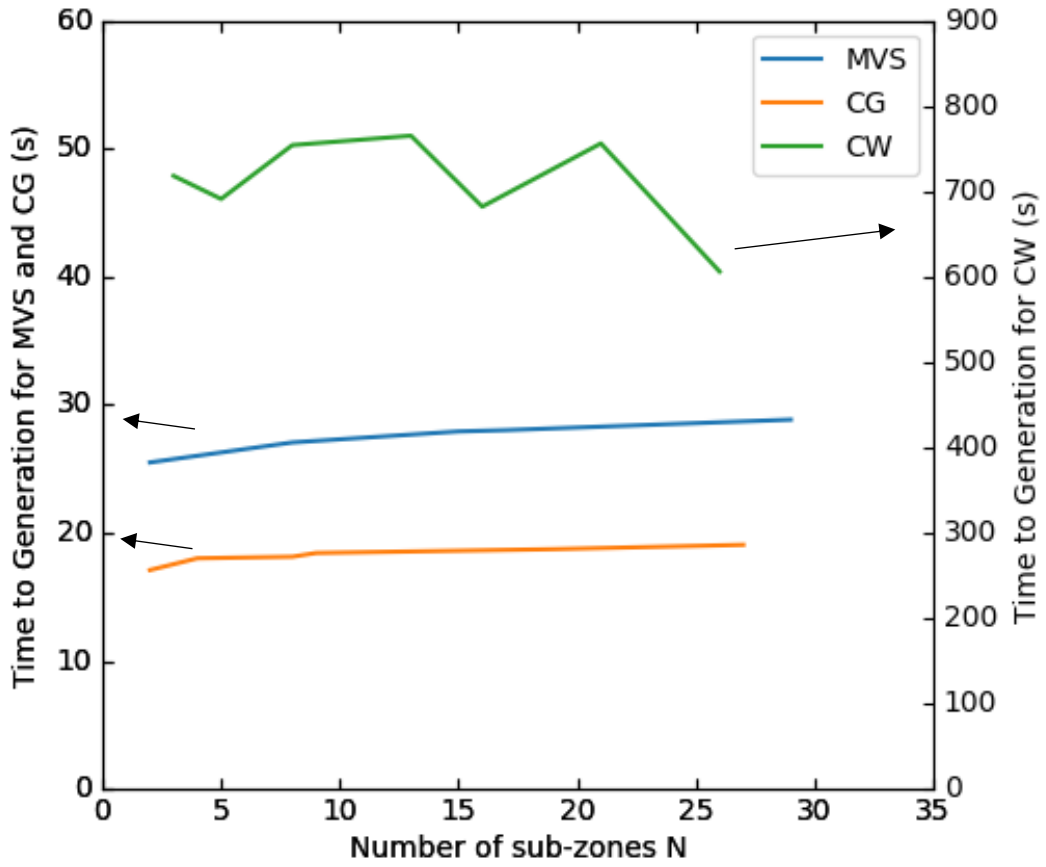


Figure 5-6: Time taken to generate a zonal model with MVS and CG (left axis) and CW (right axis)

This first study has motivated the author in selecting the MVS method for this research project, since it generates accurate zonal models in less than 30 seconds and performs better than CG and CW with regards to finely capturing features of the fluid domain such as stratification and temperature gradients.

Figure 5-7 shows the temperature prediction of the zonal model in 3 dimensions with a narrow range for the temperature scale in order to highlight the shape of each sub-zone. This figure shows how the MVS method generates sub-zones that capture the actual temperature distribution in the domain, rather than pre-defined sub-zones as in the CG method or other classic zonal methods.

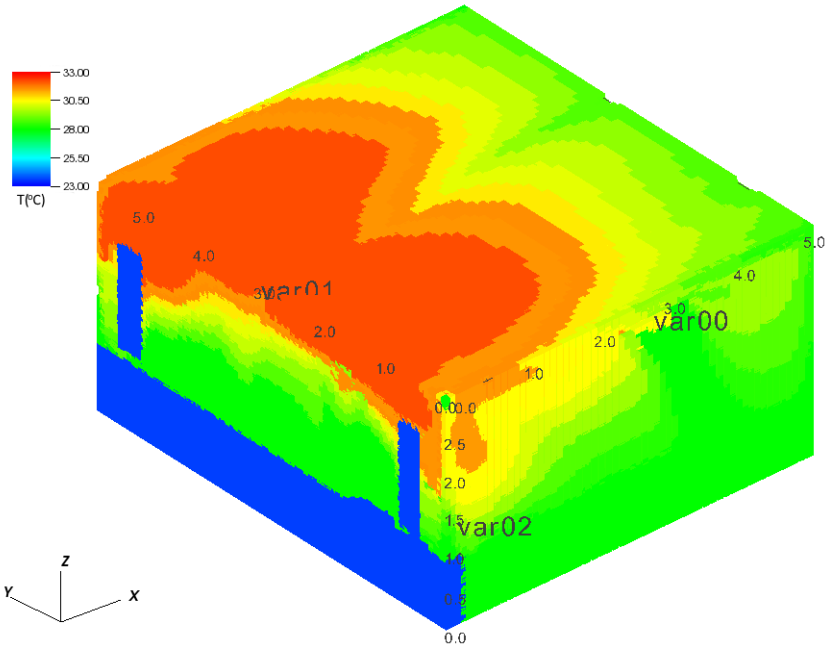


Figure 5-7: 3-dimensional view of the temperature predictions of a 17 sub-zones model extracted from case study 1 highlighting the shape of sub-zones capturing the temperature distribution in the domain

For the rest of the results chapter, results will be presented using the MVS method.

5.3. Influence of the number of sub-zones

After selecting MVS as the best of the three methods for clustering computational cells, the next step in the study was to determine the importance of sub-zone numbers on the final results with regards to error and time to solution. To do so, the proposed method was applied to the three case studies presented in Chapter 4 . For each case study, zonal models comprising 2 to 153 sub-zones were generated and solved without changing boundary conditions parameters, and the resulting temperature distributions were compared to the ones of the original CFD simulation. Figure 5-8 shows the influence of sub-zone numbers on the WMAE of the solved zonal models. It shows that for all cases, the error quickly decreases and reaches a value of around 0.08 K for case study 1, 0.1 K for case study 2, and 0.18 K for case study 3. After a certain number of sub-zones, 24 in case study 1, 52 in case study 2 and 67 in case study 3, the impact of the number of sub-zones on the error becomes minimal. Past this point, the increase in time to solution becomes significant, almost doubling between 60 and 153 sub-zones for a moderate decrease in error. This

suggests that the current method has an inherent lower limit to the error that can be practically achieved, since past a certain point the decrease in error becomes minimal and the increase in computational time becomes significant.

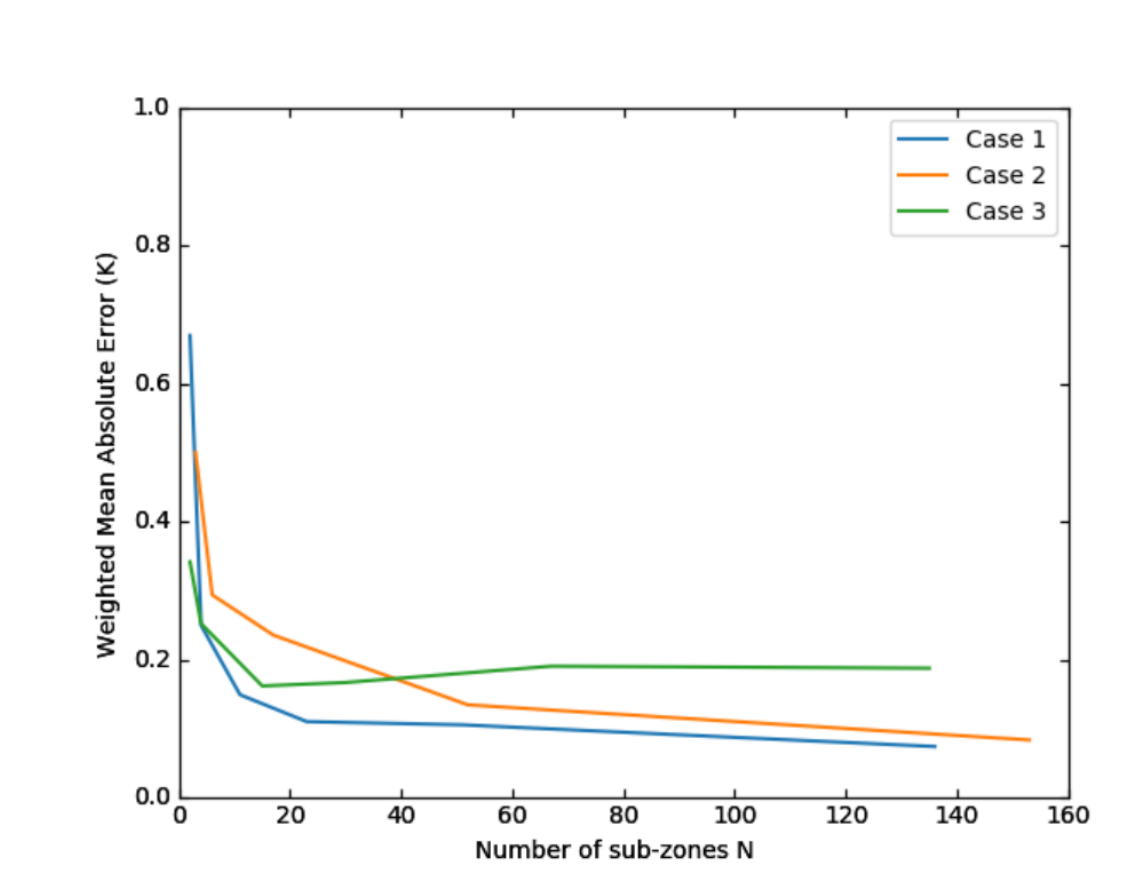


Figure 5-8: Mean Absolute Error in K of zonal model temperature prediction against CFD for the three case studies, for sub-zone numbers ranging between 2 and 153

Figure 5-9 shows the time to solution versus the number of sub-zones for each case. Results show that the time to solution increases almost linearly for all cases.

