Politecnico di Torino

Corso di Laurea Magistrale in Ingegneria Energetica e Nucleare - Poly^{2}Nuc

Tesi di Laurea Magistrale

Development of C++ tool for Reduced Order Modelling of the Molten Salt Fast Reactor multiphysics: Sensitivity Analysis application

Relatori:
Sandra Dulla
Antonio Cammi

Studente
Daniele Maria Panico
240002

Anno Accademico 2018-2019

(This page is intentionally left blank)

Acknowledgements

This thesis actually sets the end point in my career at Politecnico di Torino, many times the path seemed to be long, and now it ends like it started the day before yesterday. Thus I wish to thank some people who were with me along that path, with the awareness that I am not even able to write about engineering, so please be understanding if I am not able to do this as well.
First, I would like to thank my supervisor Prof. Sandra Dulla, for all the effort and patience she spent in her office trying to understand my confusing thoughts. Thank you for being so clear and practical in explaining me how to accomplish the task.
I am very thankful to my co-advisor Prof. Antonio Cammi, his advice always guided the path to get this work to an end, with the same enthusiasm he conveys during his lectures that were really educational for me.
Thank you to Prof. Stefano Lorenzi, for having helped me in the first stage of the work in understanding the "*mysterious*" OpenFOAM world.
A very huge "thank you" goes to Post-Doc Giovanni Stabile, developer of ITHACA. I used to stress him so much on the programming and all theory behind POD and reduced order modelling I can not understand why he still answers me so kindly and interested in my work. Thank you sincerely, your help was fundamental for me.
Thank you to Sokratia, Kelbij and all the other guys who are working on ITHACA, it was a pleasure for me to share ideas with you, our skype meeting were always fruitful.
I would like to thank my mother who always pushed me to study, her pride for what I do is always fuel for me. I would like to thank my father, He supported and believed in me even when my trust in this journey was running out, I know I have the best confident in you. I would like to thank my sister, thank you for being so "*dolono*", you are the best part of me and I love you.
Thank you Grazia, I am so lucky to have met you in my life. This period was so stressful but so full of happiness for having shared it with you. Some difficulties appear on the horizon, but I know that you are stronger than me. Your smile is always water to me. I love you.
Thank you Andrea, Mastroberti and Gerardo for simply being my best friends, I know it is difficult but it is difficult for me as well. Thank you to all the friends I met during the trip, my classmates, Gianluca and all the guys of *San Liborio*, you were my family and I share all my best experiences in Turin and Milan with you.
In conclusion, thanks to Politecnico and Poly2Nuc project, that gave me the possibility to grow up, to learn, to experience and to meet so much incredible and important people to me.

Daniele Maria Panico

(This page is intentionally left blank)

A zio Matteo,
grazie per tutta la serenità, la gioia e l'allegria
che hai sempre portato nella nostra famiglia.
Vorrei essere almeno un po' come te da grande.

(This page is intentionally left blank)

Contents

Introduction

Solving partial differential equation numerically is nowadays a standard technique to study real problem application, both in industrial or research activities, where complex physics and/or geometry are involved.

A lot of validated numerical discretization methods (*Finite Elements*, *Finite Volumes*, etc..) are available, along with dedicated softwares. Anyway, in most cases the problems analysed are actually very expensive by a computational point of view, that usually translates in very long time to get the solution. Moreover, in many applications, a large variety of system configurations have to be considered, this kind of problems are usually called *Many Query Analysis* [1], meaning that the system is asked to return some information for very large number of times (as in Sensitivity Analysis, SA).

One perfect example of difficult resolution for strandard discretization methods is represented by the multiphyscs of Molten Salt Fast Reactors [20]. Indeed, in this kind of reactor mutual and non-linear interaction happens between thermal-hydraulics and neutronics: since the fuel is liquid, the flow field influences the temperature and neutronic distributions; the temperature, with feedback mechanisms, influences the neutronic population and therefore the power generation that actually influences the temperature field as well; finally there is a feedback on velocity field given by natural convenction that establishes because of density gradients. The best way to deal with such problem is a multiphysics approach that, at some extent, aims to solve all the system of equations at the same time. This approach deletes approximation error that are commonly introduced when the different physics are considered separately. At the same time, this is also the most expensive approach as well.

One possible way to overcome these difficulties are Reduced Order Modelling techniques [1] [2]. These techniques are based on the assumption that the evolution in time of the dynamics of the system and its response into the parameters space (physical or geometrical) is governed by a reduced number of dominant modes [5]. In this work the Proper Orthogonal Decomposition with Galerkin projection is adopted [3] [4], within Finite Volume framework. In this method the governing equations are projected onto a low dimensional space called the reduced basis space that is optimally constructed starting from high fidelity simulations [5], i.e. the solutions given by the Full Order Model (FOM), also called *true solutions* or *snapshots*. In other

words, true solutions are adopted to train the Reduced Order Model (ROM) system with a set of values inside the parameters space, this set can be called *Training Set*. This stage is usually called *Offline stage*. Once the Offline stage is done, the ROM can be used to explore other configurations inside the parameters space, this stage is usually called *Online stage* [1].

The idea is to develop a complete C++ environment in order to fully carry on reduced order modelling, and use it to perform SA on the characteristics constants of Molten Salt fast Reactors. This is done adapting an existing C++ library, which name is ITHACA-FV [16]. The library is based on Finite Volume approach, in particular the solver adopted is OpenFOAM [14], a well known open-source software, both used in industrial and research application, which solutions will constitutes the *true solutions*. The FOM is based on [9]. but in this case the problem is simplified since natural convection is neglected and only laminar flow is considered. The former assumption is done because in most applications natural convection can be considered negligible, while the latter introduces aspects that, at this early stage of development, can be confusing in understanding the real possibilities of use reduced order modelling to describe the complex physics of Molten Salt Fast Reactors. A similar problem is developed in [8], but, at the best of my knowledge, this work represents the first attempt to adopt parametric variation inside the parameters domain for such complex system, varying more parameters at the same time. As regards sensitivity analysis, a new section is added to the library so that all the task is performed in unified fashion.

In conclusion, this work must be considered not as a complete reduced order model analysis, it is actually the development of the tool that is needed to carry that on. Some simple simulations are run to test the general behaviour of the software developed and start studying what are the possibilities and limitations of this approach. In other words, some very general outline are traced, but more specific and parametric tests must be performed in order to fully assess the quality of reduced order modelling for Molten Salt Fast Reactor, using the software here developed.

(This page is intentionally left blank)

Chapter 1

The GEN-IV Molten Salt Reactor

In this chapter the details of the Molten Salt Reactor are given. The first section contains a brief history and description of the concept adopted in this work. In the second one, the governing equations are reported with a brief description of them. The information about the concept, the composition of the fuel salt, and all the other physical quantities are taken from [20], on which the OpenFOAM model developed in [9] is based.

1.1 Description of Molten Salt Reactor Concept

Molten Salt Reactor (MSR) is one of the six Generation-IV designs evaluated as the more promising future reactors by Gen-IV International Forum (GIF) [13], its main characteristic is that the fuel is dissolved in molten fluoride salt. Eight main goals are established by the forum for these new concepts [13]:

- Sustainability-1:
  Generation IV nuclear energy systems will provide sustainable energy generation that meets clean air objectives and provides long-term availability of systems and effective fuel utilisation for worldwide energy production.

- Sustainability-2:
  Generation IV nuclear energy systems will minimise and manage their nuclear waste and notably reduce the long-term stewardship burden, thereby improving protection for the public health and the environment.

- Economics-1:
  Generation IV nuclear energy systems will have a clear life-cycle cost advantage over other energy sources.

- Economics-2:
  Generation IV nuclear energy systems will have a level of financial risk comparable to other energy projects.

- Safety and Reliability-1:
  Generation IV nuclear energy systems operations will excel in safety and reliability.

- Safety and Reliability-2:
  Generation IV nuclear energy systems will have a very low likelihood and degree of reactor core damage.

- Safety and Reliability-3:
  Generation IV nuclear energy systems will eliminate the need for offsite emergency response.

- Proliferation Resistance and Physical Protection:
  Generation IV nuclear energy systems will increase the assurance that they are very unattractive and the least desirable route for diversion or theft of weapons-usable materials, and provide increased physical protection against acts of terrorism.

Starting from the Oak-Ridge National Laboratory Molten Salt Breeder Reactor project, which was a thermal-neutron-spectrum graphite-moderated reactor, an innovative concept called Molten Salt Fast Reactor (MSFR) has been proposed. The primary feature of the MSFR concept versus that of other older MSR designs is the removal of the graphite moderator from the core (graphite-free core), resulting in a

breeder reactor with a fast neutron spectrum and operated in the Thorium fuel cycle.

In the MSFR concept, the nuclear fission reactions take place within the flowing fuel salt in the cavity where a critical mass is attained. The salt's thermal-hydraulic behavior is closely coupled to its neutronic behavior, because the salt's circulating time (4s) and the lifetime of the precursors of delayed neutrons (around 10s) are of the same order of.

The choice of the fuel salt composition relies on several parametric reactor studies (chemical and neutronic considerations, burning capabilities, safety coefficients, and deployment capabilities). The nominal design fuel salt composition is $LiF - ThF_4 - {}^{233}UF_4$, with percentage molar fraction $77.5 - 20 - 2.5\ mol\%$. This salt composition leads to a fast neutron spectrum in the core. With a fusion temperature of 838 K.

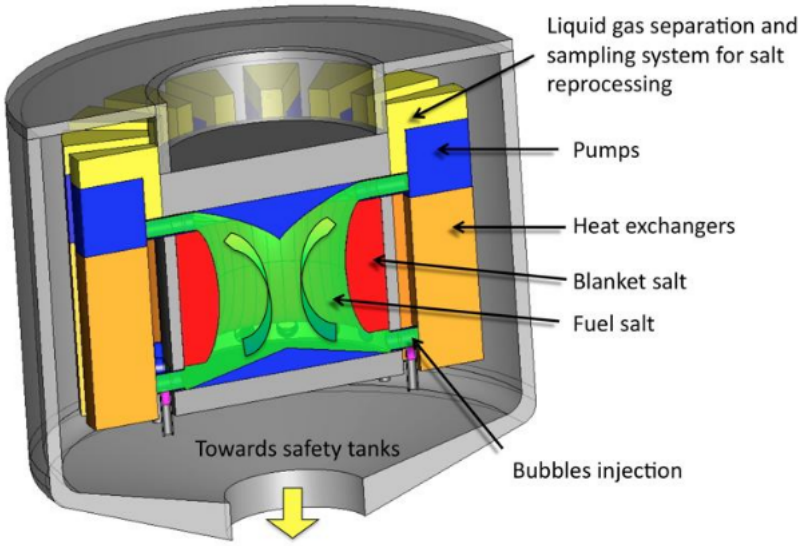


Figure 1.1: Molten Salt Fast Reactor concept, from [20].

The core active region is defined as the salt volume where most of nuclear fissions take place. It includes the flowing salt in the central cavity, the injection zone (in the bottom part of the core) and the extraction zone (top of the core). The reference concept is designed for a nominal power of 3 GWth, with a salt temperature rise preliminary fixed at $\Delta T = 100\ K$. The operating temperatures chosen in the initial simulations were 923 K (inlet temperature) and 1023 K (outlet temperature). The fertile blanket serves as radial reflector and as a neutron shield to protect the external components of the fuel loops (pipes, heat exchangers). In addition to this protection function, the fertile blanket is used to improve the breeding capabilities of the reactor.

1.2 Molten Salt Fast Reactor physics - Full Order Model

Because of its fuel peculiarity, thermal-hydraulics and neutronics are strongly coupled in the MSFR, therefore a multiphysics approach seems to be the most appropriate to obtain the fields distribution inside the reactor. Of course this is the most demanding one as well. From these consideration actually comes the idea to develop a reduced order model indeed.

In this section the governing equations of the MSFR are described. The model is almost the same developed in [9] for which an OpenFOAM solver has been already developed and tested. In this work the model is simplified since buoyancy forces are neglected and only laminar flow is considered.
The next system thus define the Full Order Model (FOM) and its solution will be the true solution:

$$\begin{cases} \dfrac{\partial \boldsymbol{u}}{\partial t} = -(\boldsymbol{u}\cdot\nabla)\boldsymbol{u} + \nu\nabla^2\boldsymbol{u} - \nabla p & \text{(1.1a)} \\ \nabla^2 p = -\nabla\cdot(\boldsymbol{u}\cdot\nabla)\boldsymbol{u} & \text{(1.1b)} \\ \dfrac{1}{v}\dfrac{\partial \phi}{\partial t} = \nabla\cdot(D\nabla\phi) + \big[\dfrac{1}{k_{eff}}(1-\beta_{tot})(\nu\Sigma)_f - \Sigma_a\big]\phi + \displaystyle\sum_{i=1}^{8}\lambda_i C_i & \text{(1.1c)} \\ \dfrac{\partial C_i}{\partial t} = -(\boldsymbol{u}\cdot\nabla)C_i + \dfrac{\nu}{Sc}\nabla^2 C_i + \dfrac{\beta_i}{k_{eff}}(\nu\Sigma)_f\phi - \lambda_i C_i \quad \forall i = 1,\dots,8 & \text{(1.1d)} \\ \dfrac{\partial T}{\partial t} = -(\boldsymbol{u}\cdot\nabla)T + \dfrac{\nu}{Pr}\nabla^2 T + (1-\beta_{h,tot})\dfrac{\Sigma_P}{k_{eff}\rho c_p}\phi + \displaystyle\sum_{j=1}^{3}\dfrac{\lambda_{h,j}}{\rho c_p}H_j & \text{(1.1e)} \\ \dfrac{\partial H_j}{\partial t} = -(\boldsymbol{u}\cdot\nabla)H_j + \dfrac{\nu}{Sc}\nabla^2 H_j + \dfrac{\beta_{h,j}\Sigma_P}{k_{eff}}\phi - \lambda_{h,j}H_j \quad \forall j = 1,\dots,3 & \text{(1.1f)} \end{cases}$$

It is a non-linear system of 15 partial differential equations. The first two equations are the momentum equation and the so called *Poisson Pressure Equation* (PPE). It is obtained taking the divergence of the momentum equation and exploiting the conitnuity equation, that for this case of incompressible flow is simply: $\nabla\cdot\boldsymbol{u} = 0$. In this way velocity and pressure are explicitly coupled, OpenFOAM deals with the pressure field in the same way as well.

The equation for the neutron flux ϕ just follows: in this case monokinetic diffusion equation, with 8 groups of precursors C_i, is used to describe the evolution in time of neutrons' population. The peculiarity of the liquid fuel is modelled in the precursors equation: they change in position not only because of diffusion, but also because they actually move, as described by the second and the first term in the equation respectively. Notice that precursors are ordered with increasing decay constant (or decreasing half-time).

The last four equations of system (1.2) deal with the conservation of energy. Temperature locally changes because of (as listed in the equation): streaming of the

fuel, diffusion caused by temperature gradients, energy produced by fission (Σ_P is the energy released per fission event), energy released later by radioactive decay of nuclei in the fuel. As regards the decay heat, it is modelled with three groups approximation, ordered with increasing time scale ($1/\lambda_{h,j}$). It is evident the similarity with the precursors and flux equations: the fraction $(1 - \beta_{h,tot})$ of energy produced by fission, "instantaneously" raises the temperature locally, the remaining part is "stored", according to $\beta_{h,j}$, by the decay heat groups that later release it causing a delayed raise in temperature.
As developed in [9], some physical constants are temperature dependent, and therefore they are changing fields in time as well:

$$\begin{cases} \rho(\boldsymbol{x},t) = \rho_0\big[1 - \beta_{TE}(T - T_0)\big] & \text{(1.2a)} \\ (\nu\Sigma)_f(\boldsymbol{x},t) = \left[(\nu\Sigma)_{f,0} + \alpha_{(\nu\Sigma)_f}\log\left(\dfrac{T}{T_{0,XS}}\right)\right]\dfrac{\rho}{\rho_0} & \text{(1.2b)} \\ D(\boldsymbol{x},t) = \left[D_0 + \alpha_D\log\left(\dfrac{T}{T_{0,XS}}\right)\right]\dfrac{\rho_0}{\rho} & \text{(1.2c)} \\ \Sigma_a(\boldsymbol{x},t) = \left[\Sigma_{a,0} + \alpha_{\Sigma_a}\log\left(\dfrac{T}{T_{0,XS}}\right)\right]\dfrac{\rho}{\rho_0} & \text{(1.2d)} \\ \Sigma_P(\boldsymbol{x},t) = \left[\Sigma_{P,0} + \alpha_{\Sigma_P}\log\left(\dfrac{T}{T_{0,XS}}\right)\right]\dfrac{\rho}{\rho_0} & \text{(1.2e)} \end{cases}$$

This dependence is introduced in order to account for thermal feedback inside the reactor.

At this point, the complex interactions among the fields can be summarized as follows: once the flow field is determined, it influences the precursors distribution and thus the flux one, which is, at the same time, a source for precursors; temperature and decay heat are determined by both flux and velocity fields, temperature field finally gives a feedback to neutronics through change in density and cross sections, thus establishing a mutual interaction. In the case in which also buoyancy forces are considered, the change in temperature changes the velocity field as well, then causing a complete interaction among the three physics of the reactor, i.e.: fluid-dynamics, neutronics and energy.

The value of the multiplication factor k_{eff} is updated as the code runs in the following way:

$$k_{eff}(t_n) = \frac{\displaystyle\int_\Omega \phi(\boldsymbol{x},t_n)d^3x}{\displaystyle\int_\Omega \phi(\boldsymbol{x},t_{n-1})d^3x} \tag{1.3}$$

Where t_n is the n-th time instant of the simulation.

Finally, the power density inside the reactor is defined as:

$$W = (1 - \beta_{h,tot})\frac{\Sigma_P}{k_{eff}}\phi + \sum_{j=1}^{3} \lambda_{h,j} H_j \tag{1.4}$$

and thus the total power P is simply:

$$P(t) = \int_{\Omega} W(\boldsymbol{x}, t) d^3x \tag{1.5}$$

In the remaining part of this section all constants values of systems (1.1) and (1.2) used in this work are listed, they are taken from [20]:

Constant	Symbol	Value	Units
Kinematic viscosity	ν	$2.46 \cdot 10^{-6}$	$[m^2/s]$
Thermal expansion coefficient	β_{TE}	$2.14 \cdot 10^{-4}$	$[1/K]$
Density	ρ_0	4125	$[kg/m^3]$
Specific heat	c_p	1594	$[kg/m^2 s]$
Prandtl number	Pr	16	$[-]$
Schmidt number	Sc	20	$[-]$
Reference Temperature	T_0	973	$[K]$

Table 1.1: Transport Properties.

Constant	Symbol	Value	Units
Inverse velocity	$1/v$	$6.55767 \cdot 10^{-7}$	$[s/m]$
Diffusion coefficient	D_0	$1.17204 \cdot 10^{-2}$	$[m]$
Diffusion expansion coefficient	α_D	$-5.9788 \cdot 10^{-5}$	$[m]$
Absorption cross section	Σ_a	$6.89269 \cdot 10^{-1}$	$[1/m]$
Absorption XS expansion coefficient	α_{Σ_a}	$7.8420 \cdot^{-3}$	$[1/m]$
Fission cross section	$(\nu\Sigma)_{f,0}$	$7.53492 \cdot 10^{-1}$	$[1/m]$
Fission XS expansion coefficient	$\alpha_{(\nu\Sigma)_f}$	$-1.7001 \cdot 10^{-1}$	$[1/m]$
Power cross section	$\Sigma_{P,0}$	$9.5731 \cdot 10^{-12}$	$[kg\ m/s^2]$
Power XS expansion coefficient	α_P	$-2.0595 \cdot 10^{-13}$	$[kg\ m/s^2]$
XS Reference Temperature	$T_{0,XS}$	900	$[K]$

Table 1.2: Nuclear Properties.

Group	Decay constant λ_i $[1/s]$	β_i $[-]$
1	$1.24667 \cdot 10^{-2}$	$21.8 \cdot 10^{-5}$
2	$2.82917 \cdot 10^{-2}$	$47.6 \cdot 10^{-5}$
3	$4.25244 \cdot 10^{-2}$	$39.3 \cdot 10^{-5}$
4	$1.33042 \cdot 10^{-1}$	$63.5 \cdot 10^{-5}$
5	$2.92467 \cdot 10^{-1}$	$103.5 \cdot 10^{-5}$
6	$6.66488 \cdot 10^{-1}$	$18.1 \cdot 10^{-5}$
7	1.63478	$22.8 \cdot 10^{-5}$
8	3.55460	$5.2 \cdot 10^{-5}$

Table 1.3: Precursors Properties.

Group	Decay constant $\lambda_{h,i}$ $[1/s]$	$\beta_{h,i}$ $[-]$
1	0.1973	$1.17 \cdot 10^{-2}$
2	$1.68 \cdot 10^{-2}$	$1.29 \cdot 10^{-2}$
3	$3.58 \cdot 10^{-4}$	$1.86 \cdot 10^{-2}$

Table 1.4: Decay Heat Groups Properties.

(This page is intentionally left blank)

Chapter 2

Reduced Order Modelling and Sensitivity Analysis

This chapter deals with the procedure adopted to obtain the Reduced Order Model (ROM) and how the sensitivity analysis is carried on. In the first subsection the concepts of offline stage, Proper Orthogonal Decomposition (POD) modes and lift-function are explained. Instead, in the second one the way to actually obtain the ROM is presented. Finally, Sensitivity Analysis (SA) is dealt in the last one.
For sake of synthesis, the description of POD-Galerkin projection with Finite Volume (FV) approach, together with the FV integrals approximation, are omitted, they are widely explained in [10]. POD modes calculation, projection procedure, results and notation is instead based on [5], [6] and [12]. The interpolation technique with Radial Basis Functions is taken from [7]. Finally, SA is based on the book by Saltelli [24].

2.1 Proper Orthogonal Decomposition

All fields in system (1.1) are approximated using the Proper Orthogonal Decomposition that consists into the decomposition of the fields into temporal coefficients and orthonormal spatial bases:

$$\begin{cases} \boldsymbol{u}(\boldsymbol{x},t) \approx \boldsymbol{u}_r(\boldsymbol{x},t) = \sum_{i=1}^{N_{\boldsymbol{u}}} a_i(t)\boldsymbol{\chi}_{u,i}(\boldsymbol{x}) & \text{(2.1a)} \\ p(\boldsymbol{x},t) \approx p_r(\boldsymbol{x},t) = \sum_{i=1}^{N_p} b_i(t)\chi_{p,i}(\boldsymbol{x}) & \text{(2.1b)} \\ \phi(\boldsymbol{x},t) \approx \phi_r(\boldsymbol{x},t) = \sum_{i=1}^{N_\phi} c_i(t)\chi_{\phi,i}(\boldsymbol{x}) & \text{(2.1c)} \\ C_j(\boldsymbol{x},t) \approx C_{j,r}(\boldsymbol{x},t) = \sum_{i=1}^{N_{C_j}} d_{j,i}(t)\chi_{C_j,i}(\boldsymbol{x}) \qquad \forall j = 1\dots 8 & \text{(2.1d)} \\ T(\boldsymbol{x},t) \approx T_r(\boldsymbol{x},t) = \sum_{i=1}^{N_T} e_i(t)\chi_{T,i}(\boldsymbol{x}) & \text{(2.1e)} \\ H_j(\boldsymbol{x},t) \approx H_{j,r}(\boldsymbol{x},t) = \sum_{i=1}^{N_{H_j}} f_{j,i}(t)\chi_{H_j,i}(\boldsymbol{x}) \qquad \forall j = 1\dots 3 & \text{(2.1f)} \end{cases}$$

The same is done with the coefficients of the equations that depend on the temperature (system (1.2)), for a handy implementation some are defined on purpose as follows:

$$\begin{cases} V(\boldsymbol{x},t) \equiv \dfrac{1}{\rho} \approx V_r(\boldsymbol{x},t) = \sum_{i=1}^{N_{const}} \alpha_i(t)\chi_{V,i}(\boldsymbol{x}) & \text{(2.2a)} \\ \Gamma(\boldsymbol{x},t) \equiv \dfrac{(\nu\Sigma)_f}{k_{eff}} \approx \Gamma_r(\boldsymbol{x},t) = \sum_{i=1}^{N_{const}} \gamma_i(t)\chi_{\Gamma,i}(\boldsymbol{x} & \text{(2.2b)} \\ E(\boldsymbol{x},t) \equiv \dfrac{\Sigma_P}{k_{eff}} \approx E_r(\boldsymbol{x},t) = \sum_{i=1}^{N_{const}} \epsilon_i(t)\chi_{E,i}(\boldsymbol{x}) & \text{(2.2c)} \\ \Sigma_a(\boldsymbol{x},t) \approx \Sigma_{a,r}(\boldsymbol{x},t) = \sum_{i=1}^{N_{const}} \zeta_i(t)\chi_{\Sigma_a,i}(\boldsymbol{x}) & \text{(2.2d)} \\ D(\boldsymbol{x},t) \approx D_r(\boldsymbol{x},t) = \sum_{i=1}^{N_{const}} \eta_i(t)\chi_{D,i}(\boldsymbol{x}) & \text{(2.2e)} \\ \Theta(\boldsymbol{x},t) \equiv \dfrac{\Sigma_P}{\rho k_{eff}} \approx \Theta_r(\boldsymbol{x},t) = \sum_{i=1}^{N_{const}} \theta_i(t)\chi_{\Theta,i}(\boldsymbol{x}) & \text{(2.2f)} \end{cases}$$

Even if, as reported in system (1.2), their dependence on the temperature is different, their all expanded up to N_{const} in order to reduce the degree of freedom of the problem (that are already large, as will be discussed in the last chapter) and to facilitate their decomposition and reconstruction implementation (see next chapter). The power density is consequently reconstructed as:

$$W_r(\boldsymbol{x}, t) = (1 - \beta_{h,tot}) E_r \phi_r + \sum_{j=1}^{3} \lambda_{h,j} H_{j,r} \tag{2.3}$$

Before explaining how the spatial bases are computed, some concepts must be clarified: as introduced in Chapter , Reduced Order Modelling consists of offline and online stage.
In the offline stage the FOM is solved a certain N_{off} number of times varying the values of some n characteristic parameters of the system. These parameters can be collected in a vector $\boldsymbol{\mu}$ so that

$$\boldsymbol{\mu} = \{\mu_1, \dots, \mu_n\} \tag{2.4}$$

Therefore, it possible to define the parameters matrix, or *Training Set*, $\widehat{M}$ as:

$$\widehat{M} = \begin{pmatrix} \boldsymbol{\mu}_1 \\ \vdots \\ \boldsymbol{\mu}_i \\ \vdots \\ \boldsymbol{\mu}_{N_{off}} \end{pmatrix} = \begin{pmatrix} \mu_{1,1} & \dots & \mu_{1,n} \\ \vdots & & \vdots \\ \mu_{i,1} & \dots & \mu_{i,n} \\ \vdots & & \vdots \\ \mu_{N_{off},1} & \dots & \mu_{N_{off},n} \end{pmatrix} \tag{2.5}$$

Each row of $\widehat{M}$ is passed to the FOM and the solution is saved at some specified time instants of the simulation t_k, $\forall k = 1, \dots, K$. These saved solutions are called snapshots, it follows that the total number of snapshots is $N_S = N_{off} \times K$.
For each field $F(\boldsymbol{x}, t) \approx \sum_{i=1}^{N_F} \omega_i(t) \chi_{F,i}(\boldsymbol{x})$ of systems (2.1) and (2.2) (for the velocity field the spatial modes are vector fields as well, as reported in system (2.1)) the snapshots are collected in the so called *snapshots matrix*:

$$\widehat{F} = (F_1, \dots, F_{N_S}) \tag{2.6}$$

$\widehat{F} \in \mathbb{R}^{N_h \times N_S}$ where N_h is the number of points in the domain where the solution is computed.

