UPC **CTTC**

Parallel object-oriented algorithms for simulation of multiphysics. Application to thermal systems.

Centre Tecnològic de Transferència de Calor
Departament de Màquines i Motors Tèrmics
Universitat Politècnica de Catalunya

Joan López Mas

Doctoral Thesis

Parallel object-oriented algorithms for simulation of multiphysics. Application to thermal systems.

Joan López Mas

TESI DOCTORAL

presentada al

Departament de Màquines i Motors Tèrmics
E.T.S.E.I.A.T.
Universitat Politècnica de Catalunya

per a l'obtenció del grau de

Doctor Enginyer Industrial

Terrassa, 15 de Desembre de l'any 2015

Parallel object-oriented algorithms for simulation of multiphysics. Application to thermal systems.

Joan López Mas

Directors de la Tesi

Dr. Joaquim Rigola Serrano
Dr. Oriol Lehmkuhl Barba

Tutor

Dr. Assensi Oliva Llena

Tribunal Qualificador

Dr. Antonio Lecuona Neumann
Universidad Carlos III de Madrid

Dr. Ricard Borrell Pol
Universitat Politècnica de Catalunya

Dr. Cristóbal Cortés Gracia
Universidad de Zaragoza

A la meva dona.

Aknowledgements

Voldria donar les gràcies al Professor Assensi Oliva per donar-me l'oportunitat de realitzar aquest doctorat. Per a ell, la meva motivació en l'àrea de la transferència de calor va ser suficient per entrar a formar part del Laboratori. En aquest sentit, també vull donar les gràcies al Professor CD-Pérez Segarra per les seves classes de Termotècnia, frenètiques però apassionants. Gràcies doncs als dos, per atraure l'interès dels qui d'alguna manera estàvem avorrits fent la carrera d'Enginyeria Industrial. Agraeixo elogiosament la col·laboració continuada dels meus directors de tesi, en Quim i l'Oriol, amb qui he treballat des de l'inici del doctorat a l'any 2007. Hem tingut bons moments junts. Aquí no explicaré res però, sobre gintònics, futbolins i bbqs a Purdue ni sobre aniversaris a la Xina. Per cert, aprofito l'avinentesa per agrair-vos que no es difongués el vídeo del karaoke de Beijing, gràcies!

D'altra banda no em puc oblidar de la resta de companys del Laboratori. Aquí s'ha de mencionar molta gent. No us puc posar a tots, així que només anomeno als qui, d'alguna manera, han suportat el meu caràcter, a vegades massa analític, a vegades massa impulsiu i contradictori. Gràcies doncs Dani Carbonell, Oscar Càmara, Joan Farnós, Alex, Joan Calafell, Lluís Jofre, Aleix, Santi i Jorge Chiva. En Giorgos, l'Oussama i la Roser s'han salvat dels cafès a la Campana però m'han ajudat molt a la recta final de la tesi, gràcies. Igualment amb el Nico Valle, l'Arnau Pont i el Jesús Ruano, pel suport amb les eines de CAD i les seves aportacions enriquidores. Per cert, he fet un script de Bash per generar automàticament els agraïments al Guillem Colomer. Vaig pensar que seria necessari per a la majoria de tesis dels pròxims 30 anys, està penjat al meu cloud. Gràcies Guillem per ensenyar-nos a treballar amb un editor de debò com el *Vim*, a visualitzar les dades amb bon gust i a escriure documents amb LaTeX. Gràcies també a l'eficient equip informàtic que tenim, heu estat sempre allà quan ha calgut algun mòdul actualitzat de *Python* o per solucionar "freakades" amb els terminals.

I would like to send my thanks to Professor Garimella for welcoming me in the

Purdue University during the summer of 2010. Thanks also to Chris Bates, for your pleasant company during my stay in the USA.

Finalment, voldria donar les gràcies a la meva família i amics pel seu suport incondicional. Especialment al meu pare, que s'ha llegit tota la tesi, al Xavier Pla per dibuixar-me un compressor sencer amb CAD, a la Cristina Cucalón que m'ha donat suport amb la redacció en anglès i sobretot a tu Pili per estar al meu costat cada dia.

Contents

Abstract

The present and the future expectation in parallel computing pose a new generational change in simulation and computing. Modern High Performance Computing (HPC) facilities have high computational power in terms of operations per second -today peta-FLOPS (10^{15} FLOPS) and growing toward the exascale (10^{18} FLOPS) which is expected in few years-. This opens the way for using simulation tools in a wide range of new engineering and scientific applications. For example, CFD&HT codes will be effectively used in the design phase of industrial devices, obtaining valuable information with reasonable time expenses. However, the use of the emerging computer architectures is subjected to enhancements and innovation in software design patterns. So far, powerful codes for individually studying heat and mass transfer phenomena at multiple levels of modeling are available. However, there is no way to combine them for resolving complex coupled problems. In the current context, this PhD thesis presents the development of parallel methodologies, and its implementation as an object-oriented software platform, for the simulation of multiphysics systems. By means of this new software platform, called NEST, the distinct codes can now be integrated into single simulation tools for specific applications of social and industrial interest. This is done in an intuitive and simple way so that the researchers do not have to bother either on the coexistence of several codes at the same time neither on how they interact to each other. The coupling of the involved components is controlled from a low level code layer, which is transparent to the users. This contributes with appealing benefits on software projects management first and on the flexibility and features of the simulations, later. In sum, the presented approaches pose a new paradigm in the production of physics simulation programs. Although the thesis pursues general purpose applications, special emphasis is placed on the simulation of thermal systems, in particular on buildings energy assessment and on hermetic reciprocating compressors.

Chapter 1

Introduction

1.1 Current context and motivations

This thesis has been developed within the framework of the collaboration between Termo Fluids S.L. and the Heat and Mass Transfer Technological Center (CTTC) of the Technical University of Catalonia (UPC). The CTTC is dedicated to academic, research, innovation and technology transfer in the heat and mass transfer and fluid dynamic field, together with their application to thermal systems and equipments. Termo Fluids S.L. is a spin-off born 9 years ago from members of the CTTC that is owner of NEST and TermoFluids source code. This company is devoted to the development of new numerical methods and software tools for High Performance Computing (HPC) in CFD&HT[1] and to the optimization of thermal systems and their equipment.

From this collaborative work emerged a large set of numerical tools for the resolution of physical phenomena such as turbulence, heat transfer, multi-phase flow, etc. Among them, it can be highlighted: CFD&HT-DNS/LES [1–5], mechanical CSD solvers [6], all of them with unstructured meshes, RTE solvers using DOM or FVM angular discretization [7–9], Montecarlo ray tracing tools, and also a set of semi-empirical simplified models for simulating fluid dynamics and heat transfer [10–15].

Today, these computational tools, as well as the inherent knowledge, must be exploited in the assessment of problems where multiple physical phenomena are involved simultaneously. Being able to combine them is essential for addressing the interests of society and industry. Real-world problems should not be analyzed as several uncon-

[1] Computational Fluid Dynamics and Heat Transfer

nected phenomena. Rather, these phenomena should be handled as a whole since they are strongly coupled.

Towards multiphysics

The present and the future expectation in parallel computing pose a new generational change in simulation and computing. Modern HPC facilities have high computational power in terms of operations per second -today peta-FLOPS and growing toward the exascale which is expected in few years-. This opens the way for using simulation tools in a wide range of new engineering and scientific applications [16]. For example, CFD&HT codes will be effectively used in the design phase of industrial devices, obtaining valuable information with reasonable time expenses. However, the use of the emerging computer architectures is subjected to enhancements and innovation in software design patterns [17] (i.e. algorithm strategy, parallel application execution, task synchronization, data structures, etc.).

So far, powerful codes for individually studying heat and mass transfer phenomena at multiple levels of modeling are available. However, there is no way to combine them for resolving complex coupled problems. In the current context, this PhD thesis presents the development of a parallel object-oriented software platform for the simulation of multiphysics systems. By means of this new software platform, called NEST, the distinct codes can now be integrated into single simulation tools for specific applications. This is done in an intuitive and simple way so that the researchers do not have to bother either on the coexistence of several codes at the same time neither on how they interact to each other. The coupling of the involved components is controlled from a low level code layer, which is transparent to the users.

This software platform has already been used in several industrial projects. To name but a few examples, NEST was used to assess the thermal performance inside a windmill nacelle [18], to analyze the thermal energy storage (TES) in tanks [19, 20], to simulate a domestic refrigerator [21], to study thermocline-like TES systems [22], to find novel solutions for improving the performance of reciprocating compressors [23] or to improve control and systems for attaining energy efficiency in buildings [24, 25], too. In addition, other applied research works have also been addressed using NEST. For example, in [26], a solar collector was tested for assessing a novel low-cost flat plate collector with plastic transparent insulation materials. In [27], a complete vapor compression refrigeration system was simulated in a modular way. And finally, in [28], NEST was used for coupling three-dimensional conjugate heat transfer simulations for analyzing fin and tube heat exchangers.

Similarly to NEST, there are several software platforms that pursue the coupling of different codes for simulating multiphysics problems. Some of them are listed in Table 1.1.

Soft./cite	**Institution**	**Description**
OpenPalm [29]	CERFACS (France)	Coupler for the concurrent execution and the intercommunication of programs. Open source under LGPL v3.
MpCCI [30]	Fraunhofer SCAI (Germany)	Application independent interface for the coupling of different simulation codes. Commercial code.
OpenMDAO [31]	NASA Glenn Research Center (USA)	Multidisciplinary Design Analysis and Optimization (MDAO) framework, written in Python. Open source Apache V. 2.0 license.
MOOSE [32]	Idaho National Laboratory (USA)	Multiphysics Object-Oriented Simulation Environment (MOOSE) for finite elment methods. Open source LGPL V2.1.
SALOME [33]	CEA - EDF CASCADE (France)	Open-source software that provides a generic platform for Pre- and Post-Processing for numerical simulation. Open source LGPL.
LIME [34]	Sandia National Laboratories (USA)	Lightweight Integrating Multiphysics Environment for coupling codes. Open source.

Table 1.1: Some software similar to the NEST platform.

Methodology

The NEST platform is designed on the basis of the Object Oriented Paradigm (OOP). This is a key point in the feasibility, usability and sustainability of the code developed in this software project. The OOP programming languages provide a high level of abstraction for modeling real-world problems. In NEST simulations, the different phenomena involved in a multiphysics problem can be represented as objects. The objects are able to perform operations into their interior, which alter their state and hence, also their boundaries. As happens in reality, the objects are in contact to each other and therefore they are phenomenologically coupled. The set of objects conforms a system, which in turn is an object itself. As such, the system performs the necessary operations on its components for attaining an integral simulation.

The entire platform is written using the C++ programming language. The use of a unique language makes the code more readable and encourages reusability, which helps significantly in its maintenance and debugging. This also makes easier the execution of the several codes in parallel, which is essential for speeding up the simulations. Moreover, the data structures of the distinct integrated codes are fully compatible so they can exchange data easily. In any case, the use of other languages is also possible. In the context of a recent research project, an external Python based software has been embedded into a NEST program for simulating the thermal performance of a house [25]. The minimum requirement to couple a code into NEST is the ability to participate in fixed-point iterations.

1.2 Objectives

The aim of this thesis is to develop a software platform on which different modeling codes (semi-empirical 1D, 3D, compressible, turbulence, radiation, etc.) can be combined in order to produce computer simulation programs for solving multiphysics systems.

The platform should be able to combine both existing and new codes. Such new codes may address physical phenomena that is not planned today. For this reason, NEST must be designed on the basis of abstract concepts such as system, system element, linkage, data exchange etc. This helps to reuse code and consequently to facilitate its maintenance and its extension so new applications can be produced in a short time.

Furthermore, the platform must be intuitive and transparent to the users. It should hide all the complexities derived from the coexistence of several models. The model developers should be able to only focus on their field of expertise and they should not bother on the coupling details.

Moreover, since the parallel execution of the simulation codes is essential for obtaining results in a reasonable time, the platform must permit to execute all the integrated codes simultaneously each into several parallel processes.

It is also an objective the development and assessment of tools for transferring discrete data between non-matching meshes. This is necessary for coupling 3D-solvers of different nature, since they often use separate domain discretization (e.g. conjugate heat transfer or fluid structure interaction). These tools must preserve the parallel execution capabilities of the involved codes.

Finally, the thesis also seeks the improvement of several former simulation tools that were developed by the CTTC and the Termo Fluids researchers. In particular, these tools are aimed to model building systems and hermetic reciprocating compressors. Although these are examples of multiphysics environments, the available codes need to be reprogrammed from the scratch to take the advantages derived from using the new paradigm defined by the NEST platform.

1.3 Structure of this thesis

The thesis is structured in six chapters, this introduction and five more chapters as follows.

In Chapter 2, the software development approach followed for constructing the NEST platform is described. Special emphasis is placed on the abstraction of systems, which is considered necessary to create a good software design. The second half of the chapter addresses the main characteristics of this design.

In Chapter 3 the data transfer problem associated to the coupling of independent modeling codes is defined. Several well-known methodologies for exchanging data between non-matching meshes are explained. This is followed by an assessment of these methods in the context of the NEST simulations. Special attention is given to the Radial Basis Function (RBF) and Weighted-Residual formulations.

The remain three chapters of the thesis are aimed to describe and show the possibilities offered by the NEST platform. First, in Chapter 4, a program to simulate the thermal performance of buildings is presented. Two different dwellings are simulated in order to illustrate its usage and the numerical results it can provide. On the other side, a simulation program for predicting the behavior of hermetic reciprocating compressors is presented in Chapter 5. The code is numerically verified and experimentally validated. The potential of this tool is shown by means of an example and several parametric studies focused on the power consumption and the cooling capacity. Although each of these programs uses its own particular models, both applications need several semi-empirical models based on the integration of the fundamental conservation laws of mass, momentum and energy. The code addressing this issue is reused in both the buildings and the compressors applications.

Finally, in Chapter 6 is presented a three-dimensional simulation of the heat transfer in the components of an hermetic reciprocating compressor. In this simulation, the different components of the compressor are solved separately, each with its appropriate discretization. The data transfer tools developed in Chapter 3 are employed here for coupling the several instantiated heat transfer solvers.

References

[1] F.X. Trias et al. Symmetry-preserving discretization of Navier–Stokes equations on collocated unstructured grids. *Journal of Computational Physics*, 258:246–267, 2013.

[2] I. Rodríguez et al. Direct numerical simulation of a NACA0012 in full stall. *International journal of heat and fluid flow*, 43:194–203, 2013.

[3] O. Lehmkuhl et al. Low-frequency unsteadiness in the vortex formation region of a circular cylinder. *Physics of Fluids (1994-present)*, 25(8), 2013.

[4] O. Lehmkuhl et al. On the large-eddy simulations for the flow around aerodynamic profiles using unstructured grids. *Computers and Fluids*, 84:176–189, 2013.

[5] O. Lehmkuhl et al. Unsteady forces on a circular cylinder at critical Reynolds numbers. *Physics of Fluids (1994-present)*, 26(12), 2014.

[6] O. Estruch et al. A parallel radial basis function interpolation method for unstructured dynamic meshes. *Computers and Fluids*, 80:44–54, 2013.

[7] G. Colomer, R. Cònsul, and A. Oliva. Coupled radiation and natural convection: different approaches of the SLW model for a non-gray gas mixture. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 107(1):30–46, 2007.

[8] R. Capdevila, C. Perez, and A. Oliva. Development and comparison of different spatial numerical schemes for the radiative transfer equation resolution using three-dimensional unstructured meshes. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 111(2):264–273, 2010.

[9] G. Colomer, R. Borrell, F. X. Trias, and I. Rodríguez. Parallel algorithms for S_n transport sweeps on unstructured meshes. *Journal of Computational Physics*, 232(1):118–135, 2013.

[10] Perez-Segarra et al. Numerical study of the thermal and fluid dynamic behaviour of reciprocating compressors. In *International Compressor Engineering Conference*, Purdue, IN, USA, July 1994. Purdue e-Pubs. Paper 971.

[11] Perez-Segarra et al. Detailed thermodynamic characterization of hermetic reciprocating compressors. *International Journal of Refrigeration*, 28:579–593, 2004.

[12] J. Rigola. *Numerical simulation and experimental validation of hermetic reciprocating compressors. Integration in vapour compression refrigerating systems*. PhD thesis, Universitat Polotèctinca de Catalunya, 2002.

[13] J. Rigola et al. Parametric studies on hermetic reciprocating compressors. *International Journal of Refrigeration*, 28(2):253–266, 2005.

[14] J. Rigola et al. Numerical simulation and experimental validation of vapour compression refrigeration systems. special emphasis on co2 trans-critical cycles. *International Journal of Refrigeration*, 28(8):1225–1237, 2005.

[15] R. Damle. Object-oriented simulation of reciprocating compressors: Numerical verification and experimental comparison. *International Journal of Refrigeration*, 34(8):1989–1998, 2011.

[16] D.E. Keynes et al. Multiphysics simulations: Challenges and opportunities. *International Journal of High Performance Computing Applications*, pages 4–83, 2013.

[17] A. Bishop and P. Messina. Scientific grand challenges in national security: The role of computing at the extreme scale. Technical report, 2009. Sponsored by the U.S. Department of Energy, Office of National Nuclear Security Administration, and the Office of Advanced Scientific Computing Research.

[18] Analysis of the thermal and fluid-dynamic behaviour of the nacelle and the tower of a wind turbine. Numerical simulation, experimental validation and application to the design of a new wind turbine. http://www.cttc.upc.edu/, 2006-2008. ECOTECNIA. Accessed: 2015-12-07.

[19] In Space Propulsion. http://www.cttc.upc.edu/, 2009-2012. Snecma, Astrium, AVIO, Mikroma, Alcimed, Bonatre. Accessed: 2015-12-07.

[20] New technologies for an efficient, ecologic and intelligent washing system for textiles of the future. http://www.cttc.upc.edu/, 2010-2012. Fagor Electrodomésticos. Accessed: 2015-12-07.

[21] Modular domestic refrigeration systems with high energy efficiency (KERS). http://www.cttc.upc.edu/, 2010-2013. Fagor Electrodomésticos. Accessed: 2015-12-07.

[22] Thermal storage for concentrating solar power plants. http://www.cttc.upc.edu/, 2011-2014. EIT-KIC InnoEnergy. Accessed: 2015-12-07.

[23] Technology development cooperation agreement for LC series. http://www.cttc.upc.edu/, 2014-2015. UPC-DONPER Co. Ltd. Accessed: 2015-12-07.

[24] Retrofitting Solutions and Services for the enhancement of Energy Efficiency in Public Edification (RESSEEPE). http://www.resseepe-project.eu, 2013-2015. European project. Accessed: 2015-12-07.

[25] Dwelling Climate Control System (DCCS). http://www.kic-innoenergy.com/innovationproject/our-innovation-projects/dwelling-climate-control-system-dccs, 2013-2016. EIT-KIC InnoEnergy. Accessed: 2015-12-07.

[26] H. Kessentini, O. Lehmkuhl, R. Capdevila, J. Castro, and A. Oliva. Development of flat plate collector with plastic transparent insulation and low-cost overheating protection system. *Applied energy*, 133:206–223, 2014.

[27] N. Ablanque et al. Modular simulation of vapour compression systems with an object oriented tool. In *International Compressor Engineering Conference*, Purdue, IN, USA, 2012. Purdue e-Pubs. Paper 1349.

[28] L. Paniagua. *Three-dimensional numerical simulation of fluid flow and heat transfer in fin-and-tube heat exchangers at different flow regimes*. PhD thesis, Universitat Polotèctinca de Catalunya.

[29] OpenPalm. http://www.cerfacs.fr/globc/PALM_WEB. CERFACS (France). Accessed: 2015-12-07.

[30] MpCCI. http://www.mpcci.de. Fraunhofer SCAI (Germany). Accessed: 2015-12-07.

[31] OpenMDAO. http://openmdao.org. NASA Glenn Research Center (USA). Accessed: 2015-12-07.

[32] MOOSE. http://mooseframework.org. Idaho National Laboratory (USA). Accessed: 2015-12-07.

[33] SALOME. http://www.salome-platform.org. CEA-EDF-OpenCASCADE (France). Accessed: 2015-12-07.

[34] LIME. http://sourceforge.net/projects/lime1. Sandia National Laboratory (USA). Accessed: 2015-12-07.

Chapter 2

Abstraction of multiphysics systems

2.1 Introduction

The complexity of software projects grows non-linearly with its size. To avoid undesirable problems such as product flaws, cost overruns or schedule delays, this growth should be controlled. For doing this it is required to have a good perspective of the complete project, understand all its parts and keep the conceptual integrity over time. Inappropriate new features or "improvements" may spoil the comprehensibility of the code [1]. Moreover, conceptual errors are difficult to repair. They are normally detected too late and involve *ad hoc* solutions at important parts of the code. This increases code entropy and consequently its maintenance gets more difficult [2].

The multiphysics framework presented in this thesis is a large software project. Several researchers are involved in it, either as developers or as users. Since this is a long term project, others who will join later will have to be trained. In that sense, facilitate code maintenance, preserving clarity, is essential to prevent the project to be discontinued. Moreover, a wide versatility is expected also. The code is susceptible to evolve as new applications are faced. Therefore, the aforementioned conceptual matters should be taken into account every time the code is modified. Obviously, covering more applications can not compromise those already work.

The contents of this chapter are aimed to explain the development approach followed in the fulfillment of this software project. Special emphasis is given to the abstraction of

[1] *I will contend that conceptual integrity is the most important consideration in system design. It is better to have a system omitting certain anomalous features and improvements, ... than having one that contains many good but independent and uncoordinated ideas.* [1]

systems. This is considered necessary to create a good software design. Further on, the main characteristics of this design are also discussed. Finally, it is provided an example that shows how the multiphysics framework is used to create a simulation tool for a particular application.

2.2 The abstraction process

The comprehension process where the concepts and rules of a particular problem are defined is known as abstraction. The abstraction process is based on the study of specific examples and on the previous experience, too. Analyzing a single situation allows identifying certain entities that make sense within the logic of the studied problem. Nevertheless, this first abstraction is likely too superficial to design a flexible program. Thinking on similar past situations or observing other cases highlights common characteristics, functionalities, phenomena, etc. At that point, code design can be expressed in terms of more abstract concepts which raises program versatility.

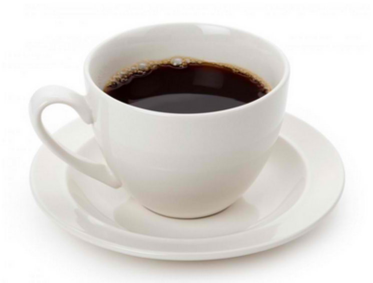


Figure 2.1: This is just a cup of coffee.

As an illustration, look at the picture in Figure 2.1. At first sight one identifies a *Cup of Coffee*. The visual perception of such concept is immediate as it is a quotidian object. However, further observation of the picture in Figure 2.2 increases our abstraction capacity. By comparison of both images you are able to see beyond the obvious and think on other more abstract concepts that are not so evident. At this time it is understood that both pictures represent a *Consumable* served on a different *Container*. Hence, beside the concepts of *Cup of Coffee* and *Gazpacho*, two more concepts are identified.

This abstraction would be useful in the context of a software for restaurants, to manage coffee orders and crockery usage, for example. Furthermore, it is obvious that these *Consumables* will be ordered by somebody so a new *Customer* entity would also be considered. This customer will sit at a *Table* where new *Orders* will be created. This comprehensive process could continue indefinitely. As our reasoning is being conditioned by new information, old experiences, the environment observation or the problem specifications, the mental model becomes more abstract. Finally, this leads to a set of intuitive terms and actions -a natural language- that allows us to create a good design including the essential parts of the attempted problem (*Consumable*, *Container*, etc.). Moreover, this design can be easily adapted and extended to consider new concepts that make sense in the context of the application (*Customer*, *Table*, *order*, *Payment*, etc.).

Figure 2.2: This is a gazpatcho andaluz.

The creation of abstractions has been a widely used resource among software researchers. Around 1960, early programming languages such as Assembly and FORTRAN overcame practical issues of using machine-code. Similarly, the first operating system platforms, such as OS/360 and Unix, protected developers from programming directly to hardware [3]. That level of abstraction allowed the development of short applications. Even though, as computer projects grew they became more complex and consequently more unmanageable [1]. Later, at the end of the 1960s, programming languages evolved towards the object oriented paradigm (OOP). The new languages enabled expressing code designs in terms of a more natural language, closer to human reasoning than to computing logics.

The multiphysics framework presented in this thesis is developed in the C++ language [4]. This is a successor of the C programming language [5] that introduces new tools for OOP design. In particular C++ provides flexible and efficient facilities for defining *classes*. Classes are programmer-created types that match the fundamental concepts of the application. Basically, class types encapsulate concept specific information and define an interface to access or modify that information. This is an important programming tool for representing the ideas of the design into the code. Furthermore, a very powerful code property provided by object oriented languages is polymorphism. As already stated by its name, this property permits the code concepts to have multiple shapes. This enables multiple levels of abstraction which contributes to code reuse. In order to illustrate this idea the code of a class hierarchy is shown below.

```cpp
class GeoShape() {                              // New type declaration.
   double area;                                 // Class members common to all
   double perimeter;                            // geometry objects.
   virtual double evalArea() const = 0;         // Abstract (open) interface
   virtual double evalPerimeter() const = 0;    // to evaluate class members.
};
```

```cpp
class Circle() : public GeoShape {       // New type extending GeoShape.
   double radius;                        // Local parameter.
   void setRadius(double r) {            // Local interface.
     radius = r;
   }
   double evalArea() const {             // Base interface specification.
     return pi*radius*radius;
   }
   double evalPerimeter() const {        // Base interface specification.
     return pi*2*radius;
   }
};
```

```cpp
class Square() : public GeoShape {       // New type extending GeoShape.
   double side;                          // Local parameter.
   void setSide(double l) {              // Local interface.
     side = l;
   }
   double evalArea() const {             // Base interface specification.
     return side*side;
   }
   double evalPerimeter() const {        // Base interface specification.
```

```
        return 4*side;
    }
};
```

2.2.1 A note on software development

The development stages of a software project are depicted in the conceptual model in Figure 2.3. Adopt each of the stages in this model is material to succeed in this multiphysics framework project. There are two reasons to think about a life-cycle. On the one hand the code needs a maintenance. All modifications a software usually needs - bug-fix, enhancement of capabilities, deletion of obsolete capabilities and improving performance - entail an iterative procedure. On the other hand, needs and conditions are never fully gathered at the first requirements analysis. This is particularly true in the context of this project due to the generality involved with multiphysics problems.

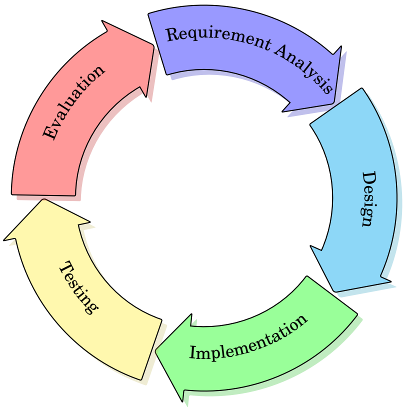


Figure 2.3: Stages of the software development life-cycle, from requirement analysis to evaluation.

The working methodology employed in the development of this software combines the introduction of new features and code refactoring[2]. The use of an OOP language is very useful in that sense since it provides tools such as the inheritance and polymorphism, which allow extending the code easily on the basis of an abstract design. It is also

[2] Changing the code without altering its behavior to improve style consistency and clarity [6]

important to accomplish the simplicity principle, known as YAGNI[3] or more generally the KISS principle[4]. These principles establish that programs must be as simple as possible and discourages the programmer to anticipate code declarations that are not strictly necessary today.

Besides the previous considerations, the programmers working on the development of this code, use a decentralized revision control systems for safe code integration, in particular the Bazaar software [9]. In this way the different researchers merge their work often into a mainline. Doing so it is necessary to keep the team in sync so one does not bother on the others changes. This is also supported by a test suite that rejects committing new code to the mainline if they are not fulfilled, which enables safety again.

In short, the development approach of this multiphysics framework project is carefully defined to achieve the main goals stated in the introductory chapter of the thesis. Obviously such methodologies are not new. Still, it is worth mentioning that not using them leads the project to the disaster as many software researchers warn [1–3].

2.3 The NEST concept

As it has been commented in the introductory chapter of the thesis, the NEST platform is aimed to fast and easy production of multiphysics simulation programs. Multiphysics involve using several levels of modeling and/or several phenomena happening simultaneously. Trying this is challenging and it requires a hard coding effort to make it feasible. Several researchers have been working on the development of this software during the last decade. Actually, the software platform consists of thousands upon thousands of code lines. Fortunately, the code is classified in several libraries so management is assigned to different work teams.

Each of these libraries implements a different level of abstraction, ranging from very abstract concepts (e.g. geometric, mathematical or algorithmic entities) to the most specific objects (e.g. components of particular technology devices). The NEST concept should be understood as the linkage of such software libraries which enable multiphysics together. In Figure 2.4 it is presented a general outlook of the software and the relationship between each library. Every block in the diagram is also explained in the subsequent review boxes.

[3] You Ain't Gonna Need It [7]

[4] Keep It Simple, Stupid [8]

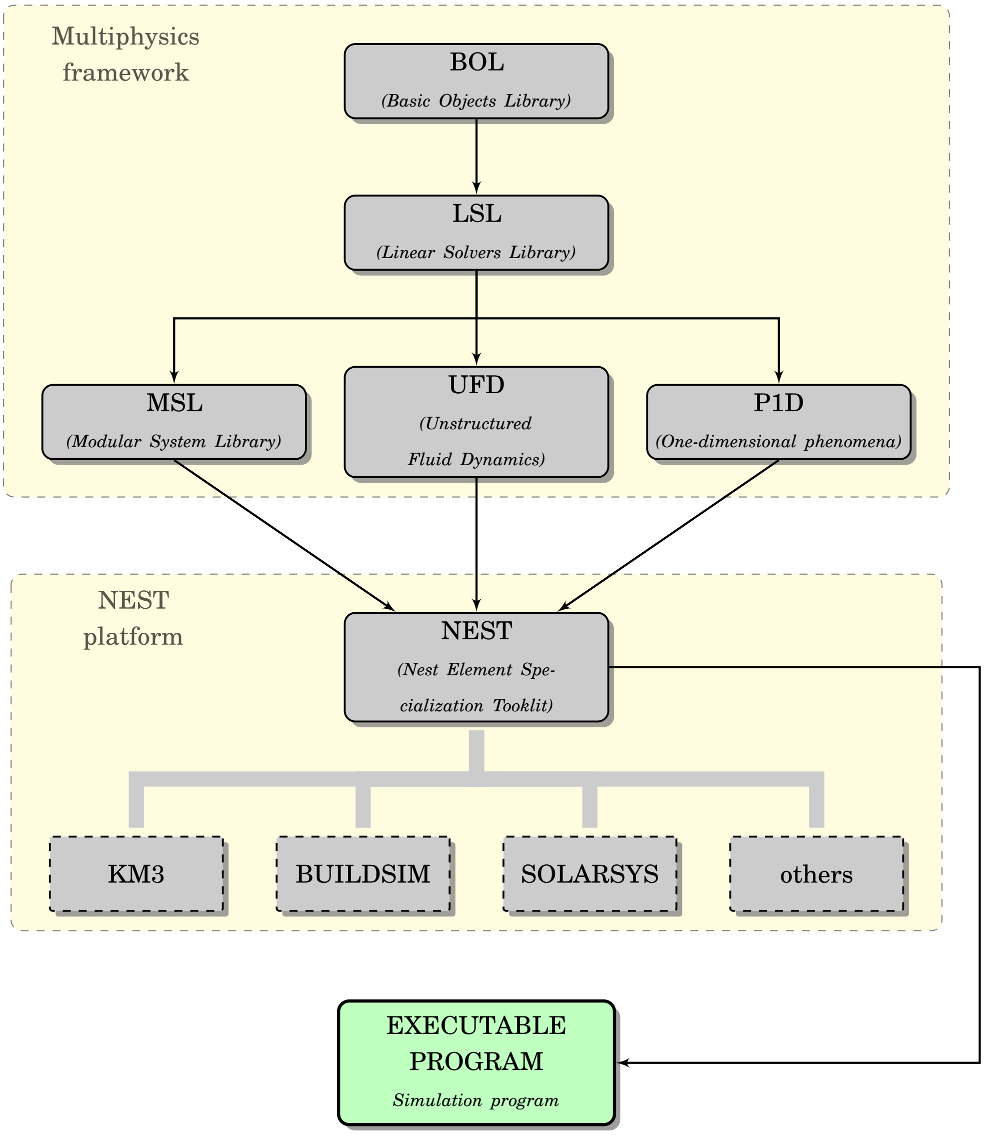


Figure 2.4: General outlook of the NEST-Multiphysics framework. The libraries are ordered from the top down, from bigger to lower abstraction.

Basic Objects Library (BOL)

Abstraction: geometric, algebraic, parallel data structures, i/o handling

Implements an object-oriented parallel unstructured finite volume API aiming good performance in the different existing parallel computational facilities [10].

Linear Solvers Library (LSL)

Abstraction: data structure, algorithmic (solver, preconditioner)

Algebraic library composed of different general purpose and application-specific algorithms to solve linear systems of equations [10].

Unstructured Fluid Dynamics (UFD)

Abstraction: boundary conditions, equations and their terms, numerical algorithms and several data structures (physical properties, unknowns, etc.).

Implements numerical models for resolving the governing equations of fluid and/or solid mechanics and heat transfer enabling several physical phenomena (single phase, multiphase, turbulence, transport of species, chemical reactions, etc.) [11]. It does also implement the algorithms for coupling the transport of different physical quantities (mass, momentum, energy, etc.) as well as the algorithms for time integration.

One-dimensional Phenomena (P1D)

Abstraction: 1D-geometry, empirical correlations, boundary conditions, equations and their terms, algorithms and several data structures (physical properties, unknowns, etc.).	Abstraction of one-dimensionally reduced numerical models. The library is mainly addressed to code reuse in applications where the global physical behavior is analyzed. It's design is similar to the UFD library. In this instance however, the code is optimized for one-dimensional phenomena in particular, which enables very fast simulations.

Modular System Library (MSL)

Abstraction: system, element, graph iterators, linkage, communication protocol, file parser, data management, system algorithms	Abstraction of multiphysics problems seen as systems and their components. Implements several classes to combine and operate system components as single abstract entities through a common interface. Enables partitioned approaches.

NEST main repository

Abstraction: pipe-flow, tank-storage, wall-through phenomena, date management.	Extends the MSL classes to define new types. Enables simulation of diverse applications involving fluid flow and heat transfer phenomena. It does not only provide types with physics entity but it also provides other concepts such as calendar tools for managing human and/or meteorological events.

Compressor repository (KM3)

Abstraction: compressor device and its components

Interface to define the several components that conform a compressor device (pipes, mufflers, resonator chambers, compression chambers, valve orifices and reeds, etc.) [12, 13].

Buildings repository (BUILDSIM)

Abstraction: industrial spaces and dwelling houses

Interface to define components that comprise part of industrial spaces and dwelling houses (rooms, envelops, doors, windows, HVAC devices, etc.) [14–16].

Solar-energy repository (SOLARSYS)

Abstraction: solar energy technology components

Interface to define components that comprise part of solar systems (panes, collector, absorber, etc.) [17, 18].

2.3.1 The Modular System Library

The development of the Modular System Library (MSL) has been the main focus on this thesis work. Its well-thought-out design is part of a long-term strategy to guarantee success in the multiphysics software goals. As shown in Figure 2.4 the library is constructed over other libraries that define distinct abstraction layers: the Basics Objects Library [10], the Linear Solvers Library [10], the Unstructured Fluid Dynamics library [11] and the One-dimensional Phenomena library. On the other side, the software in the NEST platform puts into practice the MSL approach, implementing new types for simulation of particular phenomena.

The MSL sets the fundamental logics in system environments and establishes a natural language for its management. In other words, it provides abstraction at system level. This library establishes an approach based on the extension of several basic predefined code entities to create new concepts. The most relevant basic concepts implemented into the MSL library are *Buffer*, *Element* and *System*. These concepts are represented using C++ classes since they are identified as important parts of the code with particular entity. The *System* and the *Element* classes are related to each other by composition, meaning that the life-cycle of the *Element* instances depends on the existence of the *System* they are contained in. In a similar way, the *System* class also contains a set of *Buffer* instances for storing the input and output data of the *Element* objects. The class diagram of this design is shown in Figure 2.5, which is further analyzed in the remainder of this section.

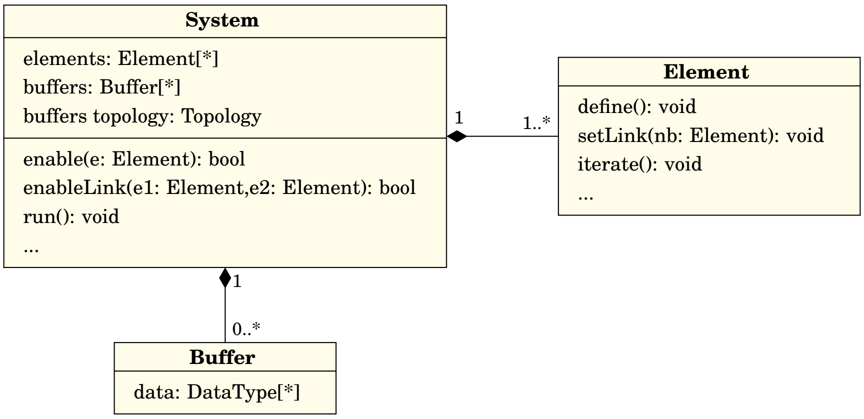


Figure 2.5: UML diagram of the *System*, *Element* and *Buffer* classes. The *System* class contains one or more *Element* objects and zero or various *Buffer* objects.

The System class The *System* class is designed to create, connect, operate and remove the different elements that compose a system. It also implements communication protocols to let the linked elements to exchange state information. This approach permits writing coupling algorithms and time integration schemes to solve complex problems by partitioning. The *System* class also enables the display of information to follow the progress of the simulation and it provides tools for safe and efficient storage of the output data into files for post-processing. This is done in a flexible way using abstract types specifically created to that end, such as the *Element* and the *Buffer* classes or iterators tools. Iterators are widely used along the code of the *System* class for visiting elements into a set of special interest.

For the sake of example, you may consider the Algorithms 1 and 2. The former is the setup procedure of the *System* class. It consists on loading the system definition from an input file (i.e. the elements to create and its linkage) and then perform some initializations or actions that are required only once at the beginning of the simulation.

Algorithm 1 *System* setup summarized in few lines.

```
$file = parseInputFile("case.in")$
for (each $type$ in $file$) do                         ▷ loading every element in the system
    if $type$ is known then
        $e = createNewElement(type)$
        $e \rightarrow define(file)$
    else
        $display("unknown\ type\ ERROR")$
for (each $link : e_l - e_r$ in $file$) do                     ▷ loading linkage definition
    if $e_l$ and $e_r$ are known and $link$ is enabled then
        $e_l \rightarrow setLink(file, e_r)$
        $e_r \rightarrow setLink(file, e_l)$
    else
        $display("unable\ to\ link\ ERROR")$
$out = createOutputFile()$                                    ▷ prepare the input file
for (each element $e$) do
    $out \rightarrow setFormat(e)$
    for (each neighbour $nb$) do                   ▷ first time the elements exchange data
        $nb \rightarrow inputs(e) = e \rightarrow outputs(nb)$
```

The second algorithm is based on a block-Jacobi scheme. This procedure is aimed to resolve a general problem that has been split into several parts (elements in the current context). Note how all actions are performed over elements of the same type. Moreover, these elements do not have any particular nature at all, as they are operated through their interface only (e.g. $e \rightarrow define()$, $e \rightarrow iterate()$, and so on). The helpfulness of the iterators is shown in the three for-loop statements. The elements are visited one by one, first to execute their particular resolution routines and later to perform post-resolution actions. Note in particular that in the inner for-loop only the subset of neighbor elements is iterated. This illustrates how iterators allow writing code in a natural way closer to human reasoning.

Algorithm 2 *System* block-Jacobi method summarized in few lines.

```
$r = \epsilon = 1e-10$
while $r >= \epsilon$ do
    for (each element $e$) do
        $e \rightarrow iterate()$
        $e \rightarrow setOutputs()$
    for (each element $e$) do
        $r = max(r, e \rightarrow evaluateResidual())$
        for (each neighbour $nb$) do
            $nb \rightarrow inputs(e) = e \rightarrow outputs(nb)$
                                        ▷ go to the next iteration or start next time step
```

The Element class The element concept derives from the need to manipulate, operate and connect the several system entities in a general way. Since the MSL is a very versatile library intended to different unconnected applications, no information about the specific details of the system components is expected to be provided at this level. Instead, only the common behavior of these components is stated. This means that the *Element* class does not set a specific state in itself, namely its code must be extended in order to let the elements have its own properties. Consequently, this class only implements a set of void methods which constitute its interface. The polymorphic character of the system elements relies on the inheritance mechanism provided by the object oriented languages. When the *Element* class is inherited, its interface can be overwritten in order to define a specific behavior. Hence, from each inheritance arises a new specific instantiable type.

Within the interface of the *Element* class it is worth to explain the methods shown in Figure 2.5. To be meaningful, a new type should at least overwrite the *define* and *iterate* methods. The former allows initializing the class parameters and it is executed during *System* setup, as shown in Algorithm 1. On the other hand, the *iterate* method implements self-resolution approaches based on its internal logics. When the elements are interconnected, this method is recursively executed to take into account the evolution of the remain elements in the system, as illustrated in Algorithm 2. This maintains the properties of the element up to date with its surroundings so the system gets well-coupled.

> The *System* class uses the *Element* interface to "say" when the elements must perform actions. Extended element types implement specific definitions of these actions so they can be combined to create systems of particular nature.

For a better understanding, an extension of the *Element* class is illustrated through the creation of an *Adder* class, which is depicted in the diagram of Figure 2.6. This is followed by the Algorithms 3 and 4, which show how the *Element* interface is overwritten.

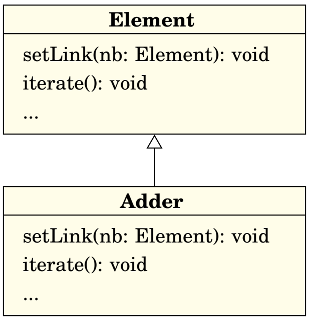


Figure 2.6: *Adder* is an inheritance of *Element*. It overrides the *Element* interface to implement its own behavior.

Algorithm 3 The *Element* class interface.

```
function define(file: InputFile)        ▷ void definition by default
function setLink(nb: Element)           ▷ void definition by default
function iterate                        ▷ void definition by default
```

Algorithm 4 The *Adder* class overrides the *Element* interface.

```
function define(file: InputFile)
    $v_{ini} = file \rightarrow load(v_{ini})$

function setLink(nb: Element)
    if nb is sumable then
        set $v_{nb}$ as input
        set $r_{sum}$ as output

function iterate
    $r_{sum} = v_{ini}$
    for (each recieved value) do
        $r_{sum} += v_{nb}$
```

Buffer class When two elements are linked they must be able to exchange information to each other. The *setLink* method, in Figure 2.5, is aimed to specify what data an element type exchanges when it is combined with the same or other types. On that point, the *Buffer* class plays an essential role. This class is a standard type for sending and receiving data between the system elements. This is done in a consistent way regardless the type and size of these data. The buffers may store single datum or large data sets of different built-in types (i.e. doubles, integers, booleans, etc.). However, the elements do not manage *Buffer* instances directly. Instead, this is done by *System* in a centralized way, as shown in the class diagram of Figure 2.7. When *System* executes *setLink*, the appropriate *input*/*output* variables are associated to single *BufferRequest* instances. These requests are reported to *System* so it is able to create all the necessary *Buffer* instances later. The set of buffers are inserted into a parallel distributed container, available in the BOL library [10].

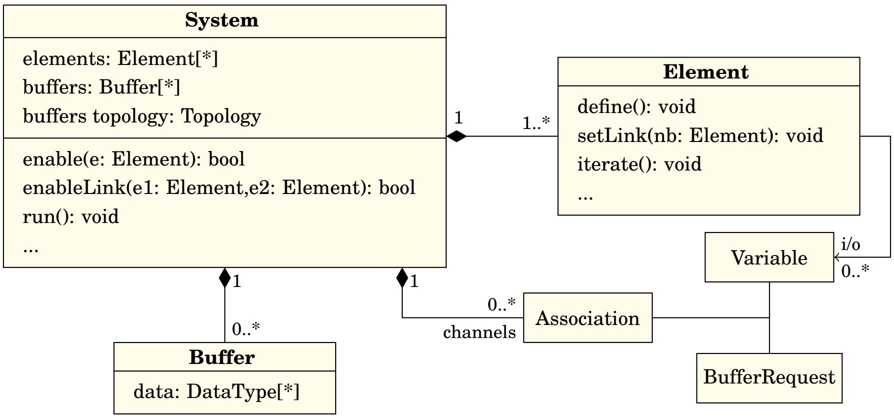


Figure 2.7: Detailed UML diagram of the *System* and *Element* classes.

Parallel Topology The simulation of a multiphysics problem using the NEST platform is staggered into three main steps: the setup, the calculation and the finalization. In normal circumstances, the calculation step is the one that takes more time along the simulation. For that reason, it is essential to speed up this process in order to achieve results in a reasonable time. For instance, you may have a look at Algorithm 2 which entails a while-loop that executes iteratively its instructions until the residual of the system achieves a certain value ϵ. The performance of the calculation step is of special importance when the simulation involve high level models (e.g CFD, RTE or CSD codes, etc.), whose boundary conditions may strongly affect its internal state, or when the number of elements in the system is significantly large, in which case convergence may slow down.

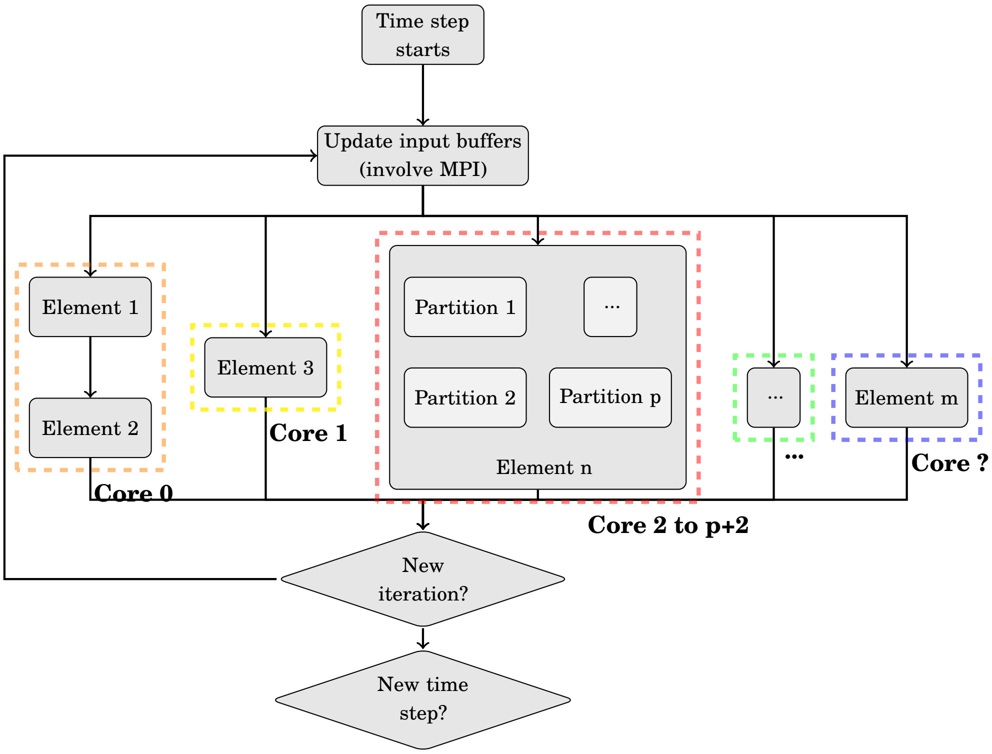


Figure 2.8: Parallel block-Jacobi flow chart.

In that sense, note that the Algorithm 2 can be executed in several processes simultaneously as shown in Figure 2.8. This interesting property enables the resolution of a multiphysics problem in parallel. To get this goal, the *System* class is instantiated several times in parallel processes, each owning a sub-set of the elements conforming the global problem. The distribution of the elements is done by means of a graph partitioning tool called METIS [19]. This tool seeks to balance the computational load between the available processors, while at the same time minimizes the number of edges that are cut in the system graph representation. In Figure 2.9 is illustrated a graph as a set of elements, the vertexes, which are connected to each other through the edges.

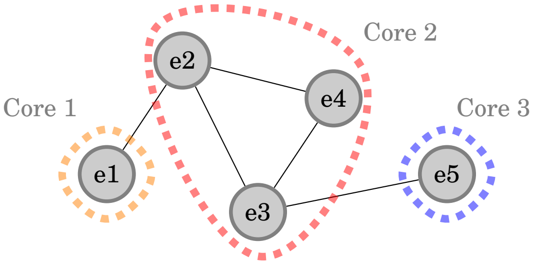


Figure 2.9: Vertex-edge graph representation of a system. Here, the graph is split in three partitions distinguished by color.

The parallelization of the code is implemented at a lower programming level using the concept of Parallel Topology defined in the BOL library [10]. A topology consists of a set of unique identifier indexes that refer to specific objects of the same type. The indexes are inserted into a topology as *Owned* entries, when the pointed object belongs to the current processor, or as *Halo* entries, when that object resides in a foreign processor. *Halo* entries should be seen as copy representations of objects that are *Owned* at other processors. The *Halo* representations are maintained up to date with their corresponding objects by means of synchronization operations based on the Message Passing Interface standard (MPI) [20].

The *System* class owns a parallel topology to deal with broken connections in the system graph. In particular, the topology manages the set of *input*/*output* buffers, which are also distributed among the several processors. By means of the halos approach, owned input buffers can be connected to output buffers that have been instantiated at foreign processors. Therefore, this makes possible to create connections between system elements instantiated at distinct processors and consequently the simulation program can be executed in parallel.

2.4 Example

The main design points of a software tool for the simulation of hermetic reciprocating compressors is discussed below. This is just aimed to illustrate how the NEST-Multiphysics framework is employed to produce a software tool for a specific application. Further details on compressors simulation using NEST is provided further on in Chapter 5.

2.4.1 Abstraction process

The abstraction process of the compressor device is based on the observation of the following characteristics. Some of the classes that were finally implemented may entail several of these items.

- *geometry*. Some regions of the overall computational domain can be differentiated by its geometry. For instance, pipes, expansions or contractions, micro-channels stacks, distributors or manifolds, etc. Partitioning the system at the boundaries of these geometric entities provides flexibility to the simulation tool since one is able to replace, remove or reorder some components easily without affecting the remain parts.

- *physical phenomenology*. Disparate phenomena usually need distinct mathematical modeling so the numerical treatment and the computational requirements are also different. It is not the same solving the fluid dynamics inside a pipe than solving the heat transfer within its walls. Therefore, for practical programming reasons and for performance also, it makes sense to develop separate code modules to implement both mathematical models.

- *level of modeling*. It can make sense to use several levels of modeling at different regions. This refers to the detail of the simulation and how does the mathematical model fit the real phenomena. For instance, you may consider that someone investigate how a slight modification in the geometry of a particular component affects the overall behavior of a system. Capturing such effects requires CFD&HT. However, extending the detailed analysis to the whole domain is usually not feasible, or it is just not necessary. Hence using simplified models (e.g. zero or one-dimensional models) in the remainder regions of the system may be a good idea.

- *device functionality*. All components in a system have some kind of functionality. Sometimes the specific purpose of some device is also a partitioning criterion. Let's consider a refrigeration system with its basic components such as the condenser, the expansion device, the evaporator and the compressor. Since each of these components is itself a complex system it makes sense to see all them as separate parts of the refrigeration unit.

2.4.2 Class diagram of a compressors program

The compressor device is modeled by means of the *Compressor* class, which inherits from the *System* class to implement its specific properties. In addition, several classes that extend the interface of the *Element* class have also been created. Some of these classes implement one-dimensional fluid models such as *Pipe*, *Resonator*, *CompChamber* or *ValveOrifice*. Other classes such as *CompBody* resolves the heat transfer within the compressor components. The class diagram of the compressor code design is presented in Figure 2.11.

Note that most compressor elements are not direct inheritances of the basic *Element* class. This contributes to reuse code since the compressor phenomena (e.g. compressible flow, heat transfer, valve motion, etc.) or some geometric characteristics (e.g. chambers, orifices, channels, etc.) are also present in other applications. For instance, the suction and discharge valves inside a compression chamber are conceptually close to the a door in a house. In this fashion, a code for building simulations could use part of the compressors implementation. Actually, in 1D simulations the valve and the door can be seen as flow constrictions based on the most left sketch of Figure 2.10. Hence, the *Orifice* class implements such flow constriction while the *ValveOrifice* implements a specific model for compressor valves.

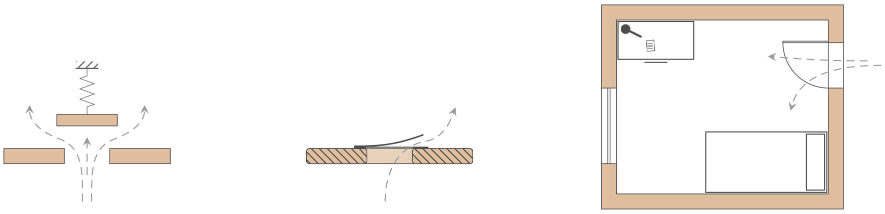


Figure 2.10: One-dimensional model of a flow constriction (left), a compressor valve reed (center) and a room (right).