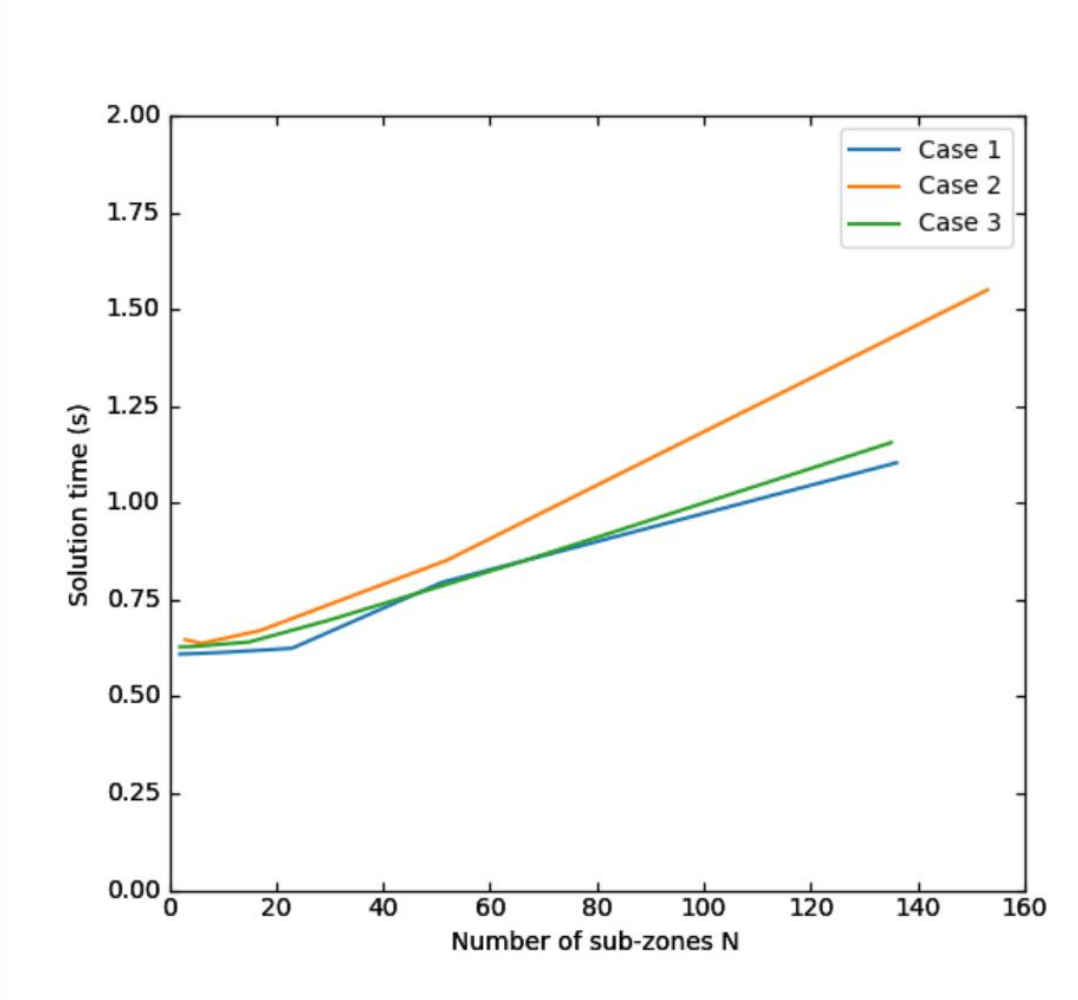


Figure 5-9: time to solution of the zonal model in seconds for the three case studies, for sub-zone numbers ranging from 2 to 153

This study has evaluated the time taken to solve zonal models with 2 to 153 sub-zones. For this range of sub-zones, the time to solution evolves almost linearly between 0.6s and 1.6s for all case studies. For case study 1 it went from 0.61 seconds for 2 sub-zones to 1.10 seconds for 136 sub-zones, which represents almost double the time for 68 times as many sub-zones. For case study 2 it went from 0.64 seconds for 3 sub-zones to 1.55 seconds for 153 sub-zones. Finally, for case study 3 it went from 0.62 seconds for 2 sub-zones to 1.15 seconds for 135 sub-zones. There is an increasing gap in the time to solution between the three methods. This is due to the fact that the three cases do not generate the same amount of sub-zone interactions: in case study 1 there is a total of 1142 mass exchanges between sub-zones (64 sub-zone model), while in case study 2 there are 1476 (52 sub-zone model) and 1249 in case study 3 (67 sub-zone model). This is due to the

geometrical shape of each sub-zone which will induce in some cases more sub-zone interfaces than another case with an identical sub-zone number.

Generating more sub-zones helps in capturing the plume as shown in Figure 5-10 which highlights the air inlet of case study 2 and compares the local error of zonal models with 17, 52 and 153 sub-zones. This is however not efficient in case study 3, where the inlet temperature is closer to that of the domain as highlighted in Figure 5-11 and therefore additional sub-zones do not necessarily capture inlet flow features more finely. In this case, the precision of the temperature values in the CFD output was the limiting factor as for small ΔT the algorithm would either wrongfully include cells in a sub-zone if the ΔT was too high, or create too many sub-zones if the ΔT was too low.

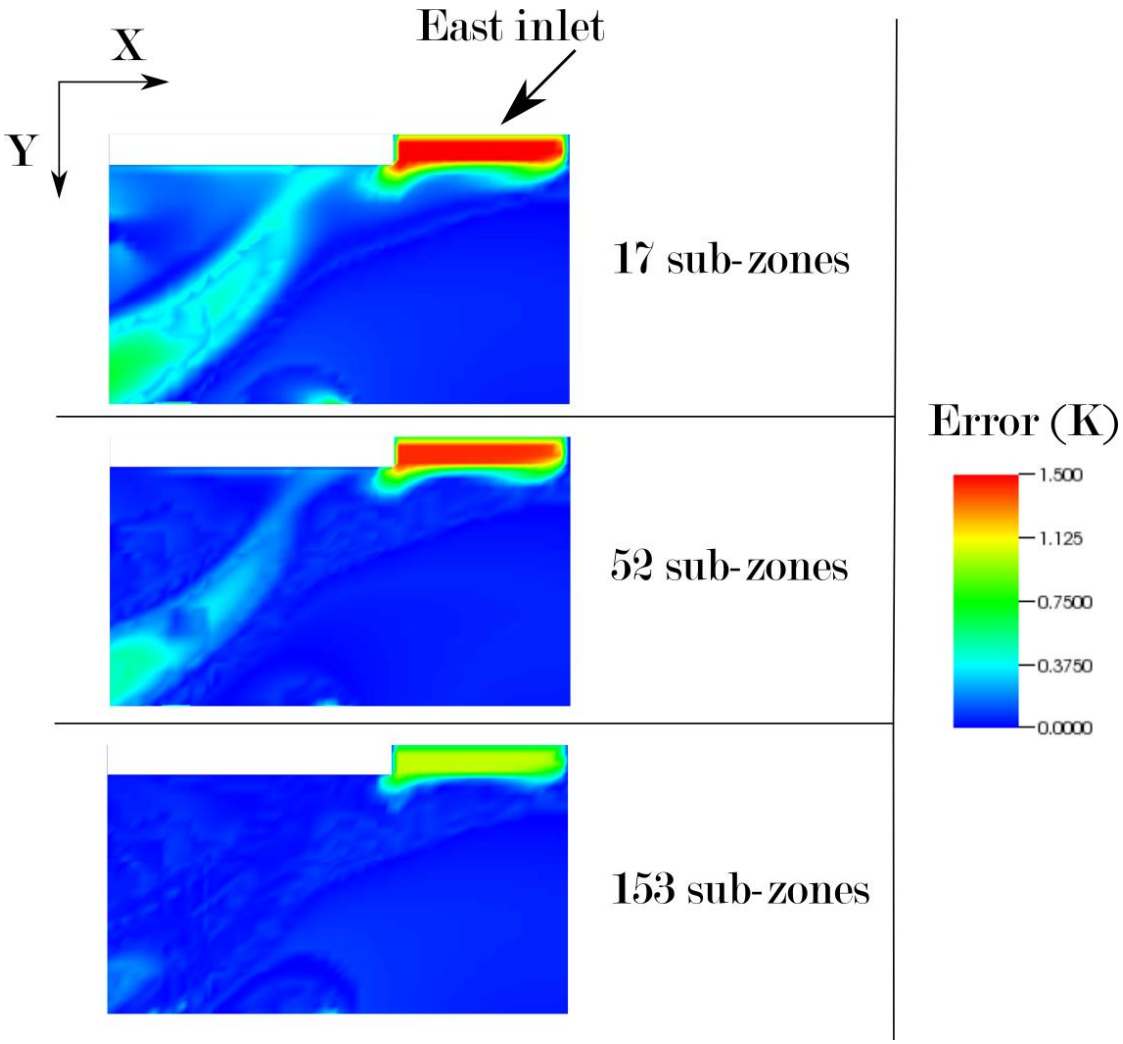


Figure 5-10: view of the error against CFD at the inlet of case study 2, taken at a XY slice at Z = 1.1 m, for a 17 sub-zones, 52 sub-zones and 153 sub-zones model