The POD bases are the orthonormal spatial bases that minimize the average error between the snapshots and their orthogonal projection onto the bases, in other words the POD space $\mathbb{V}_{POD} = \text{span}\{\chi_1, \chi_2, \dots, \chi_{N_S}\}$ is constructed solving the minimization problem:

$$\mathbb{V}_{POD} = \arg\min \frac{1}{N_S} \sum_{n=1}^{N_S} \left\| F_n - \sum_{l=0}^{N_r} (F_l, \chi_i)_{L^2(\Omega)} \chi_i \right\|^2_{L^2(\Omega)} \quad \forall N_r = 1, \dots, N_S \tag{2.7}$$

Where $\| \|_{L^2(\Omega)}$ and $(\ , \ )_{L^2(\Omega)}$ are the standard L^2-norm and L^2-inner product over the physical domain Ω respectively. To solve this problem, two different approaches can be used:

- EigenValue Decomposition (EVD)
- Singular Value Decomposition (SVD)

Only the former is explained here, the latter can be found in [5], anyway both of them are implemented in ITHACA-FV. The snapshots matrix is used to find the spatial bases in the following way:

- The correlation matrix $\widehat{C}^F \in \mathbb{R}^{N_S \times N_S}$ is computed, each of its ij-th element is given by:
$$C^F_{ij} = (F_i, F_j)_{L^2(\Omega)} \tag{2.8}$$
- An eigenvalue problem is solved:
$$\widehat{C}^F \widehat{Q}^F = \widehat{Q}^F \widehat{\lambda}^F \tag{2.9}$$
$\widehat{Q}^F \in \mathbb{R}^{N_S \times N_S}$ is a square matrix whose columns are the eigenvectors and $\widehat{\lambda}^F \in \mathbb{R}^{N_S \times N_S}$ is a diagonal matrix containing the eigenvalues λ^F_{ii}.
- The POD modes can be finally obtained as:
$$\chi_i = \frac{1}{\sqrt{\lambda^F_{ii}}} \widehat{F} \boldsymbol{Q}^F_{\boldsymbol{i}} \tag{2.10}$$
where $\boldsymbol{Q}^F_{\boldsymbol{i}}$ is the i-th column of $\widehat{Q}^F$.
- a number $\widetilde{N_F} < N_S$ of POD modes is saved, but actually an even lower number $N_F < \widetilde{N_F}$ is used at projection stage (see section 2.2). Therefore the number of modes in the expansions in systems (2.1) and (2.2) is the number of modes used in the projection stage and thus the one adopted to reconstruct the reduced solution. This is possible because POD modes are ordered so that the lower is their index, the higher is the content of energy (or information) about the system [22]. This information is given by the value of the correspondent eigenvalue or by the comulative of the eigenvalues up to that index, given that:
$$\sum_{i=1}^{N_S} \lambda^F_{ii} = 1 \tag{2.11}$$

2.2 Galerkin Projection - Reduced Order Model

First of all the boundary condition for the pressure field is introduced since it is going to be used in this section:

$$\frac{\partial p}{\partial \boldsymbol{n}} = -\nu \boldsymbol{n} \cdot (\nabla \times \nabla \times \boldsymbol{u}) \quad \forall \boldsymbol{x} \in \partial\Omega \tag{2.12}$$

That is a Neumann boundary condition (as introduced in [6]) for entire boundary $\partial\Omega$ of the domain Ω. Each equation of system (1.1) is multiplied times a generic i-th correspondent mode of the field and the $L2$-inner product is performed, this procedure is known as *Galerkin Projection*:

$$\begin{cases}
0 = (\boldsymbol{\chi}_{u,i}, -\dot{\boldsymbol{u}} - (\boldsymbol{u}\cdot\nabla)\boldsymbol{u} + \nu\nabla^2\boldsymbol{u} - \nabla p)_{L^2(\Omega)} & \text{(2.13a)}\\
0 = (\nabla\chi_{p,i}, \nabla p)_{L^2(\Omega)} + (\chi_{p,i}, \nabla\cdot(\boldsymbol{u}\otimes\boldsymbol{u}))_{L^2(\Omega)} + \\
\quad - \nu(\boldsymbol{n}\times\nabla\chi_{p,i}, \nabla\times\boldsymbol{u})_{\partial\Omega} & \text{(2.13b)}\\
0 = (\chi_{\phi,i}, -\frac{1}{v}\dot{\phi} + \nabla\cdot(D\nabla\phi) + (1-\beta_{tot})\Gamma\phi - \Sigma_a\phi + \sum_{j=1}^{8}\lambda_j C_j)_{L^2(\Omega)} & \text{(2.13c)}\\
0 = (\chi_{C_j,i}, -\dot{C_j} - (\boldsymbol{u}\cdot\nabla)C_i + \frac{\nu}{Sc}\nabla^2 C_j + \beta_j\Gamma\phi - \lambda_j C_j)_{L^2(\Omega)} & \text{(2.13d)}\\
0 = (\chi_{T,i}, -\dot{T} - (\boldsymbol{u}\cdot\nabla)T + \frac{\nu}{Pr}\nabla^2 T + \frac{1-\beta_{h,tot}}{c_p}\Theta\phi + \sum_{j=1}^{3} V\frac{\lambda_{h,j}}{c_p}H_j)_{L^2(\Omega} & \text{(2.13e)}\\
0 = (\chi_{H_j,i}, -\dot{H_j} - (\boldsymbol{u}\cdot\nabla)H_j + \frac{\nu}{Sc}\nabla^2 H_j + \beta_{h,j}E\phi - \lambda_{h,j}H_j)_{L^2(\Omega)} & \text{(2.13f)}
\end{cases}$$

Where the equation (2.13b) has been obtained with integration by part of the laplacian term and exploiting the pressure boundary equation.
The result of the projection is finally:

$$\begin{cases}
\boldsymbol{0} = -\widehat{M}\dot{\boldsymbol{a}} - \boldsymbol{a}^\intercal\widehat{\boldsymbol{C}}\boldsymbol{a} + \nu\widehat{B}\boldsymbol{a} - \widehat{K}\boldsymbol{b} & \text{(2.14a)}\\
\boldsymbol{0} = \widehat{D}\boldsymbol{b} + \boldsymbol{a}^\intercal\widehat{\boldsymbol{G}}\boldsymbol{a} - \nu\widehat{N}\boldsymbol{a} & \text{(2.14b)}\\
\boldsymbol{0} = -\frac{1}{v}\widehat{M}_\phi\dot{\boldsymbol{c}} + \boldsymbol{\eta}^\intercal\widehat{\boldsymbol{L}}_\phi\boldsymbol{c} + (1-\beta_{tot})\boldsymbol{\gamma}^\intercal\widehat{\boldsymbol{P}}_\phi\boldsymbol{c} - \boldsymbol{\zeta}^\intercal\widehat{\boldsymbol{A}}_\phi\boldsymbol{c} + \sum_{i=1}^{8}\lambda_i\widehat{S}_{\phi,C_i}\boldsymbol{d}_i & \text{(2.14c)}\\
\boldsymbol{0} = -\widehat{M}_{C_i}\dot{\boldsymbol{d}}_i - \boldsymbol{a}^\intercal\widehat{\boldsymbol{C}}_{C_i}\boldsymbol{d}_i + \frac{\nu}{Sc}\widehat{L}_{C_i}\boldsymbol{d_i} + \beta_i\boldsymbol{\gamma}^\intercal\widehat{\boldsymbol{S}}_{C_i,\phi}\boldsymbol{c} - \lambda_i\widehat{M}_{C_i}\boldsymbol{d}_i & \text{(2.14d)}\\
\boldsymbol{0} = -\widehat{M}_T\dot{\boldsymbol{e}} - \boldsymbol{a}^\intercal\widehat{\boldsymbol{C}}_T\boldsymbol{e} + \frac{\nu}{Pr}\widehat{L}_T\boldsymbol{e} + \frac{1-\beta_{h,tot}}{c_p}\boldsymbol{\theta}^\intercal\widehat{\boldsymbol{S}}_{T,\phi}\boldsymbol{e} + \sum_{j=1}^{3}\frac{\lambda_{h,j}}{c_p}\boldsymbol{\alpha}^\intercal\widehat{\boldsymbol{S}}_{T,H_J}\boldsymbol{e} & \text{(2.14e)}\\
\boldsymbol{0} = -\widehat{M}_{H_j}\dot{\boldsymbol{f}}_j - \boldsymbol{a}^\intercal\widehat{\boldsymbol{C}}_{H_j}\boldsymbol{f}_j + \frac{\nu}{Sc}\widehat{L}_{H_j}\boldsymbol{f}_j + \beta_{h,j}\boldsymbol{\epsilon}^\intercal\widehat{\boldsymbol{S}}_{H_j,\phi}\boldsymbol{f}_j - \lambda_{H_j}\widehat{M}_{H_j}\boldsymbol{f}_j & \text{(2.14f)}
\end{cases}$$

Where the " $\dot{}$ " indicates the time derivative of the term (partial or total, respectively for the fields or their coefficients), for instance:

$$\dot{\boldsymbol{c}} = \left\{\frac{dc_1}{dt}, \frac{dc_2}{dt}, \ldots, \frac{dc_{N_\phi}}{dt}\right\}$$

It must be highlighted that system (2.14) actually constitutes the Reduced Order Model (ROM) and its solution allow to reconstruct the fields of the MSFR. The various term inside system (2.14) read as:

$$M_{ij} = (\boldsymbol{\chi}_{u,i}, \boldsymbol{\chi}_{u,j})_{L^2(\Omega)} \tag{2.15}$$
$$C_{ijk} = (\boldsymbol{\chi}_{u,i}, \nabla \cdot (\boldsymbol{\chi}_{u,j} \otimes \boldsymbol{\chi}_{u,k}))_{L^2(\Omega)} \tag{2.16}$$
$$B_{ij} = (\boldsymbol{\chi}_{u,i}, \nabla^2 \boldsymbol{\chi}_{u,j})_{L^2(\Omega)} \tag{2.17}$$
$$K_{ij} = (\boldsymbol{\chi}_{u,i}, \nabla \chi_{p,j})_{L^2(\Omega)} \tag{2.18}$$
$$D_{ij} = (\nabla \chi_{p,i}, \nabla \chi_{p,j})_{L^2(\Omega)} \tag{2.19}$$
$$G_{ijk} = (\nabla \chi_{p,i}, \nabla \cdot (\boldsymbol{\chi}_{u,j} \otimes \boldsymbol{\chi}_{u,k}))_{L^2(\Omega)} \tag{2.20}$$
$$N_{ij} = (\boldsymbol{n} \times \nabla \chi_{p,i}, \nabla \times \boldsymbol{\chi}_{u,j})_{\partial\Omega} \tag{2.21}$$

For every scalar X in the system the convective term reads as:

$$C_{X,ijk} = (\chi_{X,i}, \nabla \cdot (\boldsymbol{\chi}_{u,j} \chi_{X,k}))_{L^2(\Omega)} \tag{2.22}$$

The various $\widehat{M_X}$ and $\widehat{L}_X$ terms are the mass and laplacian terms respectively and they read as:

$$M_{X,ij} = (\chi_{X,i}, \chi_{X,j})_{L^2(\Omega)} \tag{2.23}$$

$$L_{X,ij} = (\chi_{X,i}, \nabla^2 \chi_{X,j})_{L^2(\Omega)} \tag{2.24}$$

The only different term is the laplacian of the flux $\widehat{\boldsymbol{L}}_\phi$ since D is a field as well, also the definition of $\widehat{\boldsymbol{P}}_{\boldsymbol{\phi}}$ and $\widehat{\boldsymbol{A}}_{\boldsymbol{\phi}}$ is given:

$$L_{\phi,ijk} = (\chi_{\phi,i}, \nabla \cdot (\chi_{D,j} \nabla \chi_{\phi,k}))_{L^2(\Omega)} \tag{2.25}$$
$$P_{\phi,ijk} = (\chi_{\phi,i}, \chi_{\Gamma,j} \chi_{\phi,k})_{L^2(\Omega)} \tag{2.26}$$
$$A_{\phi,ijk} = (\chi_{\phi,i}, \chi_{\Sigma_a,j} \chi_{\phi,k})_{L^2(\Omega)} \tag{2.27}$$

The various $\widehat{S}_{X,Y}$ are sources of X field by Y one, in case the source is multiplied times one of the field Z in system (1.2), it becomes a tensor:

$$S_{\phi,C_i,ij} = (\chi_{\phi,i}, \chi_j)_{L^2(\Omega)} \tag{2.28}$$
$$S_{X,Y,ijk} = (\chi_{X,i}, \chi_{Z,j} \chi_{Y,k})_{L^2(\Omega)} \tag{2.29}$$

System (2.14) has $N_{tot} = (N_u + N_p + N_\phi + \sum_{i=1}^{8} N_{C_i} + N_T + \sum_{j=1}^{3} N_{H_j})$ equations for $N_{tot} + 6N_{const}$ unknowns, in practice, the equations to find the temporal coefficients of the fields of system (2.2) miss. In order to find them an interpolation procedure using Radial Basis Function g is adopted [23] :

- A new parameters matrix is defined as:

$$\widehat{M}^* = \begin{pmatrix} t_1 & \boldsymbol{\mu}_1 \\ \vdots & \\ t_K & \boldsymbol{\mu}_1 \\ \vdots & \\ t_1 & \boldsymbol{t}_i \\ \vdots & \\ t_K & \boldsymbol{\mu}_i \\ \vdots & \\ t_1 & \boldsymbol{\mu}_{N_{off}} \\ \vdots & \\ t_K & \boldsymbol{\mu}_{N_{off}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{\mu}_1^* \\ \vdots \\ \boldsymbol{\mu}_{N_S}^* \end{pmatrix} \tag{2.30}$$

  That is, the same $\boldsymbol{\mu}_i$ is concatenated with all the saved time step t_k, so that $\widehat{M}^* \in \mathbb{R}^{N_S \times n+1}$

- Let's consider any of the field of system (2.2), $Z(\boldsymbol{x}, t) \approx \sum_{i=1}^{N_{const}} z_i(t)\chi_{Z,i}(\boldsymbol{x})$, the task is to construct:

$$z_i(\boldsymbol{\mu}^*) = \sum_{j=1}^{N_S} w_{i,j} g_{i,j}(\left\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_j^*\right\|_2) \;\; \forall j = 1, \ldots, N_{const} \tag{2.31}$$

- the quantity $z_{i,k}$ is found as:

$$z_{i,k} = (Z_k, \chi_{Z,i})_{L^2(\Omega)} \;\; \forall k = 1, \ldots, N_S \tag{2.32}$$

- it is used to solve the following linear system to find the weights $\boldsymbol{w}_i = \{w_{i,1}, \ldots, w_{i,N_S}\}$:

$$\sum_{j=1}^{N_s} w_{i,j} g_{i,j}(\left\|\boldsymbol{\mu}_k^* - \boldsymbol{\mu}_j^*\right\|_2) = z_{i,k} \;\; \forall k = 1, \ldots, N_S \tag{2.33}$$

  Therefore N_{const} system must be solved in order to find all the $\boldsymbol{z}(\boldsymbol{\mu}^*) = \{z_1(\boldsymbol{\mu}^*), \ldots, z_{N_{const}}(\boldsymbol{\mu}^*)\}$, all the weights are then stored.

- In the online stage a new parameter, still containing the time, $\boldsymbol{\mu}_{new}^*$ is used to evaluate the new value of the temporal coefficient as:

$$z_i(\boldsymbol{\mu}_{new}^*) = \sum_{j=1}^{N_S} w_{i,j} g_{i,j}(\left\|\boldsymbol{\mu}_{new}^* - \boldsymbol{\mu}_j^*\right\|_2) \;\; \forall i = 1, \ldots, N_{const} \tag{2.34}$$

- In conclusion this procedure is thus adopted to evaluate $\boldsymbol{\eta}, \boldsymbol{\gamma}, \boldsymbol{\zeta}, \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{\epsilon}$ needed to solve the system, then the fields $D, \Gamma, \Sigma_a, \Theta, V, E$ are reconstructed being their modes computed like for all the other fields as explained in section 2.1

The remaining part of this section is used to explain how to enforce non-homogeneous Dirichlet boundary conditions at ROM level, this is done only for $\boldsymbol{u}$ and T. The task is accomplished computing a lift function for each Dirichlet boundary, then an homogenized field is computed as:

$$\boldsymbol{u}' = \boldsymbol{u} - \sum_{i=1}^{N_{\boldsymbol{u},Dir}} u_{i,Dir}\boldsymbol{\psi}_{\boldsymbol{u},i} \tag{2.35a}$$

$$T' = T - \sum_{i=1}^{N_{T,Dir}} T_{i,Dir}\psi_{T,i} \tag{2.35b}$$

Where $u_{i,Dir}$ and $T_{i,Dir}$ are the values of the velocity and the temperature at the considered i-th Dirichlet boundary, while $\boldsymbol{\psi}_{\boldsymbol{u},i}$ and $\psi_{T,i}$ are their lift functions respectively and both of them assume unitary value at the i-th boundary, zero value for the other Dirichlet ones and inside the domain, the same BC condition of $\boldsymbol{u}$ and T for the remaining ones. We ask for this functions to be similar to the respective fields, therefore $\boldsymbol{\psi}_{\boldsymbol{u},i}$ must be divergence free, and $\psi_{T,i}$ to be governed by a source type energy equation. Thus, they are found solving the following problems:

$$\nabla \cdot \boldsymbol{\psi}_{\boldsymbol{u},i} = 0 \quad with \quad \nabla^2 p = 0 \tag{2.36a}$$

$$(\boldsymbol{u} \cdot \nabla)\psi_{T,i} = \frac{\nu}{Pr}\nabla^2\psi_{T,i} + \frac{\Sigma_{P,0}(1-\beta_{h,tot})}{\rho_0 c_p}S_1 \tag{2.36b}$$

Where S_1 is a unitary source, with the dimension of the neutron flux. Since boundary conditions are constant in time at FOM level, in fact $u_{i,Dir}$ and $T_{i,Dir}$ do not depend on time, the equations are steady. This technique not only stabilizes the solution at ROM level, but can be used to perform parametric variation of boundary conditions as well (not tested in this work).
The POD modes for these two fields are then found using the homogeneous snapshots matrices $\widehat{U}' = (\boldsymbol{u}'_1, \dots, \boldsymbol{u}'_{N_S})$ and $\widehat{T}' = (T'_1, \dots, T'_{N_S})$. The total number of modes to reconstruct the fields is modified in:

$$N'_{\boldsymbol{u}} = N_{\boldsymbol{u}} + N_{\boldsymbol{u},Dir} \tag{2.37a}$$

$$N'_T = N_T + N_{T,Dir} \tag{2.37b}$$

And the lift functions $\boldsymbol{\psi}_{\boldsymbol{u},i}$ and $\psi_{T,i}$ will constitute the first $N_{\boldsymbol{u},Dir}$ and $N_{T,Dir}$ modes respectively. Consequently, in order to solve system (2.14) additional values of $\boldsymbol{a}(t)$ and $\boldsymbol{e}(t)$ are required, thus they are set to:

$$a_i(t) = u_{i,Dir} \quad \forall i = 1, \dots, N_{\boldsymbol{u},Dir} \tag{2.38a}$$

$$e_i(t) = T_{i,Dir} \quad \forall i = 1, \dots, N_{T,Dir} \tag{2.38b}$$

2.3 Sensitivity Analysis

In a very synthetic and non exhaustive sentence, sensitivity analysis aims at ranking the parameters of a certain model according to their influence on the output of such model.

In this work, this is accomplished in the determination of the so called Standardized Regression Coefficients (SRCs)[24]. The main idea is to linear regress a generic output y of the model:

$$y = m(\boldsymbol{\mu}) = m(\mu_1, \dots, \mu_n) \tag{2.39}$$

To do that, a matrix $\widehat{S}$, called *Sampling Set*, is built:

$$\widehat{S}^{=} \begin{pmatrix} \mu_{1,1} & \dots & \mu_{1,n} \\ \vdots & \dots & \vdots \\ \mu_{i,1} & \dots & \mu_{i,n} \\ \vdots & \dots & \vdots \\ \mu_{m,1} & \dots & \mu_{m,n} \end{pmatrix} \tag{2.40}$$

Each j-th column of $\widehat{S}$ is Monte Carlo sampled from the marginal distribution of each parameter $\mu_j \ \forall j = 1, \dots, n$. The model output is computed for each row of $\widehat{S}$, thus obtaining a vector $\boldsymbol{y} = \{y_1, \dots, y_m\}$. Before to proceed with the linear regression, $\widehat{S}$ and $\boldsymbol{y}$ must be standardized in the following way:

- Compute the mean and the standard deviation of $\boldsymbol{y}$ and all the parameter μ_j:

$$\bar{y} = \frac{1}{m} \sum_{i=1}^{m} y_i \qquad s_y = \sqrt{\frac{1}{m-1} \sum_{i=1}^{m} (y_i - \bar{y})^2} \tag{2.41a}$$

$$\bar{\mu}_j = \frac{1}{m} \sum_{i=1}^{m} \mu_{i,j} \qquad s_{\mu_j} = \sqrt{\frac{1}{m-1} \sum_{i=1}^{m} (\mu_{i,j} - \bar{\mu}_j)^2} \tag{2.41b}$$

- Compute the standardized $\widehat{S}_s$ and $\boldsymbol{y}_s$, whose elements are defined as:

$$y_{s,i} = \frac{y_i - \bar{y}}{s_y} \tag{2.42a}$$

$$S_{s,ij} = \frac{\mu_{i,j} - \bar{\mu}_j}{s_{\mu_j}} = \tilde{\mu}_{i,j} \tag{2.42b}$$

- Seek a regression model for:

$$\frac{\tilde{y}_i - \bar{y}}{s_y} = \sum_{j=1}^{n} \alpha_j \tilde{\mu}_{i,j} \tag{2.43}$$

- α_j are the coefficients that minimize the errors $\epsilon_i = \tilde{y}_i - y_i$, and are found solving ordinary least square minimization, i.e.:

$$(\widehat{S}_s^{\intercal}\widehat{S}_s)\boldsymbol{\alpha} = \widehat{S}_s^{\intercal}\boldsymbol{y}_s \tag{2.44}$$

Therefore, the output of the linear approximation are computed as:

$$\tilde{y}_i = \bar{y} + s_y \sum_{j=1}^{n} \alpha_j \tilde{\mu}_{i,j} \tag{2.45}$$

While α_js are the SRCs. One advantage of this method is that in principle it explores the entire interval of definition of each factor. Another is that each "effect" for a factor is in fact an average over the possible values of the other factors. Moreover, SRCs also give the sign of the effect of an input factor on the output, providing a simplified model of the input–output mapping.

Of course this is true if the linear approximation is capable to correctly describe the model, an important factor to quantify that is the so called Quality of the linear regression, defined as:

$$R_y^2 = \frac{\sum_{i=1}^{m}(\tilde{y}_i - \hat{y})^2}{\sum_{i=1}^{m}(y_i - \hat{y})^2} \quad R_y^2 \in [0,1] \tag{2.46}$$

As stated in [24]: "*if the fit of the regression is good, e.g.* R_y^2 *is larger than, say,0.7, this means that the regression model is able to represent a large part of the variation of* y*. This also means that the model is relatively linear. In such cases, the regression model is effective and we can base the sensitivity analysis on it*".

(This page is intentionally left blank)

Chapter 3

Implementation of the problem in ITHACA-FV

All the procedures explained until this point, are realized by the extension of an already existing library of OpenFOAM, ITHACA-FV [16] developed at SISSA university in Trieste. It is a C++ library which implements all the classes needed to compute the POD modes (both EVD and SVD), OpenFOAM objects manipulation, including I/O methods, different FOM problems and correspondent ROMs. Moreover, it also uses third party libraries, Eigen [17], Spectra [18] and Splinter [19], already provided in the source file and extensively used throughout the library.

The basic reason why the library was developed is to provide the user the possibility to perform reduced order modelling simply writing a C++ program, thus overcoming all the difficulties such a task would meet using "raw" OpenFOAM utilities, one example for all: the parametric variation of the characteristic constants of the problem. Focusing on FOM problems, a very general base class *reductionProblem*, containing all the basic methods and members to set the offline stage, is defined; then each specific problem is derived from that, with public inheritance, adding specific members and, above all, the methods to solve the offline stage and perform the Galerkin projection. In a specular way ROM problems are dealt, whose base class is *reducedProblem*. Moreover, this two parts are strictly connected since the reduced object is initialized with the full order one.

Following the implementation of the Navier-Stokes problem of the library, the classes shown in figure 3.1 are implemented.

The *msrProblem* class corresponds to the steady state problem of the MSFR. It must be highlighted that it is only used as a base class for the real problem considered in this work, in other words it contains the constants, fields, projection matrices and projection methods to fully manage the physics and Galerkin projections of MSFR, but the offline solver is not tested. Therefore, the *usmsrProblem*, the unsteady problem class, inherits all those quantities, adds all the time settings, modifies the governing equations and, above all, fully implements the offline solver (section 3.1). This is done to guarantee the symmetry in the implementation of the problems in ITHACA-FV, looking ahead to a future development of the steady state problem. Analogous speech for *reducedMSR* and *reudecusMSR* respectively (section 3.2).

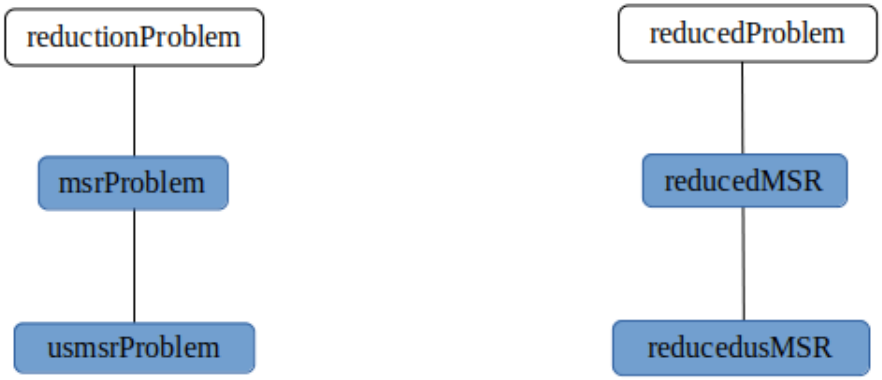


Figure 3.1: FOM and ROM classes implemented in ITHACA-FV.

Sensitivity analysis is still added to the library in order to carry on all the procedure in its environment. This is completely new in ITHACA-FV and it is added in the section that contains the basic classes to compute the POD modes, fields manipulation, etc...which name is *ITHACA_CORE*. See section 3.3 for more details.

The full code can be found on my personal fork of the library on github at `https://github.com/dakho/ITHACA-FV`

3.1 Full Order Model implementation

As written in the previous section, the problem is implemented in a C++ library, therefore an object-oriented implementation seems the more natural and effective. The object *usmsrProblem* contains all the methods to solve a parametric problem varying some specified constants of the MSFR, which are read and initialized from dictionaries [15], as a typical OpenFOAM solver. The same is true for the convergence schemes, the algorithms and the turbulence model (that in this case must be kept to *laminar*). Also the dictionary that controls the advance of the solution, *controlDict*, is necessary to start and run the simulation as usual, even if the time settings are actually overriden by FOM objects members. Another dictionary is defined, named *ITHACAdict*, which contains the settings for the POD modes calculation and projection.