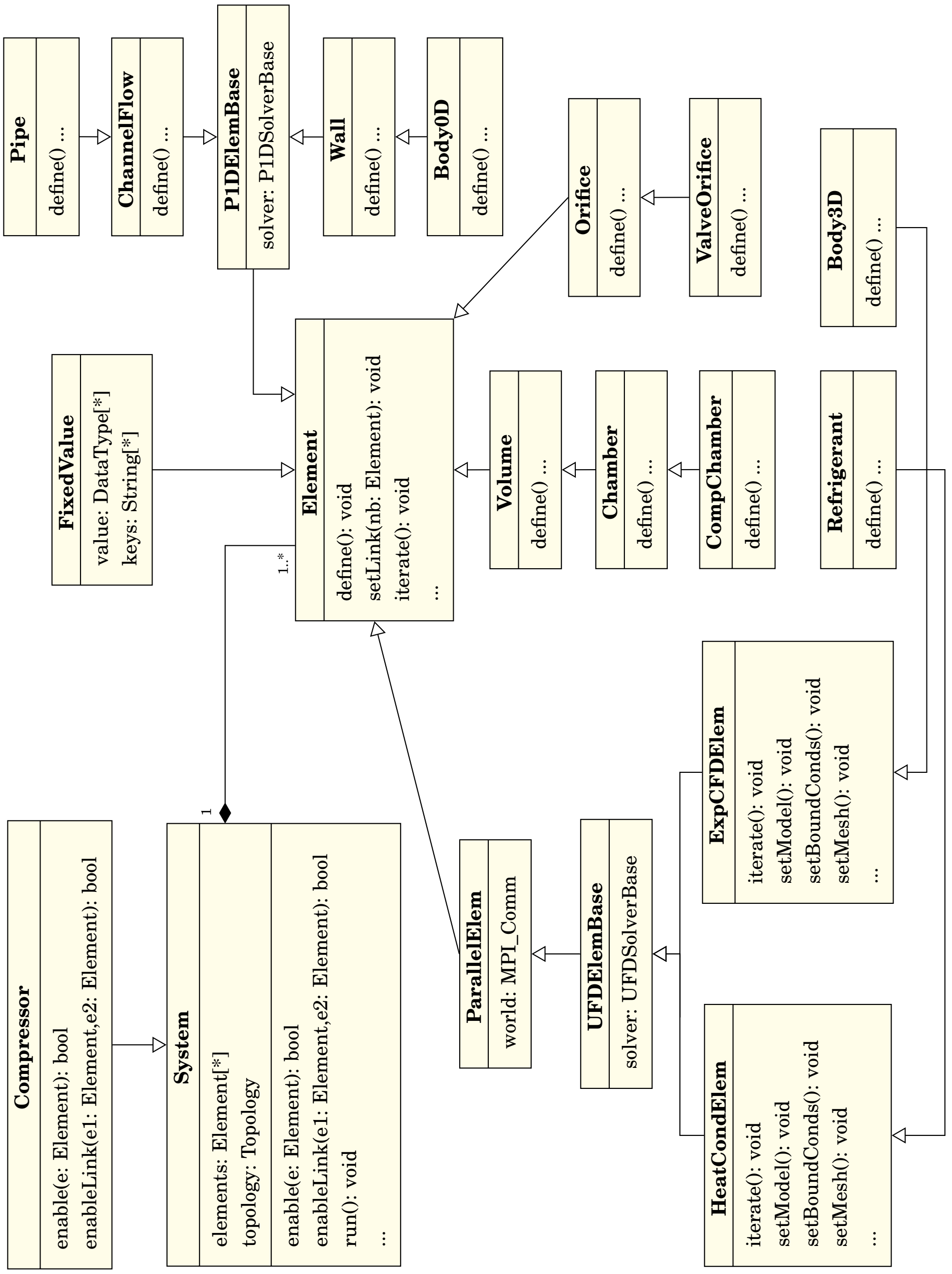


Figure 2.11: Class diagram of a compressor simulation program. For the sake of brevity class methods are not depicted.

Besides the components based on one-dimensional models, the design also offers using multi-dimensional modeling by means of the UFD library [11]. The solvers distributed with this library are based on finite volume methods and employ discretization meshes. Although these solvers work with unstructured meshes, the memory and the computational power requirements increase quickly with the mesh size. Under those conditions parallel computation is essential to achieve results in a reasonable time. From this derives the need for the *ParallelElem* class in Figure 2.12. This class sets an MPI Communicator [20] to manage a specific subset of processors exclusively available to execute the UFD solver. Whereas the *System* class enables the distribution of its elements into several processors (see Section 2.3.1), the elements can also set their own parallel processes. It must be remarked that these are complementary capabilities so multiple levels of parallelization are enabled for resolving the system. For compressors simulation in particular, the UFD solvers have been extended to create the *Body3D* and the *Refrigerant* classes. These classes make possible three-dimensional heat transfer analysis in the solid parts of the compressor and CFD&HT inside its components (e.g. shell, muffler, chambers, pipes, etc.). Some examples are available in Chapter 6 or in the *Appendix A*.

2.4.3 Steps for running a case

Before running a compressors simulation one must create an executable program that instantiates the *System* class and executes its resolution routine. This program looks like the following code.

```
#include<complib.h>
int main(int argc,char **argv) {
   mpi_init(argc,argv,true);       // Init MPI standard.
   Compressor c(argv);             // Instantiate the compressor (construction).
   c.run();                        // Run the system resolution routine.
   mpi_end();
   return 0;
}
```

The first line in the above program includes *complib.h*. This file, attached below, includes the MSL, the NEST and the KM3 sources and contains the definition of the *Compressor* class.

```
#include<msl/System.h>
#include<nest/Objects.h>
```

```
#include<km3/Objects.h>
class Compressor : public msl::System {         // New type extending System.
      string refrigerant;                       // Local inputs.
      double frequency;
      Compressor(char **argv) : System(argv) {
         refrigerant="undefined";               // Default setup, user values
         frequency=0;                           // are set at runtimie.
      }
      virtual void enableElems() {              // Base interface specification.
         enable(nest::FixedValue);
         enable(km3::Pipe);
         enable(km3::Chamber);
         enable(km3::ValveOrifice);
         enable(km3::CompChamber);
         enable(km3::RemnantGas);
         enable(km3::Body0D);
         // ...
      }
      virtual void enableLinks() {              // Base interface specification.
         enableLink(Pipe,FixedValue);
         enableLink(Pipe,Chamber);
         enableLink(ValveOrifice,Chamber);
         enableLink(Pipe,Body0D);
         enableLink(Chamber,Body0D);
         // ...
      }
};
```

In the preceding code, the *System* class is extended to the *Compressor* class which enables the essential components of the compressor and establishes the element combinations by defining the possible links. From here, the executable program to simulate compressor devices is finished and it does not require programming anymore apart from maintenance for fixing code bugs, improving or extending the capabilities.

Next step is to create a model for the particular compressor to simulate. That is, one should sketch out the configuration in a similar way as in Figure 2.12. This scheme is composed of several pipes, chambers and valves. The set of elements define the path of the fluid refrigerant inside the internal circuitry of the compressor. The fluid flows from the left -low pressure- to the right -high pressure-. To the left side there are several chamber elements representing the muffler. These components act as lungs to attenuate noise. The compression chamber in the middle of the sketch, marks out the separation between the low and the high pressure areas. This chamber is connected to the suction and the discharge valve and at the same time, it is subjected to the action of the piston. The flow is driven by the periodic motion of the piston which derives in the cyclic behavior that characterize this device. Beyond the compressor chamber is the discharge line and the refrigerant flows towards the exit.

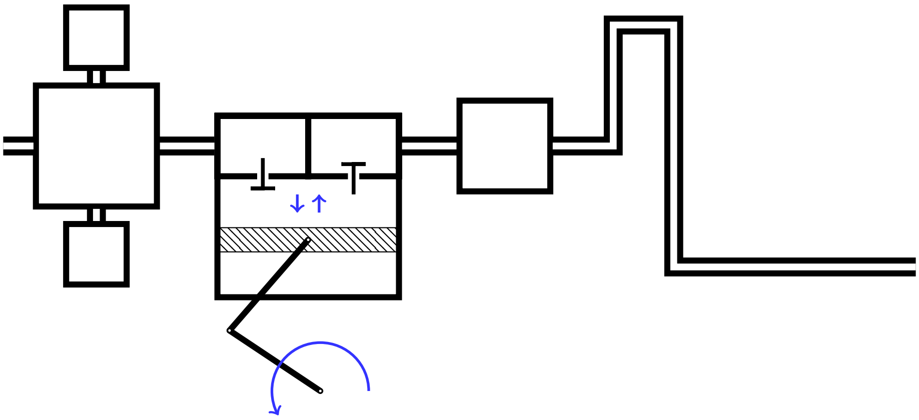


Figure 2.12: A compressor system based zero and one-dimensional models.

Once the model is outlined, it is converted to a formal markup language and written into a file. After partitioning the system, both the input file and the parallel topology are interpreted by the MSL parser to start the simulation. The steps for running a compressor simulation are depicted in Figure 2.13.

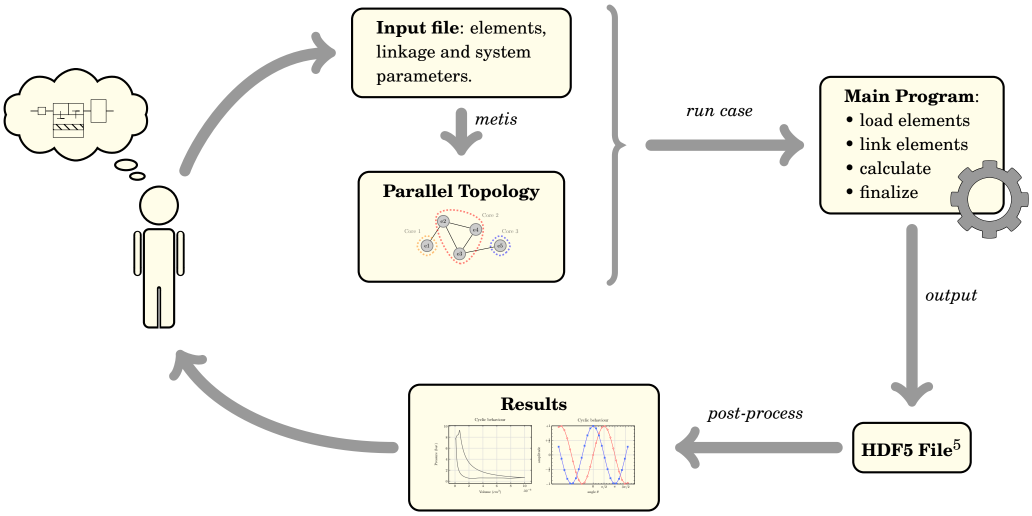


Figure 2.13: Steps for running a compressor simulation.

Finally, note the modularity of the approach. The program is able to create any virtual representation such as the one in Figure 2.12. Changing the configuration of the model by adding, modifying, replacing or removing elements is straightforward. For instance, one may be interested in studying new muffler configurations or changing the discharge pipeline length as illustrated in Figure 2.14.

[5] HDF5 stands for Hierarchical Data Format. This is a technology suite that makes possible the management of extremely large and complex data collections [21]

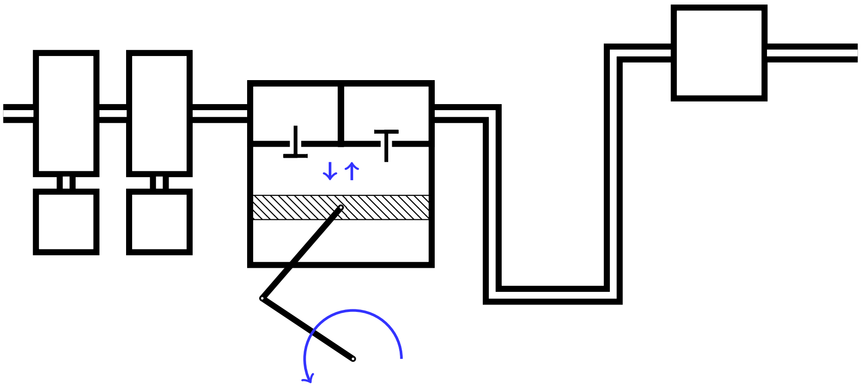


Figure 2.14: Alternative compressor model with a two-stages muffler and some modifications at the discharge line.

2.5 Conclusions

The development of software is necessarily an iterative process. At every revision the code is corrected, improved or extended. In general, it is more common inserting new code than removing obsolete parts. Consequently, the code is permanently growing in practice. Besides, this matter is particularly linked to code complexity. The more lines in the code the more difficult the reading is. This is an inherent property in software projects. Even so, complexity can be controlled doing a good abstraction of the faced problem by using object oriented programming languages and adopting the appropriate working approaches and tools.

Several researchers have been working on the development of this software during the last decade. As a result, several code libraries are now available, providing abstraction at distinct levels. The NEST concept is defined as the linkage of such libraries, which enable multiphysics together. In the scope of this work, the Modular System Library (MSL) has been paid most of the attention. The fundamental logics in system environments are defined in this library. In such a way a natural language for code design is obtained. From the abstraction process two essential entities are identified. On the one side is the *Element* concept. It represents components of the problem that have a particular entity. This entity can be of several types. For example geometric, physic, numeric, functional or a combination of them. On the other side, the set of elements are governed by several rules and relations as a whole. From this derives the *System* concept. Hence, *System* owns a set of *Element* instances on which it can perform some actions that enable the resolution of an overall problem in a coupled way. Furthermore, the code allows running the simulation into several processors. This is accomplished by means of the Parallel Typologies distributed with the BOL library.

Finally, a compressors program has been described in order to illustrate how the multiphysics framework is employed. The program enables the simulation of different configurations in a flexible and transparent way. This shows the modularity provided by the MSL library. Through this application example the steps required for running a case have been explained.

References

[1] Fred P. Brooks, Jr. *The Mithical Man-Month*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd edition, 1995. First edition: 1975.

[2] M. Fowler. Blog article: Is design dead? `http://www.martinfowler.com`. Accessed: 2015-08-20.

[3] C.S. Douglas. Guest editor's introduction: Model-driven engineering. *Computer*, 39(2):25–31, 2006.

[4] Bjarne Stroustrup. *The C++ Programming Language*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 3rd edition, 2000.

[5] Brian W. Kernighan and Dennis M. Ritchie. *The C Programming Language*. Prentice Hall Professional Technical Reference, 2nd edition, 1988.

[6] M. Fowler et al. *Refactoring: Improving the Design of Existing Code*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[7] Ron E. Jeffries, Ann Anderson, and Chet Hendrickson. *Extreme Programming Installed*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2000.

[8] Tom Dalzell. *The Routledge Dictionary of Modern American Slang and Unconventional English*. Routledge Taylor & Francis Group, 2009. p. 595, U.S. Navy "Project KISS" notes, headed by Rear Admiral Paul D. Stroop, 1960.

[9] Bazaar version control system. `http://bazaar.canonical.com`. GNU Project sponsored by Canonical. Accessed: 2015-08-20.

[10] R. Burrell. *Parallel algorithms for Computational Fluid Dynamics on unstructured meshes*. PhD thesis, Universitat Polotèctinca de Catalunya, October 2012.

[11] O. Lehmkuhl. *Numerical resolution of turbulent flows on complex geometries*. PhD thesis, Universitat Polotèctinca de Catalunya, July 2012.

[12] R. Damle. Object-oriented simulation of reciprocating compressors: Numerical verification and experimental comparison. *International Journal of Refrigeration*, 34(8):1989–1998, 2011.

[13] J. Lopez, J. Rigola, O. Lehmkuhl, and A. Oliva. A parallel object oriented code framework for numerical simulation of reciprocating compressors - introduction of solid parts modeling. In *8th International Conference on Compressors and Their Systems*, pages 731–738, London, UK, September 2013. Institution of Mechanical Engineers.

[14] D. Kizildag, O. Lehmkuhl, A. Oliva, and J. Rigola. A multi-functional ventilated façade model within a parallel and object-oriented numerical platform for the prediction of the thermal performance of buildings. In *International Conference on Solar Heating, Cooling and Buildings*, Aix-les-Bains, France, September 2014. EuroSun.

[15] R. Capdevila, O. Lehmkuhl, J. Rigola, J. Lopez, and A. Oliva. Analysis of iaq based on modeling of building envelope coupled with CFD&HT. In *3rd International High Performance Buildings Conference*, Purdue, IN, USA, July 2014. Purdue e-Pubs. Paper 165.

[16] R. Capdevila et al. Analysis of iaq based on modeling of building envelope coupled with CFD&HT. In *24th IIR International Congress of RefrigerationA*, Yokohama, Japan, August 2015.

[17] H. Kessentini, O. Lehmkuhl, R. Capdevila, J. Castro, and A. Oliva. Numerical simulation of heat transfer and fluid flow in a flat plate solar collector with tim and ventilation channel. In *International Conference on Solar Heating, Cooling and Buildings*, Rijeka, Croatia, September 2012.

[18] H. Kessentini, O. Lehmkuhl, R. Capdevila, J. Castro, and A. Oliva. Development of flat plate collector with plastic transparent insulation and low-cost overheating protection system. *Applied energy*, 133:206–223, 2014.

[19] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20:359–392, 1999.

[20] Message-Passing Interface. `http://www.mpi-forum.org`. Accessed: 2015-08-20.

[21] Hierarchical Data Format, HDF Group. `https://www.hdfgroup.org/HDF5/`. Accessed: 2015-08-20.

Chapter 3

Data transfer in partitioned coupled problems

3.1 Introduction

Partitioned approaches are divide and conquer based strategies for the resolution of highly non-linear coupled problems. The term comes from the idea that each involved phenomenon, or physics component, is resolved apart, using specifically designed simulation codes, frequently referred as *solvers* [1–4]. In this way, the overall complexity is reduced significantly since the problem is split into a set of smaller sub-problems whose numerical solution is simpler.

Unlike monolithic approaches, where one single code is used to solve all the involved phenomena simultaneously, the partitioned strategies have several appealing features that make them a good choice for general multiphysics software projects. First, there is the possibility to use different spatial discretizations or meshes. This means that each mesh can be defined only on the basis of numerical requirements, which can differ between the distinct physics components. Creating an unique mesh on the overall domain is not usually the most optimal choice. In practical applications this could lead either to overresolved regions, meaning bad computational load balancing, or to physically unresolved regions producing overall inaccuracy. Second, monolithic approaches tend to suffer from ill-conditioning of the matrices resulting from the linearitzation of the governing partial differential equations [3]. This may cause convergence problems in the resolution of the linear system of equations, which promotes the use of robust linear solvers rather than those are more efficient. Finally, the partitioning approaches have many advantages from the software engineering pointview. This is mainly because they provide code modularity, which is an important property. On the one hand, it makes possible to reuse either in-house or external -already implemented- codes. Writing entirely codes for the pursued challenges is definitively impractical. There are many well-known applications that already solve part of the computational problem (e.g. linear algebra packages, input/output data file standards, visualization toolkits, etc.). Hence, *solver* developers can focus on the details of the multiphysics problem, which simplifies much of the work. On the other hand, engineers and scientists are sometimes forced to use legacy codes. The reasons are too many. For instance, there is the need to guarantee the stability of the software by using well-tested trusted codes. Other advantages of code modularity are faster training of both new programmers and users, and also easier maintenance of the code (e.g. bug fixing, improvement or extension, etc.).

There are two challenging areas that must be addressed when partitioned approaches are used. First, there is the assessment of stability, accuracy and convergence properties of the time-integration scheme [5, 6]. Second, there is the creation of proper tools for exchanging data between *solvers*, which is critical to ensure equilibrium and continuity of the state variables within the contact interface. Although the overall accuracy depends on both, the time and the spatial coupling strategies [7, 8], this chapter is mainly focused on the second area. The former issue can be controlled by sub-iterating the partitioned scheme [1] or also, by using high order time schemes [9]. Nevertheless, the numerical experiments presented in the subsequent sections are only concerned on steady state solutions, which are obtained by means of a pseudo-transient time advancing scheme.

3.2 The data transfer problem

Data transfer methods have been paid many attention in the last decades [2, 3, 7, 10]. Most works in the literature are mainly focused on fluid-structure interaction (FSI) simulations. This problem is generally addressed using partitioned approaches, based on the separate resolution of the solid and the fluid domains. In order to establish a coupling condition, the *fluid solver* provides the pressure load to the *solid solver*, whereas this provides the structure displacements.

Despite the major focus is on FSI, the data transfer techniques used in this discipline are also applicable in other engineering and scientific fields. As a first attempt, the assessment presented in this chapter is mainly focused on diffusion problems. In this fashion, the complexity of the data transfer problem is reduced since the domains do not move or deform as happens in FSI. When this occurs, the meshes position and their overlapping must be recalculated at every time step, which involves many geometric operations that increase significantly the computational load of the simulation. Still, the work presented below is a necessary step towards the simulation of more complex phenomena using the NEST platform.

Types of data transfer

Regardless the nature of the problem, partitioned strategies always establish a set of internal boundaries whereby the data exchange takes place. Depending on the geometry of these boundaries as well as on the spatial discretization of the computational subdomains, the data exchange is *point to point*, *surface to surface* or a combination of both. In *point to point* configurations the data exchange is trivial and error free. You can take as an example when two or more zero or one-dimensional models are coupled, as happens in the house model depicted in Figure 3.1. Instead, the remain configurations need some kind of data approximation methodology to transfer data from one subdomain to its neighbors. For instance, when two three-dimensional *solvers* are coupled -*surface to surface* data exchange-, the meshes do not match at their interface (i.e. the contact boundaries). Hence, there can exist mesh dis-alignments or even gaps and/or overlappings at the interface, as happens in the example of Figure 3.2.

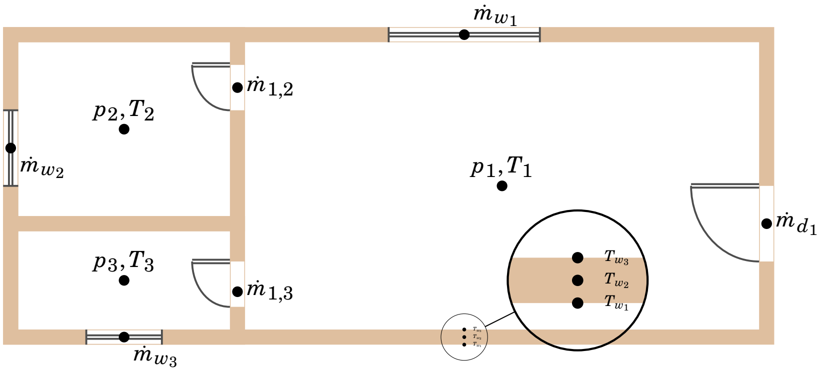


Figure 3.1: Zero and one-dimensional modeling of a house. The air in the rooms are modeled separately by means of lumped volumes. The several rooms are connected through doors or windows, which are also sub-models for evaluating the mass flow at the openings. All data exchanges in this house model are *point to point*. For instance, the rooms exchange their temperature, pressure and mass flow through the doors.

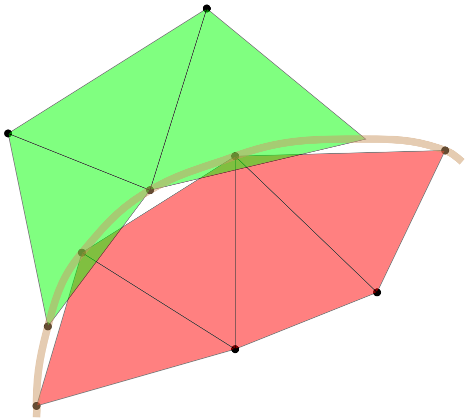


Figure 3.2: Gaps and overlappings between a pair of two-dimensional meshes. For clarity, it is preferable depicting 2D meshes instead of 3D meshes.

A note on code implementation

While resolving the data transfer method is already arduous, its implementation into a general multiphysics software can also become a hard task. This must be done in such a way that the modularity of the overall code is preserved. The *solvers* should be seen as black-boxes that have already been tested one by one. The implementation of a new data transfer methodology should never require changes in the *solvers* codes. Therefore, these tools are implemented in NEST as separate code entities. From a mathematical point of view, the data transfer methods can be seen as interpolation operators that operate with the *solvers*.

The election between the available methodologies is a trade-off between many factors: linked phenomena, detail of the models, the geometry of the *solvers* contact interface, code implementation issues, etc. None of the data transfer methods is always preferred or rejected. Sometimes, the most straightforward method performs very well. This is even possible in practical applications, as happens when several zero and one-dimensional mathematical formulations are used for modeling a system such as the one in Figure 3.1. On other occasions, the amount of data and operations involved in a data transfer process can make a methodology unfeasible. The ease of programming a certain methodology can also affect its feasibility. Specially under the NEST platform requirements commented above. Finally, the parallel performance is also very important in the context of multiphysics simulations. The methods that involve the motion of large amounts of data will likely perform very bad when they are executed in parallel.

3.2.1 Data transfer in non-matching meshes

The data transfer methodologies must satisfy some necessary conditions to achieve realistic results of the overall problem. It is well known that the transfer operations must be numerically accurate and physically conservative [2, 3, 7, 11]. By numerical accuracy one should understand small errors in a specific norm, while physical conservation stands for equilibrium at the contact interfaces. Data transfer will be repeated many times during a transient simulation which may lead to error accumulation at each transfer. Hence, these criteria are of great relevance to achieve both stability of the coupling algorithm and an accurate solution to the overall problem. For instance, the Navier-Stokes equations state that mass, momentum and energy are neither created nor destroyed. Breaking this law would result in unrealistic results, likely with unphysical meaning.

Data transfer in non-matching meshes is of special interest because it generally happens in most of the practical applications. A complete assessment on coupling methods for non-matching meshes was reported by Boer *et al.* in [7]. In that work, several methods such as the nearest neighbor interpolation, the L2 norm minimization or Weighted-Residuals formulation, and the use of the Radial Basis Functions (RBF) were discussed. Jiao and Heath [2] also reported a wide research on data transfer methodologies, particularly on the use of the Weighted-Residuals formulation. They focused their attention on a novel *common refinement* methodology that provides both accuracy and conservation to the solution. The RBF and Weighted-Residual formulation proposed by these authors are preferred in this thesis, rather than other classic methodologies such as nearest neighbor interpolation, consistent/alternative interpolation or spline interpolation. This choice is driven by the features of each methodology and its expected behavior.

Richard Franke in [12] proposed a set of criteria to evaluate the quality of scattered data interpolation techniques. These indicators are still up to date and are also considered in the current work. Between them, accuracy, sensitivity to parameters, storage requirements, timing and ease of implementation are the most relevant in the context of this thesis. First of all, the overall accuracy of the coupled problem is directly linked to the accuracy of the interpolation operation. Furthermore, sensitivity to parameters is also important. A parameter value that may give good results for one function might yield poor results for other functions sampled at the same points. Then, its choice is not always easy. If some method depends on many *ad hoc* parameters it will likely be a bad candidate. Apart from that, the timing issue has always been a concern in computer simulation. Donald Shepard was already concerned on the computational expenses in his work on the *Inverse Distance Weighted Method* (also known as *Shepard's method*) in 1968 [13]. Though computational power has grown significantly we are demanding more and more power aiming state of the art simulations [14]. The use of high performance computing facilities is a must in multiphysics simulation. Because of this, it is also important the ability of the data transfer method to work efficiently with massive datasets in parallel processes. Finally, integrating these methodologies into a wide numerical simulation framework is not always easy. Preserving code modularity and achieving good performance in parallel execution is tough programming work.

Although there are many approaches to the data transfer problem, the classic interpolation methods can not be used in the context of this work because the above requirements are not always satisfied. Some of them are not enough accurate, such as the nearest neighbor interpolation. Others are not strictly conservative, for example polynomial interpolations. Sometimes they are formulated assuming data is on regular grids or they only work well in 1D or 2D domains.

The mathematical formulation of the Weighted-Residuals and the RBF methods are briefly exposed next.

Weighted residuals

The data transfer problem consists on passing information from one mesh (the source mesh) to another mesh (the target mesh). Let f be a known function defined over the source mesh with m arbitrary control points and g be an unknown function defined over the target mesh with n arbitrary control points. Both functions can be approximated as a linear combination of shape functions as follows,

$$f = \sum_{i=1}^{m} f_i \phi_i \qquad g = \sum_{i=1}^{n} g_i \psi_i \tag{3.1}$$

The functions ϕ and ψ depend on the discretization method used for the respective *solvers* in the source and target domains (e.g. piecewise constant function in a finite volumes formulation or a piecewise linear or quadratic profile in finite elements).

The Weighted-Residuals method is based on the minimization of the L2 norm of $g - f$ [2, 15, 16]. This condition is achieved if $\partial \int_\Omega (g-f)^2 dx / \partial g_i = 0$, then by substituting equations 3.1 it can be obtained,

$$\sum_{j=1}^{n} \int_\Omega \psi_i \psi_j dx g_i = \int_\Omega \psi_i f dx \tag{3.2}$$

which can be expressed in matrix form as a $n \ x \ n$ linear system of equations $Mx = b$ (see [2] for a the detailed development). The vector x are the unknown values g_i on the target, M is the mass matrix with entries $M_{ij} = \int_\Omega \psi_i \psi_j dx$, and b the load vector with components $b_i = \int_\Omega \psi_i f dx$. If the shape functions in 3.1 are piecewise constant, the L2 formulation is reduced to the well-known method of Area-Weighted averaging [10, 17]. However, this is a general formulation that allows using linear (or quadratic) shape functions to approximate f and g. Both approaches are tested in this chapter.

The accuracy and conservation properties of the data transfer method depend on how the integrals from M and b are evaluated. The integral on the right hand side mixes shape functions from the source base and from the target mesh. Hence, it can be evaluated either over the elements of the source mesh (source scheme [16]) or the elements of the target mesh (target scheme [15]). Jiao and Heath exposed several arguments in [2] showing that the source based scheme is zeroth-order and the target based scheme is not conservative. Alternatively, they proposed to evaluate the right hand side integrals over a third mesh, or common refinement, resulting from the combination of both the source and the target meshes. With this approach the b vector integrals can be evaluated exactly using a Gauss quadrature. Their numerical experiments show that the common refinement scheme provides better accuracy and conservation properties than the others.

Radial Basis Function

The Radial Basis Function is a well-established tool for multivariate interpolation of both scattered and gridded data [18, 19]. The major advantage of the RBF method is that it is mesh-free because it only depends on pairwise distances between points. Since computing point distances is quite easy, it follows that this method can be used in any number of space dimensions. A standard RBF interpolant is obtained from a linear combination of basis functions centered at the scattered points x_j , $j = 1, ..., N$ as

$$s(x) = \sum_{j=1}^{N} \lambda_j \phi_j(||x - x_j||) \tag{3.3}$$

where λ coefficients are unknown. These coefficients are obtained from the interpolation condition, which states that the interpolant must exactly evaluate f at the data points x_j. This leads to a linear system $A\boldsymbol{\lambda} = f(x_j)$ in which A is symmetric ($A_{i,j} = \phi(x_i)$) and $\boldsymbol{\lambda}$ is a N dimensional vector containing the unknown coefficients.

Table 3.1 reports some commonly used basis functions (or kernels) that provide good approximation behavior in two-dimensional problems [18, 20]. It should be noted that changing the value of the *ad hoc* parameter ϵ, changes the shape of the basis function. This affects the conditioning of the A matrix and hence the convergence of the solution to the linear system is compromised. Furthermore, those functions are all of them defined on the entire domain. Consequently, A is a full matrix so that operating with that matrix is slow.

Name	Abbreviation	Definition
Gaussian	GA	$e^{-(\epsilon \boldsymbol{x})^2}$
Multi-Quadratic	MQ	$\sqrt{\epsilon^2 + \boldsymbol{x}^2}$
Inverse multiquadratic	IMQ	$1/\sqrt{\epsilon^2 + \boldsymbol{x}^2}$
Thin plate splines	TPS	$\boldsymbol{x}^2 log(\boldsymbol{x}/\epsilon)$

Table 3.1: Come widely used RBF kernels. For the sake of simplicity $\boldsymbol{x} = ||x - x_j||$.

Alternatively, aiming better computational properties of the interpolant, Beckert and Wendland [19] proposed a C2 kernel with compact support defined as follows

$$\phi(\boldsymbol{x}) = (1 - \boldsymbol{x}/r)^4_+ (4\boldsymbol{x}/r + 1) \tag{3.4}$$

In a similar way than ϵ in the kernels of Table 3.1, r is also a shape parameter referred as support radius. However, this parameter not only affects to the shape of the basis function in this case, but it also affects to the sparsity of the A matrix. The function is defined in such a way that the region of influence of each point is bounded by a sphere with radius r.[1] Changing the value of r affects the efficiency of the interpolation, which should be interpreted as a ratio between the accuracy and the computational time. A large support radius yields a good approximation order, whereas a small value leads to a stable system that can be easily solved [19]. Hence, the choice of an appropriate support radius is a trade-off between accuracy and feasibility.

3.3 Numerical experiments

In this section, the accuracy and conservation properties of the data transfer methods based on the Weighted-Residuals formulation and the Radial Basis Function (RBF) are examined. Two variants of the L2 norm minimization, summarized in Table 3.2, are studied. Both approaches are based on the integration of Equation 3.2 terms over a common refinement mesh as proposed in [2]. The Area-Weighted method (AW) uses piecewise constant expressions for the shape functions ϕ and ψ 3.1, while the Linear method (LIN) uses linear elements instead.

Later, the effect of the data transfer on the rate of convergence of a partitioned approach is analyzed through the Method of Manufactured Solutions or MMS. This procedure is a well-known method to verify that PDE codes are free of programming errors. The several test cases are carried out using TermoFluids [21]. Although TermoFluids *solvers* are already verified and validated, the MMS is still useful to investigate the effect of the data transfer on the overall accuracy of a coupled problem. Hence, this verification exercise is used here to compare the different transfer methods.

3.3.1 Transferring an analytical function

The accuracy of the data transfer methods from Table 3.2 is evaluated here employing analytical functions as proposed in [2]. The test consists on transferring an analytical

[1] The subscript + in Equation 3.4 means that this term is only taken for positive values.

Name	Abbreviation	Definition
Area-Weighted	AW	$\phi(x) = \psi(x) = 1$
Linear method	LIN	$\phi(x) = \psi(x) = \begin{cases} \frac{x - x_{k-1}}{x_k - x_{k-1}} & \text{if } x \in [x_{k-1}, x_k]\,, \\ \frac{x_{k+1} - x}{x_{k+1} - x_k} & \text{if } x \in [x_k, x_{k+1}]\,, \\ 0 & \text{otherwise,} \end{cases}$

Table 3.2: Two variants of the Weighted-Residuals method.

discrete function evaluated on the n control points of the source mesh to the m control points of the target mesh. The same operation is done in the reverse direction closing a cycle, as shown in Figure 3.3. In order to emphasize the cumulative error of the submitted methods, the transfer operation is repeated thousands of cycles. This is performed on two one-dimensional meshes of different size with constant step. The mesh size ratio is defined as

$$r_{cp} = \frac{n}{m} \tag{3.5}$$

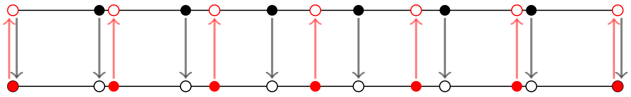


Figure 3.3: View of $mesh_1$ and $mesh_2$. In this experiments both meshes discretize the same segment.

The employed analytical function, known as Runge's function, is defined as

$$f(x) = \frac{1}{(1 + 25x^2)} \tag{3.6}$$

The results of the test are presented in Figure 3.4. Since the function to be approximated is given, the first and second derivatives are known, which makes possible using a cubic splines interpolation also. Note that the RBF is tested with different support radius $r = 0.1$, $r = 1$ and $r = 10$. In the y axis there is depicted the relative L2 norm error and in the x axis is depicted the number of transfer cycles. Through this study, you can see that the Cubic, the RBF, and the LIN methods are far accurate than the remaining ones. However, there are some other interesting observations also. First, the RBF is highly dependent of the support radius. The smallest value, $r = 0.1$, yields one of the worst results. On the other side, the AW method behaves better if one of the meshes is slightly staggered. The reasons for this are discussed below.

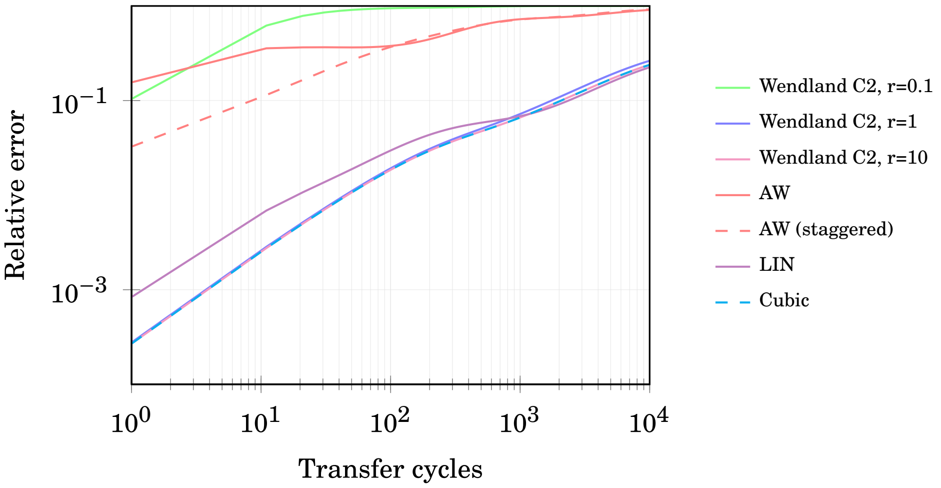


Figure 3.4: Multiple data transfers from $mesh_1$ to $mesh_2$. Using several methodologies.

Instability of the AW method

In Figure 3.5 is depicted the approximation of the Runge's function after one transfer, which is obtained by means of the AW and LIN methods. The left side chart shows data corresponding to the mesh size ratio $r_{cp} = 1.5$ and the right side chart shows data corresponding to the mesh size ratio $r_{cp} = 0.625$. The LIN method behaves well for both mesh size ratios. Instead, the AW produces high oscillations for $r_{cp} = 0.625$ (right chart). The visualization of the approximated function explains the poor accuracy of the AW method observed in Figure 3.4.

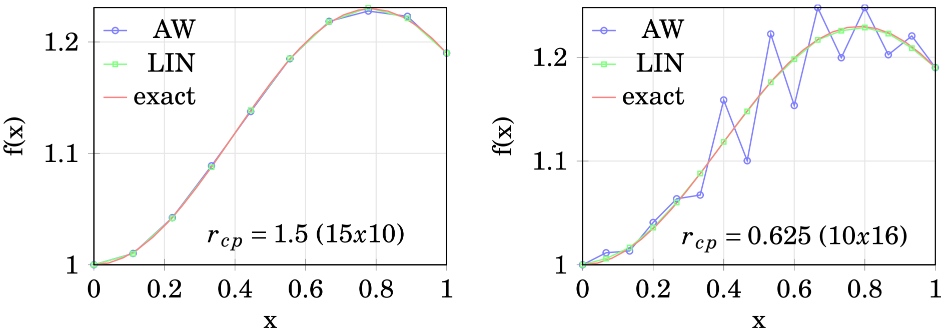


Figure 3.5: $f(x)$ profile after one data transfer. Two mesh size ratios are presented, from dense to coarse (left) and from coarse to dense (right).

For a deeper insight, a large set of tests have been run for different pairs of meshes. The LIN method is also tested here just to obtain some comparison data. In Figures 3.6 and 3.7 is shown the accuracy and conservation errors for more than 8000 pairs of meshes. The accuracy is evaluated as $\frac{||\hat{f}-f||_2}{||f||_2}$ where $\hat{f}$ is the approximated value and f is the analytical function. The conservation error is evaluated as $\frac{||\hat{I}-I||_2}{||I||_2}$ where $\hat{I}$ is the integral of the approximated data and I the integral of the analytical values. Both integrals are evaluated by means of the trapezoidal rule.

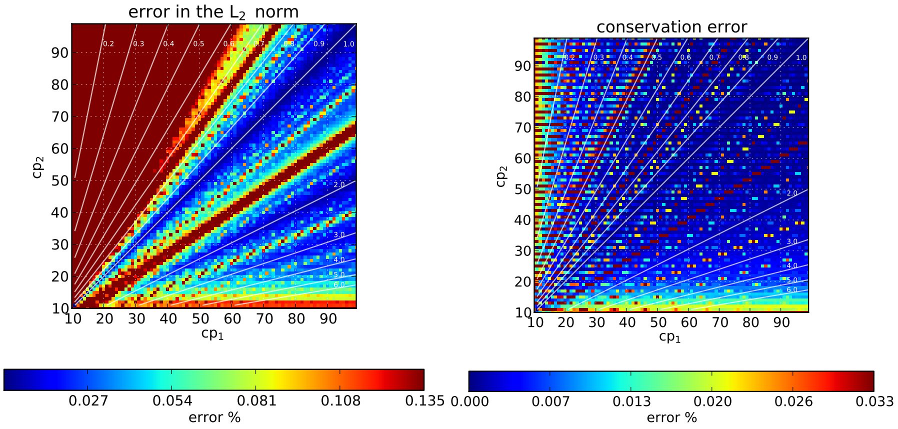


Figure 3.6: $f(x)$ error depending on mesh size ratio. Results obtained with piecewise constant elements (i.e. AW method). The scale is adapted for a better comparison with the LIN method.

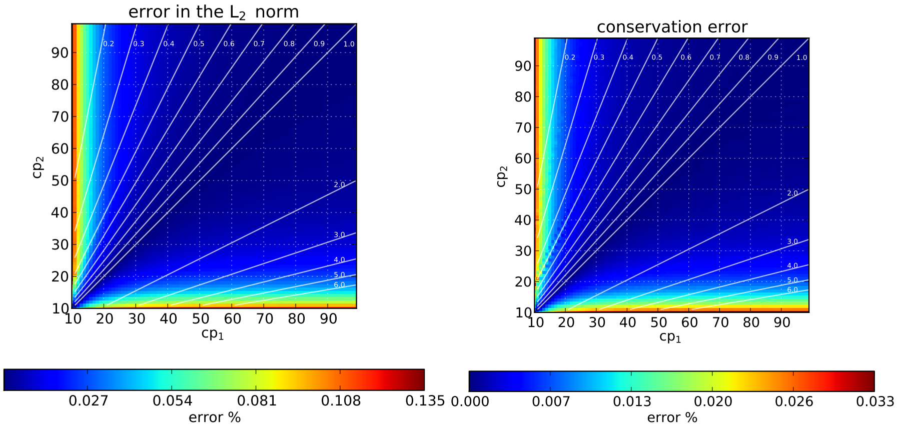


Figure 3.7: $f(x)$ error depending on mesh size ratio. Results obtained with linear elements (i.e. LIN method).

These charts show that the AW method is strongly affected by the mesh size ratio while the LIN variant errors only increase as the size of one mesh declines. In absolute terms, the maximum accuracy and conservation errors obtained with the AW variant are almost two orders of magnitude greater than those obtained using the LIN approach.

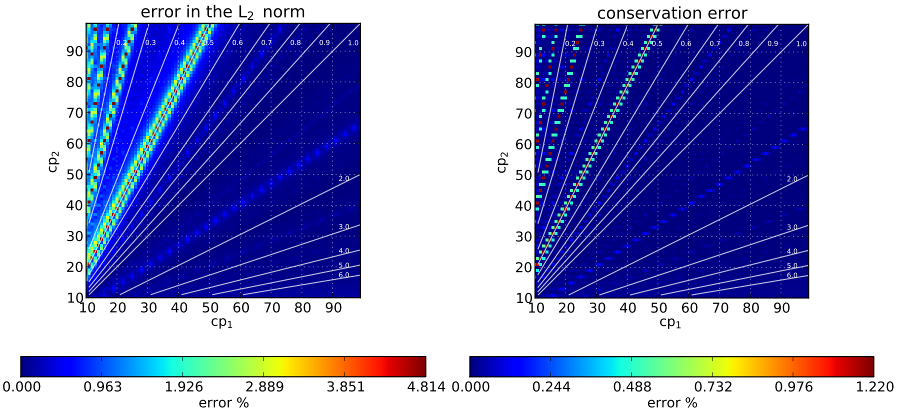


Figure 3.8: These are the same data presented in Figure 3.6 (i.e. AW method errors). However, the scale is adapted to the data range for better appreciating the biggest errors.

Furthermore, the scale of the AW errors is adapted to the full data range in Figure

3.8. An interesting pattern is highlighted using this scale. The maximum errors appear around constant mesh size ratios. This means that for certain ratios the error is always higher. Specifically, when $r_{cp} > 0.5$ the error is kept small. As the mesh size tends to $r_{cp} = 0.5$, the first big errors appear. After $r_{cp} = 0.5$ they become small again up to $r_{cp} = 0.25$, where another error band takes place. This is repeated as r_{cp} is reduced. Moreover, the lower is r_{cp} the more frequently do these error bands appear. This follows a clear sequence for r_{cp}: 1/2, 1/4, 1/6, 1/8 and 1/10. Therefore, when m is close to some even multiple of n then the errors are bigger. The maximum errors appear when m is exactly an even multiple of n. In such situation every control volume in $mesh_2$ overlays a set of control volumes from $mesh_1$, that is the meshes match by blocks as shown in Figure 3.9. This coincidence results in oscillations on the transferred data, as shown in Figure 3.5, and explains why accuracy of the AW method is reduced in Figure 3.4.

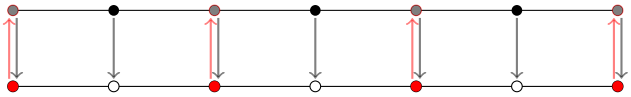
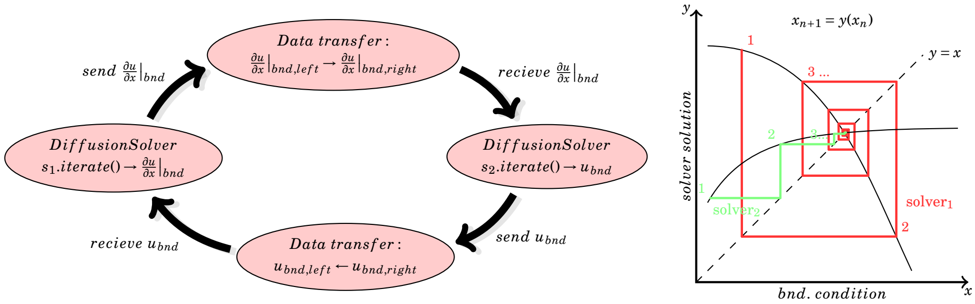


Figure 3.9: A pair of one-dimensional meshes that match by blocks.

3.3.2 Effect of the data transfer on the *solvers* rate of convergence

The Method of Manufactured Solutions (MMS) is employed here to investigate the effect of the data transfer on the overall accuracy of a coupled problem. Note that this analysis is not aimed to the understanding of some physical phenomena in particular, but it is just concerned on theoretical and implementation aspects of the data transfer methods. For this reason, the physical phenomena is limited to a pure diffusion problem within a continuous homogeneous domain, avoiding complex phenomena that would make the analysis harder but not more functional. In order to subject the data transfer methodologies to different situations, the MMS is carried out on two configurations, first a two-dimensional geometry - *2D setup* - and later a three-dimensional geometry - *3D setup* -. To convert these experimental setups into coupled problems both domains are split in two side by side parts. Under these conditions, the partitioned approach is applied using some data transfer methods included in Tables 3.1 and 3.2. The diffusion equation is solved at each sub-domain using two distinct instances of the *DiffusionSolver class*, which is a *solver* distributed with the TermoFluids software [21]. On the left side in Figure 3.10, there is a scheme representing the main levels of the resolution process.

The couping algorithm is based on a fixed-point scheme as stated on the right side in Figure 3.10.



$Data\ transfer:$
$\frac{\partial u}{\partial x}|_{bnd,left} \rightarrow \frac{\partial u}{\partial x}|_{bnd,right}$
$send\ \frac{\partial u}{\partial x}|_{bnd}$
$recieve\ \frac{\partial u}{\partial x}|_{bnd}$
$DiffusionSolver$
$s_1.iterate() \rightarrow \frac{\partial u}{\partial x}|_{bnd}$
$DiffusionSolver$
$s_2.iterate() \rightarrow u_{bnd}$
$recieve\ u_{bnd}$
$send\ u_{bnd}$
$Data\ transfer:$
$u_{bnd,left} \leftarrow u_{bnd,right}$
y
$x_{n+1} = y(x_n)$
$y = x$
1
3 ...
2
3
1
2
$solver_2$
$solver_1$
$solver\ solution$
$bnd.\ condition$
x

Figure 3.10: Scheme of the coupling procedure in the partitioned approach (left) and a graphical explanation of the expected convergence using fixed-point iterations (right). The boundary condition evolves towards an invariant situation that fits with the solution of both coupled problems simultaneously.

MMS setup

The MMS procedure is a tool for code verification [22]. Its strength is the ability to identify any code mistake that affects the rate of convergence of the numerical method. Roughly speaking, this method is aimed to check whether or not a code is solving the target equations right. Note that this is far and not concerned on verifying that the appropriate equations for capturing a phenomenon are solved. A manufactured solution is an exact solution to some Partial Differential Equation (PDE), obtained by solving the problem backwards. The analysis of this chapter is focused on the steady state diffusion equation, which is a PDE with the form

$$Lu = s \tag{3.7}$$

where L is the Laplacian operator, u is the solution, and s is a source term. The MMS consists first on creating an arbitrary expression for u and later on finding the value of s that satisfies Equation 3.7. In order to well test all the insides of the code, u must be general enough to execute all, or as many as possible, instructions in the code. In other words, it should exercise every term in the governing equation. For the steady diffusion problem the following definition satisfies this condition

$$u(x,y,z) = u_0\left[sin^2\left(\frac{x}{C_x}\right)sin^2\left(\frac{y}{C_y}\right)sin^2\left(\frac{z}{C_z}\right)\right] \tag{3.8}$$

Note that this expression is deliberately free of physical interpretation. Moreover, it only depends on the spatial coordinates as the time is not included in the expression. This is because the study is not focused on the time-discretization accuracy but on the spatial discretization accuracy only. In addition, this expression is smooth, continuous and it is infinitely differentiable. The source term s is obtained after applying the Laplacian operator from Equation 3.7 to the chosen definition of u (Equation 3.8), therefore

$$\begin{aligned} s(x,y,z) = 2u_0\Big[& \frac{1}{C_x^2}sin^2\left(\frac{y}{C_y}\right)sin^2\left(\frac{z}{C_z}\right)\left(cos^2\left(\frac{x}{C_x}\right)-sin^2\left(\frac{x}{C_x}\right)\right) \\ & +\frac{1}{C_y^2}sin^2\left(\frac{x}{C_x}\right)sin^2\left(\frac{z}{C_z}\right)\left(cos^2\left(\frac{y}{C_y}\right)-sin^2\left(\frac{y}{C_y}\right)\right) \\ & +\frac{1}{C_z^2}sin^2\left(\frac{x}{C_x}\right)sin^2\left(\frac{y}{C_y}\right)\left(cos^2\left(\frac{z}{C_z}\right)-sin^2\left(\frac{z}{C_z}\right)\right)\Big] \end{aligned} \tag{3.9}$$

The TermoFluids *DiffusionSolver class* is ready to account for artificial source terms like s in every control volume along the discretized domain. On another side, constant Dirichlet conditions are used at all the boundaries but the interface, since in this boundary the data transfer takes place. The values to fix the Dirichlet condition are directly evaluated from Equation 3.8.

With this setup, the code will provide a numerical approximation to Equation 3.7, lets say $\hat{u}$, whose error from the exact solution u can be evaluated as $||\hat{u}-u||_2/||u||_2$. The rate of convergence of the partitioned approach is defined as the tendency of this error versus the size of the discretization mesh. This is attained throughout the execution of several cases that use meshes of different sizes. These meshes are obtained by means of successive refinements in all the directions where the diffusion takes place. The number of control volumes of each new mesh is twice the size of the previous one. The choice of the refinement degree is free, although it must be constant for obtaining the right rate of convergence. Moreover, the successive mesh refinements must be applied on the meshes of both sides using the same criterion.

Next, the rate of convergence of the *DiffusionSolver* is obtained first on a *2D setup* and later on a *3D setup*. The *solver* is expected to maintain its convergence properties on both setups although, as shown in the following charts, these will depend on the employed data transfer method.

Rate of convergence of the 2D coupled problem

The geometry and the boundary conditions of this setup are depicted in Figure 3.11. Since the coupled problem is solved by means of a partitioning approach, the interface of the left sub-domain is fixed to Dirichlet condition while the right one is fixed to Neumann condition. For instance, when resolving heat diffusion, this means that the right sub-domain provides its surface temperature to the left sub-domain and at the same time it provides its surface heat flow to the former.

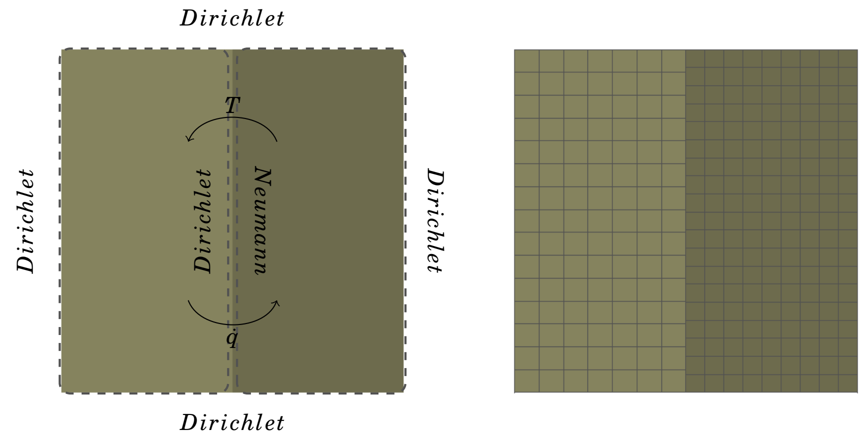


Figure 3.11: Two-dimensional sub-domains distinguished by color. The respective meshes are depicted at the right side. You should observe that there is a mesh dis-alignment.

In Figure 3.12 is presented the error of u evaluated as $||\hat{u}_{bnd} - u_{bnd}||_2/||u_{bnd}||_2$ for several pairs of meshes that match at the interface. For this particular case, the data transfer is free of error given that every face from one mesh match with some other face from the other mesh. Actually, this is the maximum accuracy that can be achieved using the partitioned approach. As expected, the rate of convergence is in accordance with the theoretical order of the spatial discretization scheme implemented in the TermoFluids *DiffusionSolver*. Note that any method would provide the same solution, even those are less accurate such as the nearest neighbor interpolation.