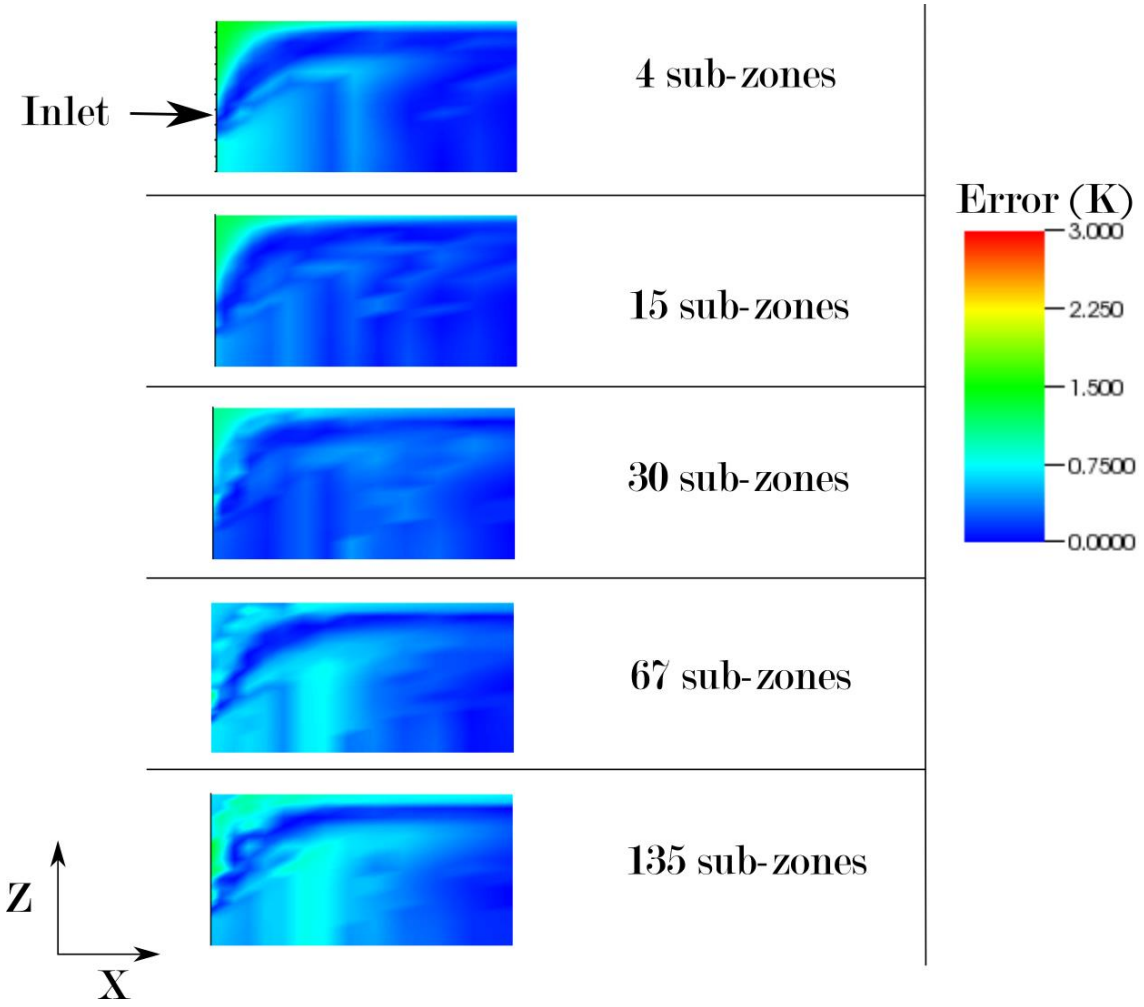


Figure 5-11: view of the error against CFD at the inlet of case study 3, taken at a XZ slice at Y = 2 m, for a 4, 15, 30, 67 and 135 sub-zones model

5.4. Fidelity of the zonal models

For each case study, a set of cases were simulated with PHOENICS CFD for different boundary condition parameters as detailed in Chapter 4 . In this study, fidelity is evaluated by the level of error against CFD for a zonal model solved for off-design boundary conditions in order to assess how finely the physical properties of the domain have been captured by the zonal model. First, in order to define a reference error a zonal model is extracted from each CFD case and subsequently solved without changing the boundary conditions, shown in blue in the following plots and named *'reference error'*. This is not the intended use of the proposed methodology, but it is useful for evaluating the error introduced when boundary conditions are changed versus the error when they are not.

Then, for each case study one scenario is defined as the *'base'* case, and a zonal model is extracted for each base case. This zonal model is then solved for different boundary condition parameters, corresponding to those of the CFD cases, and the error in temperature prediction of the zonal model against the corresponding CFD simulation can then be quantified, shown in red in the following plots and named *'zonal model'*. Figure 5-12 shows the method used to extract the *'reference model*' and the '*zonal model*'.

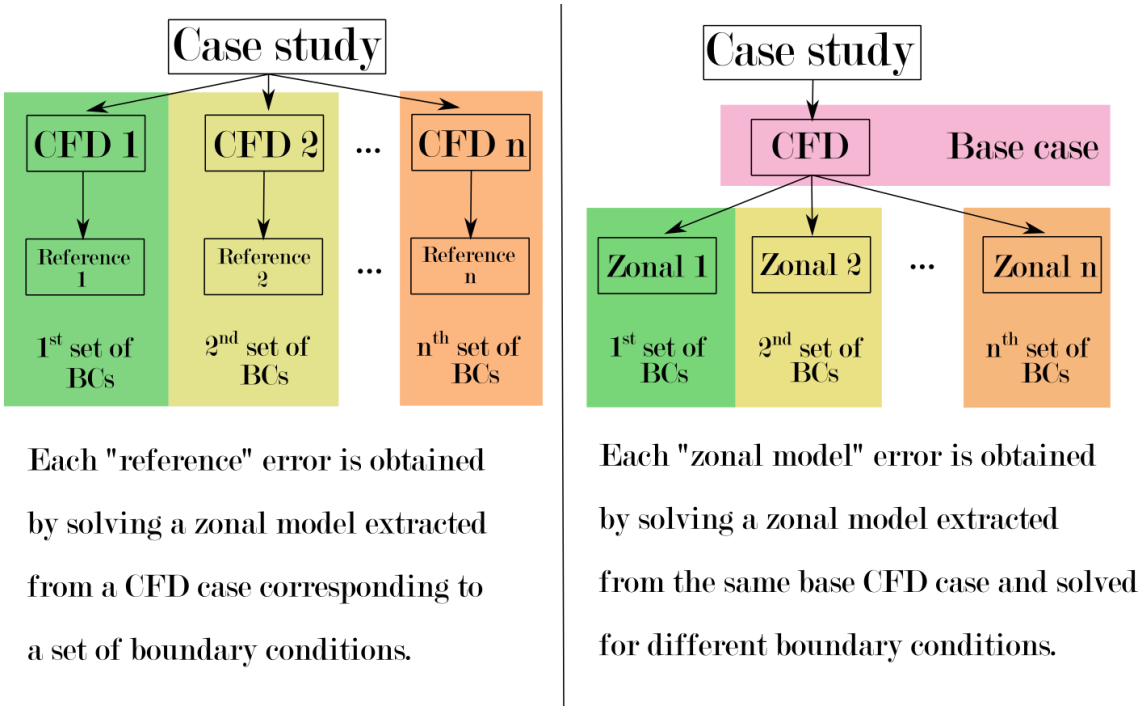


Figure 5-12: method used to extract zonal models that can be used to quantify a reference error, and the intended usage of the method for extracting zonal models from a single CFD case

Figure 5-13 shows the WMAE for the office space of case study 1, where a 24 sub-zones zonal model has been extracted from the base case with $\dot{m}_{convector}$ = 0.048 kg.s^{-1}, $T_{convector}$ = 45 °C and solved for values of $T_{convector}$ between 35 °C (T1) and 60 °C (T6). For all the cases, the error falls under 0.6 K for this range of temperatures. Figure 5-14 shows the same study when the temperatures remain the same, but the mass flow rate at the inlets is changed. The zonal model was extracted from the base case with $\dot{m}_{convector}$ = 0.048 kg.s^{-1} and $T_{convector}$ = 45 °C and solved for values of $\dot{m}_{convector}$ between 0.0528 kg.s^{-1} and 0.192 kg.s^{-1} as summarized in Table 4-2. When the change in mass flow rate is moderate, the error remains under 0.6 K but when the mass flow rates are doubled it reaches between 0.78 and 0.91 K of absolute error. Finally, Figure 5-15 shows an enlarged view of three plots as a XZ slice taken at Y = 1.17 m. It shows the temperature predictions of CFD and

the zonal model as well as the error of the base zonal model versus CFD. This result shows that the zonal model was able to capture both regions with high temperature gradients, such as the heat plume above the convector, as well as stratified air in the rest of the domain. The peak error is localized on the heat plume above the convector, which is around 1 K on the plume.

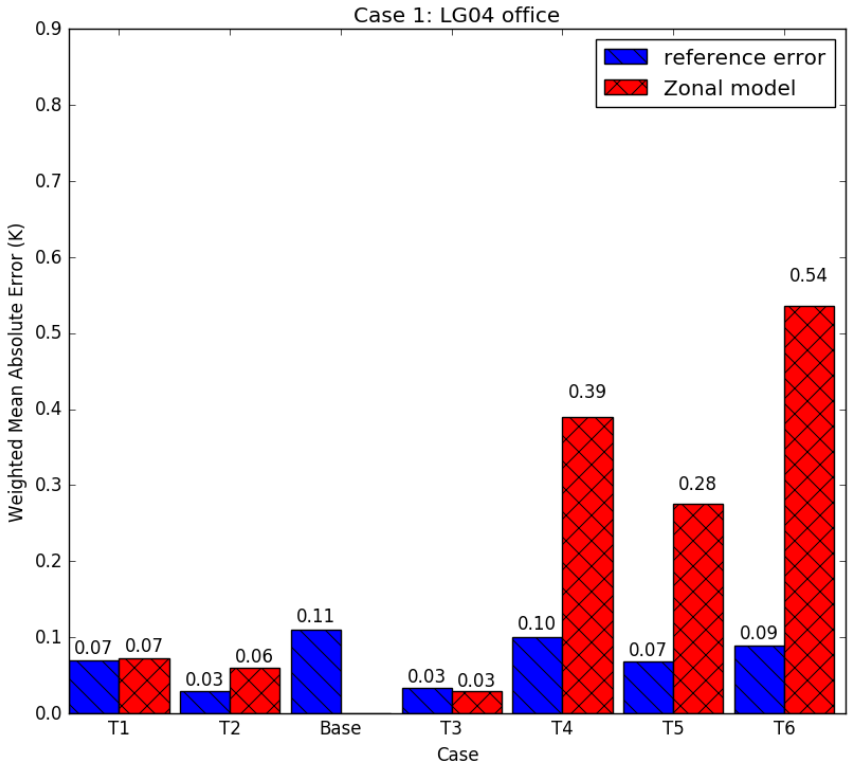


Figure 5-13: Weighted Mean Absolute Error of a 24-sub-zones zonal model against the original CFD simulation for case study 1. The temperature at the inlet was changed between 35 °C (T1) and 60 °C (T6)

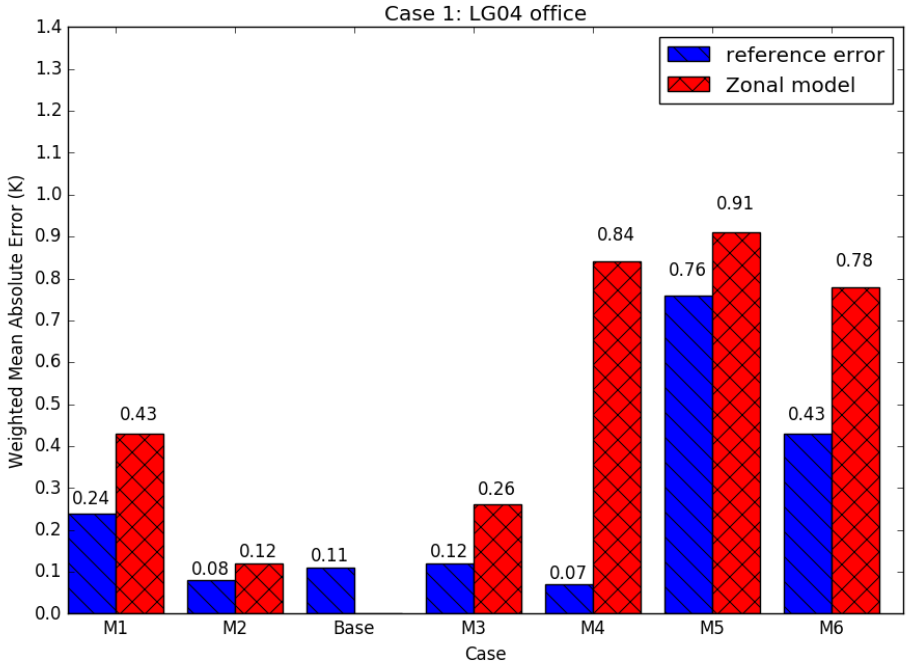


Figure 5-14: Weighted Mean Absolute Error of a 24-sub-zones zonal model against the original CFD simulation for case study 1. The mass flow rate at the inlet was changed between 0.0528 kg.s^{-1} (M1) and 0.192 kg.s^{-1} (M6)