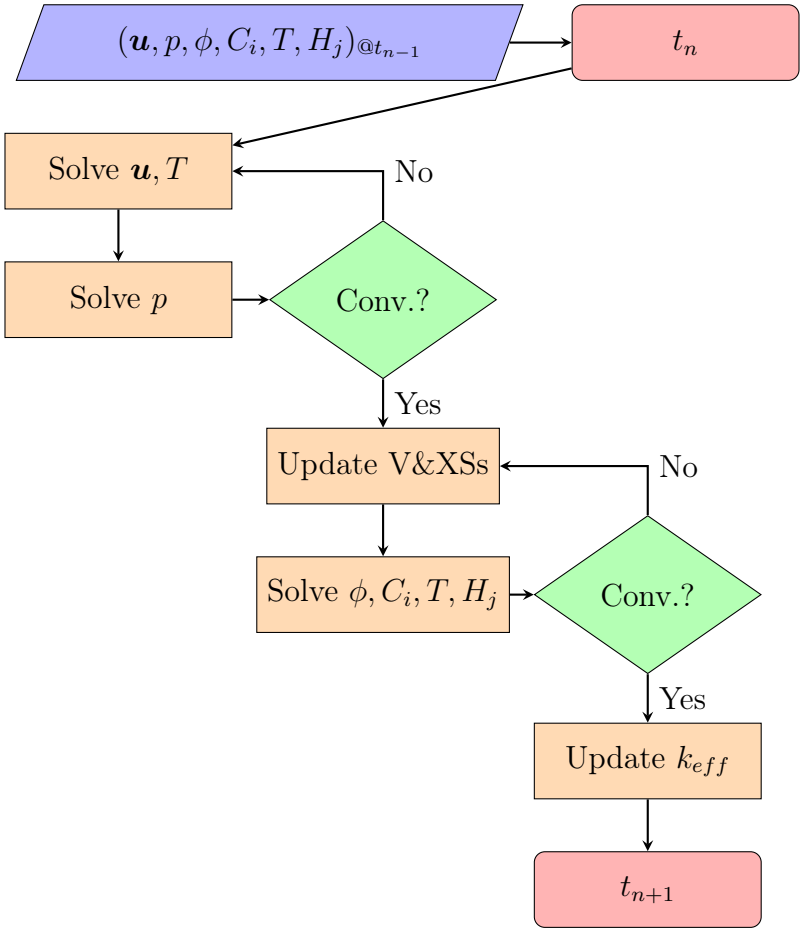


Figure 3.2: FOM solution scheme.

Focusing on the offline solver (*truthSolve* method), it is the same developed in [9], that is the combination of two PIMPLE loops [15], called *pimple* and *npimple*. The former is used to obtain the velocity and the pressure field, together with a

first guess of the temperature one; this is used to update the constants of system (1.2), after that convergence has been reached. The latter loop then returns all the remaining fields along with the temperature. After having updated the k_{eff} value, the solver moves to the next time step. This can be easily visualized in figure 3.1. The solution of p field is represented in a different block to be consistent with the way it is solved in PIMPLE algorithm, that briefly consists in: solve the discretized momentum equation to compute the intermediate velocity field; compute the mass fluxes at the cells faces; solve the pressure equation and apply under-relaxation; correct the mass fluxes at the cell faces; correct the velocity on the basis of the new pressure field; repeat until convergence.

The fields are then stored in a *List* of fields, one per each of them, and exported for every t_k and every i-th row of $\widehat{M}$ of eq. (2.2). Therefore, such a *List* contains N_S solutions and constitutes the snapshots matrix of that field.

Figure 3.3 shows instead the steps that are needed before to proceed on to the online stage. Computation of lift-functions and the consequent homogenization of $\boldsymbol{u}$ and T can be avoided if there are no Dirichlet boundaries in the case run. Lift-function for $\boldsymbol{u}$ is found using PISO algorithm [15], while SIMPLE [15] one for T. POD modes can be computed with both EVD and SVD techniques, using Eigen or Spectra solvers. This steps are still solved by object *usmsrProblem* methods and their solution stored in its members.

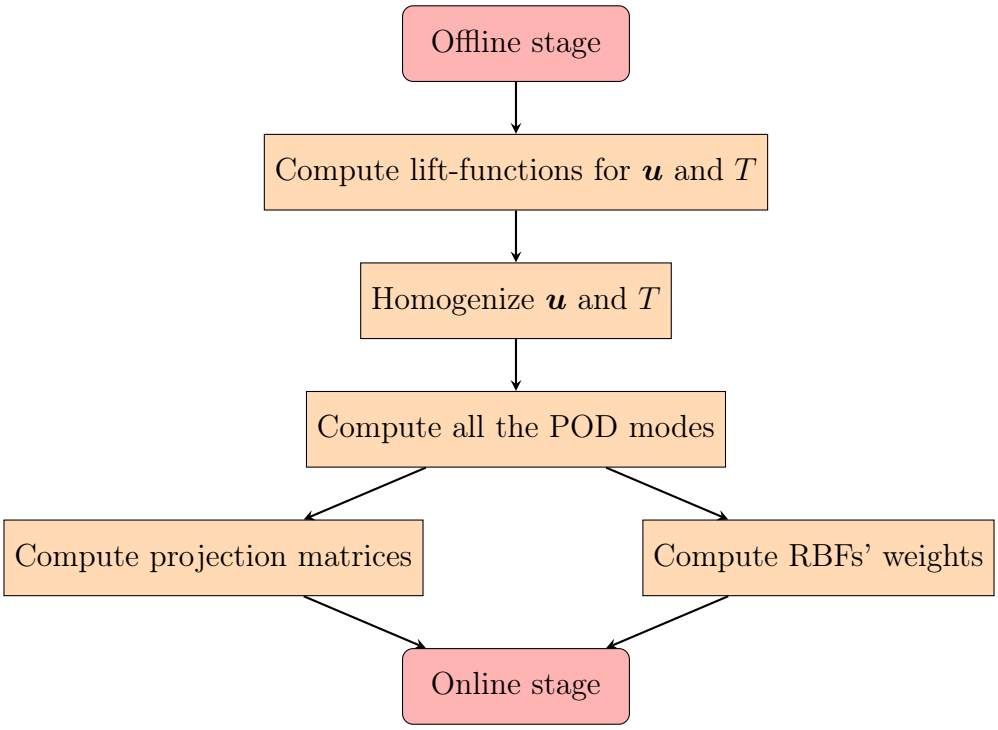


Figure 3.3: Intermediate steps between Offline and Online stage.

3.2 Reduced Order Model implementation

The object *reducedusMSR* contains the methods to solve system (2.14) and reconstruct all the fields. Having a look at system (2.14), the physics of the problem can be split into three parts: fluid-dynamics (F-D) (eq. (2.14a) and (2.14b)), neutronics (N) (eq. (2.14c) and (2.14d)) and energy (E) (eq. (2.14e) and (2.14f)) sub-systems. In fact, considering the order in which the equations are written, there is a vertical coupling among the three sub-systems, i.e. the solution of F-D permits to solve N, once this is done E can be finally solved. In particular, the velocity coefficients and the velocity plus flux ones, become known and can be passed to N and E respectively. This is schematized in figure 3.4. Differently from FOM model, no iteration between neutronics and energy is required to converge, since the information about their interaction is stored in the projection matrices relative to the fields in system (1.2).

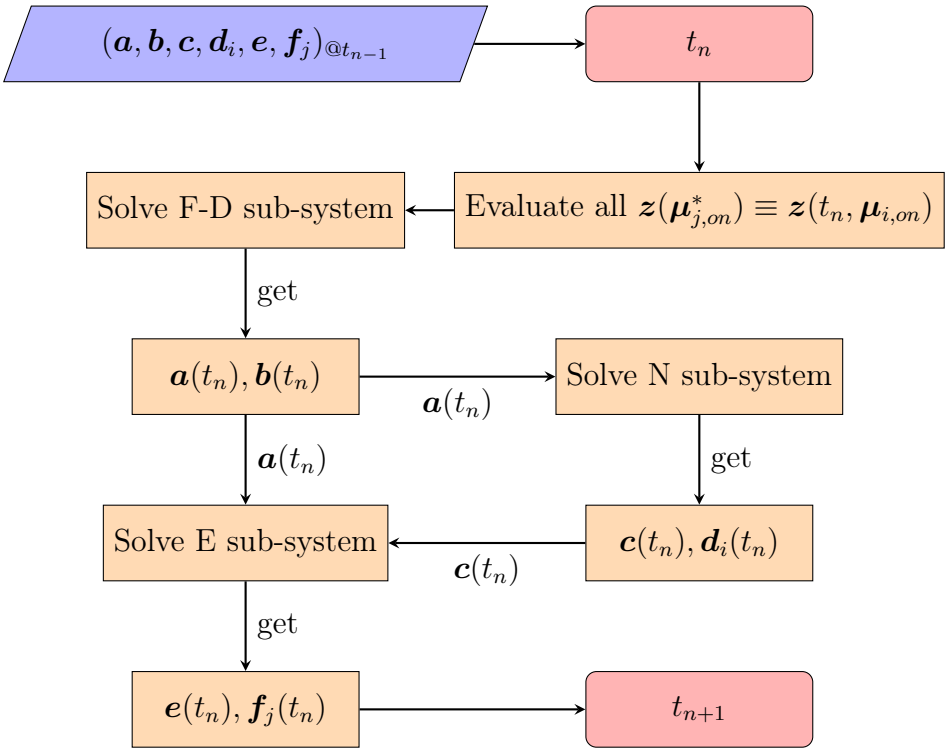


Figure 3.4: ROM solution scheme.

In the online stage solver a new set of parameters is passed to the reduced object, this, concatenated with the current time value, constitutes $\boldsymbol{\mu}^*_{j,on}$. This is used to evaluate the time coefficients of system (2.2) using RBF interpolation methodology explained in section 2.1.

At each time step the solution is found using the Newton-Raphson method, while the time discretization scheme adopted is the backward-Euler. Considering a generic field $F(\boldsymbol{x}, t) \approx \sum_{i=1}^{N_F} \omega_i(t)\chi_{F,i}(\boldsymbol{x})$ of system (2.1), the initial condition for its time

coefficients ω_i, are found solving the linear system:

$$\widehat{M_F}\boldsymbol{\omega} = \boldsymbol{b}_0 \tag{3.1}$$

Where $\boldsymbol{\omega} = \{\omega_1, \ldots, \omega_{N_F}\}$, $\widehat{M_F}$ is the mass matrix of F computed as (2.23), $\boldsymbol{b}_0$ is a known vector and each i-th element reads as:

$$b_{0,i} = (F_j(\boldsymbol{x}, t_0), \chi_{F,i})_{L^2(\Omega)} \tag{3.2}$$

Where $F_j(\boldsymbol{x}, t_0)$ is the initial condition of the field F correspondent to a chosen j-th row of the training set $\widehat{M}$ (eq. (2.5))

3.3 Sensitivity Analysis implementation

SA implementation in ITHACA-FV is accomplished defining three new classes:

- *ITHACAsampling*: it implements the Monte Carlo Inverse Transform Sampling method (as static method) to sample from the parameters distributions. Only four distributions are enabled: Normal, Uniform, Poisson and Exponential.
- *FofM* (and its derived classes): abstract class that permits to compute a desired figure of merit of both FOM or ROM fields, not only for MSFR. In this work, the average temperature and the total power are implemented (as derived classes of *FofM*). The definition of the field, on which the figure of merit is based, is the same of OpenFOAM.
- *LRSensitivity*: it defines an object to calculate the linear regression model and the SRCs. It has a *FofM* member and uses *ITHACAsampling* methods to construct the sampling set.

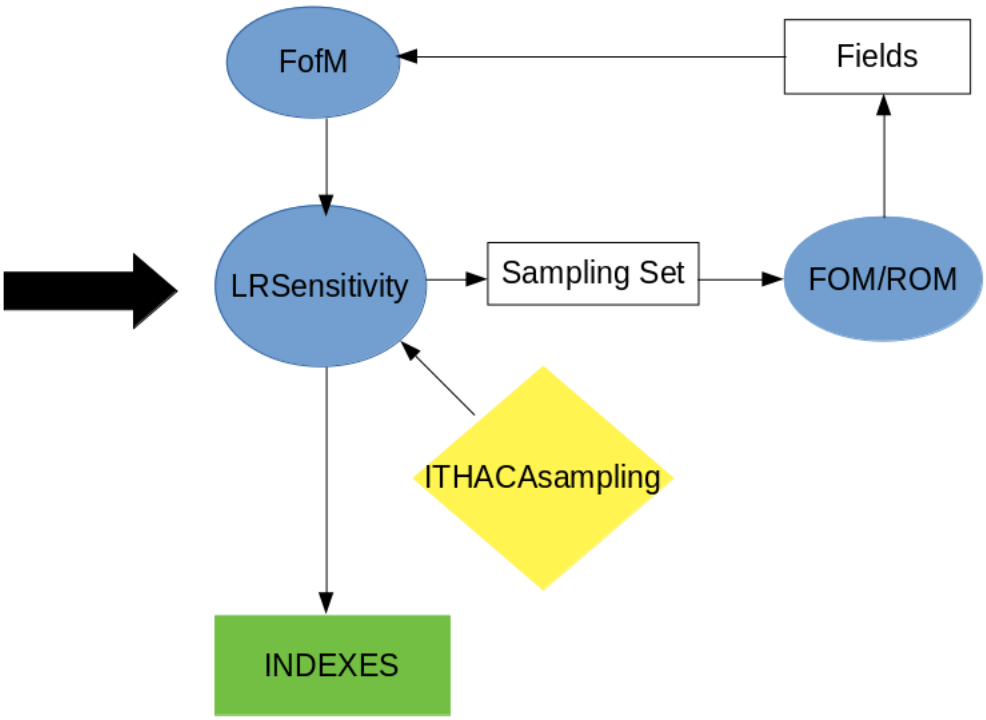


Figure 3.5: Procedure to carry on SA with ITHACA-FV.

Let us suppose that at some point in the code we want to perform SA, first of all *LRSensitivity* object must be initialized: it requires only the number of parameters considered in the analysis and the number of points to sample. Using *ITHACAsampling* methods it creates the sampling set (eq. (2.40)) rejecting the values that are

outside the training range (it must be explicitly defined by the user). This values can be passed both to FOM or ROM that return the fields of system (1.1) or (2.14) respectively. Then they are used by a *FofM* object that computes the figure of merit. Its values are assigned to *LRSensitivity* that finally computes the indexes. This procedure can be visualized in figure 3.5. In this section of the library an extensinve usage of Eigen and OpenFOAM objects is done, while distributions and the sampling technique come from C++ standard library.

(This page is intentionally left blank)

Chapter 4

Test cases and Results

In this section the results of the simulations performed are shown. In order to validate the code, all the simulations are run for a lid-driven cavity [21], widely used as a benchmark in numerical simulations. Three different simulations are performed, they differ because of the temperature and decay heat groups boundary conditions. This is done to check the influence of temperature lift-function(s) on the reduced solution. At the end of the chapter, one simulation outside training boundaries is presented as well.

In this case, the cavity consists of 0.1x0.1 m in x-z plane and the mesh is built with 50 square cells in both directions. With the same orientation as shown in 4.1 the boundaries are named: *Moving Wall* (M.W.), i.e. the top wall, *Right Wall* (R.W.), *Bottom Wall* (B.W.), *Left Wall* (L.W.).

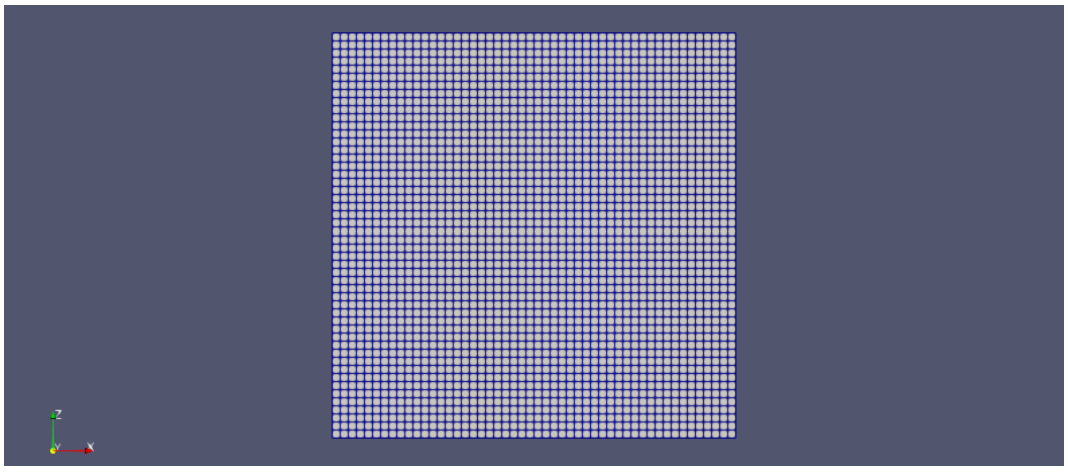


Figure 4.1: Mesh adopted for the simulations.

The Reynolds number, considering the reference viscosity of table 1.1, is set to 100, which translates in a velocity of $2.46 \cdot 10^{-3}$ $[m/s]$ in x-direction. As regards the offline stage, the final time is 150 s with time step of 0.1 s. This leads to a maximum expected Courant number of: 0.123. These information are summarized in table 4.1.

Quantity	Symbol	Value	Units
Reynolds number	Re	100	$[-]$
Characteristic length	L	0.1	$[m]$
Moving wall velocity	u_{MW}	$2.46 \cdot 10^{-3}$	$[m/s]$
Time step	Δt	0.1	$[s]$
Mesh size	$\Delta x \equiv \Delta z$	$2 \cdot 10^{-3}$	$[m]$
Courant number	Co	0.123	$[-]$
Final time	t_f	150	$[s]$

Table 4.1: Offline stage simulations numerical settings.

As regards the parametric variation during Offline, the idea is to explore the influence of characteristics parameter of the three-physics of the problem, therefore one per each of them is chosen:

$$\boldsymbol{\mu} = \{\nu, \beta_{tot}, \lambda_{h,3}\} \tag{4.1}$$

ν for the fluid-dynamics, β_{tot} for the neutronics and $\lambda_{h,3}$ for the energy equations respectively. It is further assumed that the ratios β_i/β_{tot} are kept constant to the reference values.

Consequently, these will be the quantities involved in the sensitivity analysis as well. It is supposed that all the three quantities have normal distribution, with standard deviation equal to 0.1/3 the reference value x_0, so that the 99.7% of their possible values are found in the interval $(x_0 - 0.1x_0\,;\, x_0 + 0.1x_0)$. Therefore, in the offline stage we train the system varying μ from $0.9\mu_0$ to $1.1\mu_0$, where μ_0 contains the values of ν_0, $\beta_{tot,0}$ and $\lambda_{h,3,0}$ reported in section 1.2. During Offline stage, their values are randomly sampled, with uniform distribution, inside this range. This choice may be argued, since in the sensitivity analysis the distribution is normal indeed. This is done because with a normal distribution the values are often picked very close the central value and a lot of samples may be needed to pick one value close the the extreme of the interval; by this way, with a limited number of samples, it is possible to explore more widely the training range with less points and train the ROM for points that would be unlikely but still possible in the sensitivity analysis.

Using the notation adopted throughout the text, the number of "training points", is called N_{off}, and it is chosen: $N_{off} = 31$. This choice is completely arbitrary but can effectively influence the solution, it is based on some qualitative simple tests performed at developing stage. In addition the central value of the training set, the 16-th one (or the 16-th row of matrix $\widehat{M}$ of eq. (2.5)), is replaced with μ_0, to be sure that this value is actually included in the offline stage. The time instants at which the solution is saved is chosen to be every 5 s, or every 500 time steps (plus the initial condition). Therefore K=150/5+1 and then the number of snapshots $N_S = K * N_{0ff} = 961$. Also the choice of K is arbitrary: in this case the dynamics

induced by the flow field can be considered slow, i.e. the time to establish a fully developed flow is long; the choice of K is based on this consideration as well as on some memory considerations since the simulations are run on a standard commercial laptop. Its characteristics are shown in figure 4.2. Of course the finer are the snapshots (or the higher is K), the best the ROM is expected to be capable to reproduce the true solution.
The number of saved modes for all the fields ($\widetilde{N}_F$ of section 2.1) is fixed fol all the three cases and set equal to 50. The POD modes are found using EVD technique and eigenvalue solver by Eigen. Finally, the online stage time step equals the FOM one, while the final time is 100 s, to be conservative in the capability of the ROM to reproduce the physics of the problem. Table 4.2 summarizes what just said.

Before to proceed on with SA, the ROM is tested adopting the values of $\boldsymbol{\mu}$ on which it is trained, and comparisons 5-by-5 s are shown in the following sections. The number of modes adopted for each field and test case are given in the correspondent section as well. Then the SA is performed with the distributions detailed above. Both FOM and ROM SA is performed in order to compare the outputs distributions and SRCs values.

Quantity	Symbol	Value	Units
# of offline simulations	N_{off}	31	$[-]$
Offline stage upper bound	$1.1 \cdot \mu_0$	$[\{m^2/s, -, 1/s\}]$	
Offline stage lower bound	$0.9 \cdot \mu_0$	$[\{m^2/s, -, 1/s\}]$	
# of saved time steps per simulation	K	31	$[-]$
Total # of snapshots	N_S	961	$[-]$
# of saved snapshots per field	$\widetilde{N}_F$	50	$[-]$
Online stage final time	$t_{f,o}$	100	$[s]$
Online stage time step	Δt_o	0.1	$[-]$

Table 4.2: Training settings during Offline stage and Online stage settings.

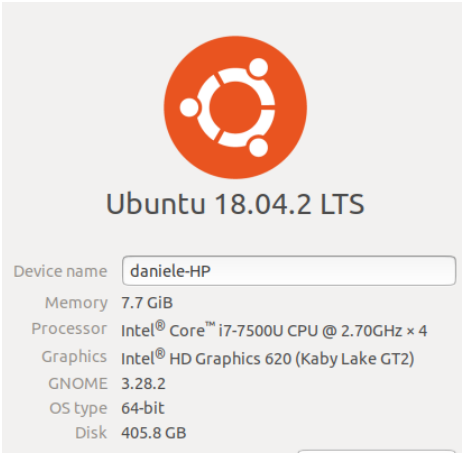


Figure 4.2: Technical characteristics of the laptop used to run the simulations.

4.1 Case A results

The initial and boundary conditions for case A are listed in the following table (using OpenFOAM nomenclature):

Field	I.C.	M. W.	R. W.	B. W.	L. W.
$\boldsymbol{u}$ $[m/s]$	$(0,0,0)$	$(2.46 \cdot 10^{-3}, 0, 0)$	*NoSlip*	*NoSlip*	*NoSlip*
$p; [Pa]$	0	*ZeroGrad.*	*ZeroGrad.*	*ZeroGrad.*	*ZeroGrad.*
ϕ $[1/(m^2 s)]$	10^{18}	0	0	0	0
C_i $[1/m^3]$	10^{19}	0	0	0	0
T $[K]$	1050	850	950	950	950
H_j $[J/m^3]$	0	*ZeroGrad.*	*ZeroGrad.*	*ZeroGrad.*	*ZeroGrad.*

Table 4.3: Initial and boundary conditions - case A.

Where *NoSlip* condition constraints the velocity to keep the same velocity of the boundary (therefore it is simply homogenous Dirichlet boundary condition in this case), *ZeroGrad.* stands for *ZeroGradient* and it is an homogeneous Neumann boundary condition. It must be highlighted that these initial conditions are completely arbitrary and must be considered as 0^- ones, in fact the FOM solver performs a preliminary iteration and finds the actual 0 condition starting from the values reported in table 4.3. The same speech is true for case B and case C.

The graphical representation of some fields can be found in the appendix. In this section other comparisons are shown. The first to be presented is the percentage L^2 error between FOM and correspondent ROM field, defined as:

$$e_F(t) = \frac{\Big[\int_\Omega d^2x \big(F(\boldsymbol{x},t) - F_r(\boldsymbol{x},t)\big)^2\Big]^{1/2}}{\Big[\int_\Omega d^2x \big(F(\boldsymbol{x},t)\big)^2\Big]^{1/2}} \cdot 100 \tag{4.2}$$

The computation of $e_F(t)$ is done every t_k, that is every 5 seconds. The results are shown in graphs 4.3 and 4.4 for the reference value of the parameters $\boldsymbol{\mu_0}$ that represents the 16-th $\boldsymbol{\mu}$ with which the system is trained, i.e. $\boldsymbol{\mu}_{16}$. Even if the temperature field presents some very hot and cold spots, lower than M.W. boundary condition (see picture 6.12), the integral of those differences can return very good results ($e_T(100s) \approx 1.6\%$). The results are even better if the difference is computed after an integration operation as demonstrated by figure 4.5 that shows the percentage difference between FOM and ROM average temperature T_m (integral average).

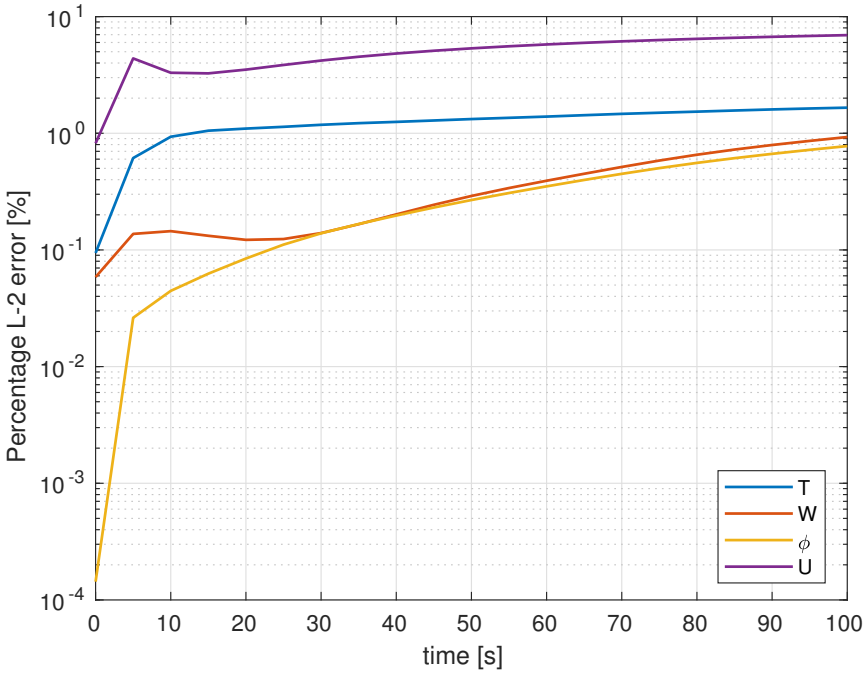


Figure 4.3: Percentage L^2 errors case A - 1.