Furthermore, the error charts from Figures 3.13 to 3.16 are obtained using the Radial Basis Function method (RBF) with several kernels and the two variants of the

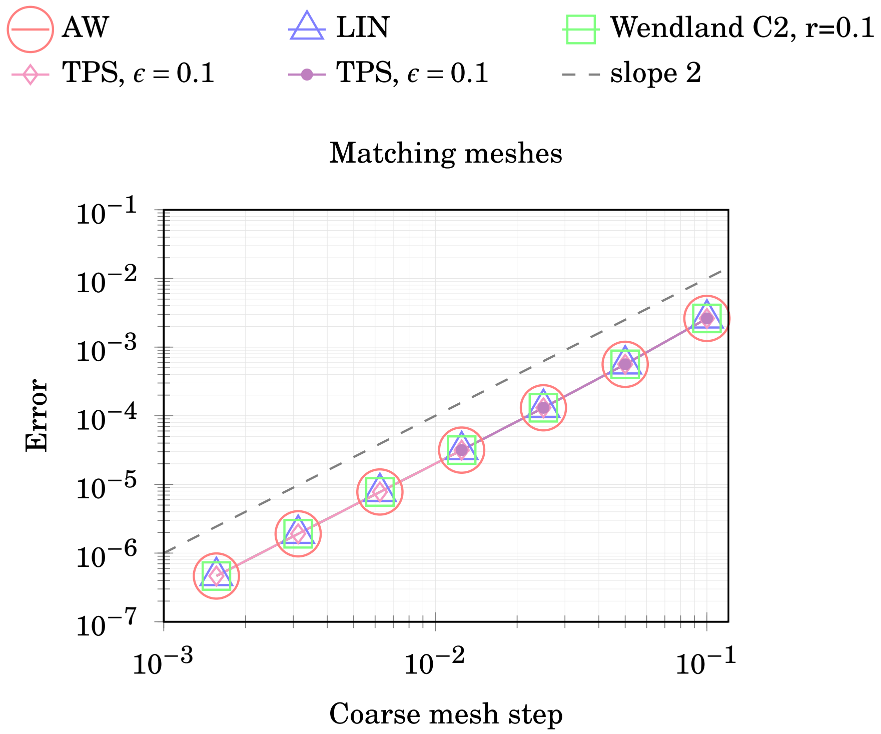


Figure 3.12: Rate of convergence when the meshes match to each other.

Weighted-Residuals formulation (AW and LIN). For every method there are two mesh setups, referred as *coarse to dense* or *CxD* (left hand charts) and *dense to coarse* or *DxC* (right hand charts). Both setups differ in the relation between the boundary conditions at the interface and the mesh sizes. While the meshes are swapped, the boundary conditions do not change. In the *CxD* setup u is transferred from the dense mesh to the coarse mesh while its derivative, $\partial u/\partial x$, is transferred in the opposite direction. In contrast, in the *DxC* setup this is done backwards. Analyzing both situations is necessary because the quality of the data transfer is not the same. The *CxD* scenario is the best one for most of the tested methods. The reasons for this are discussed below.

Focusing on every data transfer method, the RBF with the Wendland's C2 kernel (W. C2), shown in Figure 3.13, is among the methods that provide better rate of convergence and accuracy. However, this kernel is highly dependent on the support radius r, which have been set to 0.1, 1 and 10. The accuracy using $r = 0.1$ is far from the one obtained using $r = 1$ or $r = 10$. The smaller is the support radius the lower is the accuracy. According to Beckert and Wendland [19], the support radius should at least include the immediate neighbors of every interpolation point. For the coarser mesh used in this study a support radius $r = 0.1$ is close to that limit, which explains why the rate of convergence is not constant for this radius. In contrast, the results obtained with the Thin Plate Splines (TPS), Figure 3.14, show that this method is able to provide very good accuracy in most cases and have good convergence, too. Although it lacks of accuracy for coarse meshes, it does not depend so much on its *ad hoc* parameter ϵ (see Table 3.1). For its part, the Multi-Quadratic (MQ) kernel does also provide good results but, depending on its *ad hoc* parameter, the matrices may become ill-conditioned for dense meshes and some cases do not converge (e.g. MQ $\epsilon = 10^{-1}$).

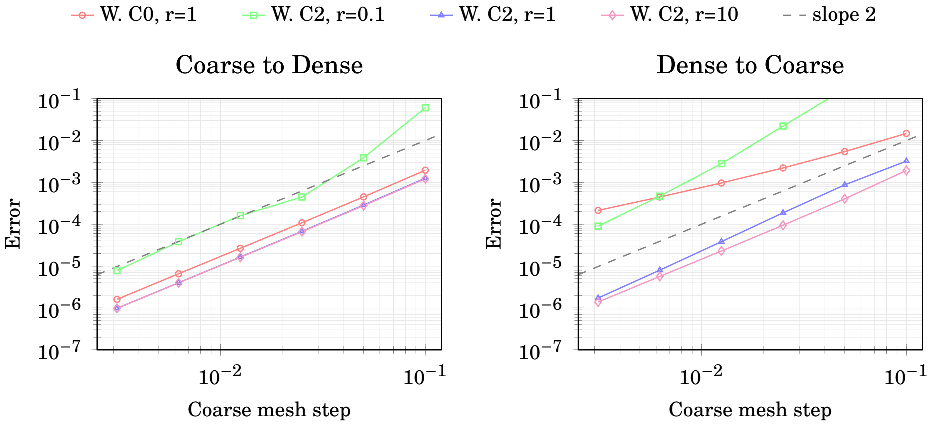


Figure 3.13: Rate of convergence of the RBF using Wendland's kernels with several support radius.

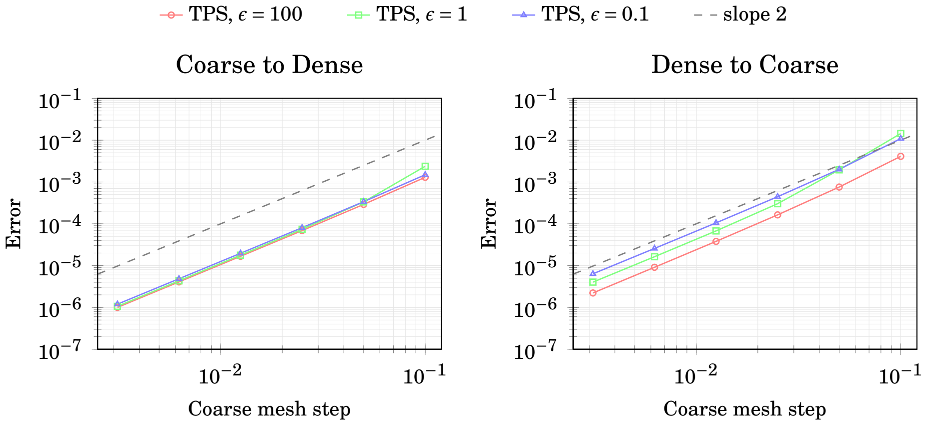


Figure 3.14: Rate of convergence of the RBF using the Thin Plate Splines kernel with several ϵ values.

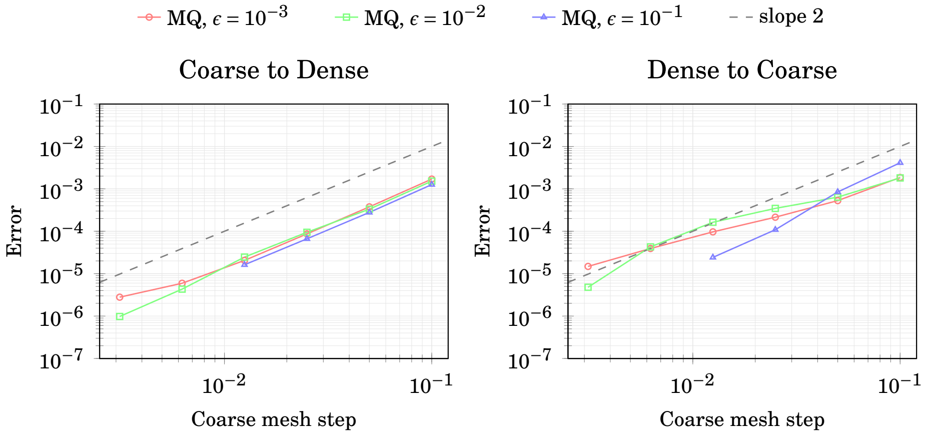


Figure 3.15: Rate of convergence of the RBF using the Multi-Quadratic kernel with several ϵ values.

In Figure 3.16 are presented the results obtained using the Weighted-Residuals formulation. While the LIN variant is more accurate than the RBF Wendland's C2 with the largest support radius, the AW approach has important accuracy problems. In fact, in the *DxC* setup, it is not able to improve accuracy with the successive mesh refinements (right side in Figure 3.16). The main reason is that this method produce high oscillations, as it was already observed in the previous Section 3.3.1. The sent (before applying the transfer) and the received (after the transfer) u and $\partial u/\partial x$ profiles at the interface are shown in Figure 3.17. Even though there are oscillations on both the left and the right charts, the data transfer operation produces significant oscillations only on u. Therefore, the left side sub-domain is receiving a damaged u, which derives to instabilities in the further evaluation of $\partial u/\partial x$, as can be seen on the right chart of Figure 3.17.

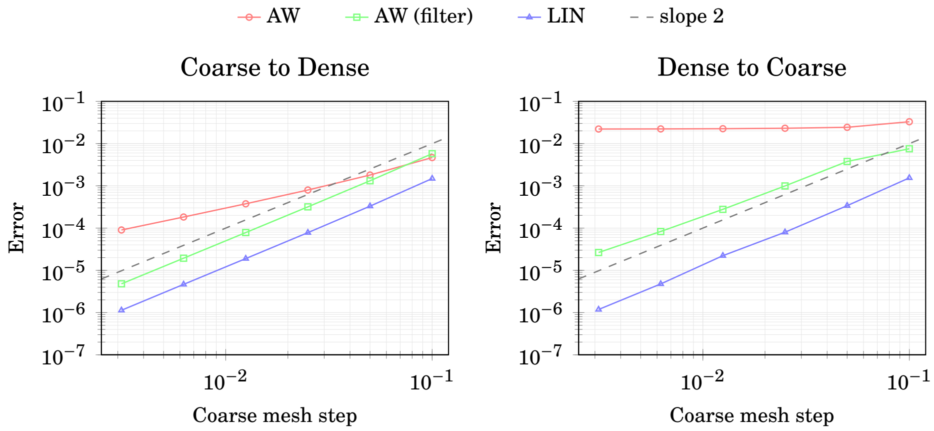


Figure 3.16: Rate of convergence of the Weighted-Residuals formulation. Both AW and LIN variants are shown.

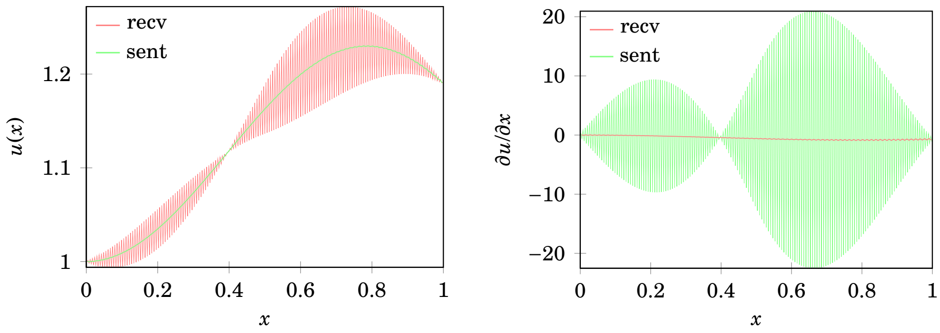


Figure 3.17: u and $\partial u/\partial x$ steady state profiles at the interface. Results using the AW method on the *DxC* setup.

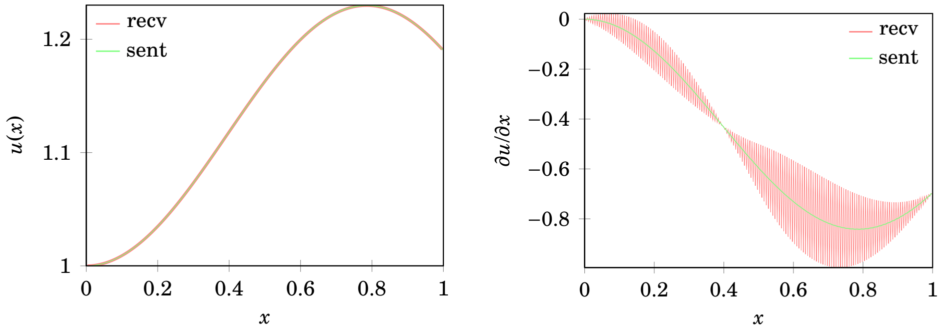


Figure 3.18: u and $\partial u/\partial x$ steady state profiles at the interface. Results using the AW method on the *CxD* setup.

In exchange, the errors for the *CxD* setup are quite better, as shown on the left side of Figure 3.16. The reason of this improvement is clear in Figure 3.18. In this case the transfer of u is much more accurate. Moreover, the oscillations only appear into the $\partial u/\partial x$ profile.

Therefore, the results presented here and those presented in the Section 3.3.1, manifest that the oscillations appear when the data transfer is done from the coarse mesh to the dense mesh and later the inaccuracies propagate over the multiple iterations. Actually, transferring from coarse to dense is logically less accurate due to the lower resolution of the source mesh. However, the study of both setups suggests that the oscillations do not have the same consequences transferring either u or $\partial u/\partial x$. According to the obtained results, transferring $\partial u/\partial x$ provides better overall accuracy, even though the transfer itself is fully inaccurate, as stated in Figure 3.18. For a better comparison of the analyzed methods, the u and $\partial u/\partial x$ profiles obtained using the LIN variant of the Weighted-Residuals formulation and those obtained using the RBF Wendland's C2 $r = 10$ method are also presented in Figures 3.19 and 3.20 respectively. The results are the same either for the *CxD* setup or for the *DxC* setup.

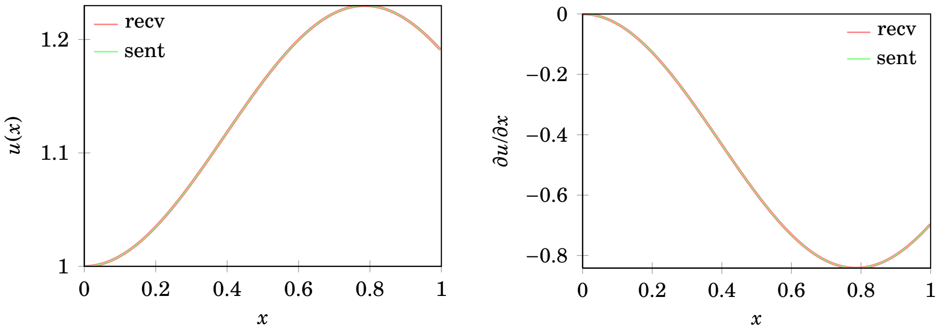


Figure 3.19: u and $\partial u/\partial x$ steady state profiles at the interface. Results using the LIN method on the *CxD* setup.

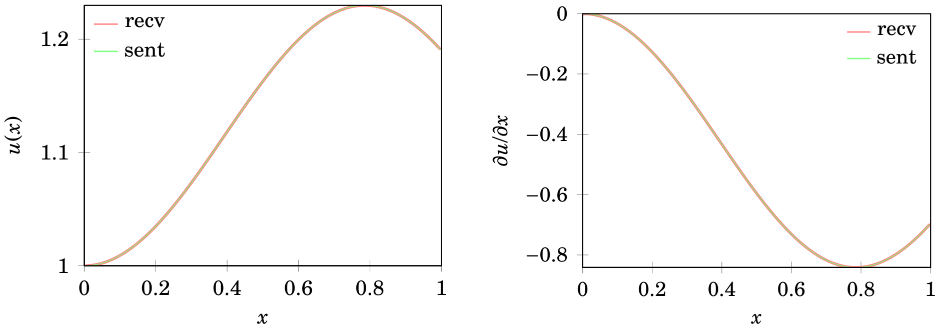


Figure 3.20: u and $\partial u/\partial x$ steady state profiles at the interface. Results using the RBF method with the Wendland's C2 $r = 10$ kernel on the *CxD* setup.

From this it follows that accurate data transfer is essential when transferring the primary variable while good conservation properties are enough when transferring its derivative. This means that important accuracy issues related to the use of AW method are significantly reduced if the Dirichlet and the Neumann sides (i.e. the boundary conditions) are adequately set.

In addition, the AW method can be improved thanks to its conservation properties. The oscillations in the received data are an identifiable "noise" over the signal that is ideally expected. In other words, the AW method is not accurate but the right solution is hidden within the received data. In order to get rid of this noise, a local average based filter is used, resulting on a significantly improvement of both the accuracy and the rate of convergence. The results of this approach are presented in Figures 3.16 and 3.21. In Figure 3.16, it can be seen that the rate of convergence using the filtered AW approach is 2, as it is expected. On the other side, in Figure 3.21, it is shown how the filter removes all the oscillations.

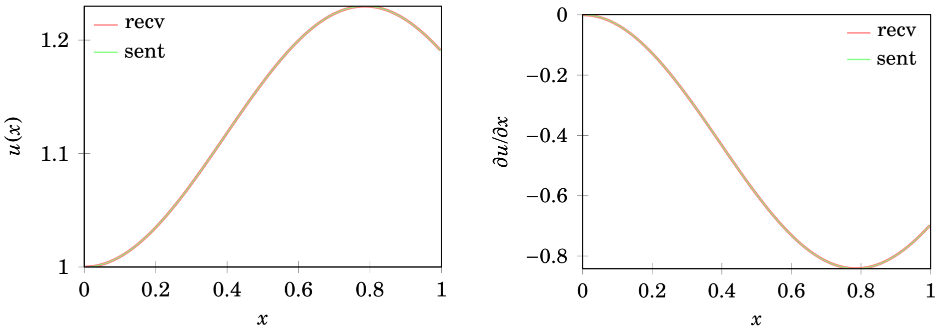


Figure 3.21: u and $\partial u/\partial x$ steady state profiles at the interface. Results using the filtered AW method on the *CxD* setup.

Rate of convergence of the 3D coupled problem

The three-dimensional geometry, the boundary conditions and some mesh examples of this setup are depicted in Figures 3.22 and 3.23. The geometry of this study is a portion of a radius 1 sphere. The full sphere is not resolved in order to reduce the computational time for each test. Since the diffusion problem is three-dimensional, the mesh refinements of the MMS study are necessary in every direction (x, y and z), which produces that the mesh size grows very fast.

As regards to the boundary conditions, all the external surfaces are fixed to Dirichlet. As in the two-dimensional case, the problem is solved by means of a partitioning approach. Arbitrarily, the interface of the inner sub-domain is fixed to Dirichlet condition, while the interface of the outer one is fixed to Neumann condition. Then, ∇u is transferred from the inside to the outside and u is sent in the opposite direction.

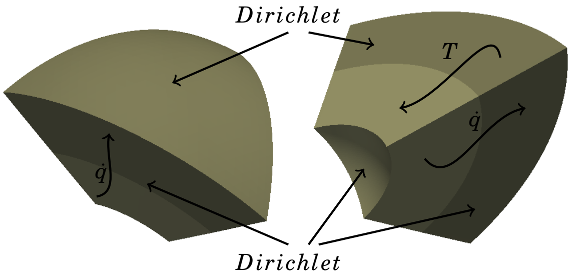


Figure 3.22: Perspective views of the three-dimensional domain. The sub-domains are distinguished by color. All external surfaces are fixed to Dirichlet condition. The inner interface surface is fixed to Dirichlet condition and the outer interface surface is fixed to Neumann condition.

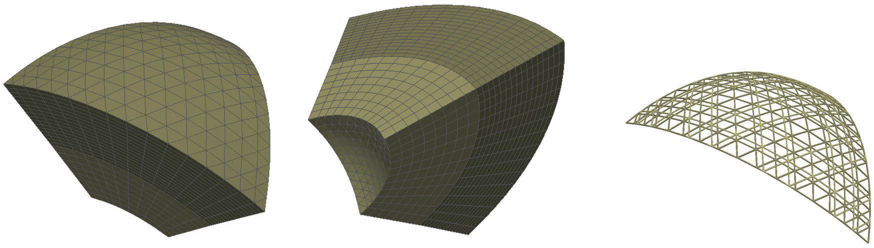


Figure 3.23: Two different views of the three-dimensional mesh (left) and detailed view of the meshes at the interface (right). Note that the meshes do not match to each other.

First of all, the rate of convergence for a set of matching meshes is presented in Figure 3.24. As expected the order of the spatial discretization scheme is preserved for all methodologies. Therefore, the employed partitioning approach is also valid for the three-dimensional setup.

Furthermore, the rate of convergence using non-matching meshes is presented in the charts from Figures 3.25 to 3.28. The Radial Basis Function method (RBF) with several kernels and the Area-Weighted variant (AW) of the Weighted-Residuals formulation are tested. Unfortunately, the Linear approach (LIN) is not directly applicable in the *3D setup* because the set of data to be transferred should be given at the vertexes of the mesh. Particularly, the TermoFluids *DiffusionSolver* is based on a Finite Volume Method (FVM), so the main variables are given at the centers of the mesh elements. A possible

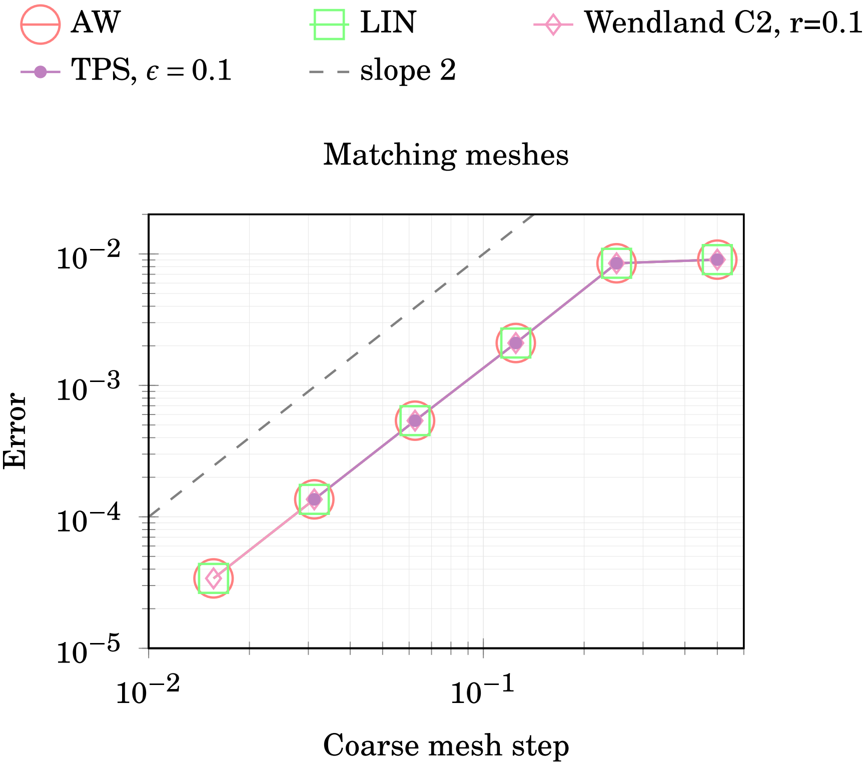


Figure 3.24: Rate of convergence when the meshes match to each other.

workaround is the use of staggered meshes. Although this is quite straightforward when 2D meshes are used, this is unfeasible using 3D meshes. Again, every method is tested on two different setups. In the first setup, the inner mesh is the coarser, *coarse to dense* or *CxD* (left hand charts) and in the second setup, the inner mesh is the dense one, *dense to coarse* or *DxC* (right hand charts). In this case, the *CxD* setup is also the best scenario. Therefore, as happened in the 2D coupling problem, transferring the primary variable from the dense to the coarse mesh is better than doing it in the opposite direction.

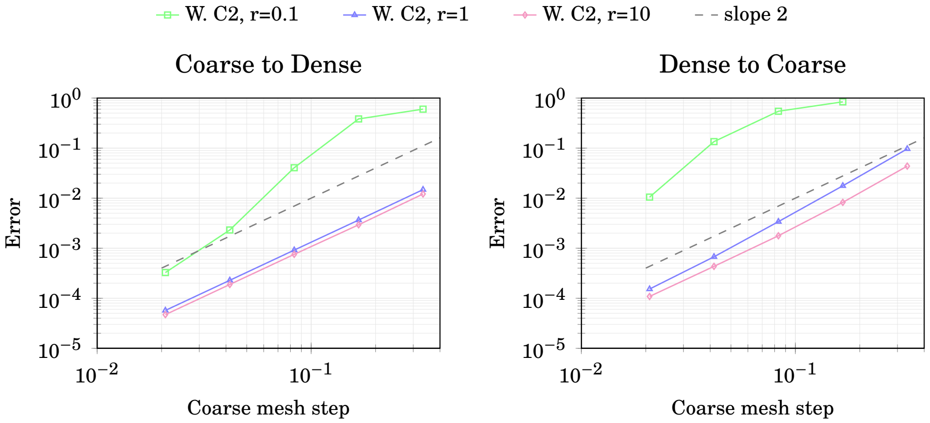


Figure 3.25: Rate of convergence of the RBF using Wendland's kernels with several support radius.

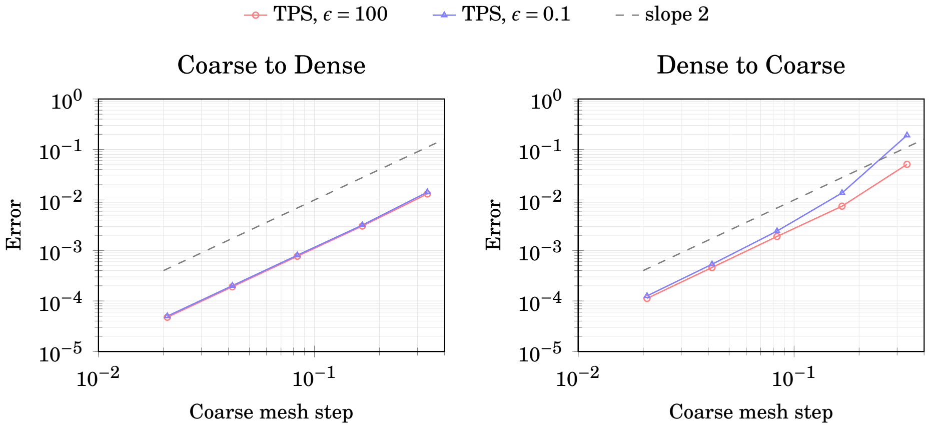


Figure 3.26: Rate of convergence of the RBF using the Thin Plate Splines kernel with several ϵ values.

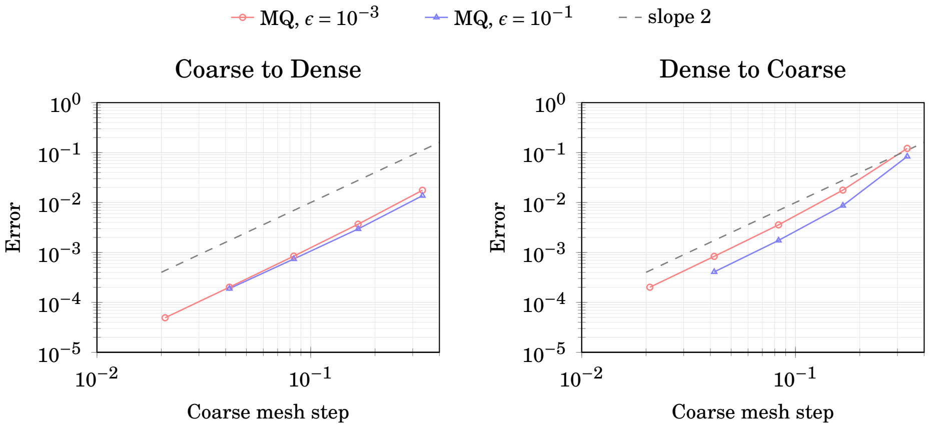


Figure 3.27: Rate of convergence of the RBF using the Multi Quadratic kernel with several ϵ values.

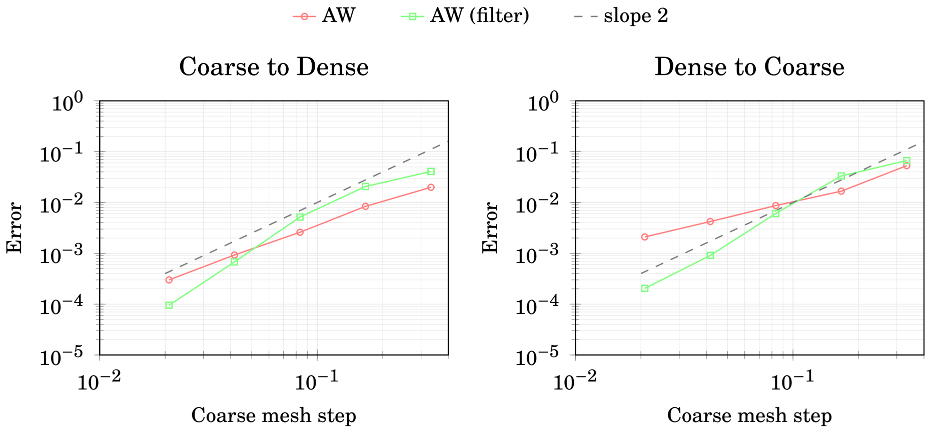


Figure 3.28: Rate of convergence of the AW method.

The observations regarding the RBF with the Wendland's C2 kernel (W. C2) are the same as in the previous 2D coupled problem. Three different support radius have been tested 0.1, 1 and 10. Newly, the smaller is the support radius the lower is the accuracy. In contrast, now $r = 0.1$ is not enough for some pairs of meshes, as can be seen at the right chart in Figure 3.25. Unlike the 2D setup, the Thin Plate Splines (TPS) is now able to provide the same accuracy, regardless the direction of the u and the ∇u transfers (see Figure 3.26). Once again, the Multi-Quadratic (MQ) method also seems a good option for coarse meshes, although it may diverge for dense meshes depending on its *ad hoc* parameter. Regarding the Area-Weighted method (AW), the accuracy properties observed in Figure 3.28 are not so bad as they were in the 2D problem. However, it is not so accurate as the remain methods. Moreover, the rate of convergence is slightly lower than 2, so this approach introduces significant error. In Figure 3.28, there are also presented the results obtained using the AW method supported with a second order Least Squares based filter. Using this approach, the rate of convergence is significantly improved, although the accuracy is only better for the most dense pairs of meshes. In order to avoid the oscillations commented in Section 3.3.1, the mesh pairs do not match in blocks at the interface. Hence, it could be expected that this phenomenon is not so relevant now. To make sure this is accomplished, the contour views of u and ∇u at the interface for three different data transfer methods are presented in Figures 3.29 and 3.30 respectively. From left to right, these results corresponds to the Area-Weighted method, the Wendland's C2 $r = 1$ method, and the filtered Area-Weighted. While u seems the same in all three cases, ∇u is clearly different for the normal Area-Weighted method. The appreciated loss of smoothness on the most left image is likely due to the effect of the oscillations. Therefore, the AW method suffers from instabilities even when the meshes are completely non matching.

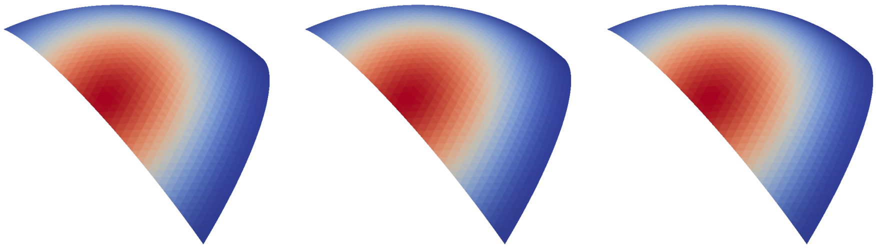


Figure 3.29: Steady state temperature at the interface between both sub-domains. From left to right: AW method, filtered AW and Wendland's C2 with support radius $r = 1.0$ (all data is depicted using the same scale). In this test the Dirichlet condition is fixed at the coarse mesh and the Neumann condition at the dense mesh.

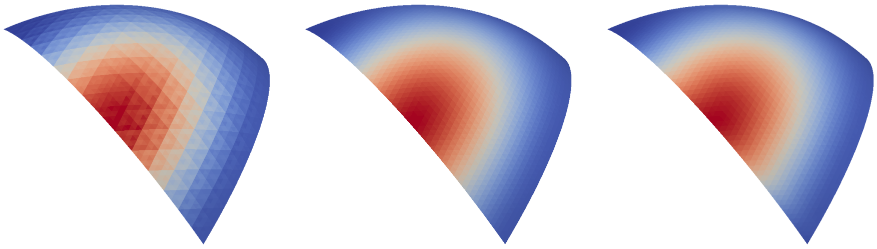


Figure 3.30: Steady state heat flow at the interface between both sub-domains. From left to right: AW method, filtered AW and Wendland's C2 with support radius $r = 1.0$ (all data is depicted using the same scale). In this test the Dirichlet condition is fixed at the coarse mesh and the Neumann condition at the dense mesh.

3.4 Conclusions

Partitioned approaches applied to numerical simulation of coupled problems require the use of data transfer methodologies. A transfer operation must be numerically accurate and physically conservative for assuring well-coupled overall solutions. In this regard, two extended methods for scattered data interpolation have been discussed and tested. On the one hand is the RBF with several kernels and on the other hand two variants of the Weighted-Residuals formulation (AW and LIN). The numerical experiments show that the AW method is able to achieve the same conservation properties than the LIN variant. However, the LIN is much more accurate than the AW, which suffers from oscillations depending on the size ratio of the contact meshes. In general, the RBF methods are able to provide good accuracy regardless the kernel. However, all kernels depend on *ad hoc* parameters that must be correctly fixed. While the TPS is not so sensible to these parameters, the Wendland's C2 function provides disparate accuracy depending on the choice of the support radius.

In all numerical experiments it is observed that transferring data from the coarse mesh to the dense mesh is less accurate than doing so in the opposite direction. There is no solution to this drawback since it is inherent to the partitioned approach. However, transferring the primary variable of the problem in that direction creates more inaccuracy than transferring its derivative. This is of special importance if the employed data transfer method lacks of accuracy, such the case of the AW method. A physical interpretation is that the energy is conserved even though the heat flow is definitively inaccurate.

Further research is necessary to decide which of the method is the most appropriate for simulating conjugate heat transfer, aerodynamics applications or multiphysics in general. On the one side, the parallel performance of the methods should be analyzed in detail. Apparently, the Weighted-Residuals formulation should behave better in parallel than the RBF methods. This prediction is based on the number of operations needed to complete one data transfer, as well as on the expected memory requirements. While the Weighted-Residuals does only take a matrix-vector product, the RBF methods involve first the resolution of a system of equations and later a matrix-vector product. In the presented numerical experiments, the mesh sizes are quite coarse in comparison to the ones used in the simulations of real cases. Actually, the tests presented in the chapter were executed in parallel using one processor for each sub-domain. Under this context, the execution times of the test cases are not so different to each other because the computational load and the latency time are quite similar. However, the RBF is expected to use much more memory as the mesh size increases. The more mesh size the more number of processors and therefore the higher is the halo perimeter. Note that the larger is the halo perimeter the larger is the data to be transferred. This is magnified when using the RBF since the growth of the halos size does not only depend on the number of parallel processes, but it also depends on the shape of the support function. Moreover, the parallel resolution of the RBF system of equations requires using iterative solvers. Hence, additional information must be provided to establish a convergence criteria. In that sense, the Wendland's C2 kernel has appealing features because the resulting RBF matrix is sparse. Instead, the TPS and the MQ lead to full matrices so they are bad candidates as the size of the interface is increased.

On the other side, the weakness of the Weighted-Residuals method is the need to resolve the intersection between the contact meshes. This is necessary to evaluate the interpolant weights. Moreover, the use of unstructured meshes is usually a good choice for a better fit of the geometry in practical applications. However, the intersection of unstructured meshes is computationally expensive and the algorithms to do so are based on heuristics strategies. The meshes do not only mismatch at the interface, but there can also exist gaps or overlapping regions. This forces an extra projection step during the mesh intersection algorithm, which adds more computational expenses and leads to an unavoidable loss of accuracy also. In exchange, the Radial Basis Function methods are mesh free. The numerical results show that the LIN variant of the weights residuals is the data transfer method that provides better accuracy. Unfortunately this method has not been successfully applied in three-dimensional FVM based *solvers*. In this kind of *solvers* the scalar fields to be transferred are cell-centered while the LIN approach expects these data to be vertex-centered.

Finally, it has been seen that the use of post-processing filters improves significantly the accuracy of the AW method. In the 2D coupled problem, a local average based filter is successfully applied to remove all oscillations from the received data. This makes it possible to obtain the expected rate of convergence and an admissible accuracy. With respect to the 3D coupled problem, the use of a second order Least Squares based filter does also contribute to improve the rate of convergence. In this case, however, the accuracy is only better for the most dense pairs of meshes. Consequently, this type of filter will lack accuracy on such regions where the meshes are coarser. It is well-known that the Least Squares method is non-conservative. This could explain the loss of effectiveness. More research is needed in this direction since the use of filtered AW methods could became a feasible alternative to the RBF in parallel simulation.

References

[1] C.A. Felippa, K.C. Park, and C. Farhat. Partitioned analysis of coupled mechanical systems. *Computer Methods in Applied Mechanics and Engineering*, 190:3247–3270, 2001.

[2] X. Jiao and M.T. Heath. Common-refinement-based data transfer between non-matching mehses in multiphysics simulations. *International Journal for Numerical Methods in Engineering*, 61:2402–2427, 2004.

[3] R.K. Jaiman, X. Jiao, P.H. Geubelle, and Loth E. Assessment of conservative load transfer for fluid-solid interface with non-matching meshes. *International Journal for Numerical Methods in Engineering*, 64:2014–2038, 2005.

[4] B. Gatzhammera, M. Mehla, and T. Neckela. A coupling environment for partitioned multiphysics simulations applied to fluid-structure interaction scenarios. *Procedia Computer Science*, 1:681–689, 2010.

[5] S. Piperno, C. Farhat, and B. Larrouturou. Partitioned procedures for the transient solution of coupled aroelastic problems. part i: Model problem, theory and two-dimeosional application. *Computer Methods in Applied Mechanics and Engineering*, 124:79–112, 1995.

[6] H.G. Matthies and J. Steindorf. Partitioned but strongly coupled iteration schemes for nonlinear fluid–structure interaction. *Computers and Structures*, 80:1991–1999, 2002.

[7] A. Boer, A.H. Zuijlen, and H. Bijl. Review of coupling methods for non-matching meshes. *Computer Methods in Applied Mechanics and Engineering*, 196:1515–1525, 2007.

[8] A. Boer, A.H. Zuijlen, and H. Bijl. Comparision of conservative and consistent approaches for the coupling of non-matching meshes. *Computer Methods in Applied Mechanics and Engineering*, 197:4284–4297, 2008.

[9] A.H. Zuijlen and H. Bijl. Implicit and explicit higher order time integration schemes for structural dynamics and fluid-structure interaction computations. *Computers and Structures*, 83(2-3):93–105, 2005.

[10] J.D. Ramshaw. Conservative rezoning algorithm for generalized two-dimensional meshes. *Journal of Computational Physics*, 59:193–199, 1985.

[11] R.W. Johnson, G. Hansen, and C. Newman. The role of data transfer on the selection of a single vs. multiple mesh architecture for tightly coupled multiphysics applications. *Applied Mathematics and Computation*, 217:8943–8962, 2001.

[12] R. Franke. Scattered data interpolation: Test of some methods. *International Journal for Numerical Methods in Engineering*, 38(157):181–200, 1982.

[13] D. Sephard. A two-dimensional interpolation function for irregularly-spaced data. *Proceeding of the 23rd ACM national conference*, pages 517–524, 1968.

[14] D.E. Keynes et al. Multiphysics simulations: Challenges and opportunities. *International Journal of High Performance Computing Applications*, pages 4–83, 2013.

[15] N. Maman and C. Farhat. Matching fluid structure meshes for aeroelastic computations: a parallel approach. *Computers and Structures*, 54(4):779–785, 1995.

[16] J.R. Cebral. Conservative load projection and tracking or fluid-structure problems. *AIAA Journal*, 35(4):687–692, 1997.

[17] J. Dukowicz. Accurate conservative remapping (rezoning) for arbitrary lagrangian-eulerian computations. *SIAM Journal on Scientific and Statistical Computing*, 8(3), 1987.

[18] A. Beckert and H. Wendland. Theory and applications of the multiquadric-biharmonic method. *Computational Mathematics Applications*, 19(8-9):163–208, 1990.

[19] A. Beckert and H. Wendland. Multivariate interpolation for fluid-structure-interaction problems using radial basis functions. *Aerospace Science and Technology*, 5:125–134, 2001.

[20] J. Duchon. Splines minimizing rotation invariant semi-norms in sobolev spaces. In *Constructive Theory of Functions of Several Variables*, volume 571, pages 85–100, Berlin, 1976. Springer. Lecture Notes in Math.

[21] O. Lehmkuhl, C.D. Perez-Segarra, R. Borrell, M. Soria, and A. Oliva. Termofluids: A new parallel unstructured CFD code for the simulation of turbulent industrial problems on low cost PC cluster. *Lecture Notes in Computational Science and Engineering*, 67:275–282, 2009.

[22] K. Salari and P. Knupp. Code verification by the method of manufactured solutions. Technical report, Sandia National Laboratories, P.O. Box 5800,Albuquerque, NM 87185-0825, 2000.

Chapter 4

Simulation of buildings using NEST

Human activities are dramatically modifying the amounts of gasses in the Earth's atmosphere. This can lead to a warming or cooling of the climate system since it disrupt the natural greenhouse effect. Today, gas emissions intensify the greenhouse effect causing global warming. The main anthropogenic emissions consist on CO_2 (57%), CH_4 (14%) and N_2O (8%)[1] gasses, mainly coming from combustion of carbon-based fuels (coal, oil and natural gas) to support growing demand of energy. This is also combined along with deforestation due to land use conversion [1]. It is estimated that CO_2 emissions have raised by 40% from the beginning of the industrial revolution in 1750, and it is projected a growth in CO_2 emissions up to 78% by 2050 if no further actions are taken [2].

As response to this discouraging outlook, several new energy policies have been promoted worldwide in the last years. At international level is the Kyoto protocol, which is now in its second commitment period (2012 Doha Amendment to the protocol [3]). Although the treaty was signed by 189 United Nations members, the major GHG emitters (China, USA and India) did not ratify it and some others withdrew in 2012 (Russia, Japan or Canada). Nevertheless, regardless the frustrated diplomatic efforts, world states are developing new national regulations for reducing GHG emissions and global warming. For instance, in the USA within the Energy Independence and Security Act (EISA, 2007) [4] and in the EU28 within the 2020 Energy Strategy [5] and the Energy Efficiency Directive (EED, 2012) [6]. According to the International Energy Agency (IEA) [7], the worldwide investment on improving the energy efficiency will

[1] Other greenhouse gasses, or just GHG, are water vapor, nitrous oxide, ozone or F-gasses such as HFCs and PFCs.

quadruple from \$130 to \$550 billions by 2035.

EU28 (ktoe)	1990	2000	2013	$\%_{1990}$	$\%_{2000}$	$\%_{2013}$
Industry	367623.5	331924.2	276637.6	34.07	29.39	25.09
Transport	284182.8	345190.5	348548.1	26.34	30.56	31.61
Residential	273578.5	294357.5	295876.5	25.36	26.06	26.83
Agriculture/forestry	31645.6	27749	23898.2	2.93	2.46	2.17
Services	108969.3	117681.2	152541.3	10.10	10.42	13.83
Others	12968.4	12646.4	5177.5	1.20	1.12	0.47
total	1078968.1	1129548.8	1102679.2	100	100	100

Table 4.1: Final energy consumption in the EU28 by sector for the years 1990, 2000 and 2013 [8] (1000 tonnes of oil equivalent).

According to the data in Table 4.1, buildings are responsible for 40% of energy consumption in the EU (residential and services sectors). In addition to the Energy Efficiency Directive [6], the EU28 also approved the Energy Performance of Buildings Directive [9] which specifically commits member states to reduce the energy consumption of buildings.

The amount of gross floor space in the EU is estimated about 25 billion m^2, which is increased at a rate of 1% per year[2] [10]. From this trend derives a growth of the energy demand associated with EU buildings also. This highlights the importance of improving the energy efficiency of the current EU stock. This is critical for the older buildings in particular. About 62% of European buildings are prior to 1990 and 17% are prior to 1960 [10]. While new buildings generally need less than 3 to 5 liters of heating oil per square meter per year, older buildings consume about 25 liters on average. Some buildings even require up to 60 liters. Improving the energy efficiency of buildings will reduce energy demands in all their stages (construction, usage and maintenance) and subsequently GHG emissions, too. Moreover, this will also provide healthier conditions for the occupants.

In sum, new policies are progressively converging to nearly zero energy standards. Zero net energy buildings, commonly now as ZEB [11, 12], combine high levels of energy efficiency with renewable energy systems (solar, biomass, windmill, etc.) to offset the energy delivered to the building over the year. The EISA 2007 specifies a zero-energy

[2] To get a better idea 25 billion m^2 is less than Belgium's (30,528 km^2) or Netherlands's surface (33,893 km^2)

target for 50% of US commercial buildings by 2040 and for 100% of them by 2050. On the other hand, the EU regulation seeks 'nearly' ZEB target for all public buildings by 2018 and from 2020 for all new public buildings.

Therefore, the energy assessment of buildings at any stage of their life-cycle is essential for attaining the current challenges. Experimentation plays an important part in building analysis in so far as it is vital for understanding the different phenomena and their governing parameters. However, full scale experiments are not possible in most cases due time and cost constraints. The building concept is totally dependent on the climate zone, its use, the location and even the culture. Examples showing the diversity in terms of typology within both the residential and the non-residential sector are: single or multi-family dwellings, apartment blocks, shops, hospitals, hotels, supermarkets, schools, sports centers, etc. In addition to typology, buildings vary greatly in terms of age, size and location.

For this reason, the development of computational tools for predicting buildings behavior is critical as they enable the construction of virtual designs providing lots of flexibility. What is even more important, is the ease of repeatability. For example, buildings codes can be used intensively for analyzing many scenarios: testing innovative materials (as yet non-existing), different locations and usages, any type of weather conditions, systems for improving comfort and so on.

In this perspective, building performance simulation are potential tools not only during the design phase but also for commissioning and operation. It can assess the commitment of new regulations or the implementation of leading-edge technologies for attaining energy efficiency while meeting expectations for human well being. To know some examples, Vietes et al. [13] ranks the major projects carried out in Europe and their achievements regarding this issue. In the DCCS project [14], the NEST software is embedded into a real-time decision-making software for predictive control. NEST was also used in the RESSEPEE project [15], aimed to advance, adapt, demonstrate and assess a number of innovative retrofit technologies and materials pursuing reductions in the energy consumption of European buildings.

4.0.1 Literature review

Numerous building energy simulation programs have been developed over the last decades. MODSIM [16], EKS [17], SPARK [18], are some of the earlier object oriented initiatives for modular simulation. COMIS [19] is based on a multizone model prepared by experts of the IEA and was widely used in the USA. At the late nineties started some projects that are still running. CONTAM [20] which is multizone airflow and contaminant transport analysis software. EnergyPlus [21] is an open-source building simulation platform for simulating both energy consumption and water use in buildings. Nowadays, SPARK models of HVAC components, systems and controls can be run within EnergyPlus. An overview of twenty major simulation programs (e.g. BLAST, EnergyPlus, TRNSYS, eQUEST, etc.) and their capabilities was presented by Crawley et al. [22] and an overview of methods for ventilation performance can be found in [23]. Recently, Damle [24] dedicated one chapter of his PhD thesis to the object oriented simulation of buildings. An up to date state-of-the-art, mainly concerned on the role of building performance simulation, is presented in [25].

4.0.2 Structure and Objective

In this chapter a building simulation program based on the NEST software platform is presented. The main goal is to show how NEST is successfully employed in the production of computer simulation programs for specific applications. This program is aimed to obtain, in a fast and reliable way, the thermal performance of buildings in order to make them more energy efficient. It is also possible to calculate the concentrations of CO_2 and/or volatile organic components (VOC) in order to evaluate the necessary air change rate and estimate an overall building air quality. In the following section the several NEST sub-models that enable this are briefly explained. After this, the potential of the code is illustrated through the simulations of two different residential setups. First, an apartment house is simulated under several circumstances depending on the occupancy and events. Later, the thermal performance of a semi-detached dwelling located in Lemmer (Netherlands) is analyzed over one year with and without space heating.

4.1 The building model

The buildings simulation program consists on a set of individual numerical sub-models that solve the conservation laws of mass, momentum and energy along the buildings spaces and components (rooms, doors, windows, facades, ground, etc.). Moreover, HVAC as well as other non-HVAC equipment that seek people comfort can also be incorporated into the simulations. The occupants behavior and the weather conditions can be considered as well. All these phenomena, equipment, behavior and events are referred as objects or elements. They should be seen as small pieces of code implemented on the top of the NEST platform.

NEST provides several tools for linking the elements so that they are able to exchange information to each other. In this way, a network of numerical components can be arranged in order to create an overall model of the building. This network can be resolved in a coupled way, resulting in the solution of the fluid-dynamics and the heat transfer over the whole building. This is illustrated in Figures 4.1 and 4.2, where the model of a room and its corresponding graph representation are depicted.

This partitioned approach provides an efficient and practical way to simulate buildings. Unlike monolithic approaches, where all phenomena are resolved by means of a single numerical code, this strategy is much flexible since it allows rapid setup of new buildings configurations. This is an appealing property in so far as there are a wide array of buildings types.

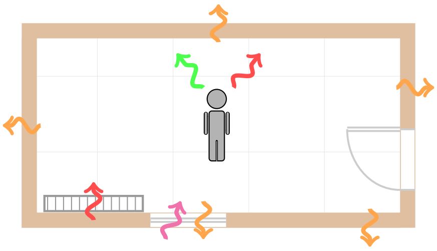


Figure 4.1: Hypothetical room model with several walls, a window, a door, a radiator and an occupant.

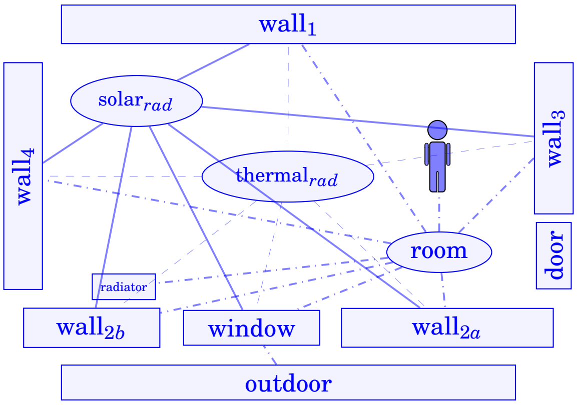


Figure 4.2: NEST model graph representation of the room model in Figure 4.1

4.1.1 NEST models for buildings

Below is given a description of the NEST sub-models implemented specifically for simulating buildings. This chapter is not aimed to make a dissertation on modeling approaches for buildings. Rather, the chapter illustrates one more application of the NEST platform. For a more detailed description of each model presented below one should refer to [24].

Rooms and Openings

The air flow between the rooms of a building is essential for analyzing the thermal performance, the ventilation and the air conditioning of building spaces. From the point of view of the NEST model, *Rooms* are lumped volumes that contain air at uniform conditions (pressure, temperature, moisture, CO_2 concentration, etc.). These conditions depend on the physical context of each room (e.g. location in the building, usage, construction materials, HVAC systems, and so on).

Rooms can interact to each other through their walls. Furthermore, they can take or release air from or to the neighbor rooms through its *Openings*. The *Openings* are balancing elements that evaluate the air mass flow and its direction at the connections of a Room element. For instance, a door is modeled as an *Opening* that connects two rooms or in a similar way, a window is modeled as an *Opening* that connects a room with the outside of a house. They can be opened and closed during the simulation. In case they are closed, the simulation program accepts an infiltration coefficient to take into account the air flow through the small gaps in the openings frame. When there are glazed windows, *Rooms* can also receive solar energy. In addition to the surrounding phenomena, *Rooms* can also change their state due to the presence of people, the activities they do or the influence of to HVAC systems, too. This is illustrated in Figure 4.3.

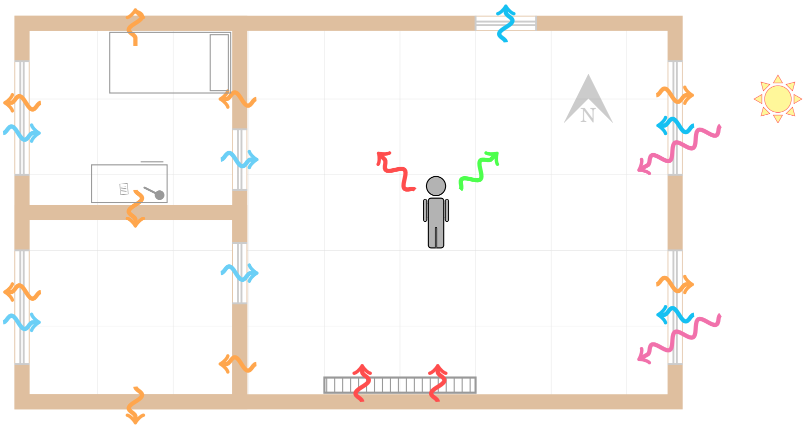


Figure 4.3: Some of the possible interactions between rooms, equipment, people and the outside of a house: infiltrations (cyan), heat sources (red), heat conduction (orange), solar irradiation (magenta) CO_2 emissions (green), etc.

The general conservation laws of the fluid flow in their semi-discrete form are presented in Equations 5.1, 5.2 and 5.3. These equations are solved in the *Room* and the *Opening* sub-models in a coupled way. The pressure-velocity coupling is resolved by means of the SIMPLE method (One-dimensional Semi-Implicit Method for Pressure Linked Equations [26]).

$$\frac{\partial m}{\partial t} = \sum [\dot{m}]_{bnd} \tag{4.1}$$

$$\frac{\partial (mv)}{\partial t} = \sum [\dot{m}v]_{bnd} + F_{bnd} \tag{4.2}$$

$$\frac{\partial [mc_pT]}{\partial t} = \sum [\dot{m}c_pT]_{bnd} + \dot{Q}_{src} + \dot{Q}_{walls} \tag{4.3}$$

The transport of species (e.g. CO_2 or air pollutants such as VOC, CO, NO_2, SO_2, etc.) is evaluated by means of Equation 4.4. The concentration of the species is denoted by C.

$$\frac{\partial(mC)}{\partial t} = \sum[\dot{m}C]_{bnd} + \dot{C}_{src}$$

Wall

The *Wall* object solves the one-dimensional heat conduction equation through bodies with simplified geometry. In the context of a building simulation, this object is used to solve the thermal interaction between the rooms and between the rooms and the exterior.

$$\rho c_p \frac{\partial T}{\partial t} = \frac{\partial T^2}{\partial^2 x} + \dot{Q}_{src}$$

The material of the wall is defined by means of the physical properties (density, specific heat and thermal conductivity) and other surface properties such as the solar and thermal reflectivity and the thermal emissivity.