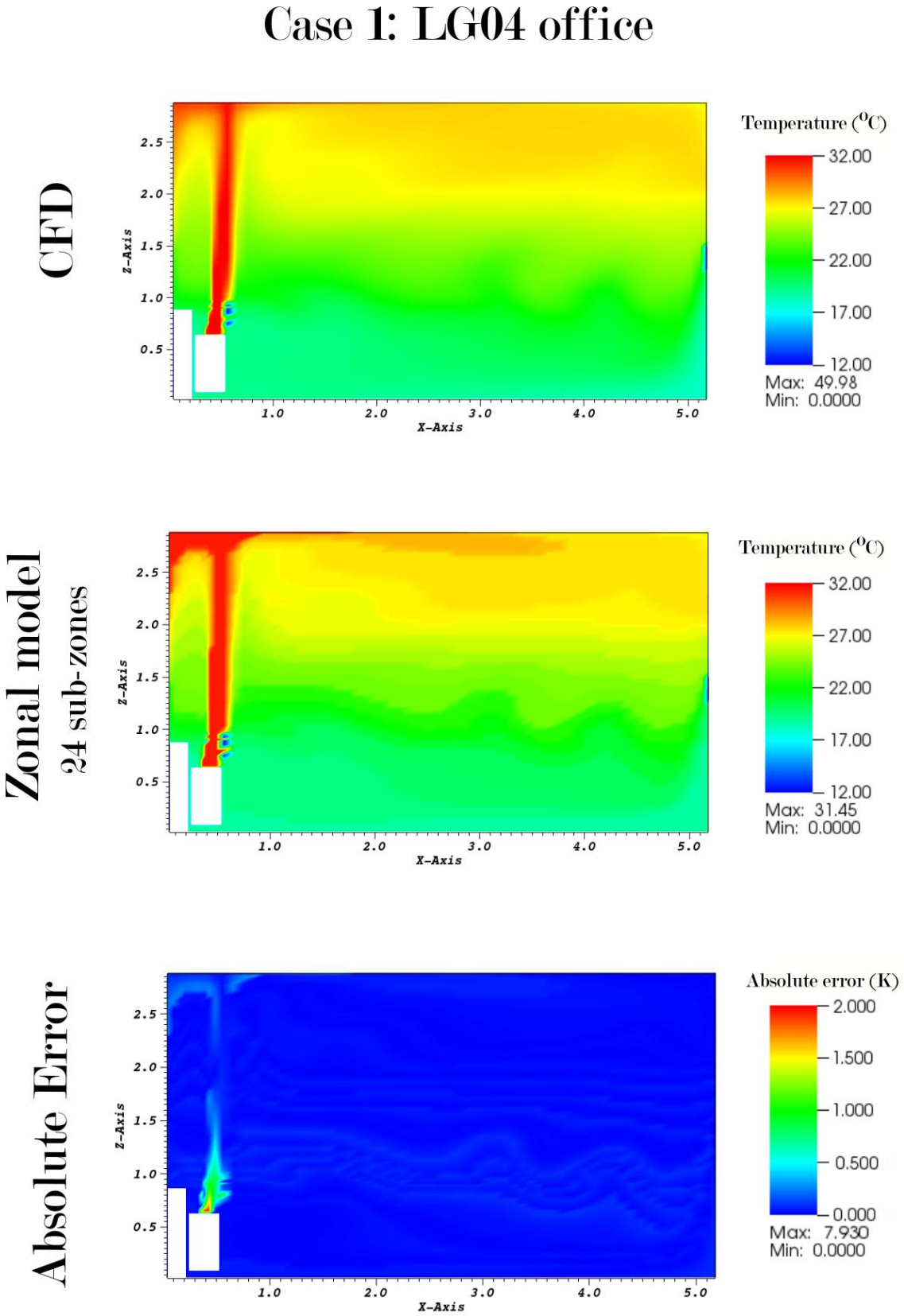


Figure 5-15: CFD temperature predictions, Zonal model temperature predictions and error between CFD and zonal model for case study 1, XZ slice taken at Y= 1.17m. The cold cells above the convector are an output error.

The same analysis has been carried out for case studies 2 and 3. Figure 5-16 shows for case study 2 the error between the zonal model predictions and CFD when the temperature of the inlet is changed between 14.82 °C (T1) and 26.82 °C (T4) as summarized in Table 4-4. The zonal model is extracted from the base case where T_{inlet} = 20.82 °C and $\dot{m}_{inlet}$ = 0.184 kg.s^{-1}. The error is in this case higher than in case study 1, but it remains globally under 0.7 K for these temperature ranges. Figure 5-17 shows the same analysis when the mass flow rate of the inlet is changed between 0.09828 kg.s^{-1} (M1) and 0.2293 kg.s^{-1} (M4) as summarized in Table 4-4. The zonal model is extracted from the base case where T_{inlet} = 20.82 °C and $\dot{m}_{inlet}$ = 0.1638 kg.s^{-1}. The absolute error remains under 0.25 K for all the cases. The zonal model has 52 sub-zones and has accurate temperature predictions for most of the domain, but ultimately fails to detect the heat sources in the domain. Figure 5-18 highlights this issue by showing XY and XZ slices taken respectively at Z = 1.1m and Y = 2.9m where the heat plumes above one person and one laptop are clearly visible on the CFD temperature predictions but not on the zonal model predictions. The error plot highlights that the plumes and the resulting thermal effects are captured by the zonal model, but their temperature is not simulated correctly which suggests that the method did not capture the heat source in the CFD results.

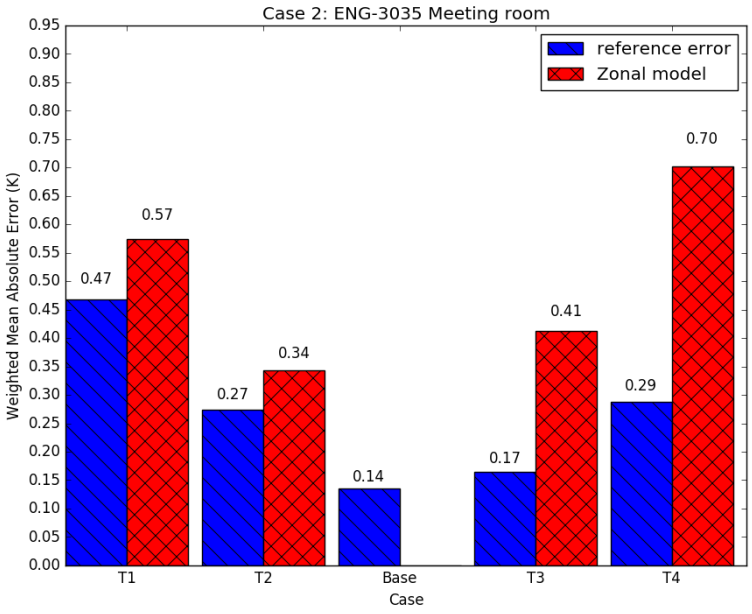


Figure 5-16: Weighted Mean Absolute Error of a 52-sub-zones zonal model against the original CFD simulation for case study 2. The temperature at the inlet was changed between 14.82 °C (T1) and 26.82 °C (T4)

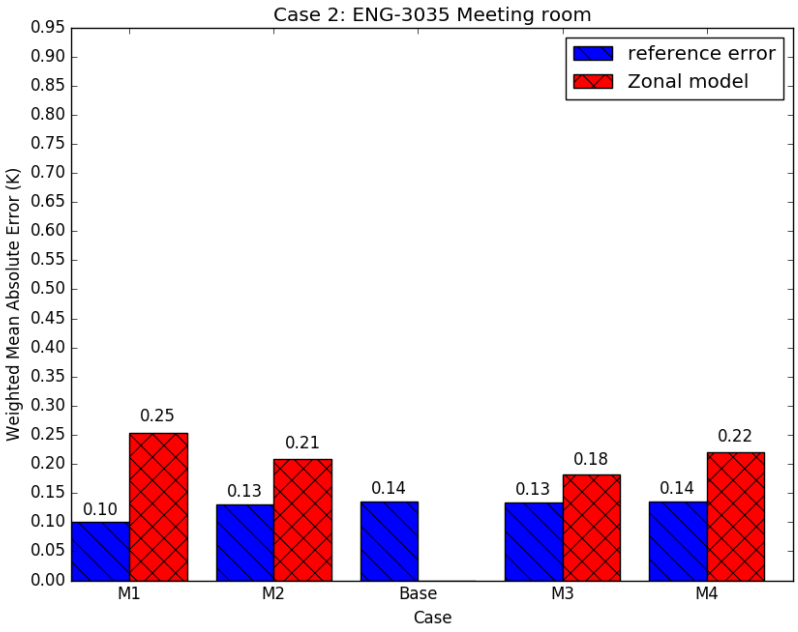


Figure 5-17: Weighted Mean Absolute Error of a 52-sub-zones zonal model against the original CFD simulation for case study 2. The mass flow rate at the inlet was changed between 0.09828 kg.s^{-1}(M1) and 0.2293 kg.s^{-1}(M4)