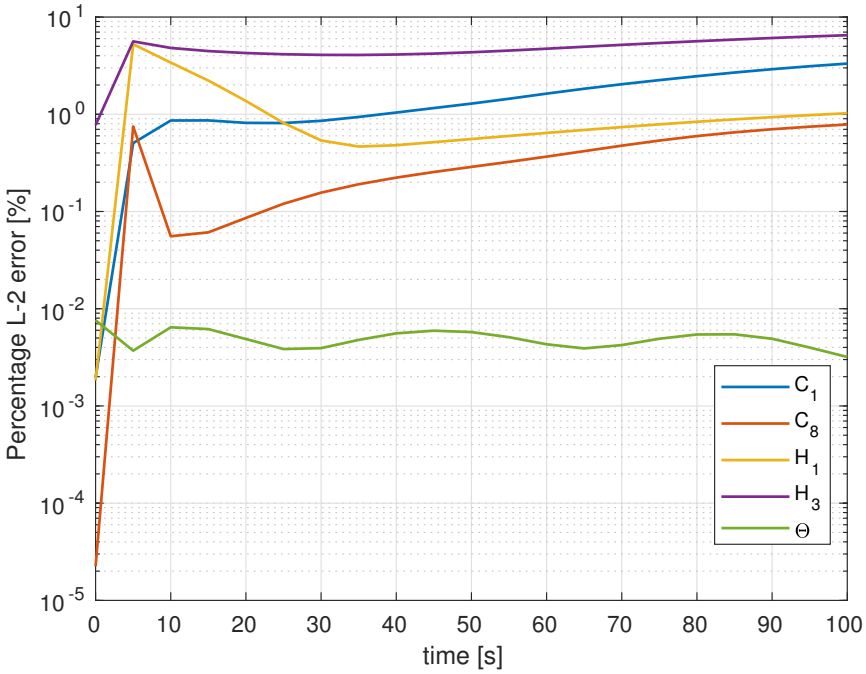


Figure 4.4: Percentage L^2 errors case A - 2.

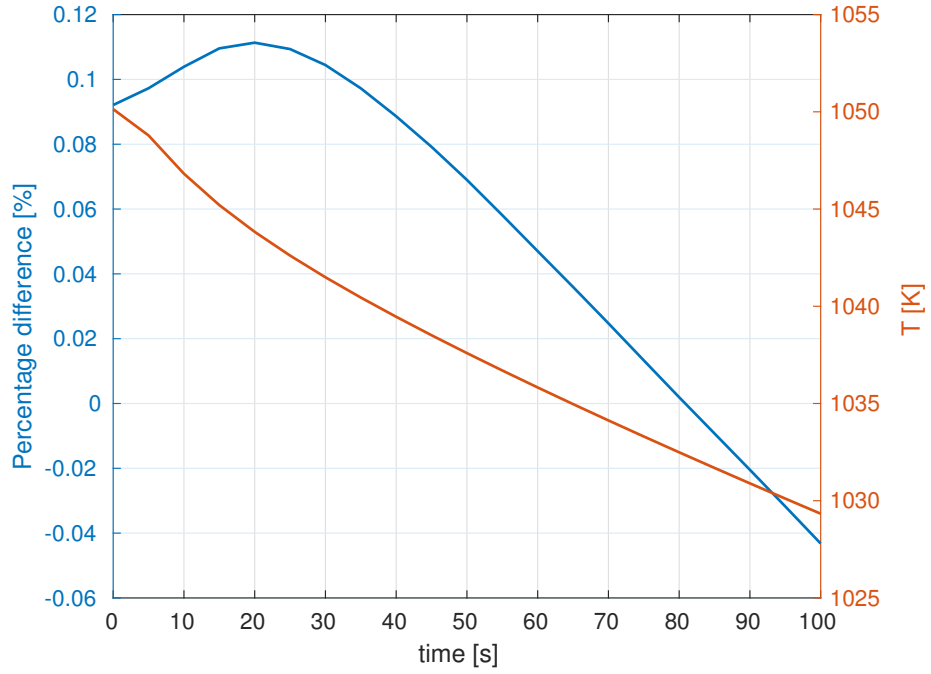


Figure 4.5: Percentage difference between FOM and ROM average temperature, computed as $(T_m - T_{m,r})/T_m \cdot 100$ - case A.

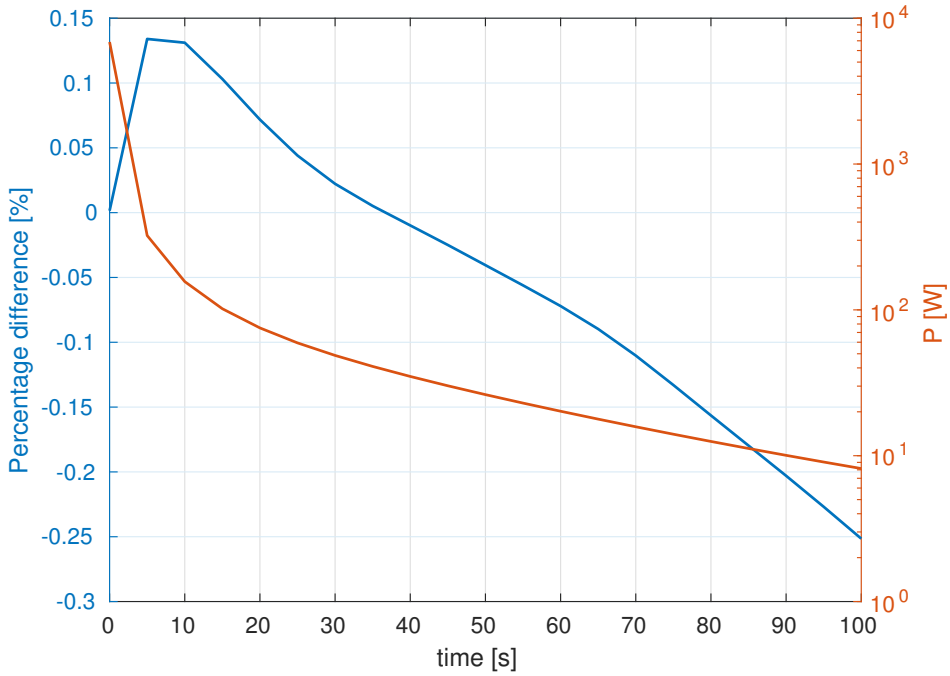


Figure 4.6: Percentage difference between FOM and ROM total power, computed as $(P - P_r)/P \cdot 100$ - case A.

The same is done for the total power inside the reactor along the transient (figure 4.6). Both graphs shows that the capability of the ROM to return integral quantities is very good. For the power density this is true also for local values, while this speech is very limited for T. Then as introduced in section 4, the last two comparisons are done for all the 31 values of $\boldsymbol{\mu}$. Figures 4.7 and 4.8 show the result of the central value $\boldsymbol{\mu}_{16}$ together with the values of $\boldsymbol{\mu}$ that leads to the upper and lower percentage difference observed.

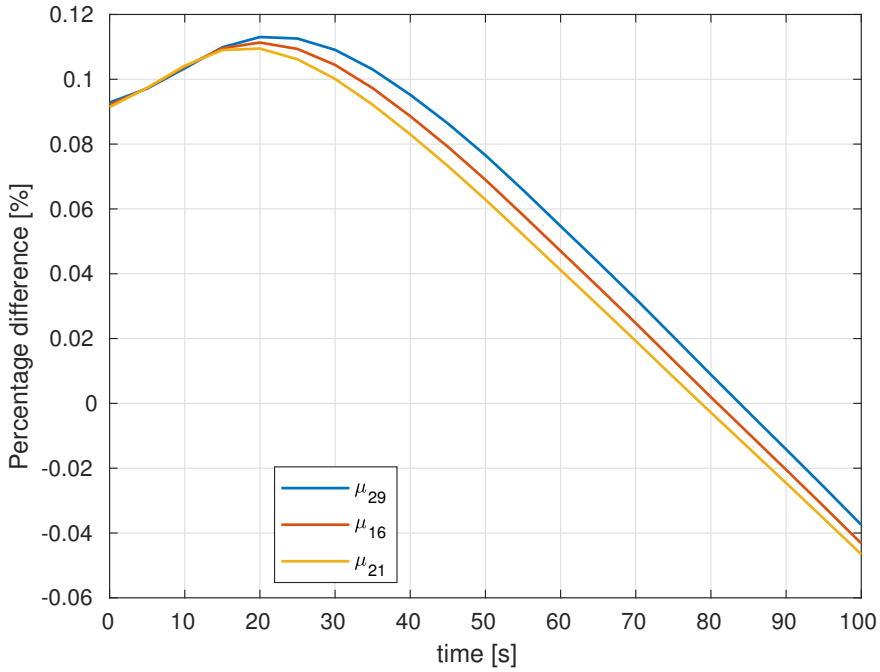


Figure 4.7: Average temperature percentage difference for different values of $\boldsymbol{\mu}$ - case A.

Even if it could be possible to select as initial condition for the time coefficients the correspondent FOM initial condition, it is chosen to take always the one the 16-th value of $\boldsymbol{\mu}$ returns, i.e. $F_j \equiv F_{16}$ in eq. (3.2) $\forall j = 1, \ldots, 31$. This is done for testing purpose to see what is the response of the system to this constraint. The results can be considered acceptable looking at figures 4.7 and 4.8. It must be recalled that here the ROM is run with the same value of $\widehat{M}$ on purpose, but in SA the values of $\boldsymbol{\mu}$ will be different. This test can say that the choice of the initial value can have an impact, but still very limited. Another choice could be to use the value of $\boldsymbol{\mu}$ that has the closer L^2-norm to the online value $\boldsymbol{\mu}_{on}$, but the parameters inside $\boldsymbol{\mu}$ have orders of magnitude of difference among them, and the norm would be dominated by the value of β_{tot}, but this dominance in the norm can not being assumed to be dominant in the result as well, this is the reason why SA is actually performed, to some extent. Then the choice of $\boldsymbol{\mu}_{16}$ seems the best, considering also that in SA stage the sampled values will be more closely located around its value.

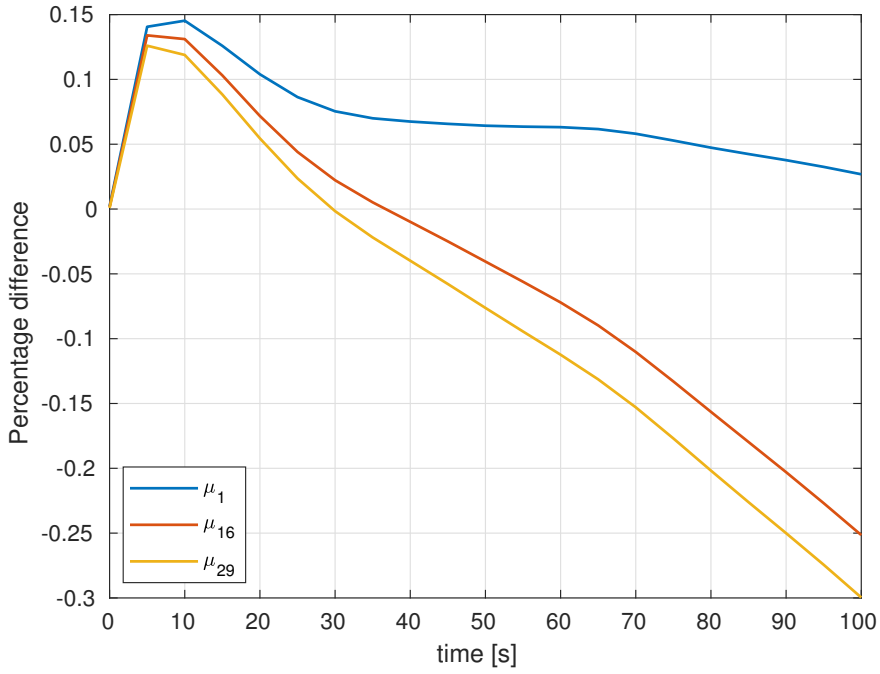


Figure 4.8: Total power percentage difference for different values of $\boldsymbol{\mu}$ - case A.

The values of $\boldsymbol{\mu}$ appearing in the plots are listed in table 4.4. The sampled maximum and minimum values of the three parameters are reported in table 4.5 instead, these constitute the Training Range adopted in sensitivity analysis.

$\boldsymbol{\mu}$	$\nu\ [m^2/s]$	$\beta_{tot}[-]$	$\lambda_{h,3}\ [1/s]$
$\boldsymbol{\mu}_1$	$2.293 \cdot 10^{-6}$	$299.97 \cdot 10^{-5}$	$3.292 \cdot 10^{-4}$
$\boldsymbol{\mu}_{16}$	$2.460 \cdot 10^{-6}$	$321.80 \cdot 10^{-5}$	$3.580 \cdot 10^{-4}$
$\boldsymbol{\mu}_{21}$	$2.698 \cdot 10^{-6}$	$348.44 \cdot 10^{-5}$	$3.622 \cdot 10^{-4}$
$\boldsymbol{\mu}_{29}$	$2.214 \cdot 10^{-6}$	$349.92 \cdot 10^{-5}$	$3.253 \cdot 10^{-4}$

Table 4.4: Values of $\boldsymbol{\mu}$ represented in figure 4.7 and 4.8.

Quantity	Min	Max
$\nu\ [m^2/s]$	$2.214 \cdot 10^{-6}$	$2.698 \cdot 10^{-6}$
$\beta_{tot}\ [-]$	$291.37 \cdot 10^{-5}$	$352.69 \cdot 10^{-5}$
$\lambda_{h,3}\ [1/s]$	$3.253 \cdot 10^{-4}$	$3.910 \cdot 10^{-4}$

Table 4.5: Minimum and Maximum values of $\boldsymbol{\mu}$ sampled at Offline stage, i.e Training Range - case A, B, C, A'.

Finally, he number of modes adopted for projection/reconstruction are listed in table 4.6. As regards the velocity and temperature fields, the first number refers to $N_{\boldsymbol{u}}$ and N_T respectively, while the second one is the number of lift-function(s) adopted, that means $N_{\boldsymbol{u},Dir}$ and $N_{T,Dir}$ respectively (see section 2.2)

Field	# of modes adopted for projection/reconstruction
$\boldsymbol{u}$	10+1
p	5
ϕ	8
C_1	18
C_2	15
C_3	15
C_4	25
C_5	25
C_6	13
C_7	20
C_8	35
T	25+4
H_1	10
H_2	10
H_3	13
System (2.2) fields	5

Table 4.6: Number of modes adopted both for projection and reconstruction stage, $\boldsymbol{u}$ and T add the number of lift-function(s) - case A, A'.

4.2 Case B results

In this case the temperature initial and boundary conditions are changed, in particular there is only one Dirichlet boundary (the *Moving Wall*), while all the others are adiabatic. To be consistent with that, decay heat groups BCs are changed to *ZeroGradient* and uniform Dirichlet respectively for *Moving Wall* and the remaining ones, as summurized in the following table:

Field	I.C.	M. W.	R. W.	B. W.	L. W.
$T\ [K]$	1023	850	*ZeroGrad.*	*ZeroGrad.*	*ZeroGrad.*
$H_j\ [J/m^3]$	0	*ZeroGrad.*	0	0	0

Table 4.7: Initial and boundary conditions for T and H_j - case B.

The values of $\boldsymbol{\mu}$ adopted in this case are the same of case A, since the seed of the pseudo-random generator is fixed (the same is true for case C). This gives the possibility to compare the system evolution with the same values of the parameters. In case B, $e_T(100s)$ increases to $\approx$ 4.8%, this could be explained looking at figure

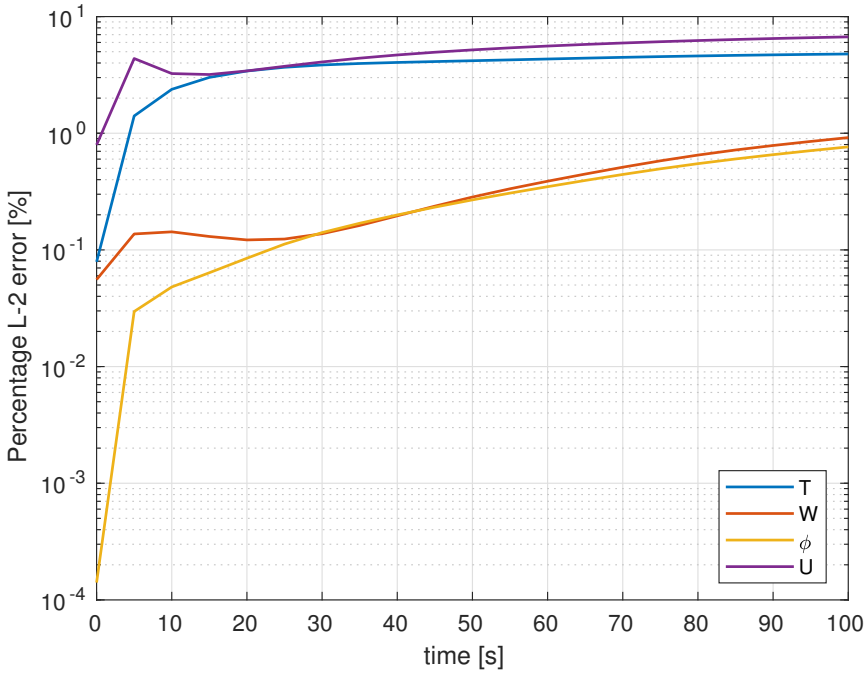


Figure 4.9: Percentage L^2 errors case B - 1.

6.22: hot and cold spots are more widely located and higher in magnitude, leading to more cells where the difference between FOM and ROM is higher, thus increasing the numerator of e_T. Interestingly, no practical difference sets for H_1 and H_3 ($e_{H_3}(100s) = 6.50\%$ for case A, 6.42% for case B) even if boundary conditions are different, the same for Θ.

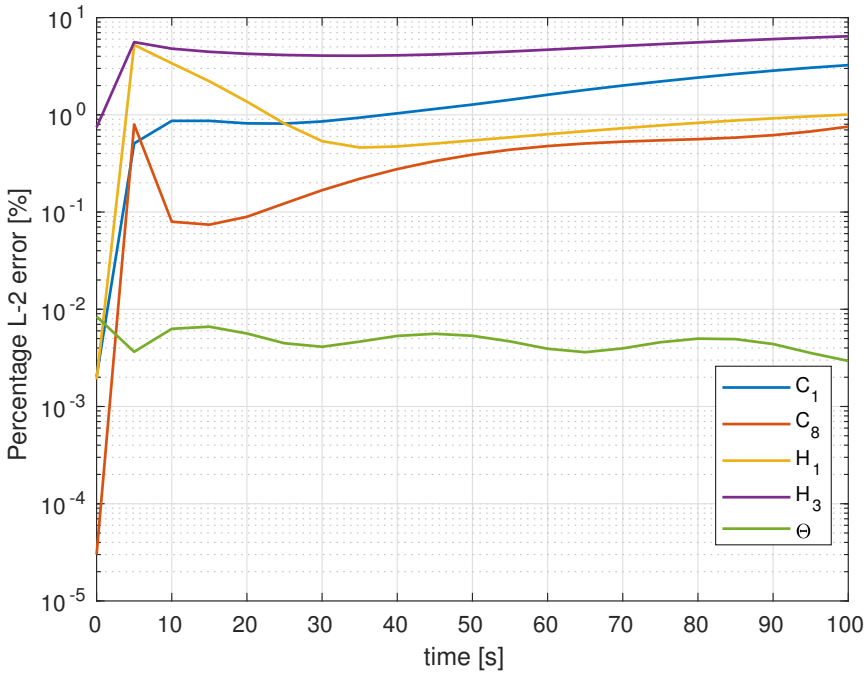


Figure 4.10: Percentage L^2 errors case A - 2.

Instead, as expected, other quantities are reproduced with the same accuracy. Some very low discrepancies are due to the different values of fields of system (1.2) because of temperature different distribution and value.

$e_F(t)$ [%]	A	B
$e_{\boldsymbol{u}}(100s)$	6.93	6.70
$e_\phi(100s)$	0.77	0.76
$e_W(100s)$	0.93	0.92
$e_T(100s)$	1.65	4.78
$e_{H_1}(100s)$	1.02	1.00
$e_{H_3}(100s)$	6.50	6.42

Table 4.8: Comparison between case A and B $e(100s)$ results for different fields.

In this case, an initial heating of the fuel happens, differently from case A, basically because the initial high power level heats it up and the three adiabatic walls help to contain the heat generated, then the mean temperature drops almost linearly because of cooling action imposed at the Moving Wall. Instead, the power dynamics is practically the same of case A, together with the percentage difference. Even if the temperature L^2 relative error is much higher, the ROM is still capable to return very good approximation of the average temperature, the percentage difference is in fact of the same order of (figures 4.11 and 4.12).

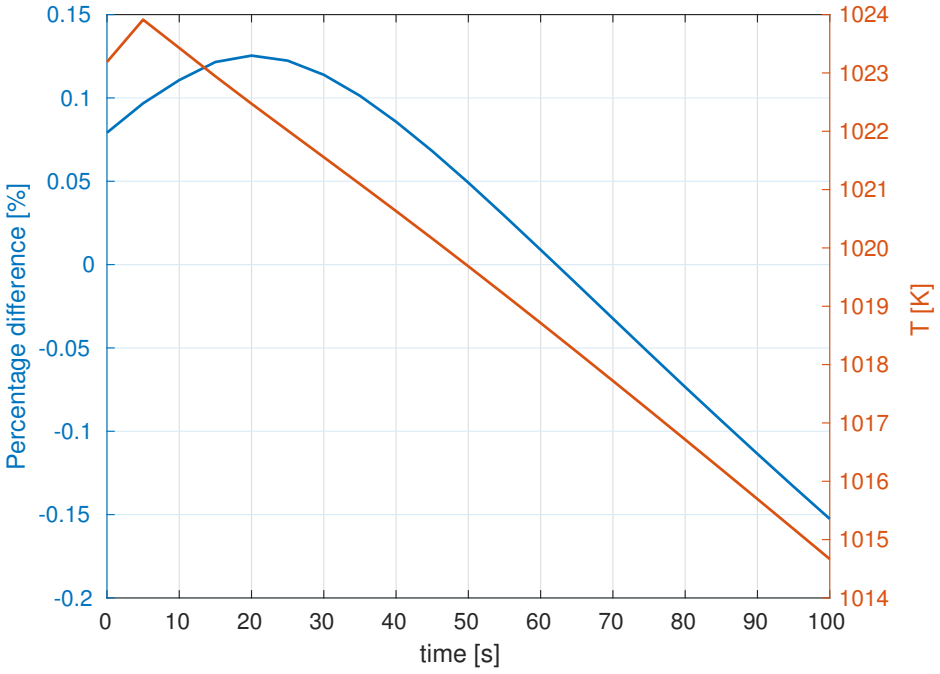


Figure 4.11: Percentage difference between FOM and ROM average temperature, computed as $(T_m - T_{m,r})/T_m \cdot 100$ - case B.

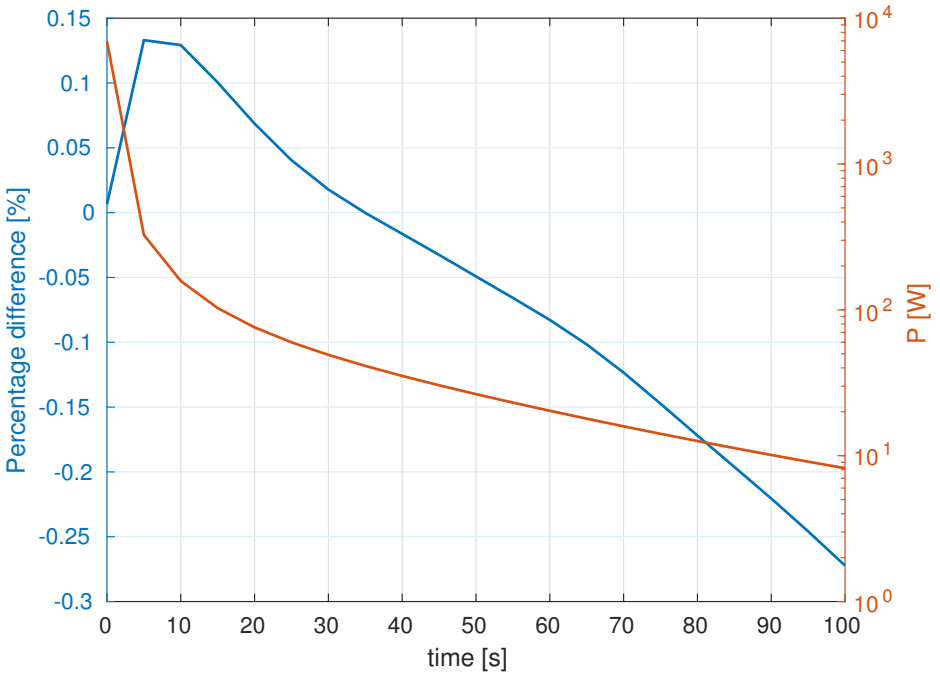


Figure 4.12: Percentage difference between FOM and ROM total power, computed as $(P - P_r)/P \cdot 100$ - case B.

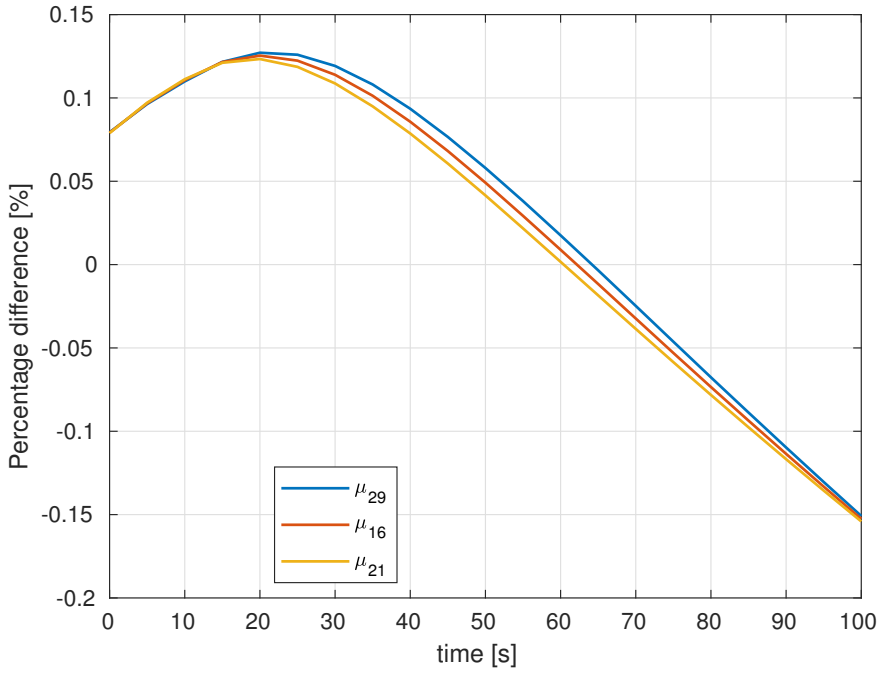


Figure 4.13: Average temperature percentage difference for different values of $\boldsymbol{\mu}$ - case B.

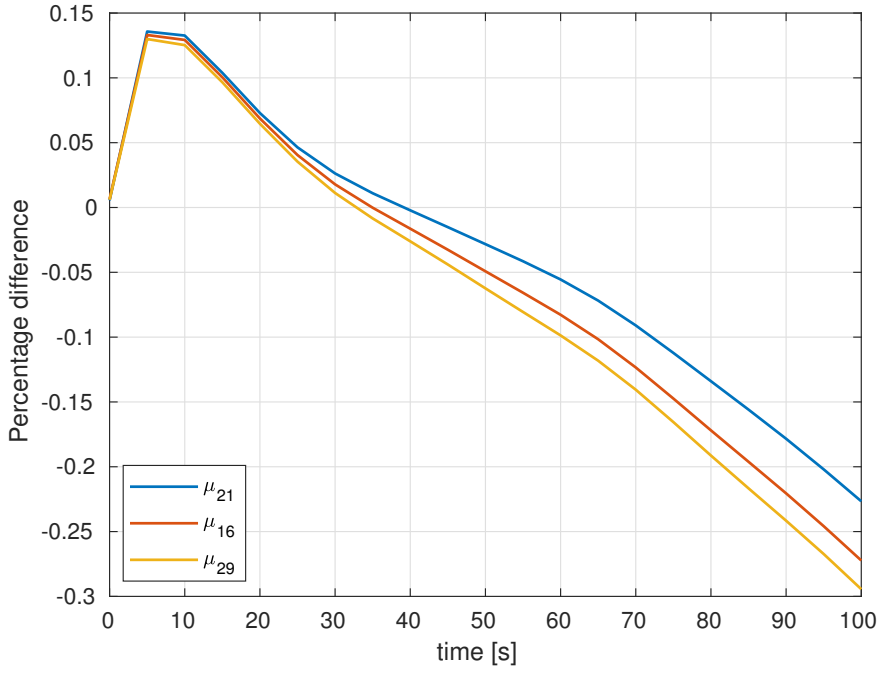


Figure 4.14: Total power percentage difference for different values of $\boldsymbol{\mu}$ - case B.

