Distribution Agreement

In presenting this thesis or dissertation as a partial fulfillment of the requirements for an advanced degree from Emory University, I hereby grant to Emory University and its agents the non-exclusive license to archive, make accessible, and display my thesis or dissertation in whole or in part in all forms of media, now or hereafter known, including display on the world wide web. I understand that I may select some access restrictions as part of the online submission of this thesis or dissertation. I retain all ownership rights to the copyright of the thesis or dissertation. I also retain the right to use in future works (such as articles or books) all or part of this thesis or dissertation.

Signature:

_____ _____

Sofia Guzzetti Date

Reduced Models and Parallel Computing for Uncertainty Quantification in Cardiovascular Mathematics

By

Sofia Guzzetti
Doctor of Philosophy

Department of Mathematics

Alessandro Veneziani, Ph.D.
Advisor

Pablo J. Blanco, Ph.D.
Committee Member

James Nagy, Ph.D.
Committee Member

Lars Ruthotto, Ph.D.
Committee Member

Accepted:

Lisa A. Tedesco, Ph.D.
Dean of the Graduate School

Date

Reduced Models and Parallel Computing for Uncertainty Quantification in Cardiovascular Mathematics

By

Sofia Guzzetti
Ph.D., Emory University, 2019

Advisor: Alessandro Veneziani, Ph.D.

An abstract of
A dissertation submitted to the Faculty of the Graduate School
of Emory University in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
in Department of Mathematics
2019

Abstract

Reduced Models and Parallel Computing for Uncertainty Quantification in Cardiovascular Mathematics

By Sofia Guzzetti

Computational fluid dynamics (CFD) has been progressively adopted in the last decade for studying the role of blood flow in the development of arterial diseases. While computational (*in silico*) investigations - compared to more traditional *in vitro* and *in vivo* studies - are generally more flexible and cost-effective, the adoption of CFD for computer-aided clinical trials and surgical planning is still an open challenge. The computational time to accurately and reliably solve mathematical models can be too long for the fast-paced clinical environment - especially in emergency scenarios, and quantifying the reliability of the results comes at an even higher computational cost. Moreover, the *in silico* analysis of large numbers of patients calls for significant computational resources. Hospitals and healthcare institutions are expected to outsource numerical simulations, which, however, raises concerns about privacy, data protection, and efficiency in terms of cost and performance. In such an articulated and complex scenario, this work addresses the challenges described above by (i) introducing a novel reduced model that guarantees levels of accuracy comparable to those achieved by high-fidelity 3D models, roughly at the same computational cost as the inexpensive yet inaccurate 1D models, by combining the Finite Element Method to describe the main stream dynamics with Spectral Methods to retrieve the transverse components; (ii) designing a new method for uncertainty quantification in large-scale networks that greatly enhances parallelism by performing uncertainty quantification at the subsystem level, and propagating uncertainty information encoded as polynomial chaos coefficients via overlapping domain decomposition techniques; (iii) providing an objective criterion to measure the performance of different parallel architectures based on the user's priorities in terms of budget and tolerance to delay, and reducing the execution time by choosing a task-worker mapping strategy ahead of simulation time, and optimizing the amount of overlap in the domain decomposition phase.

Reduced Models and Parallel Computing for Uncertainty Quantification in Cardiovascular Mathematics

By

Sofia Guzzetti
Ph.D., Emory University, 2019

Advisor: Alessandro Veneziani, Ph.D.

A dissertation submitted to the Faculty of the Graduate School
of Emory University in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
in Department of Mathematics
2019

Acknowledgments

"Ricorda sempre che, anche quando tu non avrai fiducia in te stessa, qui ci sarà sempre qualcuno che ne avrà per te". This is how amazing my advisor has been. Thank you for believing in me, even when it was hard for me to do so. Your passion for applied Mathematics is contagious and kept my determination vibrant even with the hardest research challenges. Thank you for caring about me personally and as a student, and for valuing us as a group, because - as you say - "the strength of the wolf is the pack, and the strength of the pack is the wolf".

Thanks to Simona, for her scientific rigor and support, and to Vaidy, who advised and encouraged me during my first year and throughout my studies. I learned a lot from you and I am really grateful for it!

I would like to thank Jim and Lars, and all the professors who contributed to consolidate and broaden my knowledge. You stimulated and encouraged my curiosity, and led me to explore and appreciate fields in Mathematics and Computer Science I was not familiar with. And most importantly, thank you for teaching me not to fear questions, rather to embrace and nourish them.

Pablo and Alonso, my visit was one of the happiest times of my PhD, and working happily makes a whole world of difference. Thank you for advising me so generously, and for taking care of me. Alonso, after all my "spamming", I think we brought remote collaboration to the next level! Thank you for your inifinite patience with me, you are the best.

Kevin, Moe, Khachik, and Ray: it has been so amazing working with each one and all of you! With your expertise, commitment, passion and enthusiasm even the toughest problems are not intimidating. Thank you for the time and patience you dedicated to me. Kevin, special thanks to you, for being so attentive and supportive.

Nulla di tutto ciò sarebbe stato possibile senza la mia famiglia, che mi e' sempre stata vicina anche da lontano. Grazie per aver condiviso tanto la gioia delle mie conquiste quanto la fatica delle mie battaglie, per avermi spronato e consigliato con affetto e delicatezza, per lasciarmi sempre libera e mai sola nella scelta. Siete il Bene più prezioso che ho.

Un grazie speciale alla mia nonna, esempio inimitabile di saggezza e modernità, pazienza e lungimiranza. La tua grinta e il tuo affetto sono la

mia carica per perseverare. Grazie per aver sopportato la distanza per darmi lo spazio per crescere.

Grazie alla mia famiglia di Atlanta: Massi e Ale, che cosa avrei fatto senza di voi? Quante ne abbiamo passate... grazie per le risate, i consigli, le chiacchierate e i silenzi. Continuare a vivere la vostra amicizia da vicino è stata una fortuna insperata.

Grazie a Manuela, perché ho trovato un'amica, e alle mie amiche di sempre, Marina, Mery, Chiara e Francesca: siete le mie radici.

Cecilia, Maria, Federico, David: you are the best roommates ever! Thank you for being *home* to me, for listening, supporting and cheering me up in the toughest days. I (will) miss you so much!

Jeanne, John, you certainly witnessed the hardest time during my PhD. Thank you for laughing with me, for reminding me that the world was still out there in all its beauty, for supporting me.

Myriam, muito obrigada por tudo, por receberme calorosamente no Brasil e por tratarme como uma filha.

Diversity is one of the greatest gifts I received from this experience. I won't take the risk of forgetting someone by making a list of all those who shared part of or all their path with me, in Atlanta, Livermore or Petropolis: if you are reading, this is for you too! Whether you are American, because you really made me feel welcome, or European, because you made me feel less far away from home, or Latino/a, because you reminded me that happiness is also in the simplest things, or from any other Country, because you made me experience your side of the world, thank you for making me a richer person by sharing your culture with me.

Thank *you*, for being my anchor, gentle breeze or strong wind filling my sails, and my safe harbor in the storm. Thank you for making Beauty part of my life again.

A nonna Pia

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Computational fluid dynamics (CFD) has been progressively adopted in the last decade for studying the role of blood flow on the development of arterial diseases (see, e.g., [75, 186]). Computational investigations – compared to more traditional *in vitro* and *in vivo* studies – are generally more flexible and cost-effective. In combination with appropriate image-processing techniques – see, e.g., [148, 20, 190, 88, 193, 87, 202] – CFD can be used in a *patient-specific setting*. This means that the morphological and functional conditions of a specific patient may be reproduced in mathematical terms and quantitative analyses can be performed by solving the corresponding models describing the physical and constitutive laws behind the physiopathology, generally described by systems of partial differential equations. There are several uses for this kind of analysis, including a deeper understanding of the clinical conditions, performing virtual surgery or therapy for predicting outcomes, to a personalized optimization/customization of generic procedures [182, 100, 60, 131, 139]. Recently, the concept of Computer Aided Clinical Trials (CACT) has been proposed to identify the systematic use of scientific computing within the frameworks of studies oriented to extract knowledge from clinical data [193, 81]. Similarly, Surgical Planning (SP) is the name given to an ensemble of procedures aimed at the prediction of the outcomes of an operation by exploiting the intrinsic predictive potential of numerical models.

The adoption of CFD for CACT and SP is however still an open challenge. First of all, the time for obtaining results from computational studies can be too long for the fast-paced clinical environment - especially in emergency scenarios. Furthermore,

many patient-specific data that are input to mathematical models - such as parameters or boundary conditions, are totally inaccessible from a clinical viewpoint, and the correlation between clinical outcomes and flow patterns needs to be supported by large volumes of data, which are not always available or easy to obtain. This introduces uncertainties in the outputs of the numerical solvers, which raise the question of reliability. Therefore, *quantifying* the reliability of the results is crucial for applicability in real-life scenarios, but this comes at an even higher computational cost. Finally, the computational analysis of large numbers of patients calls for significant computational resources [193]. Scientists and clinicians have access to computing platforms which could alleviate the resource bottleneck. While local (owned) resources are faster and cheaper, overall system and operating expenses have led to resource sharing and leasing paradigms, i.e., *grids and clouds*, respectively. In facts, hospitals and healthcare institutions are expected to outsource numerical simulations in a routine process more than hosting local computing facilities. This is already happening, for example, with HeartFlow [1]. However, this raises concerns about privacy and data protection, and on efficiency, as it is not trivial to identify the platform that best suits the problem to be solved in each situation. In real production settings, performance must be judiciously balanced with cost.

In short, we may say that computational hemodynamics is a field with great potential, but currently limited by time and cost constraints. As a matter of fact, differently than in proofs of concept, in CACT and SP accuracy of computations is not the only priority - as far as numerical simulations obtain sufficient reliability to correctly support clinical practice. Efficiency and robustness of the numerical procedures must be properly considered as well.

In such an articulated and complex scenario, this work addresses the challenges described above by designing ad-hoc mathematical and computational methods for cardiovascular applications (see Figure 1.1). In **Chapter 2** we present a novel reduced model tailored to 3D domains with a geometrically dominant direction. A "smart" choice of the basis functions of the discrete function spaces guarantees lev-

els of accuracy comparable to those achieved by high-fidelity 3D models, but at a computational cost of the same order as the inexpensive yet inaccurate 1D models. A new method for uncertainty quantification (UQ) in large-scale networks is presented in **Chapter 3**. The application of domain decomposition (DD) in an off-line phase allows solving the stochastic problem only locally - at the sub-system level, greatly enhancing parallelism. We introduce new multigrid (MG) methods specifically designed for networks, that - potentially combined with Anderson Acceleration - consistently improve the performance of the numerical model for uncertainty quantification. In **Chapter 4**, the DDUQ method is applied to a variation of the "smart" reduced model for the cardiovascular system presented in Chapter 2. This enables accurate uncertainty quantification in patient-specific geometries roughly at the cost of traditional cheap yet inaccurate reduced models. The infrastructure issue is tackled in **Chapter 5**. We provide an objective criterion to measure the performance of different parallel architectures based on the user's priorities in terms of budget and tolerance to delay. We show that choosing a task-worker mapping strategy ahead of simulation time can consistently reduce the execution time, and that allowing a small amount of overlap in the domain decomposition phase can further improve convergence. We conclude with an overview on possible future work.

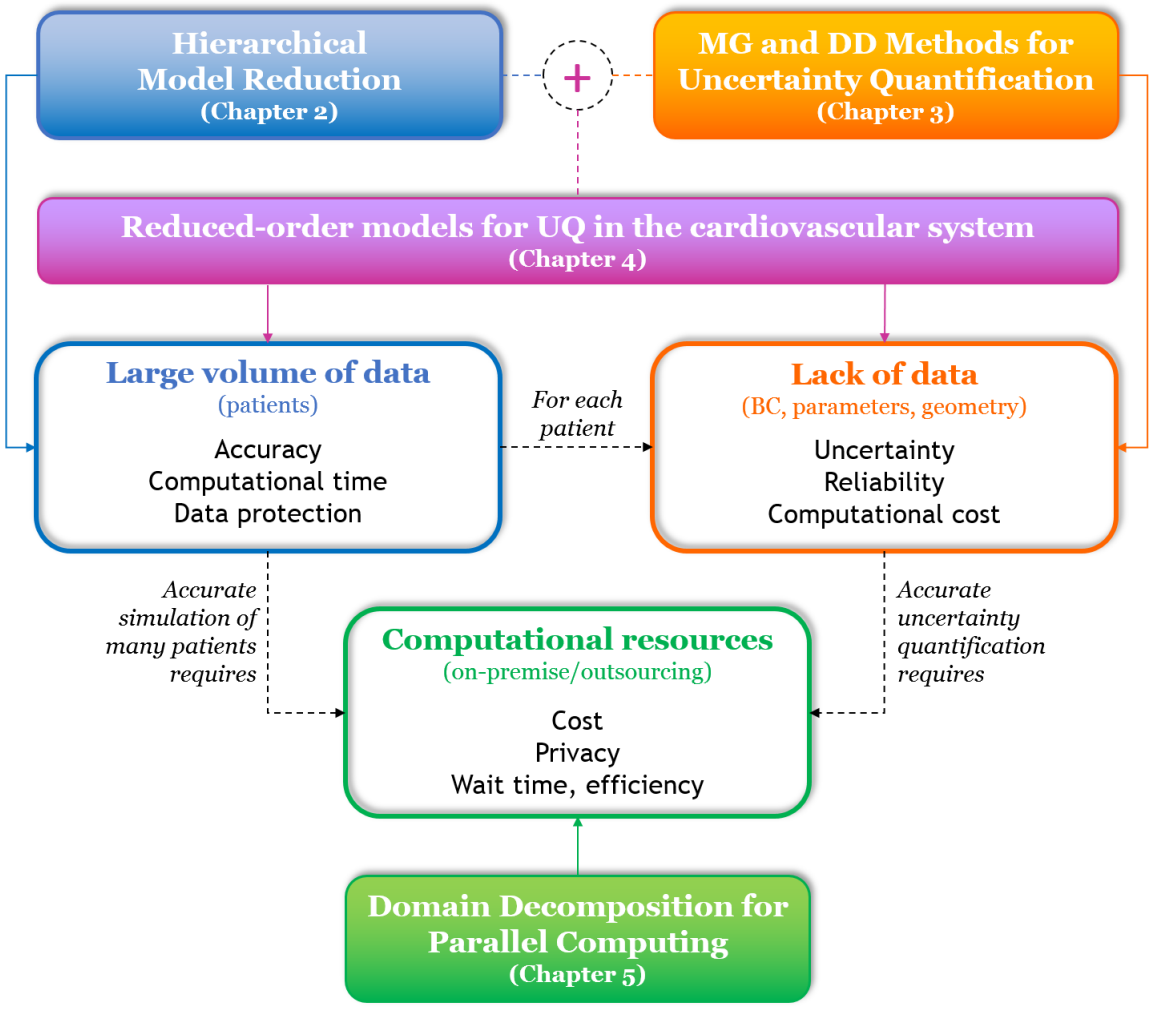


Figure 1.1: Thesis content chart. The challenges addressed in this work and the corresponding proposed solutions are in white and colored boxes, respectively. Dashed arrows represent relationships between challenges, while solid arrows show problem-solution connections.

Chapter 2

Hierarchical Model Reduction

Acknowledgements. This chapter reflects the content of the paper [98], written in collaboration with Simona Perotto and Alessandro Veneziani, that has been part of the research of the author during the years of the graduate program. In addition, we provide a numerical proof of the inf-sup stability of the reduced spaces considered, and extend numerical tests to patient-specific geometries.

2.1 Introduction and Background

Hierarchical Model (HiMod) Reduction is a method proposed in [143] for the efficient solution of Partial Differential Equations defined in domains with a geometrically dominant direction, like slabs or pipes. In the spirit of a separation of variables, the HiMod solution is regarded as the combination of the mainstream dynamics driven by the geometry and the transverse components. The latter are generally of secondary importance so to be described by few degrees of freedom of a spectral approximation introduced along the transverse direction, in combination with a finite element or an isogeometric discretization of the mainstream (see figure 2.1, left). Thus, a purely 1D numerical approximation can be promptly expanded towards the original 3D problem by a proper selection of the spectral discretization [146]. Although purely 1D models completely drop the transverse dynamics [149, 76, 119], these may be locally important. HiMod is intended to provide a unified numerical tool able to promptly

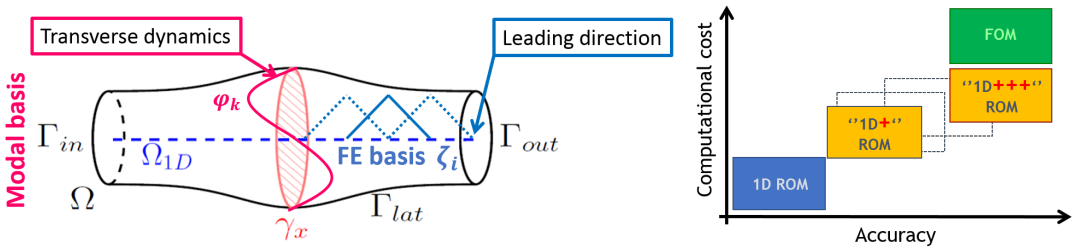


Figure 2.1: Schematic representation of a Hierarchical Model Reduction (left); computational cost as a function of accuracy for 1D ROM, HiMod, and full-order models (FOM) (right).

incorporate the transverse components of the 3D solution into a "psychologically 1D" solver (see figure 2.1, right). The HiMod reduction has been successfully employed to solve Advection-Diffusion-Reaction (ADR) problems in two-dimensional domains [143, 145, 69, 141, 146, 147, 30, 142, 144] and parallelepipeds [9].

Three dimensional problems in non-trivial geometries are considered in [127] for incompressible fluid dynamics, including domains reconstructed from patient-specific coronaries. Paper [127] proposes a specific formulation of the HiMod framework, where the problem is discretized with an approach called Transversally Enriched Pipe Elements Method (TEPEM). The physical elements are mapped to a reference slab where the Cartesian framework is exploited for the construction of the modal basis by tensor product of Legendre polynomials. The method is proved to be reliable and efficient (when compared to 3D solution), with computational costs close to the ones of a 1D solver in both idealized and real geometries.

In circular pipes, where the transverse section is (or it is close to be) a circle, like in arteries, a natural candidate for the non-axial dynamics is a polar coordinate setting. Nevertheless, the spectral approximation of Partial Differential Equations in polar coordinates suffers from several issues, consequent to the conversion from the Cartesian frame. Regularity and boundary condition enforcement add requirements to the spectral basis to use. The selection of a basis with both analytical and nu-

merical excellent features is thus challenging. For this reason, investigations on such problems are often broken into axisymmetric [167, 123, 27, 62, 21] and general cases [122]. Axial symmetry may, in fact, influence the selection of the basis. Quoting Boyd (referring to Laplace problems in 2D circular domains - yet the conclusion is general) "There is no single best basis for the disk" [37].

In this paper, for the first time polar coordinates are explored in the HiMod approach. Potentialities and drawbacks of different choices are discussed in an extensive numerical testing. The ultimate application we are interested in is computational hemodynamics, yet we perform extensive benchmarking in idealized geometries, with and without axial symmetry. We specifically consider two problems, advection-diffusion-reaction (ADR) and incompressible Navier-Stokes Equations (NSE), both related to blood flow problems. We show how accurate the hybrid FEM/Spectral approximation is, in spite of numerical problems related to the selection of the spectral basis in polar frames. Specifically, we find that a Top-Down basis resulting from the solution of a Sturm-Liouville Eigenvalue problem is a viable option for ADR, since it can be designed to incorporate any kind of boundary conditions, but it is outperformed by a Bottom-Up approach for the specific case of homogeneous Dirichlet boundary conditions. For the NSE, a Bottom-Up basis construction is numerically more robust and to be preferred, in particular resorting to the so-called Zernike polynomials (aka one-sided Jacobi-Fourier basis). The HiMod approach with these choices is an improved 1D solver to be next used in networks of pipes [33, 30].

In Section 2.2 we address the HiMod discretization in cylindrical domains in polar coordinates. We discuss several options for the construction of the spectral basis, following the excellent extensive work of Boyd [37]. In Section 2.3 we numerically test ADR, while in Section 2.4 we move to NSE. For the available Womersley solution, we compare the results presented here with [127] to discuss pros and cons of using polar coordinates vs geometrical mapping to slab-like Cartesian domains. In view of medical applications, in Section 2.5 we consider both idealized and non-trivial models for arterial domains. Conclusions are drawn in the last Section.

2.2 HiMod in Cylindrical Domains

2.2.1 The geometric setting

Let us consider a cylindrical domain with a rectilinear axis and circular section with variable radius R. We assume that the domain can be represented as a three-dimensional fiber bundle $\Omega = \bigcup_{x\in\Omega_{1D}}\{x\} \times \gamma_x$, where Ω_{1D} is a one-dimensional domain, and $\gamma_x \subset \mathbb{R}^2$ represents the two-dimensional fiber associated with the generic point $x \in \Omega_{1D}$ (Figure 2.2). In particular, the supporting fiber is the $x-$axis, while γ_x is the transverse section centered at x. The leading dynamics are aligned with Ω_{1D}, whereas the transverse dynamics are parallel to fibers γ_x. For the sake of simplicity, we assume a rectilinear axis $\Omega_{1D} =]x_0, x_1[$, but the more general case of a curved supporting fiber can be considered as well [142, 144]. For each $x \in \Omega_{1D}$, we introduce the mapping

$$\psi_x \ : \ \gamma_x \to \hat{\gamma}, \tag{2.1}$$

between the physical fiber γ_x and a reference fiber $\hat{\gamma}$. We set $\hat{\mathbf{z}} = (\hat{x}, \hat{\mathbf{y}}) = \hat{\boldsymbol{\psi}}(x, \mathbf{y}) = (x, \psi_x(\mathbf{y}))$ as the image of the physical point $\mathbf{z} = (x, \mathbf{y}) \in \Omega$ through the global map $\hat{\boldsymbol{\psi}} : \Omega \to \hat{\Omega}$, where $\hat{\Omega}$ is the reference cylinder described by the coordinates $(x, \hat{\mathbf{y}})$, with $\hat{\mathbf{y}} = \psi_x(\mathbf{y}) = (\hat{r}, \hat{\vartheta}) \in [0, 1) \times [0, 2\pi]$, so that the transverse reference fiber $\hat{\gamma}$ coincides with the unit circle (Figure 2.2). We assume ψ_x to be a C^1-diffeomorphism for all $x \in \Omega_{1D}$ and $\hat{\boldsymbol{\psi}}$ to be differentiable with respect to $\mathbf{z}$.

A cylindrical domain can be parametrized in different ways [35, 127]. A simple approach is to map the circular section into a square (or a geometry easy to separate along the Cartesian coordinates), which allows using a Cartesian framework in the reference space. In this case, the geometric map lacks regularity at the vertices of the hexahedron, as a counterpart to its conceptual simplicity. On the contrary, the choice of a cylindrical setting is intrinsically more tailored to the geometry of a pipe. Here, we introduce the HiMod polar framework and discuss pros and cons of this choice, with all the challenges pointed out in [37].

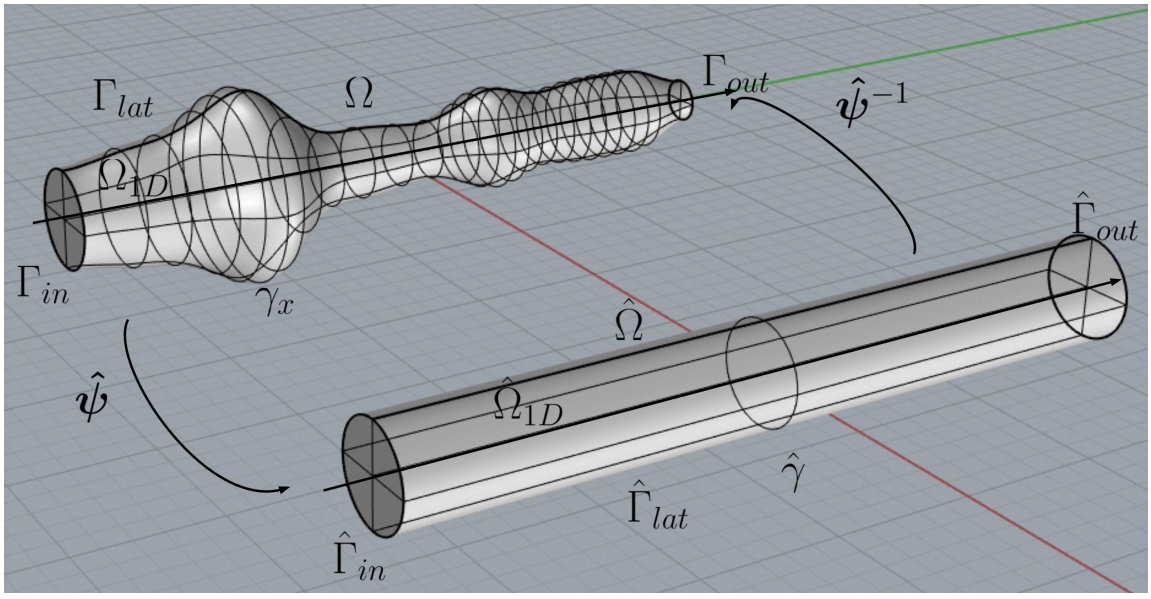


Figure 2.2: A physical domain of interest (in this case an axisymmetric geometry obtained by rotation of a curve around the x axis) mapped to the reference cylindrical domain.

2.2.2 The reference basis set

The fiber structure featured by the domain Ω has a key role in setting the HiMod reduction. We resort to different function spaces along Ω_{1D} and on the transverse fibers. The standard notation for the Sobolev spaces as well as for the space of functions bounded a.e. in Ω is adopted [40]. With reference to standard scalar ADR problems, we introduce the function space $V_{1D} \subseteq H^1(\Omega_{1D})$ on Ω_{1D}, such that the related functions vanish on Dirichlet boundaries. On the transverse reference fiber we set a modal basis $\{\hat{\varphi}_k\}_{k\in\mathbb{N}^+} \subset H^1(\hat{\gamma})$. In particular, we select functions orthonormal with respect to a weighted scalar product in $L^2(\hat{\gamma})$. Clearly, boundary conditions on Γ_{lat} have to be taken into account by the modal basis. Then, the discrete transverse function space is defined as $V_{\hat{\gamma}} = span\{\hat{\varphi}_k\}$. For a certain $m \in \mathbb{N}^+$, the combination

of spaces V_{1D} and $V_{\hat{\gamma}}$ yields the *reduced space*

$$V_m = \left\{ v_m(x, \mathbf{y}) = \sum_{k=1}^{m} \tilde{v}_k(x)\hat{\varphi}_k(\psi_x(\mathbf{y})), \text{ with } \tilde{v}_k \in V_{1D},\ \hat{\varphi}_k \in V_{\hat{\gamma}},\ x \in \Omega_{1D},\ \mathbf{y} \in \gamma_x \right\}, \tag{2.2}$$

with

$$\tilde{v}_k(x) = \int_{\hat{\gamma}} v_m(x, \psi_x^{-1}(\hat{\mathbf{y}}))\hat{\varphi}_k(\hat{\mathbf{y}})d\hat{\gamma} \qquad k \in \{1, \dots, m\}, \tag{2.3}$$

for the orthonormality of the basis.

Out of many possible choices for the construction of a basis, here we identify two possible general strategies.

(i) In the "*Top-Down*" approach, we construct directly the bivariate basis function in $(\hat{r}, \hat{\vartheta})$ by solving a Sturm-Liouville Eigenvalue (SLE) problem associated with an appropriate self-adjoint differential operator defined on the unit-disk, completed with the homogeneous boundary conditions of the original problem to solve. We mimic in this way the "educated approach" introduced in [9] for ADR problems in slabs, where tensor product splitting allows to simplify the 2D computation to 1D SLE problems.

(ii) In the "*Bottom-Up*" approach we factorize the construction of the basis by working along $\hat{r}$ and $\hat{\vartheta}$, separately, and assembling the bivariate basis afterwards. Boundary conditions are then enforced at a second stage. Several options are available, as discussed in [35, 37], which we follow closely.

We discuss these two options hereafter.

The Top-Down approach

A possible strategy to build the modal basis is to solve an auxiliary SLE problem [56, 44] on the transverse section,

$$\begin{cases} \mathscr{L}\hat{\varphi}_k = \hat{\lambda}_k w \hat{\varphi}_k & \text{in } \hat{\gamma} \\ BC = 0 & \text{on } \partial\hat{\gamma}, \end{cases} \tag{2.4}$$

BC type	Condition	Functor
Dirichlet	$\hat{\varphi}_k = 0$	$J_n(\sqrt{\hat{\lambda}_j})$
Neumann	$\mu\nabla\hat{\varphi}_k \cdot \mathbf{n} = 0$	$\frac{\sqrt{\hat{\lambda}_j}}{R} J_n'(\sqrt{\hat{\lambda}_j})$
Robin	$\mu\nabla\hat{\varphi}_k \cdot \mathbf{n} + \chi\hat{\varphi}_k = 0$	$\frac{\sqrt{\hat{\lambda}_j}}{R} J_n'(\sqrt{\hat{\lambda}_j}) + \frac{\chi}{\mu} J_n(\sqrt{\hat{\lambda}_j}) \quad (\mu, \chi > 0)$

Table 2.1: Functors associated with different types of boundary conditions.

so to include the generic homogeneous boundary conditions ($BC = 0$) in the basis in an essential way. Here $\mathscr{L}$ is a suitable differential operator, $(\hat{\lambda}_k, \hat{\varphi}_k)$ denotes a corresponding eigenpair, and w is a positive continuous weight function. Under the assumptions of the Spectral Theorem [160], the eigenfunctions associated with the operator $\mathscr{L}$ are orthogonal with respect to the L^2_w-weighted scalar product and form a complete set in the same space. Since functions $\hat{\varphi}_k$ automatically include the lateral boundary conditions, the set $\{\hat{\varphi}_k\}$ has been called an "educated" basis [9].

Scalar problems. Let $\mathscr{L}$ in (2.4) be the Laplacian (in general, it can be a self-adjoint operator) on the reference unit circle in polar coordinates. For simplicity, we consider here homogeneous Dirichlet boundary conditions, although any type of boundary conditions can be enforced. The set of eigenfunctions associated with this operator is

$$\hat{\varphi}_k(\hat{r}, \hat{\vartheta}) = \hat{\varphi}_{j,n}(\hat{r}, \hat{\vartheta}) = \frac{1}{\sqrt{2\pi}\|J_n\|_{L^2_w(0,1)}} \left(\sin(n\hat{\vartheta}) + \cos(n\hat{\vartheta})\right) \; J_n\left(\sqrt{\hat{\lambda}_j}\hat{r}\right), \quad (2.5)$$

where J_n is the Bessel function of first type of order $n \in \mathbb{N}^+$ [56, 203, 180, 59], and the frequency $\sqrt{\hat{\lambda}_j}$ is the j-th root of J_n. More in general, for each type of boundary condition (Dirichlet/Neumann/Robin), $\hat{\lambda}_j$ is obtained as the (squared) j-th root of a specific functor (see Table 2.1). As a result, the ordering of the basis functions $\{\hat{\varphi}_k\}$ depends on the two indices j and n, i.e., $k = k(j, n)$ (for more details, see [94]).

Vector problems. In view of hemodynamic applications, hereafter we construct a modal basis with the eigenfunctions of the Stokes operator, completed with homogeneous Dirichlet boundary conditions. More explicitly, for such a choice the Sturm-Liouville eigenvalue problem (2.4) reads

$$\begin{cases} \Delta\hat{\boldsymbol{\varphi}}_k^u + \hat{\lambda}_k\hat{\boldsymbol{\varphi}}_k^u = \nabla\hat{\varphi}_k^p & \text{in } \hat{\gamma}, \\ \Delta\hat{\varphi}_k^p = 0 & \text{in } \hat{\gamma}, \\ \nabla\cdot\hat{\boldsymbol{\varphi}}_k^u = 0 & \text{in } \hat{\gamma}, \\ \hat{\boldsymbol{\varphi}}_k^u = 0 & \text{on } \partial\hat{\gamma}, \end{cases} \tag{2.6}$$

being $\hat{\boldsymbol{\varphi}}_k^u$ and $\hat{\varphi}_k^p$ the eigenfunctions for the velocity and the pressure, respectively. Following the procedure adopted for scalar problems, the eigenfunctions are computed as in [158] and they read as

$$\hat{\varphi}_k^p(\hat{r},\hat{\vartheta}) = \hat{\varphi}_{j,n}^p(\hat{r},\hat{\vartheta}) = c_1\hat{r}^n\exp(in\hat{\vartheta}),$$

$$\hat{\boldsymbol{\varphi}}_k^u(\hat{r},\hat{\vartheta}) = \hat{\boldsymbol{\varphi}}_{j,n}^u(\hat{r},\hat{\vartheta}) = c_1\exp(in\hat{\vartheta})\begin{bmatrix} \dfrac{n}{\hat{\lambda}_j}\hat{r}^{n-1} - \dfrac{nJ_n\left(\sqrt{\hat{\lambda}_j}\hat{r}\right)}{\hat{r}\hat{\lambda}_j J_n\left(\sqrt{\hat{\lambda}_j}\right)} \\ in\left(\dfrac{\hat{r}^{n-1}}{\hat{\lambda}_j} - \dfrac{J_{n-1}\left(\sqrt{\hat{\lambda}_j}\hat{r}\right) - J_{n+1}\left(\sqrt{\hat{\lambda}_j}\hat{r}\right)}{2n\sqrt{\hat{\lambda}_j}J_n\left(\sqrt{\hat{\lambda}_j}\right)}\right) \end{bmatrix}, \tag{2.7}$$

respectively, where i is the imaginary unit, $\sqrt{\hat{\lambda}_j}$ runs through all the roots of J_{n+1}, for $n \neq 0$, and the coefficient c_1 is determined via the unitary norm constraint. Note that, as in the scalar case, the ordering of the basis functions depends on the two indices j and n. Although from a theoretical viewpoint functions $\hat{\varphi}_k^p$ and $\hat{\boldsymbol{\varphi}}_k^u$ are tailored to the problem we aim to solve, from a practical perspective there are some drawbacks. Specifically, we need here to drop the complex part, with a relevant loss of details. More in general (for both scalar and vector problems), Bessel functions are extremely sensitive to numerical errors [36, 59, 181, 35, 89, 44] and this may limit

their use. Therefore, for the numerical tests presented hereafter, a Top-Down vector basis is built by employing functions (2.5) for each component of the vector function.

The Bottom-Up approach

As an alternative to the Top-Down approach, a two-step procedure is proposed in [35]. Two scalar 1D Sturm-Liouville eigenvalue problems are solved for the angular and radial component, respectively. For the ϑ-component, $\mathscr{L}$ is chosen as the 1D Laplacian, leading to the standard Fourier basis, whereas a radial basis $\{\xi_n(\hat{r})\}_{n=0}^{\infty}$ is obtained by defining the differential operator as $\mathscr{L}\xi_n = -(s\xi_n')' + q\xi_n$, being $s > 0$ and q given functions of $\hat{r}$. Then, the 2D spectral basis is assembled by multiplying the basis functions associated with each component, as

$$\hat{\varphi}_k(\hat{r}, \hat{\vartheta}) = \hat{\varphi}_{j,n}^{\sin,\cos}(\hat{r}, \hat{\vartheta}) = \xi_n(\hat{r}) \begin{cases} \cos(j\hat{\vartheta}) \\ \sin(j\hat{\vartheta}), \end{cases} \tag{2.8}$$

where the superscript sin/cos specifies the type of trigonometric function considered and $j, n \in \mathbb{N}^+$. As in the Top-Down approach, the ordering of the basis functions $\{\hat{\varphi}_k\}$ depends on the two indices j and n [94]. Thus, the expansion of an arbitrary function $\psi \in L_w^2([0,1] \times [0, 2\pi))$ reads

$$\psi(\hat{r}, \hat{\vartheta}) = \sum_{j=0}^{\infty} \sum_{n=0}^{\infty} \Big[f_{jn} \xi_n(\hat{r}) \cos(j\hat{\vartheta}) + g_{jn} \xi_n(\hat{r}) \sin(j\hat{\vartheta}) \Big], \tag{2.9}$$

with f_{jn}, $g_{jn} \in \mathbb{R}$ for any $j, n \in \mathbb{N}^+$. Smoothness properties of this series are guaranteed by the Parity Theorem, that rules the combination of radial and trigonometric components in the construction of a polar modal basis. We provide the corresponding statement for completeness by collecting Theorems 2.1 and 2.2 in [37]:

Theorem 2.1. *Let $\psi(y, z)$ be a function written in polar coordinates, so that in particular*

$$\psi(\hat{r}, \hat{\vartheta}) = \psi(-\hat{r}, \hat{\vartheta} + \pi).$$

Let the corresponding Fourier series be

$$\psi(\hat{r},\hat{\vartheta}) = \sum_{j=0}^{\infty} c_j(\hat{r}) e^{ij\hat{\vartheta}} = \sum_{j=0}^{\infty} \left[f_j(\hat{r})\cos(j\hat{\vartheta}) + g_j(\hat{r})\sin(j\hat{\vartheta}) \right].$$

Then, the coefficients satisfy

$$c_j(\hat{r}) = (-1)^j c_j(-\hat{r}), \quad f_j(\hat{r}) = (-1)^j f_j(-\hat{r}), \quad g_j(\hat{r}) = (-1)^j g_j(-\hat{r}).$$

The power series of $f_j(\hat{r})$ and $g_j(\hat{r})$ contain only odd (even) powers of $\hat{r}$ if j is odd (even). Also, assume that ψ is analytic in $\hat{r} = 0$. Then

$$f_j(\hat{r}) = \hat{r}^j F_j(\hat{r}), \quad g_j(\hat{r}) = \hat{r}^j G_j(\hat{r})$$

for suitable $F_j(\hat{r})$ and $G_j(\hat{r})$ non singular functions at $\hat{r} = 0$, so that

$$f_j(\hat{r}) = \hat{r}^j \sum_{k=0}^{\infty} f_{j,2k}\hat{r}^{2k}, \quad g_j(\hat{r}) = \hat{r}^j \sum_{k=0}^{\infty} g_{j,2k}\hat{r}^{2k}$$

for suitable coefficients $f_{j,k}$, $g_{j,k}$.

The Bottom-Up approach leads to some complications for the hierarchical ordering of the spectrum. Since no 2D eigenvalue problem is defined, there does not exist any hierarchy between the r-eigenvalues and the ϑ-eigenvalues, and the corresponding eigenfunctions. Hence, the selection of the number of radial modes and of trigonometric functions is somehow arbitrary. Two criteria are proposed in [35], namely *rectangular* and *triangular* truncations. The former employs the same number of radial functions for each angular wave number j. The latter decreases the number of radial basis functions as j increases, until the highest wave number has a single radial basis function. This criterion is usually employed for spherical harmonics because it guarantees the property of "equiareal resolution". Nevertheless, such property is not guaranteed in a cylindrical setting. Therefore, depending on the specific choice of the radial basis, a rectangular truncation may be preferable, being beneficial in terms of implementation.

Family	$\xi_n(\hat{r})$, $n = n(j)$	parameters
Bessel	$J_n(\hat{\lambda}_j \hat{r})$	$\hat{\lambda}_j$: the j-th root of $J_n(\hat{r})$
Robert	$\hat{r}^j T_n(\hat{r})$	T_n: Chebyshev polynomial of degree n
Chebyshev-linear shift	$T_n(2\hat{r} - 1)$	T_n: Chebyshev polynomial of degree n
Chebyshev-quadratic shift	$T_n(2\hat{r}^2 - 1)$	T_n: Chebyshev polynomial of degree n
Jacobi	$\hat{r}^j P^{0,j}_{\frac{n-j}{2}}(2\hat{r}^2 - 1)$	$P^{0,j}_h$: Jacobi polynomial of degree h and order $(0, j)$

Table 2.2: Main families of eigenfunctions ξ_n for a 1D Sturm-Liouville eigenvalue problem in the radial coordinate. Note that the ordering of such basis may depend on the ordering of the angular basis.

The explicit definition of the basis functions $\{\xi_n\}$ in (2.8) may vary, depending on the choices of the functions s and q in the operator $\mathscr{L}$ for the radial component [56, 44]. Hereafter we focus on two possible choices explored in the sequel. Different options from [35, 37] are shown in Table 2.2 and discussed in Appendix 7.1, motivating their exclusion.

- *Shifted-Chebyshev polynomials with quadratic argument:* $T_n(2\hat{r}^2 - 1)$.

  In order to satisfy the Parity Theorem, the 2D basis functions are chosen so that the radial part is $\xi_n(\hat{r}) = T_n(2\hat{r}^2 - 1)$ if the angular index j is even, $\xi_n(\hat{r}) = \hat{r}\ T_n(2\hat{r}^2 - 1)$ otherwise. In fact, from the properties of Chebyshev polynomials, it holds $T_n(2\hat{r}^2 - 1) = T_{2n}(\hat{r})$, with $\hat{r} \in (0, 1)$. As shown in [35], the convergence rate associated with this basis is higher with respect to the convergence guaranteed by the linear-shifted polynomials described in Appendix 7.1.

- *Zernike polynomials aka One-sided Jacobi basis:* $\hat{r}^j P^{0,j}_{\frac{n-j}{2}}(2\hat{r}^2-1)$.

  This basis set is represented by Jacobi polynomials, scaled by the factor $\hat{r}^j$ (see Table 2.2). Thanks to the orthogonality constraint, these polynomials oscillate mostly near $\hat{r}=1$ and consequently the roots move closer and closer to the outer boundary for a fixed degree and by increasing j, allowing for longer timesteps [35].

Differently from the Top-Down case, there is no general approach for the construction of a Bottom-Up vector basis. In what follows, the spectral expansion (2.9) employed for scalar functions is adopted to represent each component of the vector function considered.

Enforcement of the boundary conditions. Depending on the selected radial basis $\{\xi_n\}$, some suitable manipulations may be required to enforce the boundary conditions, as they are, in general, not fulfilled by the basis functions. This can be done via a linear combination of basis functions to provide a new basis set that includes the boundary conditions. Such an approach is adopted in [19], where an "educated" basis that employs parity-restricted Chebyshev polynomials is proposed. For all $j, n = 1, 2, \ldots$, it reads

$$\begin{cases} \xi_n^E(r) = T_n(2r^2-1) - 1 & \text{if } j \text{ is even} \\ \xi_n^O(r) = rT_n(2r^2-1) - r & \text{if } j \text{ is odd.} \end{cases} \tag{2.10}$$

Another, multiplicative approach for homogeneous Dirichlet conditions consists of multiplying the basis functions by the factor $1-r^2$, that enforces the conditions without affecting the parity of the basis functions [21, 117]. This technique is adopted here for Zernike polynomials, since they feature the correct parity by construction.

2.3 Scalar Advection-Diffusion-Reaction Problems

We apply the HiMod technique to the following linear ADR problem:

$$\begin{cases} -\nabla \cdot (\mu \nabla u) + \mathbf{b} \cdot \nabla u + \sigma u = f & \text{in } \Omega \\ u = u_{in} & \text{on } \Gamma_{in} \\ \mu \nabla u \cdot \mathbf{n} = 0 & \text{on } \Gamma_{out} \\ u = 0 & \text{on } \Gamma_{lat}, \end{cases} \tag{2.11}$$

where Ω is a cylinder of length L_x and radius R, $\mathbf{n}$ is the outward unit normal vector, and the boundary is labeled according to Figure 2.2. Let $\mu, \sigma \in L^\infty(\Omega)$, with $\mu \geq \mu_0 > 0$ a.e. in Ω, be the diffusivity and the reaction coefficient, respectively, and $\mathbf{b} = (b_x, b_r, b_\vartheta)^T \in [L^\infty(\Omega)]^3$ the convective field. We assume $\nabla \cdot \mathbf{b} \in L^\infty(\Omega)$ and $-\frac{1}{2}\nabla \cdot \mathbf{b} + \sigma \geq 0$ a.e. in Ω, and $f \in L^2(\Omega)$ so to guarantee the well-posedness of the weak form of the problem, as follows from the Lax-Milgram lemma.

We set the problem on the space V_m defined in (2.2). Hypotheses of *conformity* and *spectral approximability* are required to guarantee the well-posedness of the reduced problem and the convergence to the full problem [143].

We introduce a (uniform) partition $\mathcal{T}_h$ of Ω_{1D} and the corresponding finite element space $V_{1D}^h \subset V_{1D}$, with $\dim(V_{1D}^h) = N_h$ and basis $\{\zeta_l\}_{l=1}^{N_h}$, such that a standard density hypothesis of V_{1D}^h in V_{1D} is guaranteed. The discrete counterpart of (2.2) reads

$$V_m^h = \left\{ v_m^h(x, \mathbf{y}) = \sum_{k=1}^{m} \tilde{v}_k^h(x) \hat{\varphi}_k(\psi_x(\mathbf{y})), \text{ with } \tilde{v}_k^h \in V_{1D}^h,\ \hat{\varphi}_k \in V_{\hat{\gamma}},\ x \in \Omega_{1D},\ \mathbf{y} \in \gamma_x \right\}. \tag{2.12}$$

Then, the HiMod approximation $u_m^h \in V_m^h$ for (2.11) and the corresponding test function $v_m^h \in V_m^h$ reads as $u_m^h(x, \mathbf{y}) = \sum_{k=1}^{m} \sum_{i=1}^{N_h} u_{k,i} \zeta_i(x) \hat{\varphi}_k(\psi_x(\mathbf{y}))$ and $v_m^h(x, \mathbf{y}) = \zeta_l(x) \hat{\varphi}_j(\psi_x(\mathbf{y}))$, respectively, for any $l = 1, \ldots, N_h$ and any $j = 1, \ldots, m$. Thus, we formulate the following problem:

For $k = 1, \ldots, m$ and $i = 1, \ldots, N_h$, find $u_{k,i} \in \mathbb{R}$ such that, for any $j = 1, \ldots, m$,

$l = 1, \dots, N_h$

$$\sum_{k=1}^{m} \sum_{i=1}^{N_h} \int_{\hat{\Omega}_{1D}} \left\{ \mathfrak{d}_{kj}\ \zeta_i' \zeta_l' + \mathfrak{c}_{kj}\ \zeta_i' \zeta_l + \mathfrak{b}_{kj}\ \zeta_i \zeta_l' + \mathfrak{a}_{kj}\ \zeta_i \zeta_l \right\} u_{k,i} d\hat{x} = \int_{\hat{\Omega}_{1D}} \mathfrak{f}_j \zeta_l d\hat{x}, \qquad (2.13)$$

where coefficients $\{\mathfrak{a}, \mathfrak{b}, \mathfrak{c}, \mathfrak{d}\}_{kj}$, $\{\mathfrak{f}\}_j$ collect the contribution of the dynamics transverse to $\hat{\Omega}_{1D}$ (we refer to Appendix 7.2 for an explicit expression of such coefficients). Problem (2.13) represents a linear system of m coupled 1D problems, characterized by an $mN_h \times mN_h$ block matrix A (with a hierarchical structure that we point out in Section 2.4). The indices j and k, associated with the modes, identify the macrostructure of A (they run on the block rows and block columns, respectively), whereas l and i, related to the finite element basis, identify rows and columns of each block, respectively. In this way, each $N_h \times N_h$ block A_{jk} preserves the sparsity pattern peculiar of the adopted 1D finite element approximation.

For a three-dimensional problem set on a cylindrical domain discretized with a structured grid via P1-FE, the degrees of freedom are $\mathcal{O}(N_x N_r N_\vartheta)$, being N_x, N_r and N_ϑ the number of degrees of freedom in the x, r, and ϑ direction, respectively. With a HiMod reduction the number of unknowns is $\mathcal{O}(mN_x)$. Thus, for $m \ll N_r N_\vartheta$, the resulting matrix is much smaller compared to the one associated with the full problem with a significant reduction of the computational costs [127].

2.3.1 Numerical Assessment

In order to validate the method, we test HiMod on the basis functions (2.5) obtained with the Top-Down approach for different types of lateral boundary conditions, and on functions (2.8) with homogeneous Dirichlet lateral boundary conditions. We compare the numerical properties of the different types of basis functions when approximating axisymmetric and non-axisymmetric solutions, highlighting strengths and weaknesses of each option.

We present a set of numerical experiments where the analytical solution is known to investigate the trend of the error, and a more realistic test where the analytical

solution is not available. In the latter case, the ground truth solution is provided by the standard $\mathbb{P}^1$-Finite Element method (FEM) computed on a very fine grid. All the numerical experiments have been performed using a C++ solver based on the Finite Element Library LifeV [4], on a Dell Inspiron 15R SE 7520 equipped with a 2.10GHz Intel Core i7 processor and 8GB of RAM. More details are available in [8].

Convergence analysis: The axisymmetric case

Consider problem (2.11) with $L_x = 5$, $R = 1$, $\mu = 1$, $b_x = 5$, $b_r = 0$, $b_\vartheta = 0$, $\sigma = 10$. The forcing term and the boundary conditions on Γ_{in} are adjusted to match the following analytical solutions:

$$u_{R2}(x, r, \vartheta) = (R^2 - r^2)(L_x - x)^2, \tag{2.14a}$$

$$u_{R3}(x, r, \vartheta) = -\frac{1}{4}r^2 + \frac{1}{6}r^3 + \frac{1}{12}, \tag{2.14b}$$

$$u_{R4}(x, r, \vartheta) = (R^2 - r^2)^2(L_x - x)^2, \tag{2.14c}$$

fulfilling homogeneous Dirichlet, Neumann, and Robin lateral boundary conditions, respectively.

Top-Down basis. The boundary conditions on Γ_{lat} in (2.11) can be generalized via the homogeneous Robin condition $\mu\nabla u \cdot \mathbf{n} + \chi u = 0$. For $(\mu, \chi) = (0, 1)$, and for $(\mu, \chi) = (1, 0)$, the constraint reduces to a Dirichlet and Neumann boundary condition, respectively. Each case is discussed hereafter.

- *Dirichlet BC* ($\mu = 0$, $\chi = 1$).

  Consider the exact solution (2.14a), fulfilling homogeneous lateral Dirichlet boundary conditions. Figure 2.3 (left) shows the trend of the $L^2(\Omega)-$norm of the HiMod relative error, for different values of h and m. For the largest values of h the error stagnates, since the FEM error dominates the total error. Conversely, for small values of h, the error is dominated by the modal component.

The convergence rate is linear with respect to the reciprocal of the number of modes, which is consistent with the results in [9] on slabs.

HiMod is sensibly competitive with respect to the standard FEM in terms of accuracy and efficiency. For a given number of degrees of freedom (DOF), the HiMod error is consistently about one order of magnitude lower than the FEM error (Figure 2.3, right). Conversely, the number of DOF to obtain a desired tolerance is consistently smaller for the HiMod method than for FEM. Note that, for $\mathbb{P}^1$-FEM, the number of DOF coincides with the number of vertices of the mesh, whereas for HiMod it is given by the number of FE nodes along Ω_{1D} multiplied by the number of modes. This translates into a considerable reduction of the computational time, validating HiMod as a viable strategy for replacing purely 1D solvers, with a competitive computational cost.

- *Robin* ($\mu = 1$, $\chi = 1$), *Neumann BC* ($\mu = 1$, $\chi = 0$).

  For $\mu = 1$, $\chi = 1$ an analytical solution is provided by (2.14c). Figure 2.4 (right) shows that the HiMod error decays as $O(N^{-1.5})$, being N the number of DOF. The same happens for $\chi = 0$ (Neumann), with exact solution as in (2.14b) (Figure 2.4, left). Indeed, as noted in [9] for a slab, in the particular case of Neumann boundary data, an infinitely regular function whose odd derivatives vanish at the boundary is approximated by Fourier truncated series with spectral accuracy. This justifies the superconvergent trend. The gain yielded by HiMod with respect to standard FEM is evident also for these BC.