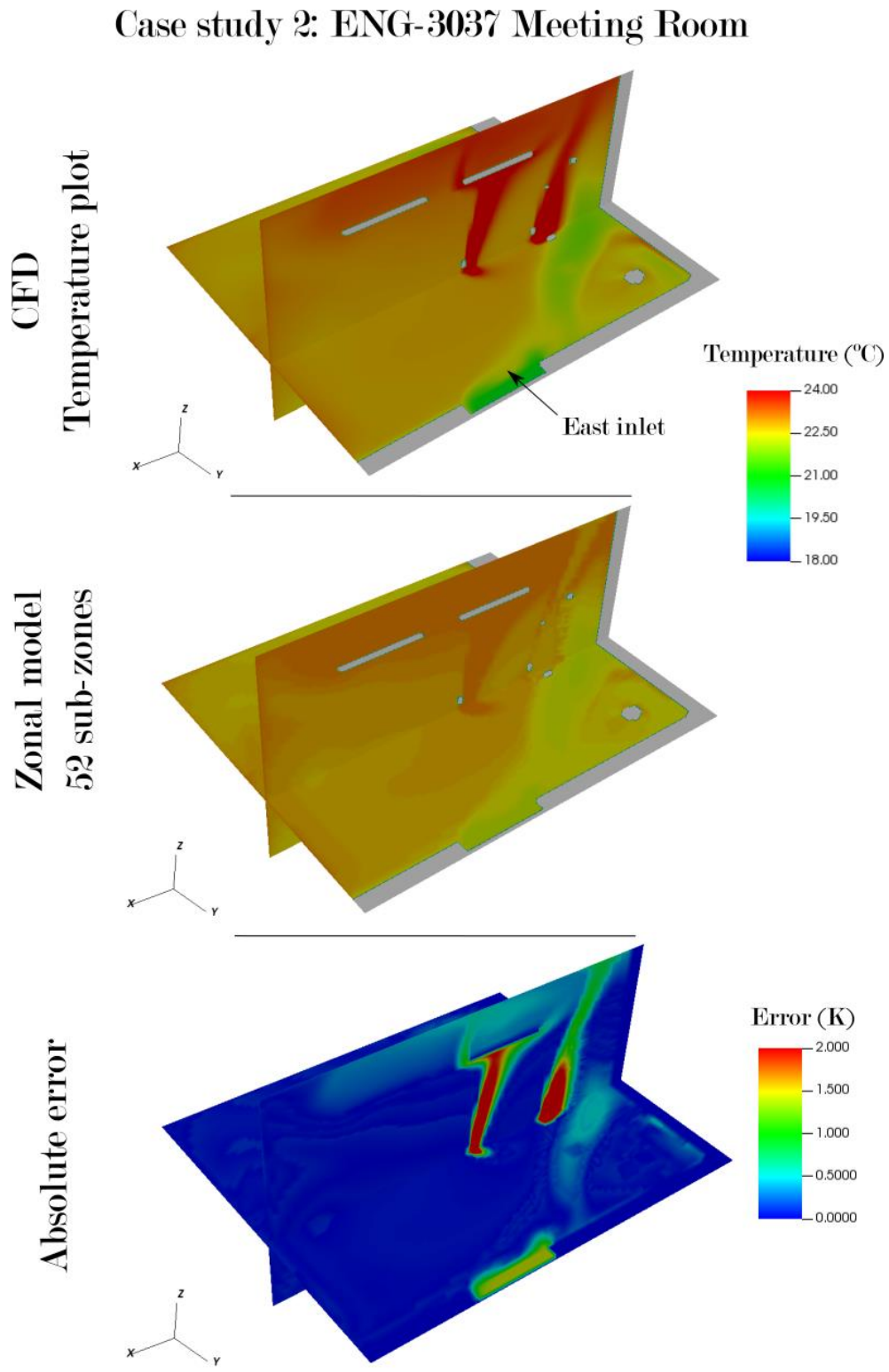


Figure 5-18: CFD temperature distribution predictions (Top) and zonal model predictions (middle) for case study 2, and error between CFD and zonal model (bottom). The XY and XZ slices were taken respectively at Z = 1.1m and Y = 2.9m. The error plot clearly shows the heat sources have not been captured by the zonal model.

Finally, for case study 3 a zonal model was extracted from the base case where T_{inlet} = 24°C and m_{inlet} = 0.18 kg.s^{-1}, and has 67 sub-zones. Figure 5-19 shows the error of the

zonal model's temperature predictions versus CFD when the temperature of the inlet is varied between 21 °C (T1) and 33 °C (T4) as summarized in Table 4-6. Figure 5-20 shows the error of the zonal model when the mass flow rate at the inlet is changed between 0.12 $kg.s^{-1}$ (M1) and 0.25 $kg.s^{-1}$ (M6). The results show that the zonal model is able to accommodate large mass flow rate changes for this simple case, with a mean absolute error under 0.35 K for all the cases. Figure 5-21 shows the CFD and zonal model temperature predictions, as well as the error between zonal model and CFD.

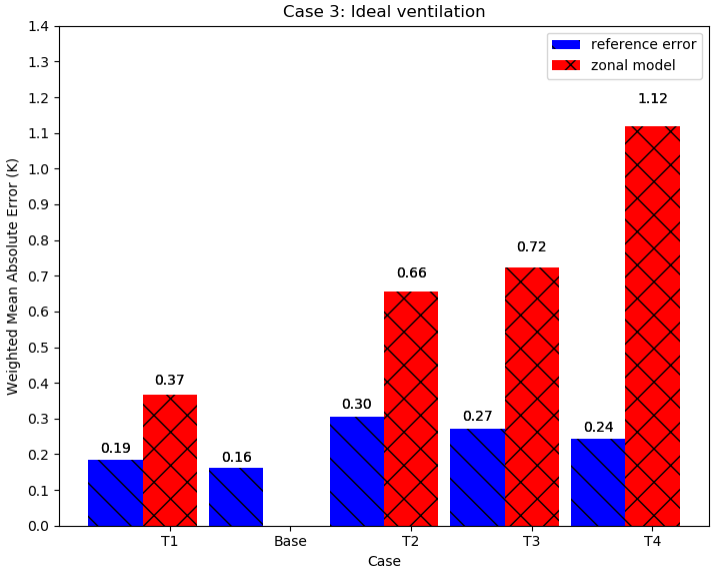


Figure 5-19: Weighted Mean Absolute Error of a 67-sub-zones zonal model against the original CFD simulation for case study 3. The temperature at the inlet was changed between 21 °C (T1) and 33 °C (T4)

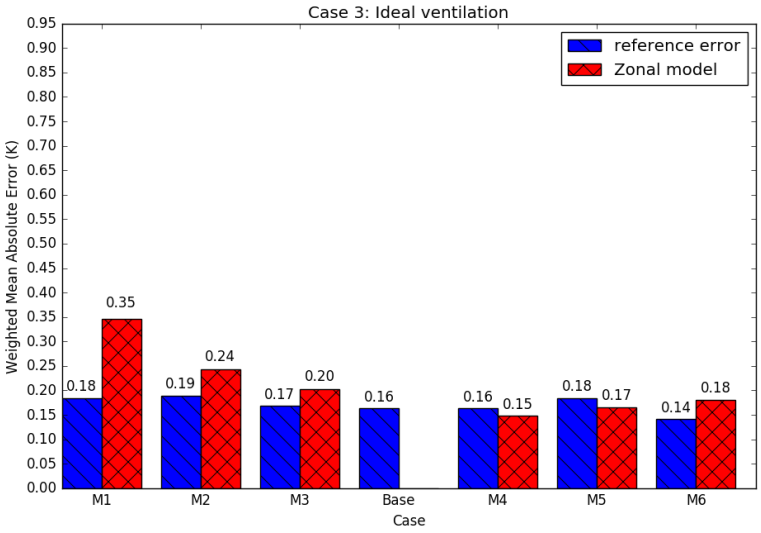


Figure 5-20: Weighted Mean Absolute Error of a 67-sub-zones zonal model against the original CFD simulation for case study 3. The mass flow rate at the inlet was changed between 0.12 kg.s-1 (M1) and 0.25 kg.s-1 (M6)

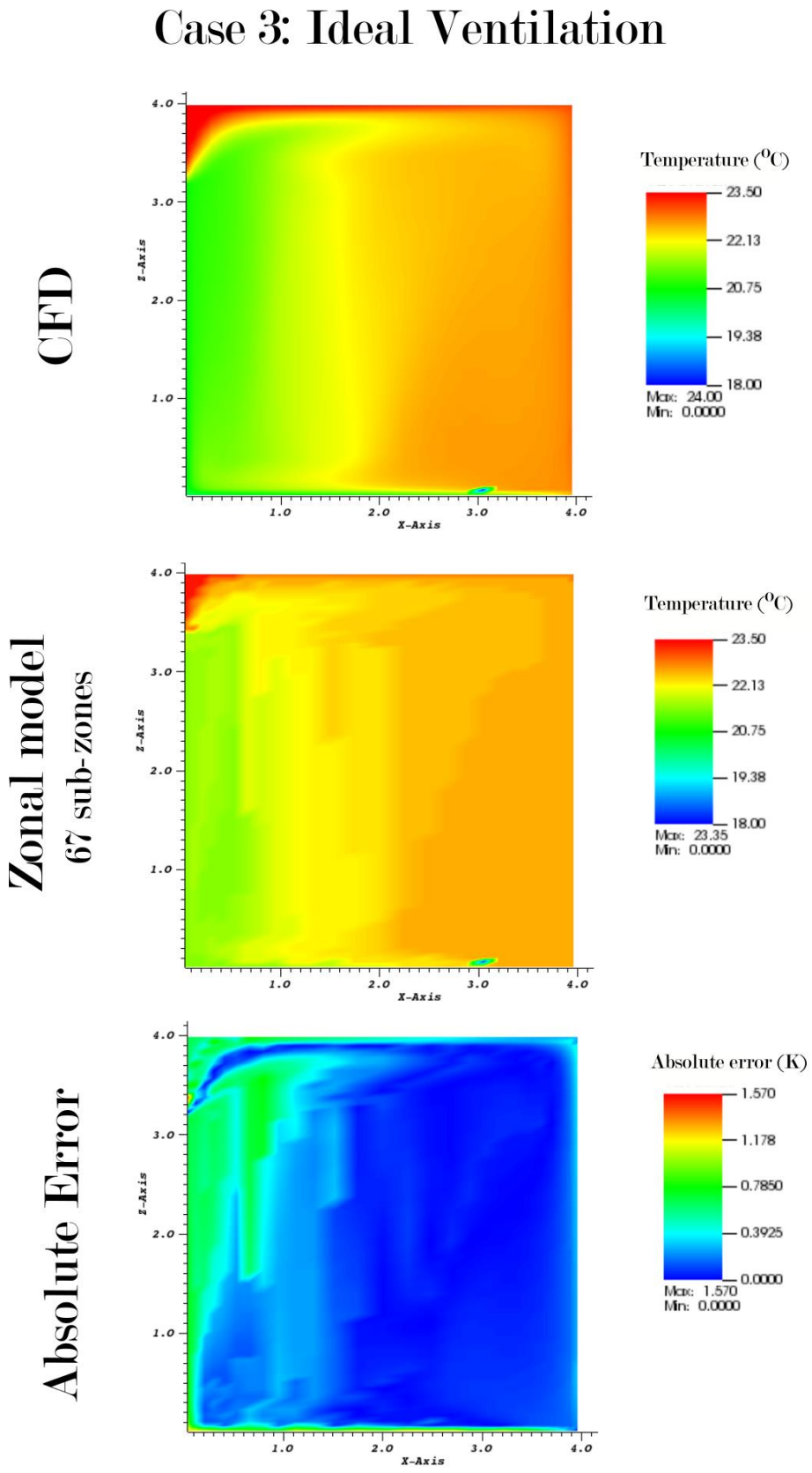


Figure 5-21: CFD temperature distribution predictions (Top) and zonal model predictions (middle) for case study 3, and error between CFD and zonal model (bottom). The XZ slice is taken at the centre of the domain in the Y direction, at Y = 2m.