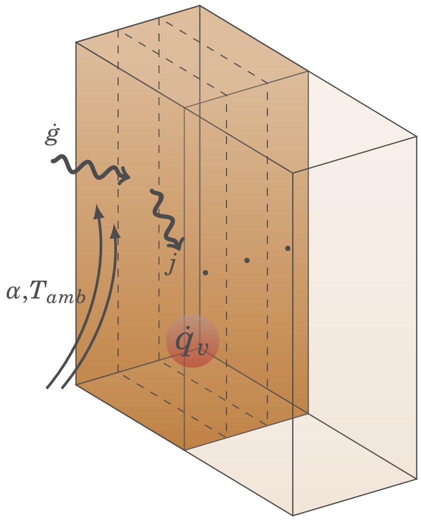


Figure 4.4: *Wall* element.

By default the wall material is assumed to be opaque. However, semi-transparent materials can also be considered only changing the optical properties of the *Wall* element (absorptivity, reflectivity and transmissivity). In this fashion, the amount of solar irradiation that reaches the interior of the building can be evaluated (i.e. in the floor or the walls of a room). The solar energy absorbed by the wall is introduced into the heat conduction equation as a source term.

Furthermore, the NEST platform enables to arrange several objects as they were a single element. These type of elements are called sub-systems. Hence, several walls of different materials can be packed into a unique *CompositeWall* object. This is an appealing property of NEST, since it enables to deal with more realistic walls as they were simple *Wall* elements. The same concept is exploited for simulating multilayer glasses with inner air chambers (e.g. double/triple pane).

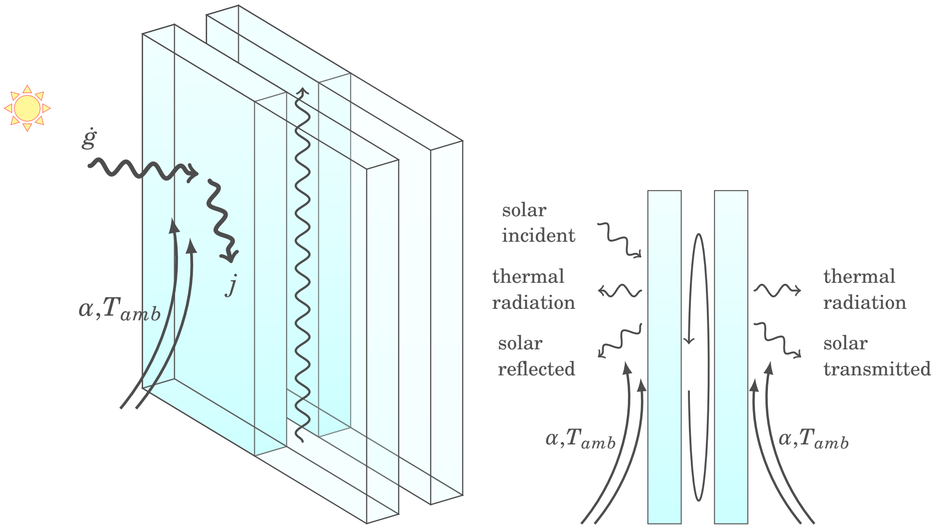


Figure 4.5: Multi-glass element.

RadBox

The *RadBox* object (radiative box) is used for solving the thermal radiation interaction between the surfaces of a room (i.e. walls and glazed windows). Hence, each room should be provided with a *RadBox* element. Every surface of the room gives its temperature and its emissivity to the *RadBox*. Then, it calculates the view factors in order to evaluate the thermal radiation corresponding to each surface by means of the Net Radiation Method [27]. The element is able to calculate the view factors of non-cubic geometries, considering shadows and the windows in the walls.

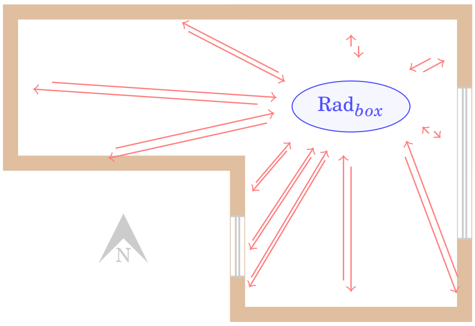


Figure 4.6: Conceptual scheme of the *RadBox* model.

FSolD

The *FSolD* object, which stands for Fictitious Solar radiation Distributor, takes the net transmitted solar radiation from the glazed windows to set the total radiation per unit area. The aim of this object is to distribute this heat input between the floor and the walls of the room. Similar to *RadBox*, the room are complemented with an *FSolD* object when there are glazed windows.

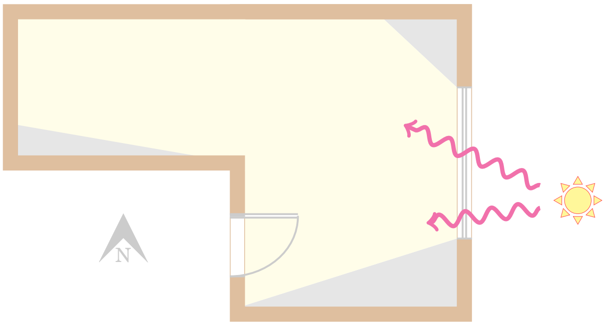


Figure 4.7: Conceptual scheme of the *FSolD* model.

Outdoor

The *Outdoor* object establishes weather conditions to the simulation of the building. It provides ambient conditions (temperature, pressure, moisture, CO_2 concentration, etc.), wind speed and solar irradiation on the facades. Moreover, it computes the thermal radiation heat of the exterior surfaces. This is done assuming the sky is a black body at T_{sky} temperature as $\dot{q}_{rad} = \epsilon\sigma(T^4_{wall} - T^4_{sky})$, where $T_{sky} = 0.0552(T_{amb})^{3/2}$ [28].

The meteorological data is obtained using Meteonorm software [29]. Any object that is in contact with the outside of the building should be connected to an *Outdoor* element, for example, an external wall or the glass of a window. The data are obtained for a certain period, a specific location and the orientation and the slope of the involved surfaces.

Occupant

The NEST buildings program includes an *Occupant* object to take into account the heat supply due to the people presence. The heat source from the people has a significant role in the thermal balance of a room or a building space. That heat is generated by the human body by oxidation. The amount of heat that a person delivers to the ambient depends on the metabolic rate. It is dissipated through the surface of the body and the respiratory tract as a combination of radiation, convection and evaporation.

The program is ready to simulate the dynamic behavior of the occupants. That is, during the simulation the occupants can change their state. For example, they can sleep, they can stay active (reading, eating, etc.) or they can have a shower, too. The interaction of each occupant with the building depends on his current activity. In that manner, for instance, the amount of heat and CO_2 that a user delivers to the environment may vary over time. In Tables 4.2 and 4.3 is reported the occupancy influence by activity.

Activity	**Metabolic rate (W)**
Sleeping	83
Seated relaxed	104
Standing at rest	126
Sedentary activity (office, studying...)	126
Domestic work - shaving, washing and dressing	180
Walking on the level, 2 km/h	198
Standing, medium activity (domestic work...)	209
Cooking, Washing dishes standing	261
Domestic work - raking leaves on the lawn	306
Domestic work - washing by hand and ironing (120-220 W)	306
Gymnastics	574

Table 4.2: Metabolic rate of different activities usually held at home.

Activity	**CO_2 emission** (m^3/h)
Sleep	0.013
Resting or low activity work	0.02
Normal activity work	0.08 - 0.13
Hard activity work	0.33 - 0.38

Table 4.3: CO_2 generation rate per person.

Obviously, the occupants can move to other rooms or they can leave the house and come back later. In addition, they can create events (open doors or windows, switch on ventilation systems, cooking, etc.) that cause changes in the apartment. Moreover, new occupants are also welcome in the simulation. This dynamic behavior is achieved by means of schedule databases. The program is continuously listening these databases for detecting any change in the rooms. You may take Figures 4.10 or 4.14 form Section 4.2 as examples.

Radiator

Radiators object are used in the NEST buildings program for heating spaces. The *Radiators* can be installed in any room. The heat power is obtained from a boiler, which is installed somewhere in the building.

The boiler operation is based on the temperature of a unique room of the dwelling. It starts when the temperature of that room is below the set point value. Actually, there is a dead band to avoid a boiler failure due to too many starts and stops.

The boiler gives the same mass flow rate to all the radiators. It takes some minutes to warm up. During this period, the temperature at the entrance of the radiators grows linearly with the time. The heat output of a radiator is evaluated as $\dot{Q} = c_p(T_{wi} - T_{wo})$. In this equation, T_{wi} is the inlet water temperature and T_{wo} is the outlet water temperature. While the inlet temperature is provided by the boiler the outlet temperature is an unknown that depends on the type of radiator (its $\dot{Q}_{50}$ [30]) and the ambient and inlet temperatures.

When a radiator uses a thermostatic valve (TRV) the water mass flow that enters into the radiator depends linearly on the rooms temperature. This function permits to completely cut off the mass flow. In this way the heating system is controlled locally instead of from a unique centralized room.

4.1.2 Global resolution algorithm

The resolution scheme of the overall model is based on a block-Jacobi like method. During this procedure, every element is visited to call the resolution routines of their particular sub-models. After this, each element changes its state (e.g. room temperature, CO_2 concentration, doors and windows mass flows, radiators heat output, etc.). Once all the elements have been iterated, the program synchronize their input/output data in order to maintain them up to date with the state of their neighboring elements (i.e. new mass flows, rooms temperature, ambient temperature, solar irradiation, etc.). This is repeated several times until the elements are well coupled so that they do not change their state. At this point the whole set of elements is converged to an instantaneous building state. Then, the program checks for changes in the schedules of the openings and the occupants. Finally, the simulation ends or it continues towards the next time step. This scheme is summarized in Figure 4.8.

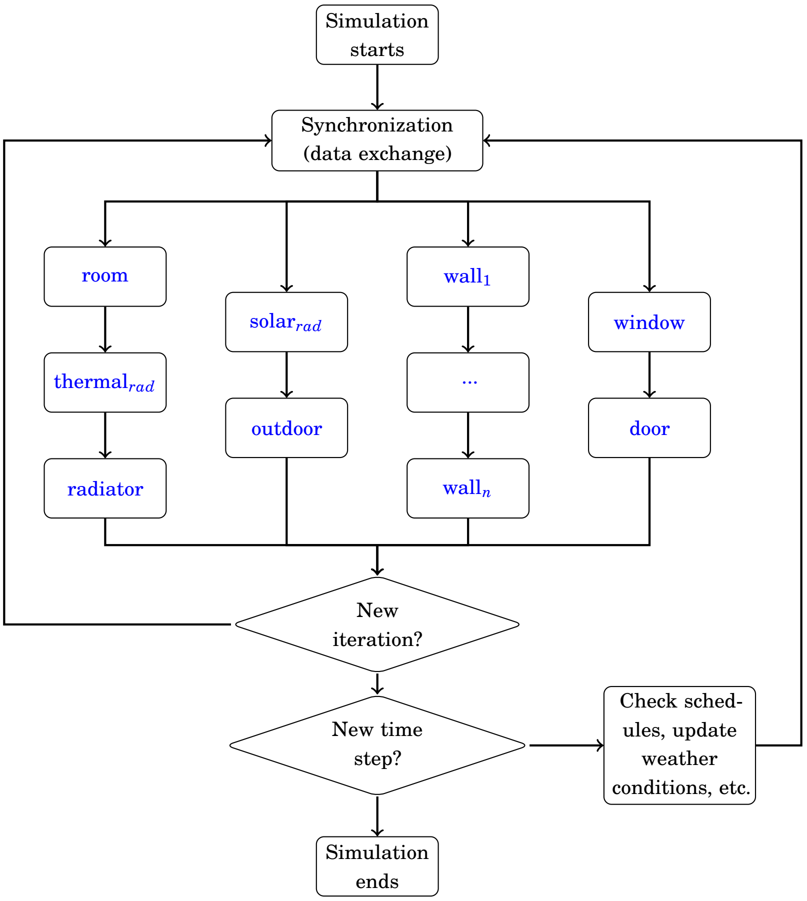


Figure 4.8: Numerical scheme for parallel resolving the hypothetical room depicted in Figure 4.1.

4.2 Simulation of an apartment

This section is aimed to show the effect of the human presence on the thermal performance and air conditions of an apartment. The design of this apartment, shown in Figure 4.9, is intentionally simple to do numerical experimentation and to show some illustrative results obtained using NEST buildings software. It is assumed that only one person is living in this apartment. Hence, there is only the essential for one tenant:

a livingroom, a bedroom and a bathroom. The apartment is specially customized for testing ventilation. The rooms are connected to each other through doors and there is a window in each room.

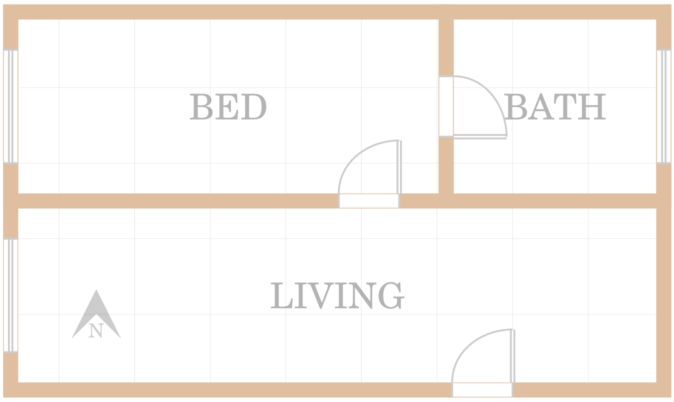


Figure 4.9: Design of the apartment.

Description of the numerical experiments

Five different numerical experiments are presented below. The first experiment is the simplest one. Note that only the tenant is present during the simulation. The schedule of the tenant is presented in Figure 4.10. The simulation is performed along several weeks until weekly steady conditions are achieved. For ease of comparison, the weather conditions have been assumed identical every day. Both, the ambient temperature and the solar irradiation on the west facade are calculated using periodic functions. The temperature varies from 13C (during the night) to 23C (at noon), whereas the solar irradiation ranges from 0 to 800 W/m^2. In particular, the bathroom is connected to an inner courtyard, which is completely shadowed by the block itself. Hence, the solar incidence is not considered on the east side of the apartment. Instead, the courtyard temperature is assumed to be the same as the west side ambient. All the outdoor objects in the NEST model are set to constant pressure and CO_2 concentration (1 atm and 400ppm). The hallway is assumed to be at constant temperature, CO_2 concentration and pressure (20C, 400ppm and 1 atm). Symmetry conditions are taken for resolving the energy equation on the floor and the ceiling boundaries. In regards to building materials, all the walls are made of air brick whereas the floor and the ceiling are made of concrete. All the windows are glazed so that there is solar incidence on the inner floor and the walls. The properties of these materials are available in Table 4.7. The remain cases, which are based on the first, introduce different new inputs to the simulation. All the numerical experiments can be compared in the summary Table 4.4.

Case num.	Desctiption
1	This is the reference case. The schedule of this case is used in the remain cases with certain differences.
2	The bathrooms fan is switched on during the shower (from 7:30 to 8:00).
3	The tenant stays at home on Thursday. In this instance, it is active from 8:00 to 23:00 in the livingroom.
4	The tenan is still at home on Thursday, this time with two guests. They spend all the time in the livingroom. While the tenant is active from 8:00 to 23:00, the guests stay active from 14:00 to 18:00.
5	Same as the previous case. However, the livingroom and the bethroom windows are opened to ventilate from 14:00 to 18:00. The living/bedroom door is also opened.

Table 4.4: Description of the different numerical experiments

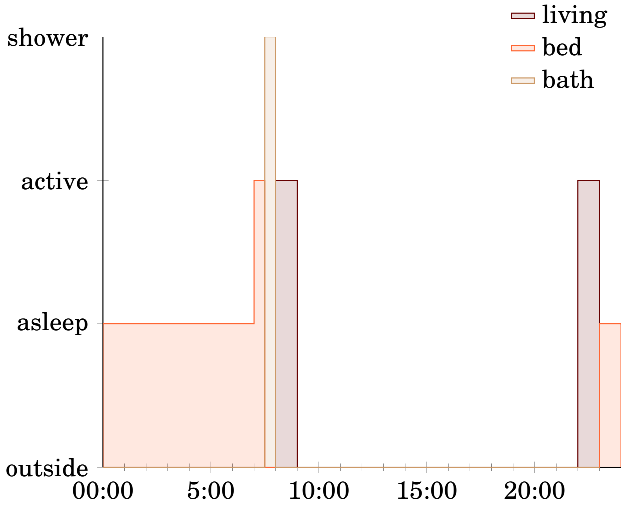


Figure 4.10: Daily occupancy schedule. The tenant wakes up at 07:00. He spends half an hour active in the bedroom and moves later to the bathroom to have a shower. Then he leaves the apartment until 22:00. He stays active in the livingroom during one hour and finally, goes to sleep at 23:00.

4.2.1 Case 1: A reference case

The numerical data corresponding to the first experiment, presented in Figure 4.11, shows a periodic behavior. This is in accordance with the assumed weather conditions and the imposed occupancy schedule shown in Figure 4.10.

During all the week, the livingroom and the bedroom are warmer than the bathroom. This is because these rooms are exposed to the solar radiation (west side). Actually, the solar peak at 12:00 is reflected quasi instantaneously in the west rooms temperature. The occupant is not in the apartment at that hour. Therefore, solar incidence is the only reason that explains the temperature peaks in these rooms. The bedroom temperature is the higher one because this room is smaller than the livingroom. Moreover, the occupant spends all the night in that room (sleeping), which also contributes to hold its temperature.

After waking up, the tenant spends some time in the bedroom. While this action has a slight effect on the temperature, the effect on the CO_2 is clearly appreciated. The daily shower is affecting the bathroom temperature and the CO_2 concentration, specially the last one. In a similar way, the presence of the occupant in the livingroom is also appreciated in the charts. It is worth mentioning that the response of the temperature and the CO_2 concentration to the tenant presence depends on his activity. Among the usages analyzed in this experiment, having a shower is the one that produces major impact on the state of the house.

Furthermore, the temperature of the bathroom oscillates in a similar way as the ambient temperature. Unlike solar irradiation, heat convection on the facade is a slower process. This, added to the inertia of the building materials, leads to a few hours shifting in the evolution of the bathroom temperature. The temperature of the livingroom and the bedroom show different evolution, first because the tenant spends more time active in those rooms and second due to the effect of the solar incidence.

The remain charts show the air changes per hour[3] at every opening. Here you should see that, regardless the position of the occupant, any action that he performs is reflected in the plotted variables. This shows the thermal and fluid dynamic coupling between all the rooms in the apartment system. Most of the events that happen during the day have very little impact on the air changes but the shower. The magnitude of the calculated air changes is close to the usual values considered in the ventilation guideline standards [31, 32].

[3] The air changes per hour is the ratio of fresh air entering to the room and its volume. It is commonly used to get an idea of the ventilation need in buildings.

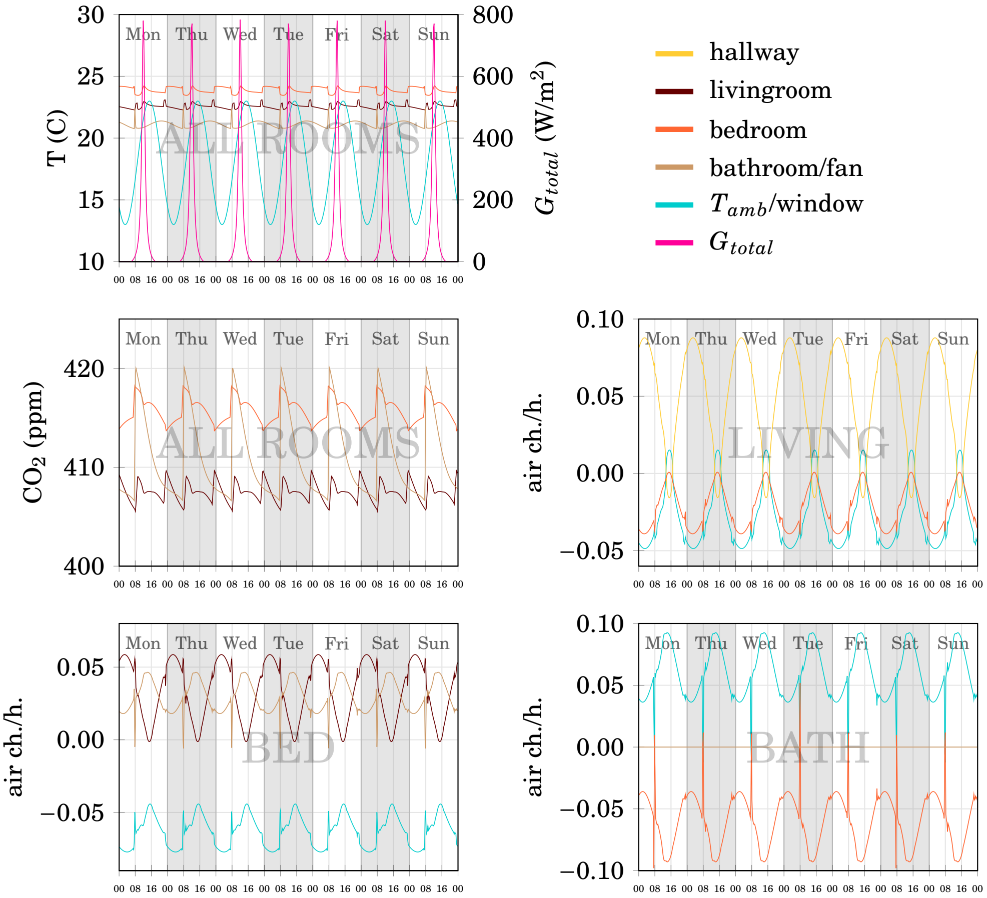


Figure 4.11: Case 1: rooms temperature, CO$_2$ concentration and air changes per hour.

4.2.2 Case 2: Analysis of the bathroom fan

In this instance the bathroom fan is on during the mornings shower. The fan pressure drop is set to 50 Pa. This leads to some changes in the weekly numerical data presented in Figure 4.12. On the one hand, the temperature evolution of the different rooms is almost the same as case 1. This time, the temperature peaks in the bathroom at shower time have been reduced. The temperature in the other rooms is affected in a similar way. On the other hand, the CO$_2$ concentrations in all rooms decreases significantly, specially in the bathroom. The livingroom is more distant from the bathroom and hence, the effect of the fan is not so important.

With respect to the air changes per hour, the forced ventilation produces higher air flow rates through all the openings. This time, all windows and doors are closed so that the air flow is due to the infiltrations through the small gaps in their frames. The charts show that the air circulates from the outside of the apartment to the ventilation duct of the fan. Most time, the sum of inflows match the total outflow at every room (some lines overlap in the charts).

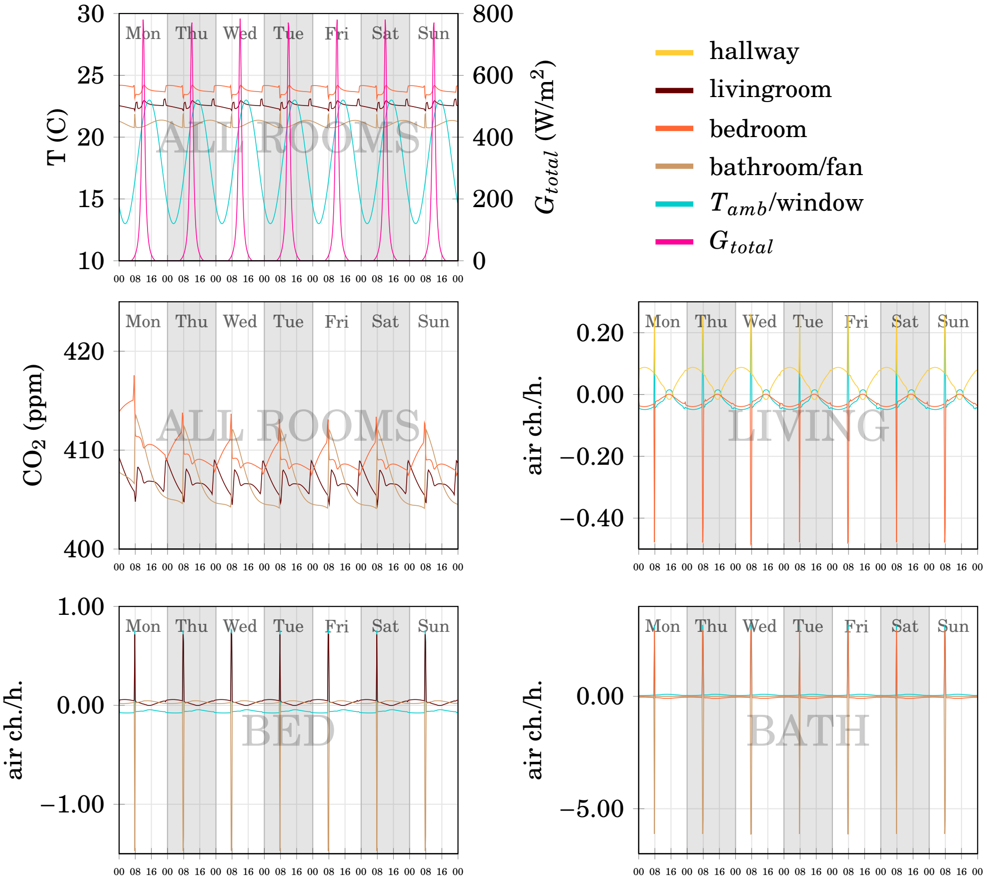


Figure 4.12: Case 2: rooms temperature, CO$_2$ concentration and air changes per hour.

4.2.3 Case 3: Analysis of occupancy activity I. Tenant alone.

Unlike case 1, in this experiment the tenant stays at home on Thursday. The numerical results corresponding to this case are presented in Figure 4.13. Up until Thursday, the evolution of all the analyzed variables is the same as case 1. After this, the presence of an occupant during all the day leads to some changes. First, only the temperature of the livingroom is slightly affected. In contrast, the CO_2 concentration in the livingroom and the bedroom increases significantly until 23:00, when the tenant goes to sleep. The CO_2 in the apartment decreases continuously during the following days (Friday, Saturday and Sunday). Even so, the CO_2 levels from the previous Monday are not recovered on Sunday. On the other side, the air changes per hour do not change considerably. Due to the warming up of the livingroom, there are mass flow fluctuations in this room that lead to changes in the amount of mass.

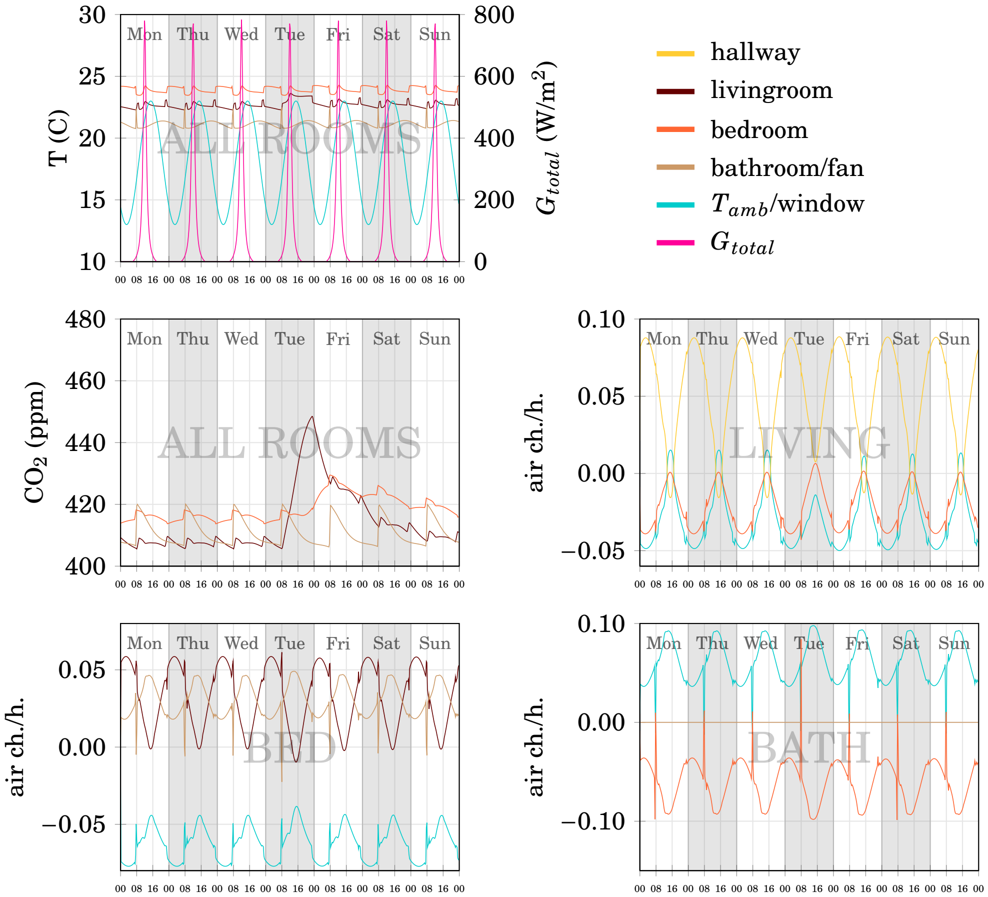


Figure 4.13: Case 3: rooms temperature, CO$_2$ concentration and air changes per hour.

4.2.4 Case 4: Analysis of occupancy activity II. Receiving guests

The behavior of the tenant in this experiment is the same as case 3. For this reason, the comments in that case are also valid here. The difference is in the number of active people in the livingroom on Thursday. Now, there are two guests in the apartment from 14:00 to 18:00. The presence of more people increases the temperature over the values obtained in case 3, basically during the visit. It is worthwhile to say that the livingroom is the biggest room in the apartment. Hence, more heat supply is necessary to increase its air temperature. On the other side, the evolution of the CO_2 is different in comparison with case 3. The concentration level reaches up to 470ppm whereas the one in the previous case was 450ppm. Although the CO_2 peak occurs at the same time in both cases (23:00), now the rate of growing changes at 18:00, when the guests leave the apartment. This is appreciated in the CO_2 chart in Figure 4.15, the peak is more rounded now. Finally, the presence of the other occupants in the house produce some fluctuations in the air changes per hour, but they are just infiltrations and they do not contribute to ventilation.

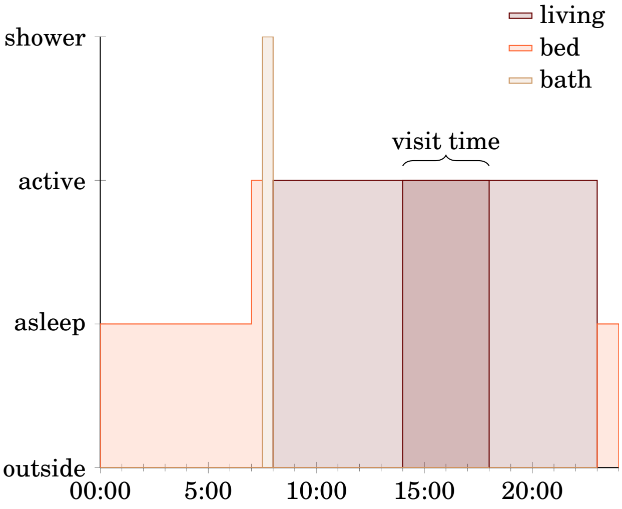


Figure 4.14: Thursday occupancy schedule. The tenant wakes up at 07:00. He spends half an hour active in the bedroom and moves later to the bathroom to have a shower. Unlike the remain days, her stays in the livingroom from 9:00 to 23:00. At 14:00 two more people enter into the apartment. Finally, the tenant goes to sleep at 23:00.

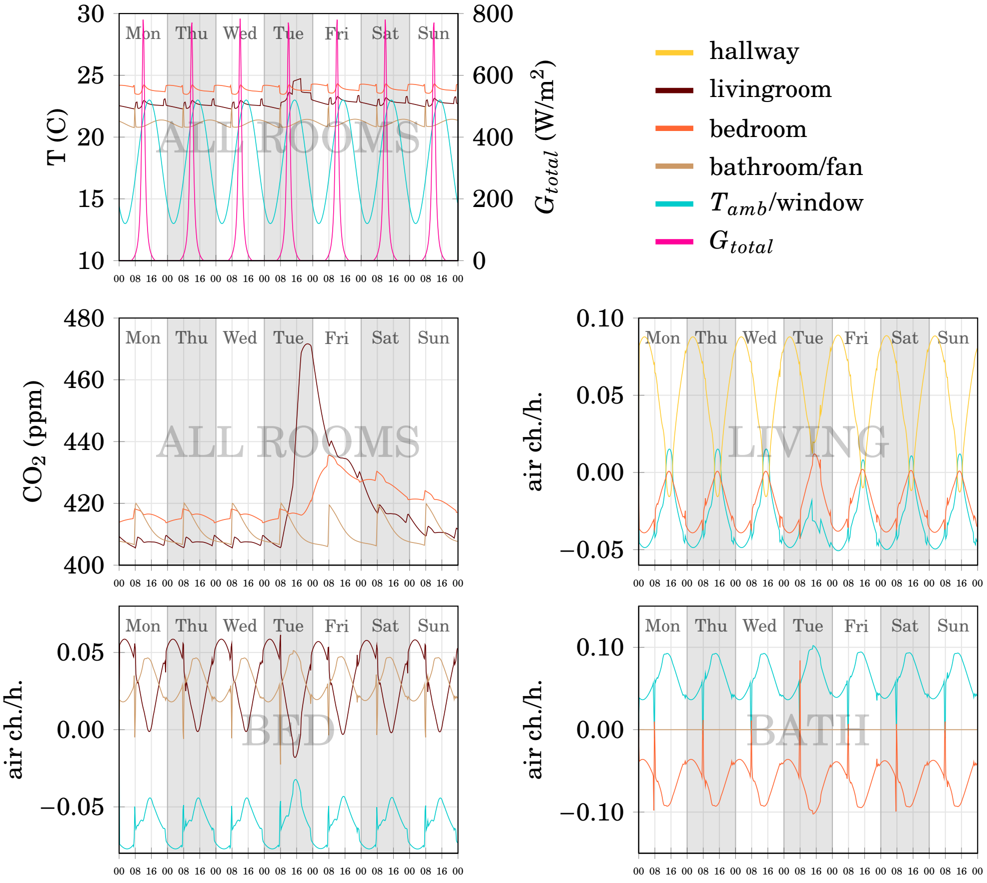


Figure 4.15: Case 4: rooms temperature, CO$_2$ concentration and air changes per hour.

4.2.5 Case 5: Analysis of ventilation using the windows

This is the last numerical experiment. The simulation events are the same than the previous case 4. However, this time the occupants decide to open the livingroom and the bedrooms windows during the visit (from 14:00 to 18:00). For a good ventilation, the door connecting these rooms is also opened. While the effect on the temperature is not really important, the ventilation alters completely the evolution of the CO$_2$ in the apartment. Moreover, the air changes per hour are also altered in the livingroom and the bedroom. In contrast, in the bathroom there are only infiltrations since the door remains closed. The air circulation during the ventilation is illustrated in Figure 4.16.

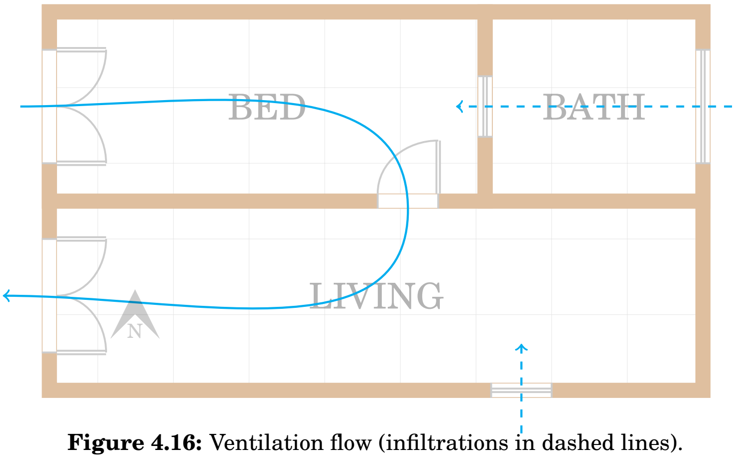


Figure 4.16: Ventilation flow (infiltrations in dashed lines).

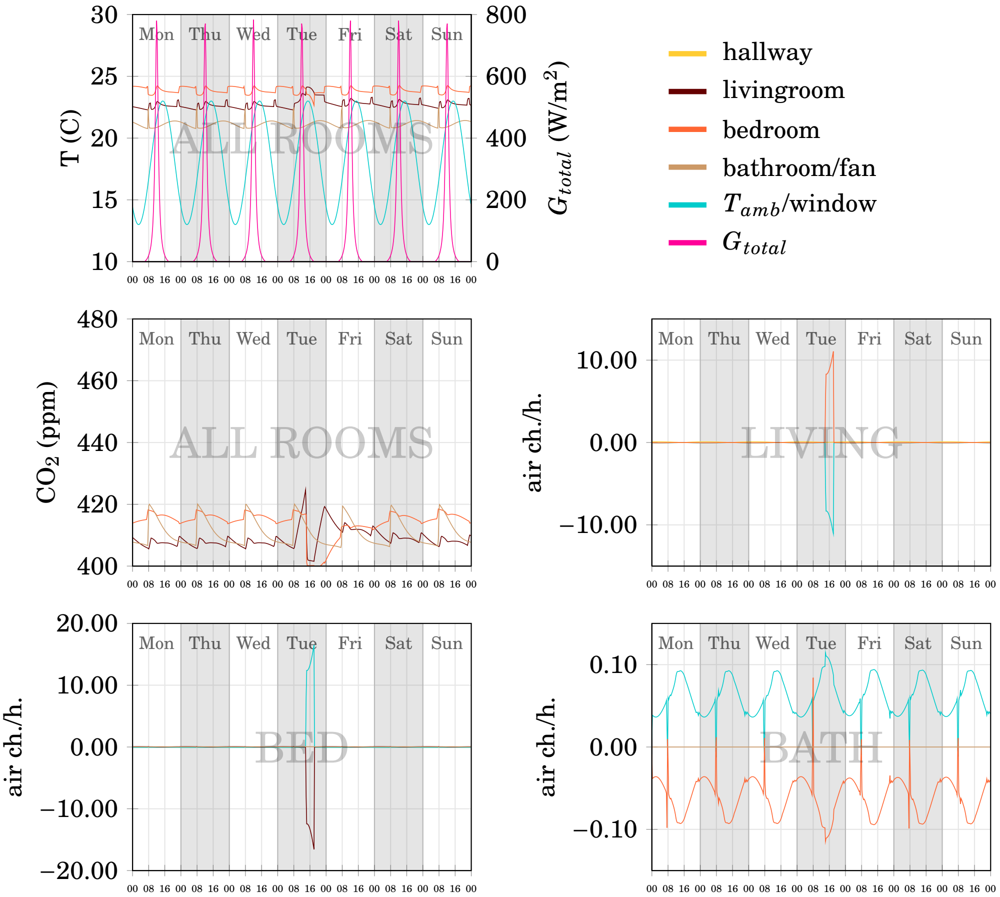


Figure 4.17: Case 5: rooms temperature, CO$_2$ concentration and air changes per hour.

4.3 Simulation of a semi-detached house

The thermal performance of a semi-detached house is analyzed below. This is a real dwelling located in Lemmer, in the Netherlands[4]. In Figure 4.18 are shown the drawings of the house. The drawings are symmetric because they depict two semi-detached dwellings. Note that the study is just focused on the right side of the whole building. The remain three sides of the house are exposed to the ambient. These facades receive

[4] Address: Kleine Beer 28 (52:51:2.96N, 5:42:44.91E)

sun radiation directly since there are not significant shadowing objects (trees, tall buildings, etc.). The building has three floors. In the ground floor there is the livingroom, which is connected to the kitchen, a scullery, the garage and one bathroom. At the east side there are the stairs for accessing to the first floor. In that floor, there are three bedrooms and one bathroom. The last floor, the attic, is an open space without interior walls, whose ceiling is the roof of the building. In Table 4.5 is provided a brief description of every room in the drawings.

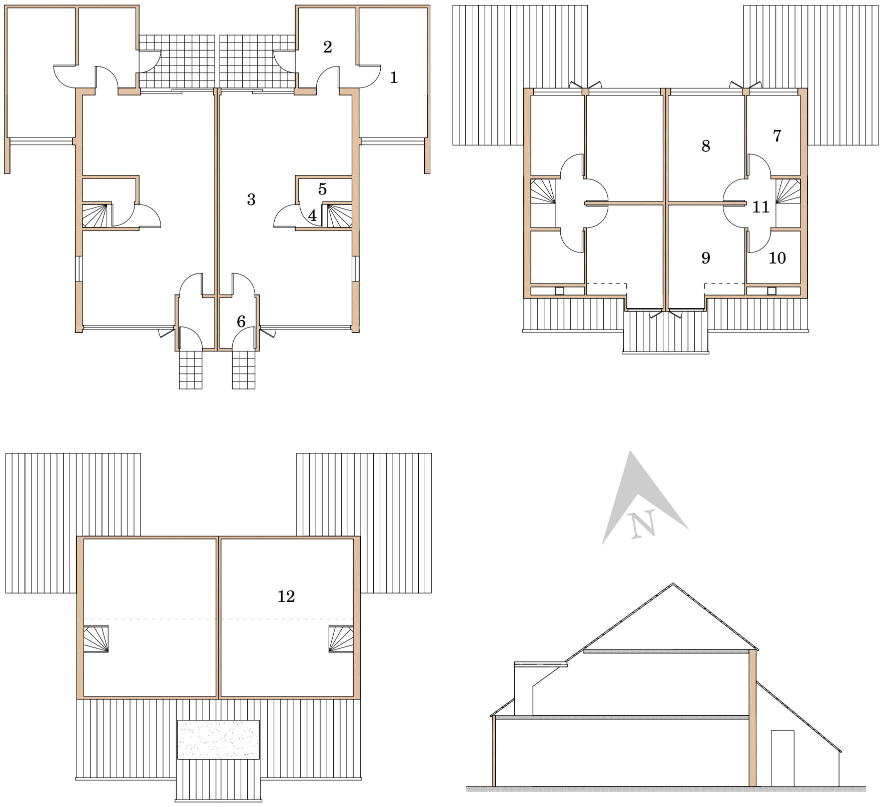


Figure 4.18: House drawings. From top to bottom and left to right: ground, first and attic front views and building lateral view. The drawing scale is 1:42.

Room	Usage	Heating	Windows
1	Garage	no heating	no windows
2	Scullery	1-plate radiator	no window
3	Livingroom & kitchen	- 3-plate radiator below front window - 2-plate radiator on west wall close to sliding doors	- double pane window at front (2.6,1.85 m) - double pane window at east (0.75x1.05 m) - sliding double pane doors back side (2.4x2.4 m)
4	Downstairs hallway	no heating	no windows
5	Toilet	no heating	no windows
6	Front door hallway	2-plate radiator on east wall	no windows
7	Bedroom/office	1-plate radiator on north wall	single pane window (1.9x1.04 m)
8	Master bedroom	1-plate radiator on north wall	single pane window (2.8x1.24 m)
9	Bedroom office	2-plate radiator on south wall	single pane window (1.31x1.04 m)
10	Bathroom	radiator/towel dryer on west wall	double pane window (0.37x0.45 m)
11	Upstairs hallway	no heating	no windows
12	Attic	boiler room, no heating	double pane window (0.98x1.01 m)

Table 4.5: Description of the usage, heating system, ventilation/windows of the different rooms of the house.

Room	usage	radiator type	width height	TRV	type	Q_{50} [W]	n
2	Scullery	1-plate	0.3x2	no	10	820	1.310
3	Livingroom	3-plate	2.75x0.4	yes	33	4825	1.307
	& kitchen	2-plate	1.25x0.6	yes	22	1634	1.328
6	Front-door hallway	2-plate	0.75x0.4	yes	21	741	1.319
7	Bedroom office	1-plate	0.95x0.7	no	10	895	1.303
8	Master bedroom	1-plate	1.12x0.7	yes	10	984	1.303
9	Bedroom office	2-plate	0.95x0.7	yes	20	1307	1.328
10	Bathroom	towel dryer	0.7x1.55	no	-	1206	1.220

Table 4.6: Summary of the different radiators installed within the dwelling.

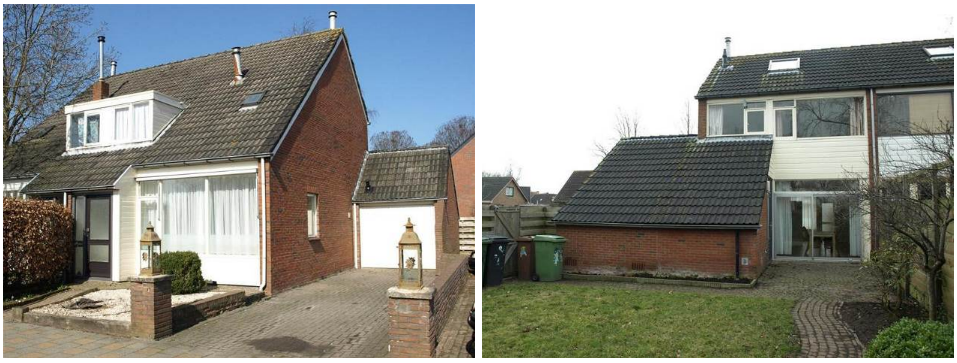


Figure 4.19: At the left side is shown a front view of the house (S-SW). Ground floor: livingroom window (room_3), first floor: bedroom window (room_9) and bath room small roof window (room_{10}). At the right side is shown a back view of the house (N-NE). Ground floor: garage and scullery (brick wall), livingrooms sliding doors, first floor: bedrooms (room_7 and room_8), attic: roof window (room_{12}).

Building materials

The front and back facades of the house are made of timber: south wall and a small part of north wall in room$_3$, a small part of east wall in room$_6$, north wall in room$_8$ and a small part of north wall in room$_7$. Timber frame walls have no brick at the inside.

The remain walls of the house are made of brick: north walls in room$_1$ and room$_2$ and all east walls of the house. All the floors are made of concrete, although the first layer is made of wood on the ground floor, tails on the first and carpet on the attic.

The materials of some other parts of the house are unknown and have been assumed or omitted. These are the inner walls (assumed as brick walls), the finishing wall materials (not considered in the simulation, only the paint in terms of thermal radiation), the garage and the scullery floor (assumed as concrete), and the roof material (assumed as brick too).

Material	λ $[W/mK]$	ρ $[kg/m^3]$	c_p $[J/kgK]$	$\epsilon_{thermal}$	α_{solar}	τ_{solar}
White plastic paint	-	-	-	0.90	-	-
Carpet	-	-	-	0.90	-	-
Air brick	0.450	1786	840	-	-	-
Concrete	1.105	2106	880	-	-	-
Glass	0.917	2700	800	0.96	0.08	0.86

Table 4.7: Thermal and optical properties of the different building materials used in the house [33–35].

Numerical modeling of the house

The semi-detached house is modeled using the aforementioned collection of NEST elements (walls, air room, thermal radiation, solar radiation distributor, single glasses,...) or groups of elements (composite walls and double/triple glasses). In Figure 4.20 is shown a graph view of the NEST building model, based only on two rooms of the first floor for simplicity.

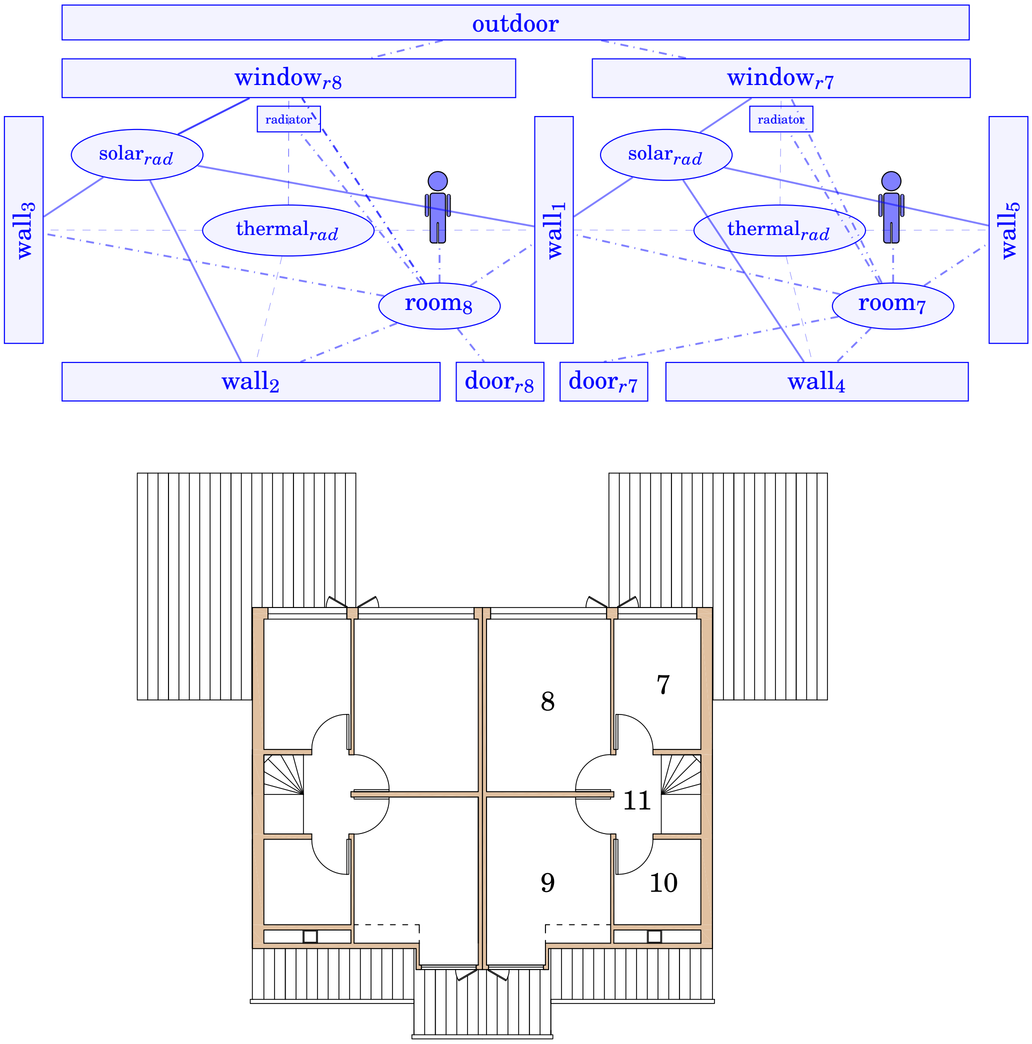


Figure 4.20: First floor front view (below) and NEST building model for rooms 7 and 8. For simplicity, the ceiling and the floor of each room have been removed.

It should be noted that this study is only concerned on the thermal performance of the house and the effect of the heating system. To this end, the whole house has been simulated with and without boiler support. However, the presence of any occupant has been omitted so that no heat, CO_2 or VOC generation are considered. Furthermore, there is no ventilation, either by means of infiltrations or by opening doors and windows. Therefore, the various rooms in the house exchange energy with each other or with the ambient by means of conduction, convection and thermal or solar radiation through the walls and the glasses.

Initial and boundary conditions

The house is initially set to a constant temperature field. This means that the elements conforming the building system (rooms air, walls, window glasses, radiators, etc.) have all of them the same initial temperature. The numerical simulations use real weather conditions obtained using Meteonorm software through an *Outdoor* object. The closest data acquisition station in Meteonorm is located in Eelde (Long: 6.34, Lat: 53.08, Netherlands). The solar irradiation is obtained for vertical and tilted walls oriented to south (azimuth +20), vertical walls oriented to east (azimuth -70) and tilted and vertical walls oriented to north (azimuth -160).

According to the solar irradiation charts shown in the following figures, during autumn and winter months there are more cloudy days than during spring and summer months. However, the days with higher solar irradiation are in February and November. In general, February and November (min -10C max 15C) are the colder months while May and August are warmer (min 5 max 25).

The cloudy days can be identified in the plots if the solar irradiation of the different orientations overlap to each other. In these days the irradiation is diffuse. On sunny days instead, there is always an initial peak of the east side irradiation, which usually is followed by a later secondary and higher peak of the south side irradiation. It should be also mentioned that the irradiation peaks match with ambient temperature peaks, which is depicted in cyan. Finally, note that the west side irradiation is not shown because this is a semi-detached house, and hence symmetry conditions are assumed there.

4.3.1 Case 1: Thermal performance without heating

The family house has been simulated without heating system. Under this assumption, the temperature variations are mainly due to the weather conditions. In that manner, the heating need of the house is highlighted. Moreover, the thermal response of every room to the ambient changes can be identified also.

The simulation has been performed over a whole year. The time step is set to 30 seconds. However, for reducing the amount of data, only one point every 3000 seconds of simulation has been saved. Hence, the resolution of the presented data is approximately one hour.

In the Figures from 4.21 to 4.24 is shown the evolution of the temperature at various rooms of the house. For a better analysis, the numerical data is shown together with the weather data. The solar irradiation is shown as the amount of solar energy incident on the facades. For the sake of brevity, only some months along the year have been shown, in particular: February, May, August and November.

Analysis of the numerical data

Over all the year, the livingroom (room$_3$) is warmer than the remain of the rooms. This is due to the fact that this room is located downstairs so that it has two more floors on it that provides insulation. In February and November the southern rooms (room$_9$ and room$_{10}$) are often warmer than the northern ones (room$_7$ and room$_8$). In contrast to room$_3$, southern rooms are only warmer than the others on sunny days, which emphasizes the benefit of installing windows in these rooms[5].

The temperatures in all the rooms follow the trend of the ambient temperature (cyan color). However, the oscillations are attenuated due to the inertia of the building. The simulation shows that the rooms with big windows are more sensible to solar irradiation. For instance, in the livingroom (room$_3$), which has a big window in the front side (south) and a glassed sliding door in the back side (north), it can be observed several temperature peaks that match with the irradiation peaks on sunny days. The same is observed for room$_9$ (bedroom) and room$_{10}$ (bathroom), which are both south oriented. Instead, the northern bedrooms (room$_7$ and room$_8$) and the attic (room$_{12}$) show smother temperature tendency. This influence is better appreciated in February and November than in May and August. On the other side, the hallway stairs (room$_{11}$) is the space that shows the lower solar irradiation influence since it has no windows and it has few contact with the ambient.

[5] This observation does only make sense in the context of the current simulation. Obviously, construction guidelines are totally different depending on the location of the dwelling.