Bottom-Up basis. Since for our current technology only Dirichlet boundary conditions can be enforced with a Bottom-Up basis, parity-restricted Chebyshev and Zernike polynomials are used to approximate solution (2.14a). We order functions (2.10) as in Table 2.3 (see [19]), and we use a rectangular truncation such that $m = 2N_rN_\vartheta$, being N_r and N_ϑ the number of radial and angular modes, respectively. The factor 2 takes into account the sinusoidal and cosinusoidal contribution in the

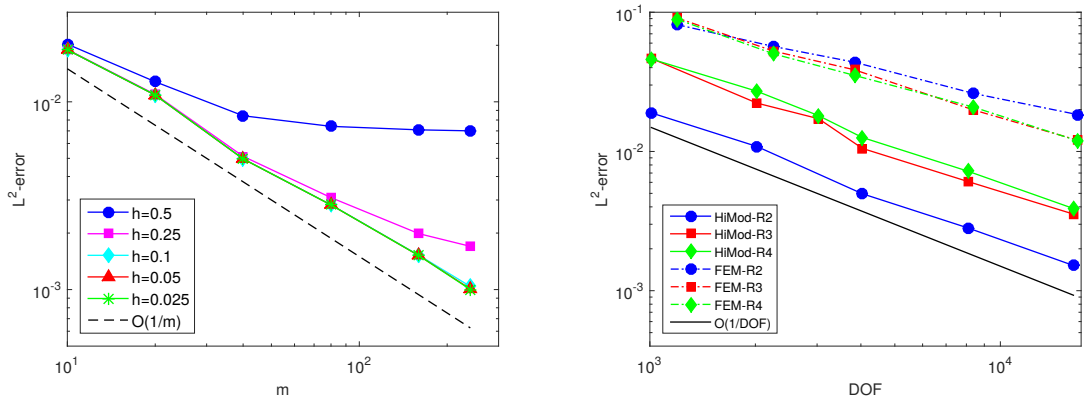


Figure 2.3: ADR with lateral Dirichlet BC: $L^2(\Omega)$-norm of the HiMod relative error as a function of h and m (left); Comparison between HiMod and FEM error for different exact solutions (right).

modal expansion. Differently, the ordering of the Zernike basis stems from the pyramidal structure of the basis, by construction. As a result, while the former basis allows for more flexibility in the distribution of the radial modes across the angular frequencies, the latter guarantees more robustness, especially when information about the solution of the problem is not known a priori.

Since the solution is axisymmetric, for the rectangular truncation of the parity-restricted Chebyshev basis we set $N_\vartheta = 1$ and $N_r = m/2$. The values of the relative error for $m = 10$ are reported in Table 2.4, for different values of h and the corresponding number of DOF. The relative error is super-convergent, decaying slightly faster than $O(h^2)$, and it is roughly constant as m increases. In fact, the radial component belongs to the discrete space, and the error is dominated by the FEM discretization. The same considerations hold for the Zernike basis, that is equivalent in terms of numerical performance.

k odd				
$\cos(0\vartheta)$	$\xi_1^E(r)$ **(1)**	$\xi_2^E(r)$ **(3)**	$\xi_3^E(r)$ **(7)**	$\xi_4^E(r)$ **(13)**
$\cos(1\vartheta)$	$\xi_1^O(r)$ **(5)**	$\xi_2^O(r)$ **(9)**	$\xi_3^O(r)$ **(15)**	...
$\cos(2\vartheta)$	$\xi_1^E(r)$ **(11)**	$\xi_2^E(r)$ **(17)**	...	
$\cos(3\vartheta)$	$\xi_1^O(r)$ **(19)**	...		

k even				
$\sin(1\vartheta)$	$\xi_1^O(r)$ **(2)**	$\xi_2^O(r)$ **(4)**	$\xi_3^O(r)$ **(8)**	$\xi_4^O(r)$ **(14)**
$\sin(2\vartheta)$	$\xi_1^E(r)$ **(6)**	$\xi_2^E(r)$ **(10)**	$\xi_3^E(r)$ **(16)**	...
$\sin(3\vartheta)$	$\xi_1^O(r)$ **(12)**	$\xi_2^O(r)$ **(18)**	...	
$\sin(4\vartheta)$	$\xi_1^E(r)$ **(20)**	...		

Table 2.3: Ordering of the Bottom-Up basis functions. The numbering in bold refers to a triangular truncation, whereas a rectangular truncation employs the same number of radial basis functions for each angular frequency.

h	DOF	Relative error
0.5	110	6.894535e-03
0.25	210	1.263055e-03
0.1	510	1.338110e-04
0.05	1010	2.428658e-05
0.025	2010	4.384685e-06

Table 2.4: ADR in a cylinder: relative error associated with the (Chebyshev) Bottom-Up basis for $m = 10$ and different values of the FEM mesh size h.

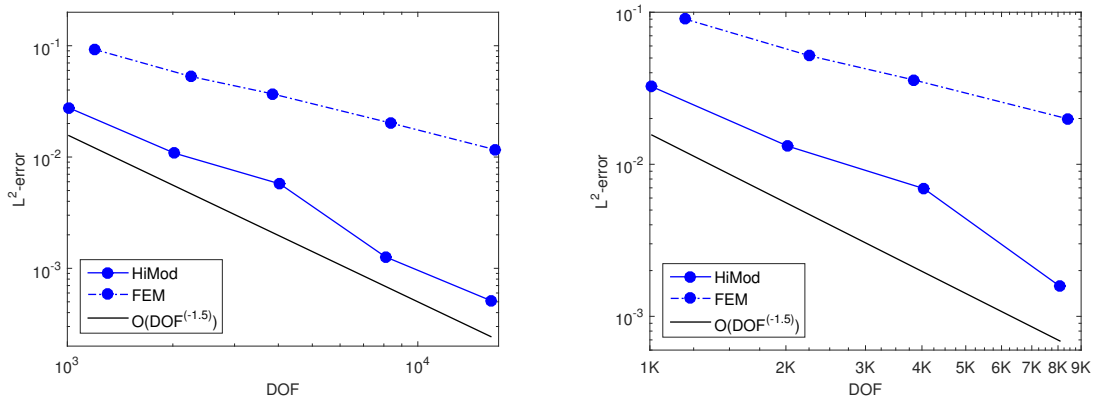


Figure 2.4: ADR with Neumann (left) and Robin (right) lateral BC: $L^2(\Omega)$-norm of the HiMod relative error as a function of the number of DOF.

Convergence analysis: The non-axisymmetric case

Consider problem (2.11) with $L_x = 5$, $R = 1$, $\mu = 1$, $b_x = 5$, $b_r = 0$, $b_\vartheta = 0$, $\sigma = 0$, and the non-axisymmetric solution

$$u(x, r, \vartheta) = r^2 \left(e^{R^2 - r^2} - 1 \right) (L_x - x) e^{\sin^2(\vartheta)}. \tag{2.15}$$

The relative errors with respect to the $L^2(\Omega)-$norm for the Top-Down and Bottom-Up basis for $h = 0.05$ and different numbers of modes are compared in Table 2.5. Note that, for the Chebyshev Bottom-Up basis, more factorizations for $m = 2N_r N_\vartheta$ may be feasible. In such a case, the two values for $N_r < N_\vartheta$ and $N_r > N_\vartheta$ are reported.

The Top-Down basis features slow convergence, and the ill-conditioning leads to the stagnation of the error. Nevertheless, for small m this option may still be viable, especially to benefit from more flexibility in the choice of the lateral boundary conditions.

For the Bottom-Up approach, as already noted in [37], the Zernike basis outperforms the parity-restricted Chebyshev basis for small/moderate m, whereas the latter converges faster for large values of m. Nevertheless, the order of magnitude of the

	Bottom-Up			Top-Down
m	Zernike	Chebyshev ($N_\vartheta \leq N_r$)	Chebyshev ($N_\vartheta \geq N_r$)	Bessel
8	2.295050e-01	3.946134e-01		4.226621e-01
16	3.417806e-02	3.267148e-01	2.268239e-01	3.627250e-01
18	3.417806e-02	4.848936e-02		3.373612e-01
24	3.417806e-02	4.100592e-02	4.848936e-02	3.401257e-01
32	1.327025e-02	4.100592e-02		3.369967e-01
40	1.327025e-02	4.092920e-02	8.566436e-03	3.487149e-01
48	9.647994e-03	4.092733e-02	8.566436e-03	3.487152e-01
50	9.647994e-03	8.191322e-03		3.487607e-01
56	9.647994e-03	4.092724e-02	7.895026e-03	3.517681e-01

Table 2.5: ADR in a cylinder: $L^2(\Omega)-$norm of the relative error associated with the Bottom-Up and Top-Down basis for a non-axisymmetric solution.

error for the two bases is comparable with few or many modal functions. The main remarkable difference between these two bases lies in regularity. Zernike polynomials satisfy all the regularity conditions by construction, yielding approximate solutions that are infinitely differentiable everywhere. Differently, as well explained in [121] for the approximation of the solution to the Bessel equation, with parity-restricted Chebyshev series "the solution is never formally regular, there is no indication of how bad the nonregularity actually is".

Drug release modeling

We aim at modeling a drug-eluting stent deployed in a blood vessel. Stents are medical devices that are used in the surgical treatment of constricted arteries. They are inserted into the vessel in order to expand the lumen to prevent or alleviate an obstruction. In particular, drug-eluting stents slowly release a drug to inhibit cell proliferation, so to avoid the so-called vascular remodeling [55].

This test case describes the effects of an advective field on a local source term. Consider problem (2.11) on the same domain as in the previous test, with $u_{in} = R^2 - r^2$ and a homogeneous Neumann condition on Γ_{out}. The physical parameters are set to $\mu = 1$, $\sigma = 0$, $b_x = 5$, $b_r = 0$, $b_\vartheta = 0$, while the forcing term is designed to model the presence of a high concentration c of drug in proximity of the wall, i.e.,

$$f(x, r, \vartheta) = c\, \mathbb{1}_{\left[\frac{L_x}{7}, \frac{L_x}{3}\right]}(x) \cdot \mathbb{1}_{\left[\frac{7}{10}R, R\right]}(r), \tag{2.16}$$

where $\mathbb{1}_{[a,b]}$ denotes the characteristic function associated with the generic interval $[a, b]$. The concentration of drug and the mesh size along Ω_{1D} are set to $c = 10$ and $h = 0.05$, respectively. Since this test case has no analytical solution, we refer to a FE approximation computed on a fine mesh of approximately $97K$ nodes. The solution obtained with the Top-Down and Bottom-Up bases for $m = \{10, 20, 40\}$, and corresponding DOF $\in \{1010, 2020, 4040\}$, is shown in Figure 2.5. Since the problem is axisymmetric, the number of angular functions for the parity-restricted Chebyshev basis is set to $N_\vartheta = 1$, so to enrich the approximation of the radial component of the solution as m increases.

As proved in the convergence tests, the flexibility in the choice of the factorization of the number of parity-restricted Chebyshev modal functions leads to a fast convergence of the HiMod solution (Figure 2.5, center). Differently, the Zernike basis is slowly convergent (Figure 2.5, right) due to the presence of angular modes associated with high frequencies, that do not bring any contribution to the approximation of the solution (Figure 2.6). Finally, ill-conditioning is responsible for the slow convergence of the Top-Down basis (Figure 2.5, left). In any case, even with a relatively large number of modes and independently of the choice of the basis, HiMod achieves the same level of accuracy as FEM with approximately 1% of the DOF ($O(1K)$ vs. $O(100K)$).

On balance, the tests presented here for the scalar case show that, in general, the Bottom-Up basis converges faster than the Top-Down basis, which suffers from ill-conditioning. Nevertheless, the latter allows to include any type of boundary

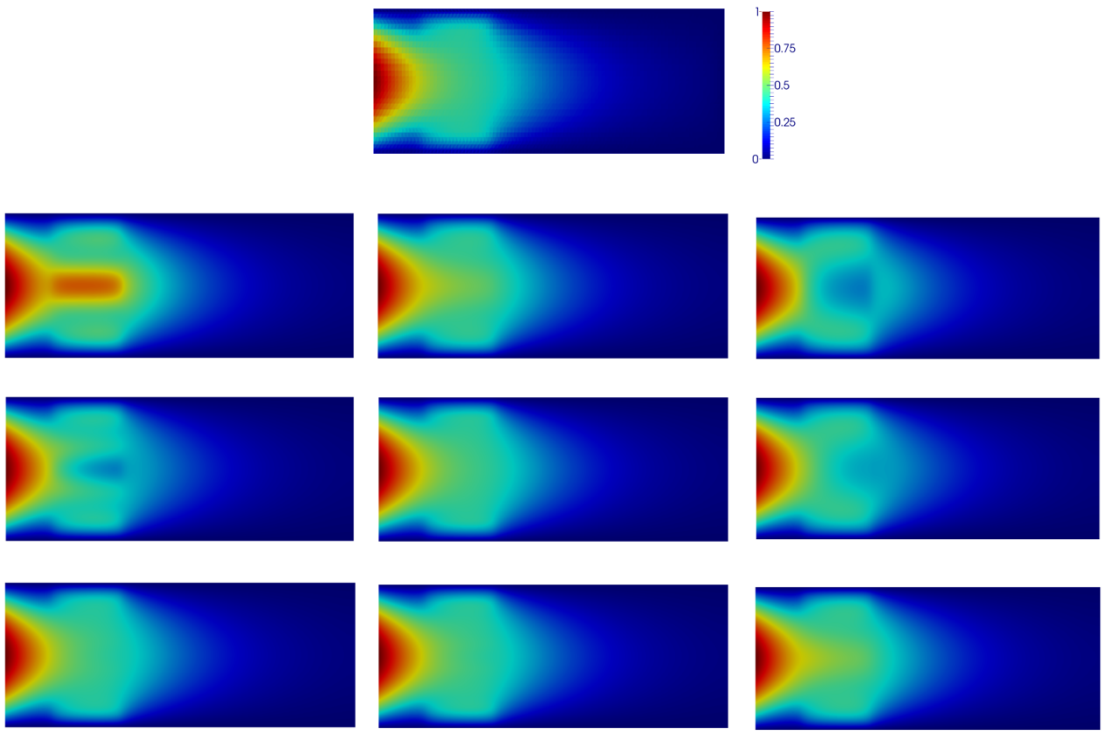


Figure 2.5: Drug-release modeling: longitudinal section along the xy-plane of FEM (top) and HiMod solution with $m = 10$ (second row), $m = 20$ (third row), $m = 40$ (bottom) for Top-Down (left), parity-restricted Chebyshev (center), and Zernike basis (right).

conditions and the background of the SLE theory guarantees that the approach is accurate enough for practical applications, confirming the excellent results for slab domains provided in [9]. We plan to use this approach for patients specific coronary geometries with bioabsorbable stents, to model the elution process [86].

If geometric or analytical features of the solution are known a priori (as in the axisymmetric case), the parity-restricted Chebyshev basis is extremely efficient, thanks to the factorization of the spectral size, that allows to tailor the enrichment of the basis to the radial or angular components, separately. However, the regularity of the

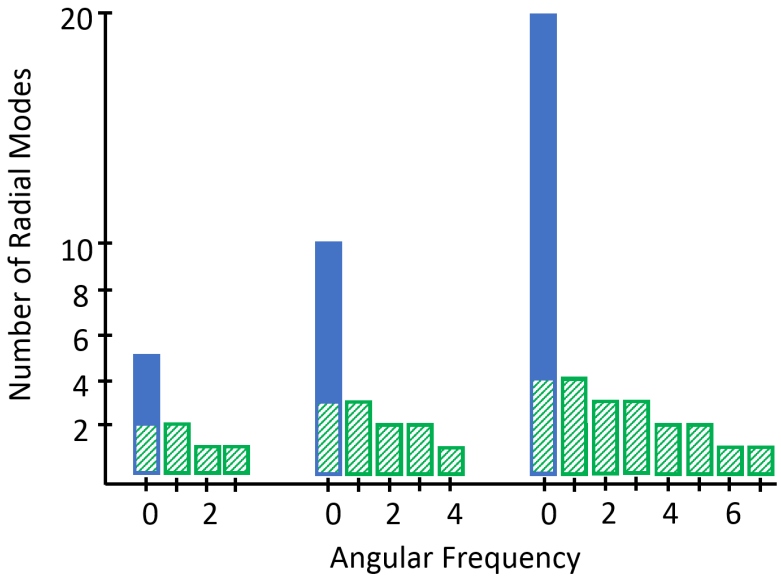


Figure 2.6: Distribution of the radial modes (y-axis) across cosinusoidal basis functions of different frequency (x-axis), for parity-restricted Chebyshev (filled) and Zernike (dashed) polynomials, with $m = 10$ (left), $m = 20$ (center), $m = 40$ (right).

approximation at the origin, as noted in [121], is not guaranteed, especially in the non-axisymmetric case.

When no information about the solution is available a priori, the Zernike basis is the most robust choice, guaranteeing convergence and regularity of the solution.

2.4 The Navier-Stokes equations in cylindrical coordinates

We extend the HiMod procedure to the generalized incompressible Navier-Stokes equations (NSE)

$$\begin{cases} -\nabla \cdot \big(2\nu\mathbb{D}(\mathbf{u})\big) + (\mathbf{u}\cdot\nabla)\,\mathbf{u} + \alpha\mathbf{u} + \nabla p = \mathbf{f} & \text{in } \Omega \\ \nabla \cdot \mathbf{u} = 0 & \text{in } \Omega \\ \mathbf{u} = \mathbf{0} & \text{on } \Gamma_{lat} \\ \mathbf{u} = \mathbf{u}_0 & \text{on } \Gamma_{in} \\ (2\nu\mathbb{D} - p\mathbb{I})\mathbf{n}\cdot\mathbf{n} = 0 & \text{on } \Gamma_{out}, \end{cases} \tag{2.17}$$

where $\alpha \geq 0$ and the kinematic viscosity $\nu > 0$ are given constants, $\mathbf{f} : \Omega \to \mathbb{R}^3$ is a given force per unit mass, $\mathbf{u} = [u_x, u_r, u_\vartheta] : \Omega \to \mathbb{R}^3$ and $p : \Omega \to \mathbb{R}$ are the velocity field and the kinetic pressure, respectively, $\mathbb{D}(\mathbf{u})$ is the strain velocity tensor, $\mathbb{I}$ is the identity tensor, $\mathbf{u}_0$ is a given inflow profile, and $\mathbf{n}$ is the outward unit normal vector. These equations can be regarded as the result of the time-discretization of the unsteady counterpart with $\alpha = O(\Delta t^{-1})$, being Δt the time step. For $\alpha = 0$ we recover the steady problem. We consider homogeneous Dirichlet conditions on the lateral boundary Γ_{lat}, and Dirichlet and Neumann conditions on Γ_{in} and Γ_{out}, respectively. We will denote by Γ_{dir} the whole portion of the boundary where Dirichlet conditions are enforced.

2.4.1 The HiMod formulation

The weak form of problem (2.17) involves the Sobolev spaces $V = [H^1_{\Gamma_{dir}}(\Omega)]^3$, $Q = L^2(\Omega)$, where $H^1_{\Gamma_{dir}}(\Omega)$ denotes the set of the functions in $H^1(\Omega)$ that fulfil Dirichlet conditions on Γ_{dir}. We introduce a modal basis $\{\hat{\boldsymbol{\varphi}}_{u,k}\}_{k\in\mathbb{N}^+} \subset [H^1_0(\hat{\gamma})]^3$ and $\{\hat{\varphi}_{p,s}\}_{s\in\mathbb{N}^+} \subset L^2(\hat{\gamma})$ for the velocity and for the pressure, respectively, to define the

HiMod reduced spaces

$$V_{m_u} = \left\{ \mathbf{v}_{m_u}(x,\mathbf{y}) = \sum_{k=1}^{m_u} \tilde{v}_k(x)\hat{\boldsymbol{\varphi}}_{u,k}(\psi_x(\mathbf{y})),\ \text{with } \tilde{v}_k \in V_{1D,u},\ \hat{\boldsymbol{\varphi}}_{u,k} \in V_{\hat{\gamma},u},\ x \in \Omega_{1D} \right\},$$
$$Q_{m_p} = \left\{ q_{m_p}(x,\mathbf{y}) = \sum_{s=1}^{m_p} \tilde{q}_s(x)\hat{\varphi}_{p,s}(\psi_x(\mathbf{y})),\ \text{with } \tilde{q}_s \in V_{1D,p},\ \hat{\varphi}_{p,s} \in V_{\hat{\gamma},p},\ x \in \Omega_{1D} \right\},$$

where m_u and m_p denotes the number of modes related to the velocity and the pressure, respectively, $V_{1D,u} \subseteq [H^1_{\Gamma_{dir}}(\Omega_{1D})]^3$ and $V_{1D,p} \subseteq L^2(\Omega_{1D})$ are the 1D spaces associated with the supporting fiber Ω_{1D} for the velocity and the pressure, respectively, and with $V_{\hat{\gamma},u} = \text{span}\{\hat{\boldsymbol{\varphi}}_{u,k}\}$, $V_{\hat{\gamma},p} = \text{span}\{\hat{\varphi}_{p,s}\}$.

We introduce a uniform 1D grid $\mathscr{T}_h$ on Ω_{1D} and we associate with this partition the FE spaces $V^h_{1D,u}$, with $\dim(V^h_{1D,u}) = N_{h,u}$ and basis $\{\boldsymbol{\zeta}_{u,l} = [\zeta_{x,l}, \zeta_{r,l}, \zeta_{\vartheta,l}]^T\}_{l=1}^{N_{h,u}}$, and $V^h_{1D,p}$, with $\dim(V^h_{1D,p}) = N_{h,p}$ and basis $\{\zeta_{p,l}\}_{l=1}^{N_{h,p}}$.

Concerning the choice of the modal basis, it is a priori possible to use a different number of modes, m_x, m_r, m_ϑ, for the three components of the velocity, with corresponding bases $\{\hat{\varphi}_{x,k}\}$, $\{\hat{\varphi}_{r,k}\}$, $\{\hat{\varphi}_{\vartheta,k}\}$. For the sake of simplicity, we assume $m_x = m_r = m_\vartheta = m_u$ and $\hat{\varphi}_{x,k} = \hat{\varphi}_{r,k} = \hat{\varphi}_{\vartheta,k} = \hat{\varphi}_{u,k}$. Analogously, we set $\zeta_{x,l} = \zeta_{r,l} = \zeta_{\vartheta,l} = \zeta_{u,l}$. Thus, the HiMod velocity and pressure are expanded as

$$\begin{aligned} \mathbf{u}^h_{m_u}(x,\mathbf{y}) &= \sum_{k=1}^{m_u}\sum_{i=1}^{N_{h,u}} \left[u_{x,k,i},\ u_{r,k,i},\ u_{\vartheta,k,i}\right]^T \hat{\varphi}_{u,k}(\psi_x(\mathbf{y}))\zeta_{u,i}(x), \\ p^h_{m_p}(x,\mathbf{y}) &= \sum_{w=1}^{m_p}\sum_{s=1}^{N_{h,p}} p_{w,s}\hat{\varphi}_{p,w}(\psi_x(\mathbf{y}))\zeta_{p,s}(x), \end{aligned} \tag{2.18}$$

while the test functions are defined as $\mathbf{v}^h_{m_u} = [\zeta_{u,b}\hat{\varphi}_{u,c}, \zeta_{u,b}\hat{\varphi}_{u,c}, \zeta_{u,b}\hat{\varphi}_{u,c}]^T$ for $b = 1,\dots,N_{h,u}$ and $c = 1,\dots,m_u$, $q^h_{m_p} = \zeta_{p,l}\hat{\varphi}_{p,j}$ for $l = 1,\dots,N_{h,p}$ and $j = 1,\dots,m_p$. We refer to Appendix 7.3 for the explicit HiMod formulation of problem (2.17).

Algebraic and implementation aspects. After the HiMod discretization of the generalized Navier-Stokes problem, the resulting matrix has a block structure with the following partitioning of degrees of freedom: $m_u N_{h,u}$ DOFs are associated with

each component u_x, u_r and u_ϑ of the velocity, whereas the remaining $m_p N_{h,p}$ degrees of freedom are related to the pressure. The outer block-structure couples the components of the velocity and the pressure, as it is standard for Navier-Stokes equations. Then, each macro-block shares the typical block-wise pattern of the HiMod reduction applied to a scalar problem (Figure 2.7).

From the implementation viewpoint, the assembly of the matrix is a bottleneck. In fact, for each pair of modes a full 3D problem has to be reduced to a 1D problem by integration of the weak formulation. However, this can be done in parallel by exploiting the block structure of the matrix. At the coarser level we parallelize the component blocks with openMP `sections`. Then, within each section, the execution of the loop on the modal functions can be accelerated via a `#pragma omp for` directive, since all the modal blocks are independent. The parallelization of the loops on the FE nodes is not efficient, because typically the FE blocks are much smaller than the macro-structure of the matrix, and the overhead associated with the creation and deletion of threads worsens the performance.

In the implementation, the innermost loop assembles the FE component, i.e., each 1D problem solves for one mode on $\hat{\Omega}_{1D}$. Another choice stems from swapping the order of assembly, so that each block provides the solution for all the modes at one FE grid point (see Figure 2.8). Although the two strategies of assembly are in principle equivalent, the efficiency of the solver may vary, especially with a view to a parallel implementation of the method. This is an issue that is still under investigation [23].

2.4.2 The inf-sup condition

As well known, the choice of finite dimensional spaces for velocity and pressure in (2.17) must fulfill the *inf-sup condition* [41, 34]. While the issue is largely investigated for finite element and spectral methods [38, 68, 153, 44], we are not aware of any theoretical result for hybrid methods that involve both the techniques. Separation of variables with different discretization techniques is considered for the inf-sup in [22, 149]. A lower bound (L_B) for the inf-sup constant is provided in [48, 25] for

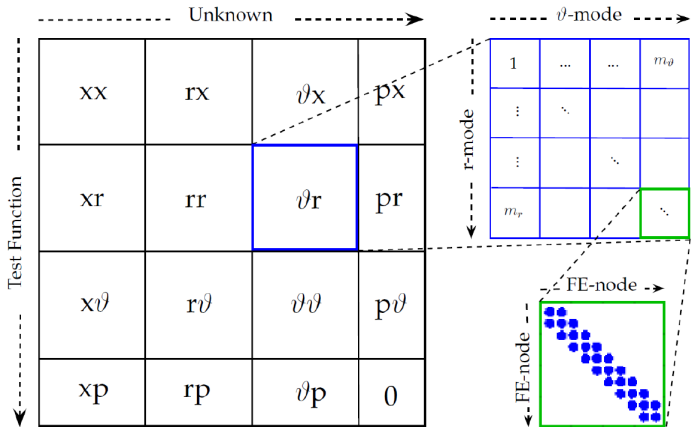


Figure 2.7: Block structure of the HiMod matrix associated with the generalized Navier-Stokes equations. The indices x, r, ϑ, p in each block highlight the coupling of the three components of the velocity with the pressure.

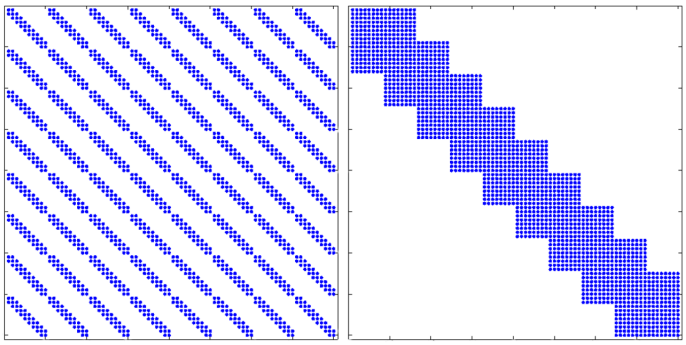


Figure 2.8: Matrix pattern of the HiMod matrix by assembling one FE problem per mode (left) or by solving for all modes per FE (right).

the analysis of incompressible solids and fluids, acoustic fluids, plates and shells, and convective-dominated flows, with mixed finite elements. A sharp lower bound (L_A) for the inf-sup constant for the Stokes problem in a semi-infinite two-dimensional channel discretized with a Laguerre-Legendre spectral method is proved in [22]. Although the present work deals with a three-dimensional setting discretized via a hybrid FE-spectral technique, the behavior of the aforementioned bounds has been tested for different pairs of modal spaces for the velocity and for the pressure. An analytical proof of the fulfilment of the *inf-sup* condition for the HiMod spaces is still a work in progress.

Consider equations (2.17) with $\alpha = 0$ and negligible non-linear term (Stokes flow). Then, the algebraic form of the discrete weak formulation can be expressed synthetically as:

$$\begin{bmatrix} \mathbf{A}_h & \mathbf{B}_h^T \\ \mathbf{B}_h & -\frac{1}{\kappa}\mathbf{T}_h \end{bmatrix} \begin{bmatrix} \mathbf{u}_h \\ \mathbf{p}_h \end{bmatrix} = \begin{bmatrix} \mathbf{f}_h \\ \mathbf{0} \end{bmatrix}, \tag{2.19}$$

where $\kappa \to \infty$, $\mathbf{u}_h, \mathbf{f}_h \in \mathbb{R}^{N_h^u}$, $\mathbf{p}_h \in \mathbb{R}^{N_h^p}$, $\mathbf{A}_h \in \mathbb{R}^{N_h^u \times N_h^u}$, $\mathbf{B}_h \in \mathbb{R}^{N_h^u \times N_h^p}$, and $\mathbf{T}_h \in \mathbb{R}^{N_h^p \times N_h^p}$ for selected discrete spaces $\mathscr{V}_h$, $\mathscr{Q}_h$ for the velocity and the pressure, respectively, with dimension $N_h^u = dim(\mathscr{V}_h)$ and $N_h^p = dim(\mathscr{Q}_h)$. Set $\mathbf{G}_h = \mathbf{B}_h^T \mathbf{T}_h^{-1} \mathbf{B}_h$, and let $\mathbf{S}_h$ be the norm matrix. Then, define the inf-sup constant as

$$\alpha_h = \inf_{\mathbf{w}_h \in \mathscr{Q}_h} \sup_{\mathbf{v}_h \in \mathscr{V}_h} \frac{\mathbf{w}_h^T \mathbf{G}_h \mathbf{v}_h}{\sqrt{\mathbf{w}_h^T \mathbf{G}_h \mathbf{w}_h} \sqrt{\mathbf{v}_h^T \mathbf{S}_h \mathbf{v}_h}}. \tag{2.20}$$

If α_h does not decrease to zero as the discretization is refined (i.e., as $h \to 0$), then the selected discrete spaces are inf-sup stable.

By adopting a heuristic approach based on the theory for the Finite Element and spectral methods (see, e.g., [38, 44]), respectively, we use piecewise quadratic velocity/linear pressure for the axial dependence, and we set $m_u = m_p + 2$ for the transverse one, as in [98]. A different choice has been pursued in [30], with $m_u = 2m_p - 1$. Figure 2.9 (left,center) shows the numerical behavior of the lower bounds L_A and L_B of α_h for the pairs $(m_p + 2, m_p)$ and $(2m_p - 1, m_p)$, respectively. For each choice of the spaces both the lower bounds are roughly constant as the size of the modal spaces

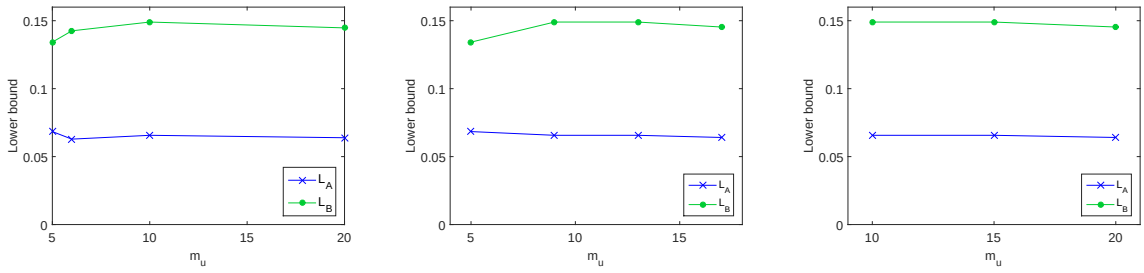


Figure 2.9: Inf-sup test: Lower bounds L_A ($\times$) and L_B ($\circ$) for the velocity/pressure modal spaces of dimensions $(m_p + 2, m_p)$ (left), $(2m_p - 1, m_p)$ (center), and $(m_u, 4)$ (right).

increases. Moreover, by fixing the size of the pressure modal space to $m_p = 4$, L_A and L_B do not decrease as the velocity modal space is enriched (see Figure 2.9 (right)). Therefore, as a preliminary analysis we can conclude that the selected modal spaces are numerically inf-sup stable.

2.4.3 Pole Conditions

The polar reference system is singular at the origin ($r = 0$). Therefore, suitable pole conditions have to be satisfied by a function in polar (cylindrical) coordinates in order to guarantee the regularity of the corresponding function in Cartesian coordinates [35, 89, 167]. The extension of such conditions to the vector case is not straightforward, since vectors are not invariant under the mapping between the two reference systems [29, 159]. For the specific case of the NSE, two types of pole conditions are derived in [122] and recalled below.

Consider the spectral expansion of the 3D velocity field and of the pressure

$$(u_x, u_\pm, p) = \sum_{j=-\infty}^{+\infty} (u_{x,j}, u_{\pm,j}, p_j)\, e^{ij\hat{\vartheta}}, \tag{2.21}$$

where $u_{\pm,j} = u_{r,j} \pm i u_{\vartheta,j}$ (for the sake of simplicity, the axial and radial dependence is suppressed in the notation), and i is the imaginary unit number.

(I) The *essential* pole conditions

$$\begin{aligned} u_{+,0} &= 0 && j = 0, \\ u_{+,1} &= 0 && j = 1, \\ u_{\pm,j} &= u_{x,j} = 0 && j > 1, \\ p_j &= 0 && j \geq 1 \end{aligned} \tag{2.22}$$

are sufficient to ensure the well-posedness of the weak formulation of the NSE [123], as they guarantee the finiteness of the integrals of the bilinear form.

(II) The *natural* pole conditions

$$\begin{aligned} \frac{\partial^k}{\partial r^k} u_{+,j} &= 0 \quad k = 1, \ldots, j, && j \geq 1, \\ \frac{\partial^k}{\partial r^k} u_{-,j} &= 0 \quad k = 1, \ldots, j-2, && j \geq 3, \\ \frac{\partial^k}{\partial r^k} u_{x,j} &= 0 \quad k = 1, \ldots, j-1, && j \geq 2, \end{aligned} \tag{2.23}$$

together with the *parity* conditions, ensure the regularity of the solution at the pole. Note that, due to the coupling of the radial and angular coefficients of a cylindrical vector spectral expansion, the extension of the Parity Theorem to the vector case implies that the modal coefficients of the transverse components have *opposite* parity compared to the scalar case [29, 122, 159]. In particular, $u_{\pm,j}$ has the same parity as $j+1$, and $u_{x,j}$ has the same parity as j.

The construction of a vector basis that satisfies (i) boundary conditions, (ii) well-posedness constraints, and, possibly, (iii) orthogonality and (iv) regularity conditions is challenging. Several methods have been proposed in the literature to simplify or reduce the constraints by decoupling the transverse components via a change of variable (see, e.g., [21, 122]), or by enforcing the constraints via additional terms in the spectral representation (tau-method, [121, 159]). However, applying such techniques to the HiMod formulation requires a considerable effort, related to the implementation of the numerical quadratures for the coefficients of the formulation. We stress here that HiMod is mainly intended to be an improved 1D solver rather than an

alternative to a 3D solver. In this respect, the efforts spent for the construction of a basis fulfilling requirements (i)-(iv) are not in the spirit of the method. The relevant question for our purpose is to what extent the fulfilment of the pole conditions is critical for our solver. This is exactly what we investigate hereafter.

2.4.4 Steady case: Poiseuille flow

We compare the performances of the Top-Down basis (2.5), solution to a 2D Sturm-Liouville eigenvalue problem, as opposed to the family of Bottom-Up basis functions of type (2.8). In both cases the scalar basis functions are used to discretize each component of the velocity. Note that, in general, only the axial component of the Top-Down and of the Zernike vector bases fulfils the essential and natural pole conditions. Nevertheless, when the velocity field is purely axial (as for the Poiseuille flow), the pole conditions on the transverse components are automatically satisfied. For the parity-restricted Chebyshev vector basis, not even the axial component is compliant with (2.22). However, such conditions are only sufficient for the well-posedness of the weak formulation of the NSE, therefore the feasibility of the solution is not compromised a priori.

For the Bottom-Up basis, we present results obtained with Chebyshev polynomials with quadratic shift. Results for the Zernike basis are similar.

Let Ω be a cylindrical domain with radius $R = 0.5$ and length $L_x = 6$. For the sake of the comparison between the two approaches, we enforce homogeneous Dirichlet boundary conditions on Γ_{lat}, and we solve the generalized Stokes equations dropping the non-linear term in (2.17), completed with the following boundary conditions:

$$\begin{cases} u_x = \dfrac{5}{L_x}\dfrac{(R^2 - r^2)}{4\mu} & \text{on } \Gamma_{in} \\ u_r = u_\vartheta = 0 & \text{on } \Gamma_{in} \\ (2\nu\mathbb{D} - p\mathbb{I})\mathbf{n}\cdot\mathbf{n} = 0 & \text{on } \Gamma_{out} \\ u_r = u_\vartheta = 0 & \text{on } \Gamma_{out}, \end{cases} \tag{2.24}$$

so that, for $\alpha = 0$ (steady case), we have the classical Poiseuille flow (see, e.g., [64]). The HiMod discretization here employed selects $m_u = 10$, $m_p = 8$ and a uniform mesh along Ω_{1D} of size $h = 0.125$.

Top-down basis

As shown in Figure 2.10 (top-left) the peak reached by the axial component of the HiMod velocity does not match the analytical profile. The pressure gradient, although constant as expected, is underestimated (Figure 2.10, bottom-left). Nevertheless, the accuracy of the approximation can be enhanced by further increasing the number of modes. Figure 2.11 (top) shows the trend of the $L^2(\Omega)$-norm of the relative error associated with the HiMod velocity and pressure as a function of the modal index, for different values of h. In this case, a refinement of the mesh does not provide any improvement in the accuracy of the HiMod solution since both the velocity and the pressure belong to the discrete space associated with Ω_{1D}. Hence, the global error is dominated by the modal error, which drops as the number of modal functions increases. Therefore, Bessel functions are inefficient in the spirit of a HiMod reduction, where few transverse modes are expected to ensure an accurate approximation. Finally, it is worth noting that the number of degrees of freedom necessary to guarantee a fixed level of accuracy is consistently smaller for HiMod than for FEM or, equivalently, for a fixed size of the problem, HiMod provides a more accurate solution than FEM (Figure 2.11, bottom).

Bottom-up basis

In order to enforce the lateral boundary conditions, the basis functions (2.10) are employed to represent each component of the velocity field. As for the pressure, standard Chebyshev polynomials with quadratic shift are used, as no boundary condition is enforced on p. We employ $m_p = 2N_rN_\vartheta$ and $m_u = 2(N_r + 1)N_\vartheta$ modes for the pressure and for each velocity component, respectively. We set $N_\vartheta = 1$, being the problem axisymmetric. Note that, in this case, it holds $m_u = m_p + 2$, which is

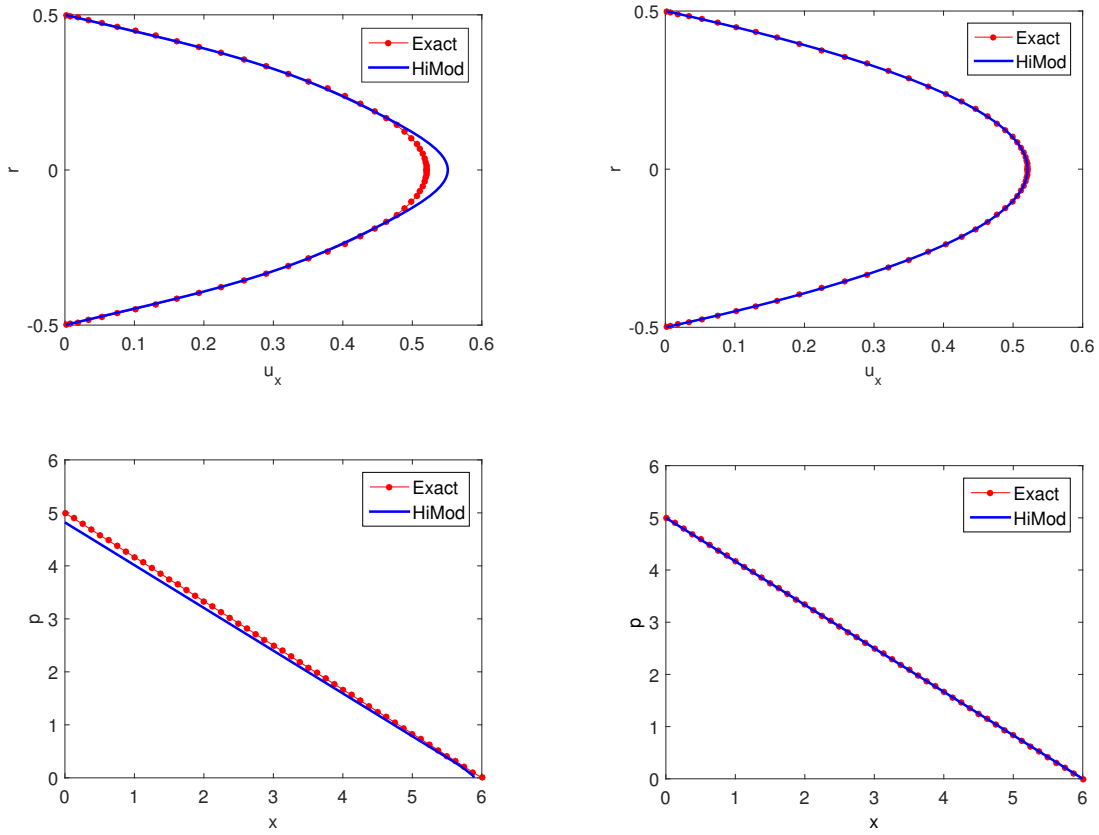


Figure 2.10: Poiseuille flow: Exact (dotted line) and HiMod (solid line) axial velocity at $x = L_x/3$ (top) and pressure drop (bottom) computed via a top-down (left) and a bottom-up (right) basis.

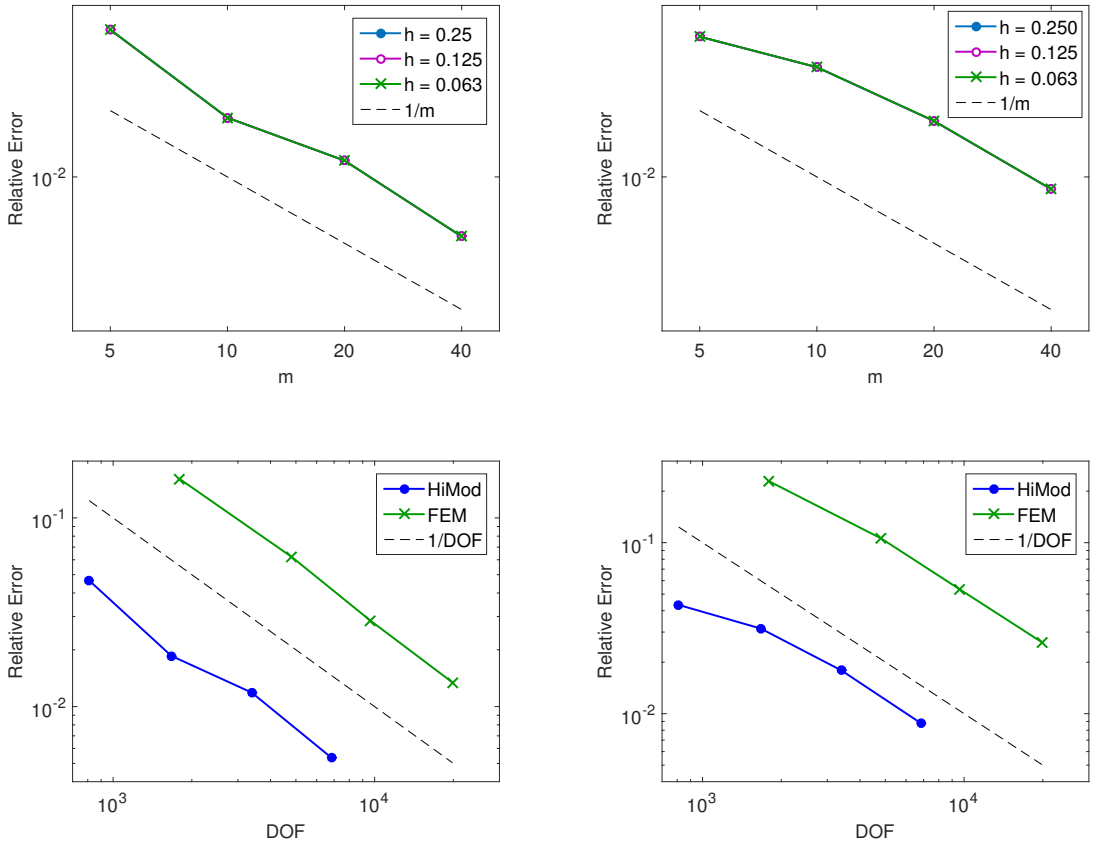


Figure 2.11: Poiseuille flow: $L^2(\Omega)$-norm of the HiMod relative error associated with a Top-Down basis for the velocity (left) and for the pressure (right), for different modes and mesh sizes (top); Comparison between the $L^2(\Omega)$-norm of the relative error associated with HiMod and FEM (bottom) for the velocity (left) and the pressure (right).

expected to yield an inf-sup stable discretization, and the essential pole conditions are automatically satisfied due to the axisymmetry of the solution.

The quadratic solution for the axial component of the velocity belongs to the approximation space, and the parity-restricted Chebyshev basis computes it with a global relative error of the order of 10^{-15} (Figure 2.10, top-right). The same considerations hold for the pressure, which is constant on each section and linear in x (Figure 2.10, bottom-right).

A comparison between the Top-Down and the Bottom-Up basis shows that the former is less efficient than the latter, as expected [35]. In fact, accuracy is improved only with a large number of modes (Figure 2.11, top). Nevertheless, also for the Top-Down approach the error associated with the HiMod approximation is much smaller than with FEM for a fixed size of the discrete problem (Figure 2.11, bottom). Based on these results, the Bottom-Up basis is preferable for the approximation of vector problems with homogeneous Dirichlet lateral boundary conditions and will be adopted in the following numerical tests.

2.4.5 Unsteady case: Womersley flow

HiMod can be generalized to unsteady problems with no particular technical issues. The generalized Navier-Stokes problem needs to be solved at each time step as the result of a standard time discretization. In particular, we solve the problem on the Womersley test case, the well known unsteady counterpart of Poiseuille profile with a time-periodic pressure drop between inlet and outlet [198]. Note that the Womersley (and the Poiseuille) flow is the exact solution both to the Stokes and to the Navier-Stokes equations. The numerical tests presented here are obtained including the non-linear term. In our case, the pressure drop is given by $AL_x e^{i\omega t}$, with constant amplitude A, frequency $\omega = \frac{2\pi}{T}$ and period T (the heart beat). The interplay between the oscillatory flow frequency ω and the effects of the kinematic viscosity ν is described by the Womersley number $\mathrm{Wo} = R\sqrt{\omega/\nu}$.

We reproduce here the test proposed in [127]. The amplitude and the period are set to $A = 1$ and $T = 1$ (i.e., $\omega = 2\pi$), respectively, while the length of the pipe and the radius are set to $L_x = 2$ and $R = 0.2$, respectively. We simulate the flow associated with different Womersley numbers, Wo $\in \{3, 5, 10, 20\}$, by varying the viscosity of the fluid. We simulate a complete period by choosing $\Delta t = \frac{T}{4000}$, and we assign Neumann inflow/outflow axial conditions to enforce the pressure drop along the axial direction, homogeneous Dirichlet conditions for the transverse components of the velocity at the inlet/outlet, and no-slip lateral conditions. The mesh size along the axis Ω_{1D} is $h = 0.125$. For the parity-restricted Chebyshev basis, we set $N_\vartheta = 1$, $N_r \in \{3, 5, 10, 20\}$, so that the total number of pressure modes is $m_p \in \{6, 10, 20, 40\}$ (and $m_u = m_p + 2$).

The HiMod solution is compared to the analytical solution in Figure 2.12 at different times and for the different values of Wo. For low Womersley numbers, characterized by an oscillating profile very close to the Poiseuille parabolic solution, HiMod is accurate even with few modes. As the Womersley number increases, the wave front flattens in the center of the pipe and only a higher number of transverse modes is able to guarantee accuracy (Figure 2.12, third and fourth rows). Table 2.6 shows the $L^2(\Omega)$-norm of the absolute error associated with the HiMod velocity. It reduces of several orders of magnitude as the number of modes increases, and the higher the Womersley number, the more apparent the reduction.

The error obtained with Zernike polynomials is of the same order of magnitude at every time for all Womersley numbers and for all numbers of modes, except for $m_p = 40$, with a discrepancy that increases with the Womersley number (see, e.g., Figure 2.13). In fact, as stated in [37] and confirmed by the numerical tests for an ADR problem, parity-restricted Chebyshev polynomials are more efficient than Zernike polynomials for large numbers of modes. Moreover, the radial modes of the Zernike basis are distributed by construction across different angular modes as m_p is increased, in contrast with the parity-restricted Chebyshev basis, that allows for the construction of an axisymmetric solution by assigning all the radial modes to a

single (constant) angular function.

Figure 2.14 highlights the phase-lag between the (normalized) axial velocity and the oscillating (normalized) pressure on the centerline at the inlet for Wo = 3 and $m_p = 10$, in accordance with [198, 101]. The HiMod solution tightly matches the analytical profile.

We compare the HiMod results with the TEPEM numerical performance. In particular, we refer to Table I in [127]. The accuracy obtained with HiMod is clearly higher than with TEPEM for approximately the same number of modes, especially for highly oscillatory flows. This is likely a consequence of the strict correspondence between the geometry and the HiMod basis, in contrast with the Cartesian map used in TEPEM. On the other hand, TEPEM is easy to implement, being the modal basis a tensor product of univariate Legendre polynomials, even if at the expense of a lack of regularity in the geometrical map.