The sub-zones are immutable once they have been generated, therefore they cannot efficiently accommodate changes in flow field. This is the case when the inlet behaves like a jet, where it penetrates the domain more and more as the mass flow rate increases. This is shown clearly by the asymmetry of the error when increasing the mass flow rate versus decreasing mass flow rates in case study 1. When the mass flow rate is decreased, sub-zones that were created for a larger jet are sufficient to capture a smaller jet while when the mass flow is increased the opposite happens. Sub-zones created initially for a smaller jet are not sufficient anymore to correctly predict the higher penetration of air in the domain. Figure 5-22 shows an error plot at the air inlet of case study 3, of a 67 sub-zones zonal model extracted from the base case ($\dot{m}_{inlet}$ = 0.18 kg.s^{-1}) and solved for mass flow rates of 0.18 kg.s^{-1}, 0.20 kg.s^{-1}, 0.22 kg.s^{-1} and 0.25 kg.s^{-1}. This figure shows that when the mass flow rate increases, the local error near the inlet increases as well.

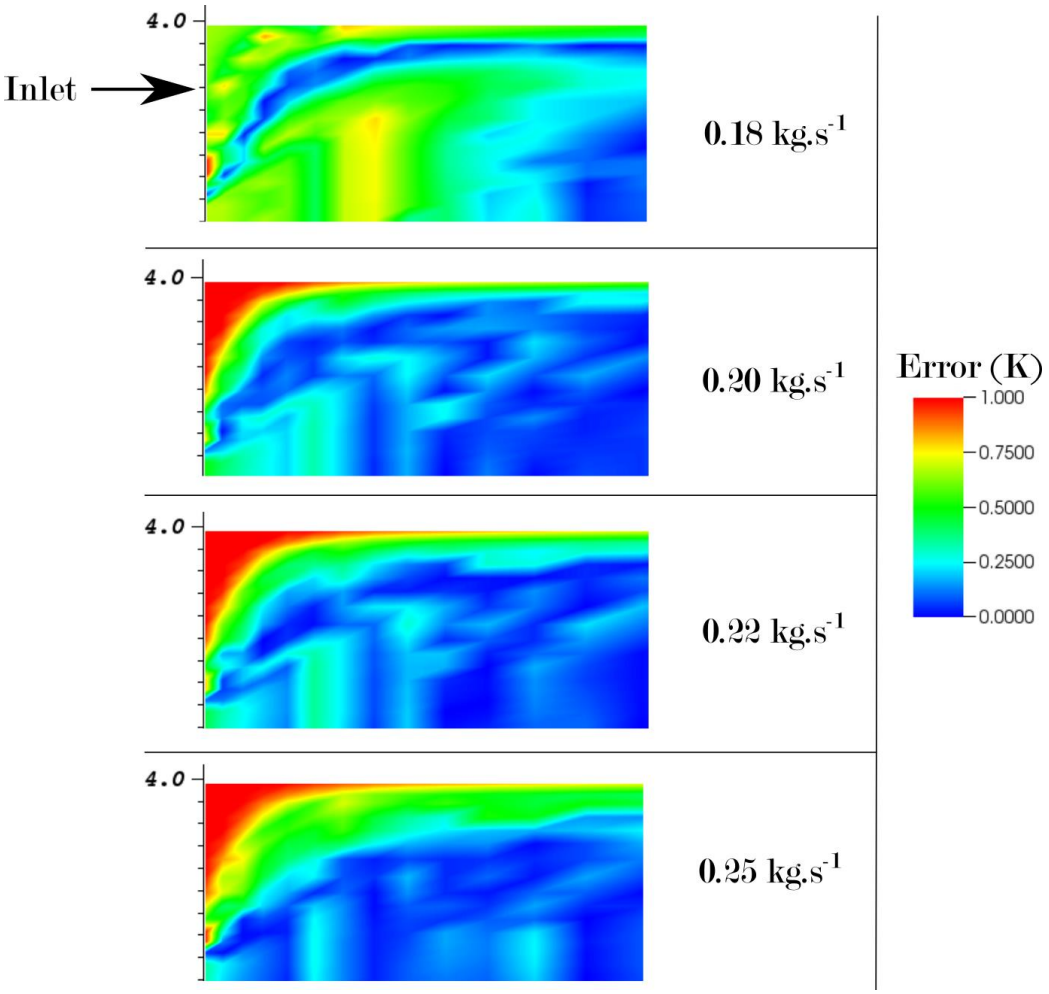


Figure 5-22: magnified view of the error at the inlet of case study 3, in the case of a zonal model extracted from the base case and solved for mass flow rates of 0.18 kg.s^{-1} to 0.25 kg.s^{-1}

5.5. Summary of the Results

This chapter has shown the results of three analysis: (1) a performance comparison between the three clustering algorithms presented in the methodology, (2) an assessment of the influence of sub-zone numbers on the error and time to solution, and (3) the fidelity against CFD when solving the zonal model for different boundary conditions parameters using the three case studies presented in Chapter 4 .

This chapter has shown how the MVS clustering method has advantages over the CG and CW methods and was therefore selected as the clustering method for the work presented in this doctoral thesis. Then, the analysis on sub-zone number and time to solution has shown how the weighted mean absolute error in each case study always converged around 0.1 to 0.2 K for zonal models solved in approximately 0.8 seconds. Furthermore, the effects of the number of sub-zones on the local error have been shown, particularly around inlets and outlets where increasing the number of sub-zones leads generally to a lower local error. Finally, the analysis on the fidelity of zonal models when solved for different boundary conditions has shown that the zonal models provide good predictions of temperature distributions when compared to CFD, except in case study 2 where the method failed to capture heat sources. The next chapter will provide a discussion of the results.

Chapter 6 Discussion

This chapter presents the advantages and shortcomings of the method. It provides a discussion of the performance of each clustering algorithm, the flexibility of the zonal models and identifies issues with the extraction of complex geometries and the use of unstructured meshes.

6.1. Clustering methods

In this work, three clustering methods have been developed in order to find a computationally inexpensive algorithm that could capture most of the features of the fluid domain. The MVS method has shown satisfactory performance, because of its ability to capture both the features of nearly uniform volumes as well as the large temperature gradients that are present around heat sources.

When defining zone types the MVS method creates a series of intervals of temperature that allow the method to cluster cells that belong to the same nearly isothermal volume of air. It therefore captures the thermal distribution of the domain, and the zone type definition step guarantees that the isothermal volumes are evenly distributed in the domain therefore capturing both high and low gradients of temperature. This allows the method to finely capture features such as stratified air and heat plumes without intervention from the user except for the definition of the desired number of sub-zones. It is an improvement over other methods that require the user to define volumes of interest around domain features such as heat sources, stratified air or jets and the corresponding equations to be used in the zonal model solver. MVS does not require the user to specify pre-defined models or sub-zone geometries for sub-zones near heat sources, boundaries or jets as the isothermal volumes are automatically captured which provides more flexibility. However, the immutability of the sub-zones once they are created renders impossible the accurate prediction of temperature distributions when the flow feature changes, such as when the parameters of a domain inlet are changed. The error is low

when the inlet parameters do not change significantly but quickly increases as the parameters deviate from the base case.

The CG method proved to have unsatisfactory accuracy in this work because the grid was defined independently of temperature distribution or flow features. The level of error when compared to CFD of the MVS method however suggests that the CG method could be improved. In fact, the most notable difference between CG and MVS is the fact that MVS finely captures isothermal volumes, therefore an improvement of the mesh definition step in CG would possibly lead to a lower level of error. This improvement could include the use of an octree partitioning method [67], which would subdivide the coarse cells into multiple cells when the temperature gradient within a coarse cell is too large. By doing so, the mesh would be refined where the temperature distribution is less uniform while maintaining large coarse cells where the domain is more uniform.

The CW method has proven unsatisfactory in this work because of the difficulty to implement a method for selecting seeds in the domain that would allow to expand cell clusters that could capture temperature distributions. The selection of local minima and maxima in a 4-dimensional dataset is time-consuming, and the expansion of clusters as it was implemented does not allow to define sub-zones of intermediate temperature as opposed to the MVS method.

The MVS method currently considers one zone criteria at a time, and this work used temperature. Weber et al. [47] have shown that in some applications the zonal models were more accurate when sub-zones were initialized using one criterion, then subdivided again using a second criterion. The method would certainly benefit from the development of a similar multi-step sub-zones generation, since it has been shown to provide more accurate predictions. Particularly, this study has highlighted edge cases directly caused by the single zone criterion method such as the cases where the temperature distribution was too uniform.

6.2. Flexibility and computational efficiency

The proposed method was applied to 3 case studies and the results were consistent. However, this highlighted several factors that lead to an increased level of error. The first is the immutability of sub-zones which has been discussed in the previous section. Then, the fact that the error when boundary temperatures were changed suggest that the fixed UA assumption was incorrect. The user could define a variable UA value in Sinda/FLUINT by means of custom solution algorithms implemented through Fortran subroutines, which was not developed in this work. Sinda/FLUINT also offers the possibility to implement fluid to fluid heat exchange, but it was not implemented in this work as Sinda/FLUINT's manual suggested that for air it would be negligible when compared to other energy exchanges within the fluid network.

In the three case studies it was possible to extract a zonal model that could be solved in under one second while reaching a low level of error. This would enable the use of the method in large-scale simulations such as yearly simulations while providing detailed information on temperature distributions. This would allow the implementation of thermal comfort in yearly energy consumption simulations, as well as providing more accurate information on boundary conditions inside building zones compared to the uniform instantly well-mixed assumption. The benefits of coupling CFD to energy simulations have been shown in [68] and the proposed method could be used in place of most of the CFD simulations used in coupled models. CFD co-simulation is used to provide snapshots of fluid flow predictions for energy simulation in order to tackle the problematics of the uniform and instantly well-mixed assumption. Using the proposed method would render simulations more rapid at the cost of a moderate error in temperature distribution predictions.

6.3. Complex Geometries

Case study 2 has shown that the current iteration of the method was not able to capture 3D geometries that were not hexahedrons. In the information extracted from the CFD simulations, objects are defined by listing the coordinates of the nodes located at their vertices. This information is not sufficient to define a straightforward approach to extracting 3D shapes, since a connections array is necessary to correctly extract them. In fact, a list of vertices alone is not sufficient to define the shape of a complex 3D object. Eventually, it must be possible to define an algorithm for extracting shapes that can be projected to a 2D plane or shapes that are the result of the extrusion of a 2D shape, but this would represent a significant amount of work that is not within the scope of this research.