The same comparison of figures 4.7 and 4.8 are shown in figures 4.13 and 4.14 respectively for case B: the lower and upper bound are found for the same values of $\boldsymbol{\mu}$ but in case B the band is much more narrow; in particular the temperature results show the same final value for the three values of $\boldsymbol{\mu}$. Finally the number of modes adopted are listed:

Field	# of modes adopted for projection/reconstruction
$\boldsymbol{u}$	10+1
p	5
ϕ	8
C_1	18
C_2	15
C_3	15
C_4	25
C_5	25
C_6	13
C_7	20
C_8	35
T	27+1
H_1	10
H_2	10
H_3	13
System (2.2) fields	5

Table 4.9: Number of modes adopted both for projection and reconstruction stage, $\boldsymbol{u}$ and T adds the number of lift-function(s) - case B.

4.3 Case C results

In this section a complete adiabatic cavity is simulated, the initial and boundary conditions are summarized in table 4.10. Then the L^2 percentage error is presented as usual (figures 4.15 and 4.16).

Field	I.C.	M. W.	R. W.	B. W.	L. W.
$T\ [K]$	973	*ZeroGrad.*	*ZeroGrad.*	*ZeroGrad.*	*ZeroGrad.*
$H_j\ [J/m^3]$	0	0	0	0	0

Table 4.10: Initial and boundary conditions for T and H_j - Case C.

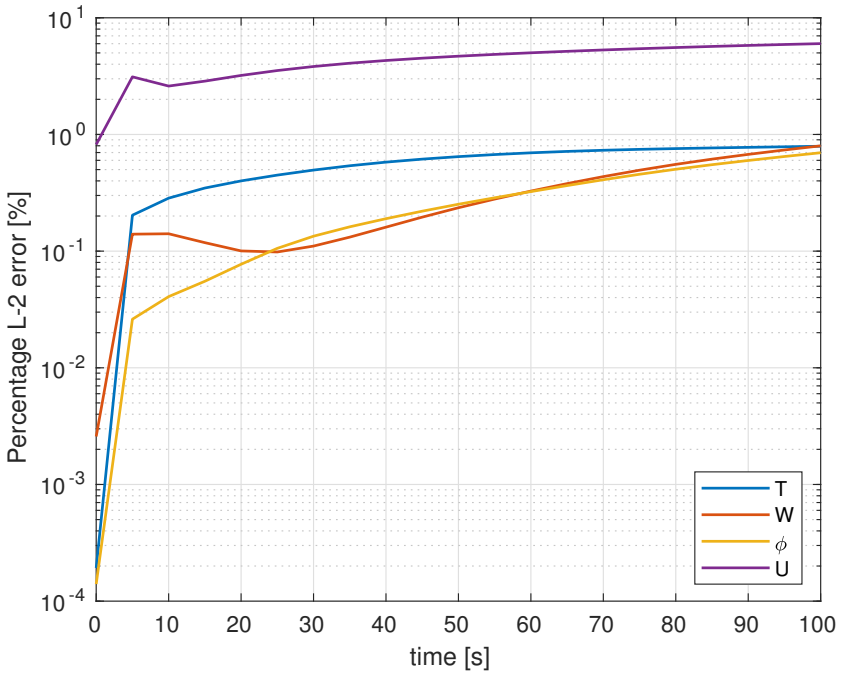


Figure 4.15: Percentage L^2 errors case C - 1.

In this case the temperature field lacks of any spatial reconstruction (see figure 6.28), a part for the initial condition. Given that, the L^2 percentage error is the best among the three different cases ($e_T(100s) \approx 0.8\%$). In addition the quantities involved as sources in the temperature equation shows a slightly improved behaviour, see table 4.11. The same goodness of integral results is shown in figure 4.17 (the total power one is shown as well figure 4.18). Again, case C returns the best result. A possible explanation might be: the POD bases minimize the average error between the snapshots and their orthogonal projection onto the bases (briefly summarizing what stated in eq. (2.7)), the lift functions do not belong to this space, therefore their presence shifts the approximated solution from this minimum, still they have a healing effect on the stability of the reduced system. Some more details are given in the conclusions of the work.

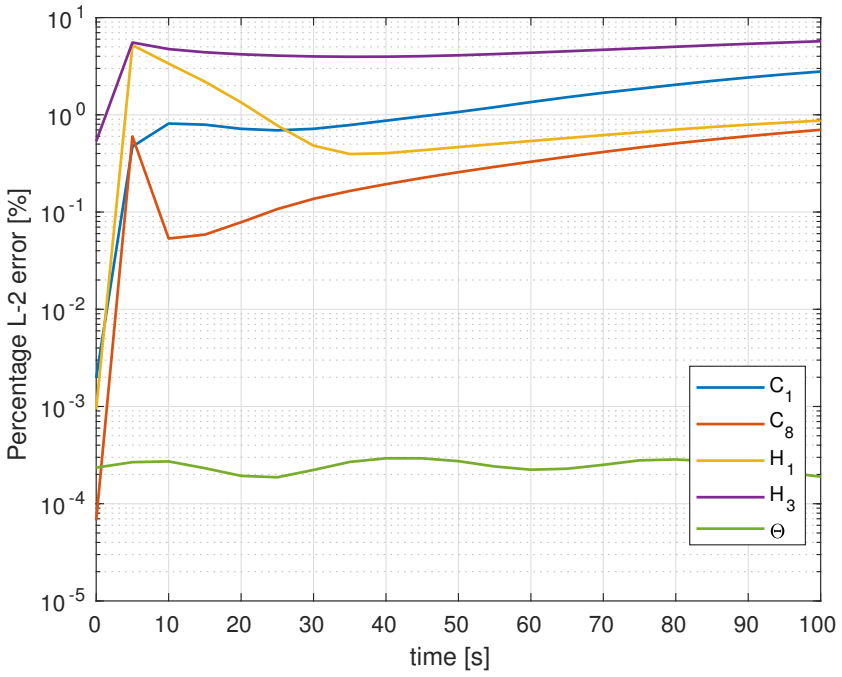


Figure 4.16: Percentage L^2 errors case C - 2.

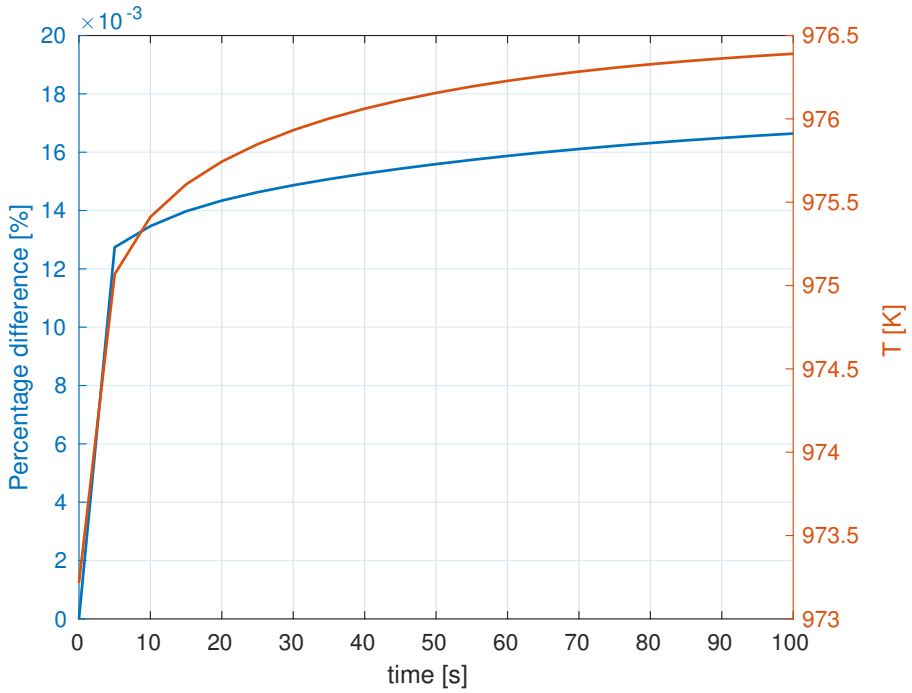


Figure 4.17: Percentage difference between FOM and ROM average temperature, computed as $(T_m - T_{m,r})/T_m \cdot 100$ - case C.

As regards the dynamics of the case, the temperature shows a stiff increase at the beginning of the transient because of the initial high power level, then the profile becomes more flat as the power drops, at the same time an homogenization of the

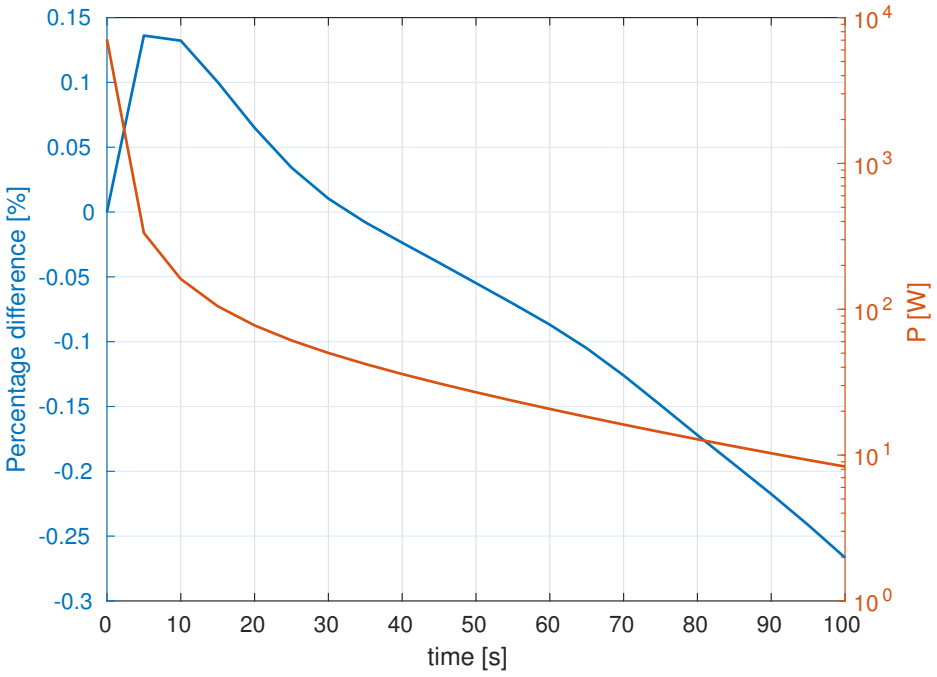


Figure 4.18: Percentage difference between FOM and ROM total power, computed as $(P - P_r)/P \cdot 100$ - case C.

field happens because of diffusion phenomena, leading to an approximately constant value throughout the cavity (figures 4.17 and 6.28). Instead, the power dynamics is more or less the same of cases A and B (figure 4.18).

The behaviour of T_m and P for different values of $\boldsymbol{\mu}$ is also shown (figures 4.19 and 4.20): in this case the values of $\boldsymbol{\mu}$ for which the minimum and maximum difference are found are different in respect to cases A and B (table 4.12); the band is even more narrow than case B, being practically zero in the case of the average temperature. In table 4.13 the number of modes adopted are listed.

$e_F(t)$ [%]	A	B	C
$e_{\boldsymbol{u}}(100s)$	6.93	6.70	6.00
$e_{\phi}(100s)$	0.77	0.76	0.70
$e_W(100s)$	0.93	0.92	0.80
$e_T(100s)$	1.65	4.78	0.79
$e_{H_1}(100s)$	1.02	1.00	0.87
$e_{H_3}(100s)$	6.50	6.42	5.71

Table 4.11: Comparison between case A, B and C $e(100s)$ results for different fields.

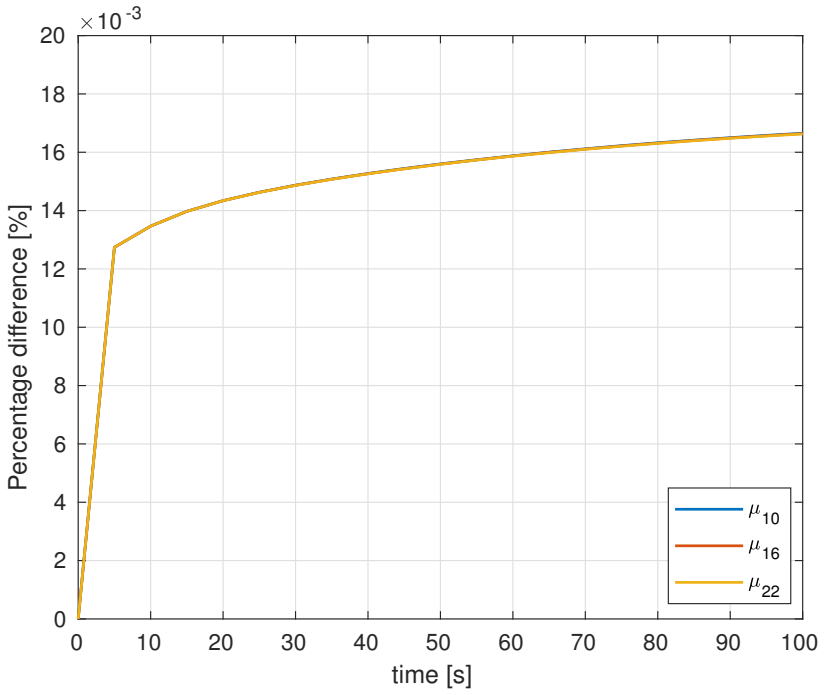


Figure 4.19: Average temperature percentage difference for different values of $\boldsymbol{\mu}$ - case A.

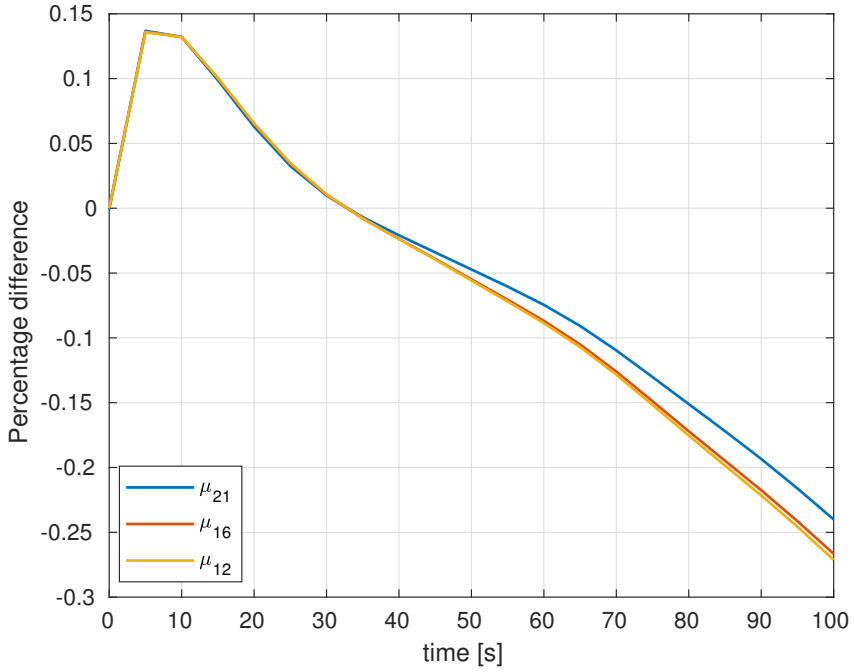


Figure 4.20: Total power percentage difference for different values of $\boldsymbol{\mu}$ - case C.

$\boldsymbol{\mu}$	$\nu\ [m^2/s]$	$\beta_{tot}[-]$	$\lambda_{h,3}\ [1/s]$
$\boldsymbol{\mu}_{10}$	$2.243 \cdot 10^{-6}$	$320.07 \cdot 10^{-5}$	$3.910 \cdot 10^{-4}$
$\boldsymbol{\mu}_{12}$	$2.396 \cdot 10^{-6}$	$328.22 \cdot 10^{-5}$	$3.271 \cdot 10^{-4}$
$\boldsymbol{\mu}_{16}$	$2.460 \cdot 10^{-6}$	$321.80 \cdot 10^{-5}$	$3.580 \cdot 10^{-4}$
$\boldsymbol{\mu}_{21}$	$2.698 \cdot 10^{-6}$	$348.44 \cdot 10^{-5}$	$3.622 \cdot 10^{-4}$
$\boldsymbol{\mu}_{22}$	$2.586 \cdot 10^{-6}$	$341.60.92 \cdot 10^{-5}$	$3.308 \cdot 10^{-4}$

Table 4.12: Values of $\boldsymbol{\mu}$ represented in figure 4.19 and 4.20.

Field	# of modes adopted for projection/reconstruction
$\boldsymbol{u}$	10+1
p	5
ϕ	8
C_1	18
C_2	15
C_3	15
C_4	25
C_5	25
C_6	13
C_7	20
C_8	35
T	27+0
H_1	10
H_2	10
H_3	13
System (2.2) fields	5

Table 4.13: Number of modes adopted both for projection and reconstruction stage, $\boldsymbol{u}$ and T adds the number of lift-function(s) - case C.

4.4 Sensitivity Analysis results

In this section the SA results are presented and commented. First of all, the marginal distribution of the parameters are recalled (all of them are Normal distributions):

Parameter	Central Value x_0	Standard Deviation σ
$\nu\ [m^2/s]$	$2.46 \cdot 10^{-6}$	$0.08 \cdot 10^{-6}$
$\beta_{tot}\ [-]$	$321.8 \cdot 10^{-5}$	$10.7 \cdot 10^{-5}$
$\lambda_{h,3}\ [1/s]$	$3.58 \cdot 10^{-4}$	$0.12 \cdot 10^{-4}$

Table 4.14: Parameters marginal distributions settings.

The number of sampling points is set to 1000, and they are sampled using Monte Carlo Inverse Transform sampling method. It must be recalled that values sampled outside the Traning Range, i.e table 4.5, are rejected. The figures of merit considered are the average temperature T_m and the total power P at $t = 100s$. First table shows their mean value and variance both for ROM and FOM, (inside the brackets):

Case	$T_{m0}\ [K]$	$\sigma_{T_m}\ [K]$	$P_0\ [W]$	$\sigma_P\ [W]$
A	1029.80 (1029.33)	0.39 (0.40)	8.186 (8.165)	0.013 (0.011)
B	1016.21 (1014.67)	0.21 (0.22)	8.231 (8.209)	0.014 (0.013)
C	976.23 (976.39)	$5 \cdot 10^{-4}$ ($2 \cdot 10^{-7}$)	8.366 (8.345)	0.043 (0.0001)

Table 4.15: Expected value and standard deviation for $T_m(t = 100s)$ and $P(t = 100s)$, FOM results inside the brackets.

The difference in the mean value actually accounts for error committed in the approximations of the fields at ROM level, in fact the percentage difference is of the same order of the ones presented in the previous three subsection (see figures 4.5, 4.11, 4.17 for the temperature field, and 4.6, 4.12, 4.18 for the power). Then, both for ROM and FOM the R^2 values are computed, as next table shows:

Case	$R^2_{T_m}$ [%]	R^2_P [%]
A	99.991 (99.983)	99.992 (99.997)
B	99.980 (99.983)	99.992 (99.996)
C	99.974 (99.985)	99.995 (99.997)

Table 4.16: Quality indexes R^2 for $T_m(t = 100s)$ and $P(t = 100s)$, FOM results inside the brackets.

The extremely high proximity to unity asses that for both the figures of merit analysed, the model is very well described by a linear approximation. Thus, the computation of the SRCs values is allowed as indexes of the importance of each of the

three parameters on the output. Next tables summarizes the results for T_m and P:

Case	ROM	FOM
A	$(-0.99996, 0.00017, 0.00012)$	$(-0.99989, 0.00097, -0.00020)$
B	$(-0.99992, 0.00049, -0.00012)$	$(-0.99992, -0.00006, 0.00043)$
C	$(-0.99837, -0.00159, 0.04506)$	$(-0.97794, -0.00206, 0.12577)$

Table 4.17: ROM and FOM average temperature T_m SRCs values, ordered as $(\alpha_\nu, \alpha_{\beta_tot}, \alpha_{\lambda_{h,3}})$.

Case	ROM	FOM
A	$(-0.99292, 0.00371, 0.14667)$	$(-0.97764, 0.00229, 0.18927)$
B	$(-0.98786, 0.00431, 0.14768)$	$(-0.98663, 0.00175, 0.17376)$
C	$(-0.98561, 0.00136, 0.15841)$	$(-0.96963, 0.00162, 0.16169)$

Table 4.18: ROM and FOM total power P SRCs values, ordered as $(\alpha_\nu, \alpha_{\beta_tot}, \alpha_{\lambda_{h,3}})$.

Let us focus first on the result of P's SA: in absolute value, the most important parameter is the kinematic viscosity, always followed by the decay constant $\lambda_{h,3}$ and then the β_{tot}. The kinematic viscosity influences the velocity field that determines all the fields of system eq. (1.1), both directly and indirectly, this could determine the dominance of this parameter in the output results. The negative sign states that a value larger than ν_0 reduces the power. The decay constant may follow because it is directly involved in the expression of the power density, thus of the total power, differently for the total fraction of delayed neutrons. No substantial difference emerges among the three different case and, most of all, the ROM is capable to catch this influence of the parameters on the output, even if the values are not exactly the same.

Concerning the average temperature T_m, also at SA stage some considerable discrepancies emerges between FOM and ROM results in cases A and B. If in case A, still persists an order from the lower to the higher SRC (but with different signs between FOM and ROM), case B is completely wrong both in signs and values of parameters β_{tot} and $\lambda_{h,3}$. Case C instead catches correctly the influence of the parameters on the average temperature, even if the value are quite different, especially for the decay constant of third decay heat group.

It must be said that 1000 runs might be still low as a number to correctly describe the statistics of the result, i.e. asymptotic condition of the Central Limit Theorem might not be reached yet. In other words, the sample average and its standard deviation, both for the parameters and the figure of merits, can still oscillate widely. Considering all the six cases analyzed, it is possible to compute the average and the standard deviation of the standard deviations computed in each case, or in formulae,

$\boldsymbol{E}\big[\sigma[\boldsymbol{\mu}]\big]$ and $\boldsymbol{\sigma}\big[\sigma[\boldsymbol{\mu}]\big]$. Their ratio in percentage terms is then:

$$\frac{\boldsymbol{\sigma}\big[\sigma[\boldsymbol{\mu}]\big]}{\boldsymbol{E}\big[\sigma[\boldsymbol{\mu}]\big]} \cdot 100 = (3.3, 1.3, 2.6)\% \tag{4.3}$$

Where $\boldsymbol{\mu}$ is ordered as usual, $\{\nu, \beta_{tot}, \lambda_{h,3}\}$, and thus their corresponding percentage ratios. Such values could justify the differences between FOM and ROM SRCs results for the temperature, since the values of $\alpha_{\beta_{tot}}$ and $\alpha_{\lambda_{h,3}}$ are very close to zero, and accordingly to the sample average and standard deviation, they could jump from negative to positive values. Of course, an higher number of runs must be performed in order to verify what just said.

Finally the result that justify all the effort in developing a ROM for MSFR is presented, i.e. the time needed to perform the SA with ROM and FOM:

Case	$t_{Offline}\ [s]$	$t_{SA,ROM}\ [s]$	$t_{SA,FOM}\ [s]$
A	$0.049 \cdot 10^6$	$0.020 \cdot 10^6$	$1.310 \cdot 10^6$
B	$0.054 \cdot 10^6$	$0.026 \cdot 10^6$	$1.300 \cdot 10^6$
C	$0.061 \cdot 10^6$	$0.037 \cdot 10^6$	$1.302 \cdot 10^6$

Table 4.19: Elapsed time for Offline stage, SA with ROM, SA with FOM,respectively for case A, B, and C.

The result assesses the huge speed up obtained using the ROM, the time needed to perform 1000 runs of 100s each with ROM is smaller than the time needed to train it, that is 31 runs of 150s. Even considering the time needed to train the reduced object, the ratios between SA for the full and reduced model account for (projection stage and RBF interpolation times are neglected because their order of magnitude is about of hundreds seconds):

$$\left[\frac{t_{Offline} + t_{SA,ROM}}{t_{SA,FOM}}\right]_A = 5.3\%$$
$$\left[\frac{t_{Offline} + t_{SA,ROM}}{t_{SA,FOM}}\right]_B = 6.2\%$$
$$\left[\frac{t_{Offline} + t_{SA,ROM}}{t_{SA,FOM}}\right]_C = 7.5\%$$

In practice, the elapsed time reduces from about two weeks to one day, more or less for all the cases. Another interesting quantity to show is time needed to perform one time step:

Case	Elapsed time per time step-ROM [s]	Elapsed time per time step-FOM [s]
A	0.020	1.310
B	0.026	1.300
C	0.037	1.302

Table 4.20: Time needed to solve one time step of the simulation during SA.

In practice the ROM is capable to compute one time step, 0.1 s, from 20 to 37% of its value in real elapsed time, while the FOM needs approximately 1300%.

4.5 Outside Training Range - Case A'

In this section, case A model is tested with $\boldsymbol{\mu} = 1.2\boldsymbol{\mu}_0$, that is outside the domain of the parameters adopted to train the ROM. In addition during offline stage, the parameters are not ordered so that for the maximum value of one parameter corresponds the maximum of the others, like in this case. Moreover, the final time at online stage is set equal to the offline final time. Therefore, really challenging settings the ROM is required to reproduce. The initial and boundary conditions are the same of case A, see table 4.3. The same comparisons of other sections are shown. The initial condition is also found in this case adopting $\boldsymbol{\mu}_0$ correspondent offline solution. The velocity L^2 percentage error shows a very high initial value, that then decreases and reaches values comparable to cases A, B and C.

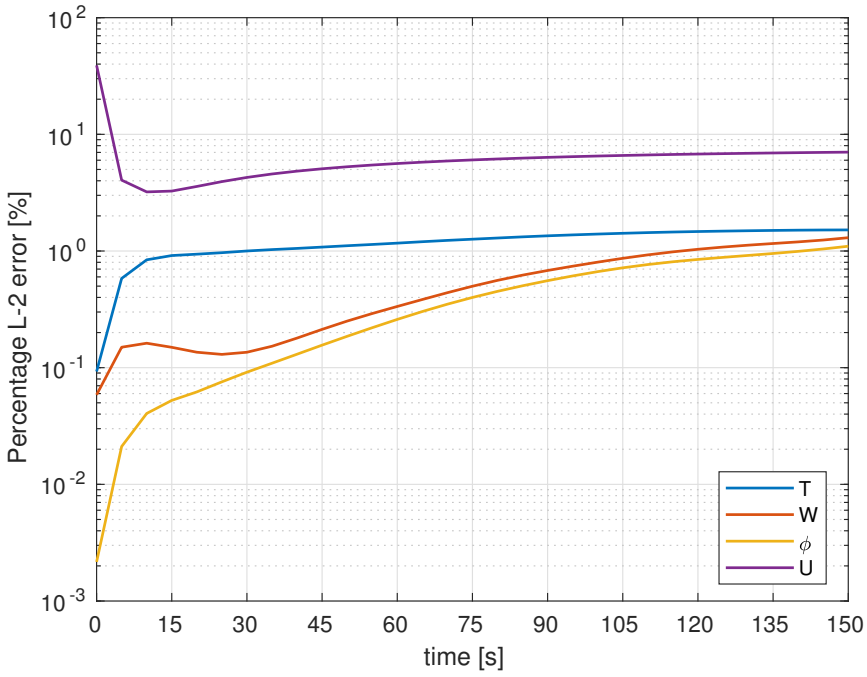


Figure 4.21: Percentage L^2 errors case A' - 1.