2.4.6 Choice of the size of the modal space

A critical aspect of the HiMod approach is the selection of the number of modes for the transverse approximation. This has been subject of several investigations [143], including adaptive *a posteriori* strategies [147, 145]. Here, for hemodynamics applications, we consider a practical solution based on the Womersley number. A similar approach is adopted in [184, 192] for the prescription of inflow boundary conditions, defined as a linear combination of elementary Womersley solutions up to the highest significant frequency, identified via the Fourier analysis of the measured data at the inlet and outlet. The approach followed here to select the number of modes is: (i) Compute a cheap yet sufficiently rich truncated modal expansion of a reference or measured solution and plot the modulus of the corresponding modal coefficients; (ii) Identify the first modal coefficient $c_{\bar{k}}$ such that the corresponding modulus and the modulus of all the following modal coefficients lie below a chosen threshold ε, i.e., $c_{\bar{k}} = \max\{c_i : \ |c_i| < \varepsilon, \ \forall i \geq \bar{k}\}$; (iii) Set the dimension of the discrete modal space to $k = \bar{k} - 1$. Note that the asymptotic decay of the modal

		t $[s]$			
		Δt	$T/8$	$T/4$	$3T/8$
Wo = 3	m_p=6	6.1605e-05	3.0130e-04	3.7547e-04	2.3274e-04
	m_p=10	6.1605e-05	3.0130e-04	3.7547e-04	2.3274e-04
	m_p=20	1.0816e-05	2.3650e-05	3.9187e-05	3.9295e-05
	m_p=40	3.8581e-08	2.2202e-05	3.9162e-05	3.7966e-05
Wo = 5	m_p=6	1.4950e-03	2.9939e-04	1.7631e-03	2.4781e-03
	m_p=10	1.4950e-03	2.9939e-04	1.7631e-03	2.4781e-03
	m_p=20	1.2740e-04	2.3878e-04	2.0445e-04	8.6548e-05
	m_p=40	2.6278e-08	2.2155e-05	5.3320e-05	7.2622e-05
Wo = 10	m_p=6	9.2405e-03	9.2813e-03	4.7665e-03	4.6525e-03
	m_p=10	9.2405e-03	9.2813e-03	4.7665e-03	4.6525e-03
	m_p=20	2.4344e-03	9.3545e-04	1.6594e-03	3.0479e-03
	m_p=40	1.8549e-08	2.0676e-05	5.8521e-05	9.0916e-05
Wo = 20	m_p=6	9.7086e-03	1.6770e-02	1.5177e-02	6.6678e-03
	m_p=10	9.7086e-03	1.6770e-02	1.5177e-02	6.6678e-03
	m_p=20	7.4874e-03	8.7910e-03	5.7941e-03	3.5640e-03
	m_p=40	1.2963e-08	1.9626e-05	6.0757e-05	9.9285e-05

Table 2.6: Womersley flow: $L^2(\Omega)$-norm of the error associated with the HiMod velocity for different Wo numbers and at different times.

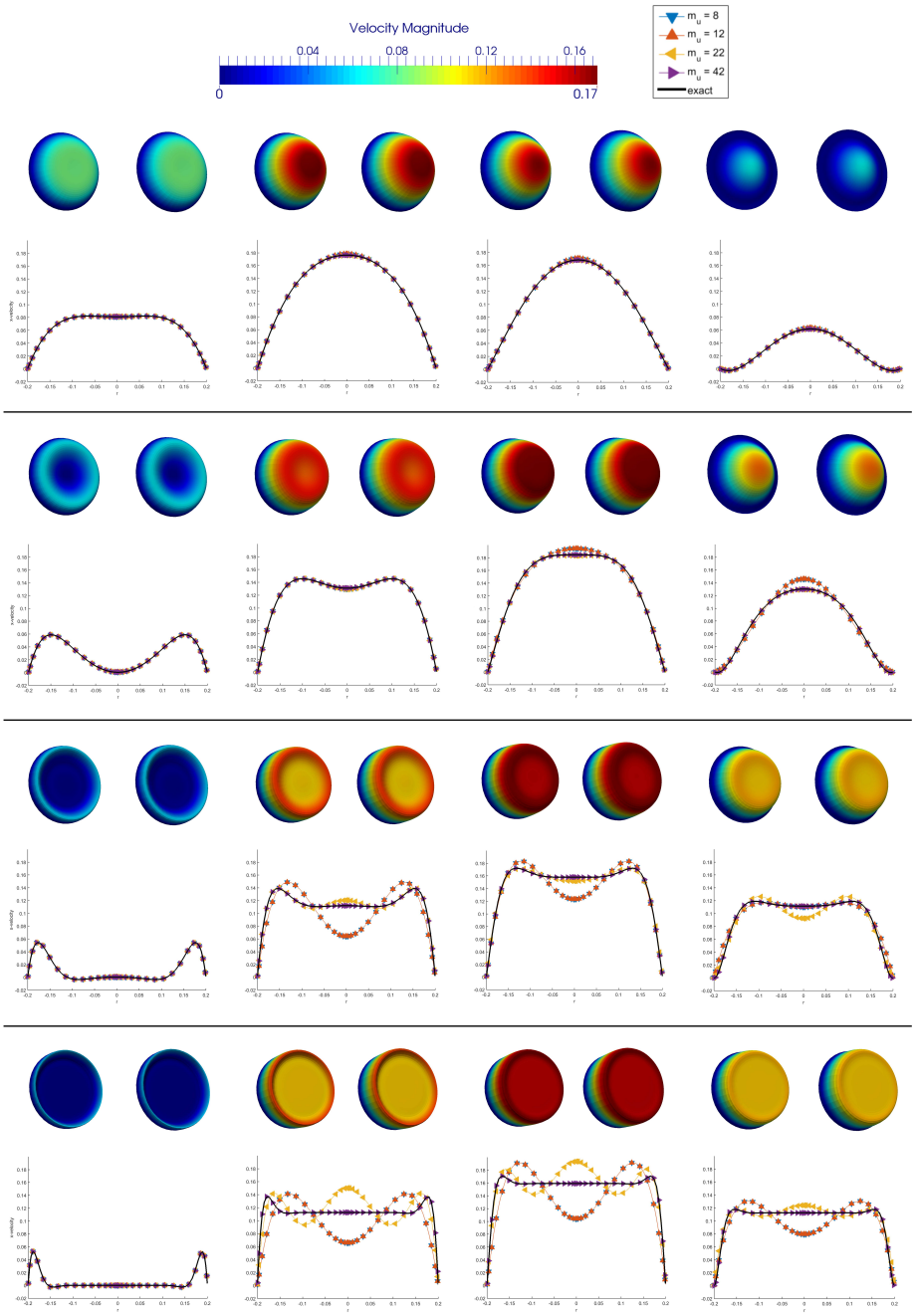


Figure 2.12: Womersley flow in a cylindrical pipe: In each panel velocity (top) and axial component (bottom) profile for the exact (left and solid line) and the HiMod (right and $\triangledown$, $\triangle$, $\triangleleft$, $\triangleright$ markers) solution at $x = L_x/2$ at times $t = 0, T/8, T/4, 3T/8$ (left-right) for $\mathrm{Wo} = 3, 5, 10, 20$ (first-fourth row).

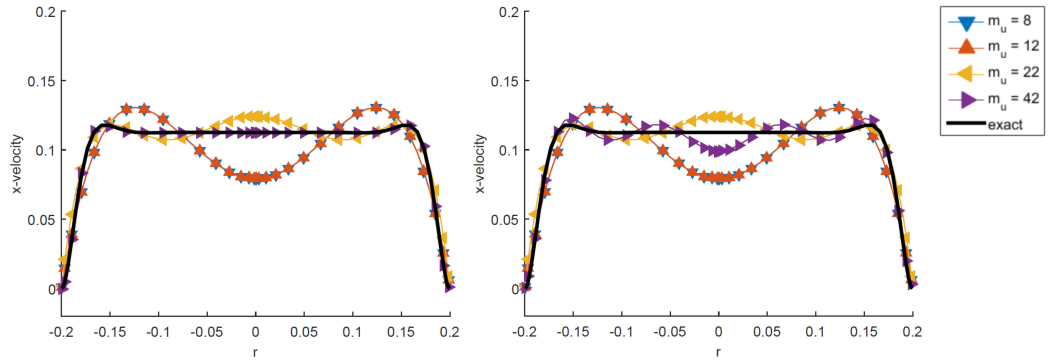


Figure 2.13: Womersley flow in a cylindrical pipe for Wo = 20: exact (solid black line) and HiMod solution (∇, $\triangle$, $\lhd$, $\rhd$ markers) with parity-restricted Chebyshev (left) and Zernike (right) basis at $t = 3T/8$.

coefficients is guaranteed by Parseval's identity. In particular, the convergence rate for Chebyshev series is proved in [36].

To test the idea, we compute the spectral size for different flows corresponding to Womersley numbers Wo $\in \{5, 10, 15, 20\}$ in a cylinder with length $L_x = 2$ and constant radius $R = 0.2$ by applying the procedure described above. Figure 2.15 shows the decay of the modal coefficients of the exact axial velocity and the selected spectral size k after a truncation with tolerance $\varepsilon = 10^{-5}$. It is worth noting that, for a fixed ε, the selected dimension of the modal space increases with the Womersley number, in accordance with the results obtained in Section 2.4.5. We also notice that, for the range of Womersley number considered, which is spanning a physiological range in humans, a number of modes ≤ 10 is enough.

Remark. HiMod and full FEM solvers solve the same problem with a different number of degrees of freedom. In particular, HiMod improves 1D solvers and runs on architectures that not necessarily belong to the category of High Performance Computing facilities. In fact, in practice, for some applications such as computational fluid dynamics in clinical routine, having access to this kind of facilities is not always possible.

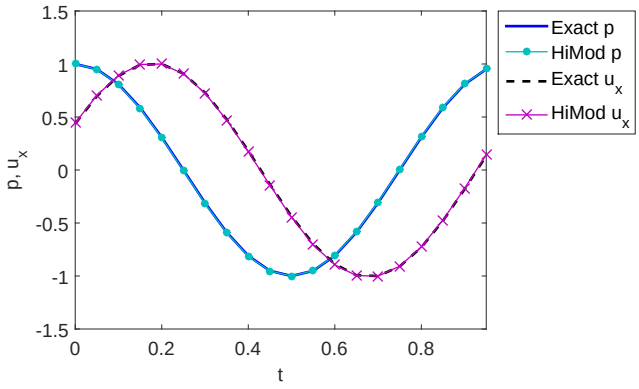


Figure 2.14: Womersley flow in a cylindrical pipe: Normalized exact (solid line) and HiMod (dotted line) pressure; normalized exact (dashed line) and HiMod ($\times$) axial velocity on the centerline at the inlet for Wo $= 3$, $m_p = 10$.

Performing a comparison based on the computational time is therefore quite problematic, as the different solvers run on different architectures, with different implementations and, ultimately, with different linear algebra cores. This circumstance was already pointed out in [127], where a comparison based on execution time required a normalization of the CPU per computing node, as TEPEM was running on much simpler architectures than FEM. We do think that the number of degrees of freedom is an objective indicator to establish a fair comparison, being independent of factors that do not strictly attain at the methods. The fact that HiMod with cylindrical coordinates is competitive with TEPEM indirectly confirms that HiMod is competitive vs. FEM also in terms of CPU time.

More in general, we argue that, with comparable architectures and the most appropriate choices of the linear algebra solvers, HiMod does outperform FEM in terms of computational time. In fact, the additional costs of the assembly of the HiMod system matrix (due to the numerical quadratures of the HiMod coefficients) is expected to be considerably compensated by the computational saving associated with the solution of a smaller yet "educated" algebraic system. Furthermore, the multi-level block-structure of the HiMod matrix makes the assembly phase easily parallelizable

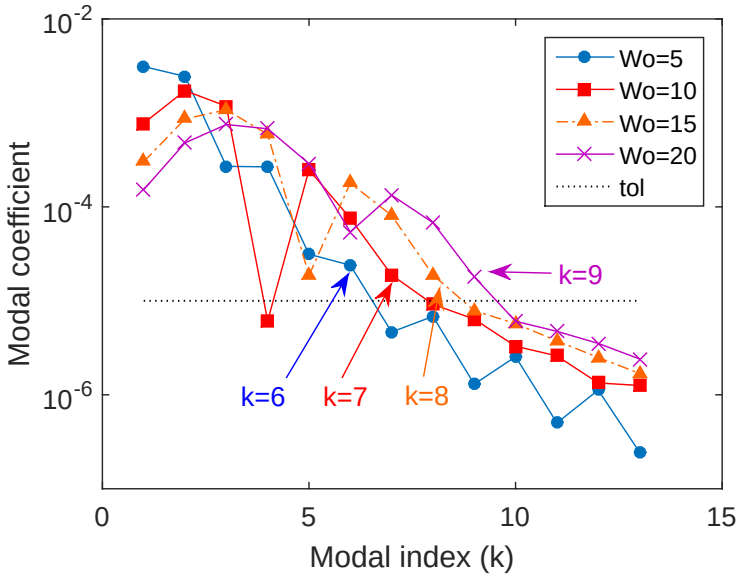


Figure 2.15: Modal coefficients of the Womersley solution for Wo = 5 (∘), Wo = 10 (□), Wo = 15 (△), Wo = 20 (×). The truncation is performed with a tolerance of $\varepsilon = 10^{-5}$ (dashed line).

both with distributed- and shared-memory paradigms. The preliminary numerical experiments run in [23] for ADR problems using distributed-memory architectures showed that superlinear speedup can be achieved. This feature is extremely convenient especially for nonlinear or time-dependent problems, where each iteration requires updating the system matrix, as it can significantly expedite the execution.

Hence, overall, we stress that the fact that HiMod may reliably run on smaller computers is per se a value of our approach, as it enables using computational fluid dynamics on small and local facilities, with a significant practical impact.

2.5 Numerical Tests in Axisymmetric and Non-Axisymmetric Domains

We present some further numerical tests inspired by computational hemodynamics. The ultimate goal of HiMod in this field is to provide an efficient way for simulating blood flow in arteries with computational costs comparable to 1D models, yet preserving a sufficient local accuracy in the transverse components. TEPEM, presented in [127], already demonstrated that this is possible. We provide here a proof of concept in a cylindrical framework by using axisymmetric and non-axisymmetric geometries. In such a context, the properties of the modal basis are crucial for the local regularity of the solution. Based on the results obtained so far, parity-restricted Chebyshev polynomials are adopted for axisymmetric problems because of their flexibility and efficiency, whereas Zernike polynomials are selected for the general case, due to their robustness and regularity. Moreover, a modified basis of Zernike polynomials is introduced by scaling the transverse components by r. This allows for the fulfilment of the pole conditions not only for the axial, but also for the transverse components of the basis, and of the parity conditions for vector functions in cylindrical coordinates. Despite the limitations and the challenges in the construction of a suitable modal basis, the HiMod solution will be able to capture the transverse dynamics induced in the flow by the geometry or by the physics with a relatively small number of degrees of freedom.

2.5.1 Axisymmetric models

We consider three types of geometries that model (i) the natural tapering, (ii) a local expansion (aneurysm), and (iii) a local constriction (stenosis) of blood vessels.

Tapered pipes. Tapering has a significant impact on hemodynamics (and, consequently, on the design of grafts and prosthesis [65, 49, 103]). Following [164], we

consider a pipe with radius

$$R(x) = -(\tan \Psi)x + R_{in}, \tag{2.25}$$

where $\Psi = \frac{(R_{out}-R_{in})}{L_x}$ is the tapering angle, and with R_{in} and R_{out} the radius of the inflow and of the outflow section, respectively (Figure 2.16, top).

We set $R_{in} = 1$, $R_{out} = 0.5$, and $L_x = 6$, and we solve the Navier-Stokes equations with the Womersley flow as an inflow velocity profile with $A = 1$, $\nu = 2.75 \times 10^{-2}$. The transverse components are set to 0 on all the boundaries, and axial Neumann conditions are enforced at the outlet. The space and time discretization steps are $h = 0.125$ and $\Delta t = 1/100$, respectively. The number of basis functions for the parity-restricted Chebyshev basis is $m_p = 16$, $m_u = 18$ ($N_\vartheta = 1$, $N_r = 8$).

The flow is driven by a space-dependent time-oscillating pressure gradient (Figure 2.17, left). The oscillation of the Womersley velocity profile at the center of the inlet section of the pipe increases along the x-axis, so that the maximal oscillation is reached at the outlet (Figure 2.17, right). On each transverse section and at each time step, the x-component of the velocity reproduces a Womersley-like profile (Figure 2.18, top). The tapering of the domain triggers transverse dynamics, and the velocity is not purely axial. The presence of a radial component is correctly detected by the HiMod solution (Figure 2.18, center). In particular, the planar components point inward as long as the peak of the axial velocity is positive, and turn outward as soon as the flow reverses. Figure 2.18 (bottom) shows the 3D HiMod approximation for the velocity at four different times on the whole domain.

Aneurysmatic vessels. An aneurysm is a balloon-like dilation in an arterial vessel (Figure 2.16, center). The growth and rupture of the bulge is related to hemodynamics factors, such as blood velocity, wall shear stress, pressure, particle residence time and flow impingement [162]. We model the radius of the bulge as a quadratic exponential function of the axial variable, i.e.,

$$R(x) = R_{in} + \kappa e^{-\left(x-\frac{L_x}{2}\right)^2}, \tag{2.26}$$

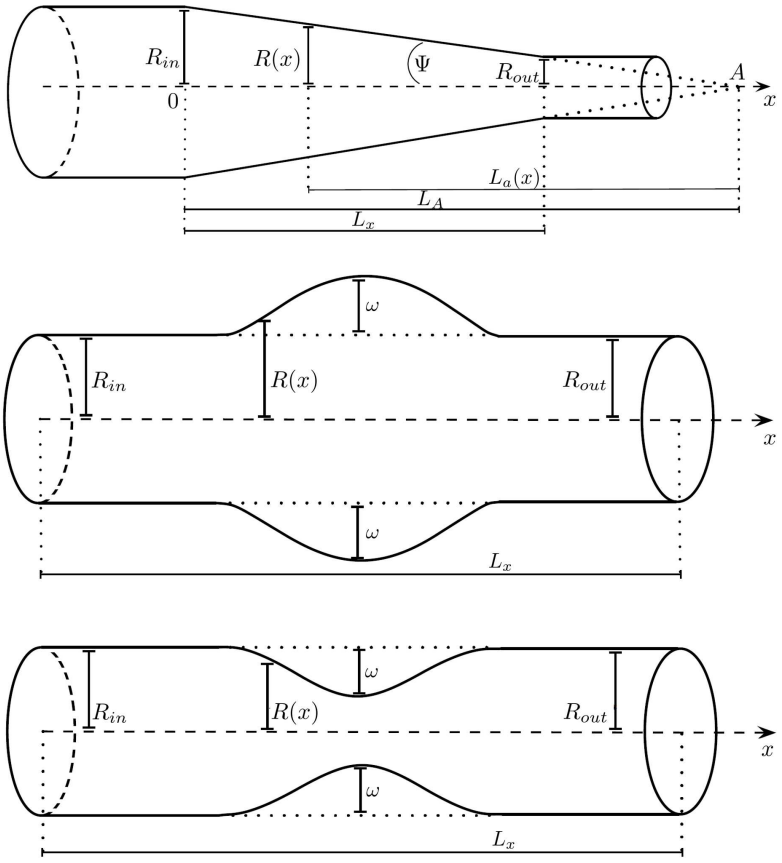


Figure 2.16: Sketch of a tapered pipe (top), of an aneurysmatic (center) and of a stenotic (bottom) vessel.

where R_{in} is the radius of the inflow and of the outflow sections, while κ, with $\kappa \in [0, 1]$, takes into account the entity of the dilation. We set specifically $R_{in} = 0.5$, $L_x = 6$, and $\kappa = 0.45$, and we solve the NSE with the same conditions and data used above for a tapered pipe.

At each time the pressure is linear along the x-axis in the inflow and outflow cylindrical segments, with an inflection point in the aneurysm. In particular, it increases where the vessel enlarges, and it reduces where the regular lumen of the vessel is restored (Figure 2.19, left). The axial velocity on the centerline drops inside the bulge, for the conservation of mass (Figure 2.19, right). We refer to [94] for a detailed analysis of the effect of the size κ of the aneurysm on the pressure and velocity profiles. The axial component of the velocity on each transverse section features a Womersley-like profile (Figure 2.20, top). The flow inversion in the bulge, in the proximity of the wall, is much milder than in the inflow and proximal sections (Figure 2.20, top-right). The transverse components of the velocity upstream of the aneurysm are directed outward until the flow reverses (Figure 2.20, center). The 3D velocity profiles on different sections are shown in Figure 2.20 (bottom).

Stenotic vessels. We consider a local constriction like in atherosclerotic plaques (Figure 2.16, bottom), modeled as in (2.26), with $\kappa \in [-1, 0]$. Due to the symmetry of the problem, the numerical solution obtained with a Womersley inflow profile is the counterpart of the solution obtained in an aneurysmatic vessel for $\kappa < 0$ and with the same data, and is therefore omitted for the sake of brevity. A full description of the results can be found in [94].

In order to assess the quality of the HiMod approximation of transverse dynamics, we are interested in solving an Oseen problem [24] by linearizing the non-linear term in (2.17) with a twisting time-dependent convective field given by $\mathbf{b}(t) = [b_x, b_r, b_\vartheta]^T = [1, 0, 2t^2]^T$. In this case we set the inflow profile as a paraboloid with amplitude A increasing in time as $A(t) = 2t^2$.

As the geometry and the rotational convective field activate the coupling of the radial and angular components of the velocity, the regularity properties of the modal

basis have a major impact on the HiMod approximation. The Cartesian transverse components (u_y and u_z) of the solution obtained with the parity-restricted Chebyshev basis and the Zernike basis are singular at $r = 0$ (Figure 2.21, top, first and second plot). On the contrary, the modified Zernike basis yields a smooth solution, thanks to the compliance with the natural pole conditions (third panel in Figure 2.21, top). The 2D plot of the transverse components of the HiMod velocity on the section at $x = 2L_x/3$ and at $t = 1$ is shown in Figure 2.21 (top-right).

2.5.2 Non-axisymmetric models

As a simplified model of a non axisymmetric geometry, we consider a pipe with elliptical section, with major and minor axis of length Y and Z, respectively, and $Z < Y$. We perturb Y and Z with a negative and positive sinusoidal function of x, respectively, so that $a(x) = Y - Y/4 \sin(4\pi x/L_x)$, and $b(x) = Z + Z/4 \sin(4\pi x/L_x)$ (Figure 2.22, first two rows). As a result, the surface can be expressed as a function of x and ϑ as

$$R(x, \vartheta) = \frac{a(x)b(x)}{\sqrt{a^2(x)\sin^2(\vartheta) + b^2(x)\cos^2(\vartheta)}}. \tag{2.27}$$

We solve a steady Oseen problem with convective field $\mathbf{b} = [b_x, b_r, b_\vartheta]^T = [10, 0, 5]^T$ and a parabolic inflow profile $u_0(r, \vartheta) = U(1 - r^2/R(0, \vartheta)^2)$ of amplitude $U = 1$, and the unsteady NSE with an oscillating parabolic inflow profile $u_{in}(r, \vartheta, t) = u_0(r, \vartheta)\sin(2\pi t)$. We set $Y = 0.2$, $Z = 0.1$, $L_x = 2$, and $\nu = 2.75 \times 10^{-2}$. The number of modes for the pressure, the mesh size along Ω_{1D}, and the time step are set to $m_p = 12$ ($N_\vartheta = 3$, $N_r = 2$), $h = 0.02$, and $\Delta t = 1/100$, respectively.

In the steady Oseen problem, the rotational component of the convective field and the lack of axial symmetry in the geometry trigger transverse dynamics in the velocity field. As already observed for the Oseen problem in an axisymmetric stenotic pipe, the fulfilment of the pole conditions is crucial for the local regularity of the HiMod solution. In fact, only the transverse components obtained with the modified Zernike basis are smooth at the origin (Figure 2.21, bottom). Therefore, for the solution

of the non-linear time-dependent problem commented hereafter only the modified Zernike basis is adopted.

The opposite sign of the perturbation of the axes of the transverse section along the main stream is reflected in the oscillatory pattern of the transverse components of the velocity (Figure 2.22, third and fourth row). The radial component features a cosinusoidal behavior, with a positive (negative) oscillation along the minor (major) axis of the section when it is decreasing (increasing) along x, and conversely. Differently, the angular component is characterized by a sinusoidal pattern, induced by the widening or narrowing of the vessel. The combination of such dynamics is visualized in the vector plots of Figure 2.22 (last row).

We stress that the transverse dynamics properly described in these results are out of reach for 1D models, while HiMod - that conceptually is a sort of 1D enriched modeling - is able to capture the local components of these dynamics by properly tuning the spectral discretization.

2.5.3 Patient-specific geometries

The reconstruction of the human vasculature from medical images has been pursued since the early 90s (see, e.g. [14, 76]). The fundamental steps are: 1) Extract the lumen of interest from each frame (segmentation), which is typically done by level-set based methods; 2) Connect the different frames, which, with a CT dataset, requires stacking the different slices obtained from the images in a volume; 3) Extract the surface of the volume typically in the form of a triangulated surface (STL file). Possibly, the surface needs smoothening operations to regularize the impact of image or segmentation artifacts. These steps are conveniently implemented in the open-source library `vmtk` [15].

In the HiMod frame, special processing procedures are required by the spacial discretization, to enhance the separation of variables. In [30], the physical domain is approximated as a sequence of slabs, called pipe-like elements. A generic element is mapped to a reference domain, defined as the cube $(\xi, \eta, \zeta) \in (-1, 1)^3$. More

precisely, each transverse section γ_x of the vessel is mapped to the reference fiber $\hat{\gamma} = (-1,1)^2$ via cubic serendipity polynomials $\{S_i\}_{i=1}^{12}$. Then, three consecutive sections are interpolated with quadratic interpolating polynomials $\{Q_k\}_{k=1}^{3}$ defined on the nodes $\{\xi_k\} = \{-1,0,1\}$. As a result, the map from the reference domain to the real geometry reads as:

$$\chi_E(\xi,\eta,\zeta) = \sum_{k=1}^{3} \chi_k^g(\xi,\eta) Q_k(\zeta) = \sum_{k=1}^{3}\sum_{i=1}^{12} \mathbf{x}_i^{(k)} S_i(\xi,\eta) Q_k(\zeta), \tag{2.28}$$

where $\left\{\mathbf{x}_i^{(k)} = (x_i^k, y_i^k, z_i^k) : \ i = 1,\dots,12\right\}$ is the set of points on the transverse section mapped from the section $\zeta = k-2$ on the reference domain. Although a Cartesian reference system is practical for the construction of a (modal) basis via tensor product, mapping a cylinder to a cube introduces singularities that may spoil the numerical approximation of the problem.

In this work we construct a different geometric map defined on a cylinder with unit radius as a reference domain, which is more tailored to hemodynamics applications. This choice leads to a simpler structure and a higher regularity of the geometric map, with a potential improvement in the accuracy of the numerical solution. More specifically, the polar reference system allows a lower-dimensional approximation of the transverse fiber. Indeed, each section γ_x is mapped to the unit disk $\hat{\gamma}$, so that the physical boundary $\partial\gamma_x$ is parametrically described by the radius as a function of the angular coordinate ϑ. Accordingly, the map χ_k^g in (2.28) reduces to a scalar 1D interpolating function of ϑ, i.e.,

$$\chi_k^g(\vartheta) = \sum_{i=1}^{\Sigma_k} \mathbf{p}_i^{(k)} P_i(\vartheta), \tag{2.29}$$

where $\mathbf{p}_i^{(k)} = \left(x^{(k)}, r_i^{(k)}\cos(\vartheta_i), r_i^{(k)}\sin(\vartheta_i)\right)$, and Σ_k is the number of sampled points on the boundary $\partial\gamma_{x^{(k)}}$. In particular, the polynomials $\{P_i\}_{i=1}^{\Sigma_k}$ are chosen as periodic cubic splines to suit the periodicity of the domain. Then, the whole sequence of N_s slices can be interpolated via piecewise cubic Hermite polynomials $\{H_k\}_{k=1}^{N_s}$, so that

the global map reads as

$$\chi_E(\zeta, \vartheta) = \sum_{k=1}^{N_s} \sum_{i=1}^{\Sigma_k} \mathbf{p}_i^{(k)} P_i(\vartheta) H_k(\zeta). \tag{2.30}$$

Figure 2.23 shows different sections interpolated on the ϑ-quadrature nodes.

We solve the Navier-Stokes Equations in the patient-specific geometry shown in Figure 2.24 (top), obtained from medical images (OCT, courtesy of Dr. Giulio Guagliumi, Hospital Papa Giovanni XXIII, Bergamo (Italy), and Marina Piccinelli). The inflow profile is obtained from real data via polynomial interpolation (see Figure 2.24, bottom-left). Despite the issues related to the employment of a vector modal basis in polar coordinates on a non-axisymmetric geometry, the model is capable of capturing the blood recirculation downstream of the stenosis (see streamlines in Figure 2.24, bottom-right), which is relevant for therapeutic purposes, as it is an indicator of the progression of the stenotic plaque.

2.6 Conclusions

In this work we extended the Hierarchical Model Reduction to geometries described by a cylindrical coordinate system. While the application of HiMod to 2D or 3D slabs is straightforward thanks to the Cartesian tensor product, the cylindrical setting requires a specific analysis. Using polar coordinates in spectral discretizations raises some challenges inherited by a HiMod reduction in a cylindrical framework. A possible approach, based on mapping the circular sections to Cartesian (square) sections was already investigated in [127]. Here we resorted to a genuine polar coordinate spectral discretization, which seems more natural for domains like the ones occurring in computational hemodynamics. Nevertheless, the selection of appropriate basis functions is troublesome both for the scalar case, as excellently pointed out in [37], and - even more - for the vector case, where additional analytical, numerical, and physical constraints need to be met.

First, we considered a standard ADR problem. The comparison between the HiMod solution and the corresponding 3D approximation highlights the competitiveness of HiMod with respect to FEM, both in terms of computational effort and accuracy of the approximation. HiMod significantly improves any 1D solver, with no meaningful detriment for the computational costs, independently of the choice of the basis. In general, a Bottom-Up approach is more efficient than a Top-Down procedure, both for axisymmetric and non-axisymmetric solutions that satisfy homogeneous Dirichlet lateral boundary conditions. However, the latter allows for more flexibility in the choice of the lateral boundary conditions. Among the Bottom-Up bases, there is no *best* choice. For problems where some a priori information (symmetry, spectral properties, ...) about the solution is available, the parity-restricted Chebyshev basis is more efficient, since it allows for more freedom to suitably distribute the radial modes across the angular components. For general problems, the Zernike basis is performing better for small numbers of modes, but it is outperformed by the parity-restricted Chebyshev polynomials for m large. Moreover, among these two specific basis types, the regularity of the solution at the pole is guaranteed by the Zernike but not by the Chebyshev polynomials.

The extension to vector problems for the incompressible Navier-Stokes equations is more problematic, as the construction of a basis that satifies (i) boundary conditions, (ii) well-posedness constraints, and, possibly, (iii) orthogonality and (iv) regularity conditions at the pole is still an open problem. As the Top-Down approach lacks approximability properties, a Bottom-Up strategy seems more viable. With this type of basis we were able to solve the incompressible Navier-Stokes equations with a small number of degrees of freedom, so to make the HiMod solver competitive with purely 1D codes, yet retaining transverse dynamics. As found for scalar problems, the parity-restricted Chebyshev vector basis is more efficient for axisymmetric problems. For general problems, the Zernike vector basis is more robust and, with a suitable scaling of the transverse components, it is able to provide smooth solutions.

In comparison with "Cartesian" strategies like in the TEPEM [127], we take ad-

vantage of the regularity of the geometrical map, so to have better performances in reducing the degrees of freedom. It is apparent that the cylindrical frame has several drawbacks, in particular from the implementation standpoint. In particular, the assembly of the matrix requires more effort in the presence of complex geometries. Furthermore, it is worth noting that other bases beyond parity-restricted Chebyshev and Zernike could be considered, including hybrid strategies using the Top-Down approach for each component of the velocity (for the easiness of including the boundary conditions) and the pressure separately, or the Logan-Shepp basis [37].

The natural follow-up of the present work is the extension of HiMod to patient-specific geometries and the set-up of a network solver where each pipe is treated as in the present paper, as done in [30] (in a Cartesian frame). Simultaneously, we plan to introduce fluid-structure simplified models in the HiMod solver, so to ultimately compare a 1D HiMod network solver with the current 1D solvers used in hemodynamics and gasdynamics, based on the Euler equations [140]. In both the cases, upon the results of the present paper, we plan to use the (modified) Zernike polynomials as at the current status of our knowledge they provide the best trade-off between approximation and efficiency. The final goal is to replace current solvers with the HiMod ones, to adjust the local accuracy for the transverse components by suitably selecting the spectral discretization parameters.

A comprehensive analysis of the inf-sup condition for the HiMod basis in a cylindrical (as well as in Cartesian) frame still requires investigation.

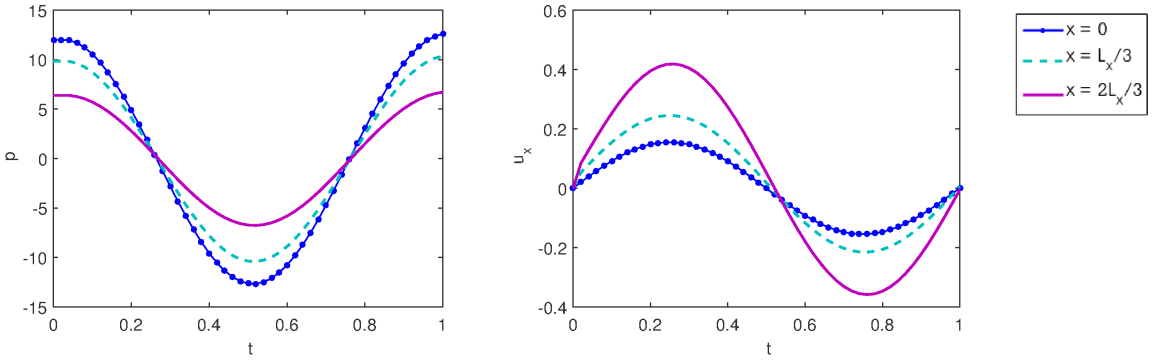
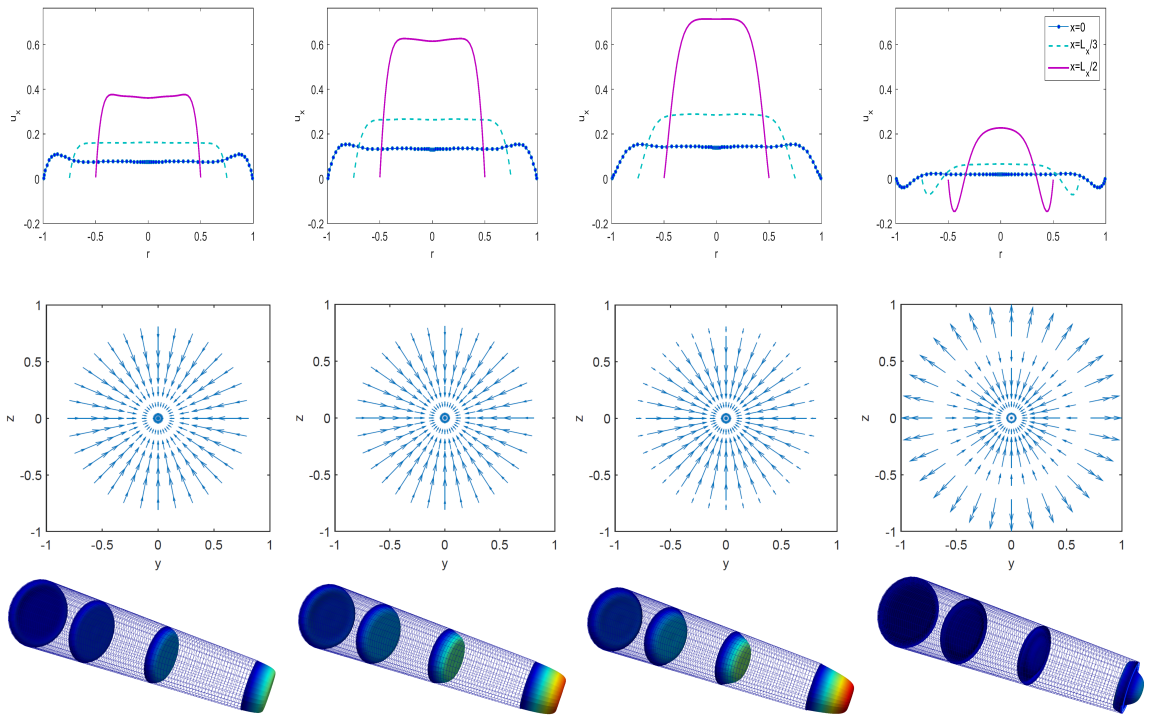


Figure 2.17: Womersley-like flow in a tapered pipe: Oscillating pressure (left) and centerline velocity (right) at $x = 0$ (dotted line), $x = L_x/3$ (dashed line), and $x = 2L_x/3$ (solid line).



(a) $t = 0.08s$ (b) $t = 0.16s$ (c) $t = 0.32s$ (d) $t = 0.48s$

Figure 2.18: Womersley-like flow in a tapered pipe: Axial velocity at $x = 0$ (dotted line), $x = L_x/2$ (dashed line), and $x = L_x$ (solid line) (top); Radial velocity at $x = L_x/3$ (center) and 3D HiMod velocity profile (bottom) at different times.

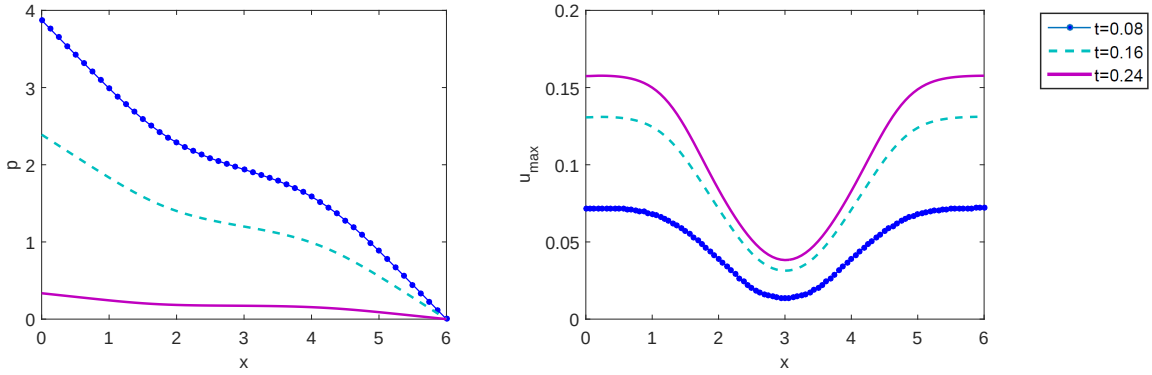
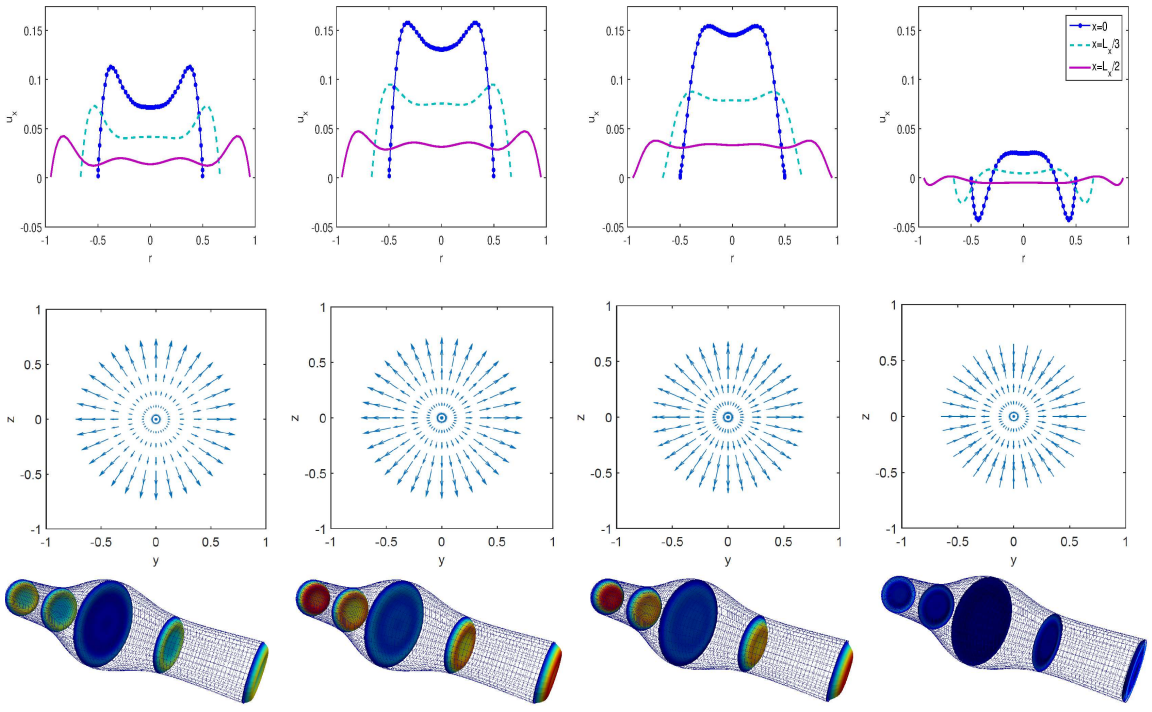


Figure 2.19: Womersley-like flow in an aneurysmatic pipe: Pressure (left) and axial velocity on the centerline (right) along the x-axis at $t = 0.08s$ (dotted line), $t = 0.16s$ (dashed line), and $t = 0.24s$ (solid line).



(a) $t = 0.08s$ (b) $t = 0.16s$ (c) $t = 0.32s$ (d) $t = 0.48s$

Figure 2.20: Womersley-like flow in an aneurysmatic vessel: Axial velocity at $x = 0$ (dotted line), $x = L_x/3$ (dashed line) and $x = L_x/2$ (solid line) sections (top); Radial velocity at $x = L_x/3$ (center); 3D HiMod velocity profile (bottom) at different times.

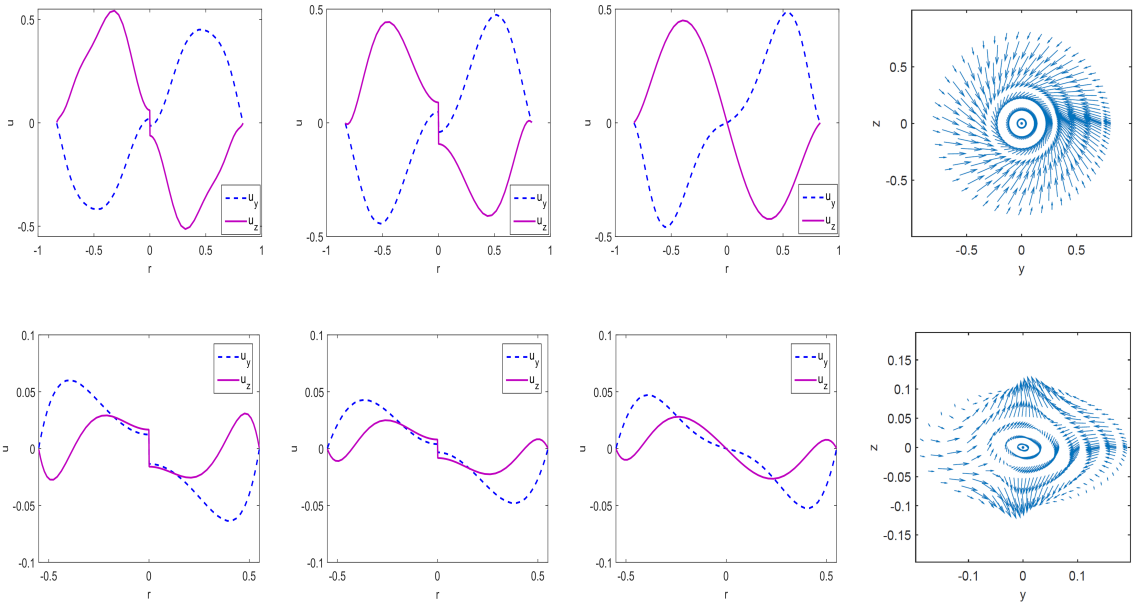


Figure 2.21: Starting flow in a stenotic pipe at $x = 2L_x/3$ and at $t = 1$ (top) and steady Oseen flow in a non-axisymmetric pipe at $x = L_x/2$ (bottom) with twisting convective field: y- (dashed line) and z- (solid line) component of the velocity along the y axis with parity-restricted Chebyshev (first column), Zernike (second column), and modified Zernike (third column) basis; Transverse components of the velocity (fourth column).

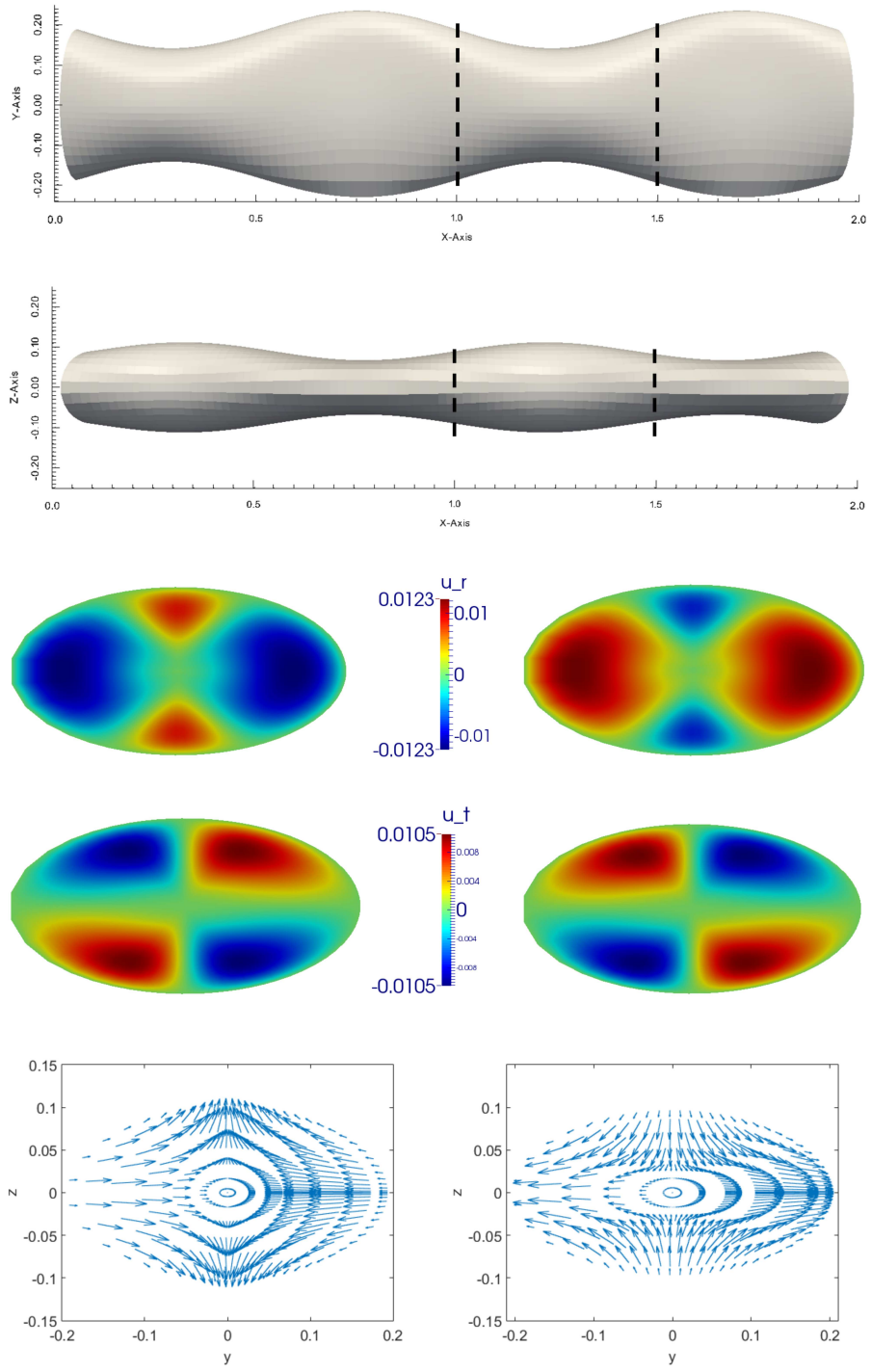


Figure 2.22: Womersley-like flow in a non-axisymmetric vessel: xy- and xz-plane view of the geometry (first and second row); color plots of the radial (third row) and angular (fourth row) components of the velocity and vector plots of the transverse components of the velocity (fifth row) at $x = 1$ (left) and $x = 1.5$ (right) at $t = 0.24s$.

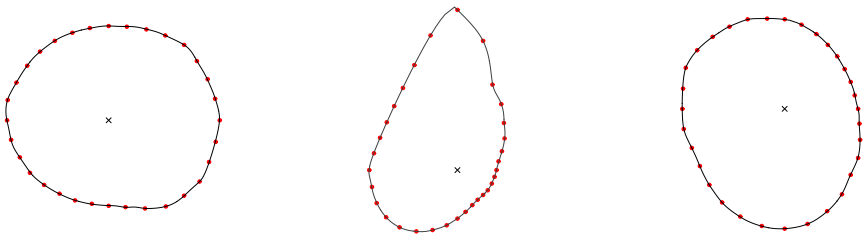


Figure 2.23: STL sections (solid line) and HiMod map (dotted line) on the ϑ-quadrature nodes along the domain. The intersection between the transverse fiber and the centerline of the vessel is marked with a cross ($\times$).

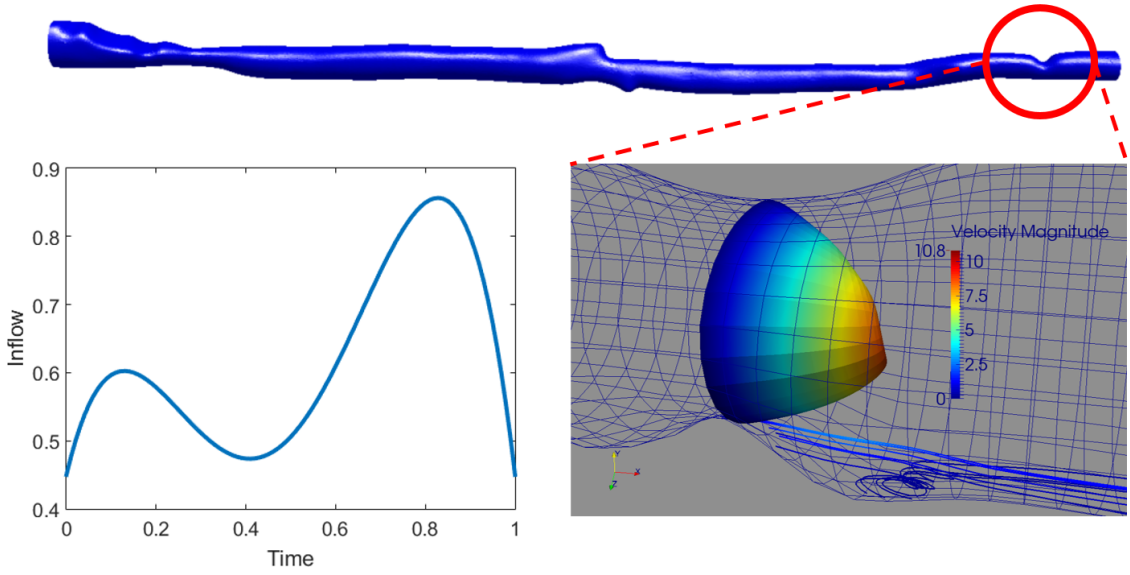


Figure 2.24: Patient-specific geometry (top); Inflow profile (bottom-left); Recirculation in proximity of the stenosis (bottom-right).

Chapter 3

Network Uncertainty Quantification via Domain Decomposition

Acknowledgements. This chapter contains part of the content of the paper [45], written in collaboration with Kevin Carlberg, Mohammad Khalil, and Khachik Sargsyan, that has been part of the research of the author during two summer internships at the Sandia National Laboratories (Livermore, CA). We also include preliminary results on acceleration techniques for the DDUQ problem, that will be extended and consolidated in [46].

3.1 Introduction

For extreme-scale decomposable systems, high-fidelity simulation and uncertainty quantification (UQ) are typically achievable at the subsystem level, but intractable for the full system. This is due to long full-system simulation times, limits on the problem size that can be reliably simulated, and challenges in integrating subsystem models characterized by different physics or scales. Even full-system surrogate models, which are typically employed to make UQ tractable, may not be possible to construct, as they rely on a training set of these (possibly infeasible) simulations.