Additionally, the Python script presents a bottleneck when confronted with multiple intersecting 3D objects such as with ANGLED-IN/OUT objects. Currently the Python script records the nodes constituting each object, and for each new object it compares the current list of nodes to all the existing objects. This comparison of large sets of nodes is a time-consuming step. If it does find an intersection, the Python script compares the two objects again to find the intersecting nodes and decide which ones should be kept in which object. This effect is unnoticeable for 2D objects, but it is not the case of 3D objects, because the addition of a 3rd dimension quickly increases the number of nodes in each object and therefore the number of nodes to compare.

6.4. Structured and Unstructured meshes

As of now, the method relies on cartesian structured grids, which is not the most common type of grids. Many CFD packages such as Ansys Fluent typically use unstructured grids. It would be worth investigating the use of unstructured grids, and the constraints it would impose on clustering methods.

6.5. Automation

As highlighted in Chapter 2, Section 2.3.1 the main drawback of zonal methods is the time and expertise required to define a zonal model. It places zonal methods on the same level as CFD in terms of investments, therefore CFD is often preferred to zonal models. By allowing a high degree of automation in the generation of zonal models, the method proposed in this doctoral thesis is an improvement over the current methods with the *sin equa non* condition that at least one CFD simulation is available to extract a zonal model from.

However, the method is not strictly automated yet, since two manual steps are needed: the manual selection of the number of sub-zones as well as the interface between the Python script extracting the zonal model and the solver. A possible solution to automatically selecting the number of sub-zones would be to (1) generate a range of zonal models for different numbers of sub-zones, and (2) solve them to find the number of sub-zones above which generating more sub-zones does not significantly lower the error of the zonal model. This would render the extraction step more time consuming, since it would need to be repeated several times, but it would negate the need for human intervention at this step. In order to automate the solution of the zonal model, solutions to automatically call Sinda/FLUINT's solver from within the Python script should be explored, as well as the possibility to change solver entirely.

6.6. Summary of the Discussion

This chapter presented a discussion on the results obtained from the three case studies presented in Chapter 4 Section 6.1 presented a justification for selecting the MVS algorithm for clustering CFD cells, since it produced the most accurate zonal models by clustering cells according to the temperature distribution in the domain at a low computational cost. Then, the study on the influence of sub-zone numbers on the accuracy of the zonal models showed that, for the three cases, there was a point at which further incrementing sub-zone numbers did not have a significant impact on accuracy but

increased time to solution greatly. The limitations of the method are the definition of sub-zones using a single zoning criterion and the immutability of sub-zones once they are generated. It provided advice on implementing multiple criteria clustering using MVS. The next chapter will present the conclusions of this doctoral thesis, as well as prospects for future developments of the current research.

Chapter 7 Conclusion

The work proposed in this doctoral thesis has demonstrated a method to automatically extract zonal models from CFD simulations and solve them in under one second for boundary conditions parameters different from the original CFD parameters. The method is based on the generation of clusters from the computational cells of CFD simulations, the generation of a zonal model and the extraction of parameters for the heat and mass transfers between sub-zones and with the domain. The zonal model can provide predictions of temperature distributions for off-design parameters and therefore leverage CFD's accuracy at a fraction of the computational cost. The results have shown that the method allows the prediction of temperature distributions in indoor spaces with a good level of accuracy, often below 0.6 K of weighted mean absolute error. One of the principal advantages of the method is its level of automation, which allows the semi-automatic generation of a zonal model along with its parameters without the need of user expertise and in a very short time. CFD simulations are typically expensive with respects to cost and time, and this method provides an additional use for CFD simulations, increasing their benefits when simulating the built environment.

The current envisioned use for the proposed method is the optimization of thermal comfort in buildings, since it is able to provide accurate predictions of thermal distributions. Currently, designers obtain predictions of thermal distributions by means of CFD simulations. In practice, the method would be applied to an initial CFD simulation and would be used to rapidly simulate a wider range of operating conditions instead of solving multiple CFD simulations. This would provide insight to the building designer and rapidly inform them on the outcomes of different design choices or operating conditions. The method, thanks to its rapidity, can be used in optimization problems. Such problems include the definition of an optimal air supply temperature for example. In the recent years work has focused on sharing the workflow across the multiple actors involved in the design and commissioning of a Smart Building through cloud applications and a clever usage of IT resources. The low computational cost of the method and its usage of low-level programming tools would ease its integration in such a workflow. Then, the low level of expertise required to apply the proposed method would allow its usage by many of the designers, including for example architects, provided that the

aforementioned integrated workflows also include an access to validated CFD simulations. Since airflow and thermal distribution have critical effects on a building liveability, providing a wider access to that information could improve the comfort of modern buildings by allowing more actors to be aware of their design choices with that regard.

A pathway to practical implementation would start by developing the features proposed in Chapter 6 in order to improve the usefulness of the results, accuracy of the method, necessary expertise and level of automation. Then, work should focus on the integration of the method to available cloud-based tools, specifically with regards to the model solver, the computational power required by the method, and the server-specific requirements. Once the method can be demonstrated to work in such environments, discussion should start with the various building design experts to integrate it with available workflow methods with regards to its dependence on CFD and its usefulness for early design stages.

References

[1] 2018 Revision of World Urbanization Prospects, 2018. https://www.un.org/development/desa/publications/2018-revision-of-world-urbanization-prospects.html, (accessed August 20, 2019).

[2] U.S. Energy Information Administration, Monthly Energy Review, 2019. https://www.eia.gov/totalenergy/data/monthly/pdf/sec2_3.pdf, (accessed January 5, 2020).

[3] G. Brown, The BRIS simulation program for thermal design of buildings and their services, Energy Build. 14 (1990) 385–400. https://doi.org/10.1016/0378-7788(90)90100-W.

[4] R. Evins, A review of computational optimisation methods applied to sustainable building design, Renew. Sustain. Energy Rev. 22 (2013) 230–245. https://doi.org/10.1016/J.RSER.2013.02.004.

[5] P.H. Shaikh, N.B.M. Nor, P. Nallagownden, I. Elamvazuthi, T. Ibrahim, A review on optimized control systems for building energy and comfort management of smart sustainable buildings, Renew. Sustain. Energy Rev. 34 (2014) 409–429. https://doi.org/10.1016/j.rser.2014.03.027.

[6] H.H.L. Kwok, J.C.P. Cheng, A.T.Y. Li, J.C.K. Tong, A.K.H. Lau, Multi-zone indoor CFD under limited information : An approach coupling solar analysis and BIM for improved accuracy, J. Clean. Prod. (2019). https://doi.org/10.1016/j.jclepro.2019.118912.

[7] K.W. Tham, Indoor air quality and its effects on humans—A review of challenges and developments in the last 30 years, Energy Build. 130 (2016) 637–650. https://doi.org/10.1016/J.ENBUILD.2016.08.071.

[8] L. Lan, P. Wargocki, D.P. Wyon, Z. Lian, Effects of thermal discomfort in an office on perceived air quality, SBS symptoms, physiological responses, and human performance, Indoor Air. 21 (2011) 376–390. https://doi.org/10.1111/j.1600-0668.2011.00714.x.

[9] L. (Leon) Wang, Q. Chen, Evaluation of some assumptions used in multizone airflow network models, Build. Environ. 43 (2008) 1671–1677. https://doi.org/10.1016/J.BUILDENV.2007.10.010.

[10] D. Etheridge, A perspective on fifty years of natural ventilation research, Build. Environ.

91 (2015) 51–60. https://doi.org/10.1016/J.BUILDENV.2015.02.033.

[11] Q. Chen, Ventilation performance prediction for buildings: A method overview and recent applications, Build. Environ. 44 (2009) 848–858. https://doi.org/10.1016/J.BUILDENV.2008.05.025.

[12] J. Ma, J. Qin, T. Salsbury, P. Xu, Demand reduction in building energy systems based on economic model predictive control, Chem. Eng. Sci. 67 (2012) 92–100. https://doi.org/10.1016/J.CES.2011.07.052.

[13] R. Yokoyama, T. Wakui, R. Satake, Prediction of energy demands using neural network with model identification by global optimization, Energy Convers. Manag. 50 (2009) 319–327. https://doi.org/10.1016/J.ENCONMAN.2008.09.017.

[14] S. Royer, S. Thil, T. Talbert, M. Polit, A procedure for modeling buildings and their thermal zones using co-simulation and system identification, Energy Build. 78 (2014) 231–237. https://doi.org/10.1016/J.ENBUILD.2014.04.013.

[15] D.B. Crawley, J.W. Hand, M. Kummert, B.T. Griffith, Contrasting the capabilities of building energy performance simulation programs, Build. Environ. 43 (2008) 661–673. https://doi.org/10.1016/J.BUILDENV.2006.10.027.

[16] O.M. Brastein, D.W.U. Perera, C. Pfeifer, N.-O. Skeie, Parameter estimation for grey-box models of building thermal behaviour, Energy Build. 169 (2018) 58–68. https://doi.org/10.1016/J.ENBUILD.2018.03.057.

[17] P. Bacher, H. Madsen, Identifying suitable models for the heat dynamics of buildings, Energy Build. 43 (2011) 1511–1522. https://doi.org/10.1016/J.ENBUILD.2011.02.005.

[18] A.C. Megri, F. Haghighat, Zonal Modeling for Simulating Indoor Environment of Buildings: Review, Recent Developments, and Applications, HVAC&R Res. 13:6 (2007) 887–905.

[19] E. Wurtz, Modélisation tridimensionnelle des transferts thermiques orienté objet, Ecole Nationale des Ponts et Chaussées, 1995. https://pastel.archives-ouvertes.fr/tel-00523621. (accessed June 15, 2019)

[20] M. Musy, Génération automatique de modèles zonaux pour l ' étude du comportement thermo-aéraulique des bâtiments, Université de la Rochelle, 1999. https://tel.archives-ouvertes.fr/tel-00492772 (accessed June 15, 2019).

[21] H. Huang, F. Haghighat, C.-S. Lee, An integrated zonal model for predicting indoor airflow, temperature, and VOC distributions, ASHRAE Trans. 111 (2005) 601–611.

[22] L. Georges, M. Thalfeldt, Ø. Skreiberg, W. Fornari, Validation of a transient zonal model to predict the detailed indoor thermal environment: Case of electric radiators and wood stoves, Build. Environ. 149 (2019) 169–181. https://doi.org/10.1016/J.BUILDENV.2018.12.020.

[23] N. Björsell, A. Bring, L. Eriksson, P. Grozman, M. Lindgren, P. Sahlin, A. Shapovalov, M. Vuolle, IDA Indoor Climat and Energy, in: Build. Simul., 1999: pp. 1035–1042.

[24] H. Fang, D. Zhao, G. Tan, A. Denzer, Study of underfloor air distribution using zonal model-based simulation and experimental measurements, Energy Build. 152 (2017) 96–107. https://doi.org/10.1016/j.enbuild.2017.07.026.