$e_F(t)$ [%]	A	B	C	A'	$e_F(t)$ [%]	A'
$e_{\boldsymbol{u}}(100s)$	6.93	6.70	6.00	6.52	$e_{\boldsymbol{u}}(150s)$	7.04
$e_{\phi}(100s)$	0.77	0.76	0.70	0.66	$e_{\phi}(150s)$	1.10
$e_{W}(100s)$	0.93	0.92	0.80	0.80	$e_{W}(150s)$	1.30
$e_{T}(100s)$	1.65	4.78	0.79	1.40	$e_{T}(150s)$	1.52
$e_{H_1}(100s)$	1.02	1.00	0.87	0.90	$e_{H_1}(150s)$	1.19
$e_{H_3}(100s)$	6.50	6.42	5.71	6.31	$e_{H_3}(150s)$	7.80

Table 4.21: Comparison between case A, B, C and A' $e(100s)$ results,$e(150s)$ A' results.

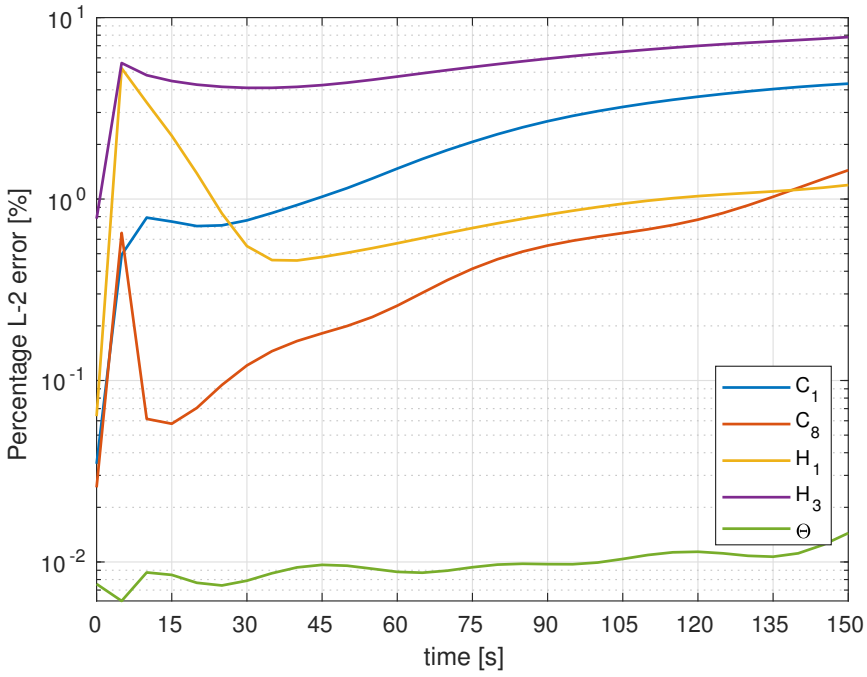


Figure 4.22: Percentage L^2 errors case A' - 2.

What is really interesting is that, in contrast to what expected, the ROM approximation is better in case A' than in case A at 100 s. Moreover, the errors obtained at 150 s do not show extremely high values, instead they are of the same order of magnitude, for some field even lower than $e(100)$ in case A (see table 4.21). This could suggest that the condition on the online final time may be to restrictive and it can be relaxed in this case.

Some very good results are obtained considering the average temperature T_m and the total power P as well, see figures 4.23 and 4.24 respectively. The consideration on the spatial distribution of temperature field are the same of case A, but the hot and cold spot do not enlarge during time and a stable condition is reached concerning this aspect.

It must be underlined that this case might only represent a lucky case, other values of $\boldsymbol{\mu}$ outside the training range could produce worse results. This test should be also done in the case $\boldsymbol{\mu}$ is not a multiple of $\boldsymbol{\mu}_0$, but constructed as $\boldsymbol{\mu} = \{a\nu_0, b\beta_{tot,0}, c\lambda_{h3,0}\}$ with a, b, c assuming any positive real value outside $[0.9, 1.1]$. Given that, the results shown in this section is quite encouraging and promising in order to use the ROM also outside training range domain.

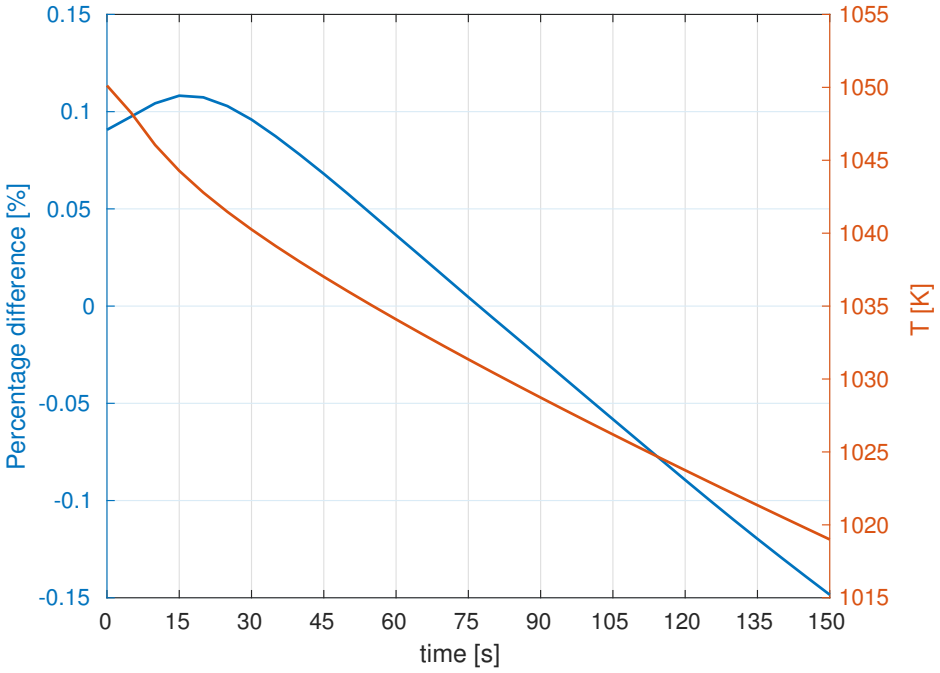


Figure 4.23: Percentage difference between FOM and ROM average temperature, computed as $(T_m - T_{m,r})/T_m \cdot 100$ - case A'.

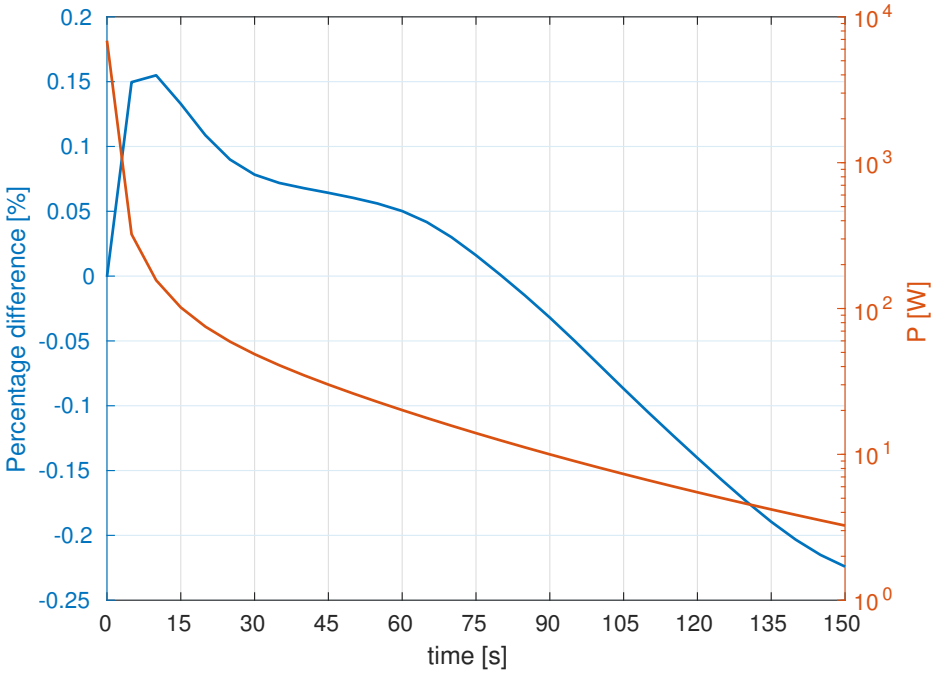


Figure 4.24: Percentage difference between FOM and ROM total power, computed as $(P - P_r)/P \cdot 100$ - case A'.

(This page is intentionally left blank)

Chapter 5

Conclusions

The work consisted in developing a reduced order model C++ software for the multiphysics of the MSFR within FV framework. Some very basic tests were performed to check the goodness of the obtained tool and start understanding the basic feature of this kind of modelling for MSFR.

What emerges is that, in the cases considered, very good local (see Appendix) and integral reconstruction of MSFR fields is returned, for every field but the temperature. This error in the approximation is then propagated to SA, which in turn shows very good result for P. The reasons appear not so clear, thus some hypotheses are outlined.

As highlighted in the introduction, this work analyzes a very low variety of possible configurations of offline and online stage. Indeed, the degree of freedom of this settings is really large, in fact one can vary (using the notation adopted throughout the work): N_{off}, K, N_Fs (i.e. the number of modes adopted per each field), SVD/EVD POD modes calculation, OpenFOAM solvers for the Offline stage. The combination of those leads an extremely high value of possibilities. Of course not all of them have to be considered, but a more specific parametric study, especially for N_{off} and K should be performed. This study must be performed in order to identify the minimum error region for the majority of the fields, considering that the temperature equation includes the highest number of terms of different fields, thus embeds all the errors they are reconstructed with. For example, in this work the error on the velocity looks too high and it can be reduced as shown in many works [5][6][12]. N_Fs parametric study is also important because it is true that increasing it, the amount of original information retained increases (or $\lambda_{N_F} \to 0$ as $N_F \to \infty$), but it changes the number of equations of system (2.14) and thus the convergence of Newton-Raphson method.

As said above, temperature equations contains sources from very different fields, including fields of system 1.2. This leads to an high number of tensors inside the temperature equation at ROM level. Therefore, another strategy to reduce the errors on T could be to define on purpose sources for T and then use RBF interpolation to reconstruct them at online stage (which showed a very good behaviour in all the

three cases), for instance:

$$S_\phi = \frac{\Sigma_P}{k_{eff}\rho}\phi \tag{5.1}$$

So that at projection stage only matrices are left instead of tensors. This could also speed up the online computation.

Another concern is related with the lift-functions. When Dirichlet boundaries are involved, POD modes do not correspond to the minimum of (2.7) for $\mathbb{V}_{POD}^{T}$ but for $\mathbb{V}_{POD}^{T'}$, i.e. for the homogeneous field T'. Therefore another kind of approximation is introduced. This is demonstrated by the fact that the L^2 error without lift-functions is the smallest. But in that case the spatial reconstruction is very bad, confirming the hypothesis that lift-functions stabilize ROM system solution. Therefore their presence seems necessary, the possibility to use other kind of governing equations to find them could be a possible path to reduce L^2 errors. Considering the case of all Neumann boundaries instead, lift-functions also for this case could be computed, similarly to what is done in [8].

As said above, a part from temperature field, all the other fields are reproduced with a good degree of accuracy also to perform some kind of spatial query on the system, like control oriented applications presented in [11].

Concerning SA, linear regression technique is fully applicable since values of R^2 are practically 1, both for T_m and P. SA shows that in the cases considered the most important parameter is the kinematic viscosity, which has a negative effect as its value increases in respect to the nominal one (both on T_m and P). As reported in the dedicated section, the number of simulations performed may not be sufficient yet to assess a value very close to the real one, but, given such consideration, the dominance of the viscosity is evident. Because of the error that temperature field shows, together with the last consideration, no further comments are given on SA results for the average temperature. Concerning instead the total power,FOM and ROM show very similar results, at least as order of magnitude and correct sign of SRCs. The increase in $\lambda_{h,3}$ has a positive effect on the total power. This can be explained simply looking at the definition of the power density in eq. (1.4). The positive effect of β_{tot} was not so evident, but it assess that an increased level of delayed neutrons increases the flux level for the time considered in the analysis, finally increasing the power. This contribution is actually much lower than $\lambda_{h,3}$'s one, accounting for some percents of it. It must be underlined that this results are obtained only in these simple cavity cases and it is too risky to generalize them on the basis of this poor analyses.

Very interesting and promising results are obtained when the ROM is tested outside Training Range region, this suggests that the model could be used to actually test unknown system configurations, which is indeed the real task in developing reduced order model.

Concerning instead the time reduction obtained using ROM, it effectively justifies the effort in reduced order modelling and it confirms once again the interest in

the development of this technique for MSFR as research branch.

Last but not least, some considerations are given about the future extension of MSFR reduced order modelling within ITHACA-FV. In order to relax both simplifications about natural convection and turbulent flows, the best idea would be to take advantage of multiple inheritance naturally adopted in object oriented programming, in particular, at FOM level: it is possible to use already provided classes *UnsteadyBB* and *UnsteadyNSTurb* that deal with buoyancy forces and turbulence respectively and compose the hypothetical following classes:

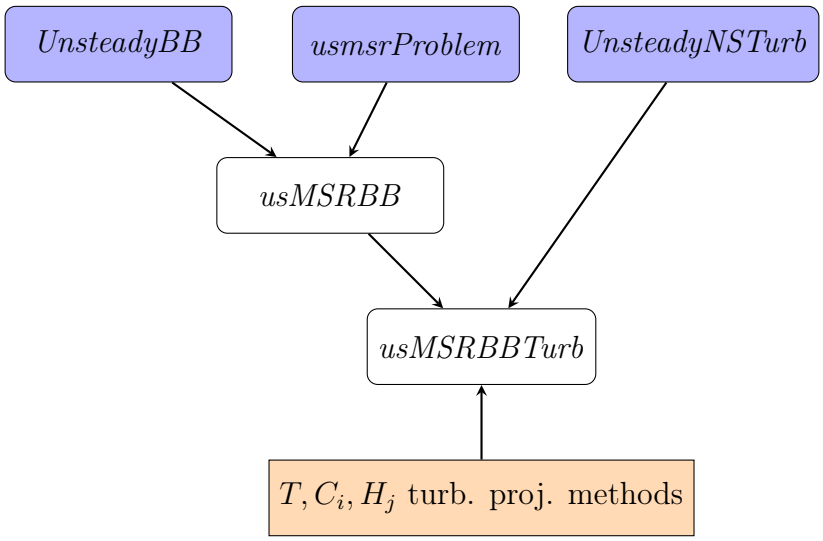


Figure 5.1: Possible new FOM MSFR classes within ITHACA-FV.

Class *UnsteadyBB* implements the projection methods for the modified pressure equation in natural convection, while *UnsteadyNSTurb* implements the projection methods associated to the turbulent viscosity [7]. Therefore the laminar natural convection MSFR class comes practically for free, while the turbulent natural convection MSFR class needs only to add the projection methods associated to the turbulent viscosity terms in T, C_i and H_j equations (see the appendix of [9]). A graphical representation is shown in figure 5.1. One needs only to modify the equations of the class [16]. Moreover, *msrProblem* class already implements precursors boundary conditions adopted in [9], that permits to simulate more realistic geometry. Finally *FofM* abstract class permits to develop whatever figure of merit associated to any FOM problem implemented ITHACA-FV and use it to perform SA adopting linear regression technique presented in section 2.3.

In conclusion, some very good results are already achieved at this stage, along with non negligible errors concerning the temperature field. The fundamental aspect is that the flexibility of the library, with this new tool implemented, enables to study very different system configurations, to deepen the considerations given here with very low programming effort and thus to perform a systematic study about the use of the POD-Galerkin reduced order modelling technique for the MSFR.

(This page is intentionally left blank)

Chapter 6

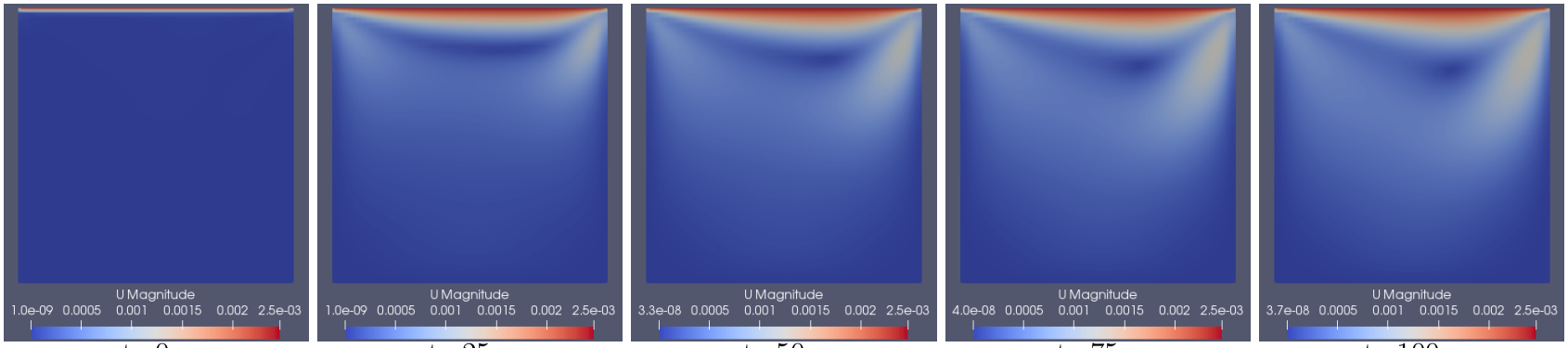
Appendix: Fields for different test case

In these sections, the graphical comparison between FOM and ROM result for different fields at some specified time is shown. Because of not so relevant differences among the cases, some are shown only for case A. The solutions are computed for the central value of $\boldsymbol{\mu}$, that is:

- $\nu = 2.46 \cdot 10^{-6}$ $[m^2/s]$
- $\beta_{tot} = 321.8 \cdot 10^{-5}$ $[-]$
- $\lambda_{h,3} = 3.58 \cdot 10^{-4}$ $[1/s]$

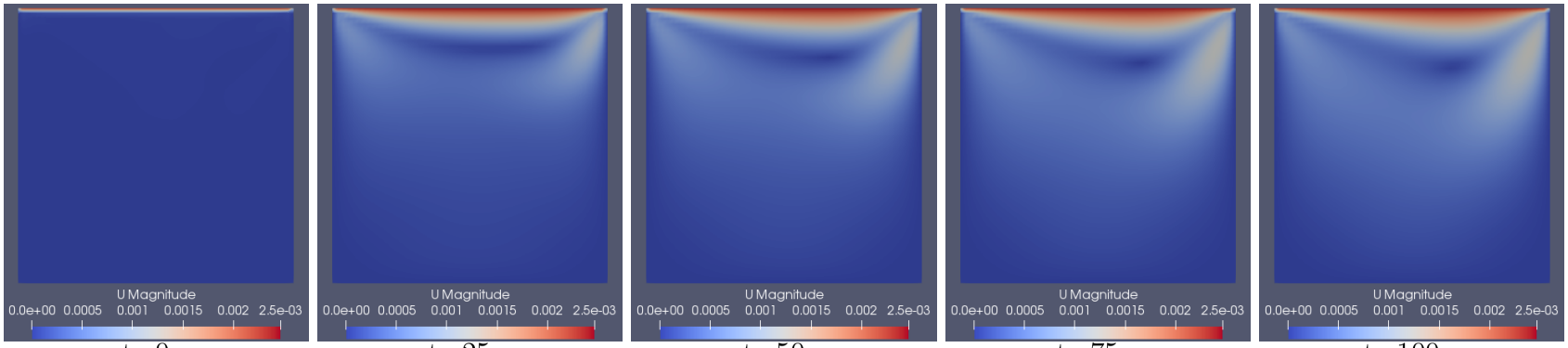
6.1 Case A

The fields represented are: $\boldsymbol{u}, \phi, C_1, T, H_3, \Gamma, W$.



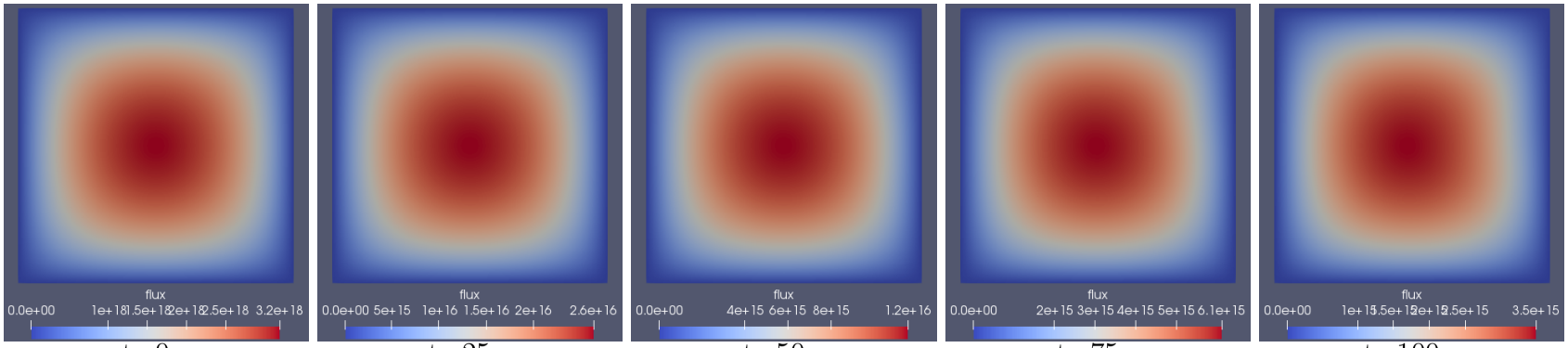
t=0 s t=25 s t=50 s t=75 s t=100 s

Figure 6.1: FOM velocity field $\boldsymbol{u}$ $[m/s]$ at different time instants - case A.



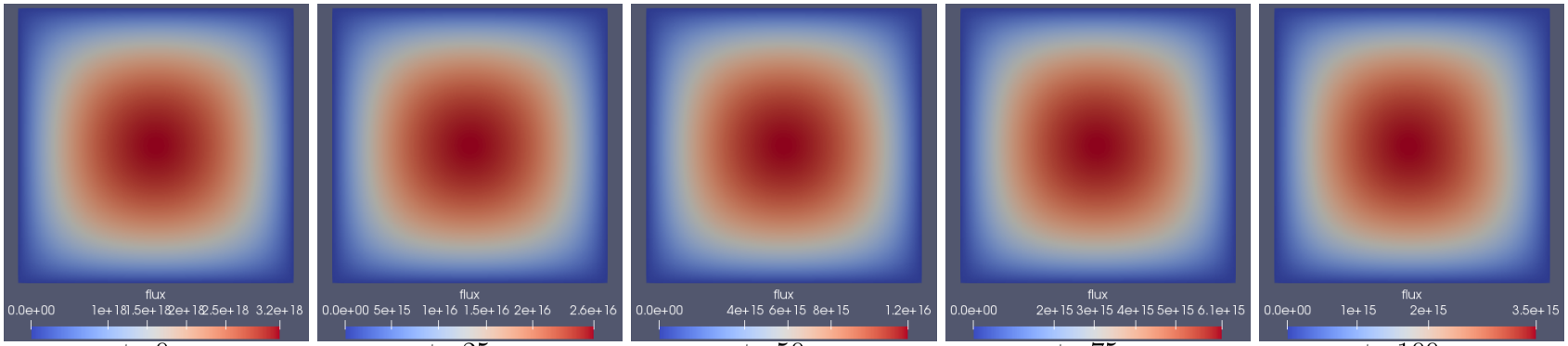
t=0 s t=25 s t=50 s t=75 s t=100 s

Figure 6.2: ROM velocity field $\boldsymbol{u}_r$ $[m/s]$ at different time instants - case A.



t=0 s t=25 s t=50 s t=75 s t=100 s

Figure 6.3: FOM neutronic flux ϕ $[1/(m^2 s)]$ at different time instants - case A.



t=0 s t=25 s t=50 s t=75 s t=100 s

Figure 6.4: ROM neutronic flux ϕ_r $[1/(m^2 s)]$ at different time instants - case A.

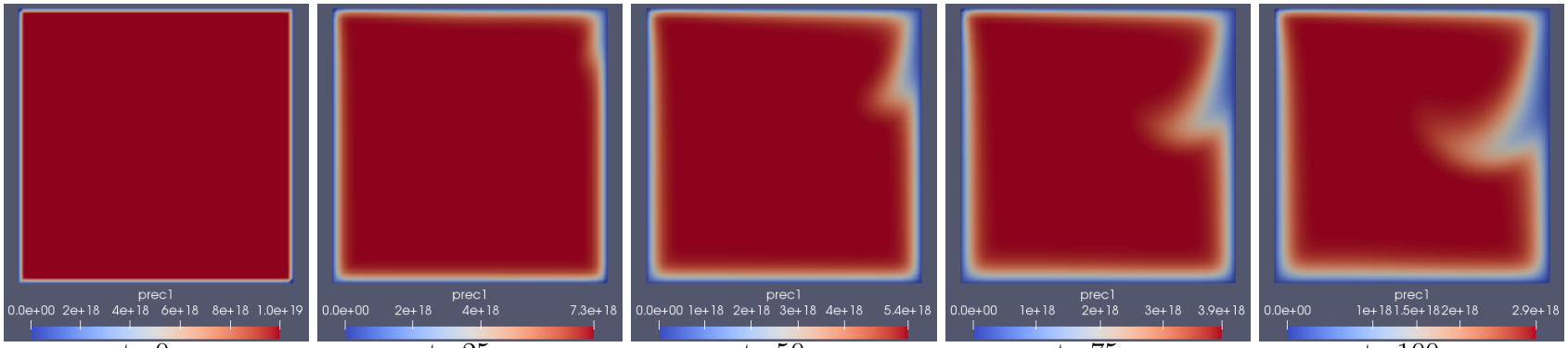


Figure 6.5: FOM first precursor group C_1 $[1/m^3]$ at different time instants - case A.