Current approaches to UQ for coupled subsystems use simplified probability representations [93, 63, 16], consider only feedforward systems [11], or restrict their attention to two-component systems [17, 53] - especially in the context of multiscale modeling [137, 134, 135]. These approaches quickly become intractable when considering non-Gaussian networks (based on elastic mass-and-spring model) with many subsystems. The most advanced work in this regard is [120], which builds on [11] to devise an additive-Schwarz-like approach for large-scale networks. However, it relies extensively on importance sampling (which assumes accurate proposal distributions can be specified a priori) and density estimation (which scales poorly with the number of coupling variables). Further, it enforces high-fidelity compatibility constraints that require full-system simulations for each sample. While this is mitigated by the use of response-surface surrogates, the uncertainty introduced by these surrogates is not characterized. The key insights of our proposed work are that (1) Reduced-order models (ROMs) enable rapid subsystem sampling, which obviates the need for importance sampling and density estimation, (2) ROMs also enable rigorous epistemic uncertainty quantification, and (3) the complete set of DD methods - combined with a hierarchy of compatibility conditions - can be considered and leveraged to enable scalable and rigorous network UQ for arbitrarily complex and large-scale networks. Subsystem model reduction has also been pursued in the literature, but almost exclusively for linear problems. In particular, the *component mode synthesis* (CMS) approach (i.e., Craig-Bampton) [105, 57] has been widely used for model reduction in component-based structures. This method performs eigenmode model reduction on each component and couples them using a primal-Schur approach. While extensions to nonlinearities localized at component interfaces [171] and low-order-polynomial (geometric) nonlinearities have been proposed [10, 132], CMS has not been extended to handle general nonlinearities. Relatedly, reduced-basis element methods have been developed for elliptic partial differential equations. These methods couple component reduced-basis approximations using either dual-Schur [124] or primal-Schur [106] approaches.

In this context, the Domain Decomposition Uncertainty Quantification (DDUQ) method that we propose is a scalable "bottom-up" approach that performs extensive uncertainty quantification and model reduction at the (tractable) subsystem level, and propagates coupling information using new techniques inspired by domain decomposition. More specifically, the approach assigns each subsystem to a node in a directed graph, and coupling variables to edges. Inter-node compatibility is guaranteed by enforcing constraints on the statistical distance between matching coupling variables, and these constraints are enforced by developing new UQ analogues to DD methods. On top of this, a ROM for each subsystem can be constructed during an "offline" pre-processing stage, and integrated with network UQ to enable efficient subsystem sampling. This approach is promising because (1) full-system ROMs and UQ analyses can be realized without any full-system simulations, (2) thanks to network UQ, full-system uncertainties can be attributed to local contributions for targeted uncertainty reduction, and (3) 'Lego-like' design enables solver modularity and scalability. This divide-and-conquer strategy supports subsystem independence, as tailored uncertainty analysis, discretizations, solvers, and ROM techniques can be applied to each subsystem separately. As a result, the level of parallelism of the underlying numerical solvers is maximized.

This Chapter is organized as follows: the DDUQ method is introduced in Section 3.2 and tested numerically in Section 3.3. Multigrid methods are briefly recalled in Section 3.4, setting the background for the design of acceleration methods specifically tailored to network problems, that are described and tested in Section 3.5. The extension to more general problems is addressed in Section 3.6, and conclusions are drawn in Section 3.7.

3.2 The DDUQ method

We consider systems composed of interconnected subsystems (e.g. components, subdomains), which characterize a wide range of applications such as electrical power

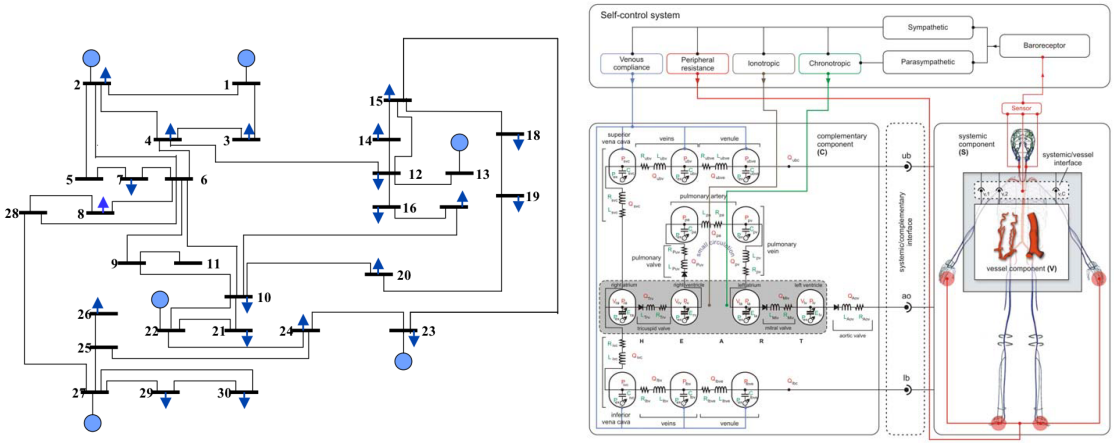


Figure 3.1: Examples of DDUQ applications to decomposable systems: 30-bus test system [118] (left), cardiovascular system [33] (right).

systems, gas transfer systems, multiscale models, or - more specifically for the interest of this work, the cardiovascular system (see Figure 3.1). The primary difference between classical DD methods for solving partial differential equations and the present UQ context lies in characterizing inter-subsystem compatibility. Classically, in a deterministic framework, inter-subsystem coupling variables correspond to the solution along the subdomain boundary: compatibility conditions enforce solution continuity across this boundary [154, 188]. In the UQ context, coupling variables correspond to random variables for data exchanged between subsystems. As such, compatibility conditions should set the statistical distance between matching coupling variables to zero, i.e., the probability density function (PDF) of a subsystem's output should match the associated PDF of the input to the coupled subsystem. For this purpose, we will consider a hierarchy of compatibility constraints: from high-fidelity (i.e.. functional equivalence) to low-fidelity (i.e., mean matching).

3.2.1 Problem Formulation

Assume we have n subsystems, each of which is characterized by random endogenous inputs $\boldsymbol{y}_i \in \mathbb{R}^{\mathsf{y}_i}$, outputs $\mathbf{x}_i \in \mathbb{R}^{\mathsf{x}_i}$, exogenous inputs $\boldsymbol{u}_i \in \mathbb{R}^{\mathsf{u}_i}$, each with a certain PDF, and a forward function $\boldsymbol{f}_i : \mathbb{R}^{\mathsf{y}_i} \times \mathbb{R}^{\mathsf{u}_i} \to \mathbb{R}^{\mathsf{x}_i}$ such that the forward function propagation task can be expressed as [45]

$$\mathbf{x}_i = \boldsymbol{f}_i(\boldsymbol{y}_i, \boldsymbol{u}_i), \quad i = 1, \ldots, n \tag{3.1}$$

(see Figure 3.2, left). At the system level, denote by $\boldsymbol{y} \in \mathbb{R}^{\mathsf{y}}$, $\mathbf{x} \in \mathbb{R}^{\mathsf{x}}$, and $\boldsymbol{u} \in \mathbb{R}^{\mathsf{u}}$ the vectorization of the subsystem inputs, outputs, and exogeneous inputs, respectively (e.g., $\boldsymbol{y} \coloneqq \begin{bmatrix}\boldsymbol{y}_1^T & \cdots & \boldsymbol{y}_n^T\end{bmatrix}^T$) with $\mathsf{y} \coloneqq \sum_{i=1}^n \mathsf{y}_i$, $\mathsf{x} \coloneqq \sum_{i=1}^n \mathsf{x}_i$, and $\mathsf{u} \coloneqq \sum_{i=1}^n \mathsf{u}_i$. Then, we can rewrite Eq. (3.1) as

$$\mathbf{x} = \boldsymbol{f}(\boldsymbol{y}, \boldsymbol{u}). \tag{3.2}$$

Here, we have defined the (system-level) forward uncertainty propagator $\boldsymbol{f} : \mathbb{R}^{\mathsf{y}} \times \mathbb{R}^{\mathsf{u}} \to \mathbb{R}^{\mathsf{x}}$ as the vectorization of subsystem propagators.

Outputs of one subsystem comprise the inputs to another subsystem (see Figure 3.2, right). This relationship can be encoded by the adjacency matrix

$$\boldsymbol{y} = \boldsymbol{I}_{\mathbf{x}}^{\boldsymbol{y}} \mathbf{x} \tag{3.3}$$

with $\boldsymbol{I}_{\mathbf{x}}^{\boldsymbol{y}} \in \{0, 1\}^{\mathsf{y} \times \mathsf{x}}$, whose entries are defined as

$$[\boldsymbol{I}_{\mathbf{x}}^{\boldsymbol{y}}]_{ij} = \begin{cases} 1 & \text{if } y_i = x_j \\ 0 & \text{otherwise.} \end{cases} \tag{3.4}$$

Substituting Eq. (3.3) into Eq. (3.2) yields a fixed point system

$$\mathbf{x} = \boldsymbol{f}(\boldsymbol{I}_{\mathbf{x}}^{\boldsymbol{y}} \mathbf{x}, \boldsymbol{u}). \tag{3.5}$$

Thus, the uncertainty propagation task reduces to solving the system of (generally nonlinear) equations

$$\mathbf{r}(\mathbf{x}, \boldsymbol{u}) = 0, \tag{3.6}$$

where

$$\mathbf{r} : (\boldsymbol{\chi}, \boldsymbol{v}) \mapsto \boldsymbol{\chi} - \boldsymbol{f}(\boldsymbol{I}_{\mathbf{x}}^{\boldsymbol{y}} \boldsymbol{\chi}, \boldsymbol{v}). \tag{3.7}$$

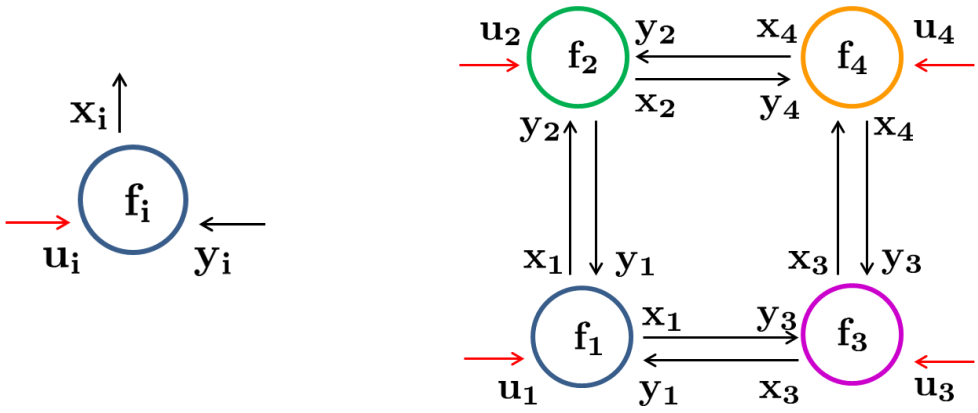


Figure 3.2: Sketch of DDUQ problem formulation: single component (left) and full decomposable system (right).

3.2.2 Uncertainty Propagation

In this work, uncertainties are represented via polynomial chaos expansions [83, 84], and propagated via *non-intrusive spectral projection* (NISP) [176]. Consider the stochastic inputs $\boldsymbol{y}$, $\boldsymbol{u}$ as functions of a multidimensional random variable $\boldsymbol{\xi} = [\xi_1, \dots, \xi_d]^T$. The corresponding stochastic representations via PCE, respectively, read as

$$\boldsymbol{y}(\boldsymbol{\xi}) = \sum_{k=0}^{p} y_k \Psi_k(\boldsymbol{\xi}), \qquad \boldsymbol{u}(\boldsymbol{\xi}) = \sum_{k=0}^{p} u_k \Psi_k(\boldsymbol{\xi}), \tag{3.8}$$

where $\{u_k\}_{k=0}^p$, $\{y_k\}_{k=0}^p$ are deterministic coefficients defined by L^2-projection, and the polynomial chaos $\{\Psi_k\}_{k\in\mathbb{N}}$ can be chosen to maximize the convergence rate, according to the Askey scheme [18, 200]. More precisely, each family of polynomials is orthogonal with respect to a weight function that corresponds to the probability density function of a known random distribution. For instance, assuming $\boldsymbol{\xi}$ to be a standard normal random variable, $\Psi_k(\cdot)$ are Hermite polynomials, orthogonal with respect to Gaussian density $\pi(\boldsymbol{\xi}) = e^{-\boldsymbol{\xi}^2/2}/\sqrt{2\pi}$. Then, uncertainty propagation consists in finding a PC representation for the output $\mathbf{x} = \sum_{k=0}^{p} x_k \Psi_k(\boldsymbol{\xi})$, as a response to the uncertain input through the propagator $\boldsymbol{f}$. More specifically, the

input/output relationship is encoded in the output PC coefficients as

$$x_k = \frac{1}{||\Psi_k||^2} \int_{\mathbb{R}^d} \mathbf{x}(\boldsymbol{\xi}) \Psi_k(\boldsymbol{\xi}) \pi(\boldsymbol{\xi}) d\boldsymbol{\xi} = \frac{1}{||\Psi_k||^2} \int_{\mathbb{R}^d} \boldsymbol{f}\left(\boldsymbol{y}(\boldsymbol{\xi}), \boldsymbol{u}(\boldsymbol{\xi})\right) \Psi_k(\boldsymbol{\xi}) \pi(\boldsymbol{\xi}) d\boldsymbol{\xi}, \quad (3.9)$$

where the norm is defined as $||\Psi_k||^2 = \int_{\mathbb{R}^d} \Psi_k^2(\boldsymbol{\xi}) \pi(\boldsymbol{\xi}) d\boldsymbol{\xi}$ and is known a priori.

The strategy for NISP (also called discrete or pseudospectral projection) is to approximate (3.9) with (sparse) quadrature rules as

$$x_k = \frac{1}{||\Psi_k||^2} \sum_{q=1}^{Q} \boldsymbol{f}\left(\boldsymbol{y}(\boldsymbol{\xi}_q), \boldsymbol{u}(\boldsymbol{\xi}_q)\right) \Psi_k(\boldsymbol{\xi}_q) \pi(\boldsymbol{\xi}_q) w_q, \quad (3.10)$$

where $\{\boldsymbol{\xi}_q, w_q\}_{q=1}^{Q}$ are selected quadrature points and weights, respectively. Note that, in (3.10), the term $\boldsymbol{f}\left(\boldsymbol{y}(\boldsymbol{\xi}_q), \boldsymbol{u}(\boldsymbol{\xi}_q)\right)$ is one deterministic solve, corresponding to the q-th realization of the inputs. Therefore, differently from other intrusive approaches such as Galerkin projection, NISP does not require any modification to the underlying deterministic solver, that can be used as a black-box.

3.2.3 Network solver

The system of nonlinear equations (3.6) can be solved by a variety of techniques. While Newton's method [109, 152] is the most common approach, it relies on availability of the gradient $\partial \mathbf{r} / \partial \boldsymbol{\chi}$, which, in this context, associates with computing the sensitivity of the output uncertainty representation with respect to the input uncertainty representation. Because this is frequently challenging to obtain, we consider classical relaxation methods (i.e., Jacobi, Gauss–Seidel, successive over-relaxation) [109, 152] that do not require gradients and promote independence of the subsystems.

Jacobi iteration. Jacobi iteration is equivalent to 'splitting' all edge data and allowing subsystems to simultaneously and independently perform uncertainty propagation at each iteration (see Figure 3.3, center). This can be interpreted as an additive Schwarz domain-decomposition approach [188] and corresponds to the iterations at the subsystem level at iteration k:

Algorithm 1: DDUQ Jacobi iteration

for *i=1,...,n* **do**
  $\bar{\mathbf{x}}_i = \boldsymbol{f}_i(\boldsymbol{y}_i^{(k)}, \boldsymbol{u}_i)$
end
$\mathbf{x}^{(k+1)} = \omega\bar{\mathbf{x}} + (1-\omega)\mathbf{x}^{(k)}$
$\boldsymbol{y}^{(k+1)} = \boldsymbol{I}_{\mathbf{x}}^{\boldsymbol{y}}\mathbf{x}^{(k+1)}$

Note that the subsystem forward uncertainty propagation tasks can be executed independently in parallel for each subsystem at a given iteration. At the full-system level, this can be expressed in terms of the outputs simply as

$$\bar{\mathbf{x}} = \boldsymbol{f}(\boldsymbol{I}_{\mathbf{x}}^{\boldsymbol{y}}\mathbf{x}^{(k)}, \boldsymbol{u}) \tag{3.11}$$

$$\mathbf{x}^{(k+1)} = \omega\bar{\mathbf{x}} + (1-\omega)\mathbf{x}^{(k)}, \tag{3.12}$$

where $\omega \in \mathbb{R}$ is a weighting factor that can be chosen to improve convergence.

Gauss–Seidel iteration. Gauss–Seidel iteration is equivalent to decomposing the network into a directed acyclical graph where each iteration corresponds to uncertainty propagation in a feed-forward network (see Figure 3.3, right). In particular, we can introduce an n-tuple (p_i) that provides a permutation of natural numbers one to n. Each permutation associates with a unique feed-forward network where p_1 corresponds to the furthest 'upstream' subsystem and p_n is the furthest 'downstream' subsystem. This approach can be interpreted as a multiplicative Schwarz domain-decomposition approach, and corresponds to the following algorithm at iteration k:

Algorithm 2: DDUQ Gauss-Seidel iteration

$\boldsymbol{y}^{(k+1)} \leftarrow \boldsymbol{y}^{(k)}$
for *i=1,...,n* **do**
  $\mathbf{x}_{p_i}^{(k+1)} = \boldsymbol{f}_{p_i}(\boldsymbol{E}_{p_i}^{(\mathrm{y}_i)}\boldsymbol{y}^{(k+1)}, \boldsymbol{u}_{p_i})$
  $\boldsymbol{y}^{(k+1)} \leftarrow \boldsymbol{I}_{\mathbf{x}}^{\boldsymbol{y}}[\boldsymbol{E}_{p_i}^{(\mathrm{x}_i)}]^T\mathbf{x}_{p_i}^{(k+1)}$
end

To develop the global-problem description of the Gauss–Seidel iteration, first we define a block decomposition of the adjacency matrix

$$\boldsymbol{P}^{(\mathsf{y}_i)}_{(p_i)}\boldsymbol{I}^{\boldsymbol{y}}_{\mathbf{x}}[\boldsymbol{P}^{(\mathsf{x}_i)}_{(p_i)}]^T = \boldsymbol{L} + \boldsymbol{D} + \boldsymbol{U}, \tag{3.13}$$

where the permutation matrix is defined as

$$\boldsymbol{P}^{(\boldsymbol{w}_i)}_{(q_i)} \coloneqq \begin{bmatrix} \boldsymbol{E}^{(\boldsymbol{w}_1)}_{q_1} \\ \vdots \\ \boldsymbol{E}^{(\boldsymbol{w}_n)}_{q_n} \end{bmatrix} \in \{0,1\}^{\sum_{i=1}^n \mathsf{w}_i \times \sum_{i=1}^n \mathsf{w}_i} \tag{3.14}$$

with $\boldsymbol{P}^{(\boldsymbol{w}_i)^T}_{(q_i)}\boldsymbol{P}^{(\boldsymbol{w}_i)}_{(q_i)} = \boldsymbol{P}^{(\boldsymbol{w}_i)}_{(q_i)}\boldsymbol{P}^{(\boldsymbol{w}_i)^T}_{(q_i)} = \boldsymbol{I}$. Further, $\boldsymbol{L} \in \{0,1\}^{\mathsf{y}\times\mathsf{x}}$ is strictly block lower triangular (with block (i,j) defined as a $\mathsf{y}_i \times \mathsf{x}_j$ submatrix), $\boldsymbol{D} \in \{0,1\}^{\mathsf{y}\times\mathsf{x}}$ is block diagonal, and $\boldsymbol{U} \in \{0,1\}^{\mathsf{y}\times\mathsf{x}}$ is strictly block upper triangular. Using these quantities, we can write this iteration in terms of the global outputs as

$$\bar{\mathbf{x}} = \boldsymbol{f}([\boldsymbol{P}^{(\mathsf{y}_i)}_{(p_i)}]^T\boldsymbol{L}\boldsymbol{P}^{(\mathsf{x}_i)}_{(p_i)}\bar{\mathbf{x}} + [\boldsymbol{P}^{(\mathsf{y}_i)}_{(p_i)}]^T(\boldsymbol{D}+\boldsymbol{U})\boldsymbol{P}^{(\mathsf{x}_i)}_{(p_i)}\mathbf{x}^{(k)}, \boldsymbol{u}). \tag{3.15}$$

Successive over-relaxation (SOR). This formulation is similar to Gauss–Seidel iteration, but allows for a relaxation factor. In particular, this method corresponds to the following algorithm at iteration k:

Algorithm 3: DDUQ SOR iteration

$\bar{\boldsymbol{y}} \leftarrow \boldsymbol{y}^{(k)}$
for *i=1,...,n* **do**
  $\bar{\mathbf{x}}_{p_i} = \boldsymbol{f}_{p_i}(\bar{\boldsymbol{y}}_{p_i}, \boldsymbol{u}_{p_i})$
  $\mathbf{x}^{(k+1)}_{p_i} = \omega\bar{\mathbf{x}}_{p_i} + (1-\omega)\mathbf{x}^{(k)}_{p_i}$
  $\boldsymbol{y}^{(k+1)}_{p_i} \leftarrow \boldsymbol{E}^{(\mathsf{y}_i)}_{p_i}\boldsymbol{I}^{\boldsymbol{y}}_{\mathbf{x}}[\boldsymbol{E}^{(\mathsf{x}_i)}_{p_i}]^T\mathbf{x}^{(k+1)}_{p_i}$
  $\bar{\boldsymbol{y}}_{p_i} \leftarrow \boldsymbol{y}^{(k+1)}_{p_i}$
end

The corresponding global problem is

$$\bar{\mathbf{x}} = \boldsymbol{f}(\omega[\boldsymbol{P}^{(\mathsf{y}_i)}_{(p_i)}]^T \boldsymbol{L}\boldsymbol{P}^{(\mathsf{x}_i)}_{(p_i)}\bar{\mathbf{x}} + [\boldsymbol{P}^{(\mathsf{y}_i)}_{(p_i)}]^T((1-\omega)\boldsymbol{L} + \boldsymbol{D} + \boldsymbol{U})\boldsymbol{P}^{(\mathsf{x}_i)}_{(p_i)}\mathbf{x}^{(k)}, \boldsymbol{u}) \tag{3.16}$$

$$\mathbf{x}^{(k+1)} = \omega\bar{\mathbf{x}} + (1-\omega)\mathbf{x}^{(k)}. \tag{3.17}$$

Note that the Gauss–Seidel method is recovered for $\omega = 1$, whereas $\omega < 1$ corresponds to underrelaxation, and $\omega > 1$ corresponds to overrelaxation.

As in typical iterative methods, we expect different permutations (p_i) to associate with different convergence rates and different degrees of exposed parallelism; for example, red–black ordering is known to expose additional parallelism in typical iterative methods, with implications on convergence. More in general, the optimal permutation can be obtained via graph coloring techniques applied to the network topology (note that the solution may not be unique) [90, 130]. Further, we note that the permutation can change between Gauss–Seidel or SOR iterations. For example, symmetric Gauss–Seidel or SOR can be recovered by reversing the order of permutation between iterations; this amounts to reversing the information flow in the uncertainty propagation between iterations.

3.2.4 Software

The solver is characterized by a nested structure. At the outer level, the network solver receives in input the global (stochastic) BC and the (uncertain) model parameters, and handles the relaxation iterative process until convergence. At each iteration, the exogenous inputs specific to each component are selected, and the endogenous inputs are exchanged. With a block-Jacobi network solver, since all the components are decoupled, the endogenous interface conditions can be exchanged simultaneously; with a Gauss-Seidel or SOR network solver, the order in which they are exchanged depends on the permutation of the network components, which can be picked, for example, to maximize the level of concurrency. Then, for each component, the local stochastic solver evaluates the Q realizations of the stochastic inputs, and calls the deterministic solver for each instance of the inputs. Finally, the output

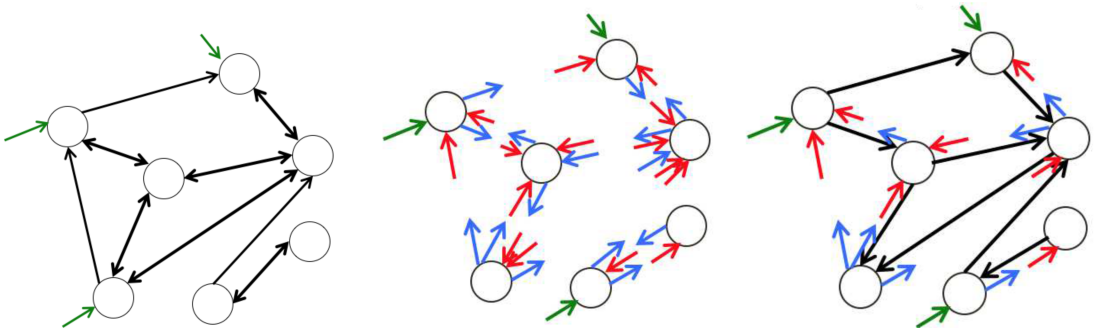


Figure 3.3: Sketch of DDUQ iterative solvers: monolithic (left), Jacobi (center) and Gauss-Seidel (right) iteration. Green, red, and blue arrows respectively denote exogenous inputs, endogenous inputs, and subsystem outputs. Black arrows in the right figure represent the feed-forward network generated by a Gauss-Seidel solver with a particular choice of components permutation.

PC coefficients are computed according to (3.10), and the termination criterion is checked by the network solver. The process is repeated until convergence.

The stochastic component of the problem is handled via the C++ library `UQTk` [61], developed at the Sandia National Laboratories, which has been embedded in a `MATLAB` implementation of the DDUQ network solver. The solver structure is sketched in figure 3.4.

3.3 Numerical tests

3.3.1 1D heat equation

In this section, numerical results are presented for the one-dimensional stationary heat equation with uncertain boundary conditions characterized by two-dimensional

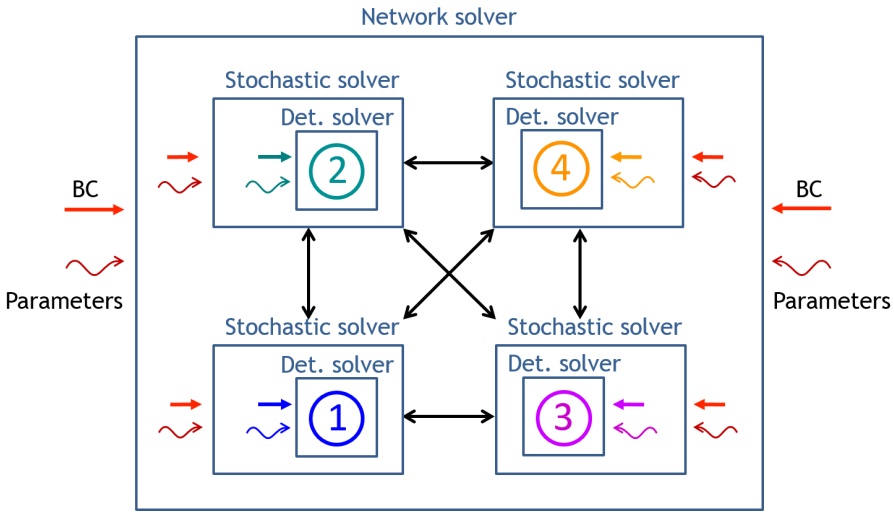


Figure 3.4: Sketch of the DDUQ network solver. Straight and wavy arrows represent exogenous stochastic boundary conditions (BC) and parameters, respectively.

PCEs:

$$-\frac{\partial^2 u\left(x,\boldsymbol{\xi}\right)}{\partial x^2}=0\ , \qquad x\in\Omega:=(0,L) \tag{3.18}$$

$$u\left(0,\boldsymbol{\xi}\right)=T_1\left(\boldsymbol{\xi}\right)=\sum_{k=0}^{K}T_{1,k}\Psi_k\left(\boldsymbol{\xi}\right) \tag{3.19}$$

$$u\left(L,\boldsymbol{\xi}\right)=T_2\left(\boldsymbol{\xi}\right)=\sum_{k=0}^{K}T_{2,k}\Psi_k\left(\boldsymbol{\xi}\right) \tag{3.20}$$

with $\boldsymbol{\xi}=(\xi_1,\xi_2)$, ξ_1 and ξ_2 independent and identically distributed (iid) standard normal random variables. The PCE coefficients $T_{1,\cdot}$ and $T_{2,\cdot}$ completely characterize the uncertain boundary conditions. We will consider the case of statistically independent random variables T_1 and T_2 with T_1 expressed strictly in terms of ξ_1 and T_2 expressed strictly in terms of ξ_2. The basis functions $\{\Psi_k\}$ are 2D Hermite polynomials of order at most p. This problem permits an analytical solution given by

$$u(x,\boldsymbol{\xi})=\sum_{k=0}^{K}\left(\left(T_{2,k}-T_{1,k}\right)x/L+T_{1,k}\right)\Psi_k\left(\boldsymbol{\xi}\right)\ . \tag{3.21}$$

Note that solution given by Eq. (3.21) satisfies Eq. (3.18) in a strong sense, i.e., it satisfies the ODE for all values of $\boldsymbol{\xi}$. This is not what we expect the solver to provide in the general sense, but we do require a solution that satisfies the governing equations projected onto the PCE polynomials via Galerkin projection. This analytical solution is convenient for code verification in terms of solution matching. This linear system also allows us to compare the solver convergence rate with ones predicted based on system eigenvalue analysis.

Domain decomposition setting

We decompose the domain of interest into n overlapping subdomains Ω_i, $i = 1, \ldots, n$, as illustrated in Fig. 3.5. As a result, boundary value problem (BVP) in Eq. (3.18) is split into n smaller boundary value problems. Let u_i denote the approximate solution in subdomain Ω_i. Furthermore, we will use $u_{i,1}$ to denote u_i at the the artificial internal boundary $\Gamma_{i,i-1}$, and $u_{i,2}$ to denote u_i at the the artificial internal boundary $\Gamma_{i,i+1}$. The solution to Eq. (3.18) is obtained by solving the following equation for each subdomain Ω_i:

$$-\frac{\partial^2 u_i\left(x,\boldsymbol{\xi}\right)}{\partial x^2} = 0 \quad \text{in} \quad \Omega_i \tag{3.22}$$

$$u_i\left(\boldsymbol{\xi}\right) = \begin{cases} T_1\left(\boldsymbol{\xi}\right), & i = 1 \\ u_{i-1,2}\left(\boldsymbol{\xi}\right), & \text{otherwise} \end{cases} \quad \text{on} \quad \Gamma_{i,i-1} \tag{3.23}$$

$$u_i\left(\boldsymbol{\xi}\right) = \begin{cases} T_2\left(\boldsymbol{\xi}\right), & i = n \\ u_{i+1,1}\left(\boldsymbol{\xi}\right), & \text{otherwise} \end{cases} \quad \text{on} \quad \Gamma_{i,i+1}\ . \tag{3.24}$$

Note that the subdomain-level BVP on subdomain Ω_i in Eq. (3.22) depends on inputs $u_{i-1,2}\left(\boldsymbol{\xi}\right)$ and $u_{i+1,1}\left(\boldsymbol{\xi}\right)$ from adjacent subdomains in the form of uncertain boundary conditions and thus the BVPs are coupled. These inputs to subdomain Ω_i are outputs from the adjacent subdomains: $u_{i-1,2}\left(\boldsymbol{\xi}\right)$ is an output of subdomain Ω_{i-1} and $u_{i+1,1}\left(\boldsymbol{\xi}\right)$ is an output of subdomain Ω_{i+1}. This coupling can be illustrated using a directional graph, as illustrated for the 4-subdomain case in Fig. 3.6. We

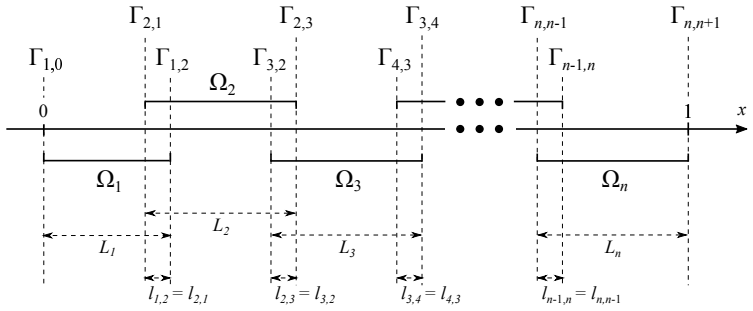


Figure 3.5: Overlapping domain decomposition for 1D heat equation.

will represent each of these inputs and outputs using PCE, i.e.

$$u_{i,j} = \sum_{k=0}^{K} u_{i,j,k} \Psi_k \left(\boldsymbol{\xi} \right), \qquad i = 1, \ldots, n \quad j = 1, 2, \tag{3.25}$$

and thus all inputs and outputs are in the form of PCE coefficients. Using notation introduced above, we have the following input and output PCE coefficients for subdomain i

$$\boldsymbol{y}_i = \begin{cases} \left[u_{2,1,1}, u_{2,1,2} \cdots \ u_{2,1,K} \right]^T, & i = 1 \\ \left[u_{n-1,2,1}, u_{n-1,2,2} \cdots \ u_{n-1,2,K} \right]^T, & i = n \\ \left[u_{i-1,2,1}, u_{i-1,2,2} \cdots \ u_{i-1,2,K}, u_{i+1,1,1}, u_{i+1,1,2} \cdots \ u_{i+1,1,K} \right]^T, & \text{otherwise,} \end{cases} \tag{3.26}$$

$$\mathbf{x}_i = \begin{cases} \left[u_{1,2,1}, u_{1,2,2} \cdots \ u_{1,2,K} \right]^T, & i = 1 \\ \left[u_{n,1,1}, u_{n,1,2} \cdots \ u_{n,1,K} \right]^T, & i = n \\ \left[u_{i,1,1}, u_{i,1,2} \cdots \ u_{i,1,K}, u_{i,2,1}, u_{i,2,2} \cdots \ u_{i,2,K} \right]^T, & \text{otherwise,} \end{cases} \tag{3.27}$$

$$\boldsymbol{u}_i = \begin{cases} \left[T_{1,1}, T_{1,2} \cdots \ T_{1,K} \right]^T, & i = 1 \\ \left[T_{2,1}, T_{2,2} \cdots \ T_{2,K} \right]^T, & i = n \\ [\,], & \text{otherwise.} \end{cases} \tag{3.28}$$

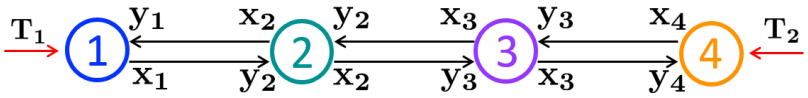


Figure 3.6: Directional graph to represent domain decomposition for 1D heat equation, with the nodes representing subdomain-level BVPs and the links representing the exchanged information in the form of subdomain-level uncertain artificial boundary values in the form of PCE coefficients.

For a general application, the input uncertainty is propagated through each subdomain-specific solver using NISP, defining the forward uncertainty propagator $\boldsymbol{f}_i$ for subdomain i. For this specific problem, the analytical solution in Eq. (3.21) could be used at the subdomain level as a propagator in lieu of the more general NISP-based method. The closed-form input-output mappings are given by

$$\mathbf{x}_i = \boldsymbol{f}_i(\boldsymbol{y}_i, \boldsymbol{u}_i) := \begin{cases} u_{2,1,k} = \frac{L_1 - l_{1,2}}{L_1} u_{1,2,k} + \frac{l_{1,2}}{L_1} T_{1,k}, \ k = 1, \ldots, K, & i = 1 \\ u_{n-1,2,k} = \frac{L_n - l_{n,n-1}}{L_n} u_{n,1,k} + \frac{l_{n,n-1}}{L_n} T_{2,k}, \ k = 1, \ldots, K, & i = n \\ \begin{cases} u_{i-1,2,k} = \frac{L_i - l_{i,i-1}}{L_i} u_{i,1,k} + \frac{l_{i,i-1}}{L_i} u_{i,2,k}, \ k = 1, \ldots, K, \\ u_{i+1,1,k} = \frac{l_{i,i+1}}{L_i} u_{i,1,k} + \frac{L_i - l_{i,i+1}}{L_i} u_{i,2,k}, \ k = 1, \ldots, K \end{cases} & \text{otherwise.} \end{cases} \tag{3.29}$$

Scalability tests

For this example, we assume Gaussian inputs with the following uncertainty characterization:

$$\begin{aligned} [T_{1,0}, T_{1,1} \cdots \ T_{1,9}] &= [1.0, \ 0.1, \ 0, \ 0.01, \ 0, \ 0, \ 0.001, \ 0, \ 0, \ 0], \\ [T_{2,0}, T_{2,1} \cdots \ T_{2,9}] &= [3.0, \ 0, \ 0.1, \ 0, \ 0, \ 0, \ 0.01, \ 0, \ 0, \ 0.001], \end{aligned} \tag{3.30}$$

mapping to the following PCEs for $T_1(\boldsymbol{\xi})$ and $T_2(\boldsymbol{\xi})$:

$$\begin{aligned} T_1(\boldsymbol{\xi}) &= 1.0\Psi_0(\boldsymbol{\xi}) + 0.1\Psi_1(\boldsymbol{\xi}) + 0.01\Psi_3(\boldsymbol{\xi}) + 0.001\Psi_6(\boldsymbol{\xi}), \\ T_2(\boldsymbol{\xi}) &= 3.0\Psi_0(\boldsymbol{\xi}) + 0.1\Psi_2(\boldsymbol{\xi}) + 0.01\Psi_5(\boldsymbol{\xi}) + 0.001\Psi_9(\boldsymbol{\xi}), \end{aligned} \tag{3.31}$$

where the polynomial chaos in (3.31) are Hermite polynomials of third order in two variables:

$$\begin{aligned}
&\Psi_0(\boldsymbol{\xi}) = 1, && \Psi_1(\boldsymbol{\xi}) = \xi_1, && \Psi_2(\boldsymbol{\xi}) = \xi_2 \\
&\Psi_3(\boldsymbol{\xi}) = \xi_1^2 - 1, && \Psi_4(\boldsymbol{\xi}) = \xi_1\xi_2, && \Psi_5(\boldsymbol{\xi}) = \xi_2^2 - 1, \\
&\Psi_6(\boldsymbol{\xi}) = \xi_1^3 - 3\xi_1, && \Psi_7(\boldsymbol{\xi}) = (\xi_1^2 - 1)\xi_2, && \Psi_8(\boldsymbol{\xi}) = \xi_1(\xi_2^2 - 1), \quad \Psi_9(\boldsymbol{\xi}) = \xi_2^3 - 3\xi_2,
\end{aligned} \tag{3.32}$$

after the Gaussianity assumption. As a result, the input probability density functions are as shown in Fig. 3.8. Note that the above characterization results in independent $T_1(\boldsymbol{\xi})$ and $T_2(\boldsymbol{\xi})$, although the overall formulation is general and can in principle include dependent BCs. Fig. 3.7 provides normalized residual convergence results regarding strong and weak scalability, for different relaxation methods described above. For weak scalability, we kept the subdomain problem size (i.e. subdomain-level finite-element nodes) fixed while increasing the number of subdomains. For strong scalability, we kept the global problem size (i.e. number of global finite-element nodes) fixed while increasing the number of subdomains.

The results (for strong and weak scalability cases) are identical for the cases involving 16 nodes (subdomains), since they involve the same network with the same discretization (subdomain overlaps), regardless of method and relaxation used. As the subdomain number decreases to 8, then to 4 and finally 2, the amount of overlap relative to the global domain size is smaller for the strong scalability cases, resulting in the observed inferior convergence rates for the same method and relaxation.

Figs. 3.9 and 3.10 provide normalized error convergence results relating to the output PC coefficients at specific physical locations, shown for both strong and weak scalability. Again, the results (for strong and weak scalability cases) are identical, at a given physical location, for the cases involving 16 nodes (subdomains) since the amount of chosen overlap is the same. Also, as the number of subdomains decreases, the amount of overlap relative to the global domain size is smaller for the strong scalability cases, resulting in the observed inferior convergence rates for the same method and relaxation. We can also see that, for a specific choice of number of

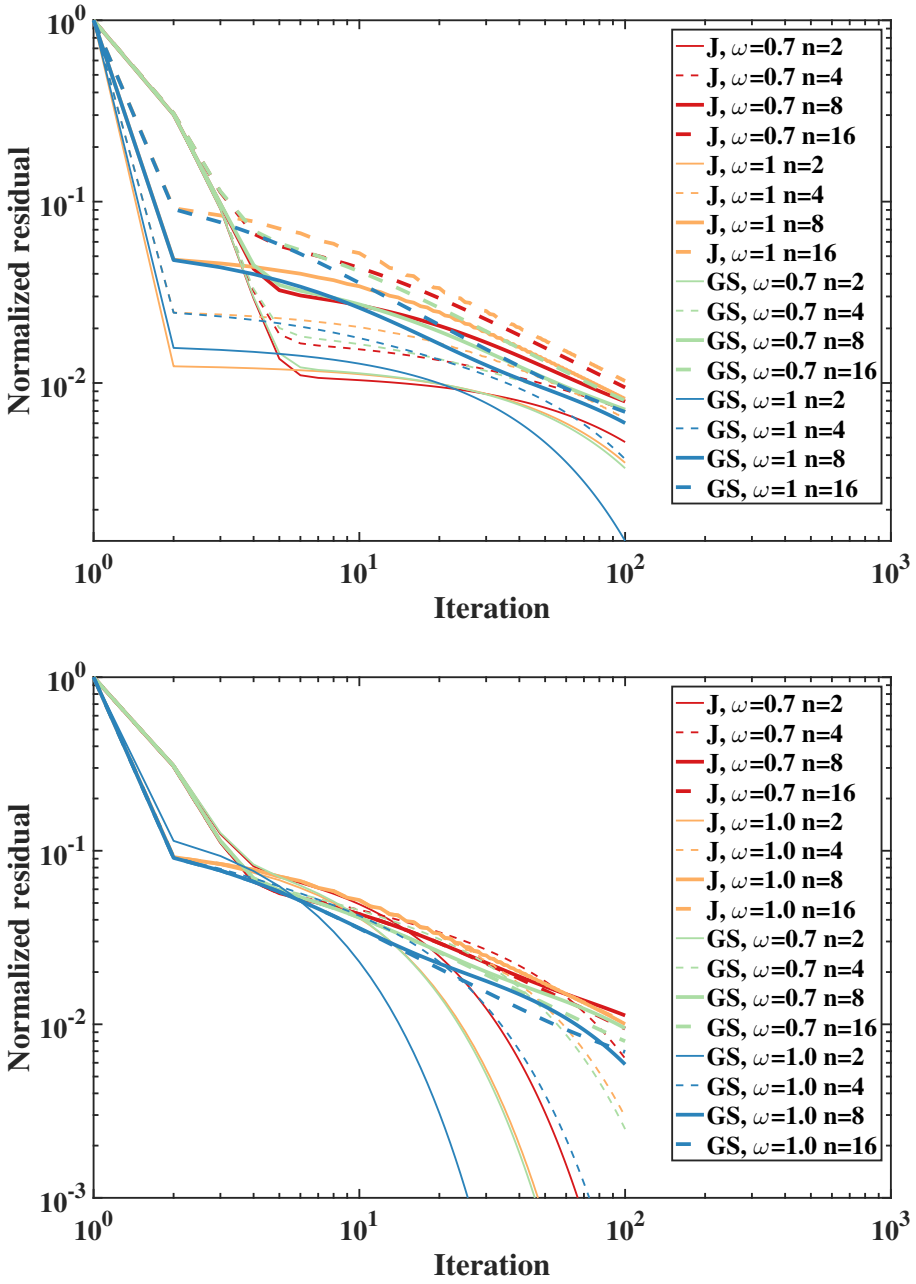


Figure 3.7: Strong (top) and weak (bottom) scalability convergence results for 1D heat equation: Iteration number versus normalized residual of interface unknowns (PC coefficients). J or G-S labels denote the use of Jacobi or Gauss-Seidel iterations, respectively.

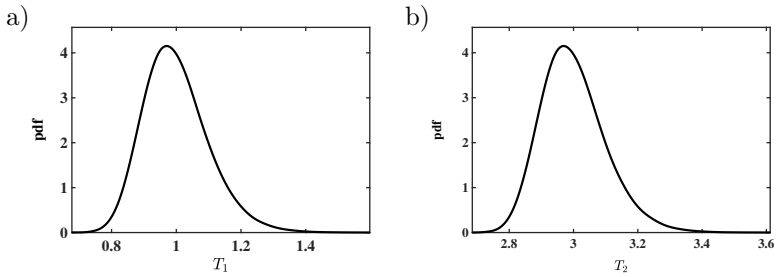


Figure 3.8: Probability density functions of uncertain inputs, i.e. boundary conditions, for the 1D heat equation.

subdomains, method, and relaxation, convergence rate is lower for locations further away from the boundary of the computational domain. This is expected since those locations are both physically and network distance-wise further away from the sources of uncertainty and thus a larger number of iterations is required to propagate the uncertainty to those locations through the network.

For validation purposes, we construct the system-wide propagation matrix (available for this linear problem with analytical subdomain level propagators, but not generally available) and apply the same methods and relaxation with the subsequent convergence results shown Figs. 3.11 and 3.12. These results are both qualitatively (and quantitatively) matching.

Figs. 3.13 and 3.14 provide strong and weak scalability results, with both serial and parallel execution timing provided. For all scalability results, all timings are obtained by performing calculations on an Intel(R) Xeon(R) Core(TM) i7-5557U CPU @ 3.10GHz with 16 GB RAM. The parallel execution time is the time that one node takes in propagating the uncertainty (since all nodes are similar in this case) for the Jacobi iterations. As for the Gauss-Seidel iterations, the parallel execution time is obtained through repeated simulations with varying permutation matrices to illustrate its effect on the computational scalability of these solvers. In this case, the permutation matrix for Gauss-Seidel that provides optimal parallel scalability is the one that results in updating (propagating uncertainty through) all the even nodes

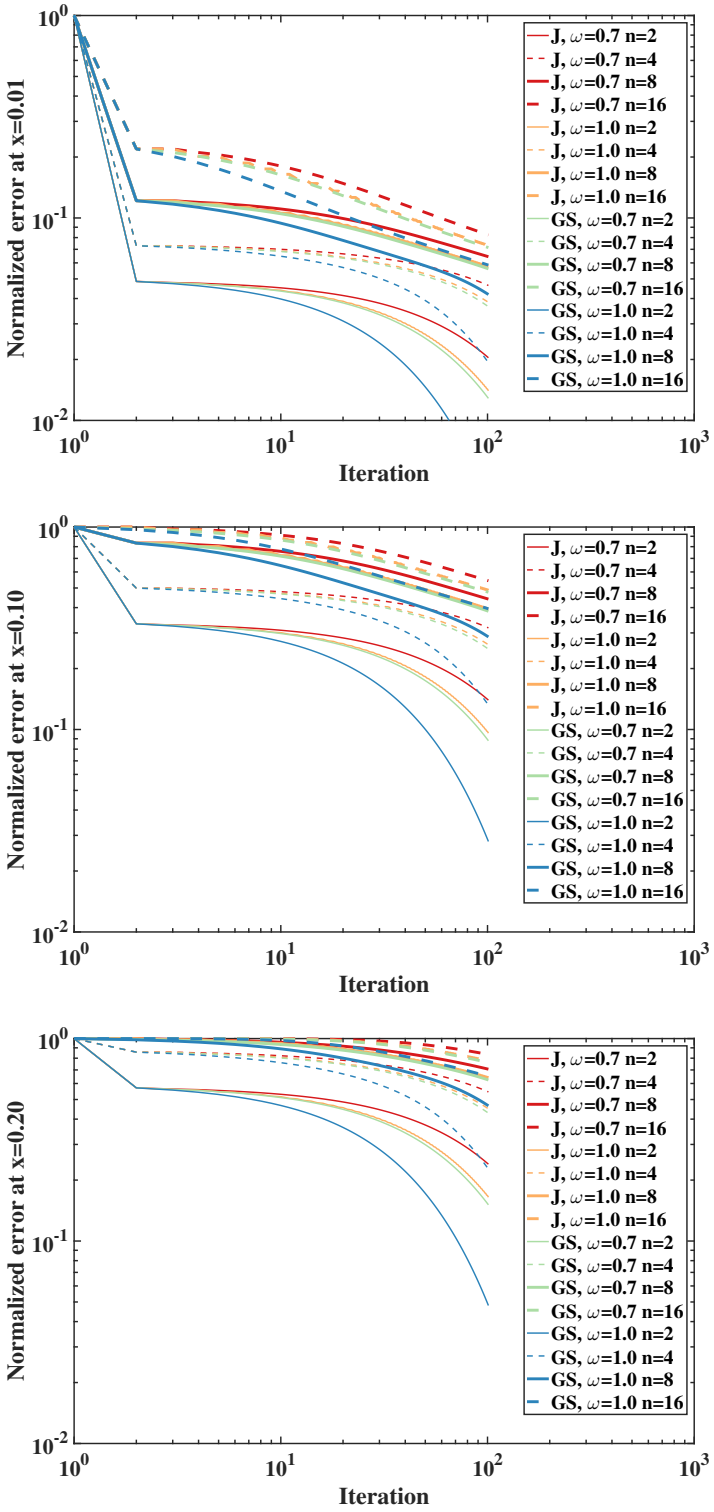


Figure 3.9: Strong scalability convergence results for 1D heat equation: Iteration number versus normalized error for solution (PC coefficients) at $x = 0.0187$ (top), $x = 0.1$ (center), and $x = 0.4$ (bottom). J or G-S labels denote the use of Jacobi or Gauss-Seidel iterations, respectively.