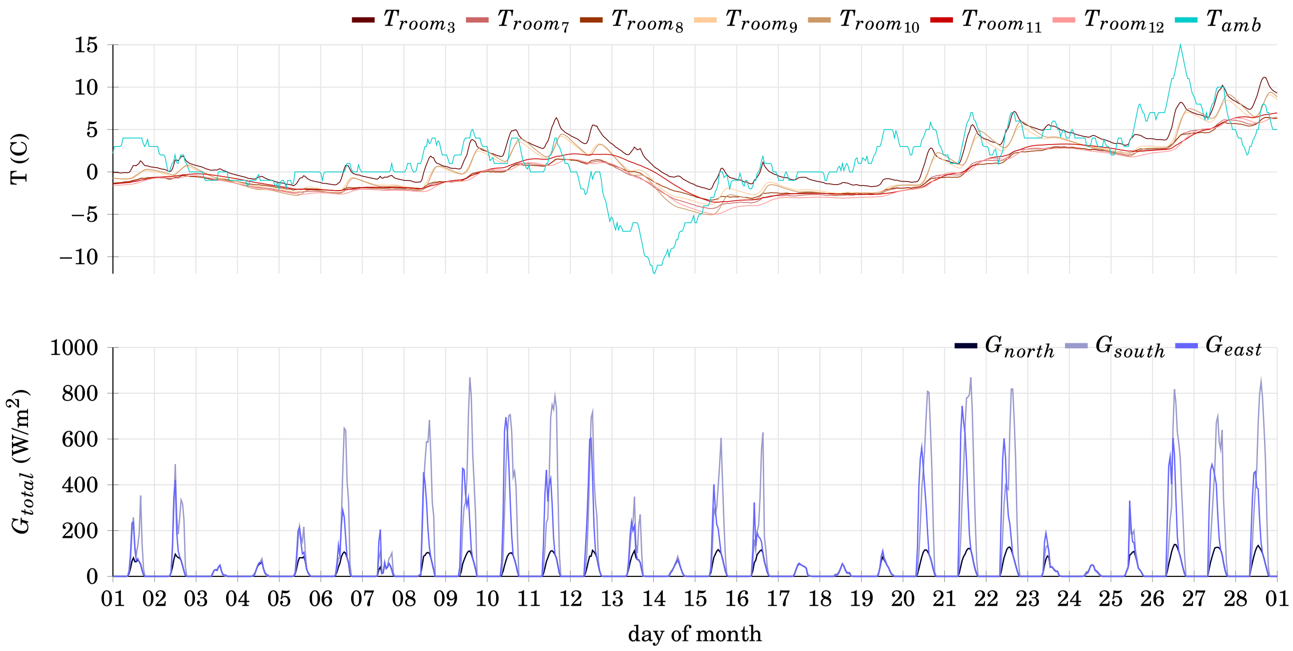


Figure 4.21: Dwelling without radiators: February.

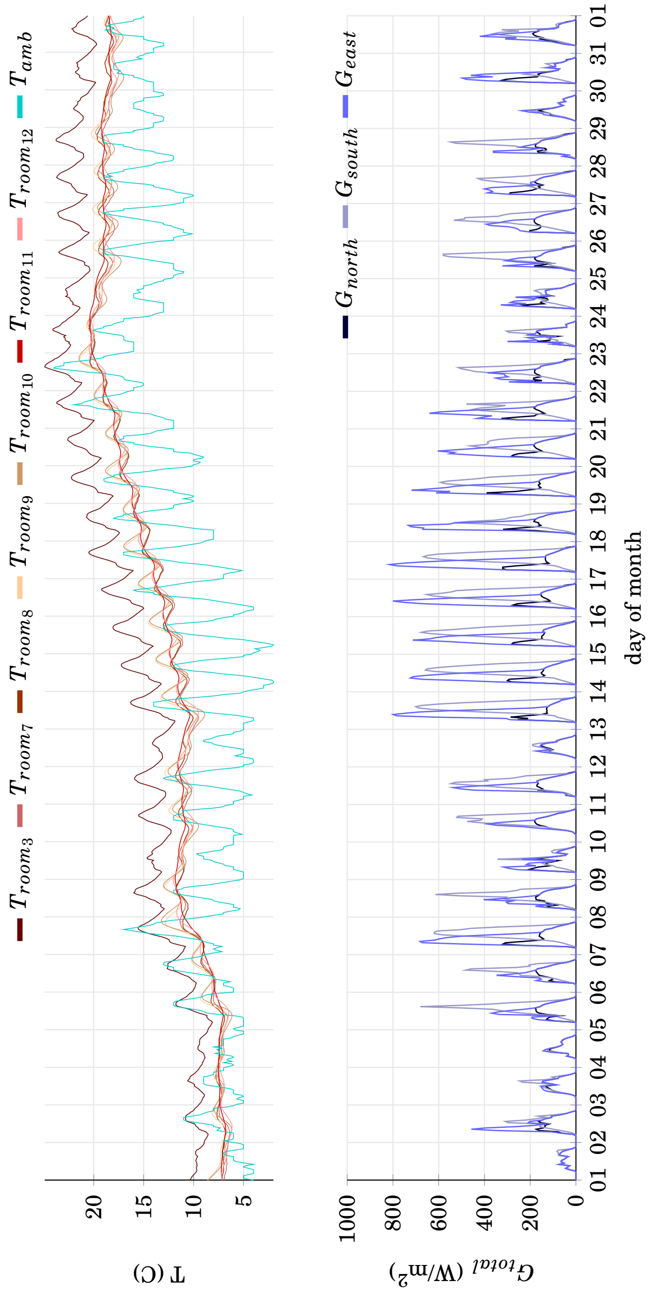


Figure 4.22: Dwelling without radiators: May.

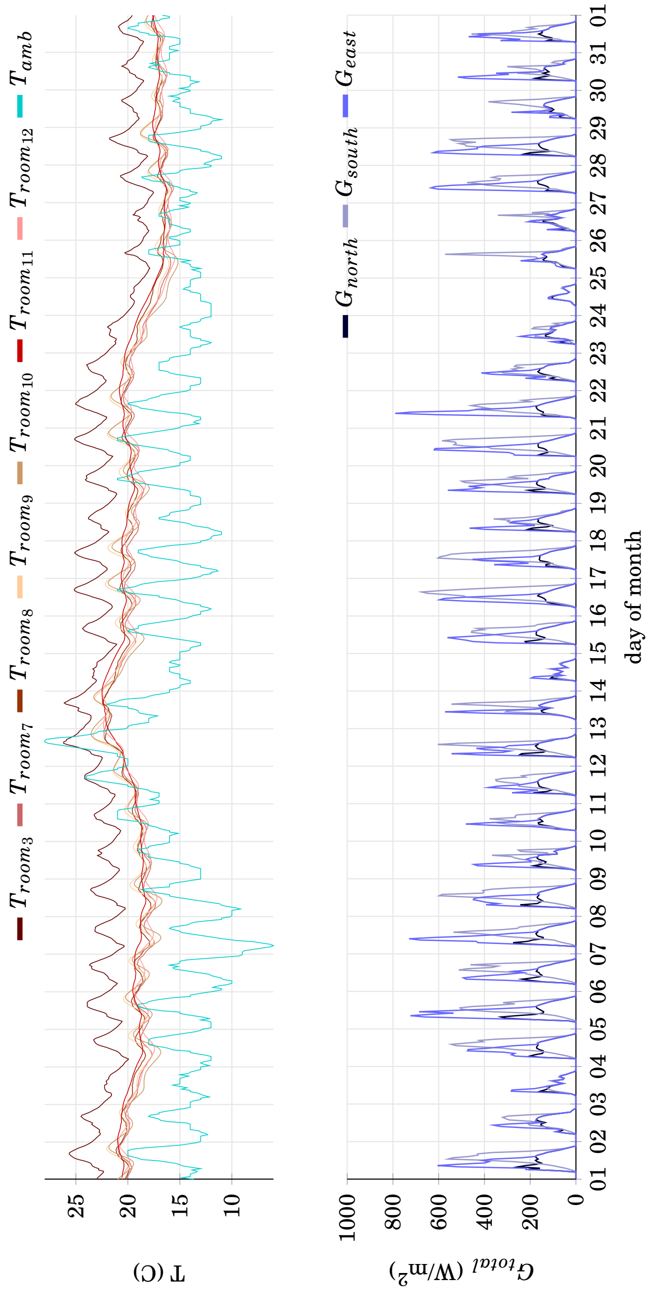


Figure 4.23: Dwelling without radiators: August.

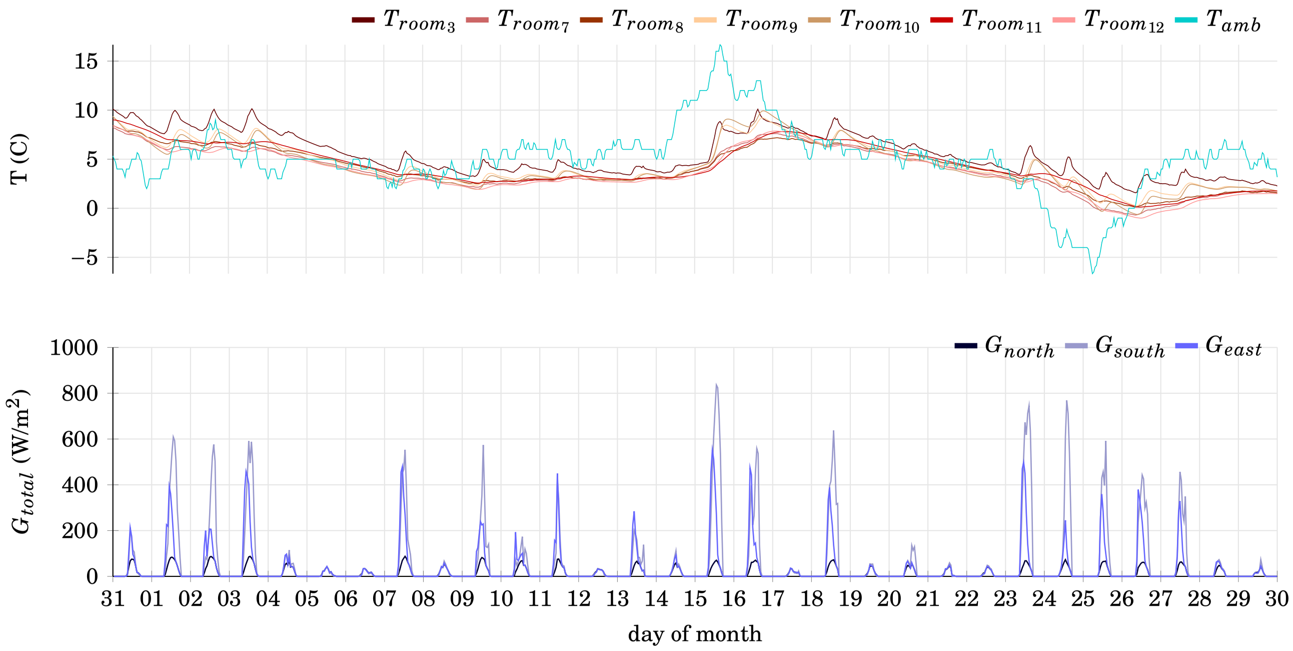


Figure 4.24: Dwelling without radiators: November.

4.3.2 Case 2: Thermal performance activating the boiler

The numerical results shown below are obtained considering the influence of a boiler system that supplies heat to the house. As shown in Table 4.5, the livingroom, the kitchen, the bedrooms, an one bathroom have one or more radiators. The other rooms (the garage, the scullery, the attic and the stairs) do not have radiators. The radiators are simulated according to the standard UNE EN-442. The boiler, which is installed in the attic, is controlled by the livingroom temperature (room$_3$). Therefore, in this simulation, all the radiators are on or off at the same time.

According to the data shown in Figure 4.25, the heating system is on every day from the January to May and from August to December. During the remain months the boiler is activated only few days.

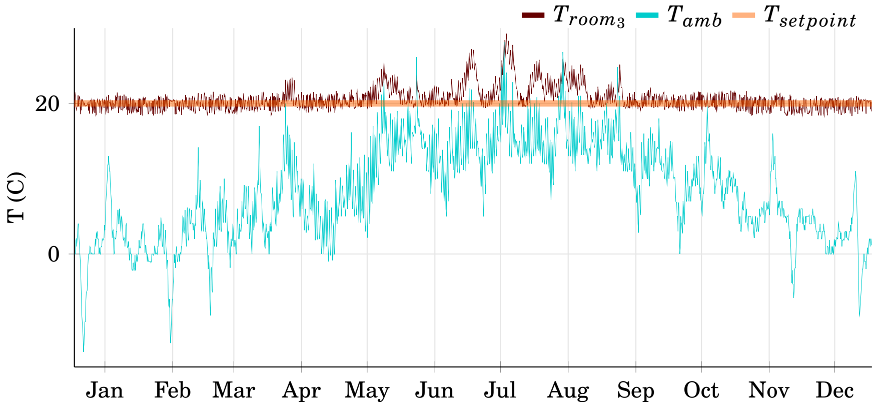


Figure 4.25: Livingroom temperature over the year. The boiler is on when T_{room_3} is below the $T_{setpoint}$. Note that months labels are shown the 15_{th} of every month.

Analysis of the numerical data

The temperature of the southern rooms (3, 9 and 10) is higher than the ones oriented to the north (7 and 8). Within these rooms, the bedroom office (room$_9$) is the warmest and the master bedroom (room$_8$) is the coldest. This is due to their position, the more wall area exposed to the east and south sides the more solar irradiation. Moreover, room$_9$ has a bigger radiator than room$_8$. This difference can be appreciated easily during the coldest months (Figures 4.26 or 4.29).

In any case, the radiators in these rooms might be undersized since the temperature is under the comfort value during the cold periods. In contrast, the bathroom temperature (room$_{10}$) is too high so that the radiator might be oversized. Regardless of the time of the year, the temperature in the rooms without heat supply is lower than the others, as can be observed for the hallway stairs (room$_{11}$) and the attic (room$_{12}$).

In February and November the temperature of the bathroom (room$_{10}$) is the highest one. Instead, in May and August the warmest room is the livingroom (room$_3$). This is in concordance to the previous simulation without the boiler. The livingroom temperature is more influenced by the solar irradiation due to the big windows at the south side.

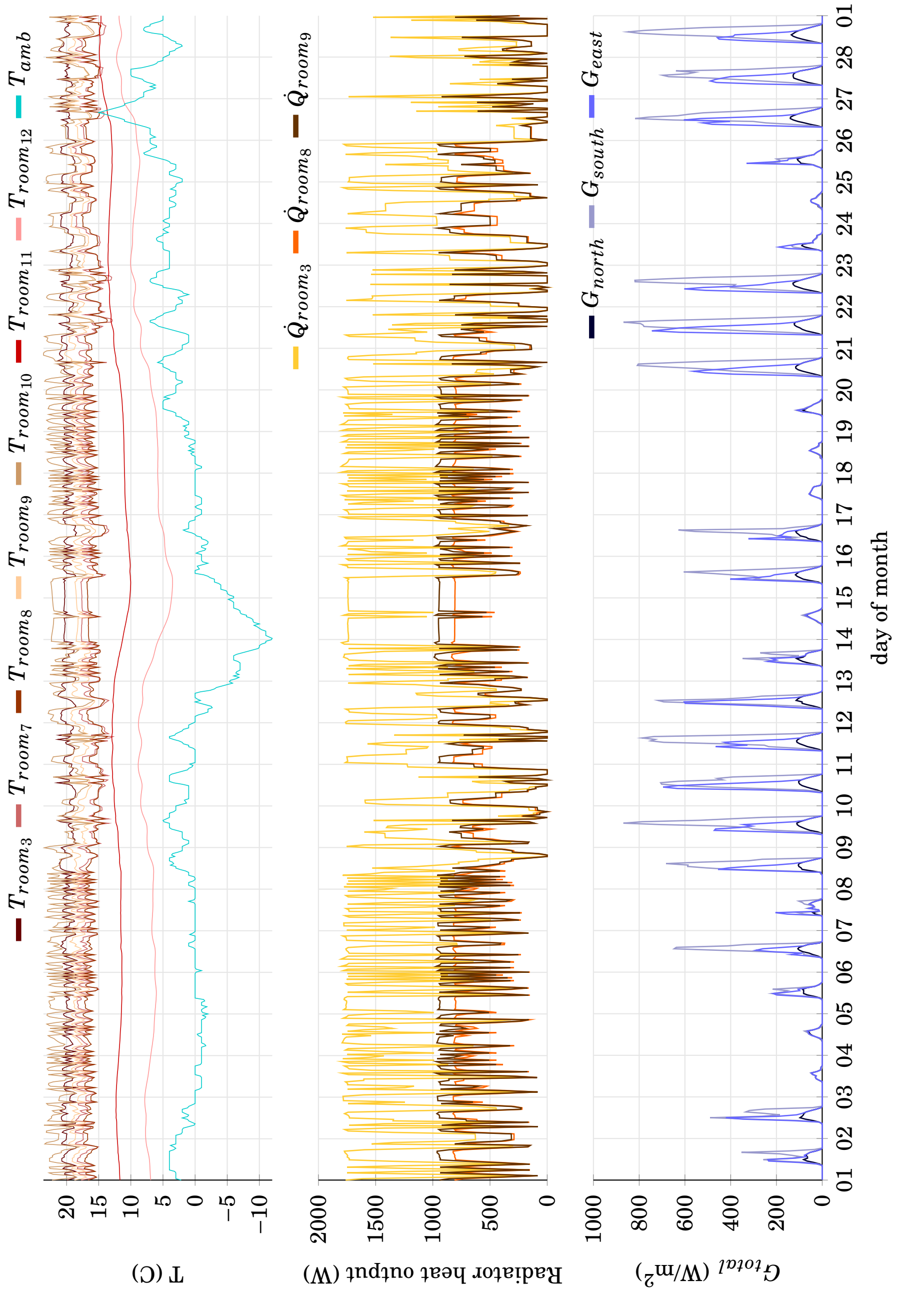


Figure 4.26: Dwelling with radiators in February. Boiler controller is placed at the livingroom (room$_3$).

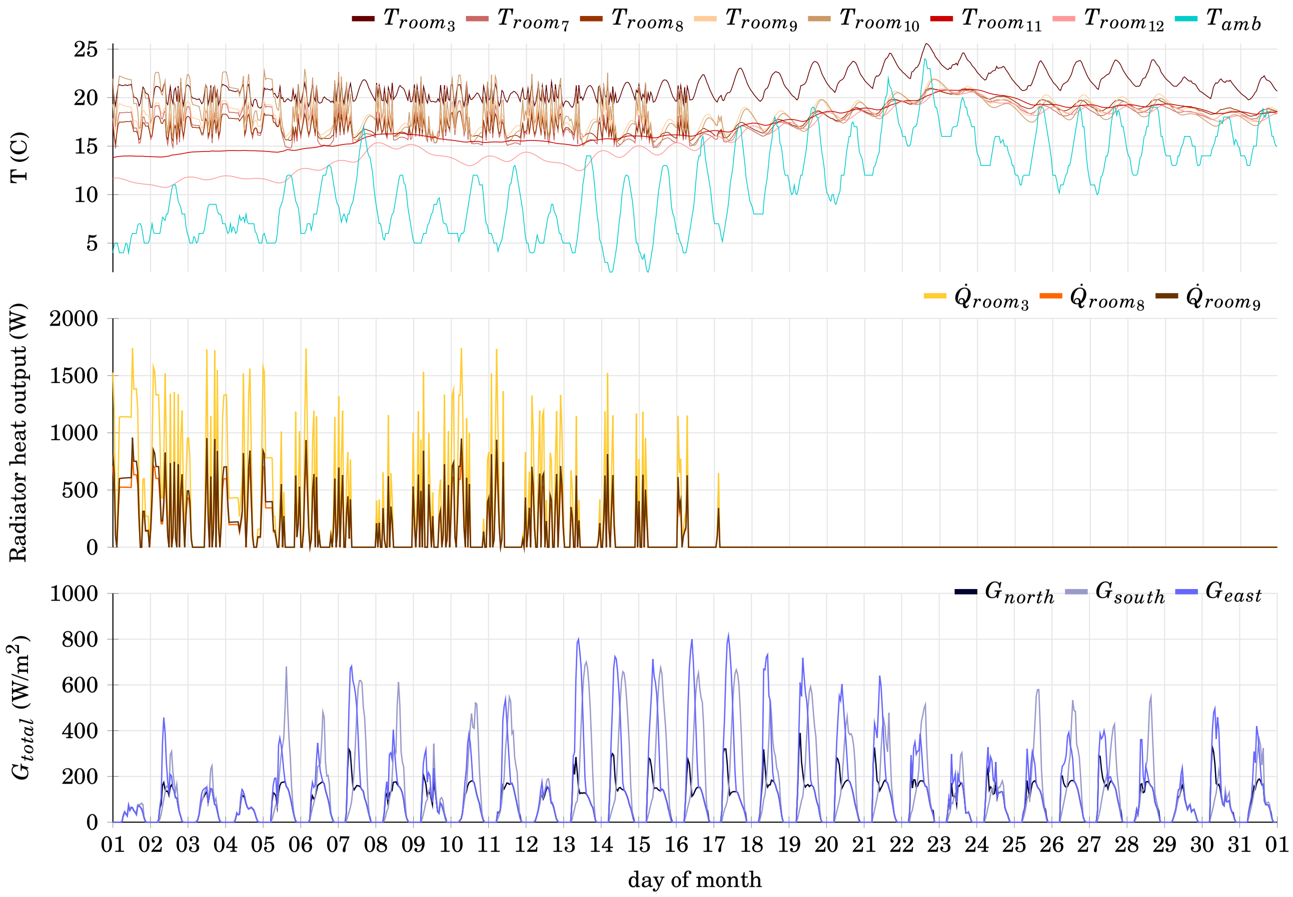


Figure 4.27: Dwelling with radiators in May. Boiler controllers is placed at the livingroom (room$_3$).

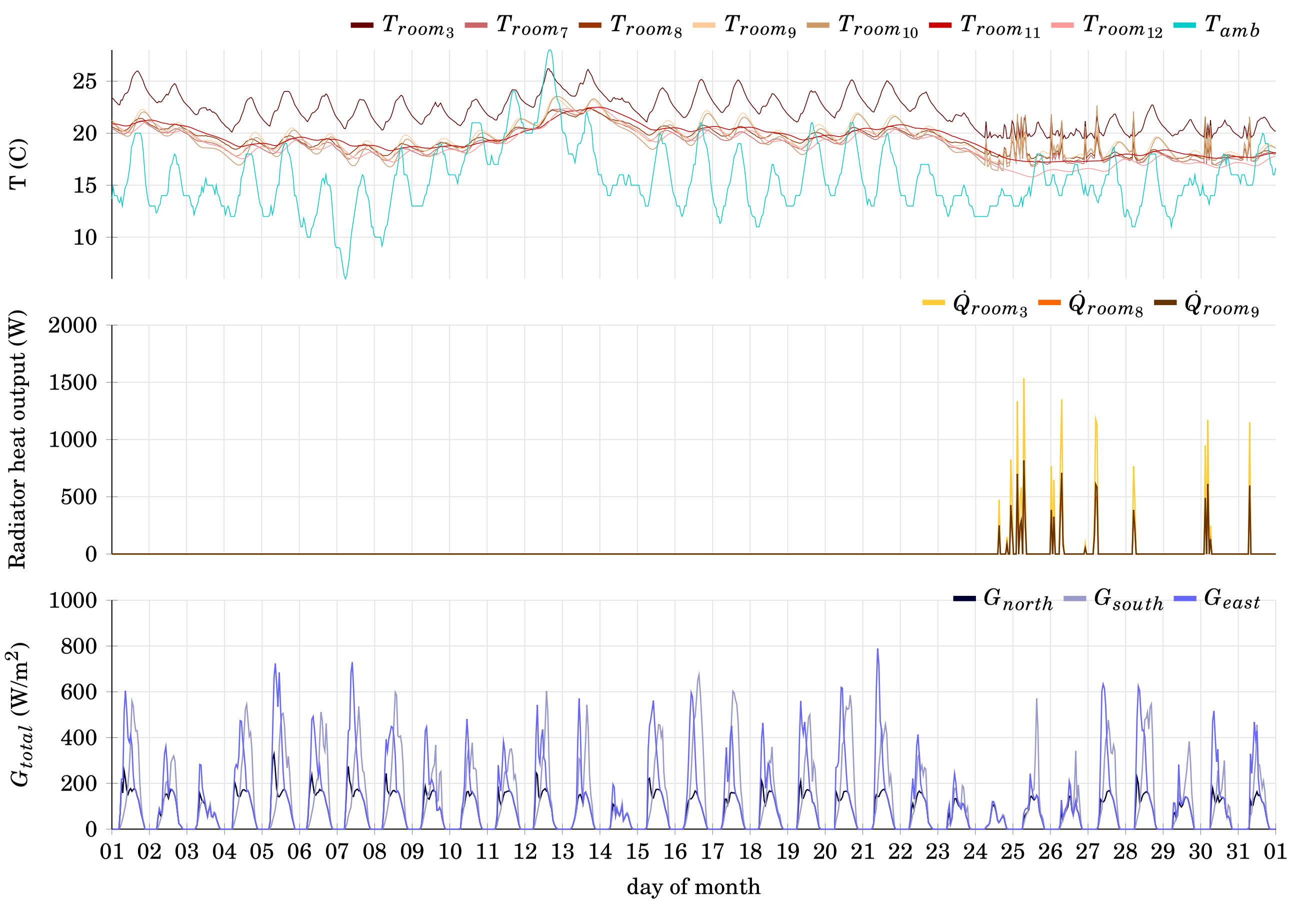


Figure 4.28: Dwelling with radiators in August. Boiler controller is placed at the livingroom (room$_3$).

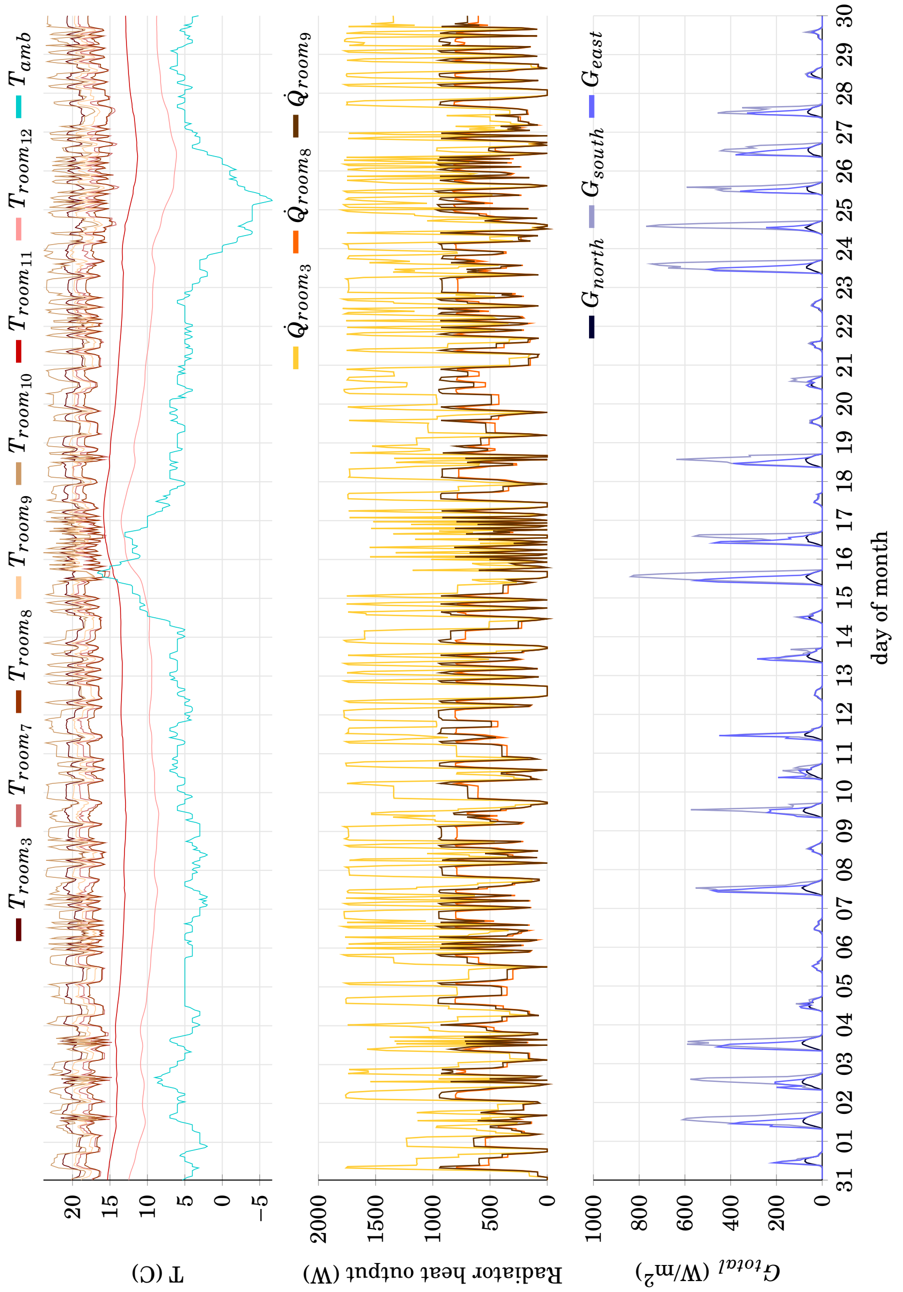


Figure 4.29: Dwelling with radiators in November. Boiler controller is placed at the livingroom (room$_3$).

4.3.3 Case 3: Thermal performance installing TRVs

In order to fix the possible undersized or oversized issues with the radiators, the same house is simulated again using thermostatic valves in some radiators. In particular, the bedroom radiators in the first floor are now enhanced with thermostatic radiator valves devices (TRVs). The boiler system is still controlled using the temperature of the livingroom (room$_3$).

Analysis of the numerical data

In contrast to the previous simulation, the use of TRVs enables to maintain all the heated rooms at a comfort temperature. This is well appreciated in Figures from 4.30 to 4.33. Remember that the hallway stairs (room$_{11}$) and the attic (room$_{12}$) are not heated so they are the coldest spaces. On the other side, room$_{10}$ bedroom is too heated since the temperature is so high during the cold periods.

Furthermore, the radiator in the master bedroom (room$_8$) delivers more heat to the house than the one in the bedroom office (room$_9$). Hence, thanks to the local control, the room$_9$ can now take advantage of its south orientation so that it is heated in a more efficient way. Moreover, the thermal performance of the first floor produces some changes in the radiator installed in the livingroom (room$_3$). In this instance, this radiator does not deliver so much heat as it does when the first floor rooms do not use TRVs, specially at sunny and warm days (see numerical results in Figures 4.26 to 4.29).

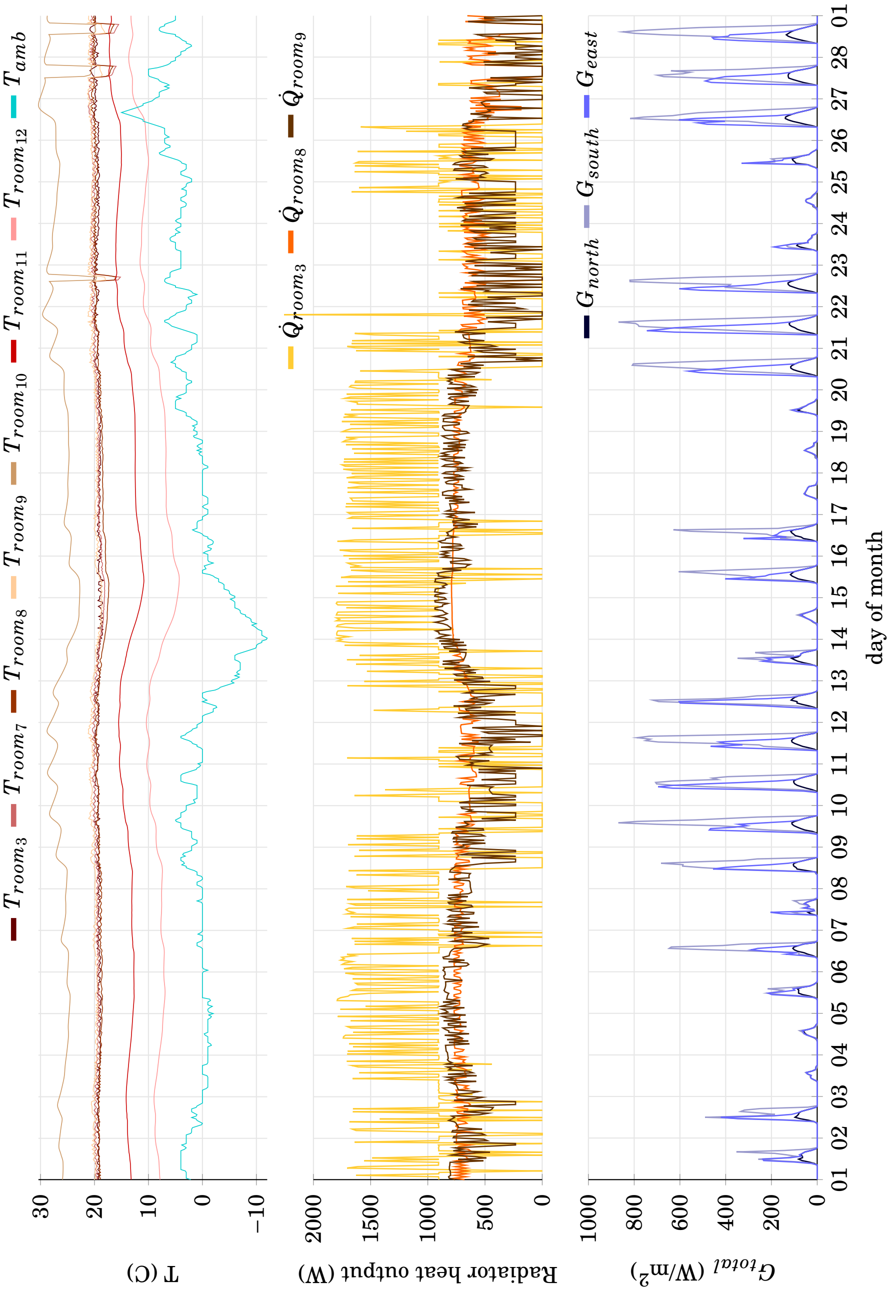


Figure 4.30: Dwelling with radiators in February.

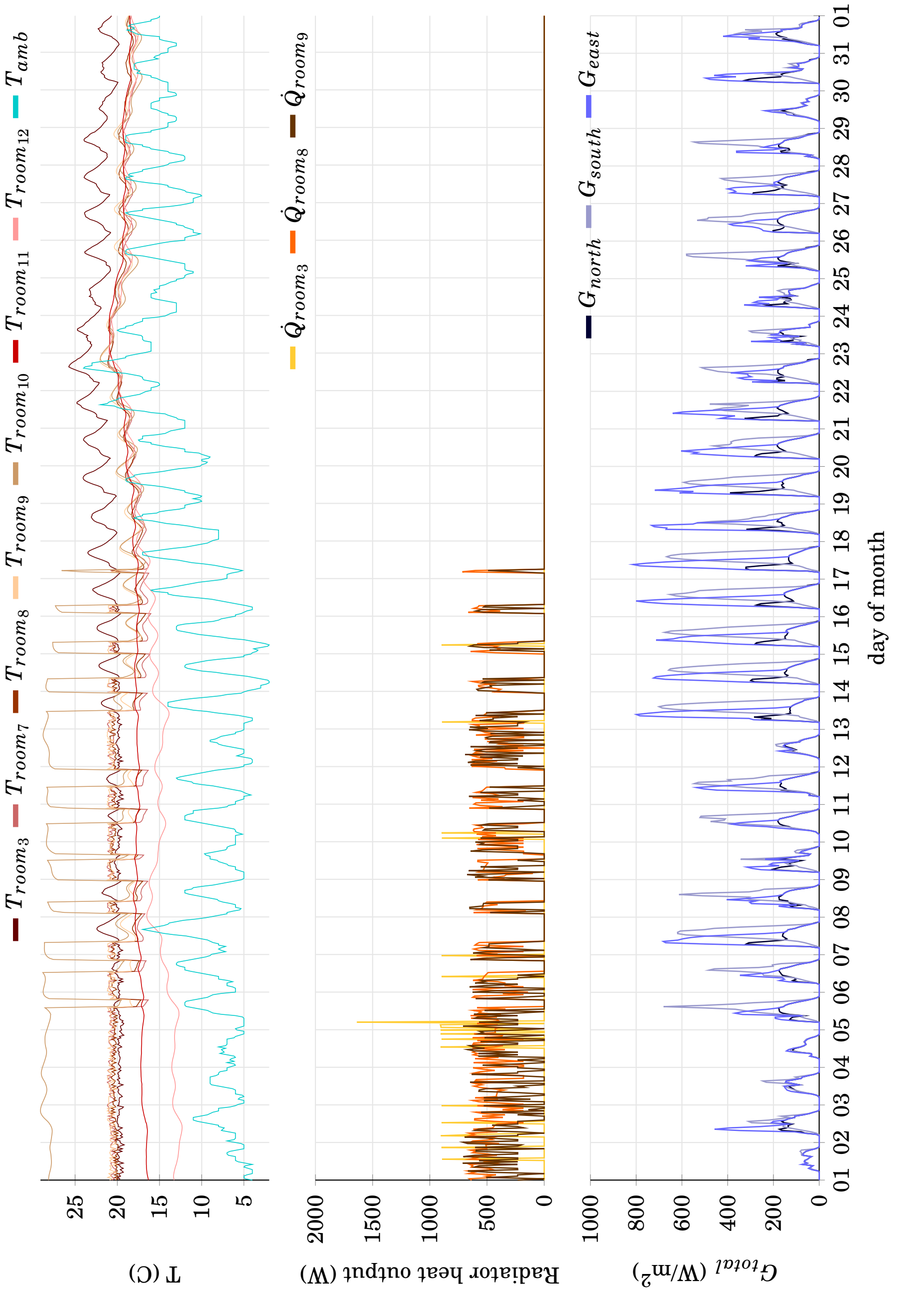


Figure 4.31: Dwelling with radiators in May.

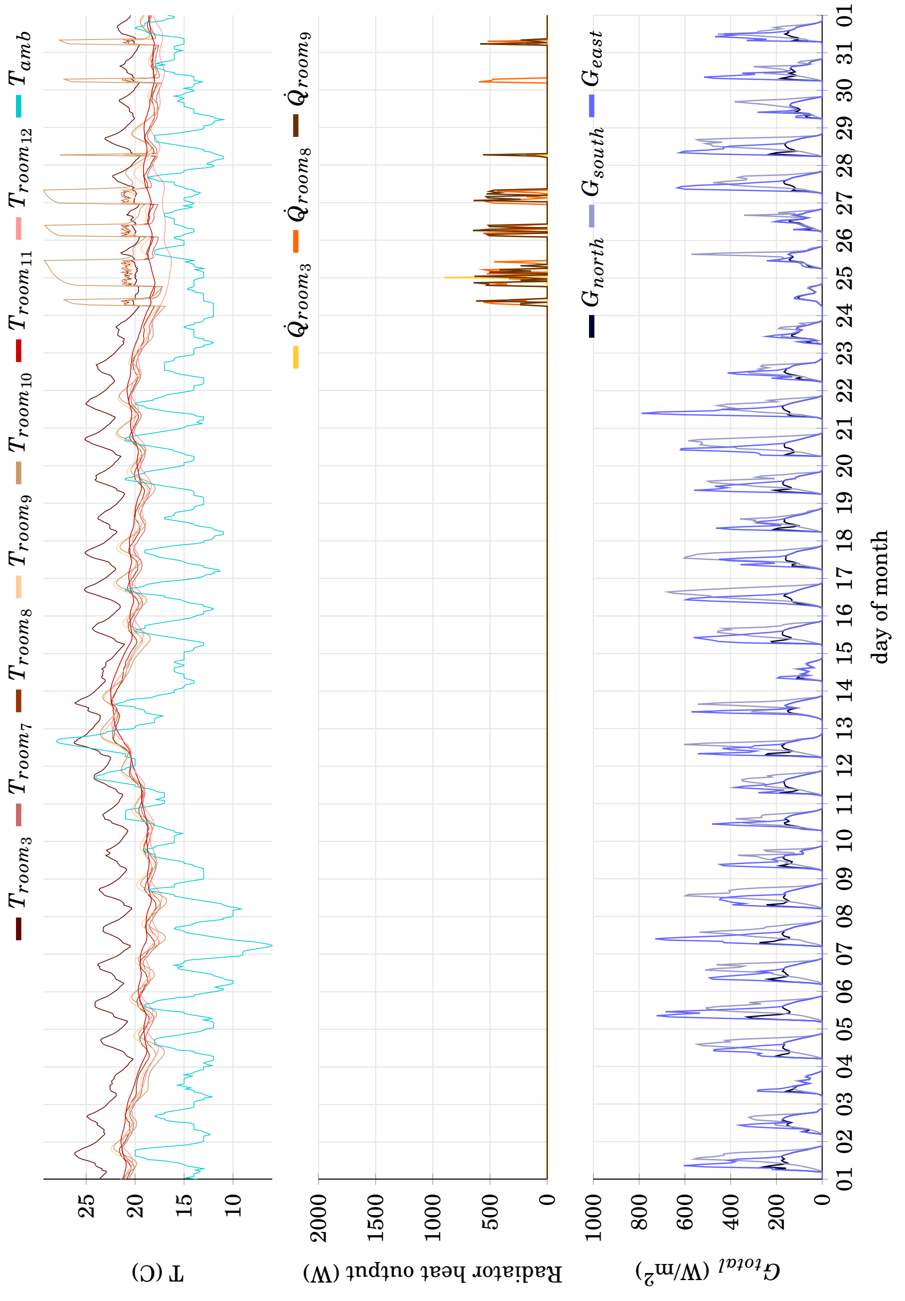


Figure 4.32: Dwelling with radiators in August.

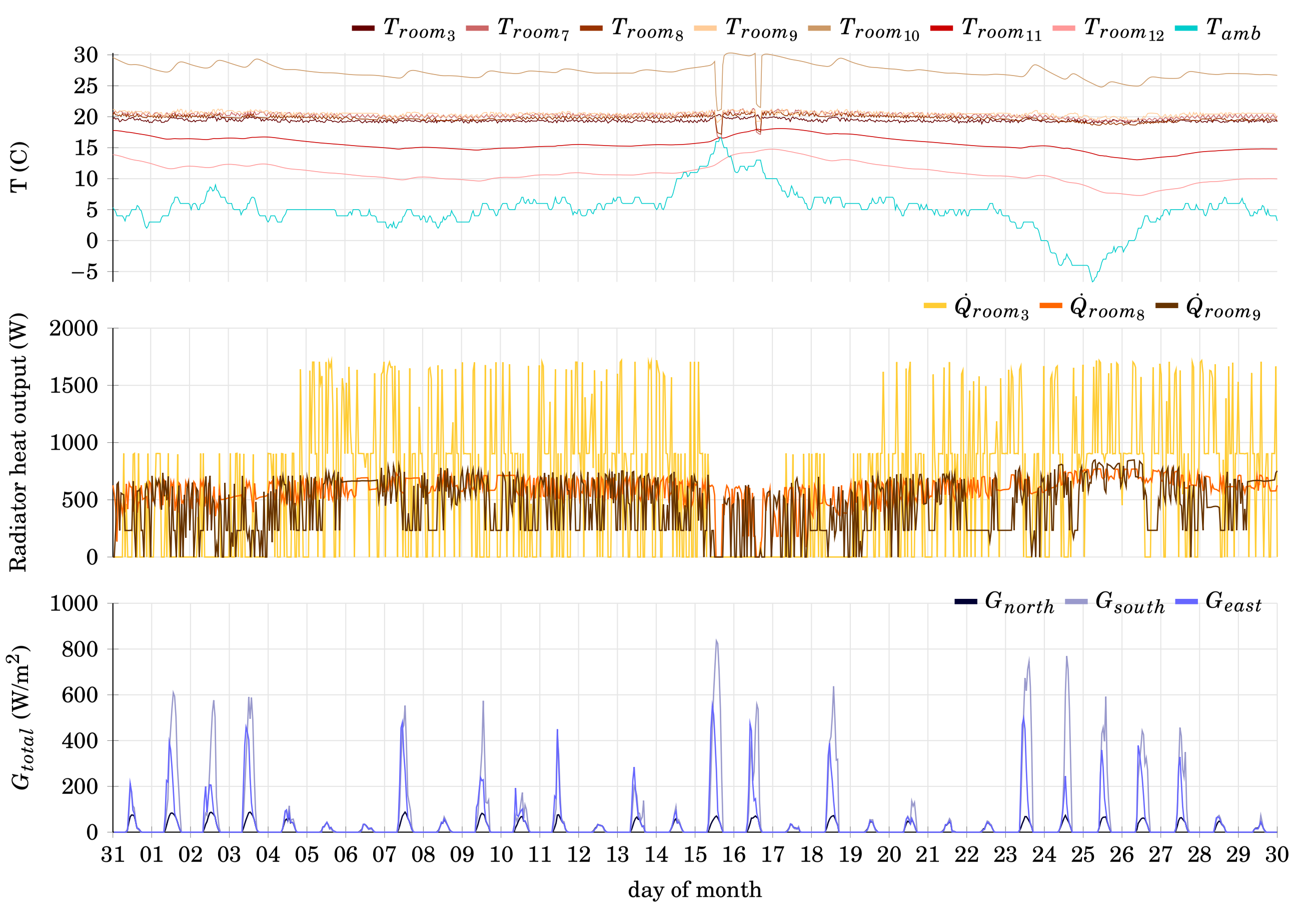


Figure 4.33: Dwelling with radiators in November.

4.4 Conclusions

Throughout this chapter it has been show how the NEST software platform is successfully employed for producing a buildings performance simulation program. It has also been highlighted the potential of the computer simulation tools in the assessment of buildings designs and its use in testing existing or novel technologies and materials.

For illustrating the potential of the developed program, a couple of residential dwellings have been simulated. These numerical experiments show different capabilities of the code. First, an hypothetical small apartment has been analyzed under several scenarios of occupancy and events. The evolution of the temperature and the CO_2 concentration in the different rooms during one week have been presented for each case. The air changes per hour in the building envelope have been presented, too. The obtained numerical data show a reasonable response of the main variables. In other words, the virtual apartment behaves as it could be expected in the reality.

On the other hand, a real semi-detached dwelling has also been simulated. In this instance, the building has been simulated over one year. In contrast to the other application example, this house is composed of three floors and each floor is split into several rooms. This emphasizes the flexibility of the developed tool for adapting to diverse type of buildings. The thermal performance of the house has been analyzed under real weather conditions with and without heating. First, the radiators have been tested in a centralized way by means of a thermostat in the livingroom. After this, the radiators have been controlled locally using thermostatic valves (TRVs). From the numerical results it is observed that the use of TRVs corrects some problems existing in the centralized approach. By means of this simulation, the implementation of the heating sub-models has been verified.

Furthermore, the quality of the presented results depends on the physical correctness of the model. The BESTEST initiative [36, 37] addresses this issue to increase the confidence in buildings performance simulation predictions. In regards to the NEST buildings program in particular, it should be noted that it is not validated yet. There are reasons for trusting in the current implementation of the code since the different sub-models were already validated by Damle in [24]. Still, the NEST code must be verified and validated also. This is a prior condition before definitively trusting the results it yields.

References

[1] International Panel on Climate Change (IPPC). Climate change synthesis report (2007). http://www.ipcc.ch/publications_and_data/ar4/syr/en/spms2.html. Accessed: 2015-11-13.

[2] Climate change data and Information Analysis Center of the US Department of Energy (CDIAC-DOE). http://cdiac.ornl.gov/pns/current_ghg.html. Accessed: 2015-11-13.

[3] United Nations Framework Convention on Climate Change (UNFCCC). Doha amendment to the kyoto protocol (2012). http://unfccc.int/files/kyoto_protocol/application/pdf/kp_doha_amendment_english.pdf. Accessed: 2015-11-12.

[4] United States Congress. Energy Independence and Security Act (2007). http://www.gpo.gov/fdsys/pkg/STATUTE-121/pdf/STATUTE-121-Pg1492.pdf. Accessed: 2015-11-12.

[5] European Commision. Energy strategy for competitive, secure and sustainable energy (2010). http://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:52010DC0639&from=EN. Accessed: 2015-11-12.

[6] European Parliament. Energy Efficiency Directive (2012). http://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32012L0027&from=EN. Accessed: 2015-11-12.

[7] International Energy Agengy (IEA). World Energy Investment Outlook. http://cdiac.ornl.gov/pns/current_ghg.html. Accessed: 2015-11-13.

[8] European Comission. Your key to European statistics. http://ec.europa.eu/eurostat/web/main/home. Accessed: 2015-11-13.

[9] European Parliament. Energy performance of buildings directive (2010). http://www.eceee.org/policy-areas/buildings/EPBD_Recast/EPBD_recast_19May2010.pdf. Accessed: 2015-11-12.

[10] Buildings Performance Institute Europe (BPIE). Europe's Buildings under the Microscope. http://www.bpie.eu/eu_buildings_under_microscope.html#.VkYCyM1pmWY. Accessed: 2015-11-13.

[11] American Council for an Energy-Efficient Economy (ACEEE). Energy Efficiency in the US: 35 Years and Continuing.

[12] A. J. Marszal et al. Zero energy building - a review of definitions and calculation methodologies. *Energy and Buildings*, 43:970–979, 2011.

[13] E. Vietes et al. European initiatives towards improving the energy efficiency in existing and historic buildings. *SIAM Journal on Scientific and Statistical Computing*, 75:1679–1685, 2015.

[14] Dwelling Climate Control System (DCCS). http://www.kic-innoenergy.com/innovationproject/our-innovation-projects/dwelling-climate-control-system-dccs, 2013-2016. EIT-KIC InnoEnergy. Accessed: 2015-12-07.

[15] Retrofitting Solutions and Services for the enhancement of Energy Efficiency in Public Edification (RESSEEPE). http://www.resseepe-project.eu, 2013-2015. European project. Accessed: 2015-12-07.

[16] P. Shalin. Modsim—a program for dynamic modelling and simulation of continuous systems. Technical report, The Swedish Institute of Mathematics, 1988.

[17] J. Clarke et al. The energy kernel system. Technical report, Department of Mechanical Engineering, University of Strathclyde, 1992. Final Report for Grant GR/F/07880.

[18] W.F. Buhl, A.E. Erdem, F.C. Winkelmann, and E.F. Sowell. Recent improvements in spark: Strong component decomposition, multivalued objects, and graphical interface. Technical report, International Building Performance Simulation Association, 1993.

[19] H.E. Feustel. Comis - an international multizone air-flow and contaminant transport model. *Energy and Buildings*, 30:3–18, 1999.

[20] G.N Walton et al. Contam 2.4b, user guide and program documentation. Technical report, Environment Division Building and Fire Research Laboratory, NIST, Gaithersburg, MD, US, 2005.

[21] U.S. Department of Energy Building Technologies Office. Energyplus: creating a new-generation building energy simulation program. https://energyplus.net/. Accessed: 2015-12-02.

[22] D.B. Crawley et al. Contrasting the capabilities of building energy performance simulation programs. building and environment. *Building and Environment*, 43:661–673, 2008.

[23] Q. Chen. Ventilation performance prediction for buildings: A method overview and recent applications. *Building and Environment*, 44:848–858, 2009.

[24] R. Damle. Object-oriented simulation of reciprocating compressors: Numerical verification and experimental comparison. *International Journal of Refrigeration*, 34(8):1989–1998, 2011.

[25] J.A. Clarke and J.L.M. Hensen. Integrated building performance simulation: Progress, prospects and requirements. *Building and Environment*, 91:294–306, 2015.

[26] S. V. Patankar. *Numerical heat transfer and fluid flow*. Hemisphere Publishing Corp., 1 edition, 1980.

[27] Siegel R. and J.R. Howell. *Thermal radiation heat transfer*. Taylor and Francis, 2002.

[28] J. A. Duffie and W. A. Beckman. *Solar Engineering of Thermal Processes*. ASHRAE, 4th edition, 2013.

[29] Meteonorm software. http://meteonorm.com/. Accessed: 2015-12-03.

[30] EN 442-2:1996/A2:2003, Radiators and Convectors Part 2: Test Methods and Rating. CEN, Brussels.

[31] K. P. Andrew et al. Ventilation for acceptable indoor air quality. Technical report, 2003. ANSI/ASHRAE Addendum n to ANSI/ASHRAE Standard 62-2001.

[32] K. P. Andrew et al. Energy-efficient design of low-rise residential buildings. Technical report, 2004. ANSI/ASHRAE Addendum n to ANSI/ASHRAE Standard 90.2-2001.

[33] Y. A. Cengel. *Heat transfer. A practical approach*. McGrawHill, 1998.

[34] Eckert and Drake. *Drake, Analysis of Heat and Mass Transfer*. McGrawHill, 1972.

[35] *Handbook Fundamentals, SI Edition*. ASHRAE, 2009. Chapter 15.

[36] Judkoff R. and Neymark J. Building energy simulation test (BESTEST) and diagnostic method. Technical report, National Renewable Energy Laboratory, Golden, CO, US, 1995.

[37] J. Neymark et al. HVAC BESTEST: a procedure for testing the ability of whole-building energy simulation programs to model space conditioning equipment. In *Proc. Building Simulation*, Rio de Janeiro, 2001.

Chapter 5

Simulation of hermetic reciprocating compressors using NEST

5.1 Introduction

Refrigeration is essential for mankind. In prehistoric times man realized that food was better preserved if it was stored into caves or buried in snow. Nowadays, we all know that low temperature slows bacterial growth so that food is preserved from spoiling. The first household refrigerators were commercialized in the USA around 1920 [1]. So far, our diet was dependent on the seasons and what could be grown in the home region. In the present there are other applications requiring refrigeration aside foodstuffs storage. Probably, the most common is air conditioning of private houses, public buildings and vehicles, although it is also widely used in many industries (cryogenics, large seed banks, medicine and health products, oil refineries, petrochemical and chemical plants, natural gas processing, etc.). Actually, refrigeration represents 15% of global electricity consumption [2].

Many of the aforementioned applications use the vapor-compression technology. This technology uses a working fluid, namely a refrigerant, that is pushed through the system and undergoes state changes repeatedly (from liquid to gas and back). During this cycle the fluid follows three stages: evaporation, compression, condensation and expansion. The latent heat of vaporization of the refrigerant is used to remove heat energy from a hot source (in the evaporator) that is later thrown to a cold source (in the condenser). To enable the several thermodynamic states the fluid is compressed during its gaseous phase. Hence, one of the basic elements of a vapor-compression refrigeration system is the compressor device. Although there are several types of compressors (mainly classified as dynamic and positive displacement) the one more used in domestic and industrial refrigeration is the reciprocating compressor, in particular the hermetic reciprocating compressor. To give just a few examples, they are used in household refrigerators and freezers, domestic and small commercial air conditioners, commercial refrigerating units (food services, ice machines or beverage dispensers), etc.