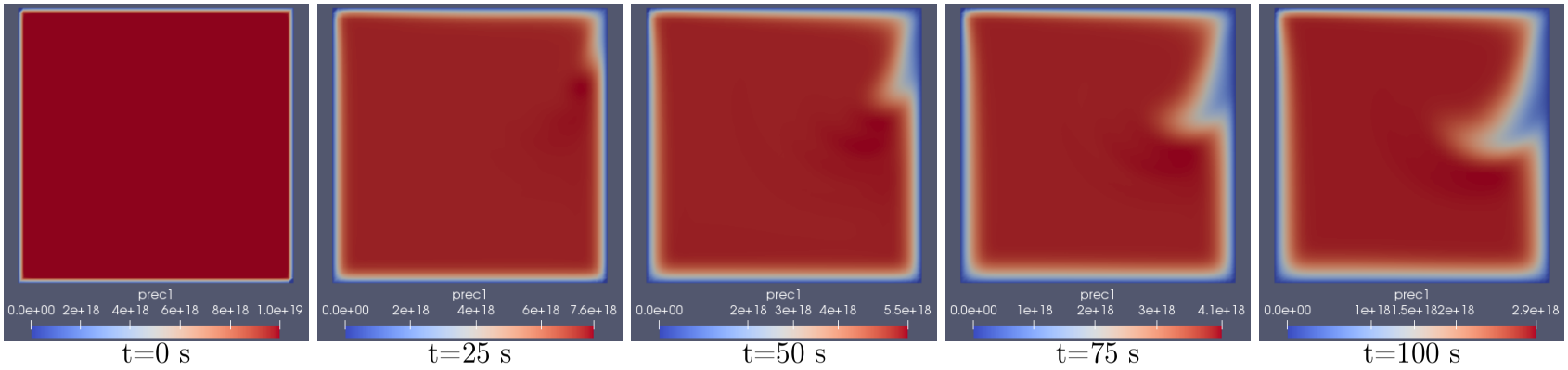
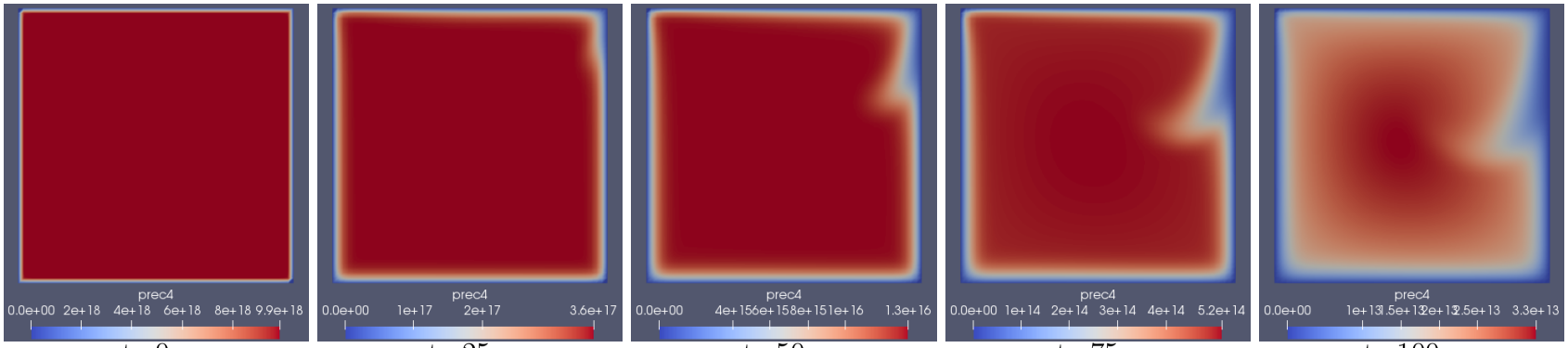
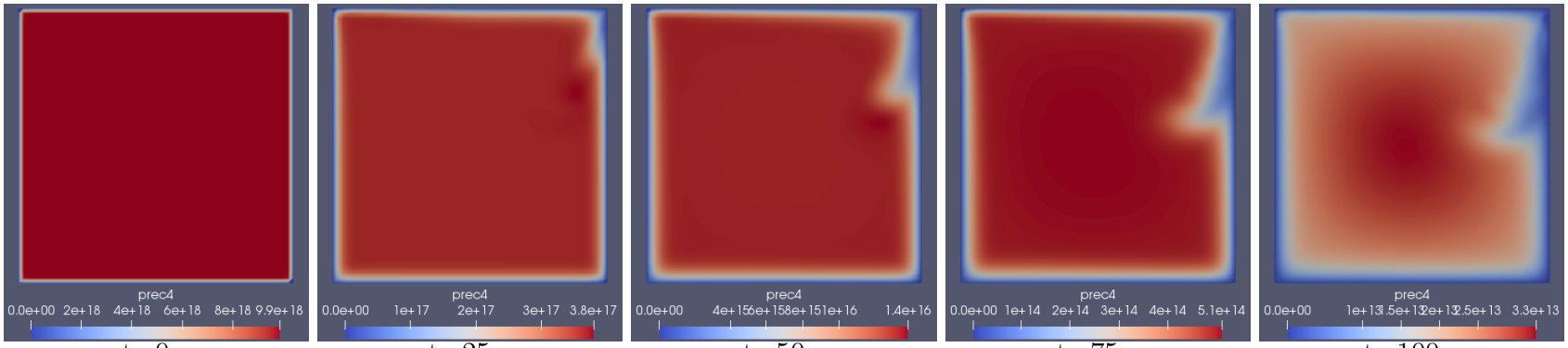


Figure 6.6: ROM first precursor group $C_{1,r}$ $[1/m^3]$ at different time instants - case A.



t=0 s t=25 s t=50 s t=75 s t=100 s

Figure 6.7: FOM fourth precursor group C_4 $[1/m^3]$ at different time instants - case A.



t=0 s t=25 s t=50 s t=75 s t=100 s

Figure 6.8: ROM fourth precursor group $C_{4,r}$ $[1/m^3]$ at different time instants - case A.

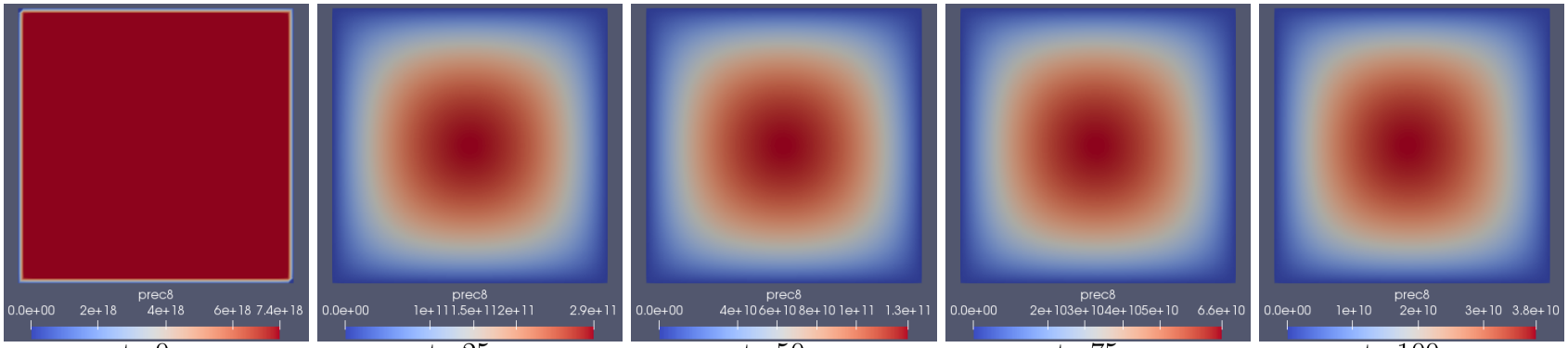


Figure 6.9: FOM eighth precursor group C_8 $[1/m^3]$ at different time instants - case A.

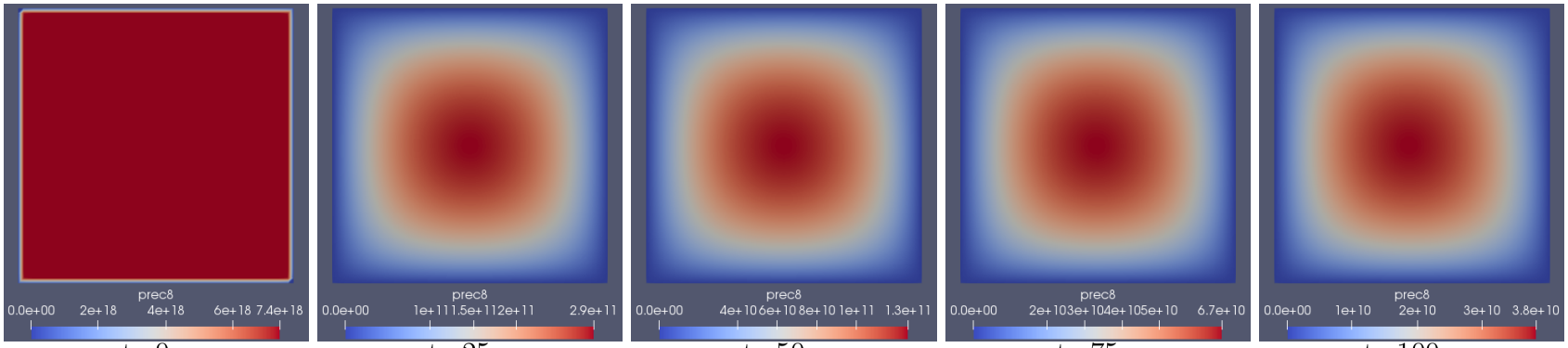


Figure 6.10: ROM eighth precursor group $C_{8,r}$ $[1/m^3]$ at different time instants - case A.

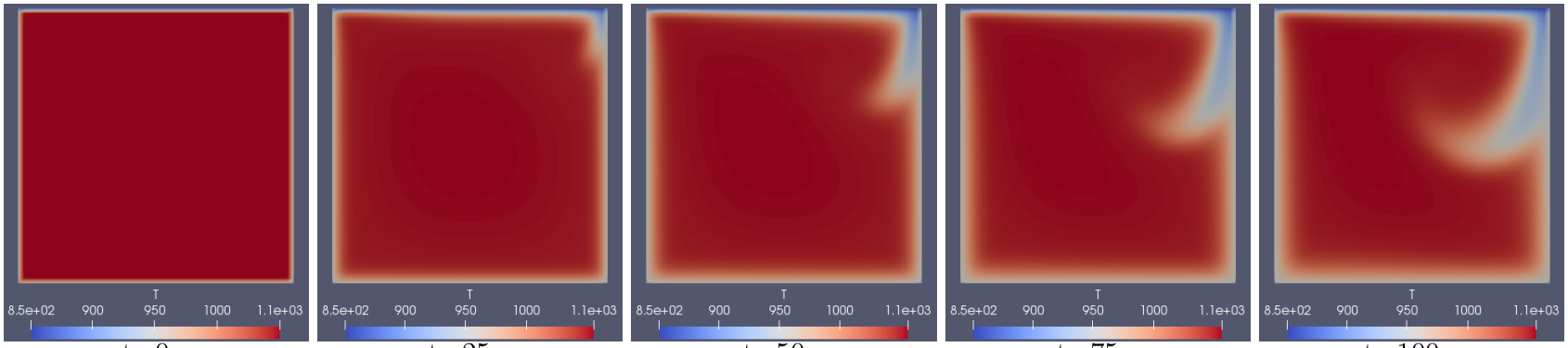


Figure 6.11: FOM temperature field T $[K]$ at different time instants - case A.

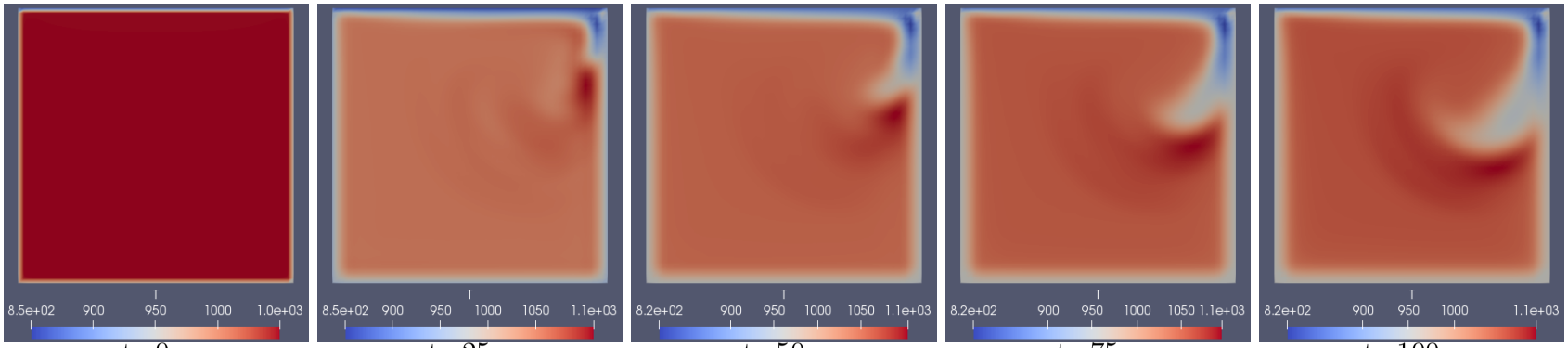


Figure 6.12: ROM temperature field T_r $[K]$ at different time instants - case A.

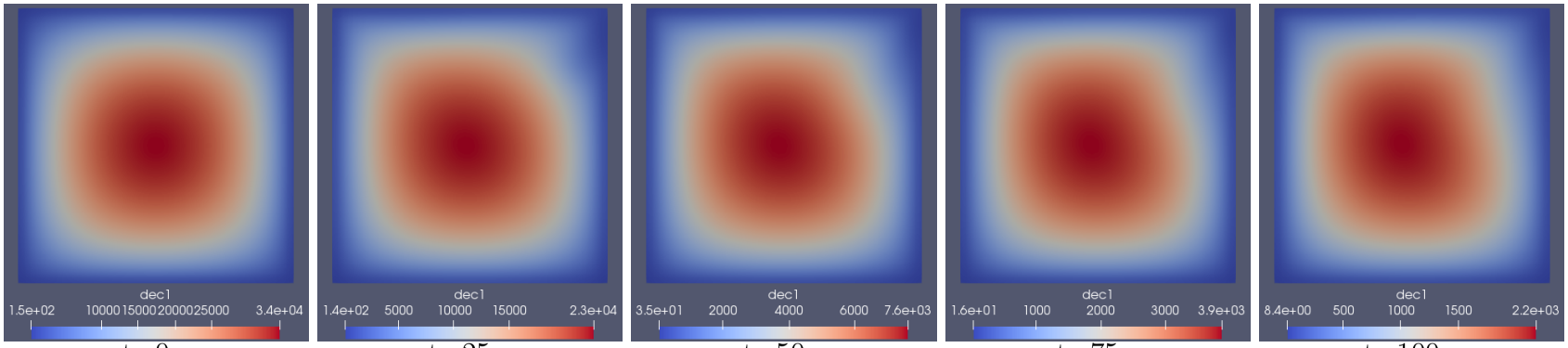


Figure 6.13: FOM first decay heat group H_1 $[J/m^3]]$ at different time instants - case A.

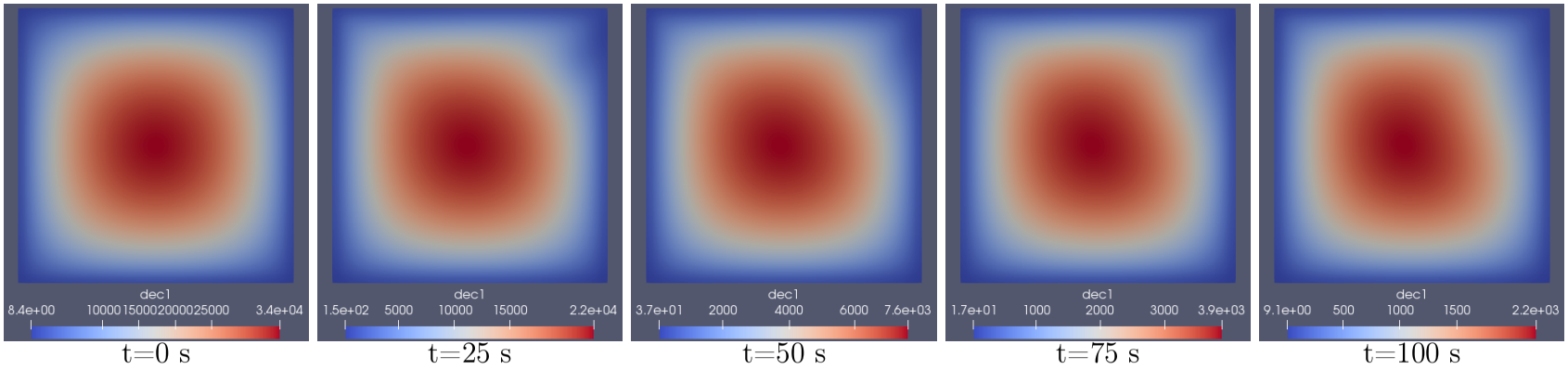


Figure 6.14: ROM first decay heat group $H_{1,r}$ $[J/m^3]$ at different time instants - case A.

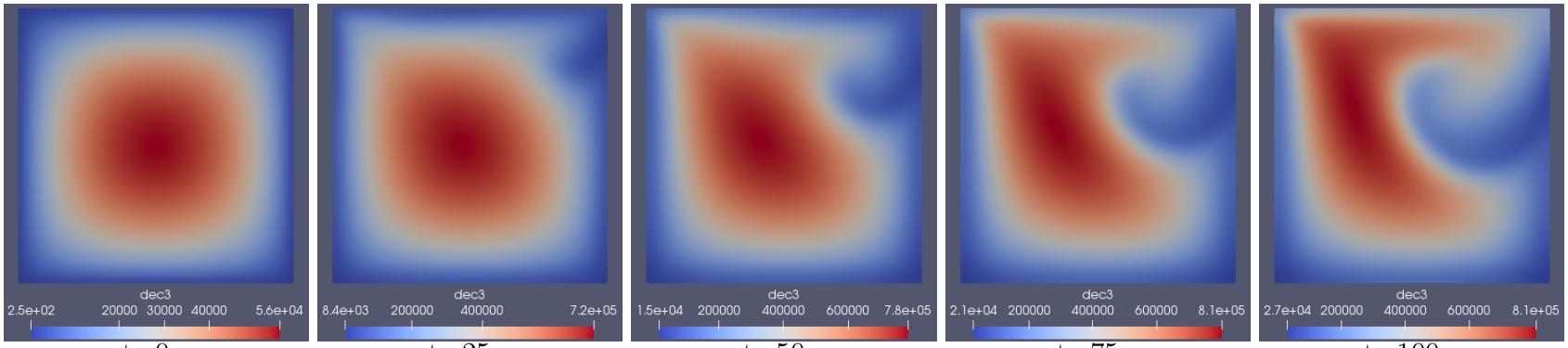


Figure 6.15: FOM third decay heat group H_3 $[J/m^3]$] at different time instants - case A.

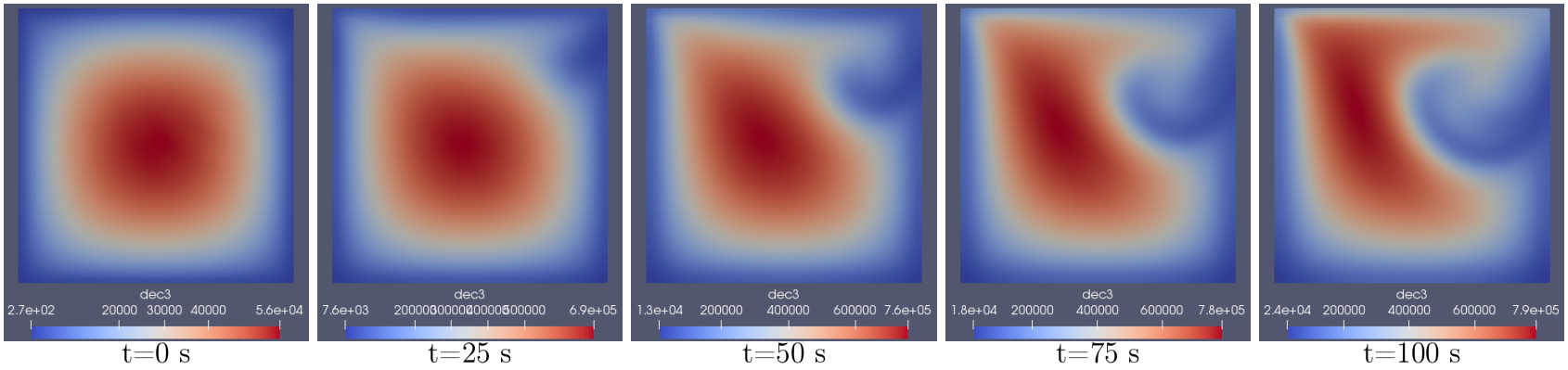
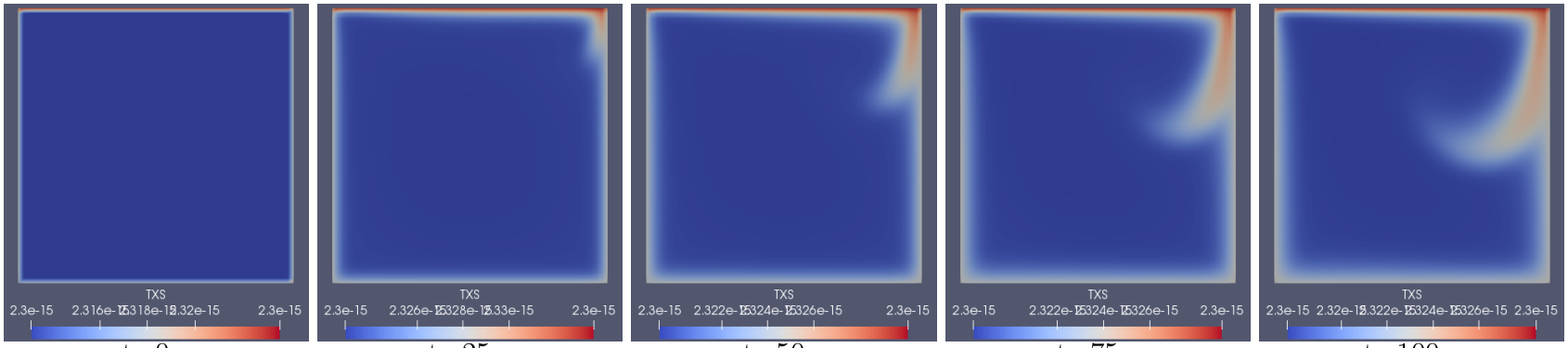
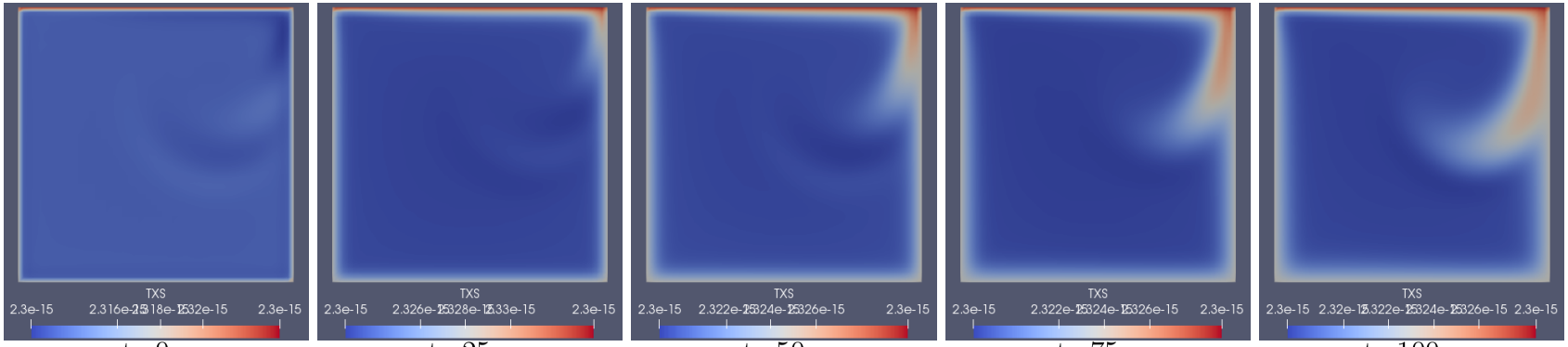


Figure 6.16: ROM third decay heat group $H_{3,r}$ $[J/m^3]$ at different time instants - case A.



t=0 s t=25 s t=50 s t=75 s t=100 s

Figure 6.17: FOM Γ $[J/m]$ cross section at different time instants - case A.



t=0 s t=25 s t=50 s t=75 s t=100 s

Figure 6.18: ROM Γ_r $[J/m]$ cross section at different time instants - case A.

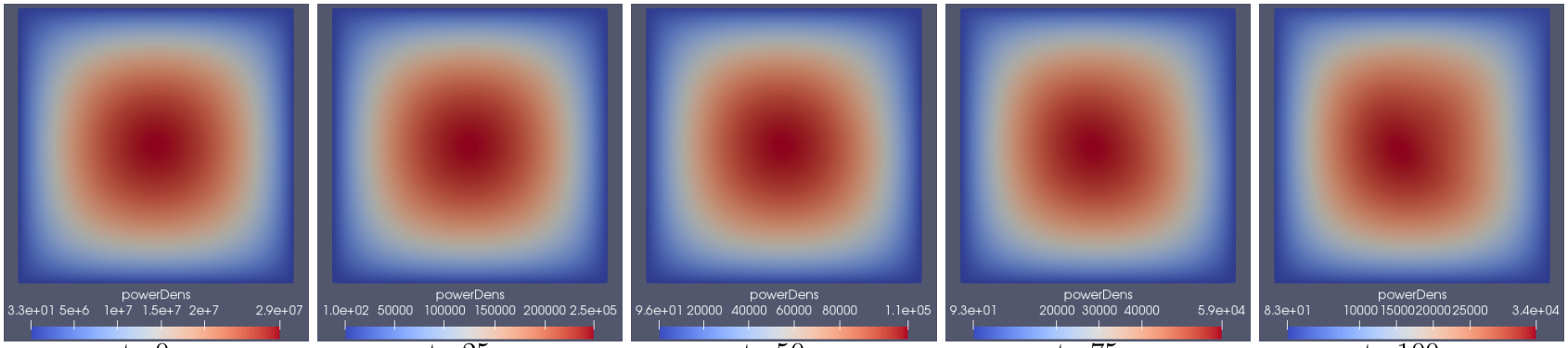


Figure 6.19: FOM power density W $[W/m^3]$ at different time instants - case A.