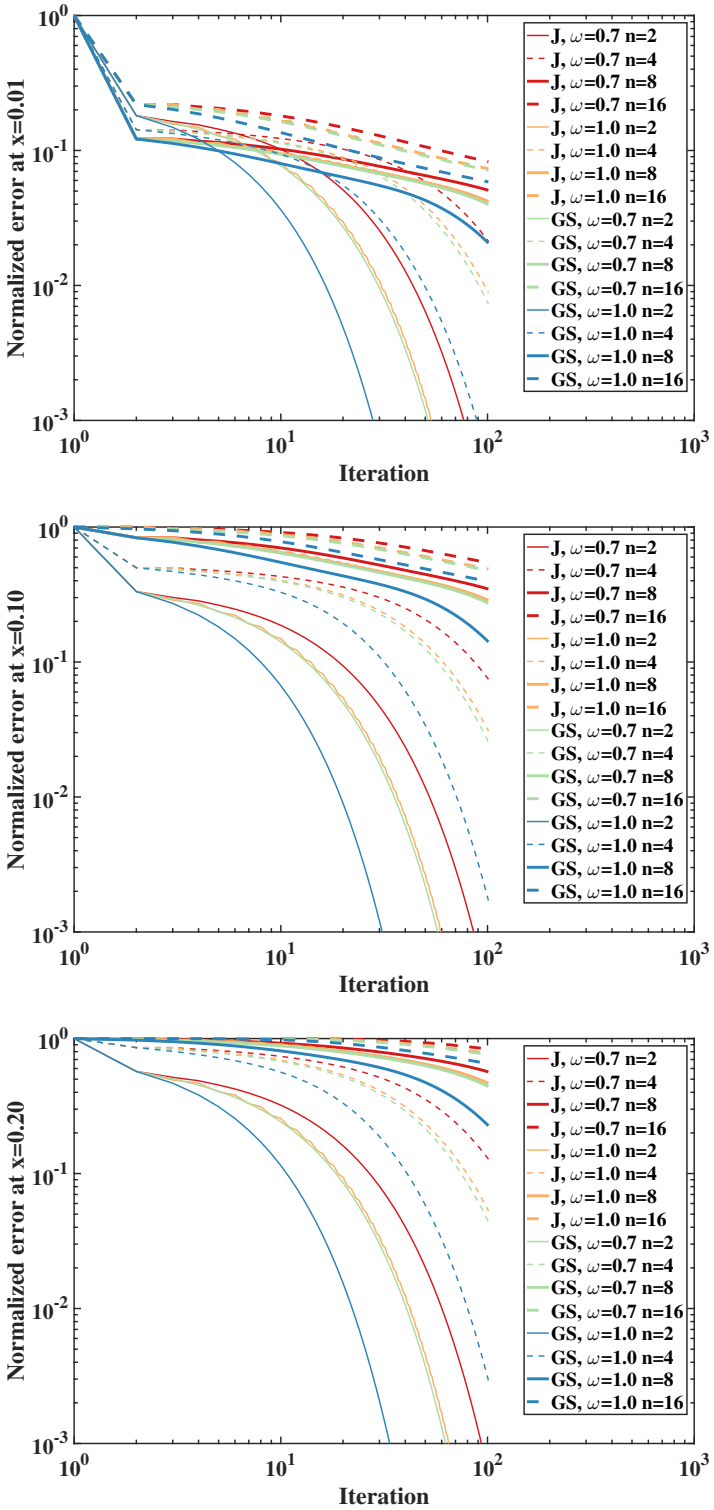


Figure 3.10: Weak scalability convergence results for 1D heat equation: Iteration number versus normalized error for solution (PC coefficients) at $x = 0.0187$ (top), $x = 0.1$ (center), and $x = 0.4$ (bottom). J or G-S labels denote the use of Jacobi or Gauss-Seidel iterations, respectively.

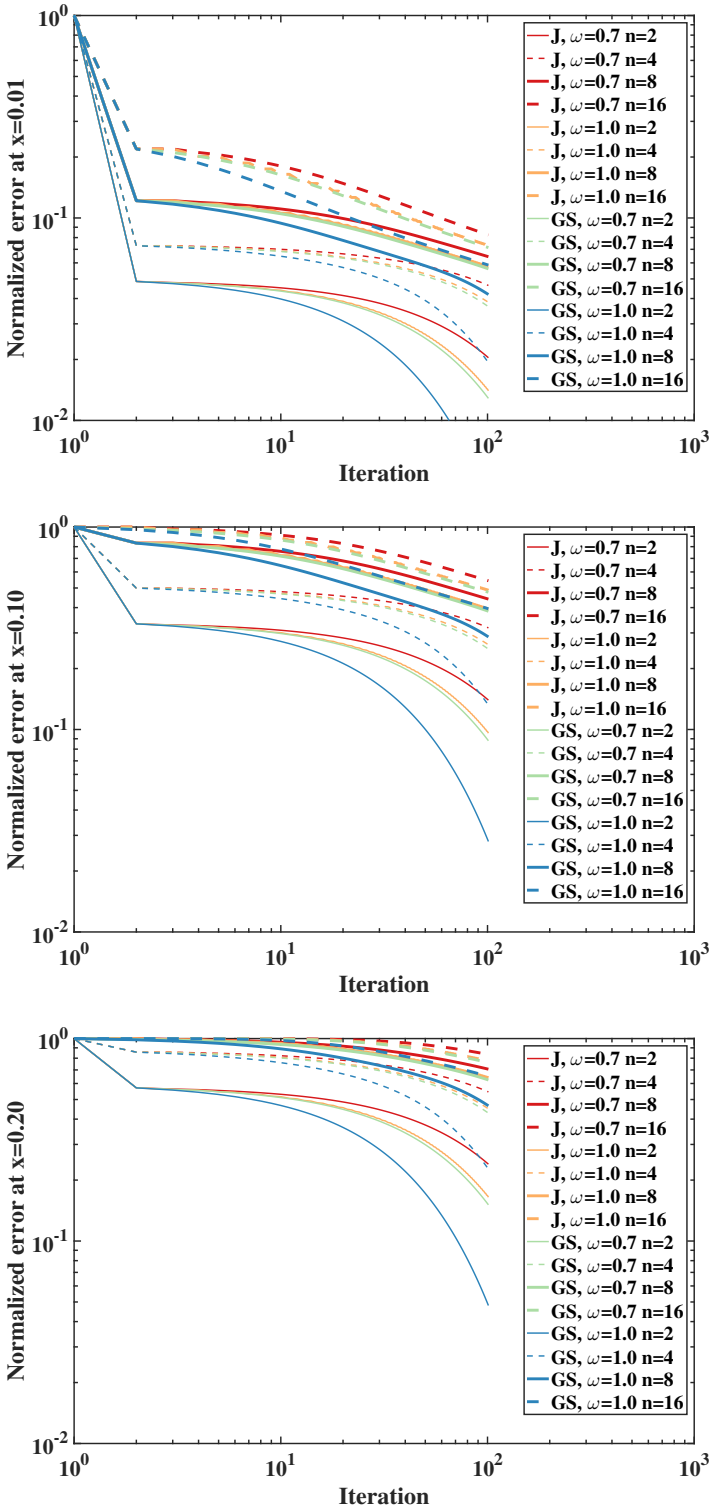


Figure 3.11: Analytical strong convergence results for 1D heat equation: Iteration number versus normalized error for solution (PC coefficients) at $x = 0.0187$ (top), $x = 0.1$ (center), and $x = 0.4$ (bottom). J or G-S labels denote the use of Jacobi or Gauss-Seidel iterations, respectively.

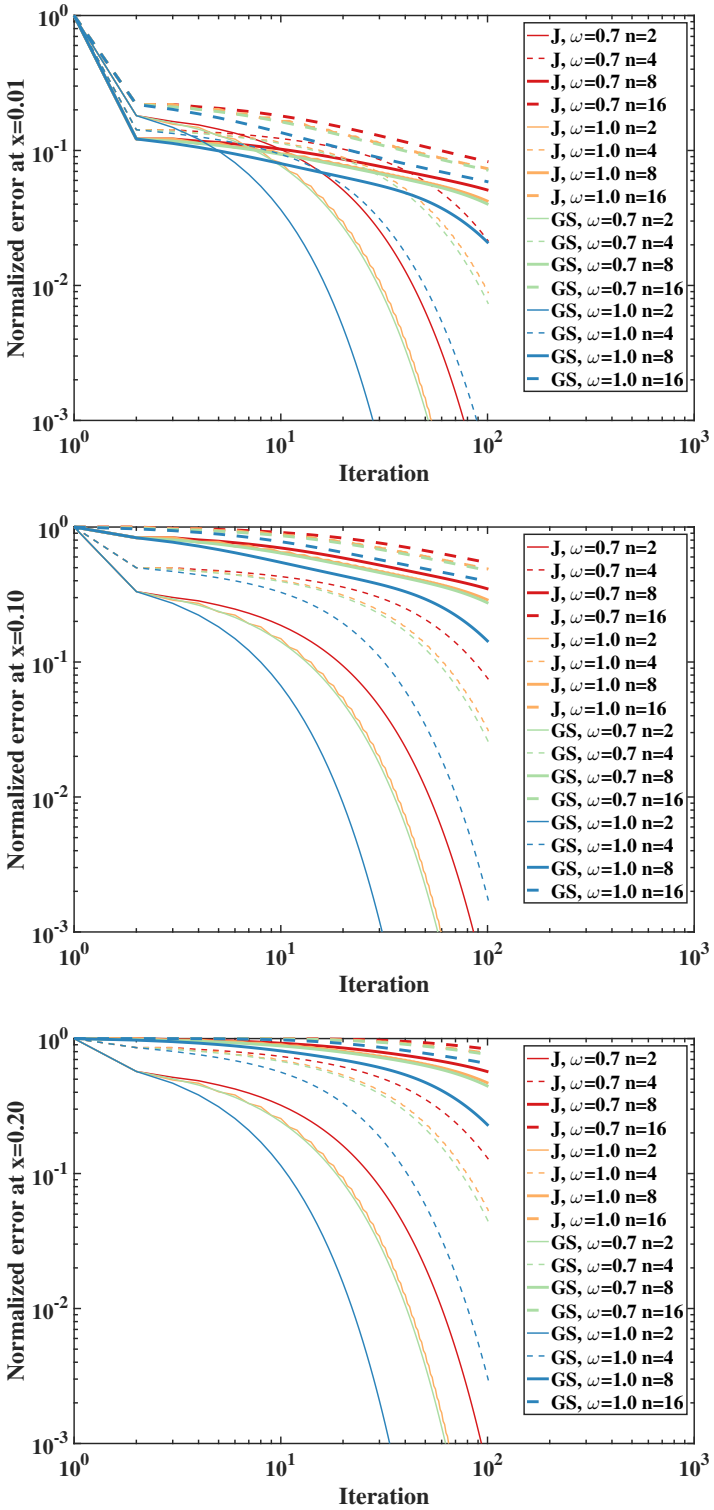


Figure 3.12: Analytical weak convergence results for 1D heat equation: Iteration number versus normalized error for solution (PC coefficients) at $x = 0.0187$ (top), $x = 0.1$ (center), and $x = 0.4$ (bottom). J or G-S labels denote the use of Jacobi or Gauss-Seidel iterations, respectively.

(subdomains) using known outputs of odd nodes and then subsequently updating the odd nodes using the new outputs from the even nodes. This is known as red-black ordering [Strang, 2016].

In summary, the numerical tests for a 1D linear heat equation show that the DDUQ solver is strongly and weakly scalable, both in its additive (Jacobi) and multiplicative (Gauss-Seidel) version, and for a wide range of relaxation parameters. While Jacobi iterations are embarassingly parallelizable, the performance of Gauss-Seidel solvers is greatly affected by the permutation of the nodes in the corresponding feed-forward network (with red-black ordering being optimal). As expected, the convergence rate is space-dependent, being slower for those locations that are far away from the sources of uncertainty (boundaries).

3.3.2 2D nonlinear heat equation

In this section, numerical results are presented for the two-dimensional stationary nonlinear heat equation with uncertain diffusion and boundary condition and characterized by:

$$-\nabla u + (e^{\mu u} - 1) = 10 \sin(2\pi x_1) \sin(2\pi x_2) \ , \qquad x \in \Omega := (0,1)^2 \tag{3.33}$$

$$u_\Gamma = T_1(\boldsymbol{\xi}) = \sum_{k=0}^{K} T_{1,k} \Psi_k(\boldsymbol{\xi}) \tag{3.34}$$

$$\mu = T_2(\boldsymbol{\xi}) = \sum_{k=0}^{K} T_{2,k} \Psi_k(\boldsymbol{\xi}) \tag{3.35}$$

with $\boldsymbol{\xi} = (\xi_1, \xi_2)$, ξ_1 and ξ_2 independent and identically distributed (iid) standard normal random variables. The PCE coefficients $T_{1,\cdot}$ and $T_{2,\cdot}$ completely characterize the uncertain boundary condition and diffusivity parameter, respectively. We will consider the case of statistically independent random variables T_1 and T_2 with T_1 expressed strictly in terms of ξ_1 and T_2 expressed strictly in terms of ξ_2. Under the assumption of Gaussian inputs, the basis functions $\{\Psi_k\}$ are 2D Hermite polynomials of order at most p.

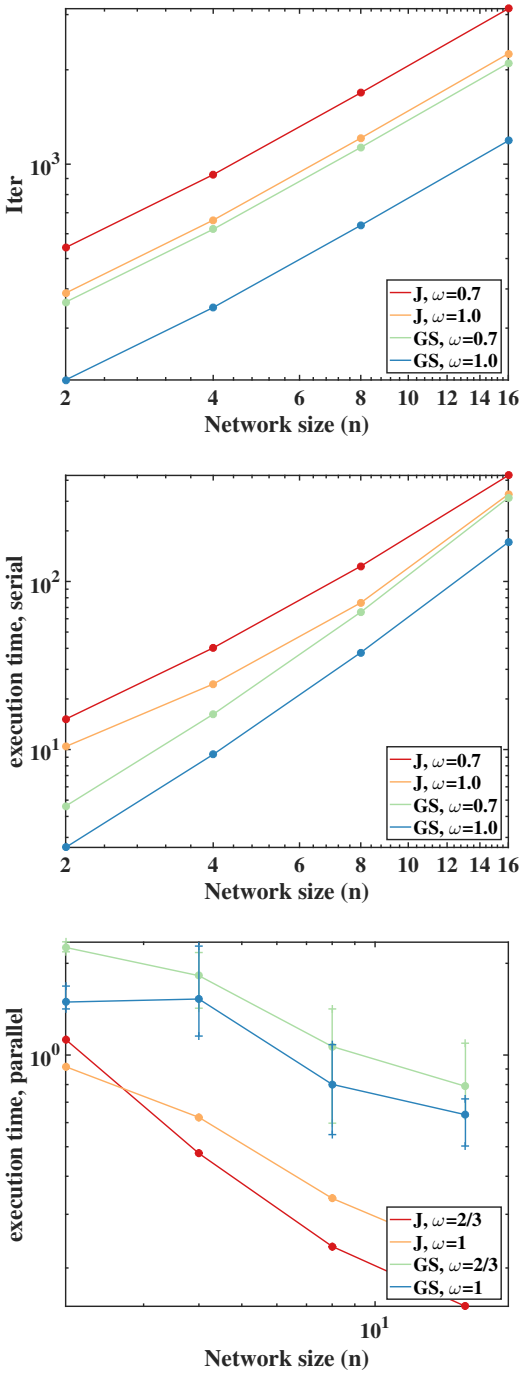


Figure 3.13: Strong scalability results for 1D heat equation: iteration count versus network size (number of subdomains) until convergence with normalized residual of 1×10^{-3} (top), serial execution time of solver versus network size (number of subdomains) until convergence with normalized residual of 1×10^{-3} (center), parallel execution time of solver versus network size (number of subdomains) until convergence with normalized residual of 1×10^{-3} with bars indicating spread due to varying permutation matrix (bottom). J or G-S label denotes the use of Jacobi or Gauss-Seidel iterations, respectively.

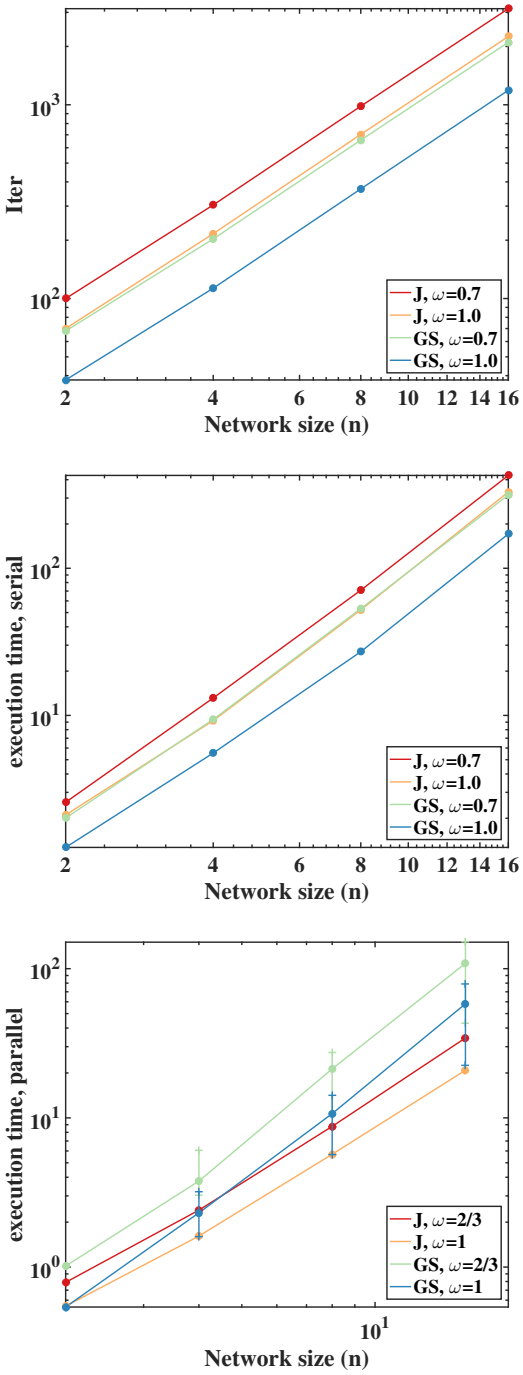


Figure 3.14: Weak scalability results for 1D heat equation: iteration count versus network size (number of subdomains) until convergence with normalized residual of 1×10^{-3} (top), serial execution time of solver versus network size (number of subdomains) until convergence with normalized residual of 1×10^{-3} (center), parallel execution time of solver versus network size (number of subdomains) until convergence with normalized residual of 1×10^{-3} with bars indicating spread due to varying permutation matrix (bottom). J or G-S label denotes the use of Jacobi or Gauss-Seidel iterations, respectively.

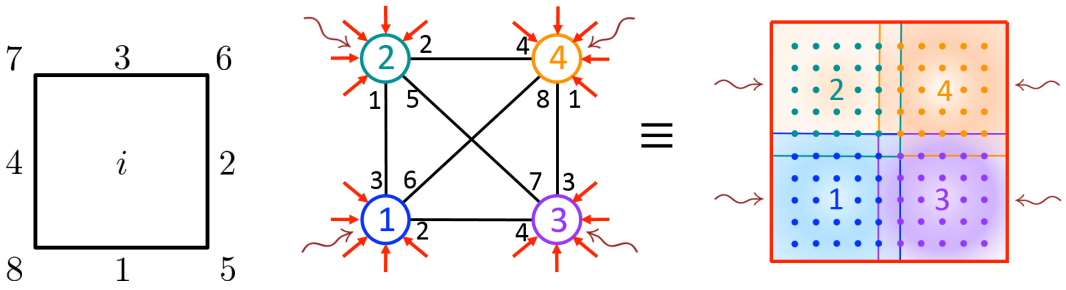


Figure 3.15: 2D nonlinear heat equation on the unit square: Template network component with corresponding port labels (left) and sketch of the DDUQ setting (right). Wavy arrows and red straight arrows represent, respectively, stochastic input parameters and exogenous boundary conditions. Black edges correspond to endogenous inputs (stochastic interface conditions).

Problem setup

To decompose the domain, we split the unit square in smaller $n_x \times n_y$ overlapping rectangles, arranged in a Cartesian grid. Network-wise, each rectangle is a component with 8 input/output ports (4 edges and 4 corners - see Figure 3.15, left), and the network connectivity is dictated by the adjacency pattern of the DD discretization. A sketch of the DDUQ setting is shown in figure 3.15 (right). The underlying deterministic solver is based on FEM.

Due to the nonlinearity in (3.35), a stochastic solution in closed form is not available. Therefore, we will compare the DDUQ solution with a high-fidelity solution, obtained via NISP on the monolithic domain. With the following input uncertainty characterization:

$$\begin{aligned}[T_{1,0}, T_{1,1} \cdots\ T_{1,9}] &= [1.0,\ 0.2,\ 0,\ 0.02,\ 0,\ 0,\ 0.002,\ 0,\ 0,\ 0]\,, \\ [T_{2,0}, T_{2,1} \cdots\ T_{2,9}] &= [1.0,\ 0,\ 0.2,\ 0,\ 0,\ 0,\ 0.02,\ 0,\ 0,\ 0.002]\,,\end{aligned} \tag{3.36}$$

mapping to the following PCEs for $T_1(\boldsymbol{\xi})$ and $T_2(\boldsymbol{\xi})$

$$\begin{aligned}u_\Gamma = T_1(\boldsymbol{\xi}) &= 1.0\Psi_0(\boldsymbol{\xi}) + 0.2\Psi_1(\boldsymbol{\xi}) + 0.02\Psi_3(\boldsymbol{\xi}) + 0.002\Psi_6(\boldsymbol{\xi}), \\ \mu = T_2(\boldsymbol{\xi}) &= 1.0\Psi_0(\boldsymbol{\xi}) + 0.2\Psi_2(\boldsymbol{\xi}) + 0.02\Psi_5(\boldsymbol{\xi}) + 0.002\Psi_9(\boldsymbol{\xi}),\end{aligned} \tag{3.37}$$

where $\{\Psi_k\}_{k=0}^{9}$ are defined as in (3.32), the high-fidelity solution is as in Fig. 3.16. Note that the above characterization results in independent $T_1(\boldsymbol{\xi})$ and $T_2(\boldsymbol{\xi})$, although the overall formulation is general and can in principle include dependent BCs.

Truncation errors

We truncated the input parameter PCE expansions to examine the effect of using a lower fidelity input uncertainty characterization on the system response. With varying truncation levels, we extract the 10 PCE coefficients for the response (characterizing a third order output PCE) and obtain the normalized L_2 and L_∞ norms of the resulting response PCE coefficients with respect to the original case (3rd-order PCEs for both inputs). The normalized errors show similar trends (as well as similar values) whether we utilize the L_2 or L_∞ error norms. For brevity, we will therefore limit the discussion to the L_2 errors that are plotted in Fig. 3.17. When examining the mean response, i.e. Fig. 3.17 subplot (a), we can observe that a truncation of either the diffusivity parameter or boundary condition PCE results in relatively small errors as long as a 1st-order characterization is used for both inputs. This is expected as the mean response of a system normally depends on the lower order moments of the uncertain inputs. It is also observed that 2nd-order PCE truncations result in less error than 1st order truncations, although the gap in the errors is not significant.

Examining higher order output PCE coefficients, there is a general trend that shows strong correlation between input order (i.e. truncation used) and error in the related output PCE coefficients corresponding to polynomials of similar seed (i.e. ξ_1 vs ξ_2). For example, consider Fig. 3.17 subplot (b), which illustrates the error in the output PCE coefficient u_1, corresponding to the Hermite polynomial $\Psi_1(\boldsymbol{\xi}) = \xi_1$. Since ξ_1 is the random seed associated with the boundary condition uncertainty, a truncation along that dimension (i.e. boundary condition PCE order) results in a greater error compared to an equivalent truncation of the input uncertainty associated with ξ_2 (i.e., parameter PCE). In other words, an accurate approximation of the PC coefficient

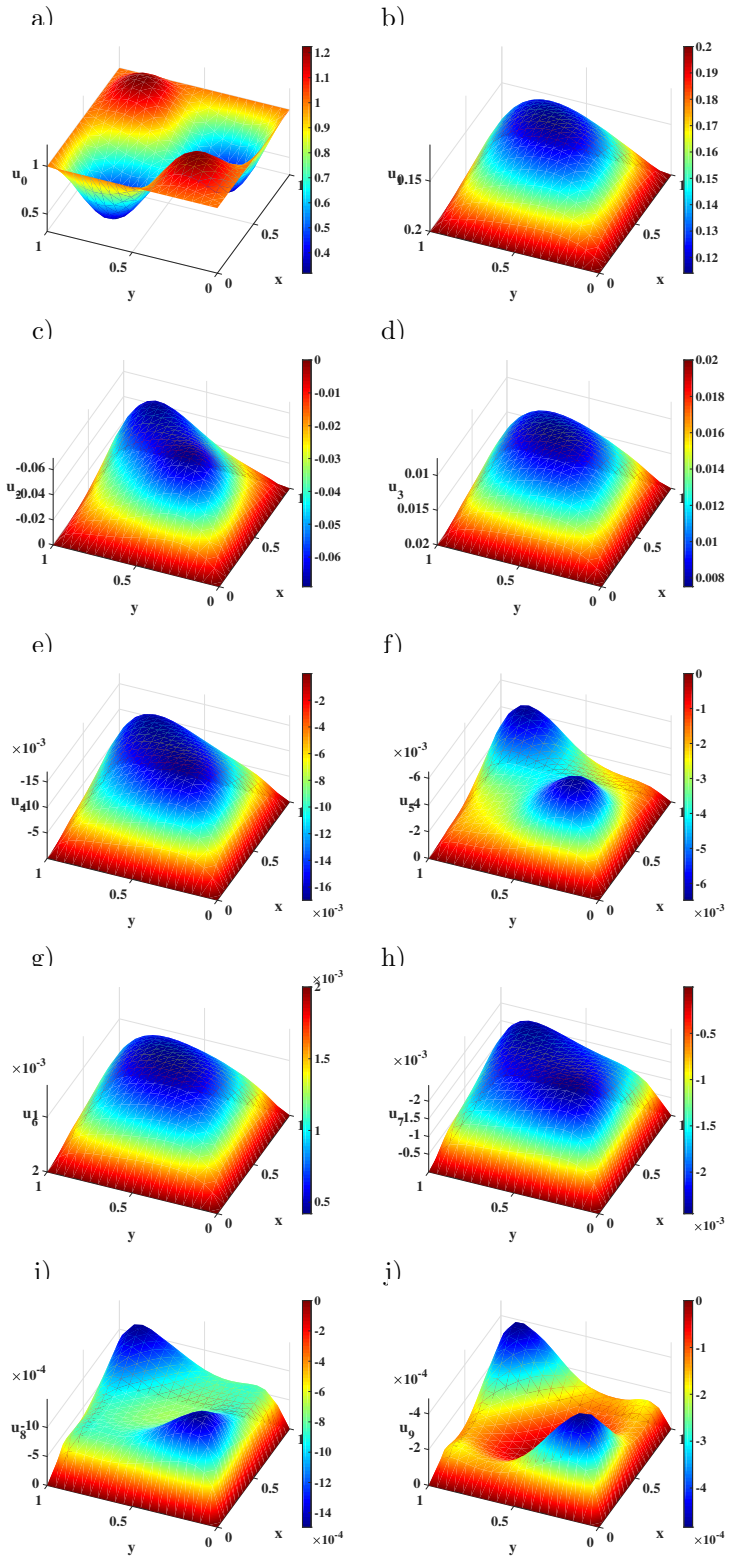


Figure 3.16: Solution PCE coefficients for the 2D nonlinear heat equation using a global NISP.

u_1 requires a *sufficiently* accurate PCE of the related input uncertainty (boundary condition). The same can be said for the truncation of the diffusivity parameter (characterized by ξ_2) and the output PCE coefficient, u_2, corresponding to the Hermite polynomial $\Psi_2 = \xi_2$, as shown in Fig. 3.17 subplot (c). Similar trends are seen for higher order output PCE coefficients. Furthermore, we can see that higher order PCE coefficients incur larger errors due to input PCE truncation than lower order ones. Since the proposed DDUQ methodology involves the propagation of a truncated boundary condition characterization from one node (subdomain) to the next, it is important to note that such truncation results in small errors in the lower order PCE coefficients which correspond to the mean response and capture most of the uncertainty (when characterized by variance) in the output.

Scalability tests

For this example, we use relaxation methods as outlined above to solve the stochastic BVP. Figs. 3.18 and 3.19 provide normalized residual convergence results regarding strong and weak scalability, respectively. For weak scalability, we kept the subdomain problem size (i.e. subdomain-level finite-element nodes) fixed while increasing the number of subdomains. For strong scalability, we kept the global problem size (i.e. number of global finite-element nodes) fixed while increasing the number of subdomains. The results (for strong and weak scalability cases) are identical for the cases involving 64 nodes (subdomains) since they involve the same network with the same discretization (subdomain overlaps), regardless of method and relaxation used. As the subdomain number decreases to 16, and then finally to 4, the amount of overlap is smaller for the strong scalability cases resulting in the observed inferior convergence rates for the same method and relaxation.

Figs. 3.20 and 3.21 provide strong and weak scalability results, with both serial and parallel execution timing provided. For all scalability results, all timings are obtained by performing calculations on an Intel(R) Xeon(R) Core(TM) i7-5557U CPU @ 3.10GHz with 16 GB RAM. The parallel execution time is the time that one

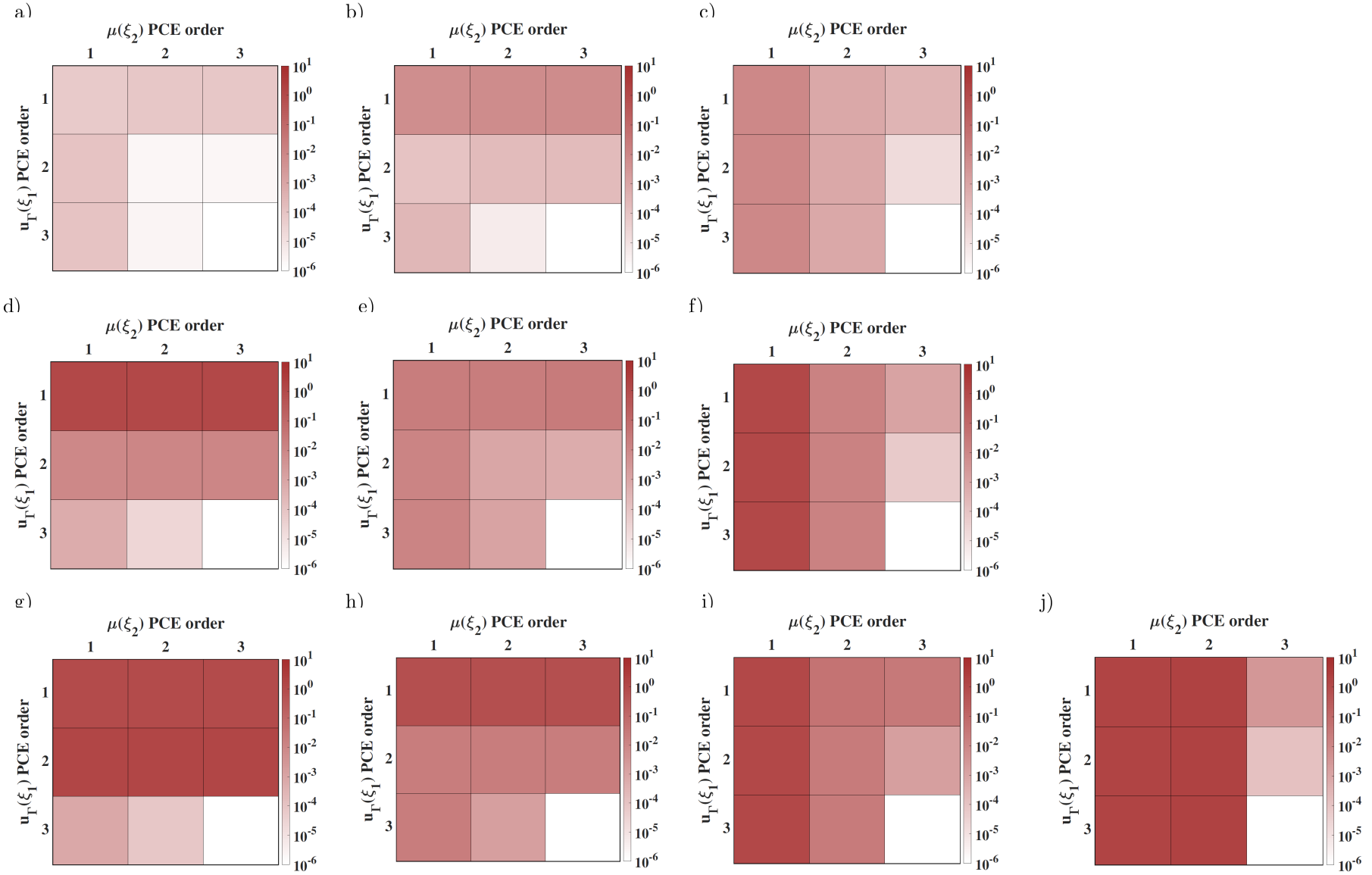


Figure 3.17: 2D nonlinear heat equation: Normalized L_2 error norms for output PCE coefficients $u_0, \ldots, u_9$ (a-j) with varying level of PCE order for the diffusion parameter (column index) and boundary condition (row index).

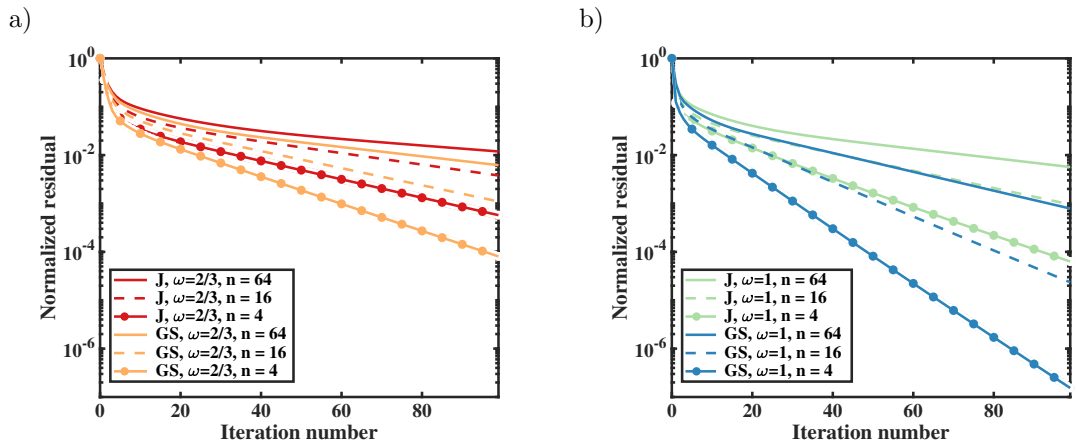


Figure 3.18: Strong scalability convergence results for 2D heat equation for $\omega = 2/3$ (left) and $\omega = 1$ (right): Iteration number versus normalized residual of interface unknowns (PC coefficients). J or G-S labels denote the use of Jacobi or Gauss-Seidel iterations, respectively.

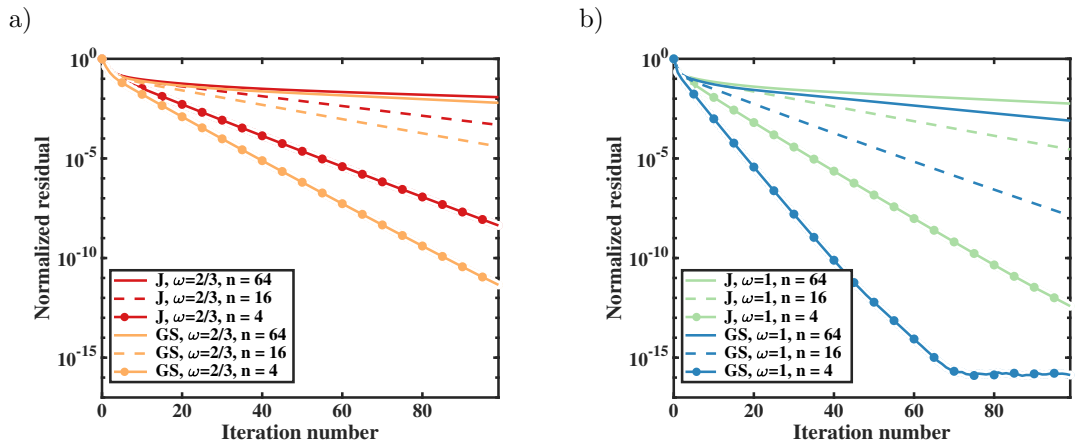


Figure 3.19: Weak scalability convergence results for 2D heat equation for $\omega = 2/3$ (left) and $\omega = 1$ (right): Iteration number versus normalized residual of interface unknowns (PC coefficients). J or G-S labels denote the use of Jacobi or Gauss-Seidel iterations, respectively.

node takes in propagating the uncertainty (since all nodes are similar in this case) for the Jacobi iterations. As for the Gauss-Seidel iterations, the parallel execution time is obtained by choosing the optimal permutation matrix that maximizes the parallelizability of the UQ propagation step through the various nodes within the network at each iteration. Fig. 3.22 provides strong scalability results for Gauss-Seidel iterations with random permutation matrix and $\omega = 1$ to illustrate the effect that the permutation matrix has on performance. Varying the permutation matrix (a total of 10 realizations) seems to have a minimal impact on the total number of iterations until convergence. On the other hand, we can see large scatter in the serial execution time and, to a greater extent, the parallel execution time. Specifically regarding the parallel execution time, the effect is detrimental in terms of scalability as the execution time increases on average going from a 16-node to a 64-node network, in contrast to that obtained using an optimal permutation matrix (see Fig. 3.20 (c)).

For the following discussion, we focus on the errors introduced in propagating uncertainty through the network using the proposed methodology in comparison to propagating uncertainty at the system level (i.e. without domain decomposition), with the later acting as the baseline solution. To isolate those errors, we run the iterations until convergence with normalized residual of 1×10^{-10}. Figs. 3.23 illustrates the effect of network size (with fixed 3-rd order PCE representation of inter-node links) and 3.24 the effect of finite PCE-order representation of links (i.e. artificial boundary values) between the nodes (for a fixed 4-node network). To do this, we provide normalized error results relating to the output PC coefficients (as a vector-valued quantity) at specific physical locations. These errors are normalized with respect to the baseline solution without domain decomposition. We can see that, as the network size increases, the errors grow due to the finite (3rd) order PCE representation of the links between the nodes. We can also see the same trends with decreasing PCE-order representation of the links. Furthermore, for a specific number of nodes (subdomains) or PCE-order of links, we see a slower convergence rate for locations further away from the boundary of the computational domain. This

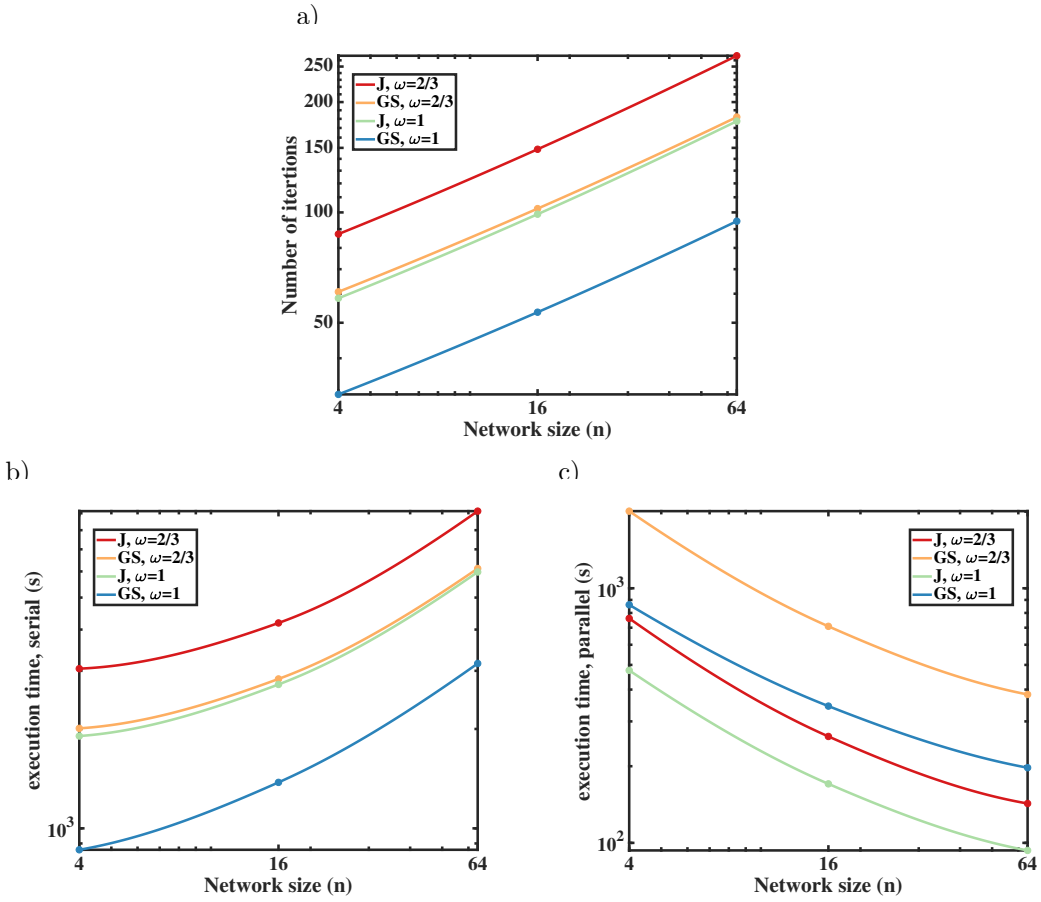


Figure 3.20: Strong scalability results for 2D heat equation: (a) iteration count versus network size (number of subdomains) until convergence with normalized residual of 1×10^{-3}, (b) serial execution time of solver versus network size (number of subdomains) until convergence with normalized residual of 1×10^{-3}, (c) parallel execution time of solver versus network size (number of subdomains) until convergence with normalized residual of 1×10^{-3}. J or G-S label denotes the use of Jacobi or Gauss-Seidel iterations, respectively. Gauss-Seidel iterations executed with optimal permutation matrix resulting in minimal parallel execution times.

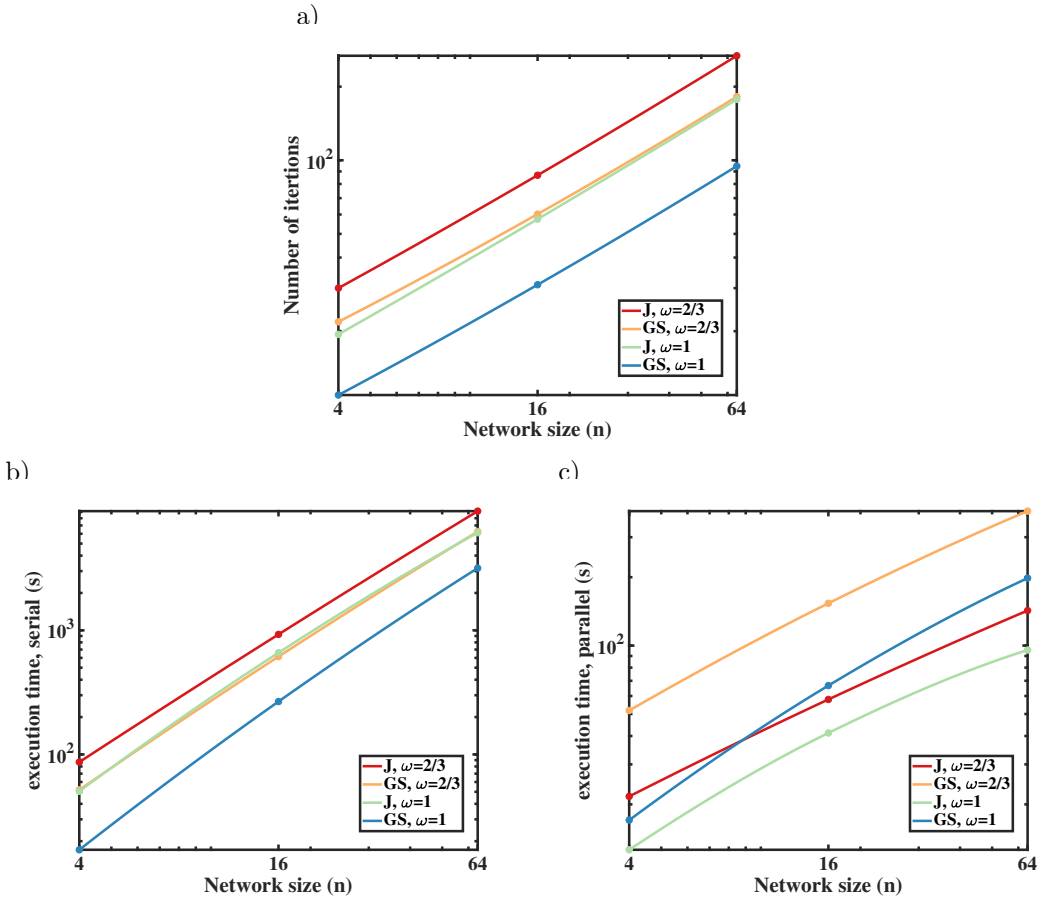


Figure 3.21: Weak scalability results for 2D heat equation: (a) iteration count versus network size (number of subdomains) until convergence with normalized residual of 1×10^{-3}, (b) serial execution time of solver versus network size (number of subdomains) until convergence with normalized residual of 1×10^{-3}, (c) parallel execution time of solver versus network size (number of subdomains) until convergence with normalized residual of 1×10^{-3}. J or G-S label denotes the use of Jacobi or Gauss-Seidel iterations, respectively. Gauss-Seidel iterations executed with optimal permutation matrix resulting in minimal parallel execution times.

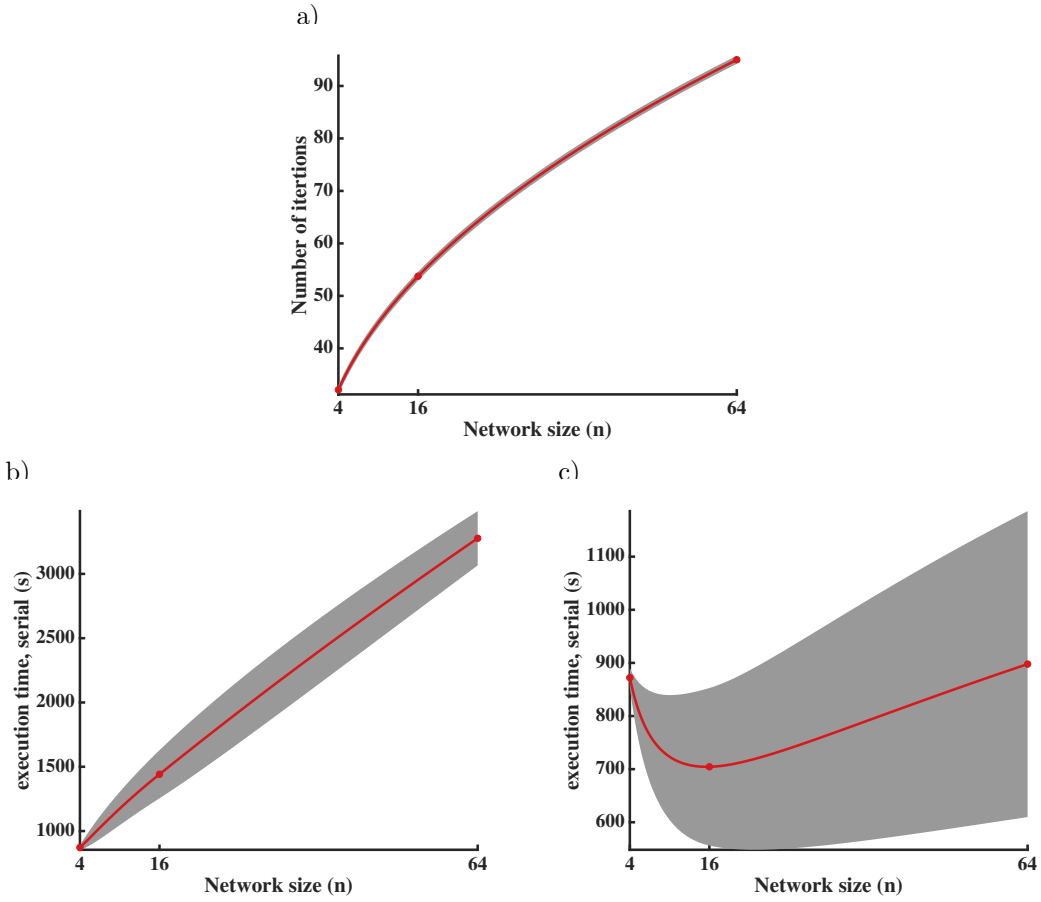


Figure 3.22: Strong scalability results for 2D heat equation with random permutation matrix for Gauss-Seidel iterations with $\omega = 1$: (a) iteration count versus network size (number of subdomains) until convergence with normalized residual of 1×10^{-3}, (b) serial execution time of solver versus network size (number of subdomains) until convergence with normalized residual of 1×10^{-3}, (c) parallel execution time of solver versus network size (number of subdomains) until convergence with normalized residual of 1×10^{-3}. Red curve represents the mean and grey region the confidence interval as plus-or-minus three standard deviations.

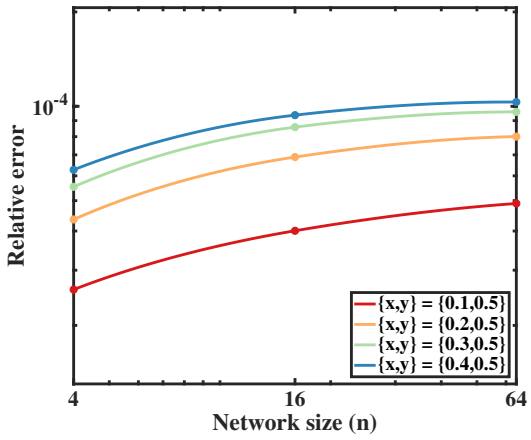


Figure 3.23: Network-related error for 2D heat equation: normalized error for solution (PC coefficients) at isolated physical locations for varying network size with fixed 3-rd order PCE representation of inter-node links.

is expected since those locations are both physically and network distance-wise further away from the sources of uncertainty and thus a larger number of iterations is required to propagate the uncertainty to those locations through the network.

3.4 DDUQ acceleration

The iterative process induced by the fixed point system (3.5) is not guaranteed to converge and, if it does, convergence cannot be more than linear [40]. Several acceleration methods can be found in the literature to overcome these issues. For instance, vector extrapolation methods [39] transform a sequence of vectors generated by some iterative process to a new set of vectors that converge faster than the initial sequence. Depending on the method used to compute such transformation, we can distinguish between polynomial methods (Reduced-Rank Extrapolation, Minimal-Polynomial Extrapolation, etc. [111]) or ε-methods (scalar or vector ε-algorithms [112]). A different approach is adopted by the family of Anderson Acceleration (AA)

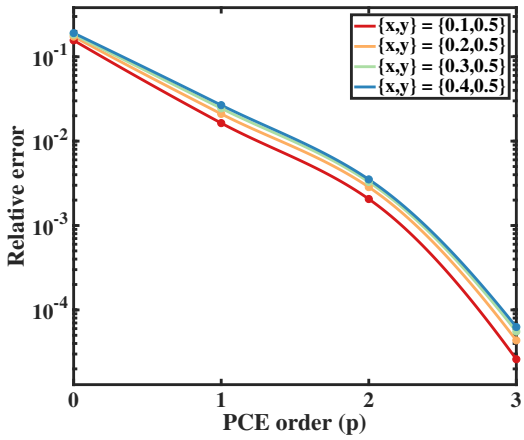


Figure 3.24: Network-related error for 2D heat equation: normalized error for solution (PC coefficients) at isolated physical locations for varying PCE-order representation of links between the nodes for a 4-node network.

methods [12]. Here the iterates are not modified, but stored and used to compute the new iterate via linear combination. It has been proved that this is analogous to the so-called Pulay mixing for electronic applications and to non-linear GMRES [102, 196]. Moreover, the authors in [71] show how AA is related to the multi-secant quasi-Newton method and, for the linear case, to GMRES.

Anderson Acceleration has been successfully applied to the DDUQ problem in [113]. However, this method is merely algebraic and unrelated to the specificity of the problem. Therefore, our goal is to improve the performance of the solver accelerated via AA, possibly exploiting more information about the (non-linear) DDUQ problem. On the one hand, since the network formulation arises from the DD approach, the mathematical and geometrical setting suggests the creation of a hierarchy of sub-problems by combining neighboring components to obtain a coarser network. On the other hand, the PC modal expansion employed to represent uncertainties in the UQ problem leads to the idea of coarsening by truncating the PCE at lower polynomial degrees. With either perspective, the multilevel structure of the problem is evident. However, while multigrid (MG) methods strongly rely on the concept

of mesh coarsening, or on the adjacency graph of the system matrix, the network problem is mesh-free, and the DDUQ problem may not admit a matrix formulation, in general. Moreover, the Spectral Theory does not apply to the PCE right away because the modal coefficients are not just algebraic quantities, since they involve multiple PDE solves and a projection on the stochastic space. Therefore, a direct application of the already well-established MG methods is not possible. In order to understand how the basic principles of MG may be applied to our problem, we start by recalling some basics concepts (see [42] for details).