Despite its unquestionable benefits, refrigeration represents a significant part of the global energy consumption with the corresponding negative consequences on earth atmosphere (global warming and climate change, depletion of natural resources, etc.). Apart from power waste, this damage is being accelerated by the consumption and subsequent uncontrolled emission of refrigerant substances. Until the end of the 1980s most compressors used CFC or HCFC[1] refrigerants due to its favorable thermodynamic properties. However, these substances were proved to contribute significantly to the ozone layer depletion [3, 4], which motivated the Montreal Protocol international agreement in 1989 for phasing they out [5]. As an alternative, CFCs and HCFCs were replaced by HFCs and PFCs which have no impact on the ozone layer but they have an impact on global warming (that is thousands of times greater than CO_2).

Now, HFCs and PFCs are included in the Kyoto Protocol (1997) that commits state parties to reduce greenhouse gases (GHG) emissions [6, 7]. The targets for the first commitment period (2008-2012) varied between nations. Some were allowed to increase their emissions by a certain amount while others were required to make significant cuts. The average target was a cut of around 5% relative to 1990 levels by 2012. Unfortunately, while the Montreal Protocol was considered as an example of exceptional international co-operational the Kyoto Protocol failed to do so. Although the treaty was signed by 189 United Nations members, the major GHG emitters (China, USA and India) did not ratify it. At present, there is a second commitment period (2012 Doha Amendment to the protocol [8]), this time, however, there was even less agreement and several big countries (Russia, Japan or Canada) withdrew from the protocol.

In 2008, the OECD estimated GHG global emissions will grow by about 52% by 2050 if no further policies are adopted [9]. In particular, energy related CO_2 emissions will grow by 78% and the use of HFCs and PFCs will nearly quadruple. It is obvious that there are needs for better well being everywhere. Developed countries do not want to

[1] These acronyms stand for chlorofluorocarbon and hydro-chlorofluorocarbon substances.

loss their privileges while developing countries will not renounce to their aspirations. In addition to the political efforts, scientists and engineers must gain insight into more efficient energy systems and processes. This is critical for changing GHG emissions projection, so also global warming. In that sense, the refrigeration technology must be improved to reduce its power consumption and move towards alternative refrigerants. In this, compressors have an important role and they need more attention.

5.1.1 Literature review

Improving compressors design by means of prototype construction and experimental testing is unpractical and very expensive. Instead, computer simulation is a reliable alternative with appealing features such as the ease to create virtual models, its safety and its flexibility. For this reason, there are many works addressing compressors behaviour prediction in the last decades.

Numerical models for predicting compressors behavior have evolved as long as the knowledge on numerical techniques and computer science have made it possible. Three decades ago, computer simulation of the fluid dynamics was prohibitive. Moreover, the role played by fluid mechanics in reducing the efficiency of these devices was in many cases ill-considered [10]. Initially the compressors were just modeled as pure thermodynamic systems that changed the gas state from a lower to a higher pressure. This approach was based on energy and mass balances in the compression chamber. The kinematics of the crankshaft mechanism and the valves motion were also modeled, which enabled the analysis of the distinct compressor stages (i.e. suction, compression/expansion and discharge) [11–13]. Those mathematical models permitted to visualize pressure-volume diagrams as well as the temperatures of the gas and the main components of the compressor. They were also able to provide pressure pulsations and flow leakage under different cylinder geometries and valve parameters. However, they were still far to predict irreversibilities due to dissipative effects, such as friction or viscosity, since they did not really solve the fluid dynamics of the refrigerant. Consequently, they did not contribute to improve the knowledge on the actual causes of compressors inefficiencies.

Later, the need to improve the compressors efficiency pushed the researchers to formulate more advanced approaches based on the coupled resolution of the conservation laws: continuity, momentum and energy. Following this idea several authors worked on the integration of the one-dimensional form of these equations over the whole compressor domain [14–16]. The methodology proposed by Pérez-Segarra et al. [15] was extended and validated in further works [17–19]. More recently, other authors focused their attention on improving flexibility of the simulation tools aiming to test distinct compressor configurations easily [20,21].

The study of a compressor device is very challenging since it involves fast-transient three-dimensional compressible and turbulent flow together with heat transfer between the components and between the gas and the components. Moreover, this is combined with moving boundaries and structure interaction due to the motion of the suction and discharge valve reeds, the piston, the motor rotation and many other vibrations transferred along the compressor. Despite the recent advances in numerical modeling and the increasingly power of the modern HPC[2] facilities, the obtention of three-dimensional numerical results by simulating the whole domain of the compressor is still unaffordable.

Even so, the use of Computational Fluid Dynamics (CFD) is still crucial for understanding energy losses in particular and compressor performance in general. Obtaining useful experimental measurements from the insides of a hermetic compressor is highly expensive and difficult so that one should move efforts to computer simulation. As a first attempt is one-dimensional modeling. These codes are able to provide detailed information such as the mass flows, the pressure drop and the temperature profiles of the refrigerant along the mainstream. They can also give information about the specific component or the overall compressor energy losses. However, they lack of information on important details such as the shape of certain components (chambers, valve orifices, discharge pipe, etc.) as they are just seen as lumped volumes, flow restrictions, or straight pipes. Here, CFD plays an important role in assessing compressors components design. There are several authors that addressed the multi-dimensional flow problem on specific parts of the compressor. This kind of studies provide valuable information on the heat transfer coefficients, which are necessary for running simplified models [22–24], on the flow structures in the valve orifices [25–29] or on the pressure perturbations and vorticity in the mufflers for analyze performance and noise [30–35].

Despite of its benefits, CFD also involve several risks that should be considered seriously. It is mandatory that the employed CFD codes are verified and validated. Moreover, using these codes is not straightforward and the afforded results should be critically analyzed before accepting them. These tools can not be used as simple black-boxes. Caution is advised when using CFD tools since its application to compressors design is quite novel [10].

In sum, although one-dimensional simulation is a simple modeling level it is still a

[2] High Performance Computing

necessary tool for preliminary designs. The main advantage versus CFD codes is the computational speed. Programs based on this approach can provide reliable numerical results in a few hours, while a CFD case may take several days in the best scenario. The low computational expenses of one-dimensional codes are derived from a significant reduction in the number of unknowns to be determined. In particular, the code presented here is ready for running in parallel clusters, which also contributes to speed up the simulation. Furthermore, thanks to its parallel capabilities the code also enables the use of different levels of modeling so that simple models can be combined with multi-dimensional models in a flexible and transparent way.

For the moment, the NEST software platform has been used in several investigations to get more insight into fluid flow through the valve orifices and the muffler components [29, 35].

5.1.2 Structure and Objective

In this chapter a compressors simulation program based on the NEST software platform is presented. The main goal is to show how NEST is successfully employed in the production of computer simulation programs for specific applications. This program is aimed to obtain, in a fast and reliable way, phenomena predictions on hermetic reciprocating compressors in order to make better preliminary designs. In the following section the one-dimensional model is briefly explained. After this, the influence of the common source of numerical errors on the numerical solution is evaluated. In order to validate the code, this is followed by a comparison of some numerical results with experimental data. These data corresponds to three different compressors working with different refrigerant fluids and conditions. Later, several detailed information is presented to show the prediction features of the compressors program. Finally, several parametric studies on the geometry of a commercial compressor are carried out. This studies contribute to obtain optimal parameters for improving the performance of the compressor.

5.2 The compressor model

The compressors simulation program consists on a set of individual numerical models that solve the conservation laws of mass, momentum and energy in the several components of the compressor (chambers, pipes, valves, compression chamber and so on). These models, or just elements, should be seen as small pieces of code written on the top of the NEST software platform. NEST provides several tools for linking the elements so that they are able to exchange information to each other. In this way, a network of numerical components can be arranged in order to create an overall model of the compressor (see Figure 5.1). This network can be resolved in a coupled way which results in the solution of the fluid-dynamics and the heat transfer over the whole compressor.

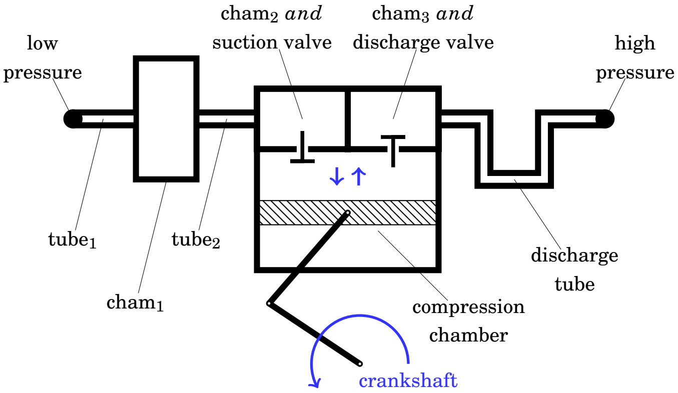


Figure 5.1: Example of a minimal compressor configuration.

This partitioned approach provides an efficient and practical way to simulate the compressor. Unlike monolithic approaches, where all phenomena are resolved by means of a single numerical code, this strategy is much flexible since it allows rapid setup of new compressor configurations. For instance, you can compare the compressor configuration in Figure 5.1 with the one in Figure 5.2. Although both schemes are quite similar enabling their simulation will likely involve a hard programming effort if a monolithic code is used. Instead, the NEST platform is just designed to deal with these situations. The compressor components are treated as puzzle pieces and they can be inserted, replaced or removed from the global computational model as appropriate.

5.2.1 NEST models for compressor components

The different models available for solving the fluid dynamics and the heat transfer phenomena are summarized in Table 5.1.

In addition, the general conservation laws of the fluid flow in their semi-discrete form are presented in the Equations 5.1, 5.2 and 5.3. These equations are solved in the

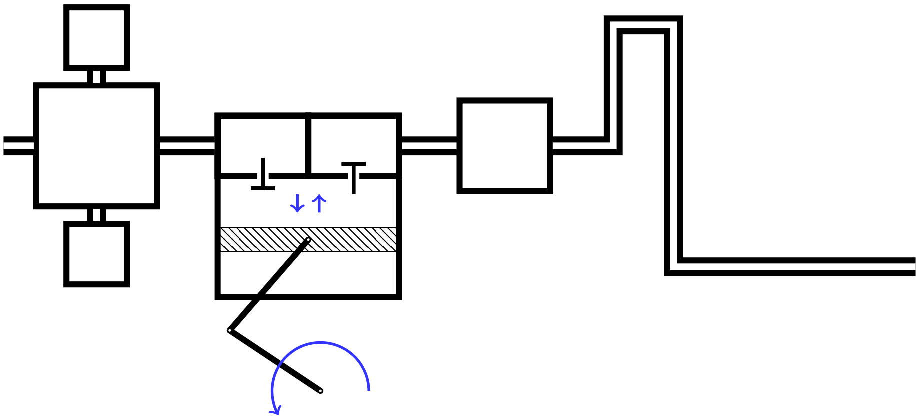


Figure 5.2: Some modifications on the example configuration from Figure 5.1.

Tube and the *Chamber* sub-models. The friction factor coefficients between the walls and the fluid flow in the *Tubes* are obtained from the classical expressions for pipes [38].

$$\frac{\partial m}{\partial t} = \sum [\dot{m}]_{bnd} \tag{5.1}$$

$$\frac{\partial (m\bar{v})}{\partial t} = \sum [\dot{m}v]_{bnd} + F_{bnd} \tag{5.2}$$

$$\frac{\partial [m(\bar{h} + \bar{e}_c)]}{\partial t} = \sum [\dot{m}(h + e_c)]_{bnd} + \dot{Q}_{bnd} + \bar{V}\frac{\partial \tilde{p}}{\partial t} \tag{5.3}$$

Aside from the *Chamber* model, the simulation program also includes a model to take into account the piston displacement into the compression chamber. The treatment of the flow in this cavity is the same as any other chamber. In that case, however, the volume of the cavity varies periodically between the TDC and the BDC[3]. The piston position is evaluated at every instant by resolving the kinematics and dynamics of the slider-crankshaft mechanism, which is depicted in Figure 5.3.

Regarding the *ValveReed* sub-model, the simplest way to evaluate the reed position and the mass flow through the valve orifice is by means of a mass-spring system analogy. This model assumes that the reed behaves as a driven harmonic oscillator [11]. In that case, the motion Equation 5.4 is obtained from the Newton's second law, where z denotes

[3] The Top Dead Center (TDC) and the Bottom Dead Center (BDC) are the positions of the piston in which it is nearest to and farther from the crankshaft.

$Tube$	One-dimensional Semi-Implicid Method for Pressure Linked Equations (SIMPLE [36]). Upwind criteria is used for convective terms.
$Chamber$	Pressure correction approach (in the same way as Tube).
$CompCham$	Extended $Chamber$ model provided with a solver routine for solving the kinematics and dynamics of the crankshaft mechanism (i.e. crank angle, free body diagram forces, motor torque, etc.).
$ValveReed$	One or multi-dimensional model for evaluating mass flow through valve orifices. The reed position is obtained by solving the mass-spring equation or by means of modal analysis.
$ExpCFDElem$	Low-Mach based CFD&HT model for compressible flow. This solver is distributed with the TermoFluids software [37].
$Body1D$	Convection, conduction and radiation heat transfer between the components of the compressor and the refrigerant fluid.
$Body3D$	Heat transfer in the actual three-dimensional geometry of the compressor components.
$FixedValue$	It is a simple NEST built-in type to fix boundary conditions (e.g suction and discharge pressures).

Table 5.1: Available numerical models for simulating hermetic reciprocating compressors.

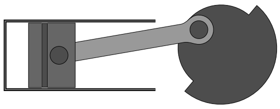


Figure 5.3: Scheme of the slider-crankshaft mechanism.

the reed position, ζ is the damping coefficient, ω is the natural frequency [4] of the reed and F_s, F_d and F_f are the spring, damping and flow forces on the mass m. For a more accurate description of the valve reed movement a multi-dimensional simulation is also possible by means of modal analysis. For the sake of brevity, this model is not described here although its formulation is available in [40].

[4] The frequency at which an object tends to vibrate in the absence of any driving or damping force is known as the natural frequency [39].

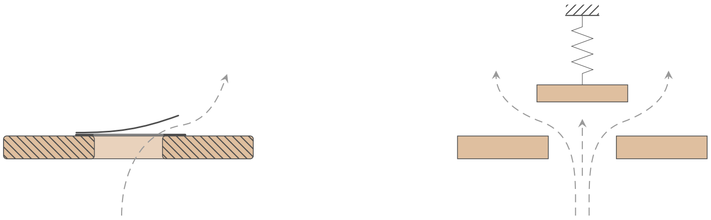


Figure 5.4: Compressor valve sketch and the mass-spring system analogy.

$$\frac{d^2z}{dt} + 2\zeta\omega\frac{dz}{dt} + \omega^2 z = \frac{F_s(t) + F_d + F_f}{m} \tag{5.4}$$

Furthermore, the following fully implicit discrete energy equation is resolved by the *Block1D* sub-model. This equation takes into account the heat conduction between the solid components (see Figure 5.5), the heat convection with the refrigerant fluid and the heat transport by radiation.

$$\frac{m\bar{c}_p(T_k^n - T_k^{n-1})}{\Delta t} = \sum_j \frac{T_k^n - T_j^n}{R_{kj}} A_{kj} + \sum_i \dot{Q}_{ki}^{conv,n} + \sum_j \dot{Q}_{kj}^{rad,n} \tag{5.5}$$

The convection heat exchanges are evaluated using empirical information. The heat transfer coefficients employed inside the chambers, the tubes and the compressor chamber, as well as the one used to evaluate heat losses to the ambient are obtained from the literature [40–43]. Finally, the radiation heat exchanges are only considered between the shell and the environment. In this case, it is assumed that radiation transport takes place between the external surface of the compressor and the sky as a black body.

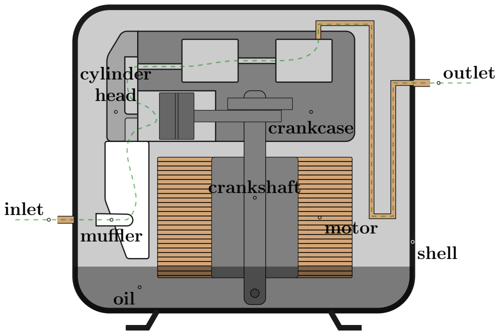


Figure 5.5: Scheme of a generic compressor with its main solid components.

Further information on the mathematical formulation of the several models is available in [17,20]. For more details on the *ExpCFDElem* and the *Block3D* models one must refer to [37,44].

5.2.2 Global resolution algorithm

The resolution scheme of the overall model is based on a block-Jacobi like method. During this procedure, every element is visited to call the resolution routines of their particular sub-models. After this, each element changes its state (e.g. pressure, velocity and temperature profiles or heat losses/gains). Once all the elements have been iterated, the program synchronize their input/output data in order to maintain them up to date with the state of their neighboring elements (i.e. boundary pressures, input/output mass flows, heat transfer, etc.). This is repeated several times until the elements are well coupled so that they do not change their state. At this point the whole set of elements is converged to an instantaneous compressor state. Then, the crank angle is shifted to the next position (i.e. the crankshaft rotates) and the simulation continues towards the next time step. This scheme is summarized in Figure 5.6.

5.2.3 Numerical verifications

Numerical verifications are aimed to check the code is free of programming errors as well as to analyze the influence of the numerical nature errors on the solution. On the one side, the code may have programming errors, which can be fixed so that they do not affect the solution anymore. On the other side, there are other source of errors, such as

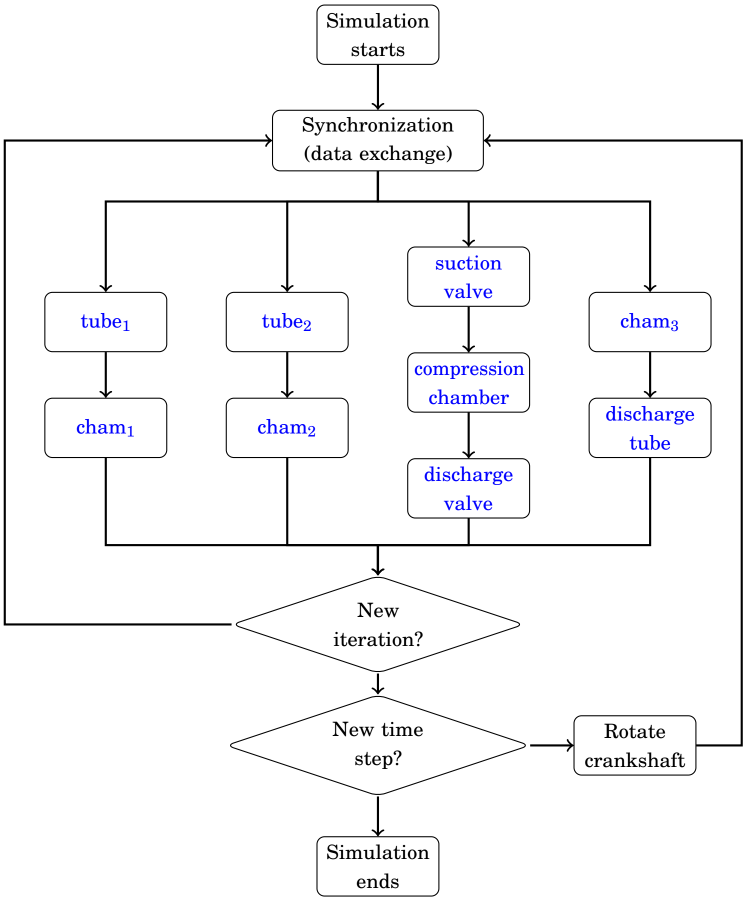


Figure 5.6: Numerical scheme for resolving the configuration example of Figure 5.1. In this instance, the nine sub-models/elements are solved in four parallel processes.

convergence errors and spatial or time discretization errors, which are inherent to the numerical methods and can not be removed but controlled and minimized.

Although the numerical studies below are classified in three subsections -convergence, spatial and time discretization-, they are focused on all the aforementioned source of errors. Actually, several programming errors were fixed during the development of these studies.

The studies were carried out on the test compressor outlined in Figure 5.7. This compressor is deliberately simple in order to reduce the computational expenses and get rid of those geometry complexities that are not strictly necessary for verifying the code[5]. The refrigerant fluid is R134a and the cylinder capacity is 7 cm^3. In all cases the evaporation temperature is fixed to -23.3 ^oC and the pressure ratio is Π = 13.

Note that the complexity of the fluid flow and heat transfer phenomena is not reduced despite its simpleness. The test compressor is provided with the most important components for assuring that the essential parts of the code are executed and hence, also tested. The remnant gas chamber produces an initial flow pressure drop and the incoming refrigerant is mixed with warm gas before flowing towards the suction line. This chamber is responsible of the strong coupling between the several compressor parts. It thermally connects all the internal components to each other and also with the shell and the environment. Although they are not represented in Figure 5.7, the solid components are also resolved. The suction and discharge valves are modeled as harmonic oscillators that vibrate at high frequencies. This motion modifies the flow boundaries and submits it to sudden accelerations and decelerations at every cycle. The compression chamber is in turn a moving boundary. It is the driven force that compresses the refrigerant to the high pressure. The kinematics and dynamics of the slider-crankshaft mechanism is solved at every time step to update the piston position and cylinder volume. Finally, the discharge tube colds the fluid down before it exits the compressor. Moreover, there are sudden flow expansions and contractions through each pipe-chamber connection.

[5] Simplicity is also a good property for introducing a new test in the compressors code test suite, which helps ensure that further code maintenance (bugs fix, extensions, improvements, etc.) will not alter already verified parts of the code.

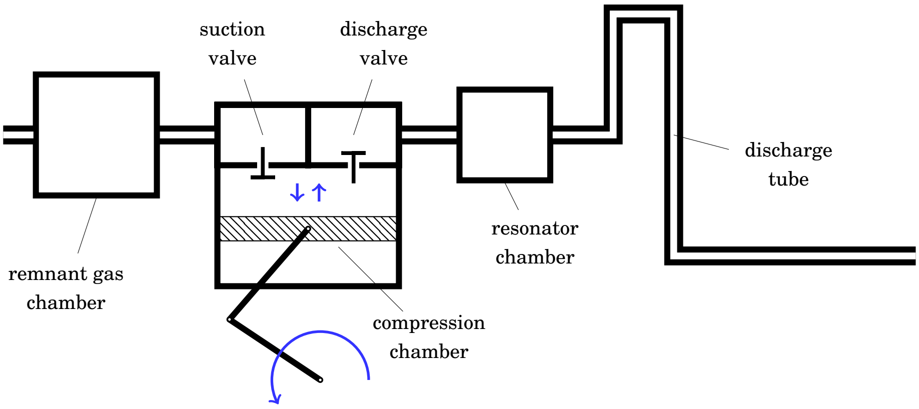


Figure 5.7: Configuration scheme of the test compressor.

The numerical results are presented by means of bar charts that should show an asymptotic tendency. The depicted parameters are the power consumption $\dot{W}_{ele}$, the ambient losses $\dot{Q}_{amb}$, the crankshaft frequency $\dot{\theta}$, the motor torque T_m, the discharge temperature T_{disch} and the energy balance relative to the consumption [6]. All these parameters are averaged values along the compression cycle when stationary conditions are achieved.

Convergence error The flowchart decision block referred as "*New iteration?*" in Figure 5.6 employs a precision value ϵ to control whether the program keeps iterating or it advances to the next time step. Specifically, the main residual values of the resolved equations must be lower than ϵ for considering their solutions (e.g. pressure, enthalpy, velocity, motor torque, etc.) are good enough to advance in the time dimension. The bar charts in Figure 5.8 show the influence of the convergence error on the overall numerical solution. Each bar corresponds to a different value of the ϵ parameter. From these data it is observed that the numerical solution is not strongly affected by ϵ, which should be greater than 10^{-4} for assuring a small overall error.

[6] This value is the ratio between the energy equation residual and the consumption. It is ideally zero.

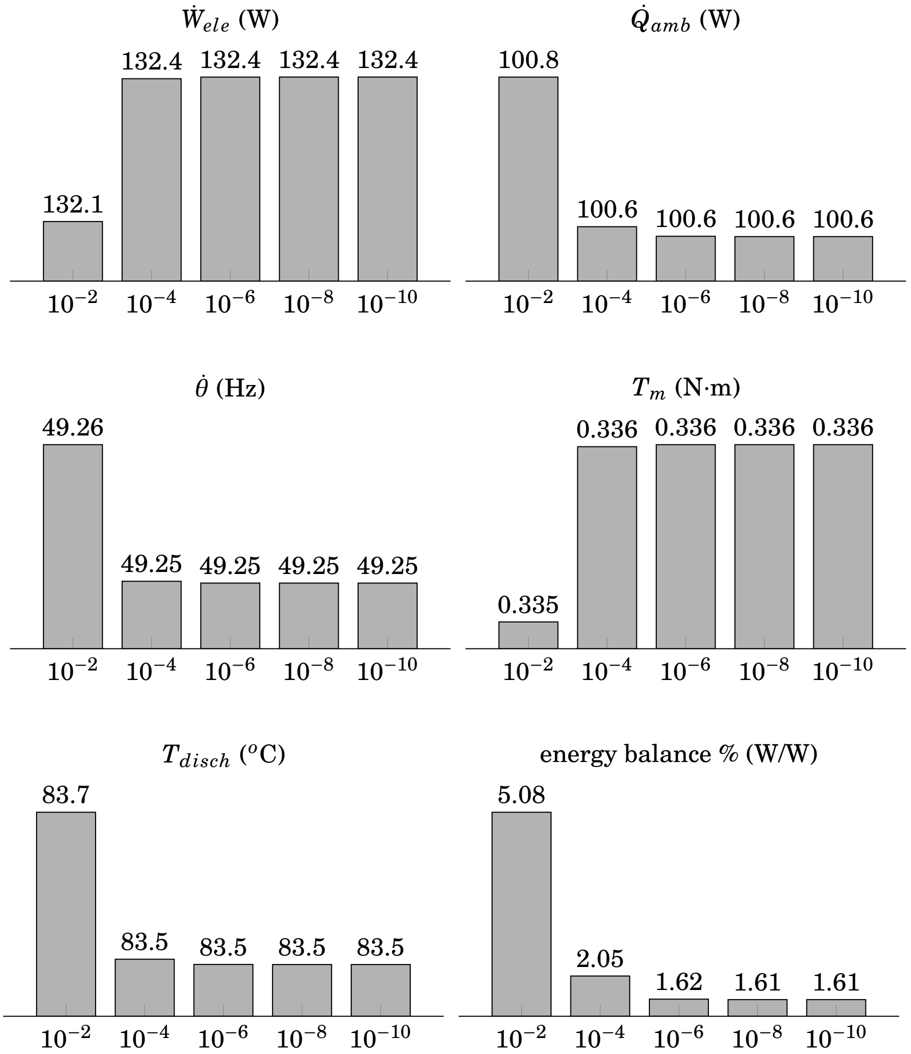


Figure 5.8: Influence of the convergence error ϵ on several global parameters of the test compressor.

Spatial discretization error The numerical models implemented in the compressors code use a spatial discretization for resolving their respective equations. Here, the one-dimensional fluid flow and heat conduction models ($Tube$ and $Block1D$ models) are analyzed. The bar charts in Figure 5.9 are aimed to show the influence of the mesh size on the overall numerical solution. In particular, the several cases are differentiated using the number of control volumes employed in the discharge tube. Regarding to the obtained results, it can be seen that the spatial discretization error is small when the mesh size is bigger than 1.[7] However, the error on the energy balance does not show an asymptotic tendency with the successive mesh refinements. Even so, this value tends to zero as expected. Hence, as the tendency of the remain parameters is acceptable, it can be assumed that the numerical results are good enough when the energy balance error is lower than 5%.

[7] $n_{unk} = n_{cv} + 2$ so that 1 control volume involve three unknowns, 2 control volumes involve 4, 3 involve 5, and so on.

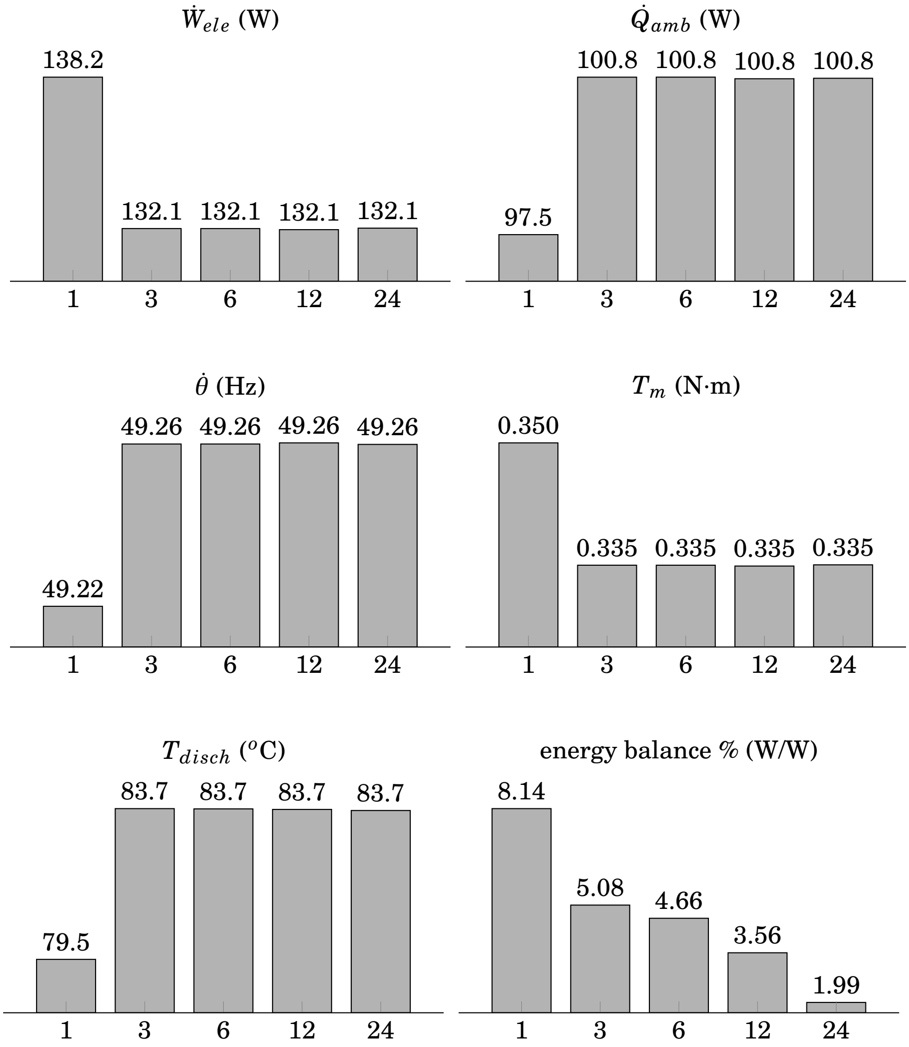


Figure 5.9: Influence of the spatial discretization error on several global parameters of the test compressor.

Time discretization error The influence of the time discretization on the overall numerical solution is evaluated in the bar charts of Figure 5.10. The presented data corresponds to various simulations performed using different time steps. Actually, due to the fast transient phenomena in the valves, the time step required in compressors simulation to achieve any converged solution is in general a very small value. For this reason, here the time discretization is shown as the number of subdivisions of a complete crankshaft rotation.[8] In that instance, unlike the convergence and spatial discretization errors, the overall numerical solution is much dependent on the time step. Actually, the analysis of the several global parameters is only asymptotic from 1440 steps.

[8] The time step value can be obtained as $dt = (\dot{\theta} n_s)^{-1}$, where n_s is the number of steps the cycle is divided in.

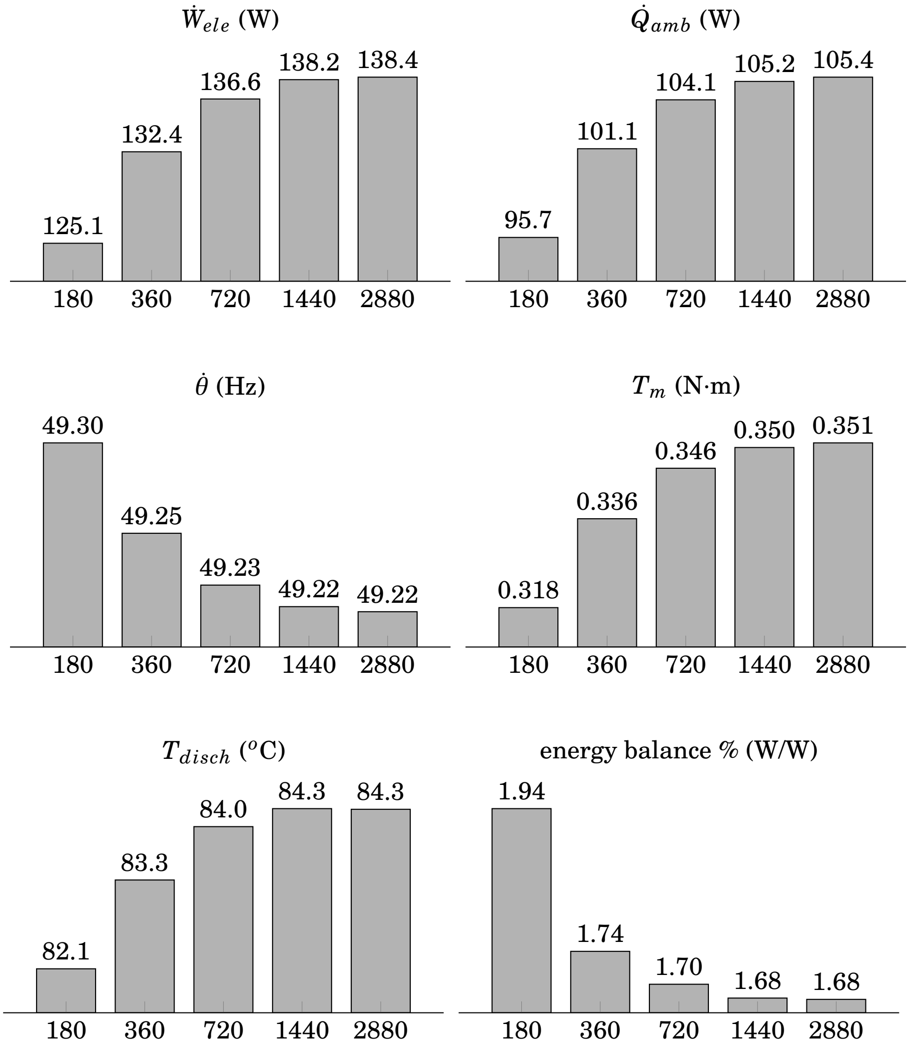


Figure 5.10: Influence of the time discretization error on several global parameters of the test compressor.

Because of the significant influence of the time discretization error in the results, some more results are presented below in Figure 5.11. These additional charts show the pressure-volume diagram, the pressure variation along the crankshaft revolution and the valves motion. These data are presented for each one of the employed time steps. In all these charts it can be seen the strong influence of the time step on the accuracy of the valve reed opening.

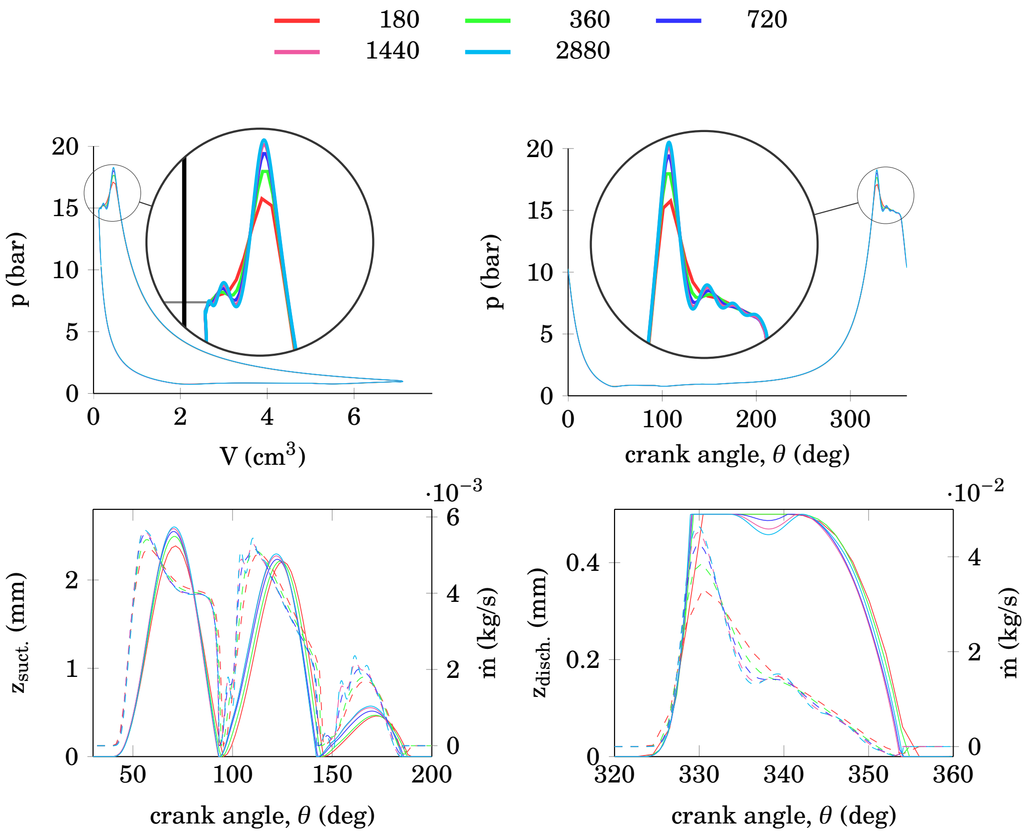


Figure 5.11: Pressure-Volume diagram, pressure profile during a compression cycle and suction and discharge valves motion. The mass flow through each valve is also represented with a dashed line in the respective plots.

5.2.4 Experimental validation

In order to validate the compressors simulation program several different commercial hermetic reciprocating compressors have been simulated. These compressors and their respective working conditions are presented in Table 5.2. C_3 compressor is simulated for four different working conditions. The experimental data is obtained from previous research works on compressors [40,45]. The comparison of the experimental data with the numerical results gives an idea on the accuracy of the compressor simulation tool, which is essential for trusting in the predictions it yields.

	cc[1] $[cm^3]$	refrigerant	Π	p_{suct} [MPa]	T_{evap} [^oC]
C_1	1.5	CO_2	2.0	4.44000	10.0
C_2	10.0	R600a	12.3	0.06265	-23.3
C_{3a}	7.5	R134a	7.5	0.20060	-10.0
C_{3b}	7.5	R134a	13.0	0.11484	-23.3
C_{3c}	7.5	R134a	17.7	0.08438	-30.0
C_{3d}	7.5	R134a	22.5	0.06614	-35.0

Table 5.2: Validation tests classified by cubic capacity, refrigerant and working conditions. Ambient temperature is 32^{o}C in all cases.

In Tables 5.3 and 5.4 are presented the experimental and the numerical values of the mass flow $\dot{m}$ (and the volumetric efficiency $\eta_v = (60\rho_{evap}V_{cyl}f_n)/\dot{m}$), the power consumption $\dot{W}_{ele}$, the $COP = \dot{Q}_{evap}/\dot{W}_{ele}$, the discharge temperature T_{disch} and the shell temperature T_{shell}. These data are time averaged values along one compression cycle once it achieves stationary conditions.

In general, the numerical prediction of all the parameters is good. In Table 5.5 are shown the prediction error of every observed parameter. The compressor mass flow and the consumption differ by at most 7% and 6% the experimental measurement. In addition, the COP is well estimated as well. The mass flow and the consumption values are always overestimated or underestimated in the same direction. This indicates a good coupling between the fluid dynamic and the energy equations and it explains the good COP predictions too (see case C_{3a} data as an example).

The discharge and the shell temperatures are in most cases well predicted. There is a remarkable variation in T_{disch} error between C_1 and C_2 cases and the remain of

[1] cubic capacity

C_3 cases. The heat transfer coefficients between the remnant gas in the shell and the several inner components are considered as constant and equal in all three compressors. This is not necessarily true and it could explain the discrepancy in the prediction errors.

Note that the estimation of the power consumption fits very well the experimental data in C_1, C_2 and C_{3c}. The model for solving the crankshaft mechanism depends on the friction factors between the moving components (i.e. bearings, crank, connecting rod and piston). Unfortunately, these data is unknown and it is difficult to obtain them experimentally. To workaround this issue, the friction factors in the mentioned cases have been adapted to fulfill the expected consumption. Apparently this strategy undermine the compressor model. Even so, the mass flow, the COP and the temperatures are still well predicted, which denotes the mathematical model is good. Moreover, the four working conditions of C_3 compressor are all of them simulated using the same inputs (refrigerant fluid, geometry and empirical information). Therefore, the estimation of the friction factor for a certain working condition enables to study the same compressor in different contexts.

Exp.	$\dot{m}$ [kg/h]	η_v	$\dot{W}_{ele}$	COP	T_{disch}	T_{shell}
C$_1$	18.42	70.5	335.1	2.485	96.9	na
C$_2$	1.85	73.1	91.3	1.89	82.6	65.9
C$_{3a}$	7.17	66.0	203.0	1.820	79.6	51
C$_{3b}$	3.54	57.4	138.0	1.335	67.5	48.5
C$_{3c}$	2.16	47.7	108.0	1.045	56.5	46.75
C$_{3d}$	1.47	41.7	91.0	0.845	53.5	44.25

Table 5.3: Experimental data of all compressors and conditions.

Num.	$\dot{m}$ [kg/h]	η_v	$\dot{W}_{ele}$	COP	T_{disch}	T_{shell}
C$_1$	19.72	75.5	338.0	2.636	101.6	59.0
C$_2$	1.77	69.9	90.4	1.822	76.0	64.5
C$_{3a}$	6.79	62.39	191.0	1.819	76.9	49.5
C$_{3b}$	3.54	57.27	135.0	1.352	65.5	46.9
C$_{3c}$	2.28	50.30	107.7	1.094	58.3	44.7
C$_{3d}$	1.58	44.20	90.7	0.901	55.0	43.0

Table 5.4: Numerical results obtained for all compressors and conditions.

Error	$\dot{m}$ [kg/h]	$\dot{W}_{ele}$	COP	T_{disch}	T_{shell}
C_1	7.1	0.9	6.1	4.9	na
C_2	-4.3	-1.0	-3.6	-8.0	-2.1
C_{3a}	-5.36	-5.91	-0.07	-3.40	-2.98
C_{3b}	-0.10	-2.21	1.28	-2.96	-3.30
C_{3c}	5.32	-0.28	4.69	3.27	-4.39
C_{3d}	7.25	-0.34	6.59	2.82	-2.87

Table 5.5: Discrepancy between numerical results and experimental data (relative errors).

5.2.5 Example

This section is aimed to show the potential of the simulation tool with regard to the numerical predictions. To this end, several numerical results obtained in the simulation of the C_1 compressor in Table 5.2 are presented here. This compressor has a cylinder capacity of 1.5 cm^3, it works using CO_2 and the simulation is performed at an evaporation temperature of +10.0 ^{o}C.

Case setup For running a simulation, the computer program requires the following data: inlet and outlet fluid flow pressure, inlet fluid temperature, compressor geometry (tubes and chambers dimensions, diameter and thickness of the valve orifices, dead volume, crankshaft mechanism, etc.), valve reed parameters (stiffness, damping coefficient and mass), electrical efficiency and torque motor curves, bearings friction factors, empirical correlations for evaluating heat transfer coefficients and walls friction factors, ambient conditions and finally the type of refrigerant fluid and the corresponding database with its thermophysical-physical properties.

Obviously, each compressor has its own parametric description. However, the empirical information is always obtained in the same way. For example, the friction factor in the tubes is evaluated by means of the Churchill correlation [46]. On the other side, the heat transfer coefficient between the gas in the shell and the compressor components is estimated by 150 W/m^2, the one between the tubes and their walls is evaluated through the Dittus-Boelter correlation, while the one in the chambers is calculated using whether the Ooi correlation [43] or the Todescat coefficients [41]. In particular, the heat transfer between the gas in the compression chamber and the cylinder walls is evaluated by means of the Liu-Zhou correlation [42]. It should be noted that no *ad hoc* coefficients are used for running the simulation.

To complete the program input, the topology of the compressor-system must be provided also. The compressor topology describes the connectivity of the elements that conform the NEST model. In Figure 5.12 are shown the compressor configuration scheme and its vertex-edge graph representation. Since the simulation tool is ready to run in parallel, the compressor elements must be distributed among the several processors. The different computer processes are indicated in the graph by color.

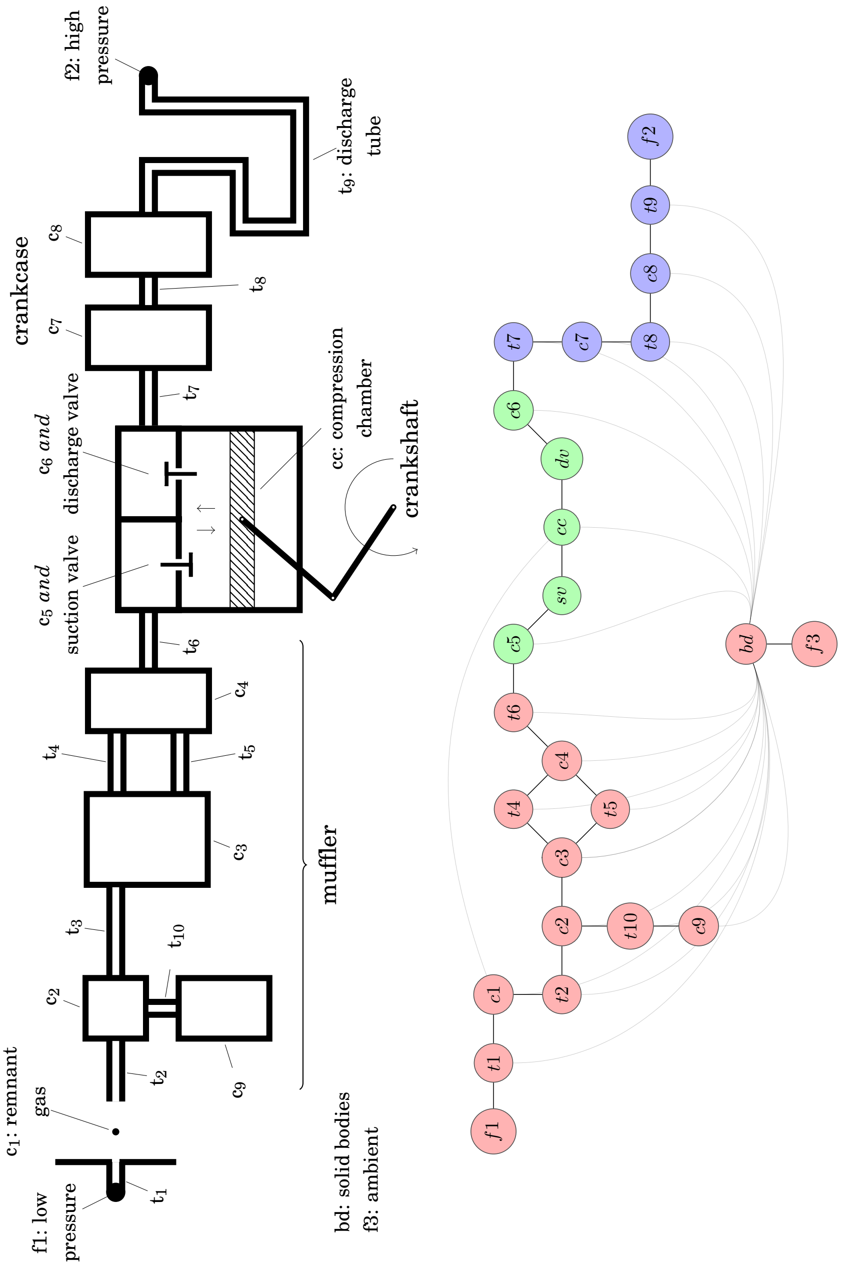


Figure 5.12: Components of the simulated compressor and its vertex-edge graph representation. The body vertex represents the set of the main solid components: muffler, crankcase, cylinder head, motor and discharge tube.

The numerical results obtained after running the simulation of the above compressor are shown below in Figure 5.13. The power consumption along a crankshaft revolution is shown in the last plot together with some other parameters. These parameters are the power transferred to the crankshaft and to the gas in the cylinder, the heat exchange between this gas and the cylinder walls and the heat due to the slider-crankshaft mechanism and the motor inefficiencies.

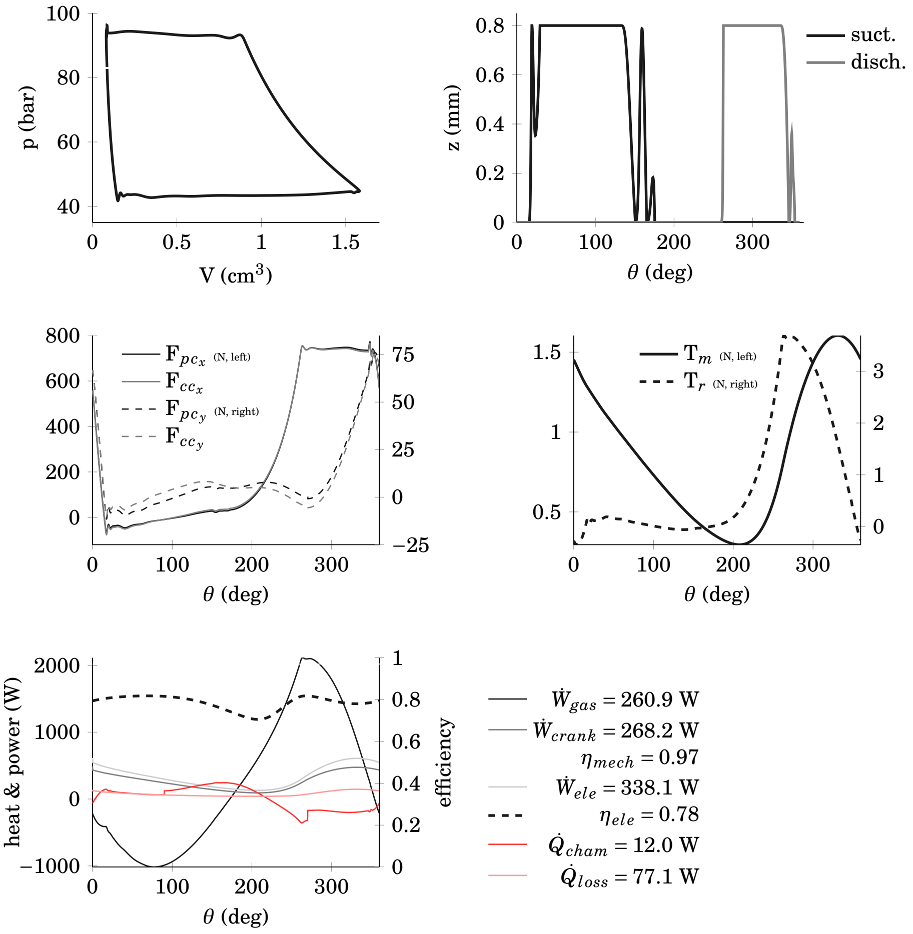


Figure 5.13: From left to right and top to bottom: pressure-volume diagram, suction and discharge valves motion, forces on the slider-crankshaft mechanism, motor and resistant torques and finally power consumption together with some other related parameters.

5.3 Parametric studies

Several parametric studies on an hermetic reciprocating compressor are presented in this section. This is aimed to show the possibilities of the simulation tool and its final objective: to gain insight into the thermal and fluid dynamic behavior of compressor devices in order to improve their design. The obtained numerical results reveal the influence of different aspects (geometric parameters, crankshaft mechanism and motor) on the performance of the compressor (i.e. cooling capacity and power consumption). All data have been presented in a non-dimensional form. In particular, the results show the percentage of variation of the main parameters taking the actual compressor design and its performance as the reference.

5.3.1 Analysis of the stroke to bore ratio influence

The aim of this study is to find the optimal relation between the stroke length and the diameter of the cylinder. For simplicity, the study is done in the basis that the volume of the cylinder ($\pi D^2 L_s/4$), the eccentricity of the crankshaft (e_c), and the length of the connecting rod (L_c) are fixed in all the numerical experiments. The slider-crankshaft mechanism and the main geometrical dimensions are depicted in Figure 5.14.

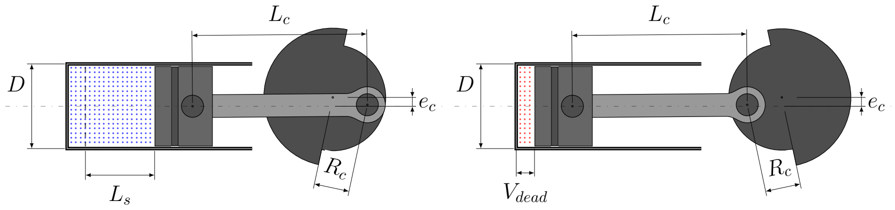


Figure 5.14: TDC (left) and BDC (right) of the slider-crankshaft mechanism. The main geometric parameters are also indicated.

Up to twelve different L_s/D ratios have been tested. The numerical results are presented in Figure 5.15. The abscissa axis quantifies the variation of the stroke to bore ratio while the ordinate axis represents the variation of the cooling capacity $\dot{Q}$, the power consumption $\dot{W}_{ele}$, and the COP. The chart shows that both $\dot{Q}$ and $\dot{W}_{ele}$ increases as the L_s/D ratio grows. At a first stage the COP does also increase although it is maintained approximately constant after the current L_s/D ratio. The point where the three lines intersect represents the position of the reference geometry. According to this chart, it can be said that the current stroke to bore ratio is good enough since the COP obtained at this value is very close to the optimum.

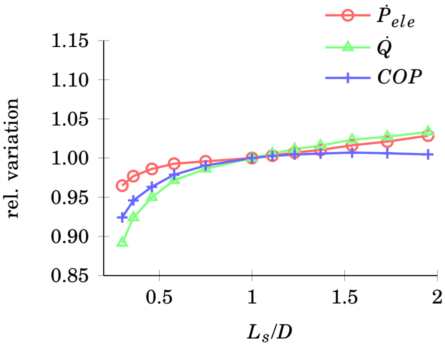


Figure 5.15: Cooling capacity $\dot{Q}$, power consumption $\dot{W}_{ele}$, and COP versus L/D ratio.

5.3.2 Analysis of suction and discharge valve orifice area

The employed numerical model defines the valves as the combination of two flow restrictions. First the fluid flows though a small orifice (axial restriction) and later it pushes and opens the valve reed (radial restriction). This study is focused on the first restriction independently of the secondary one. In particular, the area of the orifice is analyzed by changing its diameter. Since the mathematical model is based on the one-dimensional formulation of the conservation equations, the shape of the valve orifices is supposed to be cylindrical as shown in Figure 5.16.