[25] Y. Wang, Y. Chen, C. Li, Airflow modeling based on zonal method for natural ventilated double skin façade with Venetian blinds, Energy Build. 191 (2019) 211–223.

[26] F. Song, B. Zhao, X. Yang, Y. Jiang, V. Gopal, G. Dobbs, M. Sahm, A new approach on zonal modeling of indoor environment with mechanical ventilation, Build. Environ. 43 (2008) 278–286. https://doi.org/10.1016/j.buildenv.2006.04.026.

[27] J. Lau, Q. Chen, Floor-supply displacement ventilation for workshops, Build. Environ. 42 (2007) 1718–1730. https://doi.org/10.1016/J.BUILDENV.2006.01.016.

[28] L. Li, C.M. Mak, The assessment of the performance of a windcatcher system using computational fluid dynamics, Build. Environ. 42 (2007) 1135–1141. https://doi.org/10.1016/J.BUILDENV.2005.12.015.

[29] B. Zhao, P. Guan, Modeling particle dispersion in personalized ventilated room, Build. Environ. 42 (2007) 1099–1109. https://doi.org/10.1016/J.BUILDENV.2005.11.009.

[30] Z. Du, P. Xu, X. Jin, Q. Liu, Temperature sensor placement optimization for VAV control using CFD–BES co-simulation strategy, Build. Environ. 85 (2015) 104–113. https://doi.org/10.1016/J.BUILDENV.2014.11.033.

[31] Z. Zhai, Q. Chen, P. Haves, J.H. Klems, On approaches to couple energy simulation and computational fluid dynamics programs, Build. Environ. 37 (2002) 857–864. https://doi.org/10.1016/S0360-1323(02)00054-9.

[32] L.J. Lo, D. Banks, A. Novoselac, Combined wind tunnel and CFD analysis for indoor air fl ow prediction of wind-driven cross ventilation, Build. Environ. 60 (2013) 12–23. https://doi.org/10.1016/j.buildenv.2012.10.022.

[33] Y. Zhang, L. Huang, Y. Zhou, Analysis of Indoor Thermal Comfort of Test Model Building Installing Double-Glazed Window with Curtains Based on CFD, Procedia Eng.

121 (2015) 1990–1997. https://doi.org/10.1016/j.proeng.2015.09.197.

[34] L. Dong, H. Zuo, L. Hu, B. Yang, L. Li, L. Wu, Journal of Loss Prevention in the Process Industries Simulation of heavy gas dispersion in a large indoor space using CFD model, J. Loss Prev. Process Ind. 46 (2017) 1–12. https://doi.org/10.1016/j.jlp.2017.01.012.

[35] X. Zhao, W. Liu, D. Lai, Q. Chen, Optimal design of an indoor environment by the CFD-based adjoint method with area-constrained topology and cluster analysis, Build. Environ. 138 (2018) 171–180. https://doi.org/10.1016/j.buildenv.2018.04.033.

[36] F. Cao, L. Zhang, A. Pina, P. Ferrão, J. Fournier, B. Lacarrière, O. Le Corre, Study of transient indoor temperature for a HVAC room using a and room Cooling using a Study of transient indoor temperature for a Heating HVAC modified CFD method, Energy Procedia. 160 (2019) 420–427. https://doi.org/10.1016/j.egypro.2019.02.176.

[37] X. Yang, Y. Zhang, J. Hang, Y. Lin, M. Mattsson, M. Sandberg, M. Zhang, K. Wang, Integrated assessment of indoor and outdoor ventilation in street canyons with naturally-ventilated buildings by various ventilation indexes, Build. Environ. 169 (2020). https://doi.org/10.1016/j.buildenv.2019.106528.

[38] D.J. Lucia, P.S. Beran, W.A. Silva, Reduced-order modeling - New approaches for computational physics, 39th Aerosp. Sci. Meet. Exhib. 40 (2001) 51–117. https://doi.org/10.2514/6.2001-853.

[39] T.R. Report, A.J. Newman, Model reduction via the Karhunen-Loeve expansion part an exposition., Tech. Inst. Syst. Res. Univ. Maryl. 1 SRC-G (1996) 32–96.

[40] K. Li, W. Xue, C. Xu, H. Su, Optimization of ventilation system operation in office environment using POD model reduction and genetic algorithm, Energy Build. 67 (2013) 34–43. https://doi.org/10.1016/J.ENBUILD.2013.07.075.

[41] A. Sempey, C. Inard, C. Ghiaus, C. Allery, Fast simulation of temperature distribution in air conditioned rooms by using proper orthogonal decomposition, Build. Environ. 44 (2009) 280–289. https://doi.org/10.1016/J.BUILDENV.2008.03.004.

[42] A. Robert, J. Henderson, C. Turnbull, An Implicit Time Integration Scheme for Baroclinic Models of the Atmosphere, Mon. Weather Rev. 1 (1972) 329–335.

[43] W. Zuo, Q. Chen, Real-time or faster-than-real-time simulation of airflow in buildings, Indoor Air. 19 (2009) 33–44. https://doi.org/10.1111/j.1600-0668.2008.00559.x.

[44] M. Jin, Q. Chen, Improvement of fast fluid dynamics with a conservative semi-Lagrangian scheme, Int Jnl Num Meth HFF. 25 (2015) 2–18. https://doi.org/10.1108/hff-04-2013-

0119.

[45] W. Tian, T.A. Sevilla, W. Zuo, M.D. Sohn, Coupling fast fluid dynamics and multizone airflow models in Modelica Buildings library to simulate the dynamics of HVAC systems, Build. Environ. 122 (2017) 269–286. https://doi.org/10.1016/J.BUILDENV.2017.06.013.

[46] Modelica, www.modelica.org (accessed July 20, 2016).

[47] B. Weber, M. von Campenhausen, T. Maßmann, A. Bednarz, A. Jupke, CFD based compartment-model for a multiphase loop-reactor, Chem. Eng. Sci. X. 2 (2019). https://doi.org/10.1016/J.CESX.2019.100010.

[48] A. Alvarado, S. Vedantam, P. Goethals, I. Nopens, A compartmental model to describe hydraulics in a full-scale waste stabilization pond, Water Res. 46 (2012) 521–530. https://doi.org/10.1016/J.WATRES.2011.11.038.

[49] T. Tajsoleiman, R. Spann, C. Bach, K. V. Gernaey, J.K. Huusom, U. Krühne, A CFD based automatic method for compartment model development, Comput. Chem. Eng. 123 (2019) 236–245. https://doi.org/10.1016/J.COMPCHEMENG.2018.12.015.

[50] D.T. Mullen, M.M. Keane, M. Geron, R.F.D. Monaghan, Automatic extraction of reduced-order models from CFD simulations for building energy modelling, Energy Build. 99 (2015) 313–326. https://doi.org/10.1016/j.enbuild.2015.04.015.

[51] H. Rosten, D. Spalding, D. Tatchell, PHOENICS: a general-purpose program for fluid-flow, heat transfer and chemical-reaction processes., CHAM. (1983).

[52] SINDA/FLUINT, (2013). www.crtech.com/sinda.html (accessed July 20, 2016).

[53] Python Software Foundation., Python Language Reference, version 2.7, (2010) (accessed July 20, 2016).

[54] Y. Cui, H.-C. Zheng, Impact of Three-Dimensional Greening of Buildings in Cold Regions in China on Urban Cooling Effect, Procedia Eng. 169 (2016) 297–302. https://doi.org/10.1016/J.PROENG.2016.10.036.

[55] L. Zhang, M. Jin, L. Zhang, Simulated study on the potential of building energy saving using the green roof, Procedia Eng. 205 (2017) 1469–1476. https://doi.org/10.1016/J.PROENG.2017.10.369.

[56] F. Guo, P. Zhu, S. Wang, D. Duan, Y. Jin, Improving Natural Ventilation Performance in a High-Density Urban District: A Building Morphology Method, Procedia Eng. 205 (2017) 952–958. https://doi.org/10.1016/J.PROENG.2017.10.149.

[57] W. Dong, D. Liang, Simulation of the Atrium Fire Smoke Flow and Research about Control Studies, Procedia Eng. 52 (2013) 92–96. https://doi.org/10.1016/J.PROENG.2013.02.111.

[58] D.B. Spalding, The Mathematical Basis of PHOENICS, http://www.cham.co.uk/phoenics/d_polis/d_lecs/general/maths.htm (accessed December 20, 2019).

[59] D.B. Spalding, PHOENICS Encyclopedia, Schemes for Convection Discretization, http://www.cham.co.uk/phoenics/d_polis/d_enc/enc_schm.htm (accessed December 20, 2019).

[60] D.B. Spalding, PHOENICS Encyclopedia, http://www.cham.co.uk/phoenics/d_polis/d_enc/encpref.htm (accessed June 28, 2019).

[61] P. Soille, L. Vincent, Determining watersheds in digital pictures via flooding simulations, Vis. Commun. Image Process. 1360 (1990) 240–250.

[62] M. Hajdukiewicz, M. Geron, M.M. Keane, Calibrated CFD simulation to evaluate thermal comfort in a highly-glazed naturally ventilated room, Build. Environ. 70 (2013) 73–89. https://doi.org/10.1016/J.BUILDENV.2013.08.020.

[63] K. Gersten, Hermann Schlichting and the Boundary-Layer Theory BT - Hermann Schlichting – 100 Years, in: R. Radespiel, C.-C. Rossow, B.W. Brinkmann (Eds.), Springer Berlin Heidelberg, Berlin, Heidelberg, 2009: pp. 3–17.

[64] S.-J. Cao, J. Meyers, On the construction and use of linear low-dimensional ventilation models, Indoor Air. 22 (2012) 427–441. https://doi.org/10.1111/j.1600-0668.2012.00771.x.

[65] T. C. Smith, S. Yancey Smith, Specification of airflow rates in laboratories, J. Chem. Heal. Saf. 16 (2009) 27–35. https://doi.org/10.1016/j.jchas.2009.04.002.

[66] T. Marzullo, S. Yousefian, M.M. Keane, M. Geron, R.F.D. Monaghan, A comparative study of computational algorithms used in the automatic generation of reduced-order models from CFD simulations, in: Build. Simul. Appl., Unibz, 2017: pp. 225–232.

[67] D.J.R. Meagher, Octree Encoding: A New Technique For The Representation, Manipulation and Display of Arbitrary 3-D Objects by Computer, Troy, 1980.

[68] D. Kim, J.E. Braun, E.M. Cliff, J.T. Borggaard, Development, Validation and Application of a Coupled Reduced-order CFD model for Building Control Applications, Build. Environ. 93 (2015) 97–111. https://doi.org/10.1016/j.buildenv.2015.05.032.