3.4.1 The idea

Smoothing processes are characterized by a fast decay of the error in the first few iterations, followed by a drastic decrease of the convergence rate. This behavior is explained by the smoothing action of the iterative solver, that quickly filters out the high-frequency components of the error within the first few iterations, but is less efficient in damping the slow persistent modes, resulting in the necessity of a larger computational effort to achieve the same error reduction. Multigrid methods are designed to accelerate the reduction of the slow components of the error. The key idea is that slow frequencies on a fine grid are fast frequencies on a coarse grid. Hence, an approximate solution on the fine grid obtained after few smoothing steps can be corrected with an accurate approximation of the error on the coarse grid, which accelerates damping the slow components.

3.4.2 Geometric Multigrid

Consider a linear system in the form

$$\mathbf{A}\boldsymbol{u} = \mathbf{b}, \tag{3.38}$$

with the corresponding residual equation $A\mathbf{e} = \mathbf{r}$. Referring to quantities on the fine and coarse grid with the subscripts h and H, respectively, what Geometric Multigrid

(GMG) does is (see Algorithm 4): calculating an approximate solution on the fine grid with few iterations and the corresponding residual (line 1); transferring these two quantities to the coarse grid via a restriction operator R (line 2); solving the residual equation on the coarse grid to obtain the error (line 3), that is then transferred back to the fine grid via a prolongation operator P (line 4), to correct the solution (line 5). Finally, the corrected solution is used as initial guess for some post-smoothing (line 6). This procedure can be applied recursively, generating V-, μ-, or hybrid cycles.

Algorithm 4: Geometric Multigrid method (2 levels)

1 Pre-smoothing: Calculate an approximate solution $\mathbf{v}_h$ to $\mathbf{A}_h\mathbf{u}_h = \mathbf{f}_h$ and the residual $\mathbf{r}_h$;
2 Restrict $\mathbf{v}_h$ and $\mathbf{r}_h$ to the coarse grid: $\mathbf{v}_H = R\mathbf{v}_h$, $\mathbf{r}_H = R\mathbf{r}_h$;
3 Solve the coarse-grid residual equation $\mathbf{A}_H\mathbf{e}_H = \mathbf{r}_H$;
4 Interpolate the error on the fine grid: $\mathbf{e}_h = P\mathbf{e}_H$;
5 Correct the solution on the fine grid: $\mathbf{v}_h \leftarrow \mathbf{v}_h + \mathbf{e}_h$
6 Post-smoothing: Calculate an approximate solution to $\mathbf{A}_h\mathbf{u}_h = \mathbf{f}_h$ with initial guess $\mathbf{v}_h$.

3.4.3 Full Approximation Scheme (FAS) for non-linear problems

While for a linear problem like (3.38) the error can be obtained directly from the residual equation, for non-linear problems more work is necessary, since the residual itself is non-linear. More precisely, the residual equation yields the approximate solution perturbed by the error, which is a *full approximation* of the solution, rather then the error itself. Then, the error can be obtained as the difference between the perturbed and the non-perturbed approximate solution. The multigrid method for a non-linear problem is called "Full Approximation Scheme" (FAS), after this feature.

Consider a system of nonlinear equations

$$\mathbf{A}(\boldsymbol{u}) = \mathbf{b}, \tag{3.39}$$

where $\boldsymbol{u}$, $\mathbf{b} \in \mathbb{R}^n$. Let $\boldsymbol{v}$ be an approximation to the exact solution $\boldsymbol{u}$, and $\mathbf{e} = \boldsymbol{u} - \boldsymbol{v}$, $\mathbf{r} = \mathbf{b} - \mathbf{A}(\boldsymbol{v})$ the corresponding error and residual, respectively. By substituting the original equation (3.39) in the definition of the residual, we obtain

$$\mathbf{A}(\boldsymbol{u}) - \mathbf{A}(\boldsymbol{v}) = \mathbf{r}. \tag{3.40}$$

Note that, since the operator $\mathbf{A}$ is non-linear, it is not possible to express (3.40) in terms of the error $\mathbf{e}$. In order to employ the non-linear residual equation (3.40) in a non-linear multigrid method, we express the exact solution $\boldsymbol{u}$ as the current approximation corrected by the error, i.e., $\boldsymbol{u} = \boldsymbol{v} + \mathbf{e}$, so that equation (3.40) reads as

$$\mathbf{A}(\boldsymbol{v} + \mathbf{e}) - \mathbf{A}(\boldsymbol{v}) = \mathbf{r}. \tag{3.41}$$

Hence, one step of the two-level FAS algorithm can be formulated as follows:

Algorithm 5: Full Approximation Scheme (2 levels)

1. Pre-smoothing: Calculate an approximate solution $\mathbf{v}_h$ of $\mathbf{A}_h(\mathbf{u}_h) = \mathbf{f}_h$ on the fine grid and the corresponding residual $\mathbf{r}_h$;
2. Restrict the approximate solution and the residual on the coarse grid: $\mathbf{v}_H = R\mathbf{v}_h$, $\mathbf{r}_H = R\mathbf{r}_h$;
3. Solve the coarse-grid problem $\mathbf{A}_H(\mathbf{v}_H + \mathbf{e}_H) = \mathbf{A}_H(\mathbf{z}_H) = \mathbf{A}_H(\mathbf{v}_H) + \mathbf{r}_H$;
4. Obtain the error on the coarse grid as $\mathbf{e}_H = \mathbf{z}_H - \mathbf{v}_H$;
5. Prolongate the error back to the fine grid $\mathbf{e}_h = P\mathbf{e}_H$;
6. Correct the solution on the fine grid $\mathbf{v}_h \leftarrow \mathbf{v}_h + \mathbf{e}_h$;
7. Post-smoothing: Calculate an approximate solution of $\mathbf{A}_h(\mathbf{u}_h) = \mathbf{f}_h$ on the fine grid with initial guess $\mathbf{v}_h$.

where P and R are the prolongation and restriction operators, respectively, and the subscripts h, H refer to the fine and coarse grid, respectively.

3.4.4 Algebraic Multigrid

Algebraic Multigrid (AMG) is a grid-free version of the multigrid method, that can be applied when the physical locations of the unknowns are themselves unknown or immaterial. For AMG the coarse-grid unknowns are a subset of the original variables. The grid points are are the indices of the unknowns and the connections within the grid are determined by the undirected adjacency graph of the matrix. For the sake of simplicity, assume that $A = \{a_{ij}\}$ is a symmetric M-matrix, i.e., it is symmetric positive-definite with positive diagonal entries and non-positive off-diagonal entries. The selection of the coarse grid is based on the following definition:

Definition. Given a threshold value $0 < \vartheta \leq 1$, the variable u_i *strongly depends* on the variable u_j if

$$-a_{ij} \geq \vartheta \max_{k \neq i} \{-a_{ik}\}, \tag{3.42}$$

i.e., if the coefficient a_{ij} is comparable in magnitude to the largest off-diagonal coefficient in the i-th equation. Conversely, if u_i strongly depends on u_j, then u_j *strongly influences* u_i. Furthermore, we define the following sets:

- N_i, the set of all points $j \neq i$ such that $a_{ij} \neq 0$;
- C_i, the set of neighboring coarse-grid points that strongly influence i;
- D_i^s, the set of neighboring fine-grid points that strongly influence i;
- D_i^w, the set of neighboring fine- or coarse-grid points that do not strongly influence i.

It can be shown that smooth error varies slowly in the direction of strong connection. Hence, the prolongation (interpolation) operator $P = I_H^h$ can be defined as

$$\left(I_H^h \mathbf{e}\right)_i = \begin{cases} e_i & \text{if } i \in C \\ \sum_{j \in C_i} \omega_{ij} e_j & \text{if } i \in F, \end{cases} \tag{3.43}$$

where C and F are the sets of the coarse- and fine-grid variables, respectively, with $C \cap F = \emptyset$, and ω_{ij} are the interpolation weights defined by

$$\omega_{ij} = -\frac{a_{ij} + \sum_{m \in D_i^s} \left(\frac{a_{im} a_{mj}}{\sum_{k \in C_i} a_{mk}} \right)}{a_{ii} + \sum_{n \in D_i^w} a_{in}}. \tag{3.44}$$

Although this is the "traditional" formulation of AMG, alternative definitions for the coarsening (compatible relaxation-based, path length-based, ...) and interpolation (long range, adaptive, red-black, ...) can be found in the literature.

3.5 Multigrid Methods for DDUQ Network Problems in Matrix Form

For any Multigrid algorithm, the following "ingredients" are required: (i) A sequence of grids; (ii) Prolongation and restriction operators between grids; (iii) A relaxation operator; (iv) A coarse-grid version of the relaxation operator; (v) A solver for the coarse grid. Depending on the definition of the grid, different classes of MG methods can be derived. As anticipated above, in what follows we design two new MG methods by defining the grid as the (partially ordered) set of the coefficients of the PCE that describes the uncertainties (pMG), or as the network (hMG). We derive a formulation of the method for UQ problems in networks that can be expressed in matrix form, and we discuss how this approach can be generalized to any network problem.

3.5.1 p-Multigrid

In a generalized spectral expansion, the Spectral Theorem guarantees that, under suitable assumptions, the most significative information is captured by the low-order coefficients (associated with the smallest eigenvalues). Hence, solving the residual

equation for the lower frequencies is expected to be beneficial for the convergence of the full solution. The extension of such reasoning to a PCE is not straightforward. In this case, the coefficients are obtained by integrating a spatial solution over the stochastic space, and the corresponding approximation is given by a linear combination of solutions in space, associated with different samples of the parameter space. The behavior of the solution as a response to the choice of the parameters is not easily predictable, especially for non-linear problems. Therefore, dropping high-order coefficients may lead to loss of information and, consequently, to low-quality corrections.

In order to better understand the effects of a pMG scheme on a problem of uncertainty quantification, as a first step we assess the numerical behavior of the method for linear problems that admit a matrix formulation.

Prolongation and Restriction Operators. In a UQ network problem, each edge is the vehicle of the propagation of the uncertainty, in our case represented via the coefficients of a PCE of order p. Let such coefficients be stored in a vector $\mathbf{p}$ of length N, which we define as the size of the edge. The outcome of a restriction operation on an edge $\mathbf{p}^F$ of size $N^F$is a new edge $\mathbf{p}^C$ of size $N^C < N^F$. The restriction operation R can be defined in different ways. We explore the following alternatives, summarized in Table 3.1:

- *Truncate coefficient*: at each pMG level, drop the last coefficient.
- *Truncate order*: at level $\ell > 1$, drop all the coefficients associated with the order $p_\ell = p - \ell + 2$.
- *Average forward*: at each pMG level, average consecutive coefficients pairwise.
- *Average last*: at each pMG level, average the last two PCE coefficients.
- *Average by order*: at level $\ell > 1$, average all the PCE coefficients corresponding to orders $p - \ell + 2 : p$.

- *Average all orders*: for each order, average the PCE coefficients corresponding to the same order (2-level pMG only).
- *Average all*: average all the PCE coefficients (2-level pMG only).
- *Adaptive*: use the PCE coefficients at the fine level as a projector (2-level pMG only).

Type	Description	N^C	Operator
Truncate coeff.	Drop last coeff.	$N^F - 1$	$\mathbf{p}^F_{N^F} = 0$
Truncate order	Drop highest order	N_ℓ	$\mathbf{p}^F_i = 0,\ i = N^C + 1, \ldots, N^F$
Avg forward	Avg coeff. pairwise	$N^F - 1$	$\mathbf{p}^C_i = \text{avg}(\mathbf{p}^F_i, \mathbf{p}^F_{i+1})$
Avg last	Avg last two coeff.	$N^F - 1$	$\mathbf{p}^C_{N^C} = \text{avg}(\mathbf{p}^F_{N^F-1}, \mathbf{p}^F_{N^F})$
Avg by order	Avg highest order	$N_\ell + \ell - 1$	$\mathbf{p}^C_{N_\ell+\ell-1} = \text{avg}(\mathbf{p}^F_{N_\ell+1}, \ldots, \mathbf{p}^F_{N_{\ell+1}})$
Avg all orders	Avg each order	$p + 1$	$\mathbf{p}^C_\ell = \mathbf{p}^C_{N_\ell+\ell-1}\ \forall \ell = 1, \ldots, p+1$
Avg all	Avg all coeff.	1	$\mathbf{p}^C_{N^C} = \text{avg}(\mathbf{p}^F_1, \mathbf{p}^F_{N^F})$
Adaptive	Project on fine sol.	1	$\mathbf{v}^F \cdot \mathbf{p}^F$

Table 3.1: pMG coarsening types.

Then, the prolongation operator P is obtained from R by transposition ($P = R^T$).

Remark. A coarsening of type pMG does not affect the topology or the connectivity of the network. Rather, it acts on the edge size. In other words, pMG is a conservative (non-intrusive) method, since the network solver can be used as a black-box. This feature makes it extremely flexible in terms of applicability. Moreover, from a user perspective it requires no effort in terms of setup.

3.5.2 h-Multigrid

In an hMG method, the grid is defined as the network. This means that the nodes of the coarse network are "hyper-nodes", obtained by the aggregation of nodes of the fine network, according to *some* rule. As a result, the edges of the network are coarsened as well.

Definition (hMG coarsening type). Level l (≥ 2) of h-MG is of type $(m_l, n_l)_l$ if components at level $l-1$ are combined in intersecting (overlapping) sets of cardinality m_l, with intersection (overlap) of cardinality n_l.

In order to guarantee that the coarse network is well-defined, the following conditions must be satisfied:

1. $m_l \geq 2n_l$: the connectivity of the graph is uniquely defined (for each input port there exist one and only one output port connected to it);
2. $N^C_{comp} = \dfrac{N^F_{comp} - n_l}{m_l - n_l}$: all the components of the coarse network are the same size.

Condition (2) is enforced to facilitate the implementation of the coarsening but can be removed.

The definition of coarsening can be applied recursively, to obtain a multiple-level multigrid method. Figure 3.25 shows an example of $(2,1) - (3,1)$-hMG for a 1D network of 8 components. Note that, from a geometrical viewpoint, such 3-level hMG coarsening is equivalent to a 2-level $(4,2)$ hMG coarsening. More in general, a 3-level hMG of type $(m_2, n_2)_2 - (m_3, n_3)_3$ is geometrically equivalent to a 2-level

hMG of type $(m_2^{(3)}, n_2^{(3)})_2$, with

$$\begin{aligned} m_2^{(3)} &= \left\lceil \frac{m_3}{2} \right\rceil m_2 + \left(\left\lfloor \frac{m_3}{2} \right\rfloor - \left(1 - \left(\left\lceil \frac{m_3}{2} \right\rceil - \left\lfloor \frac{m_3}{2} \right\rfloor \right) \right) \right) (m_2 - 2n_2) + \\ &\quad \left(1 - \left(\left\lceil \frac{m_3}{2} \right\rceil - \left\lfloor \frac{m_3}{2} \right\rfloor \right) \right) (m_2 - n_2), \\ n_2^{(3)} &= \left\lceil \frac{n_3}{2} \right\rceil m_2 + \left(\left\lfloor \frac{n_3}{2} \right\rfloor - \left(1 - \left(\left\lceil \frac{n_3}{2} \right\rceil - \left\lfloor \frac{n_3}{2} \right\rfloor \right) \right) \right) (m_2 - 2n_2) + \\ &\quad \left(1 - \left(\left\lceil \frac{n_3}{2} \right\rceil - \left\lfloor \frac{n_3}{2} \right\rfloor \right) \right) (m_2 - n_2). \end{aligned} \tag{3.45}$$

This property can be applied recursively to obtain $(m_2^{(l)}, n_2^{(l)})_2$ at level l:

Property. An hMG level l of type $(m_l, n_l)_l$ is geometrically equivalent to a 2-level hMG coarsening of type $(m_2^{(l)}, n_2^{(l)})_2$, with

$$\begin{aligned} m_2^{(l)} &= \left\lceil \frac{m_l}{2} \right\rceil m_2^{(l-1)} + \left(\left\lfloor \frac{m_l}{2} \right\rfloor - \left(1 - \left(\left\lceil \frac{m_l}{2} \right\rceil - \left\lfloor \frac{m_l}{2} \right\rfloor \right) \right) \right) \left(m_2^{(l-1)} - 2n_2^{(l-1)} \right) + \\ &\quad \left(1 - \left(\left\lceil \frac{m_l}{2} \right\rceil - \left\lfloor \frac{m_l}{2} \right\rfloor \right) \right) \left(m_2^{(l-1)} - n_2^{(l-1)} \right), \\ n_2^{(l)} &= \left\lceil \frac{n_l}{2} \right\rceil m_2^{(l-1)} + \left(\left\lfloor \frac{n_l}{2} \right\rfloor - \left(1 - \left(\left\lceil \frac{n_l}{2} \right\rceil - \left\lfloor \frac{n_l}{2} \right\rfloor \right) \right) \right) \left(m_2^{(l-1)} - 2n_2^{(l-1)} \right) + \\ &\quad \left(1 - \left(\left\lceil \frac{n_l}{2} \right\rceil - \left\lfloor \frac{n_l}{2} \right\rfloor \right) \right) \left(m_2^{(l-1)} - n_2^{(l-1)} \right). \end{aligned} \tag{3.46}$$

For network problems arising from cartesian domains in higher dimensions, the hMG type is defined as the cartesian product of the 1D hMG coarsening in each dimension. For instance, a $(2,1) \times (3,1)$ coarsening of a 2D cartesian domain is obtained with a $(2,1)$ coarsening along the x-axis and a $(3,1)$ coarsening along the y direction.

Prolongation and Restriction Operators Preliminary studies have been performed on AMG applied to the matrix formulation of a linear 1D heat problem to understand how algebraic coarsening can be interpreted from a geometrical viewpoint. For network UQ problems that admit an algebraic formulation, prolongation and restriction operators can be designed in a similar fashion. Recalling that the unknowns at any coarse level are a subset of the unknowns at the finer level(s), a prolongation operator P can be constructed as follows:

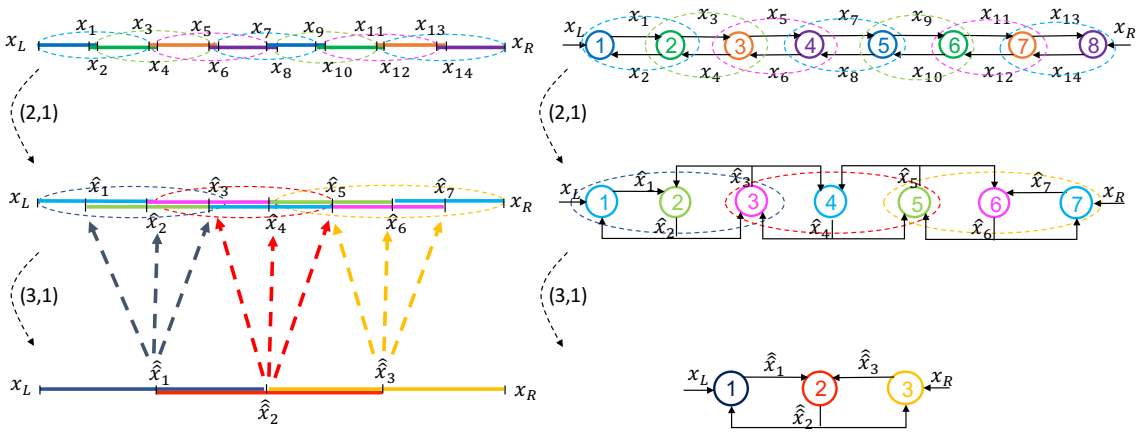


Figure 3.25: Sketch of a $(2,1) - (3,1)$ hMG type: Domain Decomposition (left) and network (right) representation. The action of the prolongation operator is represented by dashed arrows.

- If the unknown u_i on the fine grid is also an unknown on the coarse grid, the value is injected from the coarse to the fine grid;
- If the unknown u_i on the fine grid is not an unknown on the coarse grid, and the corresponding location in space lies outside the range of output locations on the coarse grid, then the value is injected from the closest location on the coarse grid;
- Otherwise, the value is averaged from the two closest locations on the coarse grid.

Then, the restriction operator is defined as the transposed prolongation operator ($R = P^T$). A sketch of the prolongation operator is shown in Figure 3.25.

3.5.3 Preliminary results

As a preliminary investigation, we consider the linear 1D heat equation. In this case, in fact, the UQ problem can be formulated as a system of linear 1D heat problems,

one for each PCE coefficient, and can be expressed in matrix form [45]. As a result, the prolongation and restriction operators can be assembled explicitly and the pMG algorithm has the same flavor of an AMG method[1].

In order to measure the performance of pMG or hMG, we define the serial cost of one MG iteration as

$$c_{MG} = 2N_F \sum_{l=1}^{L-1} \frac{c_l}{c_{max}} + N_C \frac{c_L}{c_{max}}, \tag{3.47}$$

where

- N_F, N_C is the number of iterations on the fine and coarse grid, respectively,
- L is the number of MG levels,
- c_l is the cost of one iteration at level l,
- $c_{max} = \max_{l=1,\dots,L} c_l = c_1$.

The performance of all the methods presented here is expressed in terms of the serial cost defined in (3.47). The cost of a parallel MG solver depends on the choice of the smoother. If the smoother is a Jacobi iterative solver, at each level the work of one propagation is distributed across N^l_{comp} processors, where N^l_{comp} is the number of components at level l, and executed simultaneously in one step. Conversely, if the smoother is of type Gauss-Seidel, at each level the work of one propagation is distributed across N^l_{comp} processors and executed in $N_s(\pi_l)$ sequential steps, depending on the level of parallelism of the pemutation π_l of the network components. In either case, the parallel cost can be expressed as

$$c^{par}_{MG} = 2N_F \sum_{l=1}^{L-1} \frac{c_l}{c_{max}} \frac{N^l_{seq}}{N^l_{comp}} + N_C \frac{c_L}{c_{max}} \frac{N^L_{seq}}{N^L_{comp}}, \tag{3.48}$$

where $N^l_{seq} = 1$ or $N_s(\pi_l)$ for a Jacobi or Gauss-Seidel smoother, respectively.

[1]The code has been developed from the MATAMG toolbox available at [136].

pMG

The setting of the tests is summarized in Table 3.2. The performance of the pMG method is measured in terms of relative residual as a function of number of iterations, time (averaged over 10), and cost in terms of matrix-vector products and deterministic solves[2]. In (3.47), the cost of a lower-triangular matrix-vector product is calculated as

$$c_l = 2\left(\frac{\left(N_p^l N_E\right)^2 - N_p^l N_E}{2} + N_p^l N_E\right), \tag{3.49}$$

where N_p^l is the number of PCE coefficients at level l, and N_E is the number of edges of the network (constant at each level), while the number of deterministic solves is

$$c_l = (p_l + 1)^d, \tag{3.50}$$

where p_l is the order of the PCE at level l, and d is the dimension of the stochastic space.

The results for each pMG level are shown in Figures 3.26, 3.27, 3.28, 3.29. The coarsening types "adaptive" and "average all orders" are the cheapest in terms of costs (in fact they support only two levels), whereas "average last" is the fastest in terms of time (for $L > 2$). However, in general none of the pMG coarsening types brings a significant improvement compared to the standard relaxation process and AMG features the best performance in all the cases.

hMG

The hMG method is tested on the linear 1D heat problem in matrix form with parameter setting as in Table 3.3. The hMG coarsening types tested are listed in Table 3.4. Note that not any combination of coarsening is feasible, due to the constraint (2) stated in paragraph 3.5.2.

[2]The cost of pMG with coarsening type "average by order" is currently overestimated. It is possible to work out a more precise formula for it, but it has not been done yet, since the results are not promising.

Parameter	Description	Value
L_x	Length of the physical domain	1
N_{FE}	Number of FE	10
N_{comp}	Number of network components	32
p	PCE order	5
N_p	Number of PCE coefficients	21
T_1, T_2	Stochastic boundary conditions	Random (loaded)
csnType	Coarsening type	All of above
L	Number of pMG levels	$1, \ldots, p$
intpType	Interpolation type	All of above
smoother	Smoother for pMG	Gauss-Seidel
ω	Relaxation parameter	1
perms	Permutations	$1 : N_{comp}$
$N_{Vcycles}$	Maximum number of V-cycles	$10,000$
tol	Tolerance for stopping criterion	10^{-3}
$N_F = N_C$	Interations on fine/coarse grid	$1, \ldots, 3$

Table 3.2: pMG for linear 1D heat DDUQ network problem: parameter setting.

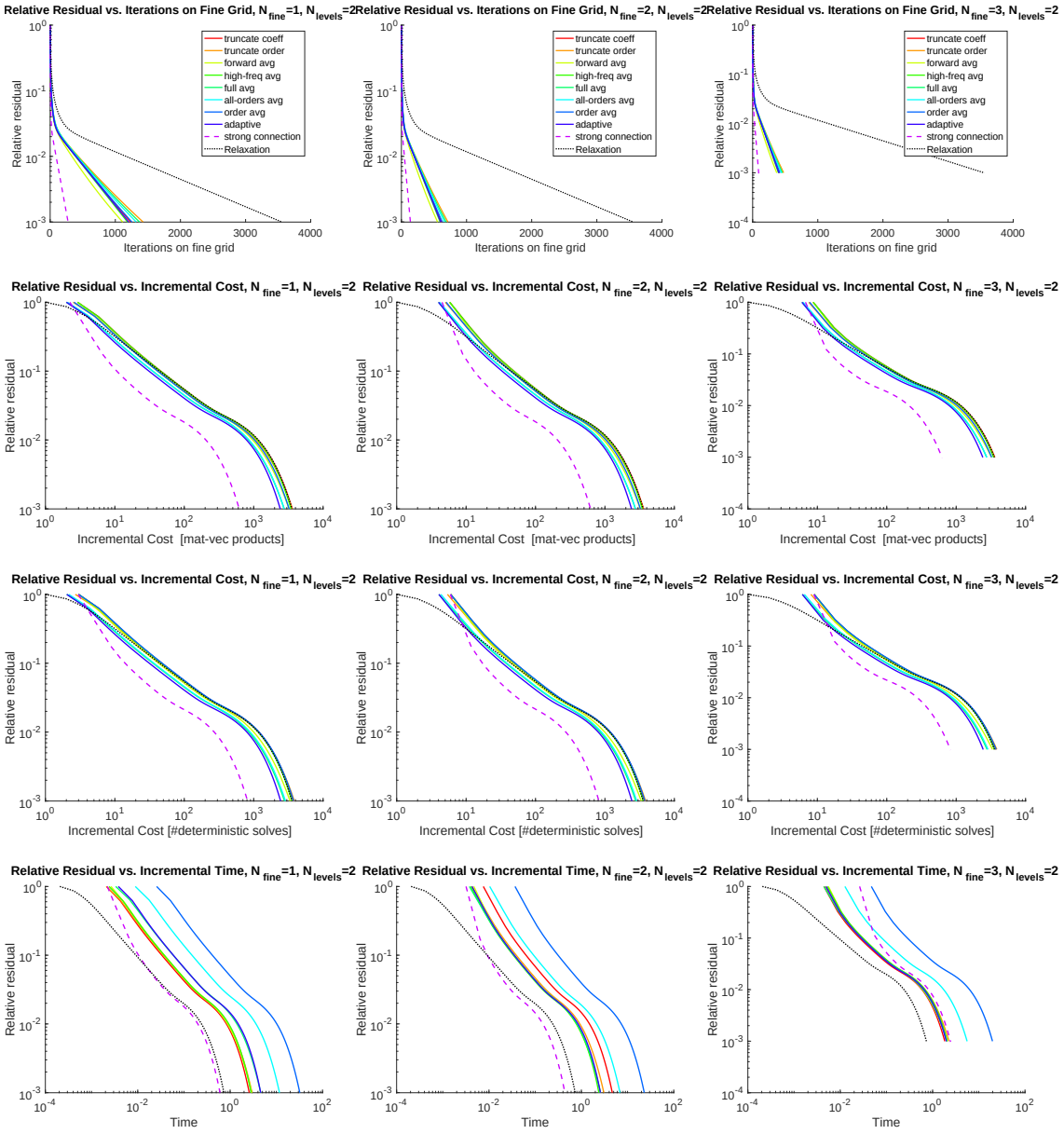


Figure 3.26: 1D heat test: 2-level pMG relative residual vs. number of iterations (top), incremental cost in terms of matrix-vector products (upper center) and number of deterministic solves (lower center), and time (bottom) for $N_F = N_C = \{1, 2, 3\}$ (left-right).

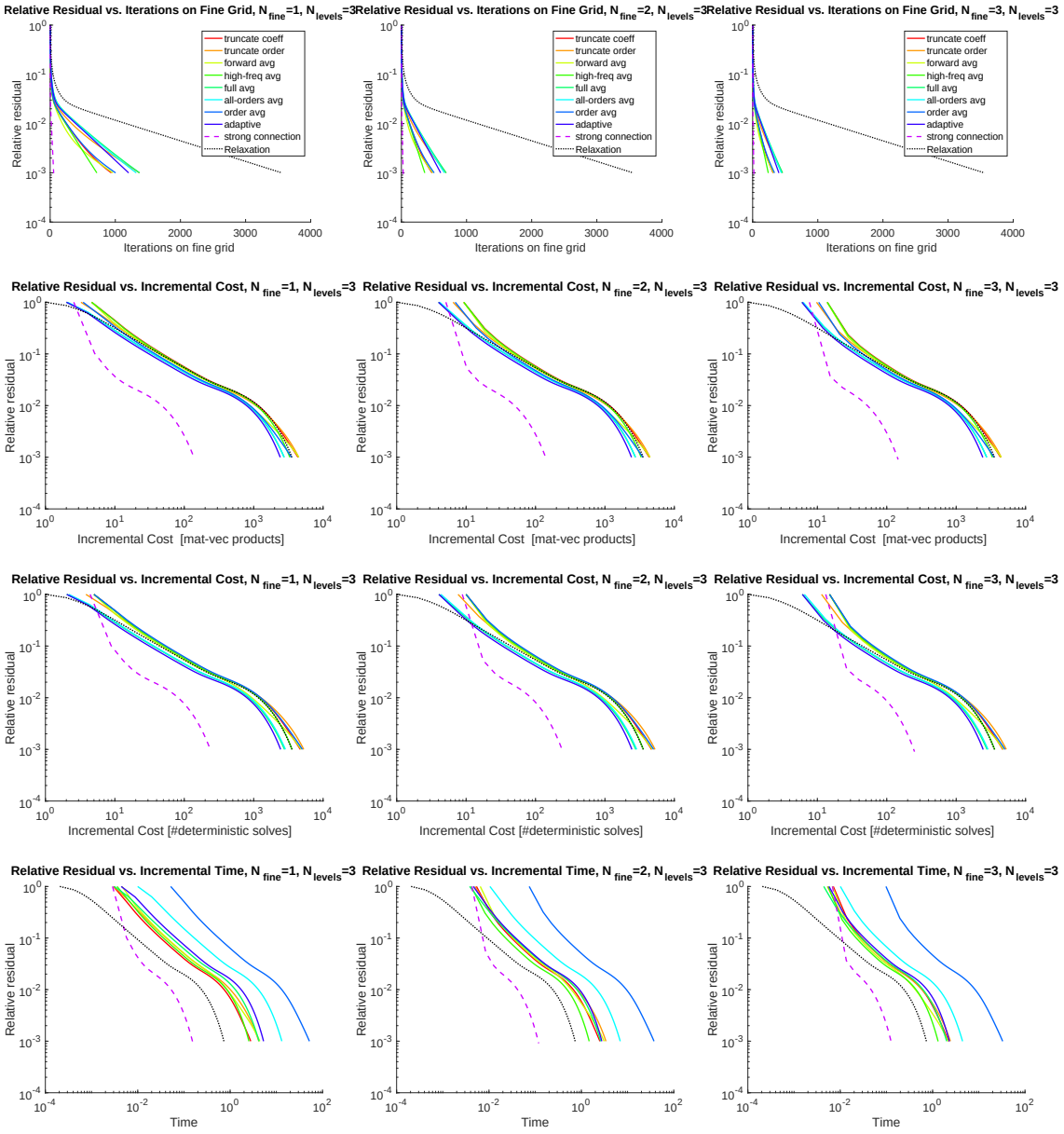


Figure 3.27: 1D heat test: 3-level pMG relative residual vs. number of iterations (top), incremental cost in terms of matrix-vector products (upper center) and number of deterministic solves (lower center), and time (bottom) for $N_F = N_C = \{1, 2, 3\}$ (left-right).

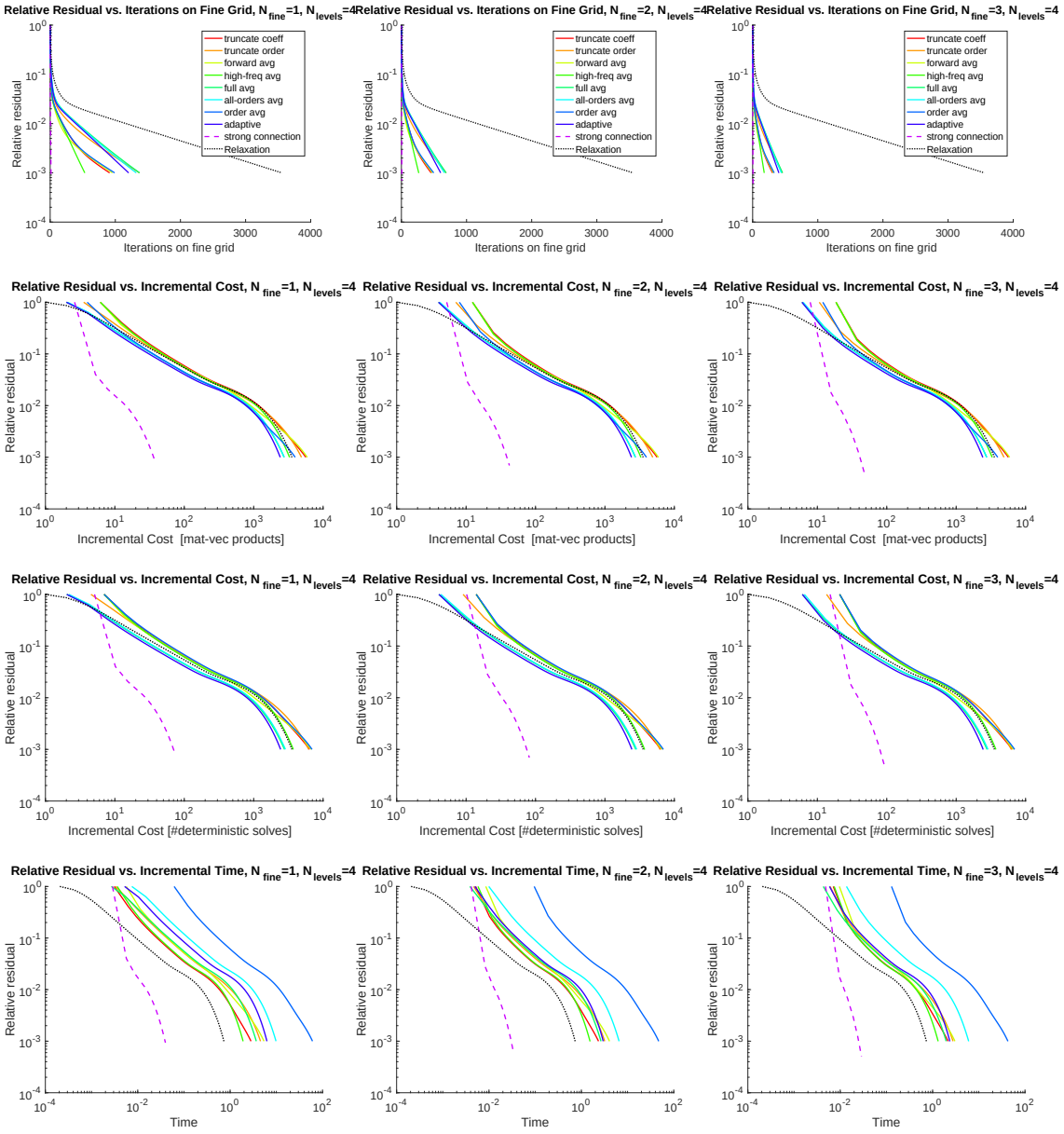


Figure 3.28: 1D heat test: 4-level pMG relative residual vs. number of iterations (top), incremental cost in terms of matrix-vector products (upper center) and number of deterministic solves (lower center), and time (bottom) for $N_F = N_C = \{1, 2, 3\}$ (left-right).

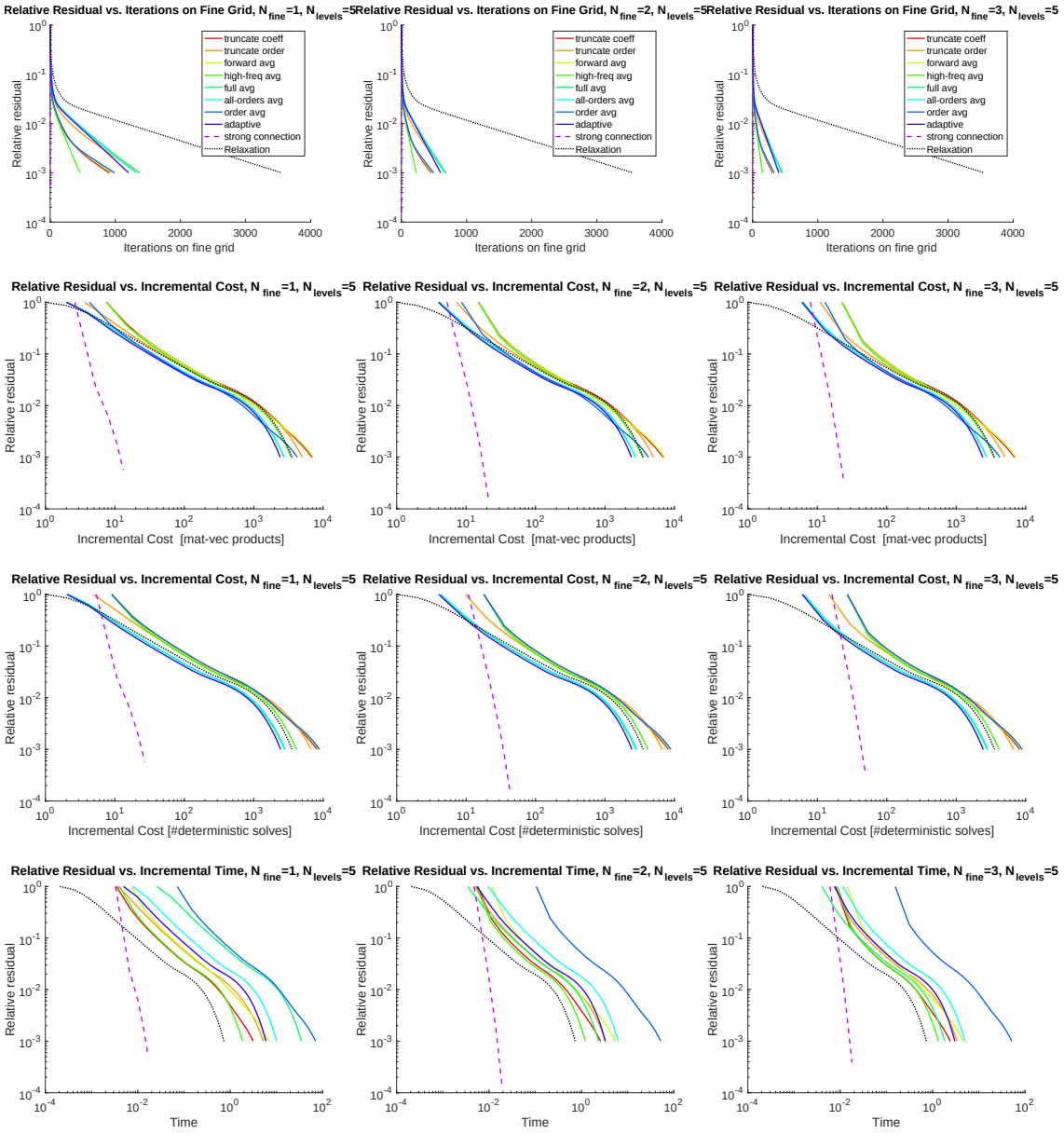


Figure 3.29: 1D heat test: 5-level pMG relative residual vs. number of iterations (top), incremental cost in terms of matrix-vector products (upper center) and number of deterministic solves (lower center), and time (bottom) for $N_F = N_C = \{1, 2, 3\}$ (left-right).

As for the pMG test cases, the performance is measured in terms of relative residual as a function of number of iterations, time (averaged over 10), and cost in terms of matrix-vector products and deterministic solves. Here, in (3.47) the cost of a lower-triangular matrix-vector product is calculated as

$$c_l = 2\left(\frac{\left(N_p N_E^l\right)^2 - N_p N_E^l}{2} + N_p N_E^l\right), \tag{3.51}$$

where N_p is the number of PCE coefficients (constant at each level), and N_E^l is the number of edges of the network at level l, while the number of deterministic solves is

$$c_l = N_{comp}^l \left(p+1\right)^d, \tag{3.52}$$

where N_{comp}^l is the number of components at level l, p is the order of the PCE, and d is the dimension of the stochastic space.

Note that, unlike a pMG coarsening, the accuracy of the stochastic representation of the unknown in terms of PCE (i.e., the edge size) is constant across levels, whereas the number of edges decreases at each level.

The results for $L = \{2, 3, 4\}$ levels are shown in Figures 3.30-3.35. First of all, as a consistency check, notice that $(2,1)$-hMG coincides with 2-level AMG (strong connection), as expected (see Figures 3.30-3.31). In general, for a 2-level hMG the slowest types are those with no overlap ($n_2 = 0$), whereas the fastest are those that maximize the overlap ratio ($(16,8)$, $(8,4)$, $(4,2)$: $n/m = 50\%$). Interestingly, among the fastest hMG types, $(8,4)$ is the best, probably because it achieves the best trade-off between accuracy and propagation speed.

For more than 2 levels, a performance analysis is less straightforward. As for the basic case, Figures 3.32-3.33 show that $(2,1) - (3,1)$-hMG is equivalent to 3-level AMG, as expected, and the slowest hMG types are those with no overlap at the second level ($n_2 = 0$). Although it is hard to identify the coarsening type that has the best performance for $L \geq 3$, a comparison across the levels shows that number of iterations, costs and times decrease significantly as the number of levels increases (see also Figures 3.34-3.35). This justifies further investigation on more complex cases.

Parameter	Description	Value
L_x	Length of the physical domain	1
N_{FE}	Number of FE	10
N_{comp}	Number of network components	64
p	PCE order	5
N_p	Number of PCE coefficients	21
T_1, T_2	Stochastic boundary conditions	Random (loaded)
csnType	Coarsening type	see Table 3.4
L	Number of hMG levels	$2, 3, 4$
intpType	Interpolation type	As above
smoother	Smoother for hMG	Gauss-Seidel
ω	Relaxation parameter	1
perms	Permutations	$1 : N_{comp}$
$N_{Vcycles}$	Maximum number of V-cycles	$10,000$
tol	Tolerance for stopping criterion	10^{-6}
$N_F = N_C$	Interations on fine/coarse grid	$1, 2, 3$

Table 3.3: hMG for linear 1D heat DDUQ network problem (matrix form): parameter setting.

2 levels	3 levels	4 levels
$(2,0)$	$(2,0)-(2,1)$	
	$(2,0)-(4,2)$	
$(2,1)$	$(2,1)-(3,1)$	$(2,1)-(3,1)-(3,1)$
	$(2,1)-(7,3)$	
$(4,0)$	$(4,0)-(4,1)$	
	$(4,0)-(4,2)$	
$(4,1)$	$(4,1)-(3,1)$	$(4,1)-(3,1)-(4,1)$
	$(4,1)-(9,3)$	
$(4,2)$	$(4,2)-(3,1)$	$(4,2)-(3,1)-(3,1)$
	$(4,2)-(6,1)$	
$(8,0)$	$(8,0)-(2,1)$	
	$(8,0)-(4,2)$	
	$(8,0)-(3,1)$	
	$(8,0)-(5,1)$	
$(8,1)$		
$(8,4)$	$(8,4)-(3,1)$	$(8,4)-(3,1)$
	$(8,4)-(8,1)$	
$(16,4)$	$(16,4)-(3,1)$	
$(16,8)$	$(16,8)-(3,1)$	

Table 3.4: hMG types for linear 1D heat DDUQ network problem.

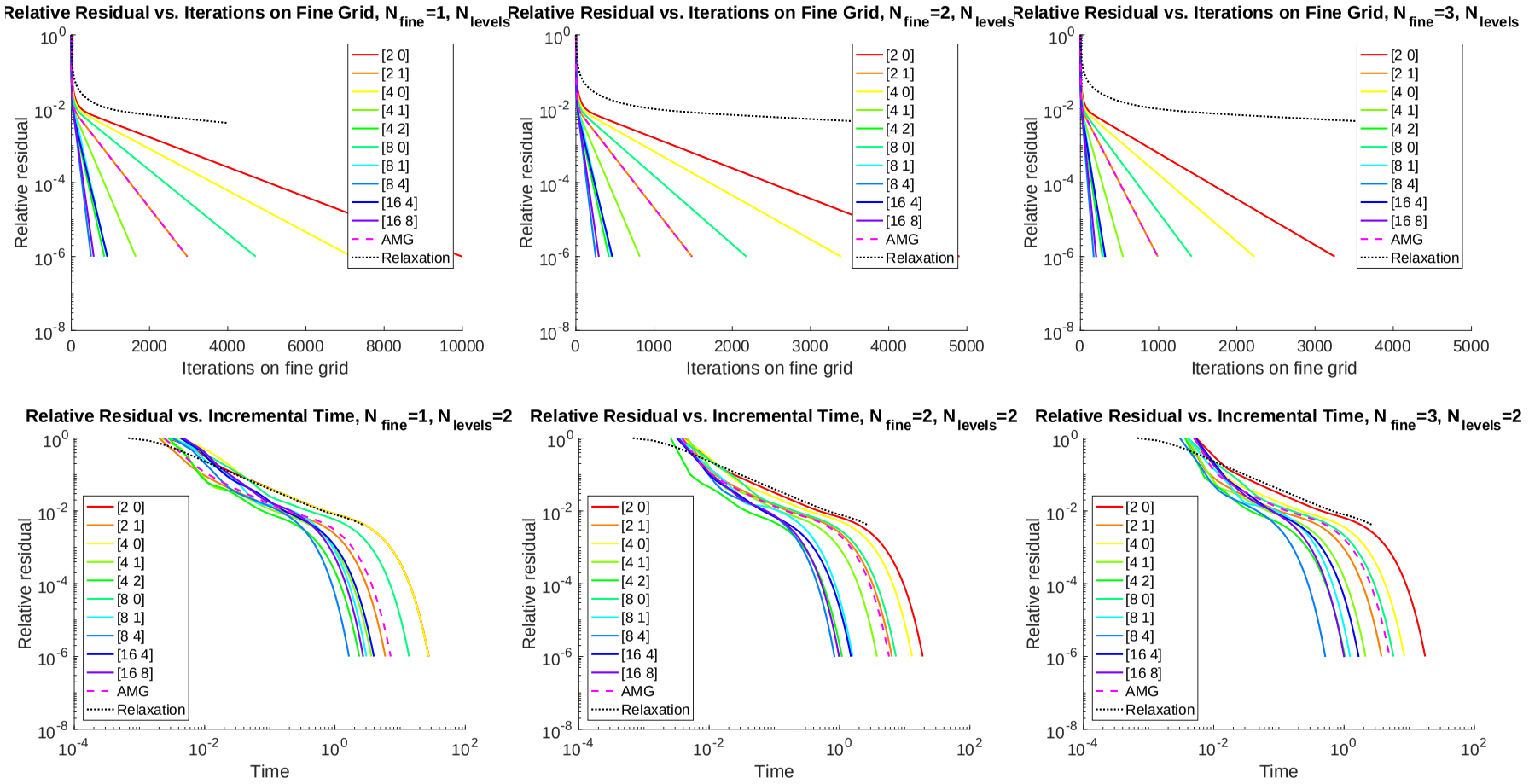


Figure 3.30: 1D heat test: 2-level hMG relative residual vs. number of iterations (top) and time (bottom) for $N_F = N_C = \{1, 2, 3\}$ (left-right).

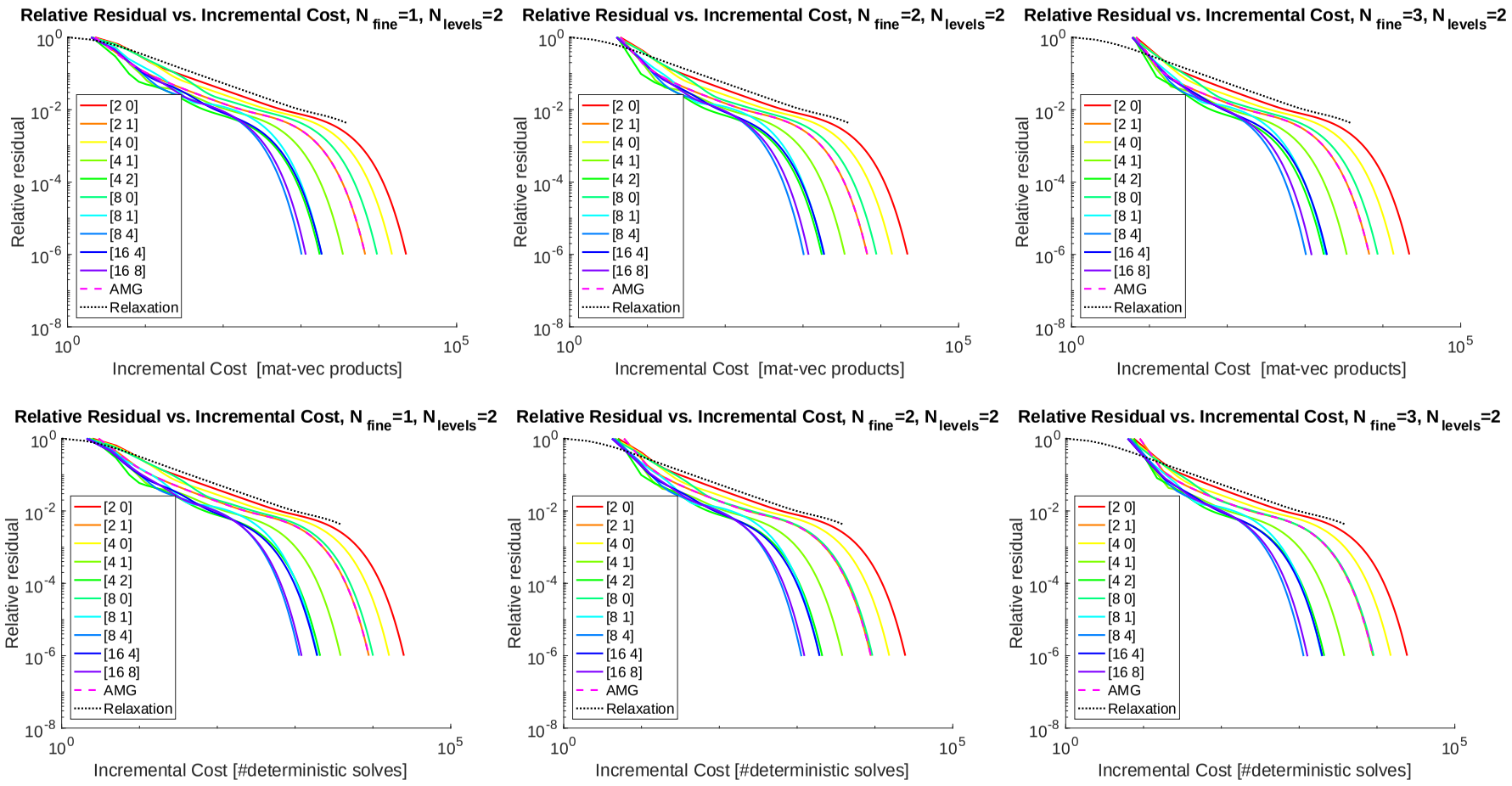


Figure 3.31: 1D heat test: 2-level hMG relative residual vs. incremental cost in terms of matrix-vector products (top) and number of deterministic solves (bottom) for $N_F = N_C = \{1, 2, 3\}$ (left-right).