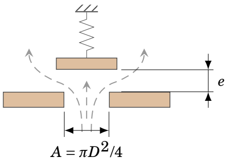


Figure 5.16: Valve model and main dimensions.

The study is done on the suction valve first and on the discharge valve later. Twelve different orifice diameters have been tested for each valve. For investigating the effect of the orifice size in insulation, the changes on one valve have not been combined with the changes on the other valve.

The several values of the cooling capacity $\dot{Q}$, the power consumption $\dot{W}_{ele}$ and the COP obtained in both studies are presented in Figures 5.17 and 5.18. Again, the presented data are relative to the original compressor. Hence, the abscissa quantifies the variation of the orifices area and the ordinate axis shows performance variation.

Suction valve The tendencies in Figure 5.17 seems reasonable. As the suction orifice grows the cylinder is able to aspirate more gas and therefore the cooling capacity increases. The more amount of gas flowing into the cylinder the more power is required to compress it to the high pressure.

Furthermore, it is observed that both $\dot{Q}$ and $\dot{W}_{ele}$ have a maximum value around the half of the original area. At this point, the compressor reaches the best volumetric efficiency. In the remain experiments both parameters achieve an asymptotic tendency. In regards to the COP, it grows fast at the smaller areas and later it gets asymptotic. Therefore, up to the original suction valve orifice, the bigger is the orifice the better is the performance of the compressor. After this point nothing changes.

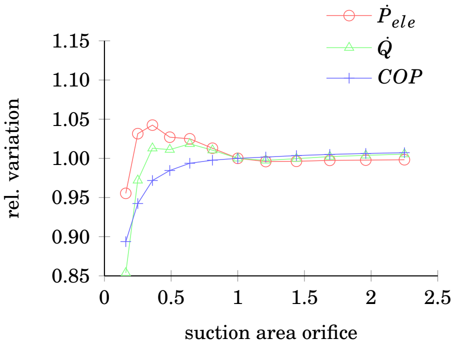


Figure 5.17: Cooling capacity $\dot{Q}$, power consumption $\dot{W}_{ele}$, and COP versus orifice size.

Discharge valve In the same way, the compressor performance parameters for the discharge orifice study are depicted in Figure 5.18. In that instance, the power consumption gets down as the discharge orifice is enlarged. This was expected since the suction area is maintained constant in all these experiments so the aspiration capacity is the same.

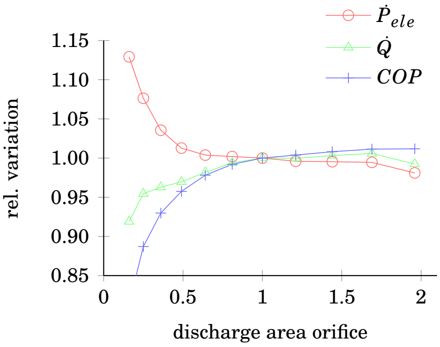


Figure 5.18: Cooling capacity $\dot{Q}$, power consumption $\dot{W}_{ele}$, and COP versus valve size.

In sum, these results show that the larger is the area of both orifices the higher is the COP of the compressor. However, it is likely that not all of the presented cases are feasible. For instance, there can be size restrictions due to the small space where the valves are usually placed. Consequently, the choice of the more appropriate orifice size is a trade-off between performance and manufacturing issues. In addition, the size of the suction orifice is also dependent on the muffler exit. Moreover, the enlargement of the discharge orifice does also increase the dead volume so that special attention should be paid in that sense, too.

5.3.3 Analysis of suction and discharge valves thickness

This study is aimed to test different values of the suction and discharge valve stiffness K (see Equation 5.4). This parameter is linked to the thickness of the valve reed, as denoted by Equation 5.6.

$$K = m\omega^2 = (\rho A_r e_r)\omega^2 \tag{5.6}$$

Note that the volume is evaluated as $A_r e_r$. This is a reasonable approximation since the geometry of a compressor valve reed results from the extrusion of a two-dimensional shape, as illustrated in Figure 5.19. The natural frequency is obtained from a frequency analysis, which was done using a commercial software [47]. This parameter depends on the shape of the valve as well as on its thickness.

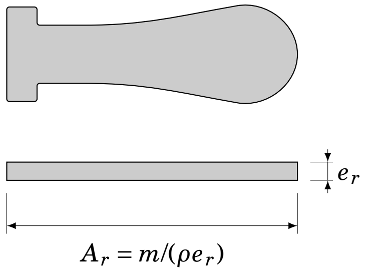


Figure 5.19: Valve reed drawing with its main dimensions.

The suction and the discharge valve reeds have been studied separately. Each valve was simulated using nine different thickness e_r. First, in Figure 5.20 are depicted the stiffness and the natural frequency versus the thickness. A small change on the thickness involve a big change on the stiffness. Although the behavior is the same on both valves, it must be noted that the absolute values are totally different.

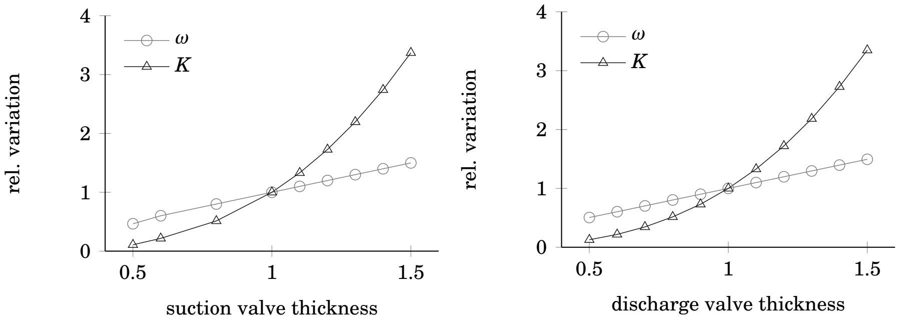


Figure 5.20: Stiffness and natural frequency of the suction (left) and discharge (right) valve reeds versus their thickness.

On the other side, the cooling capacity $\dot{Q}$, the power consumption $\dot{W}_{ele}$ and the COP are shown in Figure 5.21. The results show that the suction valve reed performs better for low thickness values (i.e. low stiffness) while the discharge one performs better for the higher values (i.e. higher stiffness). Even so, the influence of the thickness (or stiffness) on the compressor performance is not so strong. Despite this observation, the compressor reliability could be compromised due to these changes since using a too thin reed may lead to a valve failure.

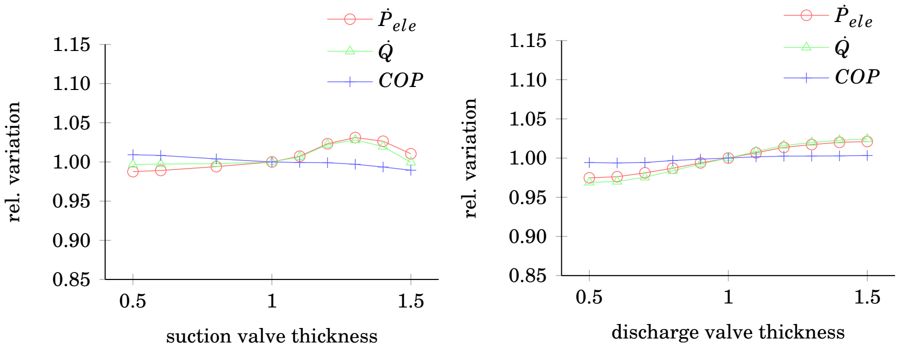


Figure 5.21: Cooling capacity $\dot{Q}$, power consumption $\dot{W}_{ele}$, and COP versus valve thickness for both studies.

5.3.4 Analysis of discharge valve stop

Most hermetic reciprocating compressors have an opening control device in the discharge valve. This device usually consists on a small bridge placed downstream over the discharge orifice, as shown in Figure 5.22. This element restricts the movement of the valve reed up to a certain height. Although this reduces the mass flow and it is one more restriction that causes pressure drop, it reduces the degree of flexion of the reed which extends its lifetime. This device, also known as valve stop, is analyzed here in order to know the minimum value that still provides good performance.

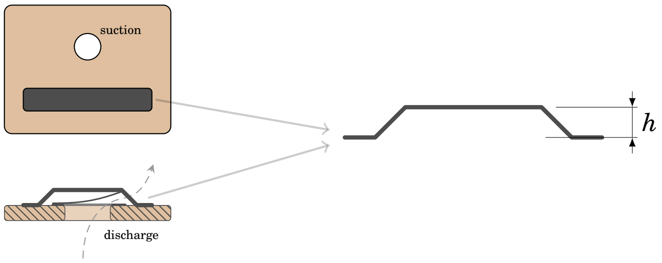


Figure 5.22: Valve plate (above) and detail of the discharge orifice (below). The stop device is installed after the discharge valve. This restriction prevents the valve reed from suffering too high deformations.

The cooling capacity $\dot{Q}$, the power consumption $\dot{W}_{ele}$ and the COP obtained from several simulation cases are represented in Figure 5.23. These numerical results show that the *COP* grows fast up to a certain value, which is maintained constant in the remain of the tests. Note also the high dependency of the power consumption on the valve stop. Using a wrong valve stop height may seriously damage the compressor performance.

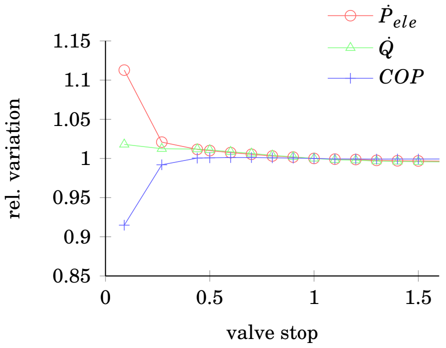


Figure 5.23: Cooling capacity $\dot{Q}$, power consumption $\dot{W}_{ele}$, and COP versus valve stop.

5.3.5 Analysis of dead volume

The dead volume is the clearance between the inner part of the valves plate and the piston when it reaches the BDC[9] (see Figure 5.14). Obviously, a collision between the piston and the valves plate would immediately lead to a crankshaft failure due to the force of the impact. Moreover, it would produce an undesired high noise. Hence, the dead volume is essential for the good functioning of the compressor. However, the clearance must be adequately fixed. The larger is this volume the less suction capacity is obtained, which directly affects to the volumetric efficiency.

In Figure 5.24 are presented the numerical results obtained from the simulation of seven compressor configurations, each one with a different dead volume. In the chart the cooling capacity $\dot{Q}$, the power consumption $\dot{W}_{ele}$ and the COP are depicted versus the dead volume. As expected, there is a high influence of the dead volume on the cooling capacity. Unfortunately, the consumption increases as the clearance decreases. Consequently, the influence of the dead volume on the COP is not so relevant. Nevertheless, the study shows that the compressor performs slightly better at smaller

[9] Bottom Dead Center

volumes.

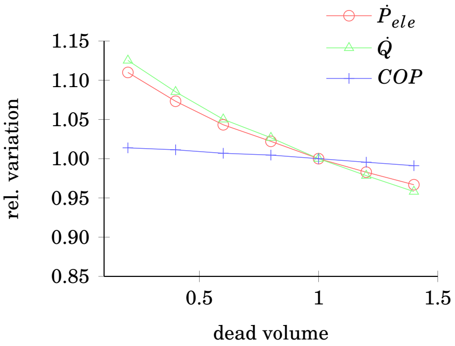


Figure 5.24: Cooling capacity $\dot{Q}$, power consumption $\dot{W}_{ele}$, and COP versus dead volume.

5.3.6 Analysis of electrical efficiency

This analysis is aimed to show the influence of the motor on the performance of the compressor. To this end, several cases have been carried out, each one assuming a different electrical efficiency of the motor. In Figures 5.25 are represented the cooling capacity , the power consumption and the COP versus the several tested motors. Moreover, in Figure 5.26 are depicted the mass flow and the temperature of the compressor shell. While the mass flow is not so affected, the temperature of the shell colds down when better motors are mounted. This was expected since the motor power losses are converted to heat sources, which contributes to increase the compressors temperature.

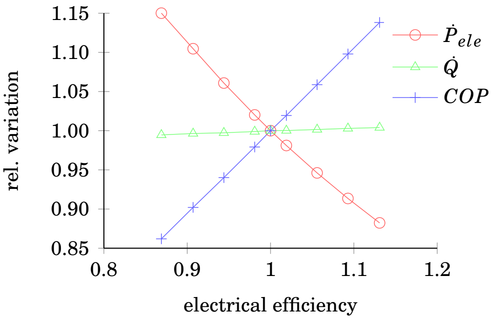


Figure 5.25: Cooling capacity $\dot{Q}$, power consumption $\dot{W}_{ele}$, and COP versus the several tested motors.

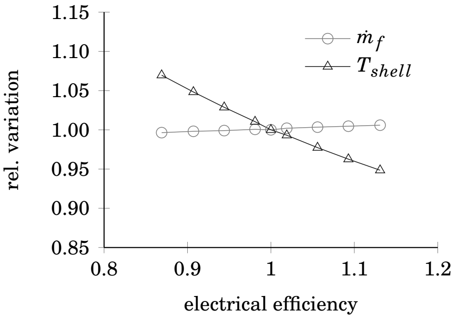


Figure 5.26: Mass flow and shell temperature versus the several tested motors.

5.4 Conclusions

A compressors simulation program build on the top of NEST software platform has been presented in this chapter.

Several numerical verifications have been analyzed in order to assess the quality of the numerical solutions provided by the program. Among these verifications it has been checked the typical source of errors in numerical methods: convergence errors and spatial and time discretization errors. Furthermore, the compressors program has also been validated using experimental data of three compressors working with different refrigerant fluids and distinct conditions. The main global parameters (mass flow, power consumption and COP, as well as the shell and the discharge temperature) obtained numerically are in good agreement with the experimental data. Therefore, this tool is ready for prediction of compressor behavior and hence, it is also ready for being used in compressor preliminary designs or in improvement of already existing compressors.

As an example, a real commercial compressor has been analyzed by means of several parametric studies. The chosen compressor is very well positioned among the several compressors of the same category that are available in the market. The simulation tool is able to show that the device is well designed since the cooling capacity and the consumption are close to their optimal values.

It is worthy to mention that the tool can be used in combination with CFD&HT codes since it enables multiple levels of modelization among the several compressors parts. Actually, this was already tested is [48, 49]. In those works the remnant gas in the suction area and inside the whole shell was simulated using TermoFluids [37], while the remain parts of the compressor where simulated using simplified models such as the ones presented in this chapter. Although this research needs deeper analysis, this can be used for assessing the flow behavior in the suction area (optimal aspiration orifice to muffler entrance distance, pressure drop and inlet gas heating benefit, etc.) or for obtaining correlations for evaluating the heat transfer between the solid components and the refrigerant. More recently, the presented compressors program has also been used to generate reliable transient boundary conditions for analyzing the muffler component. In [35, 50] are presented firsts attempts for understanding the influence of the thermal and fluid dynamic phenomena on the compressor performance and the noise it produces. Some results obtained in these works are presented in the Appendix A of this thesis.

Finally, aside of the commented research lines the current compressors code already deserves some more work. The code is ready for parallel execution. Actually, the numerical simulations presented here were obtained using several processors. However, the parallel performance has not been check nor alternative global resolution algorithms that may accelerate the resolution of the overall problem. For the moment, only a block-Jacobi like and a Gauss-Seidel algorithms have been tested for simulating compressors.

References

[1] ASHRAE. Industry and history. https://www.ashrae.org/about-ashrae/ashrae-and-industry-history/domelre-first-electric-refrigerator. Accessed: 2015-11-09.

[2] D. Coulomb. Conference opening. In *Slovak Association for Refrigeration and Air conditioning*, Papiernika, Slovakia, September 2013.

[3] M. H. Molina and F. S. Rowland. Stratospheric sink for chlorofluoromethanes: Chlorine atom-catalysed destruction of ozone. *Nature*, 249(5460):810–812, 1974.

[4] J. C. Farman, B. G. Gardiner, and J. D. Shanklin. Large losses of total ozone in antarctica reveal seasonal clox/nox interaction. *Nature*, 315(6016):207–210, 1985.

[5] United Nations Environment Programme (UNEP). The Montreal protocol on substances that deplete the ozone layer. http://ozone.unep.org/en/treaties-and-decisions/montreal-protocol-substances-deplete-ozone-layer. Accessed: 2015-11-09.

[6] United Nations Framework Convention on Climate Change (UNFCCC). The Kyoto protocol (1998). http://unfccc.int/resource/docs/convkp/kpeng.pdf. Accessed: 2015-11-09.

[7] United Nations Framework Convention on Climate Change (UNFCCC). National greenhouse gas inventory data for the period 1990-2012. http://unfccc.int/resource/docs/2014/sbi/eng/20.pdf. Accessed: 2015-11-11.

[8] United Nations Framework Convention on Climate Change (UNFCCC). Doha amendment to the kyoto protocol (2012). http://unfccc.int/files/kyoto_protocol/application/pdf/kp_doha_amendment_english.pdf. Accessed: 2015-11-12.

[9] Organisation for Economic Co-operation and Development (OECD). Climate Change: Meeting the Challenge to 2050. http://www.oecd.org/env/39762914.pdf. Accessed: 2015-11-11.

[10] B. G. Shiva. CFD for positive displacement compressors. In *International Compressor Engineering Conference*, Purdue, IN, USA, July 2004. Purdue e-Pubs. Paper 1689.

[11] J. P. Elson and W. Soedel. Simulation of the interaction of compressor valves with acoustic back pressures in long discharge lines. *Journal of Sound and Vibration*, 34:211–220, 1974.

[12] R. Prakash and R. Singh. Mathematical modeling and simulation of refrigerating compressors. In *International Compressor Engineering Conference*, Purdue, IN, USA, July 1974. Purdue e-Pubs. Paper 132.

[13] P. J. Singh. A digital reciprocating compressor simulation program including suction and discharge piping. In *International Compressor Engineering Conference*, Purdue, IN, USA, July 1984. Purdue e-Pubs. Paper 444.

[14] A. B. MacClaren et al. A comparison of numerical solutions of the unsteady flow equations applied to reciprocating compressor systems. *Journal of Mechanical Engineering*, 17(6):271–279, 1975.

[15] Perez-Segarra et al. Numerical study of the thermal and fluid dynamic behaviour of reciprocating compressors. In *International Compressor Engineering Conference*, Purdue, IN, USA, July 1994. Purdue e-Pubs. Paper 971.

[16] F. Escanes. *Numerical simulation of the thermal behaviour of diferent components of compression refrigeration systems. Experimental validation. Application to non-contaminant refrigerants*. PhD thesis, Universitat Polotèctinca de Catalunya, 1995.

[17] Perez-Segarra et al. Detailed thermodynamic characterization of hermetic reciprocating compressors. *International Journal of Refrigeration*, 28:579–593, 2004.

[18] J. Rigola et al. Parametric studies on hermetic reciprocating compressors. *International Journal of Refrigeration*, 28(2):253–266, 2005.

[19] J. Rigola et al. Numerical simulation and experimental validation of vapour compression refrigeration systems. special emphasis on co2 trans-critical cycles. *International Journal of Refrigeration*, 28(8):1225–1237, 2005.

[20] R. Damle. Object-oriented simulation of reciprocating compressors: Numerical verification and experimental comparison. *International Journal of Refrigeration*, 34(8):1989–1998, 2011.

[21] H. Jian et al. Generic network modeling of reciprocating compressors. *International Journal of Refrigeration*, 45:107–119, 2014.

[22] G. W. Recktenwald. *Numerical modelling of the flow and heat transfer in the cylinder of a reciprocating compressor*. PhD thesis, Purdue University, 1986.

[23] J. Polman. Heat transfer in a piston-cylinder system. *International Journal of Heat and Mass Transfer*, 24:184–187, 1981.

[24] A. A. Kornhauser and J. L. Smith. Application of a complex nusselt number to heat transfer during compression and expansion. *Journal of Heat Transfer*, 116:536–542, 1994.

[25] P. Cyklis. CFD simulation of the flow though reciprocating compressor self-acting valves. In *International Compressor Engineering Conference*, Purdue, IN, USA, July 1994. Purdue e-Pubs. Paper 1016.

[26] C. J. Deschamps et al. Turbulent flow though valves of reciprocating compressors. In *International Compressor Engineering Conference*, Purdue, IN, USA, July 1996. Purdue e-Pubs. Paper 1135.

[27] Perez-Segarra et al. Numerical study of turbulent fluid flow through valves. In *International Conference on Compressors and their Systems*, London, UK, July 1999.

[28] J. Rigola et al. Numerical simulation of fluid flow through valve reeds based on large eddy simulation models (les). In *International Compressor Engineering Conference*, Purdue, IN, USA, July 2008. Purdue e-Pubs. Paper 1917.

[29] J. Rigola et al. Numerical analysis of the turbulent fluid flow through valves. geometrical aspects influence at different positions. *IOP conference series: materials science and engineering*, 90:12–26, 2015.

[30] P. Ma et al. Heat and mass transfer in a separated flow region for high prandtl ans schmidt numbers under pulsatile conditions. *International Journal of Heat and Mass Transfer*, 90(17):2723–2836, 1994.

[31] M. Ibrahim and W. Hashim. Oscillating flow in channels with a sudden change in cross section. *Computer Fluids*, 23(1):211–224, 1994.

[32] E. C. Mladin and D. A. Zumbrunnen. Dependence on heat transfer to a pulsating stagnation flow on pulse characteristics. *Journal of Thermophysics and Heat Transfer*, 9(1):181–192, 1995.

[33] A. Nakano and K. Kinjo. CFD applications for development of reciprocating compressors. In *International Compressor Engineering Conference*, Purdue, IN, USA, July 2008. Purdue e-Pubs. Paper 1842.

[34] E. L. Pereira et al. Performance analysis of reciprocating compressors through computational fluid dynamics. *Proc. Inst. Mech. Eng. Part J. Process Mech.*, 222:183–192, 2008.

[35] J. Lopez et al. Compressible 1D-3D simulation of a muffler with pseudosound prediction levels. In *24th International Congress of Refrigeration*, Yokahoma, Japan, August 2015.

[36] S. V. Patankar. *Numerical heat transfer and fluid flow*. Hemisphere Publishing Corp., 1 edition, 1980.

[37] O. Lehmkuhl, C.D. Perez-Segarra, R. Borrell, M. Soria, and A. Oliva. Termofluids: A new parallel unstructured CFD code for the simulation of turbulent industrial problems on low cost PC cluster. *Lecture Notes in Computational Science and Engineering*, 67:275–282, 2009.

[38] C. Donald and M. H. Hobart. *Pipeflow: a practical and comprhensive guide*. John Wiley & Sons, 1 edition, 2012.

[39] L. R. Kinsler et al. *Foundamentals of Acoutics*. John Wiley & Sons, 4 edition, 2000.

[40] J. Rigola. *Numerical simulation and experimental validation of hermetic reciprocating compressors. Integration in vapour compression refrigerating systems*. PhD thesis, Universitat Polotèctinca de Catalunya, 2002.

[41] M. L. Todescat et al. Thermal energy analysis in reciprocating hermetic compressors. In *International Compressor Engineering Conference*, Purdue, IN, USA, July 1992. Purdue e-Pubs. Paper 936.

[42] R. Liu and Z. Zhou. Heat transfer between gas and cylinder wall of refrigerating reciprocating compressors. In *International Compressor Engineering Conference*, Purdue, IN, USA, July 1984. Purdue e-Pubs. Paper 441.

[43] K.T. Ooi. Heat transfer study of a hermetic refrigeration compressor. *Applied Thermal Engineering*, 23:1931–1945, 2003.

[44] O. Lehmkuhl. *Numerical resolution of turbulent flows on complex geometries*. PhD thesis, Universitat Polotèctinca de Catalunya, July 2012.

[45] J. Rigola et al. Thermal and fluid dynamic behavior of transcritical carbon dioxide hermetic reciprocating compressors: Experimental investigation. In *International Conference on Compressors and Their Systems*, London, UK, 2005c.

[46] S. W. Churchill. Friction factor equation spans all fluid-flow regimes. *Chemical Engineering*, 84:91–102, 1977.

[47] Solid works software. http://www.solidworks.com/. Accessed: 2015-11-09.

[48] J. Lopez et al. Numerical study of suction gas flow in the shell of hermetic reciprocating compressors. In *International Conference on Compressors and their Systems*, pages 395–404, London, UK, September 2009.

[49] J. Lopez et al. Use of a low-mach model on a CFD&HT solver for the elements of an object oriented program to numerically simulate hermetic reciprocating compressors. In *International Compressor Engineering Conference*, Purdue, IN, USA, July 2012. Purdue e-Pubs. Paper 2220.

[50] J. Rigola et al. Numerical analysis of suction mufflers. In *International Compressor Engineering Conference*, Purdue, IN, USA, July 2014. Purdue e-Pubs. Paper 2381.

Chapter 6

Using NEST for resolving the heat transfer phenomenon in hermetic reciprocating compressors

6.1 Introduction

The available computational power has increased significantly in the last decade thanks to the rise of computer clusters. This sets a new paradigm in the development and the usage of software tools for physics simulation. This scenario enables the engineers and scientists not only to incorporate the latest modeling techniques to their codes (LES turbulent flow models, fluid-structure interaction, multiphysics coupling, etc.) but also to exploit classical physical models not feasible before such computing growth (multi-phase modeling, three-dimensional compressible flow, etc.). This chapter, though an illustrative compressor simulation, aims to show that today's software is a reliable tool to understand better and improve industrial devices and processes.

Nowadays, hermetic reciprocating compressors are so optimized that it is becoming increasingly difficult to improve its efficiency. For this reason, one must go to the design details in order to remove the small inefficiencies that prevent improvements in modern compressors. Here, the CFD&HT codes have an essential role. So far, powerful software tools based on one-dimensional mathematical models made it possible optimized compressor designs by means of parametric studies (e.g. [1,2]). These tools provide essential global information on the compressor performance such as the cooling capacity and the consumption. Other data such as the transient mass flows in the pipes, the transient temperature maps of the refrigerant, the average temperature of the main solid components, the local pressure perturbation, piston leakage, the sliding-crankshaft mechanism strains and much other local information is also provided indirectly in these one-dimensional simulations (see example in Figure 6.1).

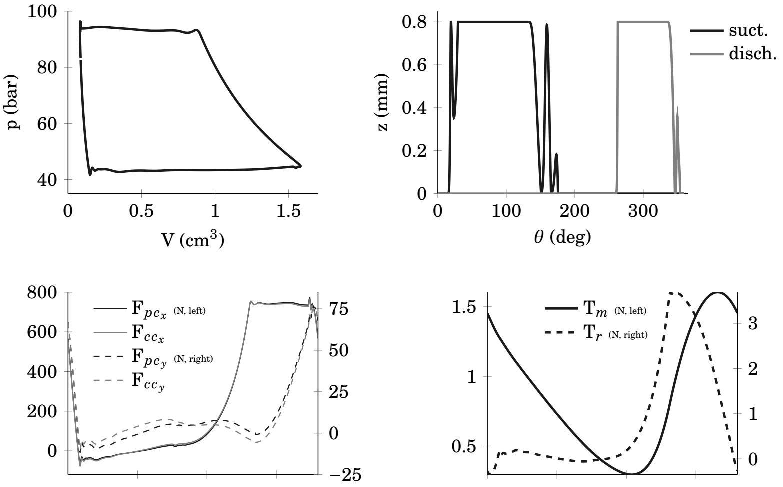


Figure 6.1: Numerical data obtained from a 1.5 cm^3 CO_2 compressor using NEST one-dimensional models.

However, since these models are one-dimensional, their geometry model is too simplified and consequently the effect of the real geometry details on compressor efficiency is not always captured. For instance, the pipes are usually connected to chambers searching an expansion effect of the refrigerant. This expansion depends on the geometries of both the pipe and chamber elements. Although the shape of the pipe is a well-known and simple canonical geometry, the shape of the chamber can be really complex. Even so, one-dimensional mathematical models for solving this flow transitions only depend on the pipe cross sectional area, its length, and the chamber volume. The mathematical model will provide very realistic results of the flow into the pipe. Nevertheless, the resolution of the flow expansion depends on empirical data, which is not always easy to obtain. Furthermore, some of the common assumptions in expansion-contraction flows are not always reasonable in the context of compressor simulations. Finally, the shape of the chamber is not considered in the model, at least the real one, since it is modeled as a cylinder whose inputs are a diameter and a height.

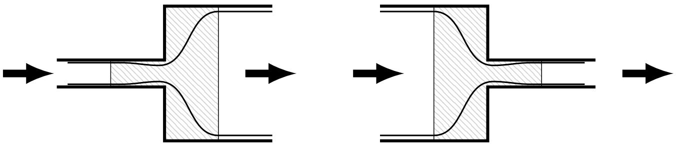


Figure 6.2: Expansion-contraction example.

On the other hand, it is important to note that compressors are particularly complex thermal systems. Not only the time scales of the fluid motion are really small but also all their components are in contact with each other and they are continuously exchanging energy. Moreover, the energy flows in a closed loop due to the presence of the shell which is, excluding the refrigerant itself, the only element that can remove energy to the ambient. Actually, removing heat from a compressor is a difficult matter.

Therefore, understanding the heat transfer details into the compressor device is very important, since the major part of its inefficiencies are linked to thermal effects [3,4]. Both the geometry and the materials of the compressor components (suction muffler, cylinder head, cylinder body, discharge line etc.) are important variables in heat removal. Parameters such as the volume and the thermal diffusivity or the surface size, its roughness and its optical properties are directly related to this process. For this reason, the visualization of the heat fluxes in the compressor components, with real three-dimensional geometries and materials, may help to identify unnecessary effects that produce thermodynamic inefficiencies. In particular, this information could help to reduce the effects of the superheating on the volumetric efficiency of the compressor. Innovative designs of the cylinder head and the cylinder body structures can be analyzed, searching for smart thermal paths aimed to move heat far from the muffler surrounds. For instance, one could ask itself whether the expansion chambers in the cylinder body structure are well positioned or not. By means of multi-dimensional numerical simulation, one could find better positions where to place the chambers.

There are two ways to well understand the thermal behavior of compressor components and its fluid refrigerant in real devices: experimentation and detailed numerical analysis. Experimental techniques, such as the use of thermocouples or infrared thermography may help to obtain compressor temperature maps. However, these techniques are far from easy. They need specific setup for every new prototype, they are time expensive, they are intrusive and some of them lack of detailed information. Moreover, they do not provide information on the heat exchanges between compressor components [3].

Alternatively, numerical analysis is much flexible. It can be applied to different designs, with almost no cost, and most times the effort is portable to other devices and technologies since software developers usually focus their codes to general multiphysics purposes, as in the NEST platform.

6.2 NEST features for multi-dimensional simulations

6.2.1 Multi-model capability

In the simulation of complex systems it is usually interesting to model each component separately by means of a partitioned approach. This brings some advantages from the point of view of software engineering. A big one is the capability to use legacy codes. Another practical advantage is that replacing or removing some component does not affect the remain components. Hence, you are able to arrange various compressor configurations in a fast an very flexible way since there is no need to modify the program code.

The NEST platform is ready for doing this. In the simulation of hermetic reciprocating compressors in particular, the heat transport is assessed by splitting the compressor into several parts. The main separation is between the refrigerant fluid and the solid components. However, both the refrigerant as well as the set of solid parts are divided into some more sub-parts at the same time. In order to illustrate this, in the scheme of Figure 6.3 are depicted the several sub-models used for modeling a compressor. This partitioning is based on the several phases that the fluid is submitted to all along the compression path (suction, muffler, compression and discharge line). On the other hand, the main solid components of the compressor are shown in Figure 6.4. The several parts have distinct shapes and they are made of different materials. The separate treatment of each part permits a better modeling of the heat transfer, since each sub-model can be customized to the particular mathematical and numerical requirements of each part.

Finally, since the model of a particular element depends on the state of their surrounding elements, NEST provides the necessary tools to exchange information between each others. This is implemented in a transparent way, which allows that the users focus their attention on the physical problem instead of spending time understanding coupling issues and software implementation details.

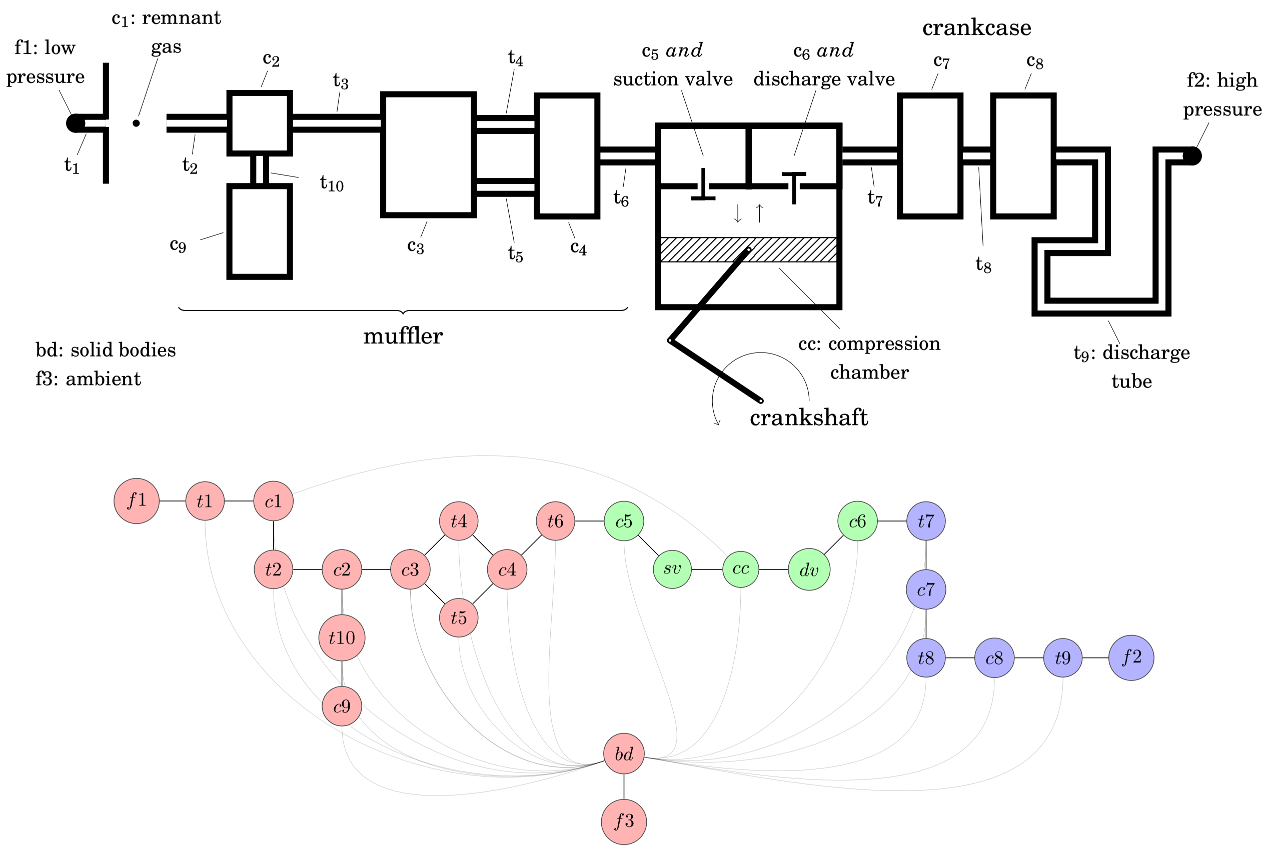


Figure 6.3: NEST model of a compressor and its vertex-edge graph representation. The body vertex represents the set of the main solid components: muffler, crankcase, cylinder head, motor, oil and discharge tube. The distinct colors represent different parallel processes.

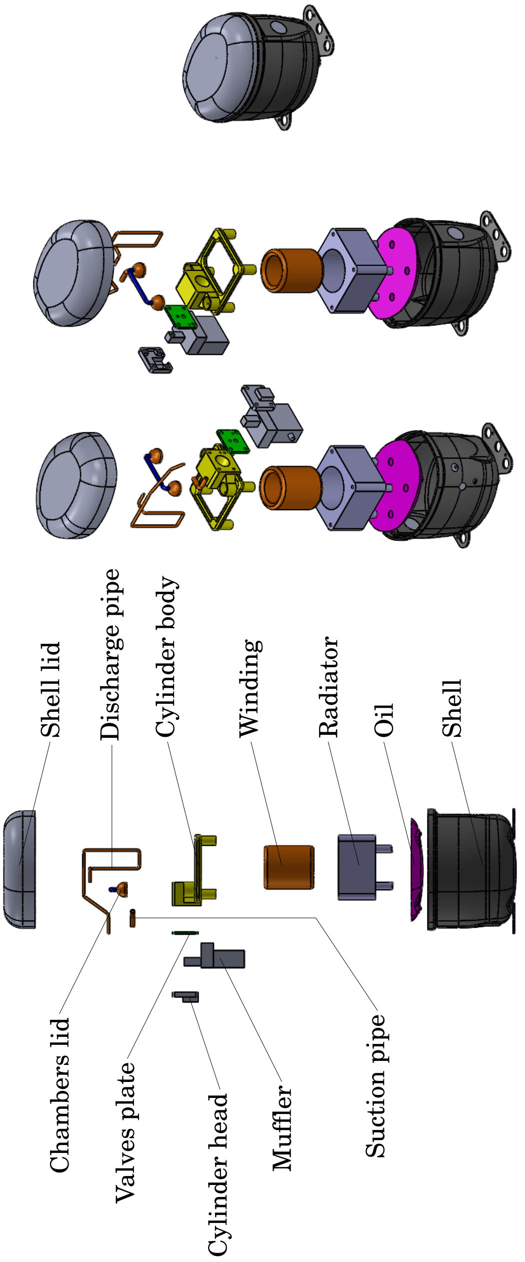


Figure 6.4: Exploited view of the main components of an hermetic reciprocating compressor.

6.2.2 Parallel capability

It is well-known that the computational expenses are very high when detailed simulation by means of CFD&HT analysis is necessary. However, the last advances in parallel computation makes it possible to solve complex industrial problems. Parallel computation is an essential software capability to make use of CFD&HT advances in compressors modeling. NEST software platform is also ready for this. When some model is computational expensive it can be executed in parallel using several processors. Of course this does not affect the resolution of the whole compressor system. Such parts that are solved in parallel are still seen as separate software entities (system elements). This does not alter the data exchange approach either.

Moreover, there is a second parallelization level that allows solving different set of elements in several processors. For illustrating this, in Figure 6.5 is depicted a block-Jacobi based parallel algorithm for solving the overall compressor model. Therefore, the computational load of every model can be efficiently distributed since several elements can be resolved simultaneously.

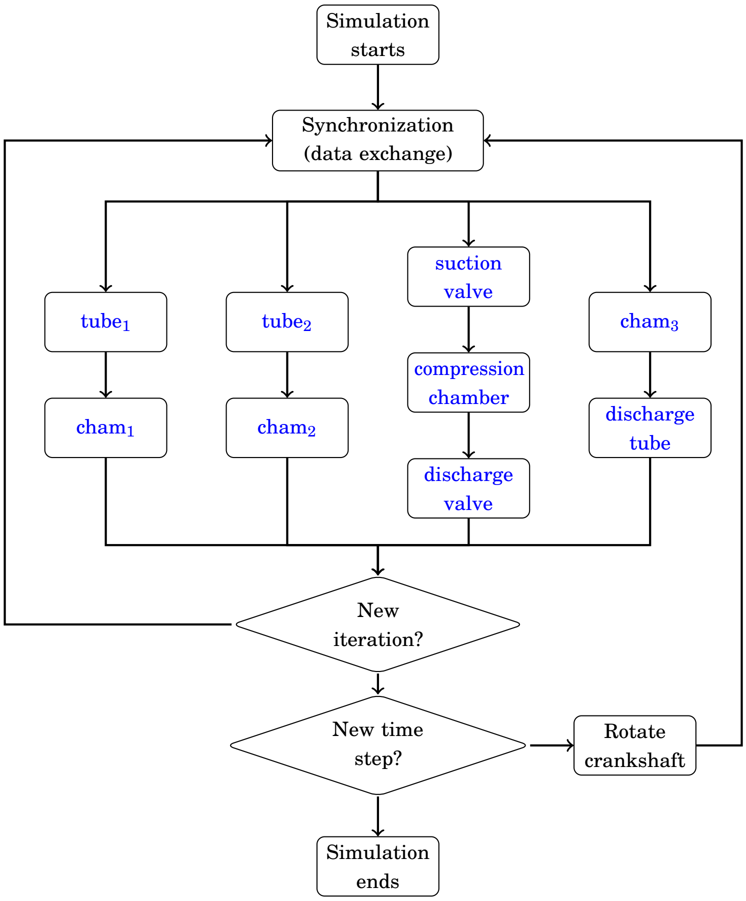


Figure 6.5: Char flow illustrating a block-Jacobi like algorithm. This algorithm, implemented in NEST, can be executed in several parallel processes.

6.2.3 Non-matching mesh adaptation

One important advantage of the partitioned approach is the possibility to use distinct meshes in each multi-dimensional sub-model. This makes the meshing process much easier since each part is treated apart, which significantly simplifies the geometry of the domain to mesh. For example, you can construct orthogonal meshes in the discharge pipeline, whereas the cylinder body structure or the shell can be meshed using an unstructured meshes. In Figure 6.6 are shown the meshes for some of the compressor parts.

An important drawback of this approach is the mismatch of the meshes at the contact boundaries of the components. This makes the data exchange very hard since the data must be transferred properly, assuring accuracy and conservation of the transmitted variables. Moreover, this operation should be fast as it must be performed many times during the simulation. Data transfer is not a straightforward problem. Actually, this is considered a challenging problem in multiphysics simulation [5]. Nevertheless, the data transfer methods implemented in NEST permit the use of non-matching meshes.

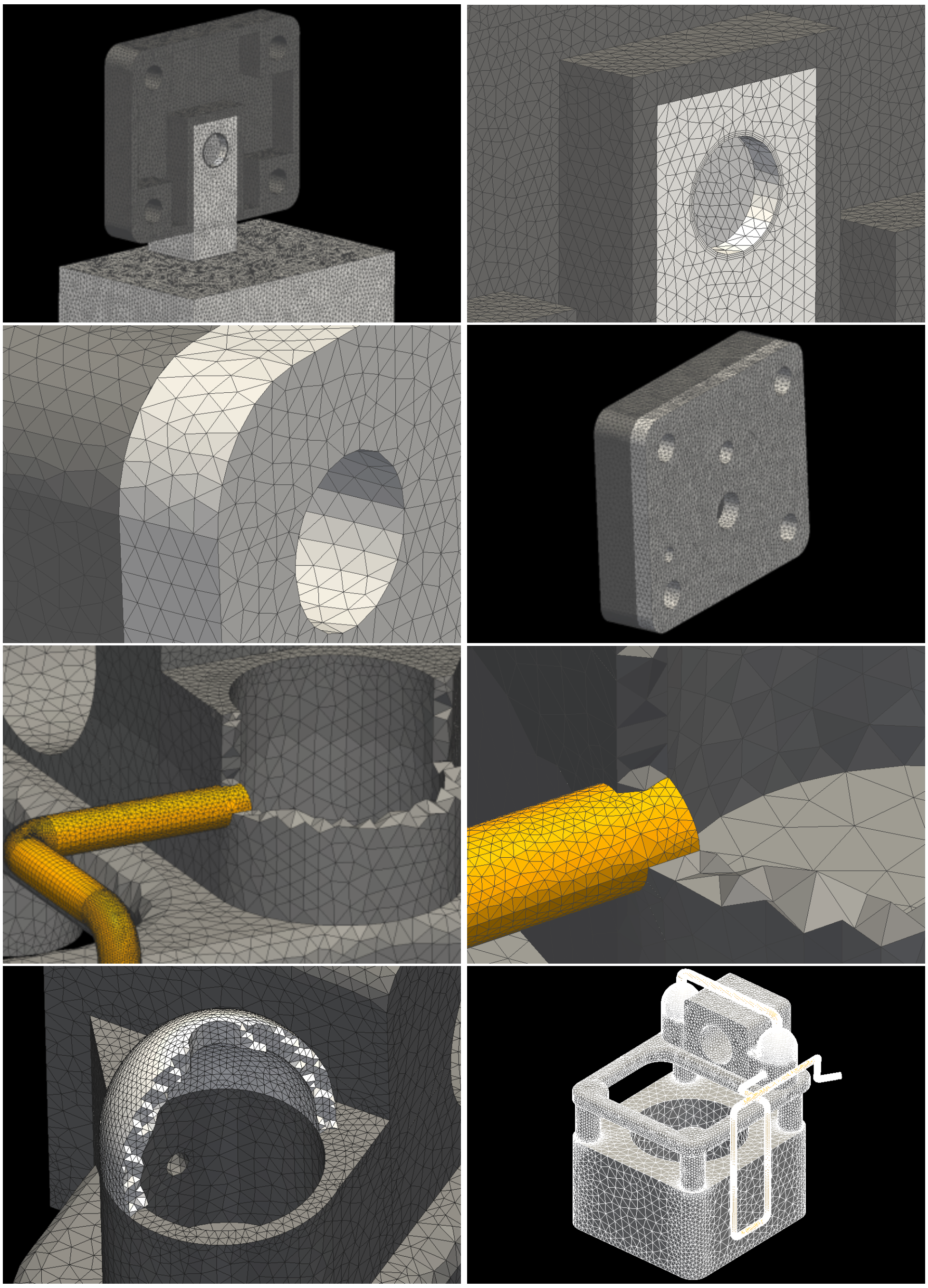


Figure 6.6: Some views of the meshes used in the numerical experiment. Every compressor component is meshed apart so the meshes do not match at the interfaces.

6.3 Numerical experiment

In this section the temperature map in a three-dimensional hermetic reciprocating setup is presented. These data are obtained from the numerical resolution of the heat transfer in the compressor components using NEST software platform. In the background, NEST uses the TermoFluids *DiffusionSolver* [6] for resolving the energy equation in the several computational domains. In order to have a complete view of the temperature distribution and the heat flows, the computational domain is composed of all the components where the fluid refrigerant circulates through as well as those are in contact. To get closer to a real-world compressor the parts are made of different materials, for instance the shell is made of steel and the discharge pipeline is made of cooper.

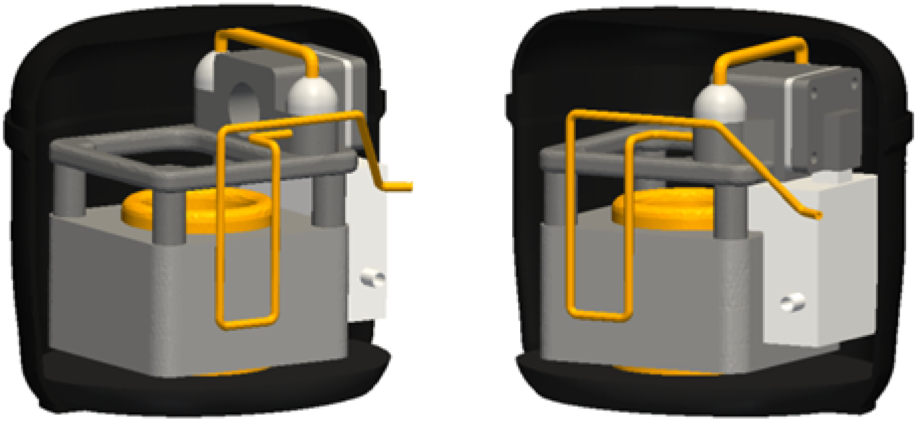


Figure 6.7: General view of the simulated compressor components.

For the sake of simplicity, some elements have been simplified or they have not been considered in the simulation. For example, the valve plate is represented as a pierced plate and the valve stop device is not considered. The program is ready to simulate a more detailed geometry. However, the more detailed is the geometry the more computational expenses. This numerical experiment is not focused on any particular compressor design, but it is aimed to show new approaches oriented to improve compressors design.

Since the overall compressor model is spitted into several sub-models, every component is discretized apart from the others. Hence, there is one discretization mesh per component. In this way, high quality meshes can be obtained easily. Moreover, this makes it possible to modify, replace or remove compressor components without affecting the meshes of the surrounding components. Information about the mesh of each component is presented in Table 6.1.

Mesh	**Cells (x10^3)**	**Partitions**	**Component**
$mpipe_1$	5	1	Muffler
$mpipe_2$	5	1	Muffler
$mpipe_3$	50	2	Muffler
$mcham_1$	350	12	Muffler
$mcham_2$	250	8	Muffler
$cylhead$	170	6	Cylinder head
$valveplate$	120	4	Valve plate
$cylbody$	300	12	Cylinder body
$ckchamlid_1$	30	2	Crankase
$ckchamlid_2$	30	2	Crankase
$ckpipe_2$	10	2	Crankase
$radiator$	120	4	Motor
$winding$	25	2	Motor
oil	85	4	Oil
$shell$	775	20	Shell
$dtube_{beg}$	30	2	Discharge tube
$dtube_{end}$	20	2	Discharge tube
$dtube_{bends}$	170	6	Discharge tube
$dtube_{extrusions}$	70	4	Discharge tube
Sum	2615	96	Compressor

Table 6.1: Details of the several meshes used in the simulation.

The boundary conditions for each of these components are fixed using the Newton's approach $\dot{q} = h(T_{wall} - T_{gas})$. The gas temperature data is based on the refrigerant temperature profile shown in Figure 6.8. This data is obtained from a simulation of a $9.6cm^3$ isobutane compressor using NEST one-dimensional models. The heat transfer coefficients are taken from this simulation as well. They were evaluated using the empirical correlations implemented in the NEST compressors sub-models. The ambient temperature is set at 20C.

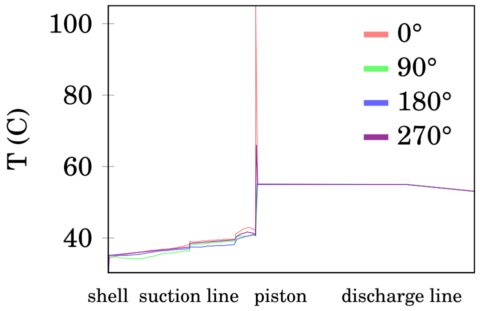


Figure 6.8: Temperature profile of the fluid refrigerant at several crankshaft positions obtained from the simulation shown in Appendix A.

The following figures show several views of the compressor temperature map. One can clearly identify the cold (dark color) and hot (light color) areas in the general perspective views of Figure 6.9. As expected, the shell is the coldest part while the cylinder surround is the hottest one. Other interesting effects can be observed here. For instance, the chambers in the cylinder body structure are heated by remnant heat flows coming from the cylinder surface. This is a desired effect since the heat is dissipated from the cylinder surrounds to colder parts of the cylinder body structure, which contributes to create a colder environment around the suction muffler. This is very important since the volumetric efficiency can be reduced significantly if the refrigerant is too heated before entering into the cylinder. Furthermore, the backside of the cylinder body structure is about 15C colder than the cylinder surround (see left-right views in Figure 6.10). This indicate that there is room for intensifying heat removal from the cylinder. On another hand, the images also show how the heat is transferred to the shell first through the radiator and later through the oil pool. Actually the top part of the shell is colder than its bottom. This means that the heat is mainly dissipated thanks to the oil, being the heat convection though the remnant gas not so relevant. In addition, the simulation also shows how the discharge pipe colds down as it is closer to the shell. This temperature gradient indicates that this pipe is also a heat removal path. However, one can not forget that the refrigerant leaves the compressor through this pipe. Hence, the state of the discharged gas depends on what happens along this pipe.

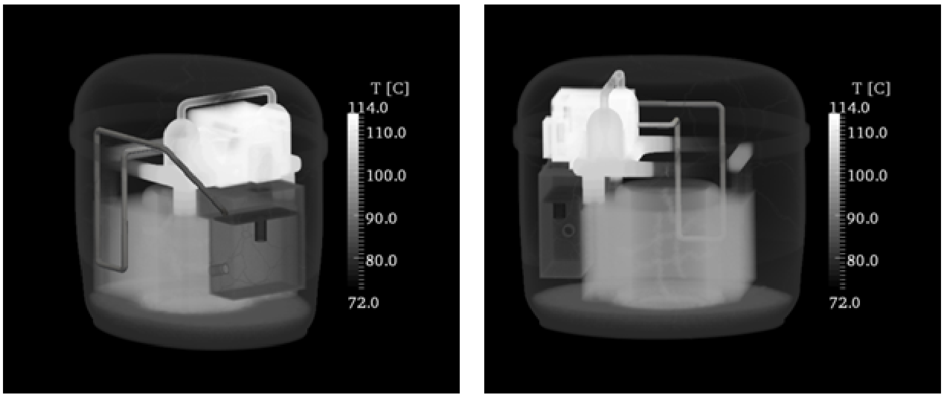


Figure 6.9: Temperature distribution in the compressor setup. The optical properties of these images have been modified in order to emphasize specific temperature levels. Due to this rendering some temperatures are not appreciated.

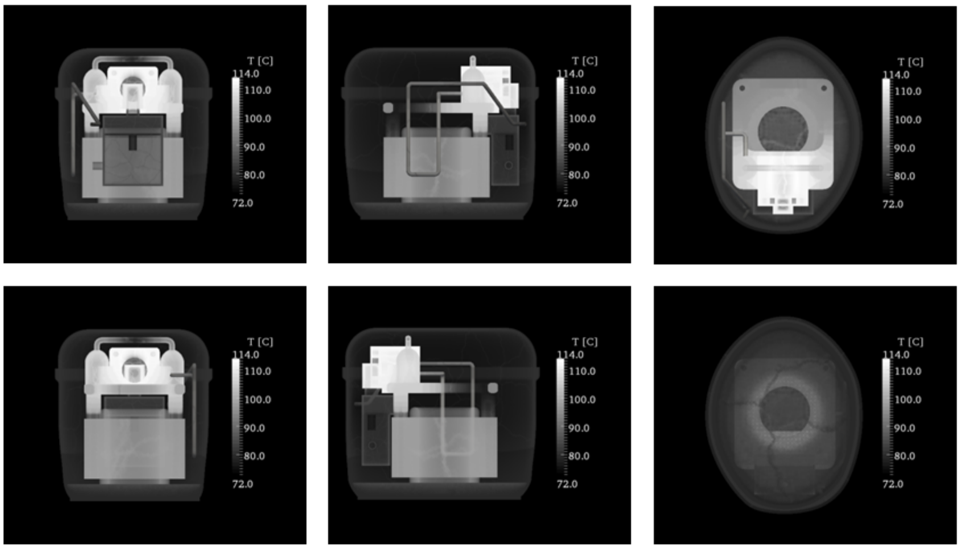


Figure 6.10: Temperature distribution in the planar views of the compressor setup. Front-back (left), left-right (middle) and top-bottom (right).