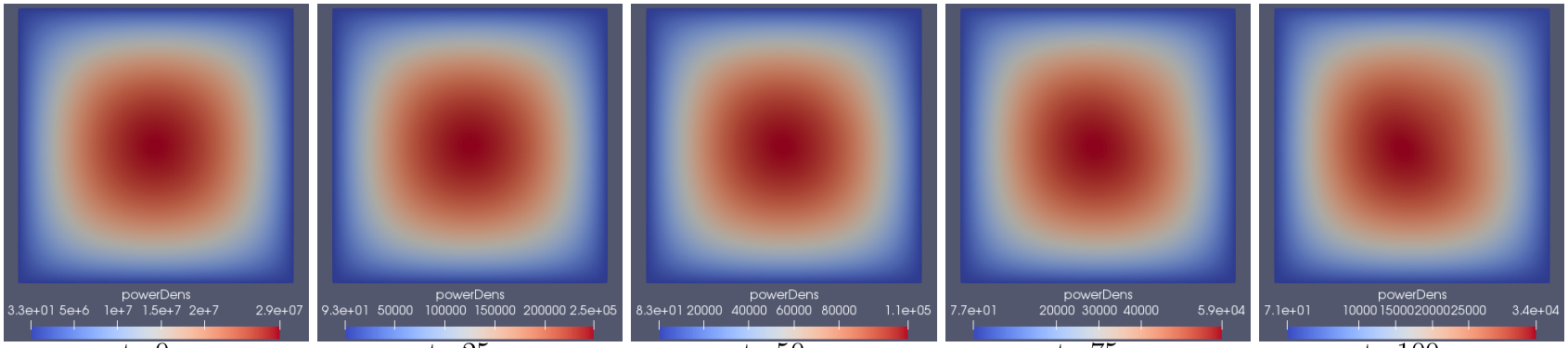
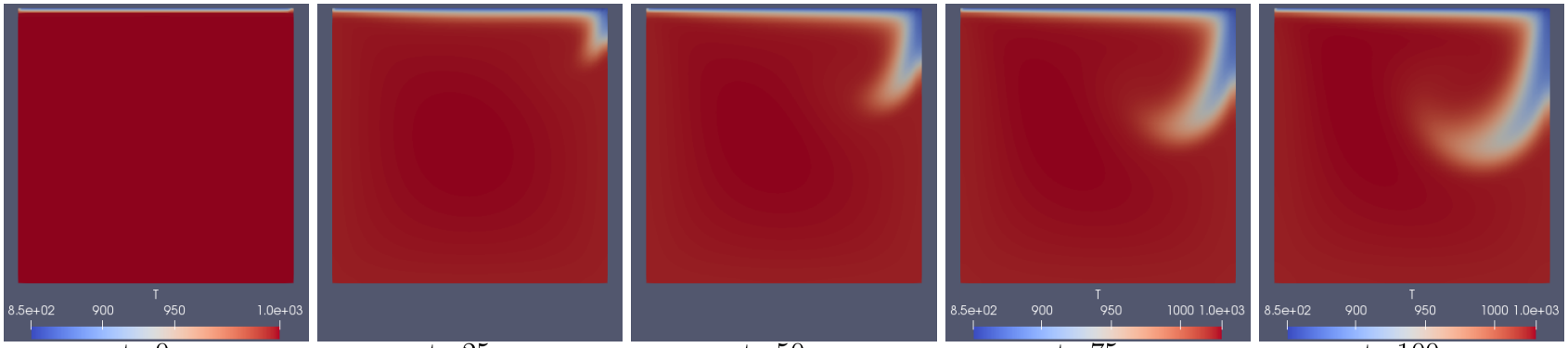


Figure 6.20: ROM power density W_r $[W/m^3]$ at different time instants - case A.

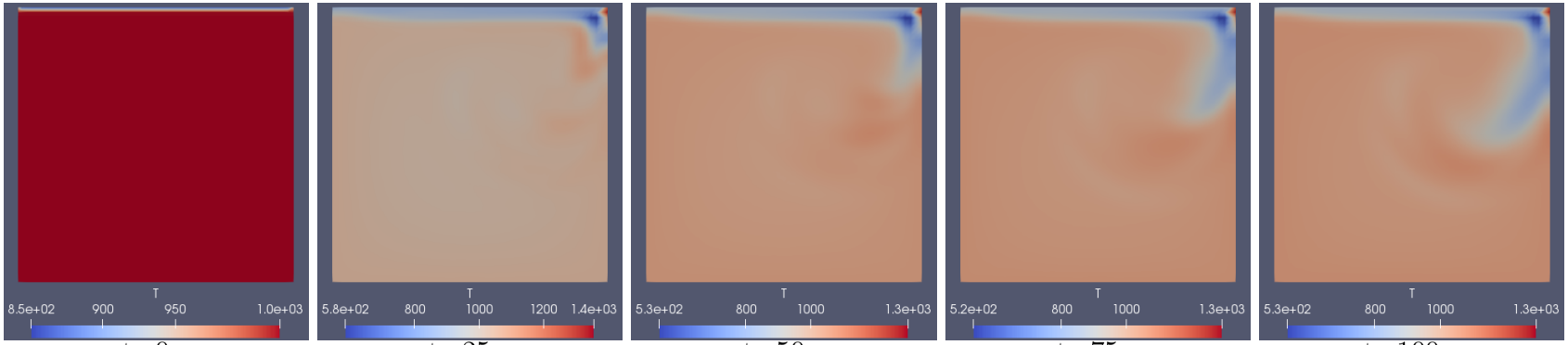
6.2 Case B

The fields represented are: T, H_3, W.



t=0 s t=25 s t=50 s t=75 s t=100 s

Figure 6.21: FOM temperature field T $[K]$ at different time instants - case B.



t=0 s t=25 s t=50 s t=75 s t=100 s

Figure 6.22: ROM temperature field T_r $[K]$ at different time instants - case B.

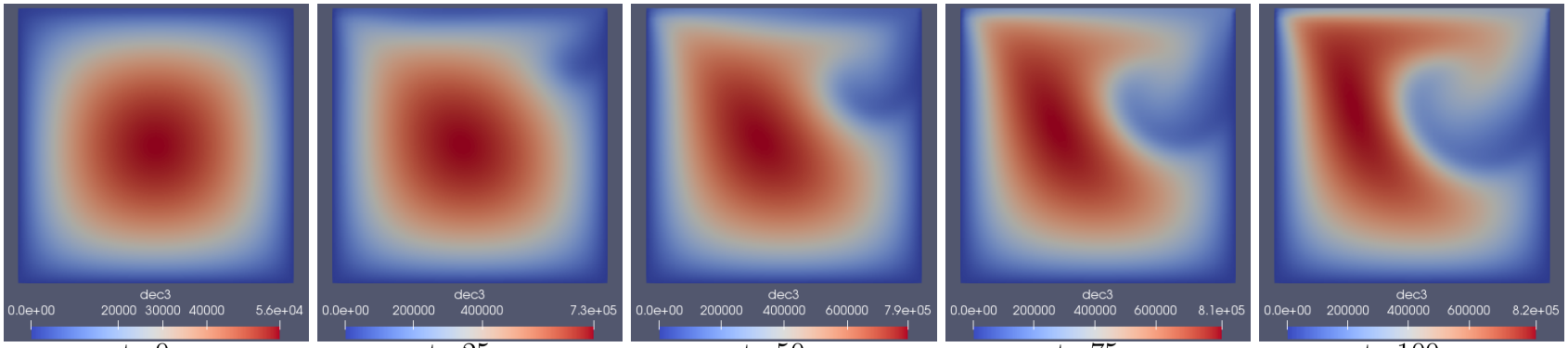


Figure 6.23: FOM third decay heat group H_3 $[J/m^3]$] at different time instants - case B.

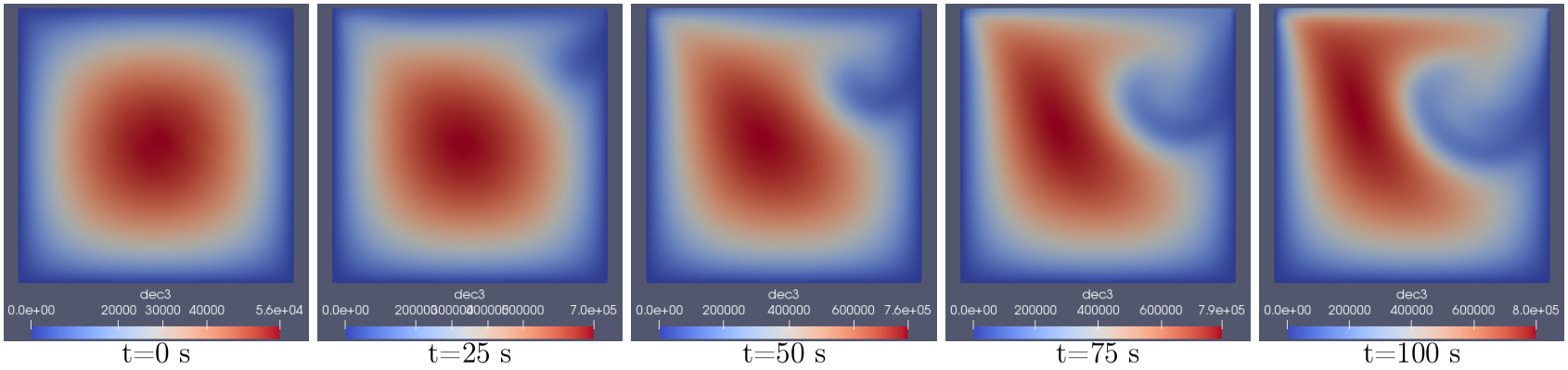


Figure 6.24: ROM third decay heat group $H_{3,r}$ $[J/m^3]$ at different time instants - case B.

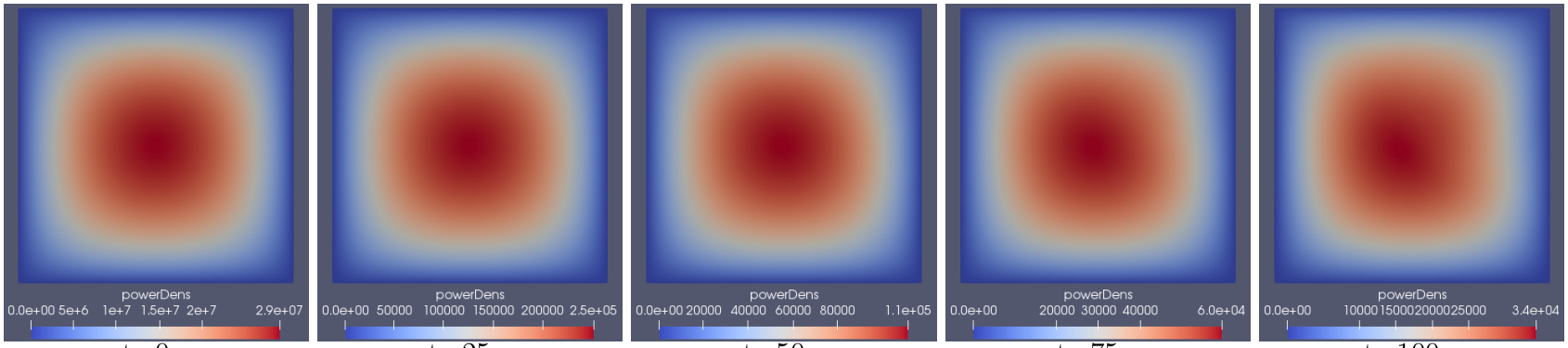


Figure 6.25: FOM power density W $[W/m^3]$ at different time instants - case B.

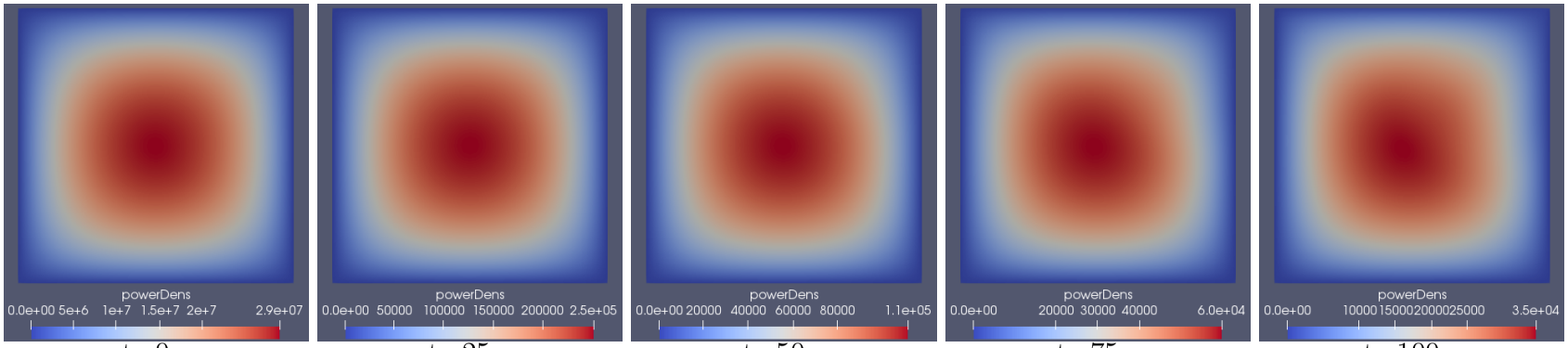


Figure 6.26: ROM power density W_r $[W/m^3]$ at different time instants - case B.

6.3 Case C

The fields represented are: T, H_3, W.

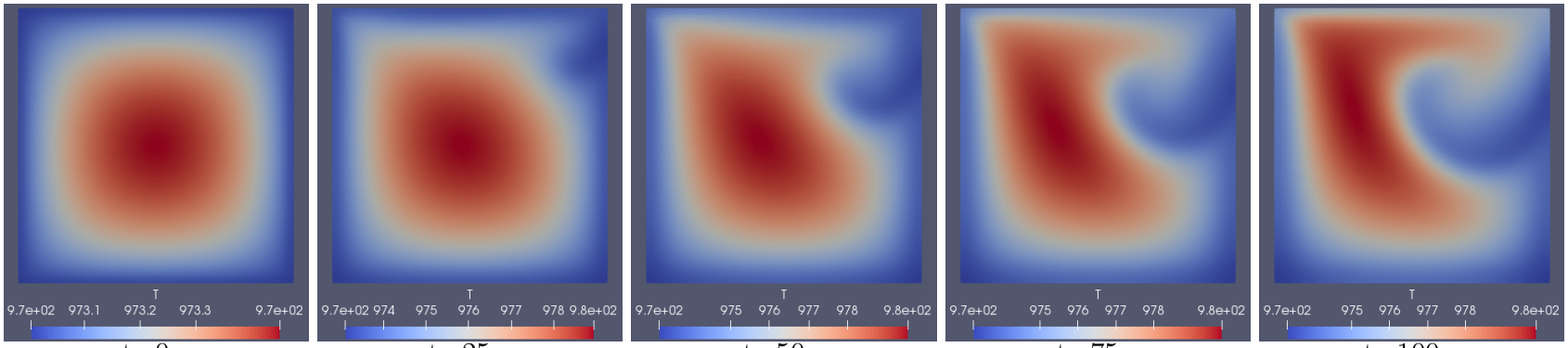


Figure 6.27: FOM temperature field T $[K]$ at different time instants - case C.

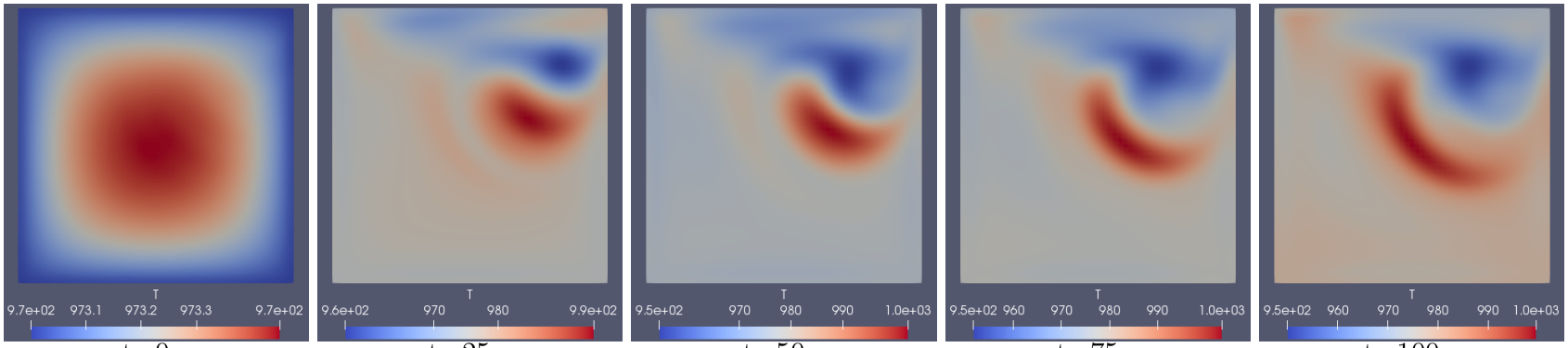


Figure 6.28: ROM temperature field T_r $[K]$ at different time instants - case C.

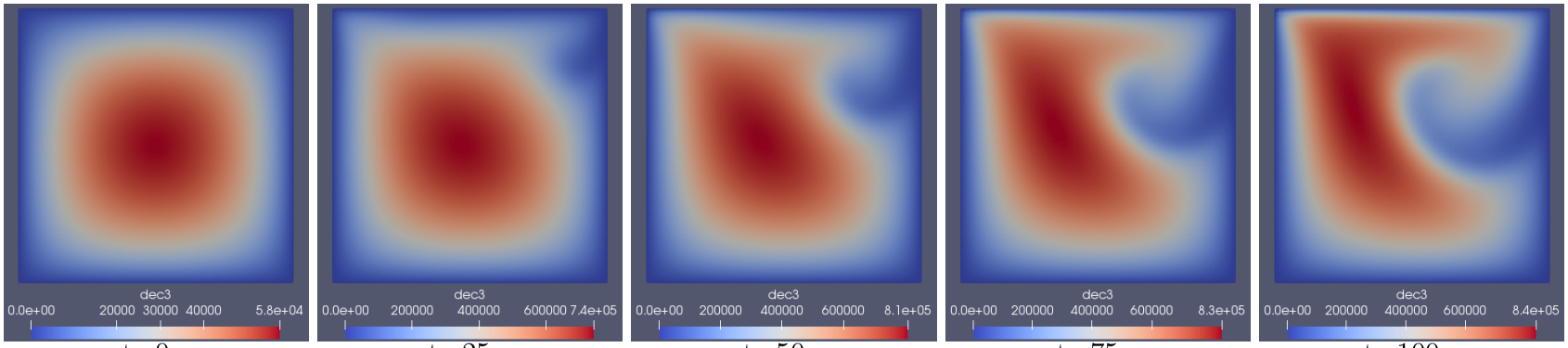


Figure 6.29: FOM third decay heat group H_3 $[J/m^3]$] at different time instants - case C.

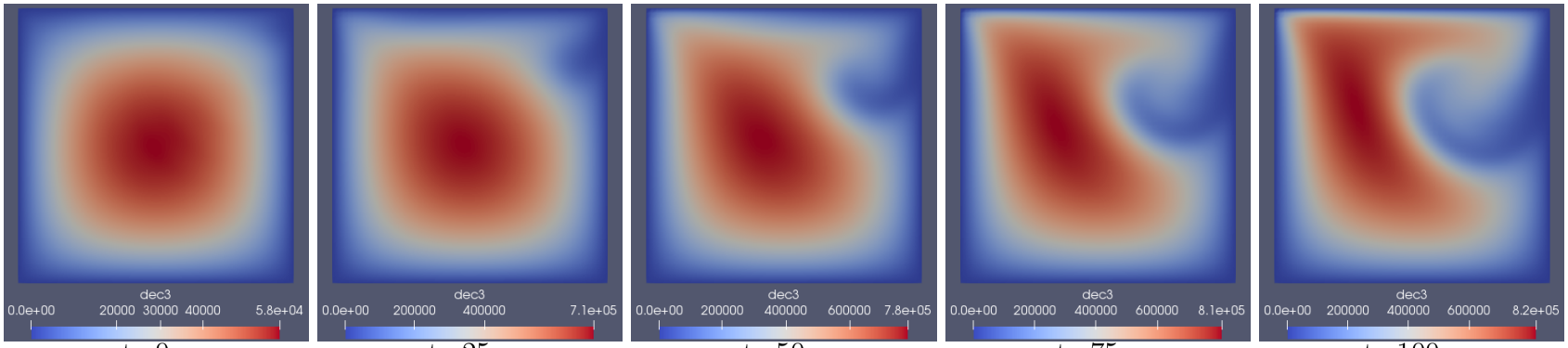


Figure 6.30: ROM third decay heat group $H_{3,r}$ $[J/m^3]$ at different time instants - case C.

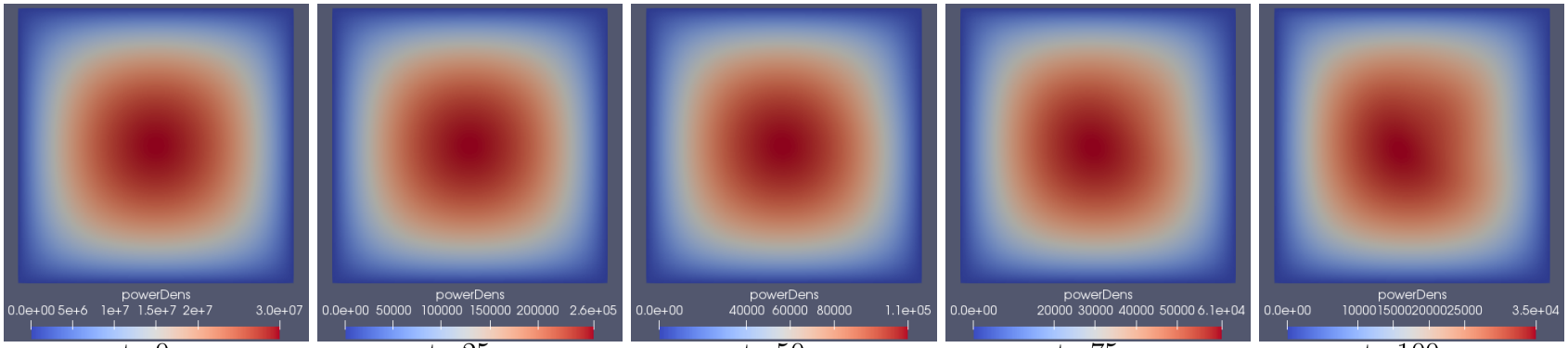


Figure 6.31: FOM power density W $[W/m^3]$ at different time instants - case C.

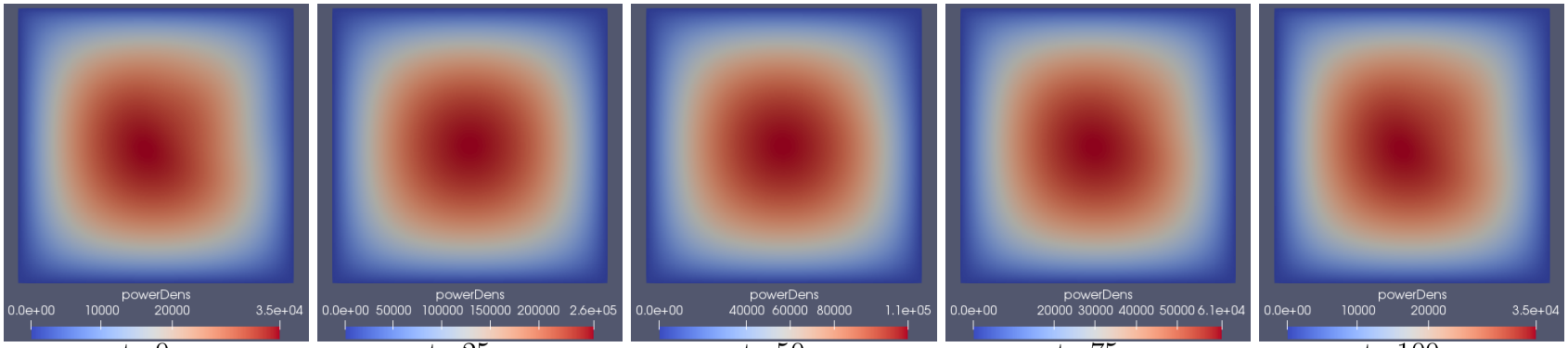


Figure 6.32: ROM power density W_r $[W/m^3]$ at different time instants - case C.

(This page is intentionally left blank)

Bibliography

[1] Quartetoni, A., Manzoni, A., Negri, F. (2016). Reduced Basis Methods for Partial Differential Equations - An Introduction, *Springer*, Berlin Heildeberg, 1st ed.

[2] Hesthaven, J. S., Rozza, G., Stamm., B. (2016). Certified Reduced Basis Methods for Parametrized Partial Differential Equations. *Springer International Publishing*, Berlin Heildeberg.

[3] Lorenzi, S., Cammi, A., Luzzi, L., and Rozza, G. (2016). POD-Galerkin method for finite volume approximation of Navier-Stokes and RANS equations. *Computer Methods in Applied Mechanics and Engineering*, Volume 311, Pages 151–179.

[4] Bergmann, M., Bruneau, C.-H., and Iollo, A. (2009). Enablers for robust POD models. *Journal of Computational Physics*, Volume 228(2), Pages 516–538.

[5] Stabile, G., Hijazi, S., Mola, A., Lorenzi, S. and Rozza, G. (2017). POD-Galerkin reduced order methods for CFD using Finite Volume Discretisation: vortex shedding around a circular cylinder. *Communications in Applied and Industrial Mathematics*, Volume 8, Pages 210-236.

[6] Stabile, G., Rozza, G. (2018). Finite volume POD-Galerkin stabilised reduced order methods for the parametrised incompressible Navier–Stokes equations. *Computers & Fluids*, Volume 173, Pages 273-284.

[7] Hijazi, S., Stabile, G., Mola, A., Rozza, G. (2019). Data-Driven POD-Galerkin reduced order model for turbulent flows. *Preprint submitted to Elsevier.*

[8] Vergari, L. (2018). Development of a reduced order modeling for parametrized thermal-hydraulics and neutronics problems. *Master thesis at Politecnico di Milano.*

[9] Aufiero, M., Cammi. A, Geoffroy, O., Losa, M., Luzzi, L., Ricotti, M. E., Rouch H. (2014). Development of an OpenFOAM model for the Molten Salt Fast Reactor transient analysis. *Chemical Engineering Science*, Volume 111, Pages 390-401.

[10] Lorenzi, S., Cammi, A., Luzzi, L., Rozza, G. (2016). POD-Galerkin method for finite volume approximation of Navier–Stokes and RANS equations. *Computer Methods in Applied Mechanics and Engineering*, Volume 311, Pages 151-179.

[11] Lorenzi, S., Cammi, A., Luzzi, L., Rozza, G. (2017). A reduced order model for investigating the dynamics of the Gen-IV LFR coolant pool. *Applied Mathematical Modelling*, Volume 46, Pages 263-284.

[12] Georgaka, S., Stabile, G., Rozza, G., Bluck, M., J. (2018). Parametric POD-Galerkin Model Order Reduction for Unsteady-State Heat Transfer Problems. *arXiv preprint arXiv:1808.05175*.

[13] GIF (Generation IV International Forum), *Generation IV Goals*, `https://www.gen-4.org/gif/jcms/c\_9502/generation-iv-goals`, 2013.

[14] OpenFOAM, *The open source CFD toolbox*, `https://www.openfoam.com/`.

[15] Greenshields,C. J. (2018). User Guide version 6. *OpenFOAM Foundation Ltd*, London.

[16] ITHACA-FV (In real Time Highly Advanced Computational Applications for Finite Volumes), `https://mathlab.sissa.it/ithaca-fv`

[17] Eigen, `http://eigen.tuxfamily.org/index.php?title=Main\_Page`

[18] SPECTRA (Sparse Eigenvalue Computation Toolkit as a Redesigned ARPACK), `https://spectralib.org/`

[19] Splinter, `https://github.com/bgrimstad/splinter`

[20] Brovchenko, M., Merle-Lucotte, E., Rouch, H., Alcaro F., Aufiero, M., Cammi, A., Dulla S. et al. (2013). Optimization of the pre-conceptual design of the MSFR - EVOL project. *Deliverable D2.2*.

[21] Bozeman, J. D., Dalton, C. (1973). Numerical study of viscous flow in a cavity. *Journal of Computational Physics*, Volume 12(3), Pages 348-363.

[22] Berkooz, G., Holmes, P.J., Lumley, J. (2003). The Proper Orthogonal Decomposition in the Analysis of Turbulent Flows, *Annual Review of Fluid Mechanics*, Volume 25, Pages 539-575.

[23] Lazzaro, D., Montefusco, L. B. (2002). Radial basis functions for the multivariate interpolation of large scattered data sets. *Journal of Computational and Applied Mathematics*, Volume 140(1-2), Pages 521– 536.

[24] Saltelli, A., Tarantola, S., Campolongo, F., Ratto, M. (2004). Sensitivity analysis in practice : a guide to assessing scientific models. *Wiley & Sons Ltd*, New York.