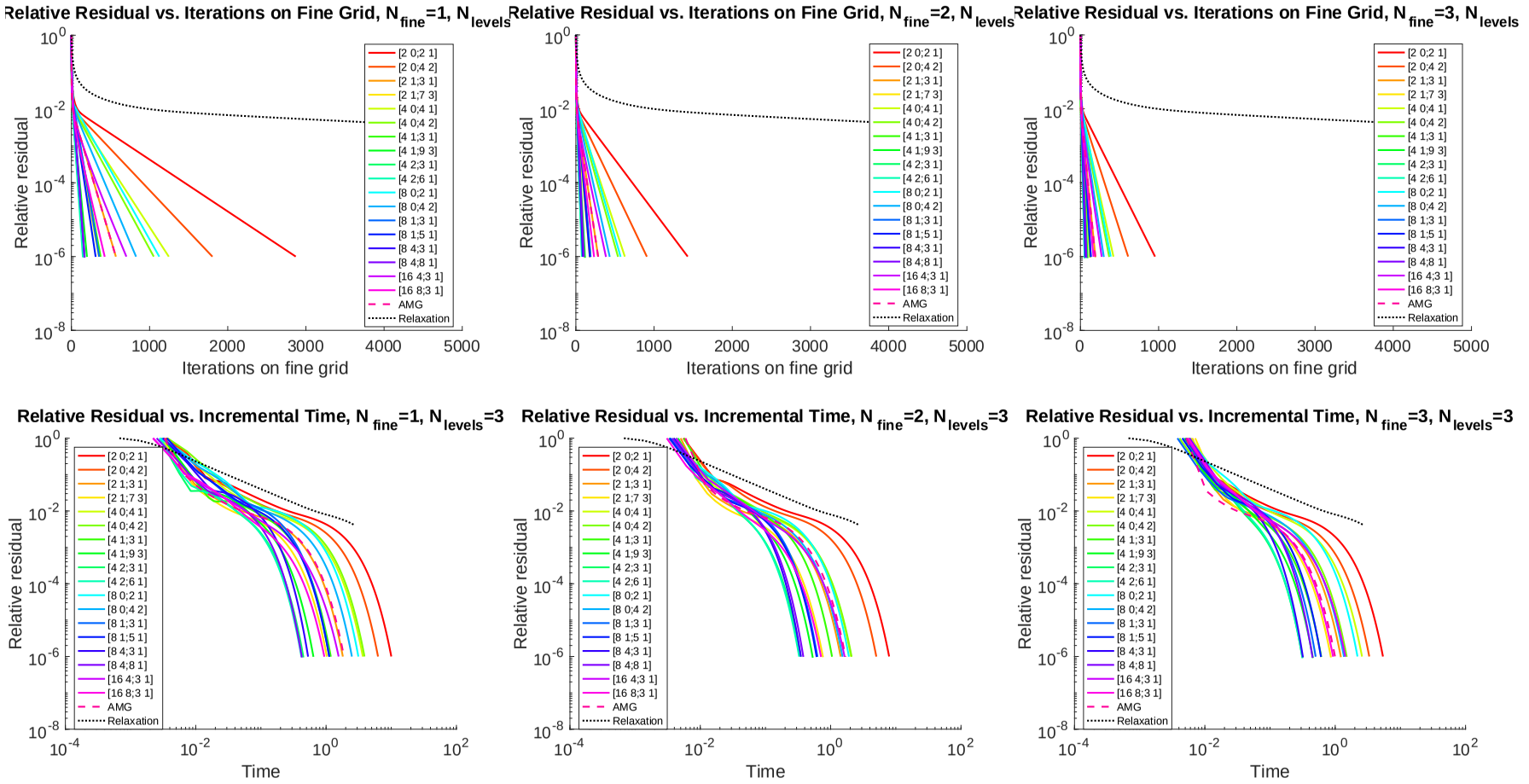


Figure 3.32: 1D heat test: 3-level hMG relative residual vs. number of iterations (top) and time (bottom) for $N_F = N_C = \{1, 2, 3\}$ (left-right).

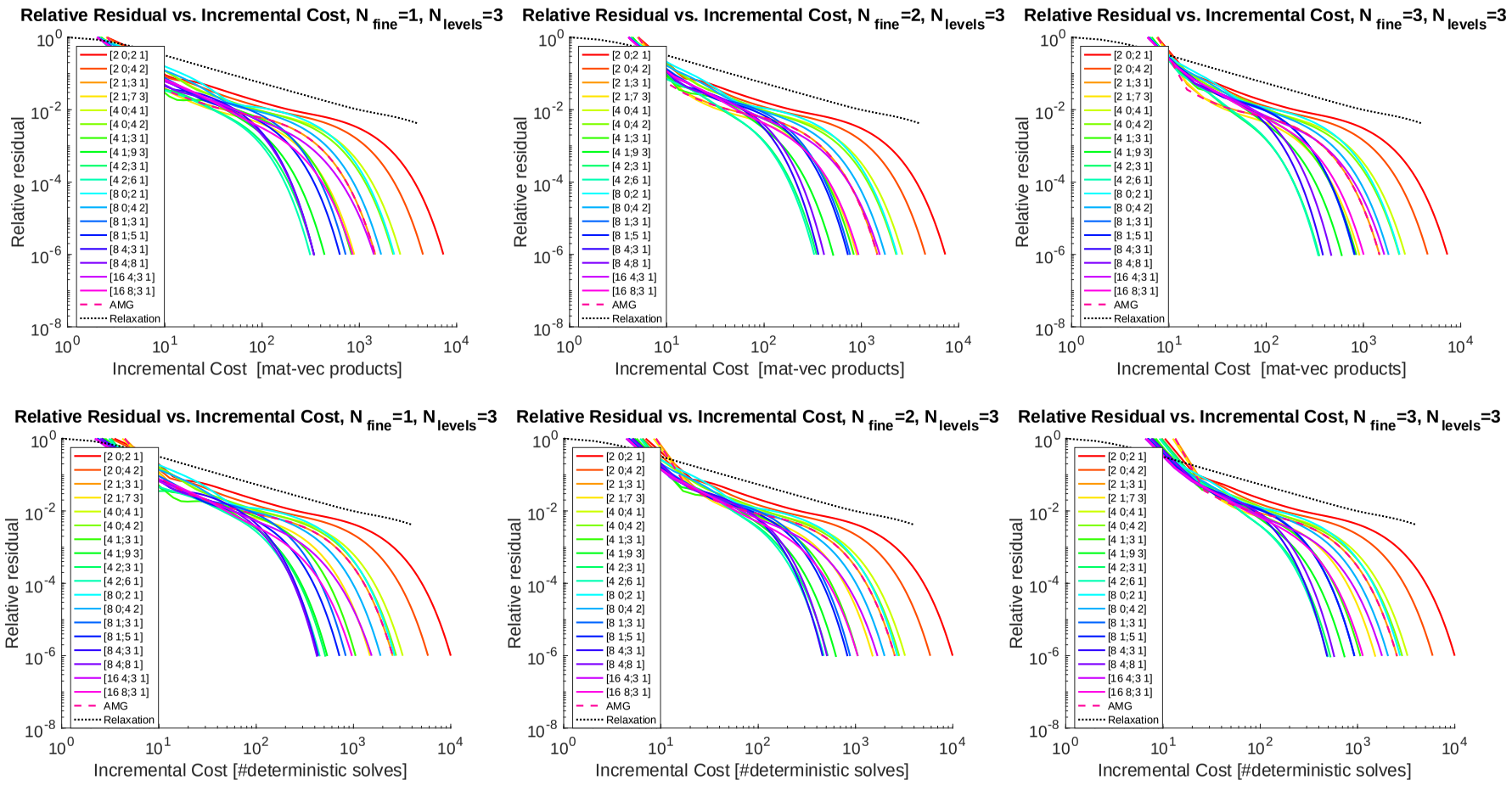


Figure 3.33: 1D heat test: 3-level hMG relative residual vs. incremental cost in terms of matrix-vector products (top) and number of deterministic solves (bottom) for $N_F = N_C = \{1, 2, 3\}$ (left-right).

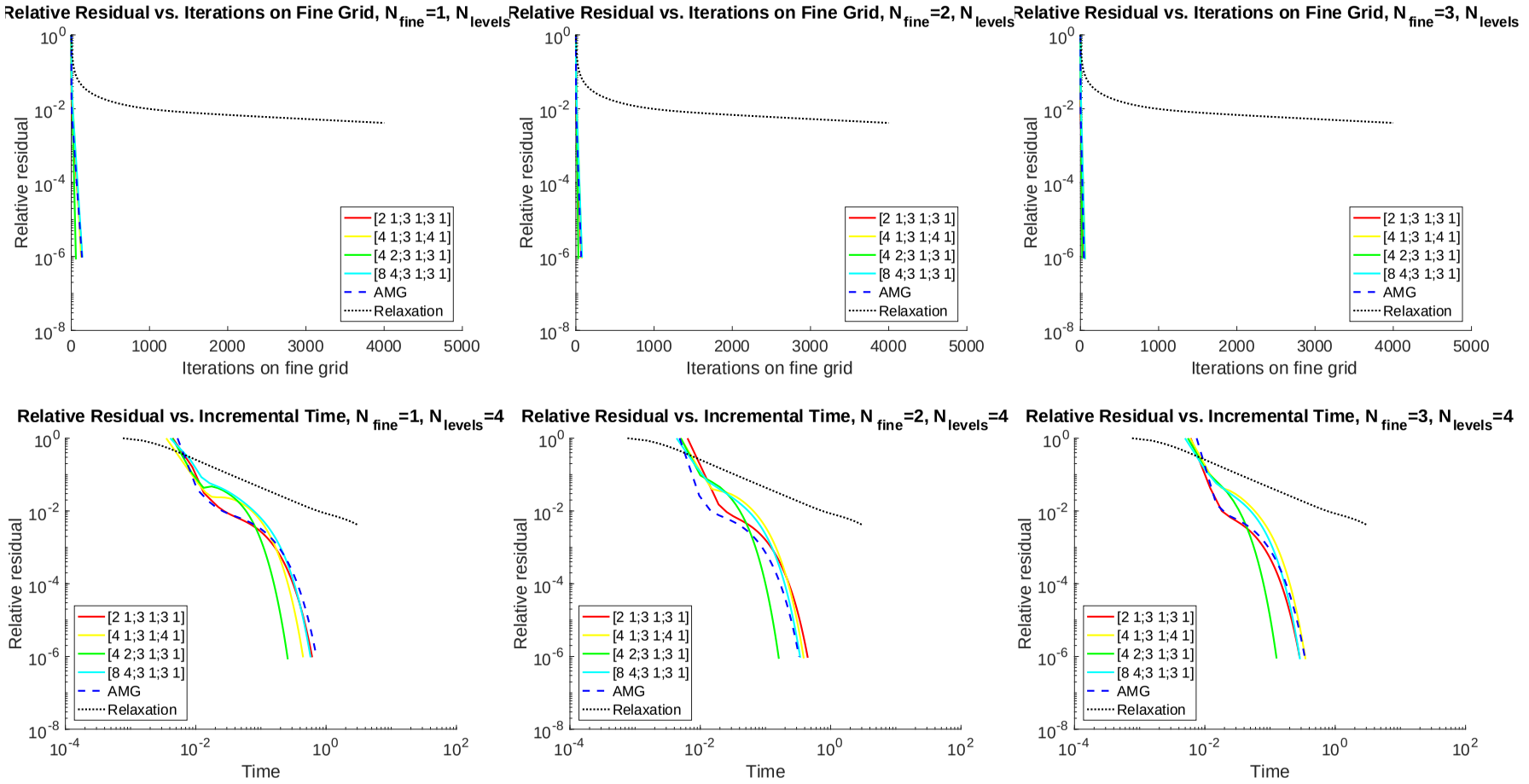


Figure 3.34: 1D heat test: 4-level hMG relative residual vs. number of iterations (top) and time (bottom) for $N_F = N_C = \{1, 2, 3\}$ (left-right).

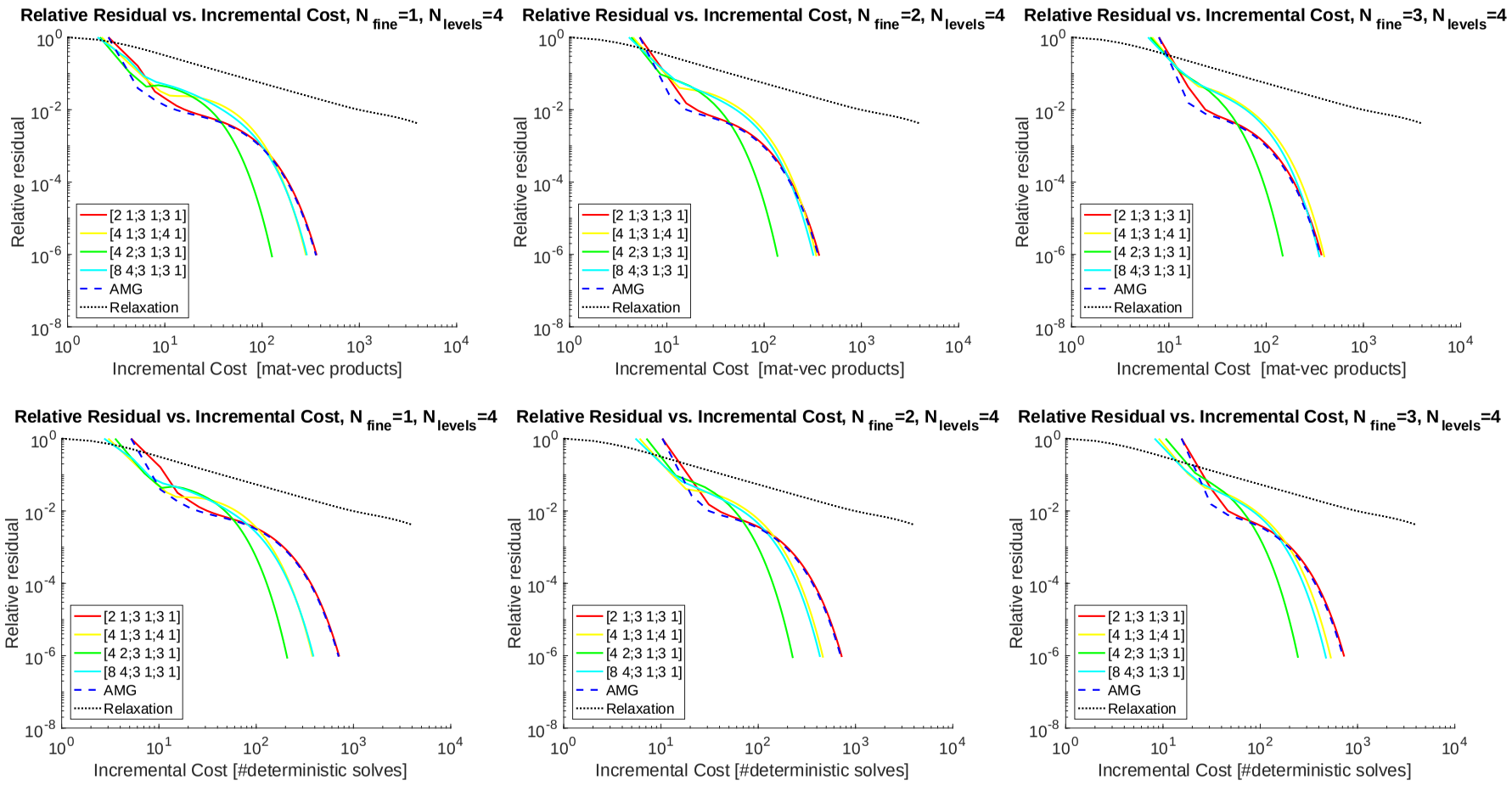


Figure 3.35: 1D heat test: 4-level hMG relative residual vs. incremental cost in terms of matrix-vector products (top) and number of deterministic solves (bottom) for $N_F = N_C = \{1, 2, 3\}$ (left-right).

3.6 h-Multigrid Methods for UQ in Networks

A matrix formulation of the network problem may not be available, either because of the intrinsic non-linearity of the problem, or because of the heterogeneity of the components. Therefore, while the notion of network coarsening can still be applied, the prolongation and restriction operators as well as the smoother and the solver on the coarse network need to be re-designed. In what follows we discuss how it is possible to generalize the matrix version of the methods to network form, highlighting some delicate issues that need particular care, or desirable mathematical properties that require some effort to be guaranteed. Numerical results are work in progress and will be presented in [**?**].

3.6.1 Prolongation and restriction operators

Since the hyper-nodes of the coarse networks are aggregates of the nodes of the fine network, the edges of the coarse network are a subset of the edges of the fine network. Consequently, the restriction operator R selects the edges of the fine network that are also edges of the coarse network and injects the corresponding values of the solution. Conversely, the construction of a prolongation operator is more involved. First of all, notice that the effect of a network propagation at the coarse level is that new values for the outputs are generated not only for the edges of the coarse network, but also for those edges of the fine network that are dropped during the coarsening. If the mesh is preserved across levels, such values are directly accessible. Differently, if the mesh of a component of the coarse network is coarser than the corresponding one at the fine level, the values can be retrieved via interpolation. Either way, the solution after a propagation in the coarse network is defined both on the edges of the coarse network and on "dummy" edges within each component, corresponding to the dropped edges of the fine network. The dummy edges are inherited from the fine-level network and remain inactive during the propagation in the coarse network. However, since the solution to a propagation in the coarse network is defined herein,

it can be injected to the fine network.

It is desirable for a generic MG method to satisfy the variational property defined by [42]

$$\boldsymbol{f}_H(\mathbf{x}_H, \mathbf{u}_H) = R\boldsymbol{f}_h(P\mathbf{x}_H, \mathbf{u}_h). \tag{3.53}$$

For an hMG method this is hard to obtain because of the non-uniqueness of the solution on the overlapping region, but the prolongation operator can be designed so that relation (3.53) holds (at least) for some choices of propagation. For instance, suppose that the propagator $\boldsymbol{f}_H(\mathbf{x}_H, \mathbf{u}_H)$ is one iteration of Gauss-Seidel or Jacobi, and that the prolongation operator fulfills the additional property that, if an output edge of component i of the fine network is an output edge of component j of the coarse network, then the input edges of component i are injected from component j. Then, the variational property (3.53) is satisfied. To see this, consider the Domain-Decomposition setting in Figure 3.36 (top) with a 2-level coarsening of type $(3, 1)$, and corresponding network representation as in Figure 3.36 (center). Suppose that all the outputs are initialized to zero, with some non-zero exogenous inputs $\mathbf{x}_L$, $\mathbf{x}_R$. Then, after one iteration of Jacobi on the coarse grid, the solution is as in Figure 3.36 (top). Now, we wish to prolongate such solution to the fine network, run one Jacobi iteration on the fine network, restrict back to the coarse network and find the same outputs we started with.

Consider output $\mathbf{x}_2$ of the fine network, which is output $\hat{\mathbf{x}}_1$ of component Ω_1^C of the coarse network. The value at such location on the fine network depends on outputs $\mathbf{x}_1$ and $\mathbf{x}_7$. If the prolongation operator injects the value to output $\mathbf{x}_7$ from the corresponding location of the component Ω_2^C, the solution at output $\mathbf{x}_2$ on Ω_2^F will be inconsistent with the solution on Ω_1^C. Therefore, since output $\mathbf{x}_2$ is generated by component Ω_2^F, which belongs to component Ω_1^C, all the inputs to component Ω_2^F have to be injected from Ω_1^C.

More formally, consider the dual network in Figure 3.36 (bottom), which is obtained from the network above by switching the role of nodes and edges (subdomains and outputs, respectively). This representation is convenient to highlight the dependen-

cies between edges. The outputs of the fine network that are outputs of the coarse network are highlighted in red. For instance, output $\mathbf{x}_2$ is determined by output $\mathbf{x}_1$ and output $\mathbf{x}_7$ through component Ω_2^F. Note that overlapping components leads to the replication of outputs (in the intersection). For each "red" node ($\mathbf{x}_2 \in \Omega_1^C$ and $\mathbf{x}_6 \in \Omega_2^C$), select the adjacent input nodes ($\{\mathbf{x}_1, \mathbf{x}_7\} \subset \Omega_1^C$ and $\{\mathbf{x}_3, \mathbf{x}_5\} \subset \Omega_2^C$). In order for the variational property to be satisfied, the solution on this instance of the fine edges has to be injected. Any other duplicate ($\mathbf{x}_7 \in \Omega_2^C$, $\mathbf{x}_3 \in \Omega_1^C$) has to be discarded.

Property (1) is relatively easy to implement in the 1D case because each coarse edge can correspond to one and only one fine edge. However, as the dimensionality of the domain grows, each edge of the coarse network can be a subset of edges of the fine network (see Figure 3.37), which makes the variational property hard to implement.

3.6.2 Smoother and coarse-grid operators

The relaxation operator is a block-Jacobi or block-Gauss-Seidel network solver. A Jacobi iteration is equivalent to "splitting" all edge data so that all the components can perform uncertainty propagation independently and simultaneously at each iteration. Gauss-Seidel, instead, is equivalent to decomposing the network in a directed acyclic graph, so that each iteration corresponds to propagation in a feed-forward network.

Note that, since each coarse component is a sub-network, one may define the local solver on each coarse component to be a network solver. However, in such a way a coarse network solve would be equivalent to a solve on the fine network, with a particular choice of permutations. Therefore, in order to apply a MG method, the coarse local solver must be a (stochastic) solver on the whole physical coarse component. Moreover, the cost of one deterministic solve on a coarse component should be approximately the same as the cost of a deterministic solve on a fine component. In a traditional Multigrid fashion, this can be achieved by coarsening

the mesh on each coarse component. Alternatively, one may fix the mesh size, and reduce the cost of a deterministic solve via Reduced-Order Models.

3.6.3 The definition of the residual

We wish to solve the network problem (3.5) on the fine network. To easy the reading, with abuse of notation we will omit the adjacency matrix $\boldsymbol{I}_{\mathbf{x}}^{\boldsymbol{y}}$, to highlight the fixed-point nature of the problem, which leads to the iterative procedure

$$\mathbf{x}_h^k = \boldsymbol{f}_h\left(\mathbf{x}_h^{k-1}, \mathbf{u}_h\right), \tag{3.54}$$

with corresponding residual

$$\mathbf{r}_h^k = -\mathbf{A}_h(\mathbf{x}_h^k, \mathbf{u}_h) = \boldsymbol{f}_h\left(\mathbf{x}_h^k, \mathbf{u}_h\right) - \mathbf{x}_h^k. \tag{3.55}$$

The definition (3.55) is non-intrusive, since it only involves the network solver (3.5). Differently, a more intrusive definition could include the residual of the PDE solver defined on each subsystem. For the sake of completeness, we present the FAS formulation for this case as well.

Consider the local stochastic problem defined on the i-th subsystem

$$Q_i(\mathbf{x}_i, \mathbf{u}_{p,i}) = \mathscr{F}_i(f_i, \mathbf{x}_i, \mathbf{u}_{BC,i}), \tag{3.56}$$

where Q defines a (PDE) problem on each local component, $\mathbf{u}_{p,i}$ are the (stochastic) parametric inputs, f_i is the forcing term of the deterministic problem, and $\mathbf{u}_{BC,i}$ are the (stochastic) exogenous boundary conditions, and the corresponding vectorization

$$\mathbf{Q}(\mathbf{x}, \mathbf{u}_p) = \mathscr{F}(f, \mathbf{x}, \mathbf{u}_{BC}). \tag{3.57}$$

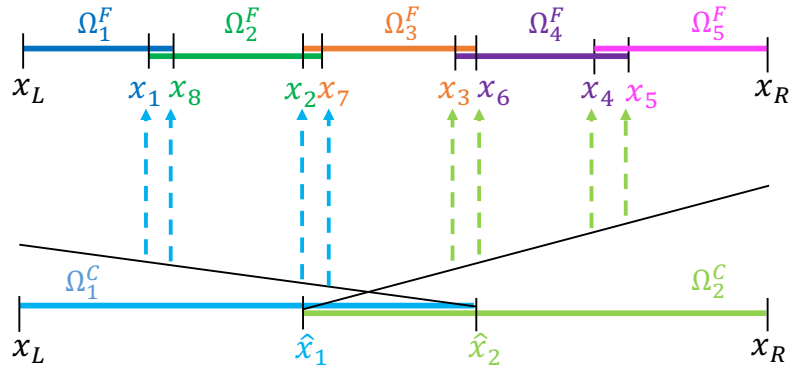
This leads to the iterative process on the fine network

$$\mathbf{Q}_h(\mathbf{x}_h^k, \mathbf{u}_{h,p}) = \mathscr{F}_h(f_h, \mathbf{x}_h^{k-1}, \mathbf{u}_{h,BC}), \tag{3.58}$$

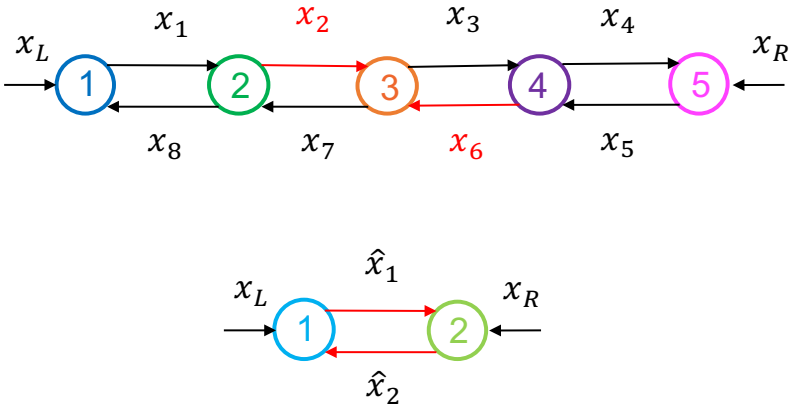
with corresponding residual

$$\mathbf{r}_h^k = \mathscr{F}_h(f_h, \mathbf{x}_h^k, \mathbf{u}_{h,BC}) - \mathbf{Q}_h(\mathbf{x}_h^k, \mathbf{u}_h). \tag{3.59}$$

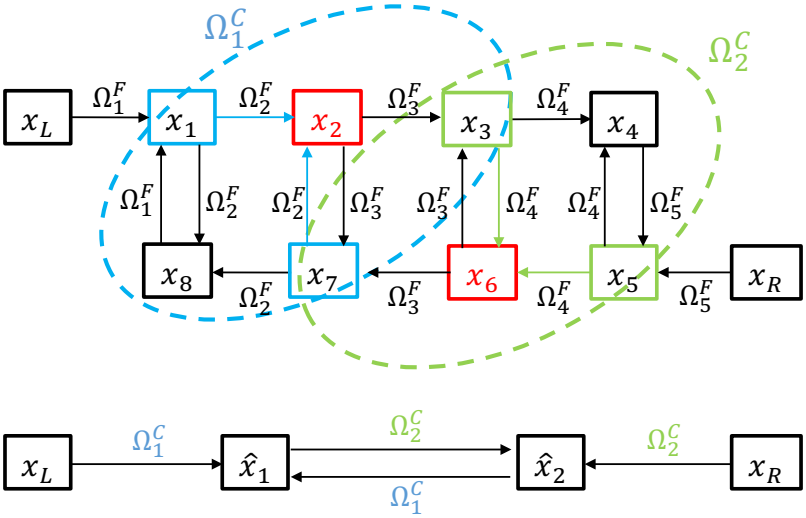
One step of FAS consists of the following steps:



(a) Domain-Decomposition setting: prolongation operator (dashed arrows) between fine (top) and coarse (bottom) domain.



(b) Fine (top) and coarse (bottom) network representation.



(c) Fine (top) and coarse (bottom) dual network.

Figure 3.36: Variational property for hMG: 1D example.

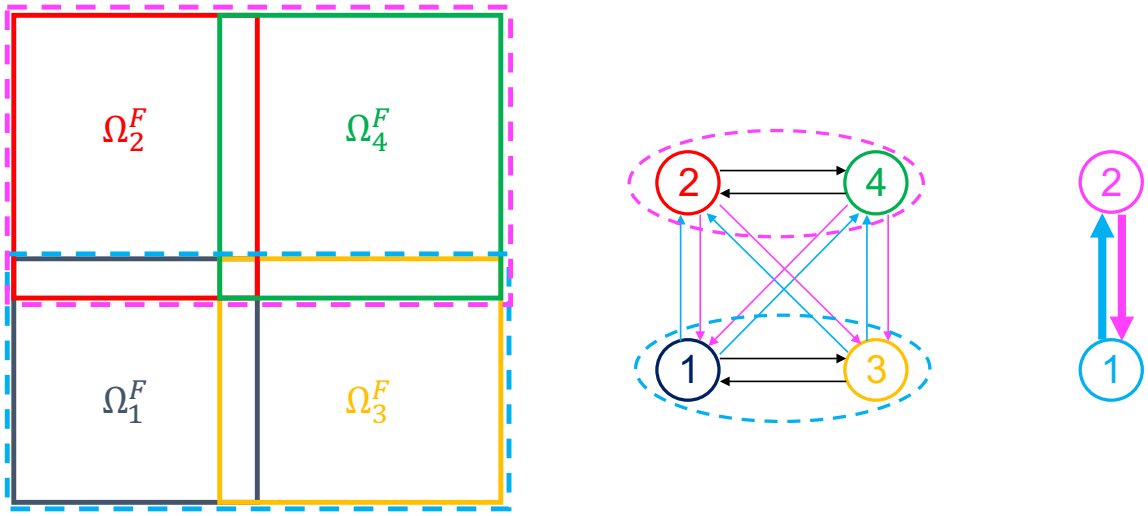


Figure 3.37: $(2,0) \times (1,0)$-hMG coarsening: Domain Decomposition setting (left), fine (center) and coarse (right) network representation. The action of the coarsening is depicted with dashed lines. Note that the (thick) edges of the coarse network are subsets of (thin) edges of the fine network.

1. Pre-smoothing: obtain $\mathbf{x}_h^k$ and $\mathbf{r}_h^k$;
2. If R and P are the restriction and prolongation operators between the fine and the coarse network, respectively, restrict $\mathbf{x}_{H,0} = R\mathbf{x}_h^k$.
3. Compute $\mathbf{Q}_H(\mathbf{x}_{H,0}, \mathbf{u}_{H,p})$, where $\mathbf{Q}_H$ is the global stochastic solver on the coarse network. Note that this operation consists of:
   - Evaluating the PCE coefficients of each local solution at the quadrature nodes (or samples);
   - Building the local matrix corresponding to each sample and applying it to the sampled solution;
   - Calculating the PCE coefficients of such resulting samples.

Solve the coarse-network problem

$$\mathbf{Q}_H(\mathbf{z}_H, \mathbf{u}_{H,p}) = \mathbf{Q}_H(\mathbf{x}_{H,0}, \mathbf{u}_{H,p}) + R\mathbf{r}_h^k \tag{3.60}$$

4. Compute the error $\mathbf{e}_H = \mathbf{z}_H - \mathbf{x}_{H,0}$ and prolongate it to the fine network: $\mathbf{e}_h = P\mathbf{e}_H$;

5. Correct the solution on the fine grid $\mathbf{x}_h^k \leftarrow \mathbf{x}_h^k + \mathbf{e}_h$;

6. Post-smoothing: Calculate an approximate solution of $\mathbf{Q}_h(\mathbf{x}_h^k, \mathbf{u}_{h,p}) = \mathscr{F}_h(f_h, \mathbf{x}_h^{k-1}, \mathbf{u}_{h,BC})$ on the fine network with initial guess $\mathbf{x}_h^k$.

The MG formulation that derives from definition (3.59), however valid, in some sense defeats the purpose of the DDUQ method to treat each component (and the relative solver and discretization) as a black-box. Therefore, definition 3.55 will be used in future numerical experiments.

3.7 Conclusions

In this Chapter we addressed the issue of propagating uncertainties in large-scale networks, which is often impractical or even intractable, due to the computational burden. While domain-decomposition based methods for UQ in large-scale problems have been proposed in the literature, they are still tied to the need of (unaffordably) expensive high-fidelity simulations, or to poor scalability properties. The DDUQ method that we propose is a bottom-up approach that exploits the benefits of Domain Decomposition in an offline phase to decompose the system in elementary components, so that UQ (and potentially model reduction) can be performed online only locally, at the (tractable) subsystem level. This approach is (i) rigorous, since it enforces strong compatibility constraints by matching random variables; (ii) weakly and strongly scalable with respect to the number of components; (iii) general, because each network component can be treated as a black-box with its own

physics, solver, discretization, etc.; (iv) modular, because it facilitates the design of the network configuration by handling components as in a Lego-like approach.

While this approach facilitates the parallel execution of the solver, with a dramatic reduction of the execution time, the underlying fixed-point iteration is known to feature slow convergence. Besides Anderson Acceleration, we propose novel multigrid methods tailored to the network problem, to further accelerate convergence. While lower-order truncations of the PCE are not beneficial, coarsening the network by clustering adjacent components improves convergence even more than the traditional algebraic multigrid method. The method has been validated on a wide variety of MG configurations for 1D problems that admit an algebraic formulation. We refer to [**?**] for more elaborate numerical results in higher-dimensional domains.

We explore the potential of DDUQ combined with model reduction in Chapter 4, on 3D vector problems. Further investigations will address non-stationary problems, and the combination of the multigrid methods proposed here with Anderson Acceleration.

Chapter 4

Reduced-order models for uncertainty quantification in the cardiovascular network via DDUQ

Acknowledgements. This chapter contains part of the content of the paper [96], written in collaboration with Alonso Mansilla Alvarez, Pablo Blanco, Kevin Carlberg, and Alessandro Veneziani, that has been part of the research of the author during a visiting period at the Scientific Computing National Laboratory of Brazil.

4.1 Introduction

Over the last decades, we assisted to a breakthrough of computational hemodynamics into clinical practice, as a tool to support clinicians in the decision process of surgical planning and clinical trials [185, 54, 76]. This change, that is expected to consistently affect the implementation of clinical protocols, has been possible thanks to the recent advances in mathematical and computational models, as well as to the availability and accessibility of computational resources. However, simulating the complex physiological, biological, and mechanical dynamics of the whole cardiovascular system (CVS) or of portions of it - not to mention the integrated simulation of the interaction between the CVS and other physiological compartments - is still an

open challenge. For such large-scale problems, the amount of computational power required by high-fidelity models, such as the Finite Element Method (FEM), can be prohibitive, and the access to off-site or remote computational platforms is often impractical for many users, such as hospitals, due to privacy and security policies that have to be enforced at all times [92, 91]. Moreover, (i) the time due to the physical transfer of data, (ii) the potential wait time associated with the use of shared resources, (iii) the execution time of the simulation, and (iv) the post-processing operations, may make numerical models impractical for fast-paced clinical environment [97] (see Chapter 5).

Reduced-order models (ROMs) have been developed and used to alleviate the complexity of the problem. One-dimensional models, for example, average the cross-sectional dynamics, resulting in a 1D description of the blood vessel, representative only of the main-stream dynamics [104, 76, 33]. Then, the effect of other physiological compartments or of the peripheral circulation on the local hemodynamics can be encoded in the boundary conditions via lumped-parameter models [197, 76, 33]. These models are widely employed to study the propagation of pressure waves induced by the mutual interaction between the fluid and the wall of compliant blood vessels. In fact, the interplay between anomalous pressure waves and pathologies like atherosclerotic plaques or devices like stents is of great interest from a medical viewpoint [161, 85, 177]. However, the transverse dynamics of the blood flow, which plays a major role in the initiation and development of diseases like stenoses or aneurysms [115], are completely ignored, which makes these models often impractical from the clinical perspective.

In addition, both high- and low-fidelity mathematical and numerical models still suffer from several limitations, such as (i) underlying assumptions that may not be true in general [179]; (ii) missing data, such as boundary conditions, physical forces, parameters, geometry, flow-split, and material properties, among others [72, 194, 43]; (iii) intrinsic variability of parameters and data - not only from patient to patient, but also within the same individual, depending on physiological conditions (e.g., rest

vs. stress) [76, 74]; (iv) numerical errors in the approximation process. These limitations motivated the design of mathematical and numerical techniques to *quantify* the model uncertainty (or reliability). However, because of these factors, feasibility and reliability of numerical simulations are in fact competing goals. Quantifying the uncertainty (reliability) of accurate mathematical models is often computationally unfeasible, while simplified, computationally affordable models are not accurate enough for clinical purposes.

Uncertainty quantification (UQ) studies on the cardiovascular network (CVN) have been performed, e.g., in [201, 51, 166, 189, 165]. In [201], uncertainties are described via a generalized polynomial chaos expansion (gPCE), and sensitivity analysis is performed via stochastic collocation methods. The study is applied to 1D models of a simple bifurcation with one source of uncertainty (a model parameter encoding geometric and mechanical properties of the vessel) for each branch, and of a realistic network of 37 vessels, with one random parameter for each segment. A more sophisticated analysis is carried out in [51] by splitting the flow into an elastic and a visco-elastic component, and studying the sensitivity of the solution in a one-dimensional arterial network with 103 segments and 47 outflow boundaries. Flow rate, terminal resistance, reference area of the arterial wall, and Young modulus, are first considered individually as univariate sources of uncertainty. Then, wall thickness, viscosity, density, characteristic time, capacitance, and external and venous pressure are added to the former uncertainties and studied as an ensemble. Finally, 47 values of resistance and 103 instances of reference area - one for each outflow and segment, respectively - are considered. While these studies are realistic in terms of dimension of the stochastic problem and in the sources of uncertainty, they still rely on the simplistic 1D reduced models.

UQ preliminary studies in 3D vascular models via stochastic collocation were first performed in [165], and applied to idealized models of an abdominal aortic aneurysm (AAA) and of a carotid artery bifurcation, and to a patient-specific geometry of a Fontan surgery for congenital heart defects. Uncertainties are modeled via simple

distributions (Gaussian, Uniform) and the number of random parameters is limited to one or two. Schiavazzi et al. in [166] assess the confidence of virtual surgery hemodynamics predictions for single ventricle palliation. First, distributions of boundary conditions that match the observed right pulmonary artery flow split and average pulmonary pressure are obtained via Bayesian parameter estimation. Then, the uncertainty is propagated to characterize predictions of post-operative hemodynamics in models with and without pulmonary artery stenosis. The solution of the forward problem is obtained with 3D geometric multiscale simulations, that required the computational power of a large parallel cluster. The more recent studies in [189] assess confidence in predictions of wall shear stress and wall strain in 3D patient-specific geometries with arterial and venous grafts, obtained with multi-scale models that account for uncertainty in peripheral compartments and material properties. A stochastic submodeling approach is developed to reduce the computational cost of several multi-scale solutions required by the stochastic model. A polynomial surrogate is trained on a set of full-model simulations, to predict the boundary conditions of the sub-model (the grafts). In this way, a large number of less expensive simulations can be run only in the regions of interest.

Although far from being exhaustive, two major challenges or limitations related to conducting UQ analysis in the CVS stand out: (i) Reduced 1D models can be inaccurate in capturing anomalies of the physiology in presence of cardiovascular pathologies like stenoses or aneurysms, especially since quantities of clinical interest such as the wall shear stress (WSS), related to cross-sectional dynamics, are inaccessible; (ii) Full 3D models are extremely costly and require computational resources that may not be easily accessible by users like healthcare institutions.

In this work we propose a new flexible reduced-order model of the CVS to retrieve a level of accuracy comparable to 3D models roughly at the same computational cost as 1D models, while efficiently quantifying its reliability. This is achieved by (i) enabling rapid and scalable simulations in decomposable systems - such as the cardiovascular network, by propagating the uncertainty information via Domain De-

composition (DD) techniques, and performing uncertainty quantification and model reduction locally, at the component level, following the guidelines of the DDUQ method introduced in the previous Chapter, and by (ii) replacing the simplified 1D models with *educated* reduced models capable of retaining the local cross-sectional dynamics, by properly enriching the one-dimensional main dynamics, as illustrated in Chapter 2.

This Chapter is organized as follows. Section 4.2 presents an overview of the so-called TEPEM reduced-order model, already mentioned in Chapter 2. In Section 4.3, we describe the UQ problem for hemodynamics applications, and the DDUQ-TEPEM coupling proposed for the efficient and accurate quantification of uncertainties. Numerical examples are reported and analyzed in Section 4.4. In Section 4.5 the DDUQ method is validated on unsteady stochastic problems based on 1D reduced models, and concluding remarks are drawn in Section 4.6.

4.2 The Transversally-Enriched Pipe-Element Method

Recently proposed as a numerical strategy oriented to simulate the blood flow in three-dimensional patient-specific vasculature, the Transversally-Enriched Pipe-Element Method (TEPEM) has proven to be an effective strategy to deal with the trade-off between accuracy and computational burden, which strongly limits the massive use of current numerical approximations in real clinical applications. In [127, 128], this methodology was introduced and extensively tested on synthetic and patient-specific vasculatures. Several numerical results addressed in former works highlight the TEPEM ability to speed up hemodynamic simulations while maintaining a level of precision comparable with the one provided by high-fidelity FE method. As for Hi-Mod, the effectiveness of this novel numerical strategy relies on the combination of an *educated* discretization of the geometrical domain - tailored to geometries that feature a leading direction coupled with transverse dynamics that can be locally relevant, and a sort of enriched one-dimensional approximation for the physical fields

which allow us to take into account the transverse dynamics, commonly dropped by classical reduced-order models and of major relevance to understand the local flow dynamics. Although TEPEM and HiMod were originally conceived to speed-up blood flow simulations, the strategy is quite general and can be applied to any problem set on pipe-like domains (e.g., gas/water pipes networks, fluvial networks, etc.). The basic ingredients of TEPEM are described in this section.

4.2.1 Pipe discretization strategy

In the TEPEM methodology, the region of interest is assumed to be a connected network of tubular structures, as presented in Figure 4.1. Let us consider a domain $\Omega \subset \mathbb{R}^3$ of this type. By slicing this structure along the centerline we obtain a one-dimensional partition composed by a reduced number of elements, named as pipe-elements, and generically denoted by $\mathscr{K}$. This partition, $\mathscr{T}_h(\Omega)$, is characterized by the mesh parameter h, that stands for the axial length of the pipe-elements.

Each pipe-element $\mathscr{K} \in \mathscr{T}_h(\Omega)$ clearly identifies the axial direction, where the fluid is driven, and the direction where the secondary dynamics occurs, named as *transversal direction*. For each element, a geometrical mapping $\chi_{\mathscr{K}} : \mathscr{K}_0 \mapsto \mathscr{K}$ is defined between the reference pipe-element $\mathscr{K}_0 = [-1,1]^3$ (in $\xi\eta\zeta-$space) and the physical pipe element (in $xyz-$space) as:

$$\chi_{\mathscr{K}}(\xi,\eta,\zeta) = \sum_{k=1}^{3}\sum_{i=1}^{12} \mathbf{x}_i^k \mathcal{S}_i(\xi,\eta) Q_k(\zeta), \tag{4.1}$$

where $\{\mathcal{S}_i : i = 1,\dots,12\}$ is the set of cubic Serendipity functions defined in the square $[-1,1]^2$ (see [204]), the set $\{Q_k : k = 1,2,3\}$ is the classical Lagrangian basis for the space $\mathbb{P}_2([-1,1])$ (quadratic polynomials defined on the segment $[-1,1]$), and the points $\{\mathbf{x}_i^k\}$ are selected on the lateral surface of the deformed element. As a result, the geometrical representation of each pipe-element relies on the combination of piecewise cubic and quadratic polynomials. The main difference between TEPEM and HiMod lies precisely in the mapping between the physical and the reference

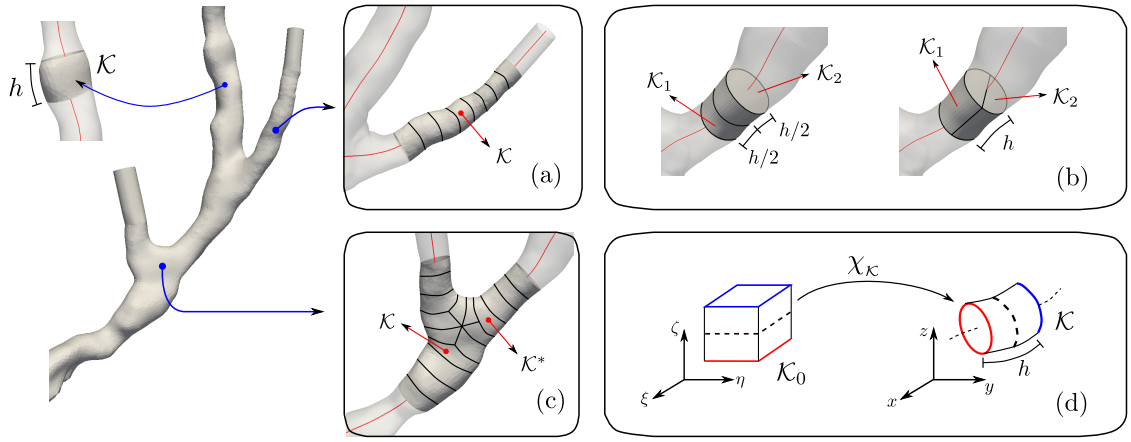


Figure 4.1: Details of a pipe-type discretization of a patient-specific vasculature, red line stands by the centerline. (a) Discretization of a non-branched region slabbing the geometry along the centerline. (b) Mesh refinement by axial (left) and transversal (right) split. (c) Discretization of a junction through the use of the transition element $\mathscr{K}^*$. (c) Mapping relating deformed and reference pipe-element.

element: while the former method maps a tubular structure into the unit cube - which is very convenient because of the Cartesian setting but introduces singularities (corners), the latter employs a unit cylinder as a reference domain, resulting in a smooth map (which is essentially a scaling of the physical radius) that, however, suffers from the intrinsic singularity of a polar reference system (the origin).

Special attention is needed to perform a pipe-type discretization in complex regions, as junctions, due to the lack of a mainstream direction inside those structures. As seen in panel (a) in Figure 4.1, tubular branches are discretized by slabbing the geometry along the centerline and condensing the local secondary dynamics in a single element. As commonly performed in classical by-element strategies, each element can be subdivided to locally refine the mesh. In this case, each pipe-element (with length h) can be split in two ways: (i) In the axial direction, obtaining two pipe-elements with axial length $h/2$, or (ii) In the transversal direction, obtaining two pipe-elements with the same length h, each covering a portion of the transversal section. Both al-

ternatives are outlined in panel (b) in Figure 4.1. This refinement strategy allows us to naturally cluster two (or more) pipe-elements in the transversal section of regions with higher complexity (geometrical or physical) and, specifically for the case of discretize branched domains, decompose the mesh in one sub-mesh where a single pipe-element is employed for the local secondary dynamics (non-bifurcated branches) and another where two pipe-elements are considered in the transversal direction (for the discretization of the junctions). Finally, these two sub-meshes are conformally coupled through the introduction of the so-called transition element, denoted by $\mathscr{K}^*$, as can be seen in panel (c) in Figure 4.1. For a deeper description of the pipe-type discretization, as well as further details of the transition element, the reader is referred to [126, 128].

4.2.2 Transversally enriched approximation

The TEPEM and HiMod can be understood as a smooth fashion to link one-dimensional approximations (where the transversal dynamic is commonly dropped) and full three-dimensional approximations. The ability to *a priori* control the model capabilities relies in the way in which physical fields are approximated in the reference element. Any scalar field w, defined in the domain Ω, is approximated with TEPEM by the function w^h defined by

$$w^h \circ \chi_{\mathscr{K}}(\xi,\eta,\zeta) = \sum_{k=1}^{\mathsf{s}+1} \left(\sum_{i,j=1}^{\mathsf{p}+1} w_{ijk}^h \phi_i(\xi)\phi_j(\eta) \right) \varphi_k(\zeta), \qquad \mathscr{K} \in \mathscr{T}_h(\Omega), \tag{4.2}$$

where $\mathscr{T}_h(\Omega)$ is a pipe-type discretization of Ω, p and s are positive integer numbers, and the sets $\{\phi_i : i = 1, \ldots, \mathsf{p}+1\}$ and $\{\varphi_i : i = 1, \ldots, \mathsf{s}+1\}$ are Lagrangian bases for the spaces $\mathbb{P}_{\mathsf{p}}([-1,1])$ and $\mathbb{P}_{\mathsf{s}}([-1,1])$, respectively. The parameters p and s are denominated as transversal and axial polynomial order, respectively, and the functions $\{\phi_i\}_i$ and $\{\varphi_i\}_i$ as transversal and axial approximation functions. Formally, any scalar field w is approximated through a function w^h living in the finite-dimensional

space $\mathbb{T}_h^{\mathsf{p},\mathsf{s}}$, defined as

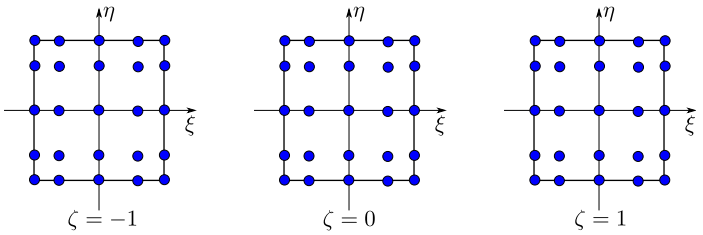
$$\mathbb{T}_h^{\mathsf{p},\mathsf{s}} = \left\{ w^h \in L^2(\Omega) : w^h \circ \chi_{\mathscr{K}} \in [\mathbb{P}_{\mathsf{p}}]^2 \times \mathbb{P}_{\mathsf{s}},\, \mathscr{K} \in \mathscr{T}_h(\Omega) \right\}. \tag{4.3}$$

Notice that the TEPEM and HiMod approximation spaces (4.3) and (2.12), respectively, share the same structure of a one-dimensional approximation enriched with a transversal modal basis. However, while the radial HiMod basis functions are coupled to the angular functions through the angular frequency, the TEPEM transversal functions are completely decoupled thanks to the Cartesian reference setting. Moreover, by the way in which the approximation functions are defined, this structure allows to independently control the model capabilities according to the direction, enriching the capacities to represent the transverse phenomena by increasing the value of the parameter p or, analogously for the axial direction, by increasing the value of the parameter s. Assuming that the local transversal dynamics on each pipe-elements can be satisfactorily approached by considering a reduced number of transversal functions, while the axial dynamics can be approximated employing low order polynomials, we choose $4 \leq \mathsf{p} \leq 10$ and, for the axial dynamic, $1 \leq \mathsf{s} \leq 2$, leading to a drastic reduction of the computational burden (we refer to Chapter 2 for more details on the problem size reduction).