Moving towards more lightweight geometries of the cylinder body structure could help to reduce the accumulated heat in the solid parts. Indirectly, this could improve volumetric efficiency since it would reduce the temperature of the refrigerant in the suction area. In a similar way, the choice of the employed materials is also critical. In order to show the effect of changing the suction muffler material, a second simulation has been carried out using exactly the same geometry and boundary conditions. In this case, however, the suction muffler is not made of plastic material as usual. Instead, it is done of a hypothetical material, whose thermal diffusivity is ten times the one of the plastic muffler. Figures 6.11 and 6.12 show several comparative views of the obtained numerical results. Remember that in these views, the darker is the color the colder is the material. The main difference relies on the temperature, when the thermal diffusivity is low the muffler is colder. Therefore, this numerical experiment shows why suction mufflers are made of plastic.

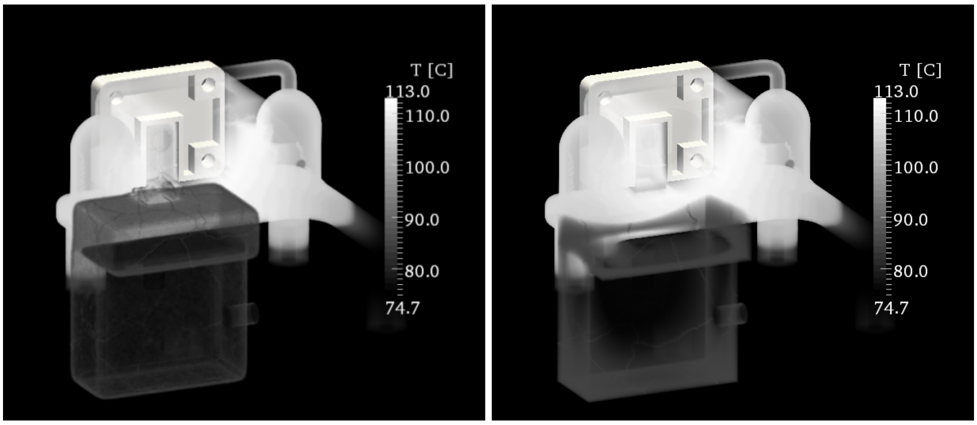


Figure 6.11: Temperature distribution in general views of the suction muffler and its surrounds. Low (left) and high (right) thermal diffusivity.

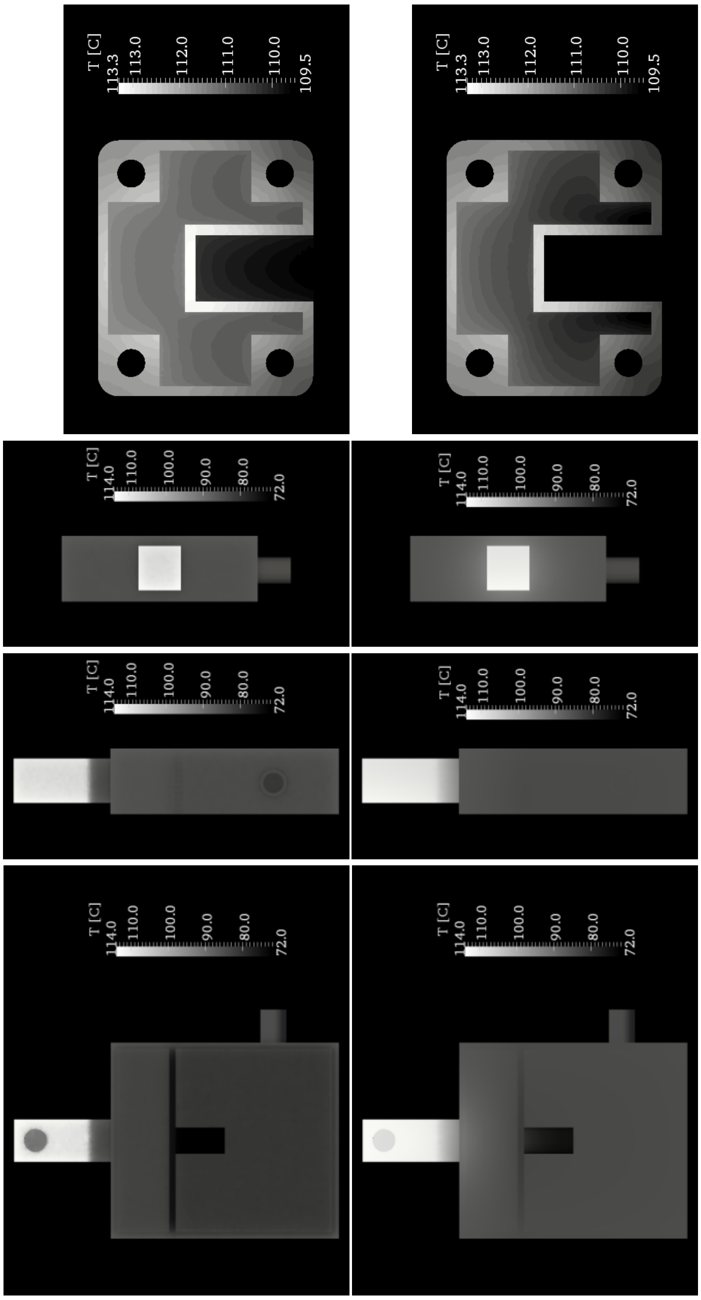


Figure 6.12: Temperature distribution in the planar views of the suction muffler and the cylinder head. Lower thermal diffusivity material (upper row) and higher thermal diffusivity material (lower row).

Figure 6.13 shows a better perspective of both suction mufflers. In these figures it is easier to see that the outlet pipe of the muffler is at different temperature. In this zone, the heat transfer between the refrigerant and the muffler walls is intensified and it is much important than the one in the previous chambers. This results show that the muffler discharge is an important design consideration.

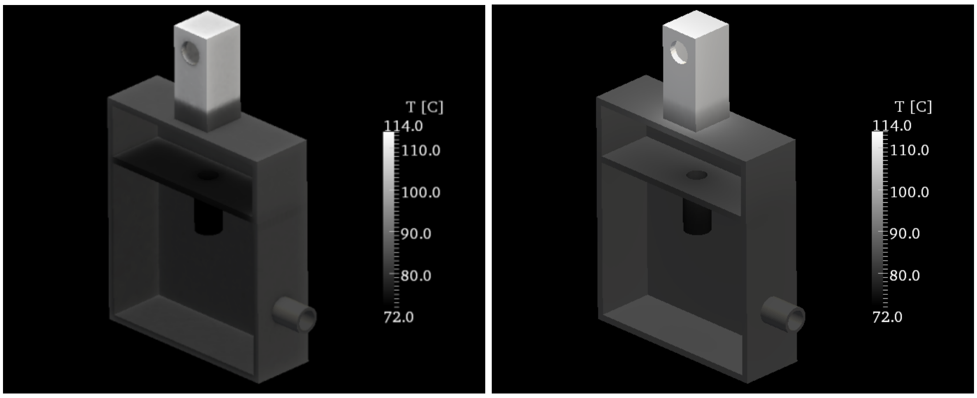


Figure 6.13: Temperature distribution in perspective views of the suction muffler. Low (left) and high (right) thermal diffusivity.

6.4 Conclusions

The NEST platform, using TermoFluids [6] software in the background, has been used to solve the heat conduction in an hermetic reciprocating compressor.

Several NEST capabilities have been tested in this simulation. First, the heat transfer has been analyzed in the whole set of compressor components by means of a partitioned approach. This involves the use of several instances of the TermoFluids *DiffusionSolver*, which exist simultaneously while they are executed on several parallel processes. Using distinct instances, has made it possible the use of different meshes and materials. The employed meshes are based on geometric particularities of each component. Consequently, they do not match at the contact boundaries, which permitted to test the NEST data transfer tools also.

Various three-dimensional numerical results of the temperature distribution in the compressor components have been presented and discussed. The obtained numerical results are in good agreement with the expected thermal behavior of a compressor.

Although this simulation is far from being a novel contribution to compressors design, it proves that the NEST platform deserves further research.

This research can be focus on solving the convection of the remnant fluid refrigerant around the solid components using coupled CFD&HT. In addition, many other compressor details should be considered in this kind of simulations. Some are still far to achieve in numerical simulation, like the oil recirculation or those related with noise generation such as motor vibrations. Some others are already possible, although they make the simulations more computationally expensive. For example, thermal radiation is an important factor in the energy balance and could be introduced by resolving the RTE numerically. Some other details rely on the geometry and the employed materials. Innovative designs pursuing a general cold down of the compressor components could be investigated. This could be focused on many details. For instance, the joining point in the discharge line at the shell side is quite simplified in the simulation model. In real-world compressors this join is usually mechanically fastened using cooper rivets. This is an important consideration because it alters the heat transfer to the shell and the ambient.

In sum, the advances in high performance computer and consequently the new possibilities in CFD&HT are not the definitive solution in the improvement of industrial devices. There is still much work to do in this research area and today's simulation tools must be used carefully. Well-thought numerical experiments are necessary to provide substantial information on designs defects and suggest novel enhancements. While the one-dimensional models lack of detail in the provided results, CFD&HT models are still too computationally expensive for using them in industrial problems. The combination of both one-dimensional models – in a low level - and three-dimensional models – in a high level - is the workaround to this problem. When this is possible, the numerical results such as the ones presented in this chapter can be complemented with estimations of the COP and hence, with an overall compressor performance analysis.

References

[1] Perez-Segarra et al. Detailed thermodynamic characterization of hermetic reciprocating compressors. *International Journal of Refrigeration*, 28:579–593, 2004.

[2] J. Rigola et al. Parametric studies on hermetic reciprocating compressors. *International Journal of Refrigeration*, 28(2):253–266, 2005.

[3] B. G. Shiva. Heat transfer in reciprocating compressors - a review. In *International Compressor Engineering Conference*, Purdue, IN, USA, July 1998. Purdue e-Pubs. Paper 1349.

[4] F.A. Ribas et al. Thermal analysis of reciprocating compressors - a critical review. In *International Compressor Engineering Conference*, Purdue, IN, USA, July 2008. Purdue e-Pubs. Paper 1907.

[5] X. Jiao and M.T. Heath. Common-refinement-based data transfer between non-matching mehses in multiphysics simulations. *International Journal for Numerical Methods in Engineering*, 61:2402–2427, 2004.

[6] O. Lehmkuhl, C.D. Perez-Segarra, R. Borrell, M. Soria, and A. Oliva. Termofluids: A new parallel unstructured CFD code for the simulation of turbulent industrial problems on low cost PC cluster. *Lecture Notes in Computational Science and Engineering*, 67:275–282, 2009.

Chapter 7

Conclusions

7.1 Contributions

The main contributions made in the thesis are listed below.

- Development of a methodology to solve complex problems involving several physical phenomena simultaneously. Using distinct details of modeling is enabled.
- The above methodology has been implemented as an object-oriented software platform for fast production of reliable multiphysics applications. The distinct integrated models are separate code entities that conform a global simulation program. There is the possibility to use legacy codes as long as they permit fixed-point iterations. There is some experience embedding Python codes, too.
- Development of a strategy for executing the several integrated codes in parallel. Moreover, each code can be executed in parallel for fulfilling particular numerical requirements.
- Development of tools, based on different known methodologies, for enabling data exchange between the codes. Special emphasis is put on accuracy and conservation of the transferred variables. The biggest challenge here is to couple two domains using different spatial discretization so that their meshes do not match at the contact boundary. The tested methods are compatible with the parallel approach. One of them, the Weighted Residuals, performs better in parallel.

- Update former applications for simulation of general buildings and hermetic reciprocating compressors to take advantage of the NEST features. The potential of such computational tools is illustrated in both cases.

7.2 Concluding remarks

This PhD thesis has been focused on the development of the NEST platform. This software is aimed to simulation of multiphysics through the integration of distinct physics codes. These codes model the different involved phenomena using the appropriate mathematical and numerical approaches. All this is addressed as a long term software project, using well though out strategies for reducing code complexity so that the code maintenance is possible over the years.

The NEST platform is designed on the basis of the Object Oriented Paradigm (OOP). In this fashion, the code is written in a more natural way so its instructions are closer to human reasoning than to the computer logic. On the one side, the use of OOP has several appealing features related to the management of software projects. On the other side, this is a practical way to create computer simulation programs based on partitioned approaches. In this regard, NEST is provided with several data transfer tools for coupling properly the several integrated codes. Furthermore, NEST is also ready for executing all the instantiated physics components simultaneously. At the same time, such components are able to run in parallel, which is essential to take advantage of the recent advances in CFD&HT.

The NEST platform is build on the top of several more specific software libraries that distribute tools and algorithms to solve fluid dynamics and heat transfer problems. The whole software framework has been described with special emphasis on the Modular System Library (MSL). This library, which constitutes the kernel of the pursued computer simulation programs, sets the fundamental logic in system environments and establishes a natural language for its management.

In addition, the several possibilities offered by the NEST platform have been shown by means of several example applications. Specifically, a program for assessing buildings performance and one for simulating hermetic reciprocating compressors have been presented. Both programs are based mainly on the use of zero and one-dimensional models of the distinct phenomena involved in each case. Some of these phenomena are common to both applications. On that case, their models are implemented in a general way in order to reuse them in the two programs. For instance, the resolution of the transient heat and mass transfer is implemented only once. Apart from this, the available building models are aimed to simulate rooms, walls, openings such as windows and doors, ventilation, infiltrations, HVAC equipment, events and occupancy management and so on. Apart from this, the available compressor models are aimed to the simulation the refrigerant intubation, resonators, expansions and contractions, moving boundaries (e.g. valve motion or compression chamber), kinematic and dynamic of the rigid bodies, etc. All these models can be linked to each others to conform a general model of the overall problem (i.e. a house, an apartment or any type of residential or non-residential building or, also, a compressor device). The compressors application in particular, has been numerically verified and experimentally validated. This assures the quality of the numerical solution and make the code more reliable.

Finally, the heat transfer on the main components of an hermetic reciprocating compressor has been assessed using a three-dimensional diffusion transport *solver*. Particularly, since each of these components has been modeled separately, this simulation has shown how the NEST platform can be effectively used to couple several instances of a multi-dimensional code. For attaining an accurate and conservative coupling of the various compressor components, the data transfer tools for transferring data between non-matching meshes, also developed in this thesis, have been applied here. Moreover, each one of the several *solver* instances has been executed in multiple parallel processes. This has made possible to test the parallel capabilities of NEST.

7.3 Future actions

The objectives that were initially fixed at the beginning of this thesis have been successfully fulfill. However, some ideas for doing further research are exposed below.

The NEST platform deserves more attention on the parallel approach. So far, this issue has not been the main concern as long as the integrated codes can be executed simultaneously in several parallel processes. At this moment, every one of these parallel processes is organized in a *root-rank* strategy. This means that the interaction of the physical components -and their partitions- with their neighbors is managed through a unique computer process. Although this is an easy solution for code implementation, the *root-rank* approach may lead to important data communication bottle-necks. Hence, from the performance point-view, this is not the most optimal design.

Furthermore, multiphysics not only entails several spatial discretizations but also different time discretization demands due to the mixed scales of time. You may take as an example a domestic refrigerator. This is a complex vapor compression based system, composed on four basic equipments (condenser, expansion device, evaporator and compressor) that are, in turn, other complex systems each of different nature. The compressor in particular, as it has been shown in this thesis, requires very small time steps for solving it well. Its time discretization strategy is not necessarily the optimum for the other components. Addressing problems of such detail is really challenging. Using the same time step for advancing time in the distinct temporal discretization schemes is very practical but it is highly inefficient, too. For this reason, more research is required on efficient integration of distinct time stepping techniques. This will let the several models to use their own time step in the basis of local requirements, instead of imposing an unique overall value. For now, the first attempts on this matter are focused on the creation of nested systems. In this way, a NEST system can be composed of a mix of elements and sub-systems that behave like standard elements. These sub-systems, as such, are able to work using different time steps, although they must be smaller that the one of the container system. However, this approach has several synchronization constrains and it is still too naive so that it needs more dedication.

Also, the global resolution algorithms implemented currently in the NEST platform are limited to block-Jacobi and Gauss-Seidel based methodologies. Further work should be done in the assessment of alternative non-linear systems solvers, such as those who are based on Newton's method or genetic algorithms. The advances in this direction will help first to better understand the behavior of the already used algorithms and later to speed up the simulations. The second is important in the attainment of optimal designs of dwellings, other type of buildings or any industrial device.

Despite the efforts on the presented data transfer tools, more research is necessary to decide what methods are the most appropriate to simulate conjugate heat transfer, aerodynamics applications or multiphysics in general. Among these methods, the Weighted Residuals formulation seems to be the more appropriate in NEST simulations. However, this method requires computationally expensive pre-processing geometric operations to calculate the intersection and projection of the two meshes in contact. Although a tool for doing this operation has been developed, it should be improved to make it more consistent first and faster, later. Both, robustness and speed will be necessary to pursue Fluid Structure Interaction problems (FSI) using NEST. Again, the parallel performance is very important here and it should be analyzed in detail.

In regards to building performance simulation, it is important to do some numerical verifications, in the same way it has been done for the compressors program. The correctness of the mathematical models should be checked, too. Even though the CTTC researchers have previously validated these models in former software projects, this is still useful for verifying the new NEST based code. Within the buildings research literature, there are several well-known reference cases called BEST-TEST that aim towards this goal. Once this is done, the NEST building models can grow in many directions (coupling of multi-dimensional models with zero or one-dimensional models, space optimization, assessment of energy consumption, lighting, etc.). This depends on the applications that will be faced in the near future. In any case, the modeling of the event management and the presence of the occupants in buildings deserves more attention in both their code design and their theoretical basis.

Finally, regarding to the use of NEST in compressors simulation, it would be interesting to quantify the compressor performance (COP) when one or several of their components are simulated by means of CFD&HT analysis. For example, in the numerical experiment of the last chapter, it would be interesting to see the effect of the muffler material on the overall performance of the compressor. To do so, the multi-dimensional codes must be well-coupled with the zero and one-dimensional models, first. Moreover, due to the small time steps required in the modeling of the fluid dynamics, the coupling of the fluid performance and the heat transfer in the components will be very time consuming. Therefore, before addressing this level of simulation the data transfer tools must perform very well in parallel execution. In addition, the data transfer problem is defined between canonical and amorphous geometries. These geometries are discretized using regular and unstructured meshes respectively. This is still not considered in the available tools.

In sum, there is much research to be done in NEST simulations. Some of the work is focused on the NEST platform itself. Other work relies on the improvement and extension of the single physics modeling. However, NEST is a general purpose code designed to grow, so that physical phenomena that is not planned today can be integrated tomorrow.

Appendix A

Detailed analysis in hermetic reciprocating compressors

Simplified zero and one-dimensional models are very good for obtaining overall information about the performance of an hermetic reciprocating compressor. However, for understanding the flow behavior inside the components you must focus on detailed analysis. There are some regions of special interest that need for more insight to improve them. For example, the flow in the suction area is usually separated into two parts, the one that directly flows to the muffler and the one that expands to the shell. Although this affects the flow pressure and the fluid temperature, there is few information on the optimum configuration (e.g. entrance-muffler distance, shape of the muffler nozzle, cold-hot gas mixing ratio, etc.). Furthermore, the muffler element is essential for reducing noise, but it is a flow impediment that produces pressure drop and consequently it also affects performance. Hence, studying the refrigerant flow in some parts of the compressor, using CFD&HT, could provide valuable information.

Below are shown the results from different simulations that illustrate the use of detailed numerical analysis combined with other one-dimensional models using NEST. The results in the first section correspond to the simulation of the remnant refrigerant fluid in the shell of the compressor. The results in the second section correspond to the refrigerant fluid flow inside the muffler. In both cases, a Low-Mach based TermoFluids *solver* is employed for modeling the fluid flow in the indicated components. Particularly, a compressible TermoFluids *solver* is used in the muffler analysis, too. On the other side, the remain components of the compressor are simulated using simplified models in order to fix realistic boundary conditions. In this fashion, the multi-dimensional calculations are adapted at every time step to the instantaneous position of the crankshaft, the piston and the valves so that the transient behavior of the downstream fluid flow is considered.

It is interesting to highlight the importance of this type of analysis. First, in the study of compressor/system components of special interest, the multi-dimensional approach provides detailed information while the zero and one-dimensional models provide transient boundary conditions to the simulation. Second, the use of simplified models reduce the number of unknowns along the compressor so that simulations are faster and the global behavior of the systems can be studied as well. Thus, the parallel object oriented tools for multi-level modeling have been an important aspect in this PhD thesis and are under constant progress, while the whole detailed simulation is not possible.

A.1 Numerical analysis of the remnant gas

A.1.1 Computational domain, mesh and boundary conditions

In Figure A.1 is shown a general view of the computational domain. It is defined by the fluid that occupies the volume between the two edge-rounded cylinders. The inner cylinder (4) represents the set of internal components (i.e. the crankcase, the motor, the compression chamber, the suction muffler and so on), the external cylinder (5) represents the compressor shell and the input (1) and the output (2) orifices are the compressor and the muffler inlet respectively. The gap between these orifices is known as the suction area. Part of the refrigerant is driven directly to the muffler and the remain of the gas circulates into the shell (i.e. the enclosure between (4) and (5)). The discharge tube (3) is also considered in the computational domain. The dimensions are set based on a real compressor.

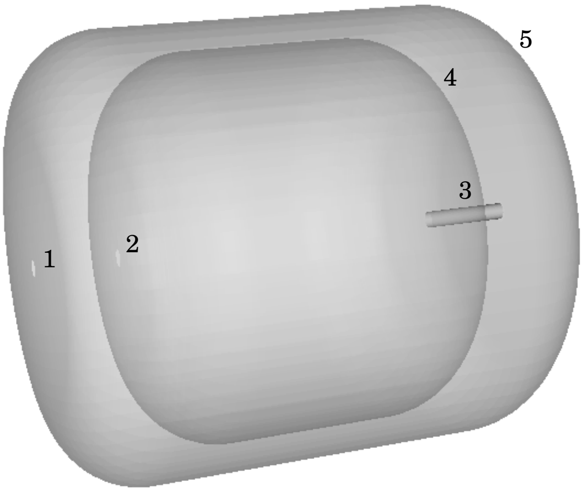


Figure A.1: Isometric view of the computational domain: suction (1), muffler inlet (2), discharge tube (3), inner cylinder (4) and outer cylinder (5).

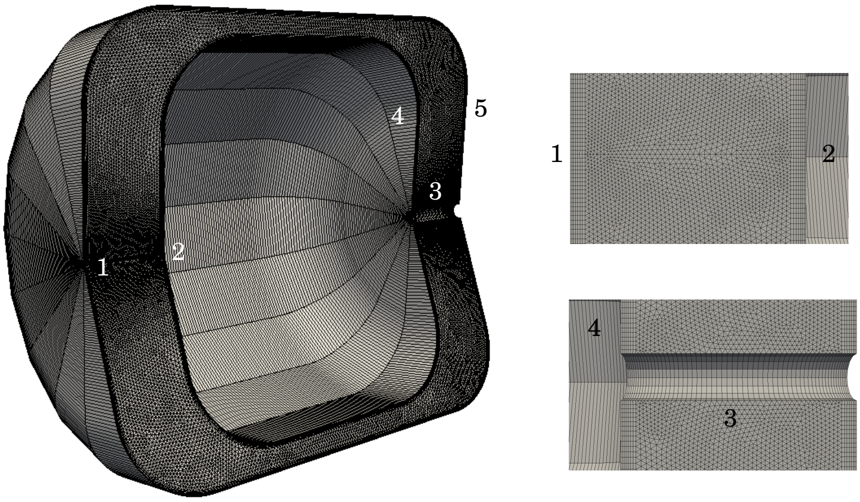


Figure A.2: Isometric and zoom views of the unstructured mesh: suction (1), muffler inlet (2), discharge tube (3), inner cylinder (4) and outer cylinder (5).

In Figure A.2 are depicted the discretization mesh and some zoom views. This mesh is constructed from the cylindrical extrusion of a two-dimensional mesh of the half domain (with 16 planes from 0° to 360°), which explains its symmetry. This type of mesh allows the use of Fast Fourier Transform (FFT) solvers so that a significant reduction of the computational cost is obtained. The mesh is concentrated in the suction area as well as on the walls by means of a prismatic layer. The resulting mesh has about 250000 CVs.

The model for the remnant gas is connected to a set of simplified models that establish dynamic boundary conditions to the *solver* domain. Velocity and temperature are given at the input orifice (1) while pressure is fixed at the output (2). This pressure is calculated at every time step to adapt it to the instantaneous position of the crankshaft. The temperature and the mass flow on the multi-dimensional *solver* side is averaged at each instant to update the boundary conditions of its one-dimensional neighbor model. Hence, a point to point connection is enough for coupling both levels of modeling. Initially, the entire fluid is at the same temperature. The walls are set to Dirichlet boundary condition and no-slip conditions. The outer cylinder surface (5) is set to 40C (cold wall), the inner cylinder surface (4) is set to 50C (hot wall) and the discharge tube temperature is given by a lineal function between the hot and the cold walls.

The refrigerant fluid is isobutane (R600a) working at ASHRAE standard conditions of $T_{cond} = 55$ C and $T_{evap} = -23.3$ C, while the suction and discharge pressures are $p_{suct} = 0.6265$ bar and $p_{disch} = 7.726$ bar ($\Pi = 12.33$). The thermo-physical properties of the isobutane are obtained from a pre-computed database accessed though the substance enthalpy and pressure.

A.1.2 Numerical results

The large number of CVs and the complexity of the phenomena involve the use of parallel computing. The detailed numerical data was obtained using four nodes from an old version of the CTTC-JFF cluster for three weeks. Each node had 8 cores so that the simulation run on 32 CPUs. In addition, an extra node was used for simulating the coupled one-dimensional models.

The results show mixed natural and forced convection at high Reynolds number. A lot of turbulent scales are observed in the instantaneous contour maps shown in Figures A.3 and A.4. The refinement of the mesh on the walls is critical for solving well the viscous effects at the boundary layer. The jet in the suction area evolves very fast producing two main eddies at both sides (upward and downward) like in a jet impingement. These eddies are in contact with the hot wall (4) so that they are heated as their distance from the main jet increases (the color evolves from light blue to green). The eddies stop their translation when they reach the top and bottom of the domain. At this point, the fluid in the suction area begins to warm up. Concurrently, the temperature also increases at the top and bottom gaps and later the heat is transferred towards the right side of the domain. Initially, the heat flow on that side is mainly dictated by diffusion (red to blue transition on the inner wall). This is probably due to the low temperature difference between the inner and the outer cylinders. After several instants, the thermal boundary layer on the discharge tube becomes detached, which contributes to warm up the surroundings of the tube (most of the domain is green). Despite the entire domain is heated, the suction jet remains cold for the whole simulation.

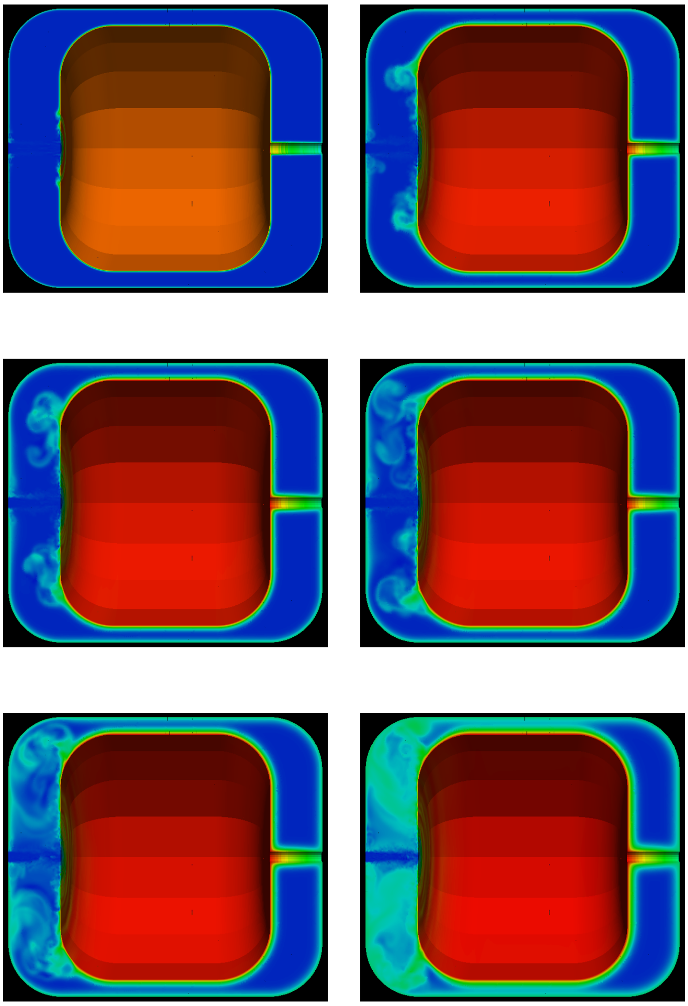


Figure A.3: Temperature field at the first time steps of the simulation. Cold regions are identified in blue while hot regions are in red.

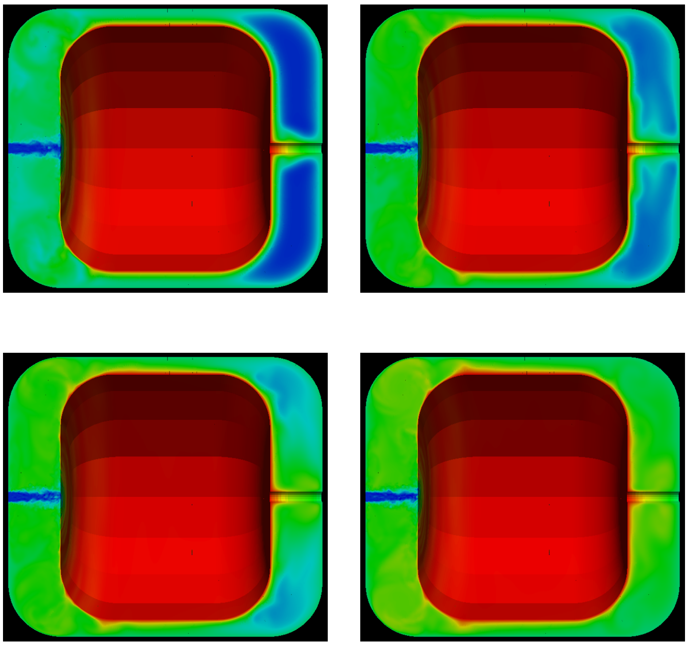


Figure A.4: Temperature field at the first time steps of the simulation. Cold regions are identified in blue while hot regions are in red.

Figure A.5 shows pressure, mass flow and temperature evolution during one cycle at the inlet of the muffler.

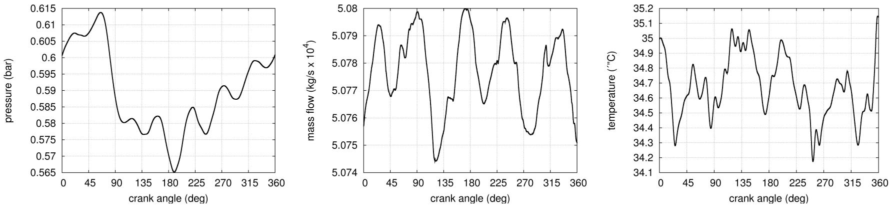


Figure A.5: Pressure, mass flow rate and temperature profiles at the muffler inlet.

As expected, the pressure evolution in this area is highly oscillatory. This is also appreciated in the mass flow and temperature data. While the pressure is the same at the beginning and at the end of a crank rotation, the other variables are still in transient evolution. This is in accordance with the time-averaged data shown in the charts of Figure A.6. This charts show the evolution of the discrete time-averaged values of pressure, temperature and velocity. For obtaining this, the compressor cycle is divided into 10 time-blocks to average various cycle pieces separately. In this way, the perturbation of the piston frequency on the variables is decoupled from the turbulence phenomenon. Each cycle is bounded by vertical lines. After achieving steady cyclic conditions the bars pattern should repeat itself. The left column charts depict the variables at a specific position in the suction area, while the right column charts depict the variables at a specific position around the discharge tube. These data show that the flow is still transient in the discharge area while it is quite more stationary in the suction jet. This is due to the natural convection phenomenon on the back side of the compressor, which is slower.

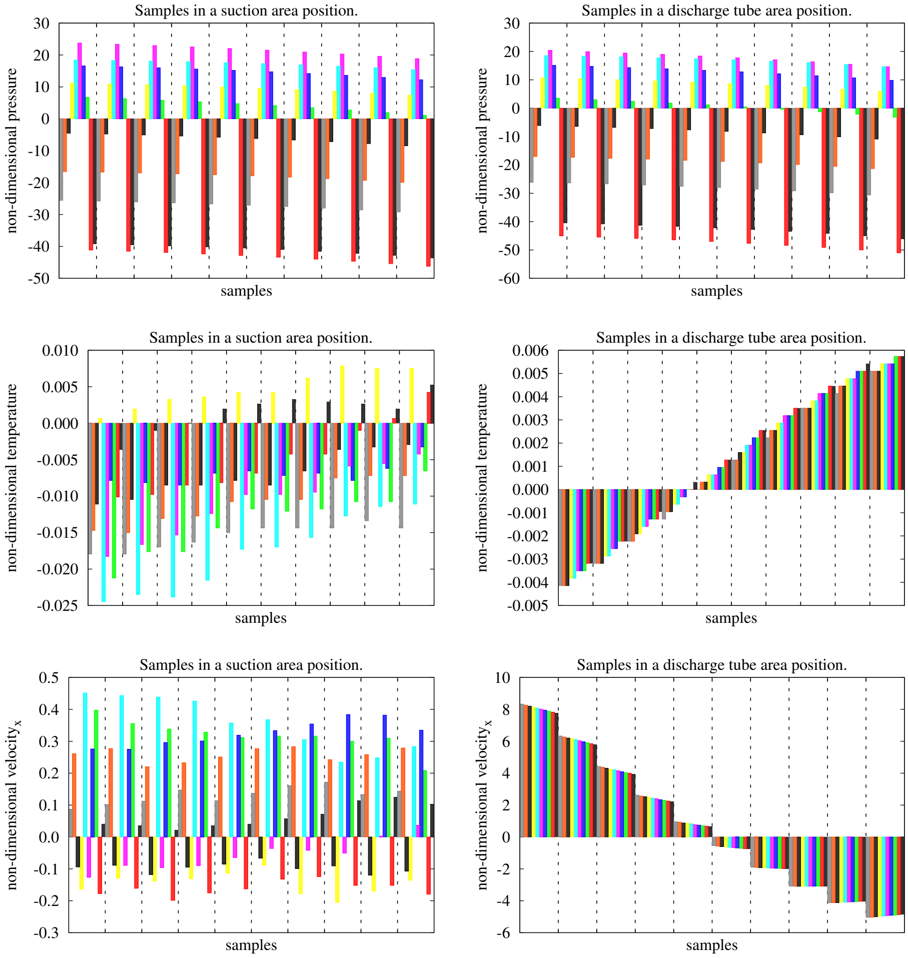


Figure A.6: Pressure, temperature and velocity (x component) time-averaged samples at two different positions during 10 cycles.

In Figure A.7 are presented the temperature profile of the fluid along the compressor components at four crank angles (left) and the pressure-volume diagram in the compression chamber (right). First, the refrigerant is heated during its suction in the muffler. Here, the evolution of the temperature is weakly dependent on the crank angle. In exchange, there is a high temperature peak when the gas is compressed inside the cylinder. Despite the use of the CFD&HT solver, the PV diagram has the expected shape and pressure ratio. The pressure peaks during the discharge stage appear due to the valve reed characteristics.

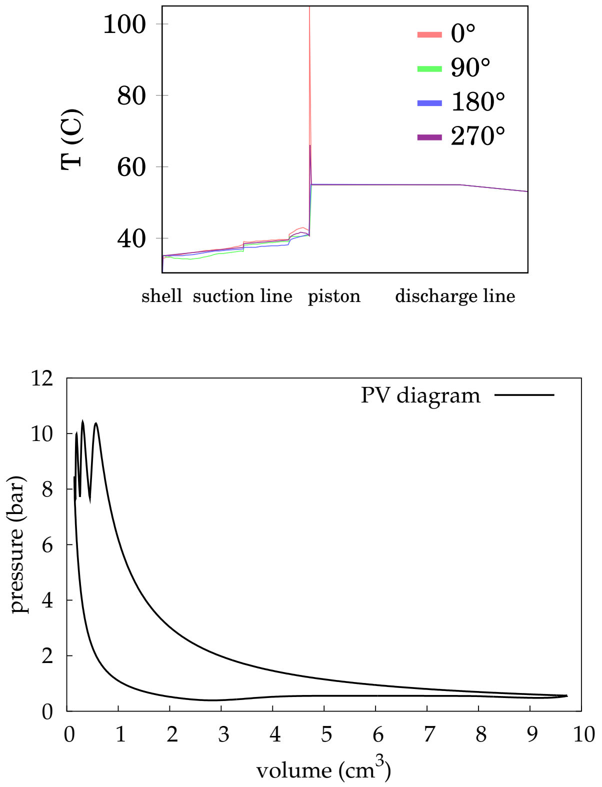


Figure A.7: On the upper part, temperature profile of the refrigerant fluid along the compressor (from muffler to discharge tube) for several crankshaft positions. Below, the PV diagram of the compressor. These data are calculated in the one-dimensional models.

A.2 Numerical analysis of Suction Mufflers

A.2.1 Computational domain, mesh and boundary conditions

In Figure A.8 are shown two different muffler geometries intentionally designed for comparison purposes. Both mufflers are composed of two main chambers. The first chamber (1) height is 2.7 times the second one (2), while the width of both chambers is twice their thickness. In the left side muffler (A), the refrigerant expands into the first chamber after entering through the inlet pipe (3). Then, the gas flows to the second chamber through a short pipe and later it leaves the muffler through the exit pipe to discharge into the compression chamber (4). The shape and dimensions of the right muffler (B) is exactly the same as the left one (A). However, the inflow is intubated towards the secondary chamber, in this instance. There is only a small gap at the entrance for enabling the gas to expand for attenuating pressure perturbation. Hence, these academic configurations permit to show the influence of direct versus indirect flow from the muffler inlet (3) to the suction valve (4).

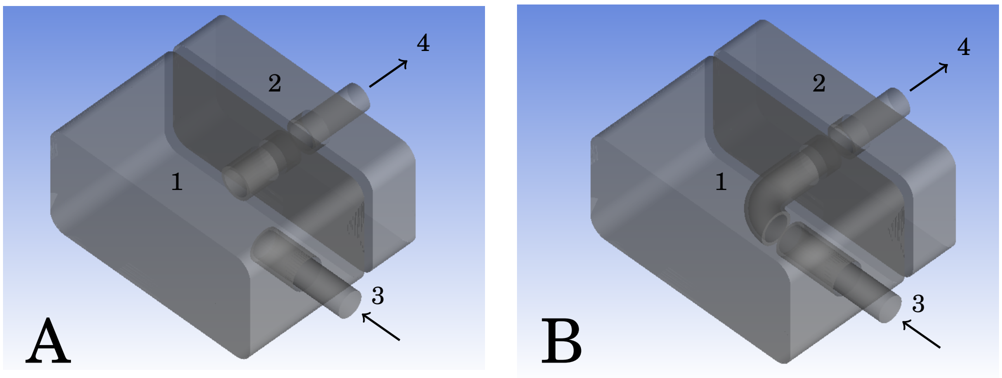


Figure A.8: Muffler geometries: indirect (left) and direct (right) fluid flow from first chamber to second chamber.

Both configurations present a mesh depicted in Figure A.9. The mesh consists of tetrahedral cells exclusively. During the meshing process, special attention has been given in the regions where steep gradients are present (jet path, tubes interior etc.), while a coarser mesh is used in regions where the flow is dictated mainly by natural convection. Indicatively, the average density of the meshes under consideration ranges between 5 and 10 cells/cm^3 approximately.

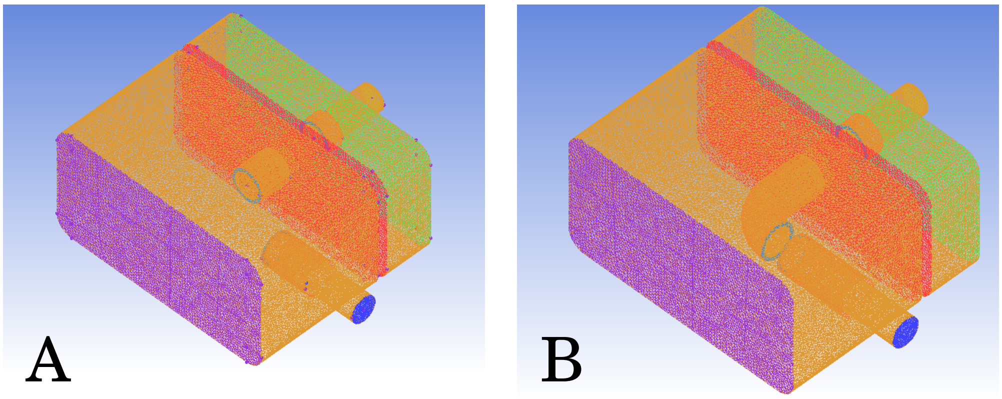


Figure A.9: Isometric view of the mesh for both configurations.

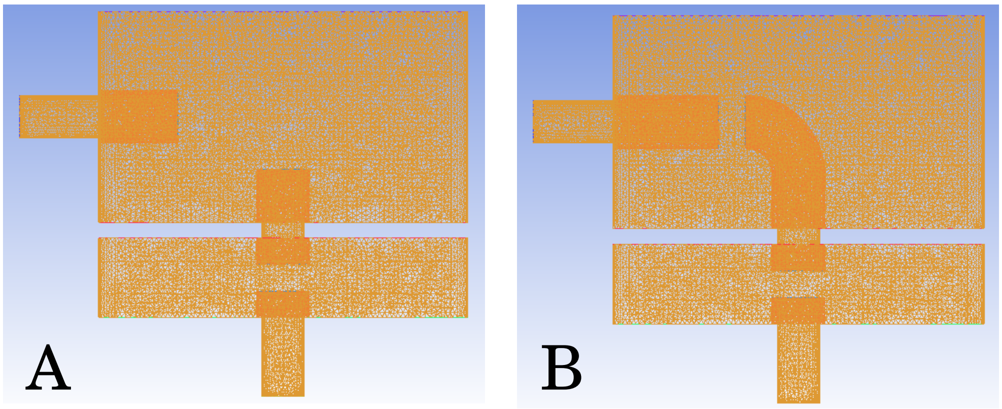


Figure A.10: Front view of the mesh for both configurations.

Dirichlet (no-slip) condition is imposed for the velocity at all the walls of the muffler. At the inlet, both velocity and pressure conditions are used in two different setups. The energy boundary condition is Dirichlet at the inlet and the walls (i.e. temperature is fixed), while at the outlet the energy condition is Neumann (i.e. null derivative). The flow condition at the outlet is given as pressure. This pressure is calculated in a separate simulation of the remaining compressor components (valves, compression chamber, resonator chambers and discharge tube). Hence, this is not a coupled simulation but a segregate one.

A.2.2 Numerical results

The numerical results obtained from the simulations of both configurations are presented below with the aim to show the influence of the geometry on the pressure drop.

First, in Figure A.11 it is illustrated the pressure at the outlet of the muffler together with the movement of the suction valve (number (4) in Figure A.8). These data are imposed as the flow condition at the outlet of the muffler. The chart shows well the coupling between the fluid flow and the cylinder dynamics. When the valve opens, there is an important pressure drop due to the suction of the gas towards the cylinder. Later, when it closes there is a high pressure peak followed by a peak of the same magnitude but opposite direction. This is due to the rebound effect of the valve reed. The second valve opening is quite attenuated this time. Although the valve lifts weakly again, the flow is definitively blocked so that the pressure starts to oscillate. These oscillations are slowly damped until the valve opens again in the next cycle.

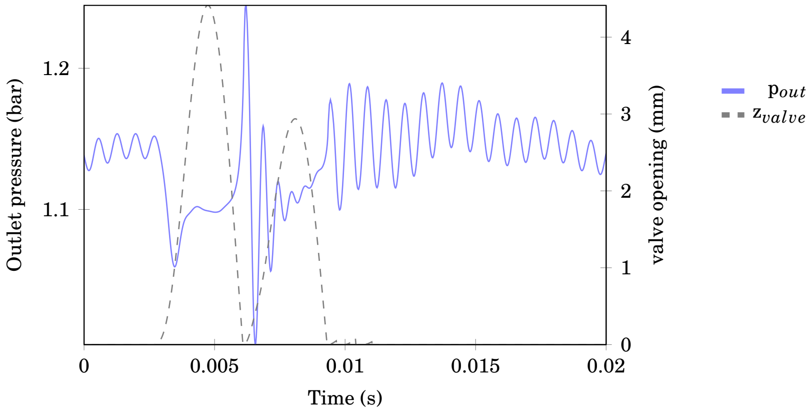


Figure A.11: Pressure condition at the muffler outlet. The suction valve motion is also depicted.

In order to give a better idea of the flow structure inside the muffler, two instantaneous velocity contours are shown in Figure A.12. In geometry (A) the entire mass flow rate is impinged through the first chamber and consequently goes mainly indirectly to the second chamber. On contrary, in geometry (B), the inlet mass flow rate comes directly through the bended tube to the muffler outlet and only a few percentage of fluid is flowing to the first chamber.

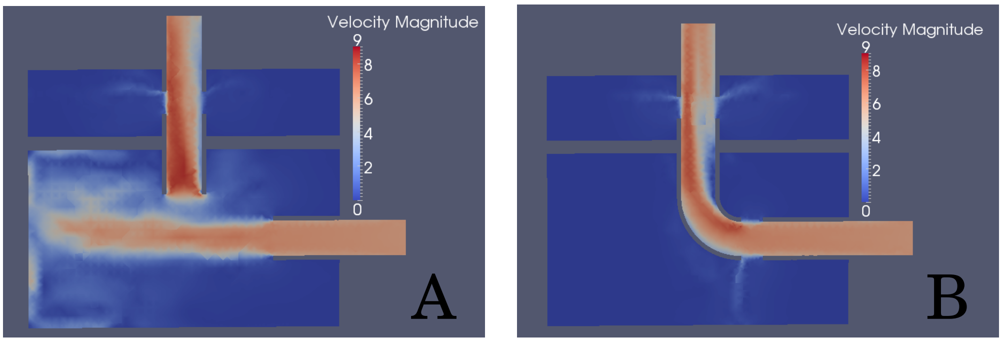


Figure A.12: Instantaneous velocity contours in both geometries.

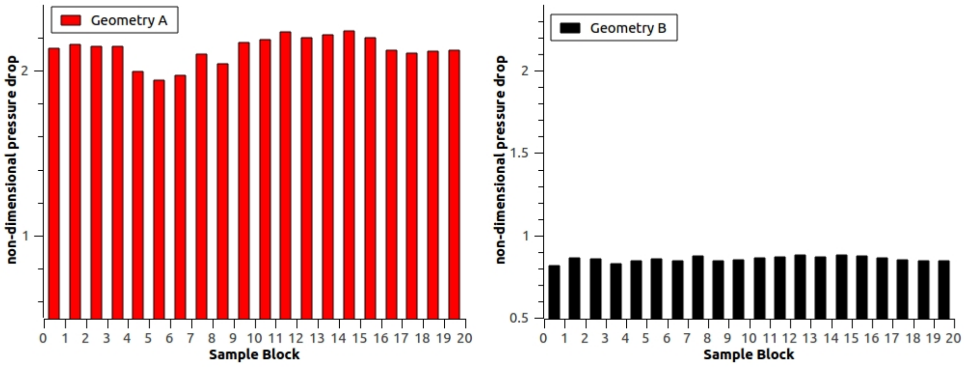


Figure A.13: Non-dimensional pressure drop during a whole compression cycle. These data are time-averaged by blocks.

The aim of this simulation is to compare the pressure drop in both geometries. Although configuration (B) is expected to be better than configuration (A), the pressure reduction is unknown and should be quantified to assess the trade-off between the fluid-dynamic benefits and the possible noise issues due to the raise of pressure perturbation. In Figure A.13, the non-dimensional pressure drop between the inlet and the outlet is shown as several time-averaged sample blocks over one compressor cycle.

Note that the inlet mass flow rate is imposed constant. Despite of not being representative of the compressor operational conditions, the case of constant inlet mass flow rate illustrates very well the significant difference in pressure drop that arises from the two geometries.

A.2.3 Use of compressible CFD solvers

Unfortunately, the Low-Mach solver is not appropriate for simulation of compressor mufflers. This solver is specific only for fluid phenomena with low density variation. Consequently, the Low-Mach approach is not able to capture mass accumulation in the muffler chambers. This can be appreciated in Figure A.14, the curve of the inlet mass flow almost matches the curve of the outlet mass flow. This is a characteristic of incompressible flows rather than compressible.

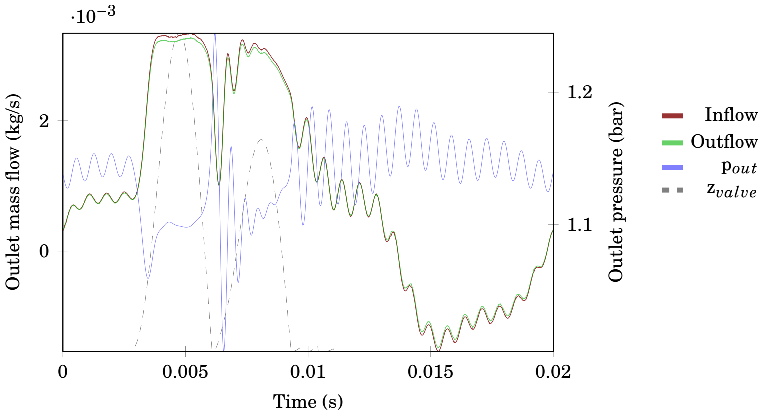


Figure A.14: Low-Mach based *solver* results: inlet and outlet mass flow rates obtained for the (A) geometry.

The recent advances on simulation of compressible flow were applied to obtain the numerical results presented below. The problem setup is similar to the one established in the previous numerical analysis. In this instance, the muffler geometry was slightly modified. Furthermore, velocity flow condition is fixed at the entrance of the muffler while pressure flow condition is fixed at the exit. The inlet condition is shown in Figure A.15. The pressure condition is exactly the same it was used in the previous section (see Figure A.11). In particular, a mixed no-slip flow condition is used at the outlet. By means of this approach, the boundary emulates the movement of the valve so that the flow is cut when the valve is closed. This is in concordance with the valve motion depicted in Figure A.11.

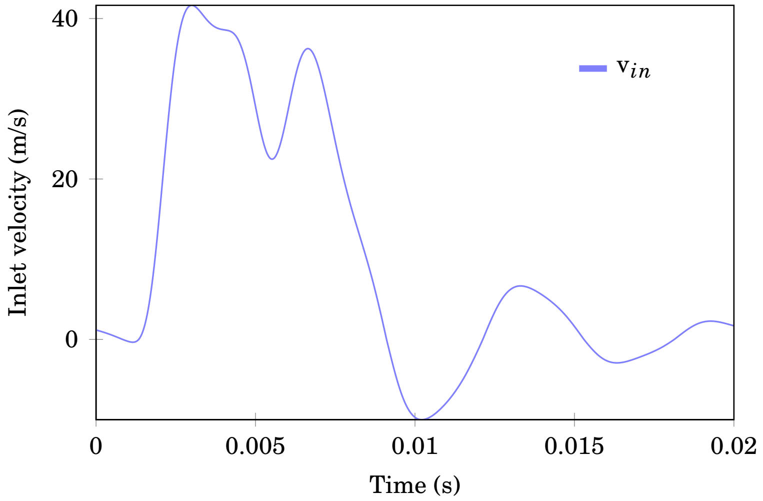


Figure A.15: Velocity of the refrigerant fluid at the inlet over the whole cycle.

The computational domain contains the muffler geometry as well as two buffering zones at the inlet and the outlet that ensure stability during the simulation. A tetrahedral unstructured mesh with more than 1.7 Million of elements is used, with refinements at the inlet/outlet buffers and at the areas where expansions are expected to happen.

In Figure A.16 is depicted the time-averaged pressure contour inside the muffler. The maximum pressure is located at the inlet (left image), which is entirely reasonable since the fluid goes from higher to lower pressures. The minimum pressure is located at the outlet (right image).

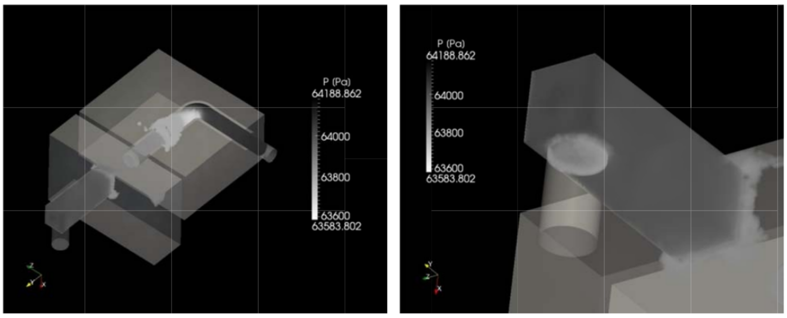


Figure A.16: Time-averaged pressure contours. The scale of these images has been adapted to emphasize the compressibility effects.

The time-averaged Mach number is shown in Figure A.17 (left). This image shows clearly the main path from the inlet to the outlet. The maximum Mach is achieved in the bended tube. As expected, in the first and second chambers there are two flow expansions shown in white color.

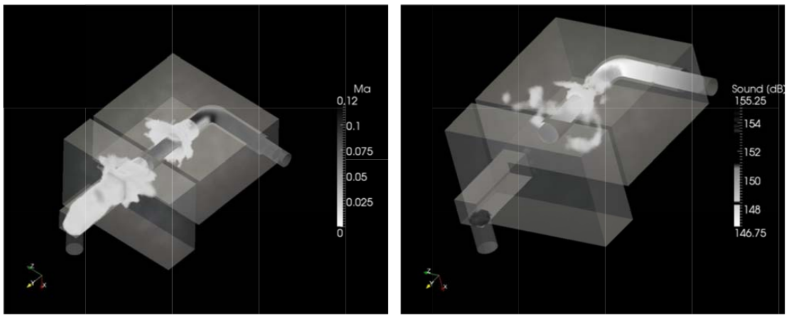


Figure A.17: Time-averaged Mach number (left) and noise level (right) contours. The scale of these images has been adapted to emphasize the compressibility effects.

Finally, the picture of the three-dimensional noise is exposed in Figure A.17 (right). In order to emphasize the maximum and the minimum noise levels, the ranges 148-149 dB and 151-154 dB have been omitted. Hence, the more silent parts are the inlet and some areas of the first chamber, while the part where the noise is higher is at the muffler exit.