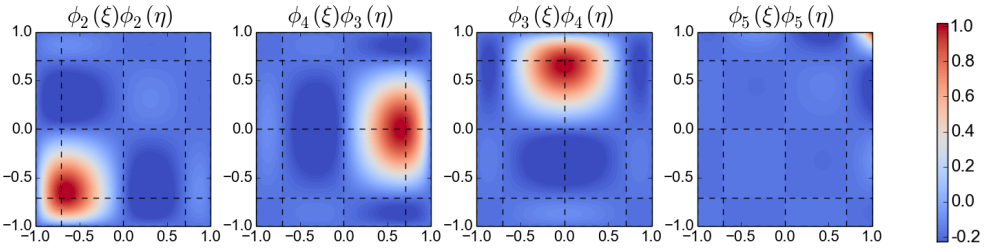
In [31], it was noticed that spurious oscillations appear in the TEPEM approximation if polynomials defined on equidistant nodes are employed as transversal functions. In fact, it is known from interpolation theory that the use of high-order Lagrange polynomials defined on a uniformly spaced set of nodes is affected by spurious oscillations, commonly referred to as *Runge phenomenon* (see [67]). Differently, the Chebyshev-Gauss-Lobatto (CGL) nodes, defined as

$$x_i = -\cos\left(\frac{i-1}{\mathsf{p}}\pi\right) \qquad i = 1, \ldots, \mathsf{p}+1, \tag{4.4}$$

for a fixed value of p, are proved to mitigate the spurious oscillations by clustering nodes near the end-points of the interval $[-1, 1]$ (see Figure 4.2). Therefore, in order to avoid the appearance of spurious oscillations in the TEPEM solution, the selected



(a) Degrees of freedom by transversal plane. A total of 75 degrees of freedom are considered.



(b) Magnitude of four transversal basis for the case $\mathsf{p} = 4$.

Figure 4.2: (a) Geometrical distribution of the degrees of freedom, on a generic pipe-element, for the combination $\mathsf{p} = 4$ and $\mathsf{s} = 2$. (b) Planar view of some functions on the transversal basis for $\mathsf{p} = 4$.

functions employed for the transversal approximation are Lagrange polynomials defined on the CGL set.

4.3 Uncertainty quantification on blood flow problems

As illustrated in Section 4.1, the traditional methods that can be found in the literature for quantifying uncertainties in large-scale problems suffer from the intensive computational cost associated with high-fidelity models, or from inaccuracy due to the reduced 1D models. In this context, DDUQ and TEPEM feature the perfect capabilities to "fill the gap" between feasibility and reliability: while, on one hand,

the DDUQ method provides an effective way to quantify the model uncertainties by promoting the independence of the subsystems, on the other hand, the TEPEM provides an effective way to solve the local problems maintaining accuracy and reduced computational burden.

Thus, in this work we embed the TEPEM solver into the DDUQ framework, in order to provide relevant statistical information about quantities of clinical interest, such as the wall shear stress, with a level of detail currently out of reach for the traditional one-dimensional models, and in a fraction of the computational time demanded by full three-dimensional approaches.

4.3.1 Blood flow problem

The blood motion is described by the incompressible Navier-Stokes equations, completed with suitable boundary conditions [76]. Let us denote by $\Omega \in \mathbb{R}^3$ an isolated region from the cardiovascular system and by Γ its boundary. The whole boundary can be subdivided into three regions: inlet (Γ_i), outlet (Γ_o) and lateral (Γ_L) boundaries. We assume inlet and outlet boundaries to be planar sections, and Γ_L to be a smooth surface (see Figure 4.3). At either boundaries Γ_i and Γ_o, Neumann or Dirichlet boundary conditions can be prescribed. In more realistic scenarios, we may expect that not all the data needed by the mathematical model is available (defective data problems). In this case, we may consider other approaches for the prescription of boundary conditions, like Lagrange multipliers [72], the distribution of a total resistance among multiple outlets [43, 194], the control approach [78, 79], or generally the geometric multiscale approach [155]. Over the lateral boundary Γ_L, no-slip boundary conditions are considered for the velocity field.

The flow dynamic is totally described through the velocity ($\mathbf{u}$) and pressure (p) fields, and the variational formulation of this problem reads as follows: find $(\mathbf{u}, p) \in$

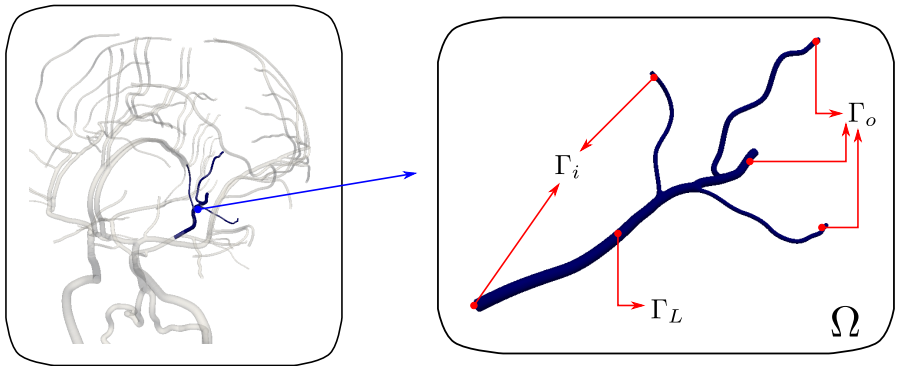


Figure 4.3: Schematic setting for the model problem. In the example, the domain Ω is an isolated region from the intracranial system.

$\mathscr{V} \times L^2(\Omega)$ such that

$$\int_\Omega \left[\rho \frac{\partial \mathbf{u}}{\partial t} \cdot \hat{\mathbf{u}} + \rho(\nabla \mathbf{u})\mathbf{u} \cdot \hat{\mathbf{u}} + 2\mu \boldsymbol{\varepsilon}(\mathbf{u}) \cdot \boldsymbol{\varepsilon}(\hat{\mathbf{u}}) - p \operatorname{div} \hat{\mathbf{u}} - \hat{p} \operatorname{div} \mathbf{u} \right] d\Omega = \int_{\Gamma_i} t_i \mathbf{n} \cdot \hat{\mathbf{u}} \, d\Gamma_i + \int_{\Gamma_o} t_o \mathbf{n} \cdot \hat{\mathbf{u}} \, d\Gamma_o \qquad \forall (\hat{\mathbf{u}}, \hat{p}) \in \mathscr{V} \times L^2(\Omega), \quad (4.5)$$

where ρ and μ are the fluid density and viscosity, respectively, $\boldsymbol{\varepsilon}(\cdot) = (\nabla(\cdot))^S$ is the symmetric part of the gradient, $\mathbf{n}$ is the outward normal unit vector, the hat sign $\hat{(\cdot)}$ denotes an admissible variation of field $(\cdot)$, and t_i and t_o are given data which stand for the magnitude of the normal component of the traction vector imposed at Γ_i and Γ_o, respectively. The space $\mathscr{V}$ is defined as

$$\mathscr{V} = \{\mathbf{u} \in \mathbf{H}^1(\Omega); \mathbf{u}|_{\Gamma_L} = \mathbf{0}, \, (\boldsymbol{\Pi}_\mathbf{n}\mathbf{u})|_{\Gamma_i} = \mathbf{0}, \, (\boldsymbol{\Pi}_\mathbf{n}\mathbf{u})|_{\Gamma_o} = \mathbf{0}\}, \quad (4.6)$$

with $\mathbf{H}^1(\Omega) = [H^1(\Omega)]^3$, and where $\boldsymbol{\Pi}_\mathbf{n} = \mathbf{I} - \mathbf{n} \otimes \mathbf{n}$ is the projection operator over the surface with normal unit vector $\mathbf{n}$. Since the domain is an isolated region of the cardiovascular system, we assume that the incoming and outgoing fluid is fully developed by enforcing the condition $(\boldsymbol{\Pi}_\mathbf{n}\mathbf{u}) = \mathbf{0}$ on the inlet and outlet boundaries.

In the TEPEM scope, the numerical solution of the fluid flow problem is obtained by discretizing the domain Ω with a pipe-type mesh $\mathscr{T}_h(\Omega)$, and by approximating

the velocity and pressure fields by the functions $\mathbf{u}^h \in \boldsymbol{\mathscr{V}}_h^{\mathsf{p}}$ and $p^h \in \mathscr{Q}_h^{\frac{\mathsf{p}}{2}}$, respectively, in the discrete spaces defined as

$$\boldsymbol{\mathscr{V}}_h^{\mathsf{p}} = \left[\mathbb{T}_h^{\mathsf{p},2}\right]^3 \cap C(\overline{\Omega}) \cap \boldsymbol{\mathscr{V}}, \qquad \mathscr{Q}_h^{\frac{\mathsf{p}}{2}} = \mathbb{T}_h^{\frac{\mathsf{p}}{2},1} \cap C^*(\overline{\Omega}), \tag{4.7}$$

where $\mathsf{p} \in \mathbb{N}$ is an even parameter which stands for the transversal order of the model, and $C^*(\overline{\Omega})$ stands for the space of continuous functions with discontinuities at the interfaces of transition elements. This combination, inspired by classical inf-sup stable spaces for FEM and spectral methods, is expected to fulfill the inf-sup condition due to the absence of spurious modes in all the simulations addressed so far. Furthermore, this combination satisfies the discrete inf-sup test proposed by Chapelle in [48], which can be understood as a numerical proof of the pair stability (see [126]). In general, the inf-sup condition for HiMod-type discretizations is an open problem. A specific proof is available only for 2D cases, with a combination of polynomials (axial)/sinusoidal (transverse) functions (see [26]).

4.3.2 Geometrical decomposition of the vasculature

In [128], a simple procedure to discretize patient-specific vasculatures in pipe-type elements was presented. This procedure can be easily adapted to perform an automatic decomposition of the whole vasculature with overlapping components, taking each bifurcation in the geometry as a basic component of this subdivision. Then, each component is discretized with a pipe-type mesh. The decomposition process, outlined in Figure 4.4, follows these steps:

(i) An overlapping by-bifurcation partitioning of the centerline is performed. Since the decomposition of a one-dimensional structure is a trivial process, this step do not introduce further complexities in the whole algorithm.

(ii) A global decomposition is performed by applying the algorithm presented in [128] to each centerline segment, to construct the vasculature components and the corresponding pipe-discretization. In this step, artificial internal boundaries

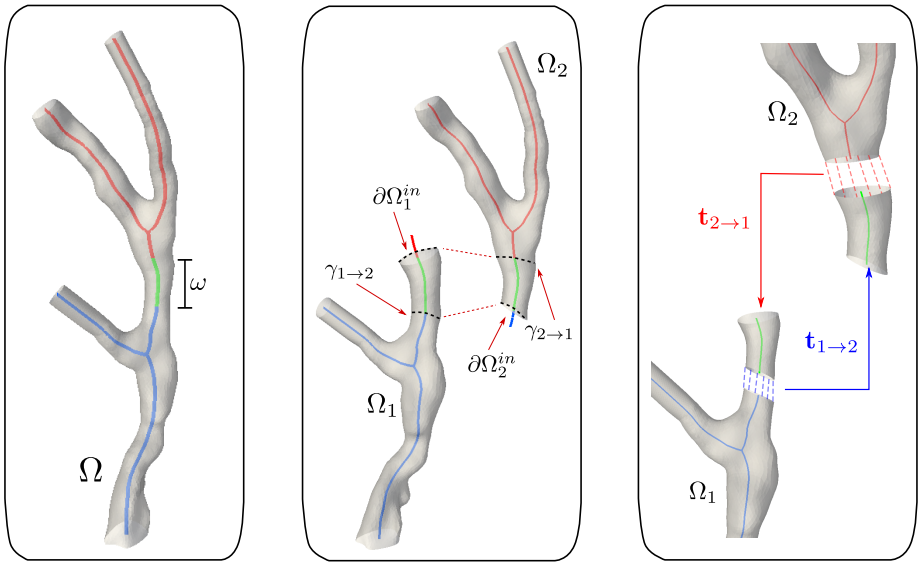


Figure 4.4: Details in the geometrical decomposition. Left: Decomposition of the centerline in two components (red and blue lines) with a common portion (green line). Center: Generation of the three-dimensional components with the creation of interfaces ($\partial\Omega^{in}$). Right: Local decomposition of each component.

are created for each component (denoted as $\partial\Omega^{in}$) and where suitable boundary conditions need to be imposed to achieve (at the convergence) the continuity of the global velocity field.

(iii) An auxiliary internal decomposition on each three-dimensional component is performed by splitting the overlapping section from the rest of the whole component, which allows exchanging information between neighboring subdomains. As we detail in the next subsection, each subdomain is connected to its neighbors by a Lagrange-multiplier approach that enforces the continuity of the velocity field on corresponding interface sections of adjacent components. For example, the continuity of the velocity field over the internal cross-section of Ω_2 (denoted as $\gamma_{2\to1}$) that corresponds to the artificial boundary $\partial\Omega_1^{in}$ of component Ω_1 is enforced through an outer traction that plays the role of a Lagrange

multiplier. It is important to highlight that this subdivision is performed at the level of the pipe discretization of each component, without the creation of any additional component for the DD algorithm.

This algorithm exploits the pipe structure of the domain of interest to reduce the complex task of decomposing a three-dimensional geometry and meshing the generated overlapping components into a simple routine of decomposing a one-dimensional centerline. This 1D-like decomposition is extremely inexpensive. As matter of fact, the geometrical decomposition and meshing process for the larger geometry addressed in this work (Section 4.4) takes less than 0.5 minutes.

4.3.3 Formulation by subdomain

Let us consider an overlapping partition $\{\Omega_1, \ldots, \Omega_n\}$ of the region of interest Ω, i.e., $\Omega = \bigcup_{k=1}^{n} \Omega_k$, with $\Omega_k \subset \Omega$ and $\Omega_i \cap \Omega_j \neq \emptyset$ for any i, $j \in \{1, \ldots, n\}$, $i \neq j$. In order to ease the notation in the definition of the blood flow problem, we introduce the forms

$$\mathscr{N}_\Omega((\mathbf{u}, p); (\hat{\mathbf{u}}, \hat{p})) = \int_\Omega \left[\rho \frac{\partial \mathbf{u}}{\partial t} \cdot \hat{\mathbf{u}} + \rho(\nabla \mathbf{u})\mathbf{u} \cdot \hat{\mathbf{u}} + 2\mu \boldsymbol{\varepsilon}(\mathbf{u}) \cdot \boldsymbol{\varepsilon}(\hat{\mathbf{u}}) - p \operatorname{div} \hat{\mathbf{u}} - \hat{p} \operatorname{div} \mathbf{u} \right] d\Omega \tag{4.8}$$

$$\mathscr{L}_\Omega((\hat{\mathbf{u}}, \hat{p})) = \int_{\partial\Omega} \mathbf{t} \cdot \hat{\mathbf{u}} \, d\Gamma, \tag{4.9}$$

then, the so-called monolithic problem can be formulated as: find $(\mathbf{u}, p) \in \boldsymbol{\mathscr{V}} \times L^2(\Omega)$ such that

$$\mathscr{N}_\Omega((\mathbf{u}, p); (\hat{\mathbf{u}}, \hat{p})) = \mathscr{L}_\Omega((\hat{\mathbf{u}}, \hat{p})) \qquad \forall (\hat{\mathbf{u}}, \hat{p}) \in \boldsymbol{\mathscr{V}} \times L^2(\Omega). \tag{4.10}$$

Using the same notation, in each component Ω_k $(k = 1, \ldots, n)$ we define the local problem as: find $(\mathbf{u}, p, \mathbf{t}) \in \boldsymbol{\mathscr{V}}_k \times L^2(\Omega_k) \times \boldsymbol{\mathscr{T}}_k$ such that

$$\begin{aligned} \mathscr{N}_{\Omega_k}((\mathbf{u}, p); (\hat{\mathbf{u}}, \hat{p})) + \sum_{i=1}^{n} \int_{\gamma_{k\to i}} \left[[\![\mathbf{u}]\!] \cdot \hat{\mathbf{t}}_{k\to i} + [\![\hat{\mathbf{u}}]\!] \cdot \mathbf{t}_{k\to i} \right] d\Gamma = \\ \mathscr{L}_{\Omega_k}((\hat{\mathbf{u}}, \hat{p})) + \sum_{i=1}^{n} \int_{\gamma_{i\to k}} \mathbf{t}_{i\to k} \cdot \hat{\mathbf{u}} \, d\Gamma \qquad \forall (\hat{\mathbf{u}}, \hat{p}) \in \boldsymbol{\mathscr{V}}_k \times L^2(\Omega_k) \times \boldsymbol{\mathscr{T}}_k, \end{aligned} \tag{4.11}$$

with $\mathbf{t}_{i\to k}$ the traction computed in the component Ω_i and imposed as a boundary condition on the component Ω_k, $[\![\cdot]\!]$ denotes the jump of the function and the spaces $\boldsymbol{\mathscr{V}}_k$ and $\boldsymbol{\mathscr{T}}_k$ defined by

$$\boldsymbol{\mathscr{V}}_k = \left\{ \mathbf{u} \in \mathbf{H}^1(\Omega_k); \mathbf{u}|_{\Gamma_L \cap \partial\Omega_k} = \mathbf{0},\ (\boldsymbol{\Pi}\mathbf{u})|_{\Gamma_i \cap \partial\Omega_k} = \mathbf{0},\ (\boldsymbol{\Pi}\mathbf{u})|_{\Gamma_o \cap \partial\Omega_k} = \mathbf{0} \right\}, \tag{4.12}$$

$$\boldsymbol{\mathscr{T}}_k = \left\{ (\mathbf{t}_{k\to 1}, \ldots, \mathbf{t}_{k\to n}) : \mathbf{t}_{k\to i} \in \mathbf{H}^{1/2}(\gamma_{k\to i}),\ i = 1, \ldots, n \right\}. \tag{4.13}$$

Notice that, imposing the traction on each artificial boundary, we are ensuring that each local problem is well-defined with the corresponding spaces for velocity and pressure. Also, the space $\boldsymbol{\mathscr{T}}_k$ is formed by the Lagrange multipliers employed to enforce the continuity in the velocity field in the component Ω_k. In a domain decomposition framework, the values of the traction from neighboring subdomains are enforced on the corresponding artificial boundaries $\partial\Omega_k$ as boundary conditions. Then, the local problems are solved (simultaneously in an additive strategy), and the traction on the internal interfaces $\gamma_{k\to i}$ is computed to be exchanged at the next iteration. The process is repeated until convergence is reached (we refer to Section 4.4 for details on the convergence criterion).

4.3.4 The DDUQ-TEPEM algorithm

Our solver is characterized by a nested structure (see Figure 3.4). At the outer level, the network solver receives global stochastic BC and model parameters as inputs, and handles the relaxation iterative process until convergence. At each iteration, the exogenous inputs specific to each component are selected, and the endogenous inputs are exchanged. With a block-Jacobi network solver, since all the components are decoupled, the endogenous interface conditions can be exchanged simultaneously; with a Gauss-Seidel or SOR network solver, the order in which they are exchanged depends on the permutation of the network components, which can be picked to maximize the level of concurrency. Then, for each component, the local stochastic solver evaluates the Q realizations of the stochastic inputs, and calls the TEPEM

deterministic solver for each instance of the inputs. Finally, the output PC coefficients are computed according to (3.10), and the termination criterion is checked by the network solver. The process is repeated until convergence. The general features of the method are reported in Chapter 3 and in [45].

The stochastic component of the problem is handled via the C++ library `UQTk` [61], developed at the Sandia National Laboratories, which has been embedded in a `MATLAB` implementation of the DDUQ network solver.

4.4 Numerical results

To study the numerical and predictive capabilities of the coupling TEPEM-DDUQ for the quantification of uncertainties in fluid flow simulations, two sets of studies are addressed: (i) weak scalability tests in phantom branched domains, and (ii) accuracy tests in more realistic vasculatures generated from the one-dimensional ADAN model [32]. The scalability properties of the DDUQ method have been previously tested in [45] on linear and non-linear scalar equations in 1D or 2D elementary domains, as reported in Chapter 3. The numerical results feature excellent scaling for all the variations of the network solvers. Here, for the first time we extend the scalability benchmarking of the DDUQ solver to *non-linear vector* problems in *3D* domains. The performance of the solver with such setting is not easily predictable, due to the complexity and intrinsic randomness of the underlying physics induced by the higher-dimensional non-linearity of the governing equations. In addition, while the stochastic solver in [45] relies on an underlying high-fidelity FE deterministic model, here we attempt accurate UQ based on deterministic ROM, to alleviate the cost of subsystem sampling. Finally, we move from academic proof of concept to real-life applications by simulating UQ in patient-specific geometries.

4.4.1 Test setting

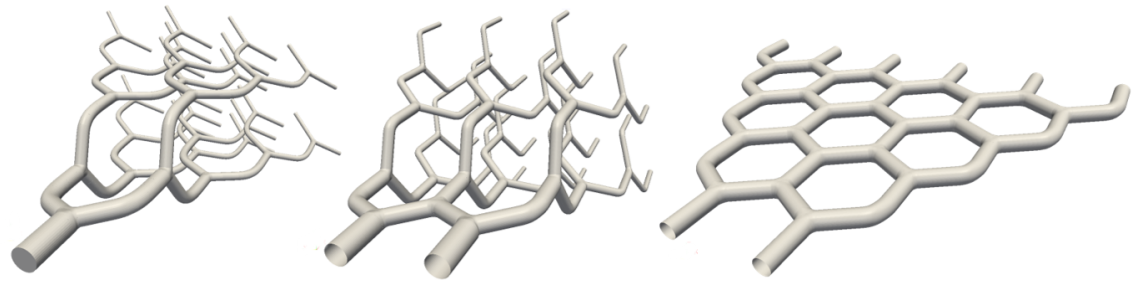
For all the cases addressed, the fluid flow is simulated in a steady state regime with deterministic fluid density $\rho = 1.04$ g/cm^3 and viscosity $\mu = 0.04$ P (although stochastic model parameters are also supported by the method). Since a stochastic solution in closed form is unavailable, statistical moments of the quantities of interest (QoI) are computed in the whole domain monolithically as a reference solution. The uncertainties are assumed to be Gaussian, with variability set to 10% of the corresponding average value, and are represented via first-order PCE. The additive version of the DDUQ network solver with Anderson Acceleration [12] is employed, with termination criterion based on relative residual, where the tolerance is set to $\varepsilon = 10^{-4}$, and the normalization factor is the norm of the first DDUQ Jacobi iterate with zero initial condition. It has been verified numerically that the level/order of the quadrature rules is accurate enough to capture the variability of the physics.

Since the focus of this preliminary work is showing the effectiveness of the strategy, rather than quantifying the potential computational saving, the TEPEM inner solver uses a direct serial algebraic LU solver on the local problem. The natural parallelism of the system matrix assembling phase will be exploited in future studies. Differently, due to the large size of the monolithic domain, the reference solution sampling is performed by TEPEM using an iterative GMRES method. Within each subsystem (in a DDUQ or monolithic setting) the sampling is sequential, although the use of shared-memory paradigms for parallel execution can be applied and will be subject of future studies.

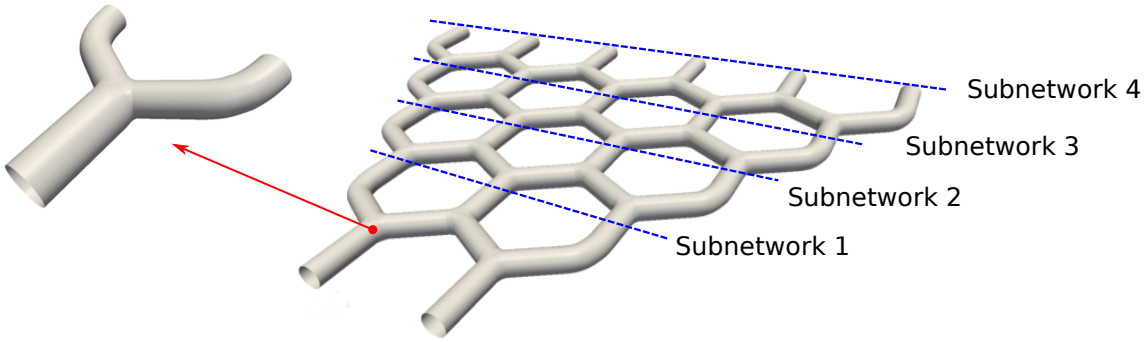
All the numerical experiments presented here have been performed on Intel(R) Core(TM) i7-2600 CPU @ 3.40GHz machines.

4.4.2 Scalability tests

Weak scalability is tested on three different network configurations. Networks 1 and 2 feature a binary structure that mimics the physiological topology of the human



(a) From left to right: Network 1, Network 2, Network 3.



(b) Basic component employed in the whole system decomposition and subnetworks.

Figure 4.5: Geometrical setting for the weak scalability test. Three phantom branched geometries are considered and decomposed employing a single bifurcation as basis component.

vascular system, with one and two global inflows, respectively. Network 3 is characterized by a denser connectivity, that allows modeling multiple internal merging flows. In each network, the basic component is a bifurcated vessel with constant radius, where each branch may feature different length or radius, depending on the network configuration. The description of these geometries is outlined in Figure 4.5. To study how the problem scales with respect to the problem size, each network is subdivided in four levels or subnetworks.

As said before, we consider stochastic boundary conditions and deterministic density and viscosity, although stochastic model parameters are also supported by the

method. In particular, since the number of outlets increases with the network size, uncertainty is localized only at the inlets, so that the dimension of the stochastic space is constant for every subnetwork. It is also worth stressing that uncertainty in boundary data for cardiovascular simulations is an important topic, in view of massive use of numerical modeling in the clinical routine, as done, for instance, by the company HeartFlow [1].

Deterministic stages are solved employing the TEPEM with transversal polynomial order $\mathsf{p} = 4$, which was verified to be enough to obtain convergence of the Navier-Stokes equation. The spatial discretization considers 50 pipe-elements on each basic component, corresponding to an axial discretization with size $h = 0.1$, and the overlap between components is composed by 6 pipe-elements.

For each network configuration, the cost of a local stochastic solve is fixed as the network size increases. The scalability of the solver is measured in terms of number of iterations to convergence and ideal parallel execution time as functions of the network size. In an ideal scenario with no constraints on computational resources, the additive network solver is embarrassingly parallelizable by defining a task as one local stochastic solve, and by creating a one-to-one task-processor mapping. This way, the ideal parallel time T_p (i.e., not including overhead) is computed as the sum of the execution time of the slowest processor at each iteration, over all iterations. Analogously, the ideal serial time T_s is the sum of the execution times of all components over all iterations. In formulae,

$$T_p = \sum_{j=1}^{n_\varepsilon} \max_{i=1,\dots,n} T_i^j, \qquad T_s = \sum_{j=1}^{n_\varepsilon} \sum_{i=1}^{n} T_i^j, \tag{4.14}$$

where n_ε is the number of iterations to convergence, n is the number of network components, and T_i^j is the execution time of component i at iteration j. Then, the ideal speedup is given by $S = T_s/T_p$.

Figure 4.6 shows that the number of iterations and the ideal parallel time grow sub-linearly with respect to the number of network components, which highlights the scalability properties of the DDUQ approach also for vector problems in 3D

geometries, with a roughly linear ideal speedup. The accuracy of the DDUQ solver is tested against a high-fidelity solution obtained with a traditional UQ approach in the corresponding monolithic domain. The quantities of interest relative error of mean value and standard deviation normalized by the high-fidelity solution is bounded as the network size increases (see Figure 4.7). As assessed in [45], this confirms that the local PCE truncation error is not propagated through the iterative process. A detailed report of the accuracy of the proposed strategy, together with the total number of components and pipe-elements needed to discretize each subnetwork, is presented in Table 4.1. Note that, as for the DDUQ scalability studies on 2D scalar problems in Chapter 3, the error on higher-order moments (i.e., in this case, the standard deviation), is higher than the corresponding error on lower-order moments. Moreover, the error on the expected value decreases with the network size. This is due to the fact that, in a domain decomposition framework, adding components entails additional exchange of information through the interfaces, leading to a more accurate approximation. This corrective effect does not happen for higher-order moments, since the error is dominated by the PCE truncation error.

4.4.3 Towards realistic geometries

Increasing the geometrical complexity on the numerical assessment, we study the accuracy of the proposed strategy on two domains, named Geometry A and Geometry B, which are constructed based on the one-dimensional ADAN model according to the radii information. These two cases represent different levels of complexity: (i) In the number of sources of uncertainties, and (ii) In the geometrical structure (and consequently on the network size), as can be seen in Figure 4.8. As before, to measure the accuracy of the coupling DDUQ-TEPEM we compare the predicted results against a reference solution obtained with traditional UQ approach in the whole domain. Besides the average value and standard deviation of the quantities of interest, we are also interested in the spatial distribution of the coefficient of variation (CV) of the QoI, also known as relative standard deviation [70]. The CV is computed

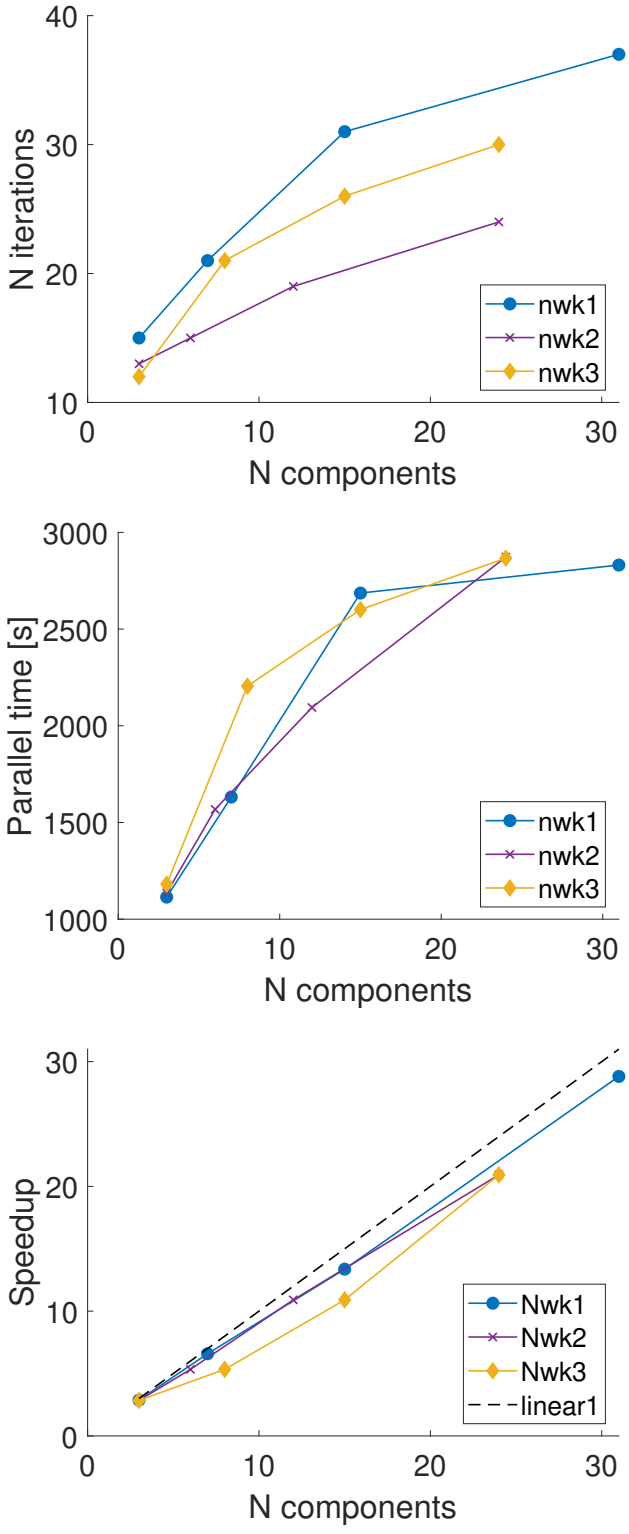


Figure 4.6: TEPEM-DDUQ weak scalability: number of iterations (top), ideal parallel time (center), and speedup (bottom) of network 1 ($\circ$), network 2 ($\times$), network 3 ($\diamond$).

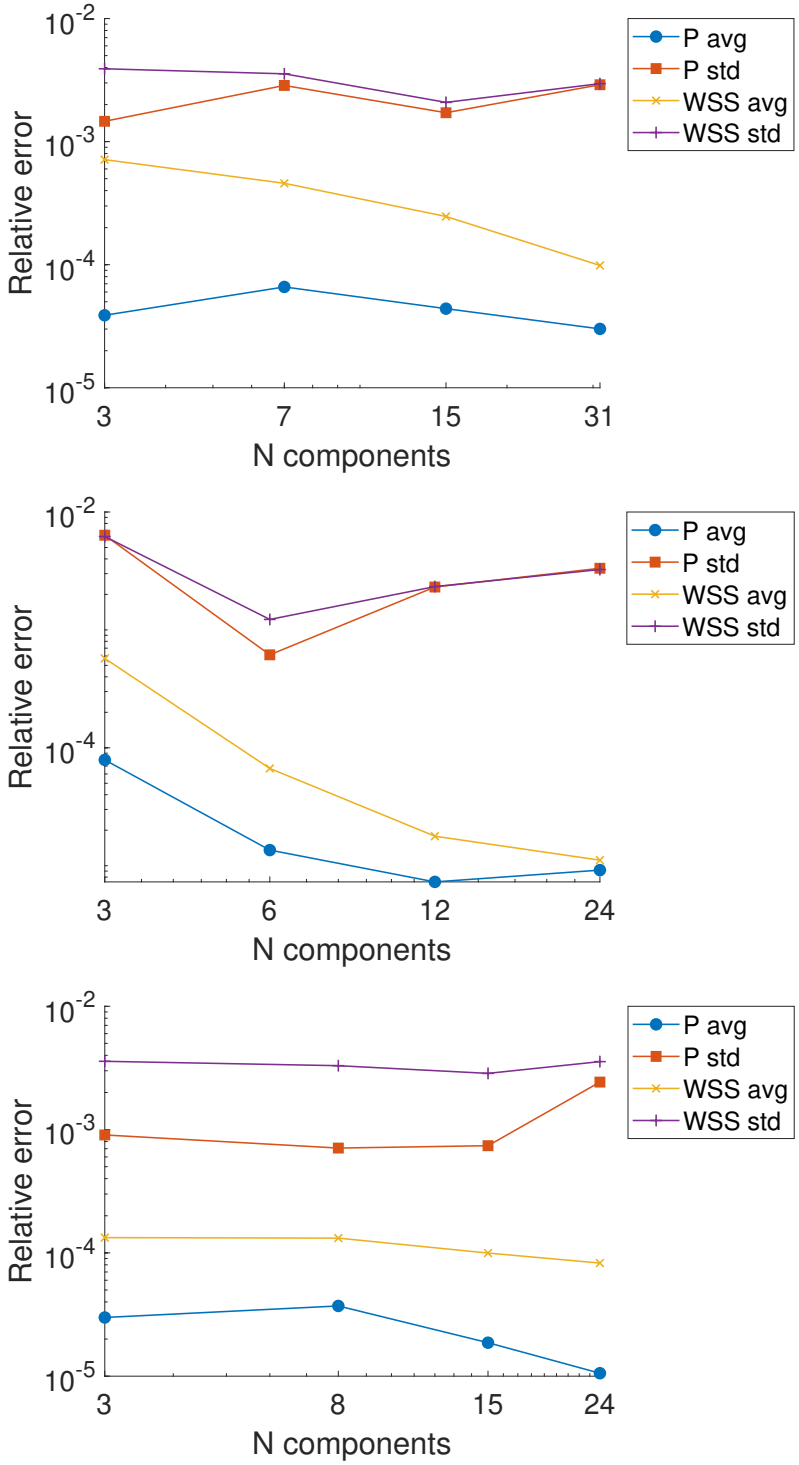


Figure 4.7: TEPEM-DDUQ weak scalability: relative error for pressure expected value (∘) and standard deviation (□), and WSS expected value (×) and standard deviation (+) for network 1 (top), network 2 (center) and network 3 (bottom).

Table 4.1: Number of components (DD) and pipe-elements (Pipe) employed to discretize each subnetwork. Also, are presented the relative error in the average (avg) and standard deviation (std) for the pressure and wall shear stress between the DDUQ-TEPEM solution and the monolithic solution for the three networks considered on the scalability test.

		Elements		Pressure error		WSS error	
		DD	Pipe	Avg	Std	Avg	Std
Network 1	Subnetwork 1	3	196	$3.88 \cdot 10^{-5}$	$1.46 \cdot 10^{-3}$	$7.14 \cdot 10^{-4}$	$3.91 \cdot 10^{-3}$
	Subnetwork 2	7	448	$6.60 \cdot 10^{-5}$	$2.86 \cdot 10^{-3}$	$4.59 \cdot 10^{-4}$	$3.56 \cdot 10^{-3}$
	Subnetwork 3	15	952	$4.39 \cdot 10^{-5}$	$1.72 \cdot 10^{-3}$	$2.47 \cdot 10^{-4}$	$2.08 \cdot 10^{-3}$
	Subnetwork 4	32	1 960	$3.01 \cdot 10^{-5}$	$2.90 \cdot 10^{-3}$	$9.86 \cdot 10^{-5}$	$2.96 \cdot 10^{-3}$
Network 2	Subnetwork 1	3	195	$7.90 \cdot 10^{-5}$	$6.35 \cdot 10^{-3}$	$5.73 \cdot 10^{-4}$	$6.20 \cdot 10^{-3}$
	Subnetwork 2	6	402	$1.36 \cdot 10^{-5}$	$6.15 \cdot 10^{-4}$	$6.68 \cdot 10^{-5}$	$1.23 \cdot 10^{-3}$
	Subnetwork 3	12	810	$7.29 \cdot 10^{-6}$	$2.31 \cdot 10^{-3}$	$1.78 \cdot 10^{-5}$	$2.33 \cdot 10^{-3}$
	Subnetwork 4	24	1 626	$9.17 \cdot 10^{-6}$	$3.33 \cdot 10^{-3}$	$1.11 \cdot 10^{-5}$	$3.25 \cdot 10^{-3}$
Network 3	Subnetwork 1	3	196	$2.99 \cdot 10^{-5}$	$9.01 \cdot 10^{-4}$	$1.33 \cdot 10^{-4}$	$3.58 \cdot 10^{-3}$
	Subnetwork 2	8	462	$3.71 \cdot 10^{-5}$	$7.08 \cdot 10^{-4}$	$1.32 \cdot 10^{-4}$	$3.30 \cdot 10^{-3}$
	Subnetwork 3	15	827	$1.87 \cdot 10^{-5}$	$7.40 \cdot 10^{-4}$	$9.97 \cdot 10^{-5}$	$2.86 \cdot 10^{-3}$
	Subnetwork 4	24	1 291	$1.06 \cdot 10^{-5}$	$2.43 \cdot 10^{-3}$	$8.27 \cdot 10^{-5}$	$3.56 \cdot 10^{-3}$

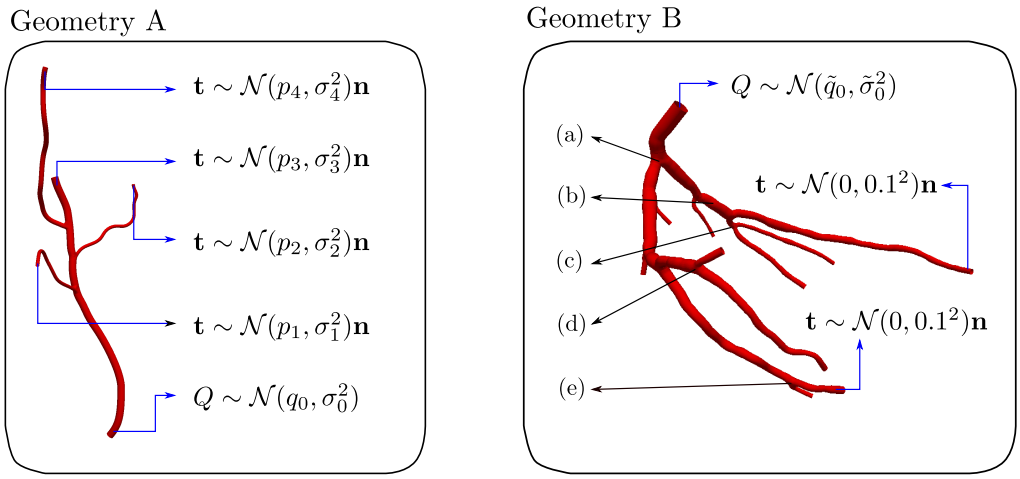


Figure 4.8: Three-dimensional geometries constructed based on the 1D ADAN model. (a) Isolated section from the intracranial system. Uncertainties are imposed on the inlet and also on the four outlets with 10% of variability on each boundary ($\sigma_0 = 0.1q_0$, $\sigma_i = 0.1p_i, i = 1, \ldots, 4$). (b) Left coronary arterial tree. Uncertainties are imposed on the inlet (10% of variability) and two outlets (variability of 0.1). The other boundaries are outlets with deterministic zero traction.

as the ratio of the standard deviation to the average value, and provides a measure of the dispersion of the output distribution that is independent from the variable's unit measure. More specifically, the higher the CV, the higher the QoI variability.

The first geometry (left panel of Figure 4.8) is a small structure composed by 7 branches, 3 bifurcations, and 5 global boundaries. This structure is decomposed in three subnetworks - one per bifurcation. Uncertainties are introduced at each global boundary, with a level of variability of 10%. Specifically, the inlet flow is equal to $q_0 = 0.25$ cm^3/s and, on the outlets, the normal component of the traction is defined by the values $p_1 = 1\,800$ dyn/cm^2, $p_2 = 1\,500$ dyn/cm^2, $p_3 = 1\,600$ dyn/cm^2 and $p_4 = 2\,000$ dyn/cm^2. These values are selected to be in concordance with physiological values. Notice that the number of uncertainty sources increases the dimension of the PCE space (dim = 5) and also the number of samples needed for

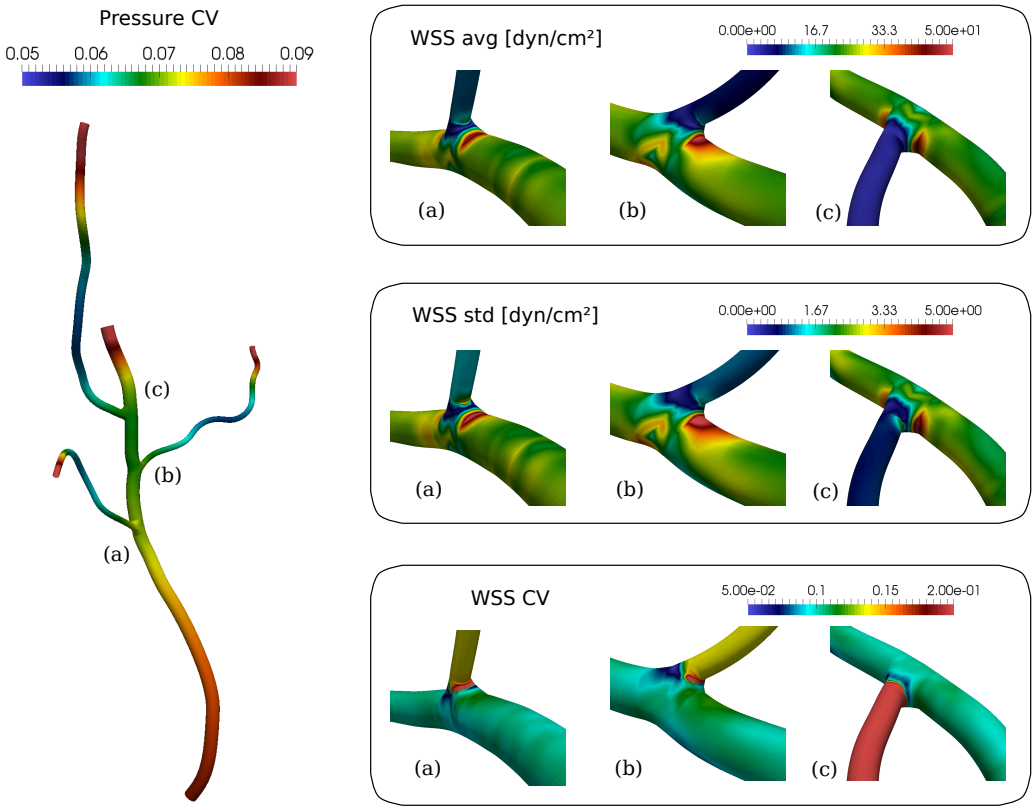


Figure 4.9: Geometry A. Left: Global view of the coefficient of variation of the pressure computed via DDUQ-TEPEM strategy. Right: Quantities of interest related to the WSS on three selected regions.

the convergence (samples = 81), in comparison with the simpler examples addressed in the scalability test part.

In Figure 4.9 the pressure CV in the whole geometry as well as the detailed view of the WSS statistical moments on the three junctions are presented. As expected, the pressure CV is higher in proximity of the sources of the uncertainty (inlets and outlets), and lower in the interior of the domain, with a maximum value of the same order of the input variability percentage (10%). The pattern of the WSS spatial distribution in proximity of the bifurcations is similar in mean value and standard deviation. However, in this case the CV is much higher in the smaller vessels, indi-

cating that a low mean WSS is typically characterized by a higher variability.

For the second geometry, a left coronary arterial tree, we consider three sources of uncertainty. On the inlet, a flow value of $\tilde{q}_0 = 0.5$ cm^3/s with variability of 10% is considered. Two outlets are selected, as seen in the right panel of Figure 4.8, to consider a stochastic traction with zero mean value and a variation of 0.1. Over the other outlets, zero traction is considered. The geometrical complexity of this case is higher in comparison with the former cases. Therefore, a higher spatial variability of the QoI is expected. On this larger structure, the proposed strategy allow us to obtain a detailed spatial description of the statistical moments and, for a better visualization, five regions were selected to present the average, standard deviation and CV of the WSS field (see Figure 4.10). As seen in the first geometry, the average and standard deviation spatial patterns are quite similar (but with different order of magnitude), and with a maximum value for the average of the same order of the imposed variability (10%). In the same figure we can see that the higher values of the CV are localized near the junctions (specially at the tip of each bifurcation), suggesting the need of further research to check a possible relation between this (or other) statistical index with regions prone to develop cardiovascular diseases.

A detailed comparison of the errors summarized in Table 4.2 confirms the accuracy of the DDUQ method on 3D non-trivial geometries, even when using ROM for sampling. In the same table are also detailed the number of component, pipe-elements employed in the discretization of the whole geometry and the parameters related to the PCE space.

4.5 DDUQ for unsteady problems

In this section we extend the DDUQ method to unsteady problems. In this case, we follow a discretize-then-split-procedure. This means that we first discretize in time, then we apply the DDQ approach at each time step. Here, we consider the traditional 1D models, that describe the fluid-structure interaction (FSI) between

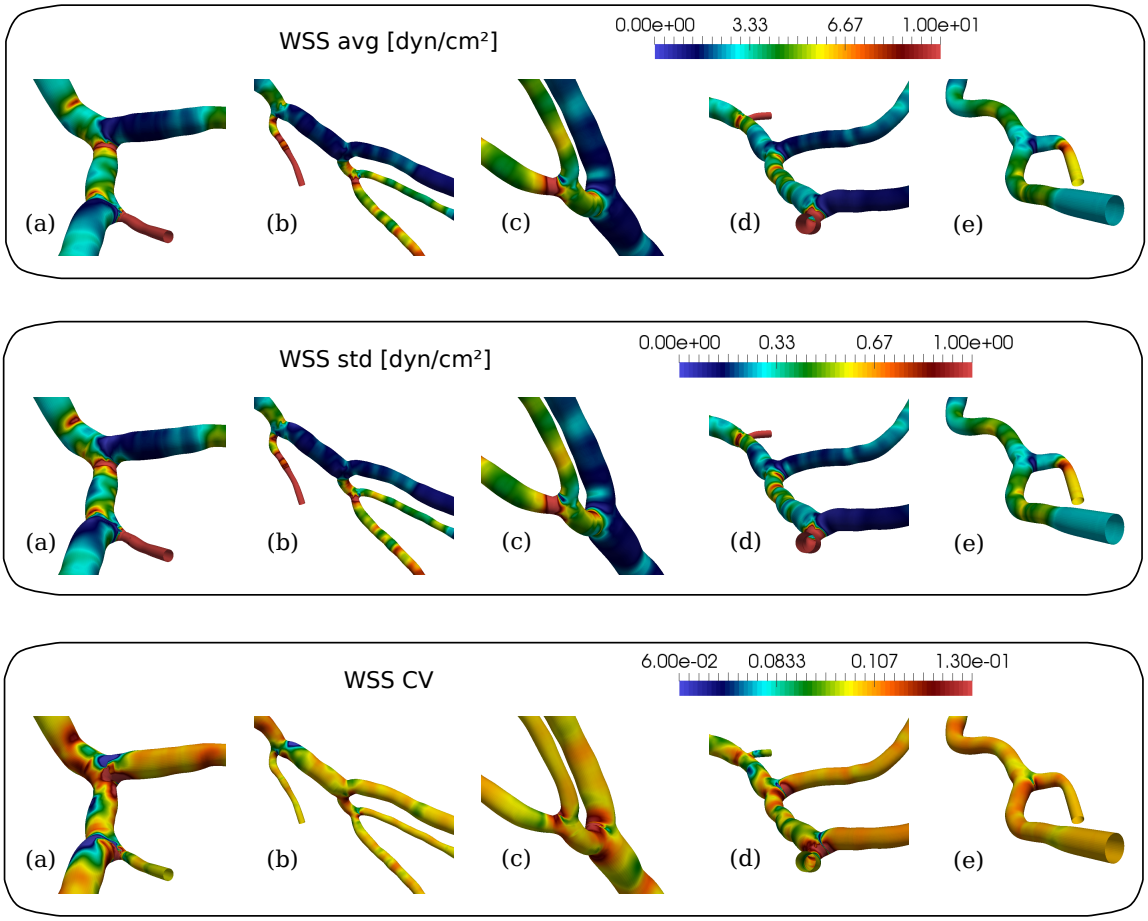


Figure 4.10: Geometry B. WSS average (top), standard deviation (std) and coefficient of variation (CV) on five selected regions.

Table 4.2: Relative error in the average and standard deviation for the pressure and wall shear stress between the DDUQ-TEPEM solution and the monolithic solution for the three networks considered on the scalability test. Also, the number of components considered on the DD partition and the number of pipe-elements for each subnetwork are reported.

Geo	Components		PCE space		Pressure error		WSS error	
	DD	Pipe	Dime	N_S	Avg	Std	Avg	Std
A	3	296	5	81	$8.84 \cdot 10^{-5}$	$5.87 \cdot 10^{-4}$	$1.26 \cdot 10^{-4}$	$1.06 \cdot 10^{-3}$
B	9	719	3	37	$2.71 \cdot 10^{-4}$	$1.74 \cdot 10^{-3}$	$2.75 \cdot 10^{-4}$	$2.15 \cdot 10^{-3}$

blood and vessel wall. In future work, we plan to model FSI with TEPEM, and to embed it in the evolutionary version of the DDUQ solver.

4.5.1 Reduced 1D models

Reduced 1D models have been widely used in the literature as an efficient tool to represent the main dynamics of a large portion of the cardiovascular system at an affordable cost (see, eg., [76, 201, 51, 125, 155, 73, 74, 77]). Although these models are extremely simplified compared to the real 3D dynamics, they are easy to interpret, which makes them "user-friendly", especially when the users are clinicians. In what follows we recall the 1D governing equations following the notation and definitions in [76], we describe how the DDUQ approach can be applied to unsteady problems, and we validate it by comparing numerical results with the literature.

Fluid-structure interaction model

Consider a compliant pipe with rectilinear axis along the x direction, spanning the interval $[x_{in}, x_{out}]$. The governing equations - conservation of mass and momentum - are derived from Reynold's transport theorem [151] by averaging, respectively, the

unit constant and the velocity on the cross-section, which yields [76]

$$\begin{cases} \dfrac{\partial A}{\partial t} + \dfrac{\partial Q}{\partial x} = 0 \\ \dfrac{\partial Q}{\partial t} + \dfrac{\partial}{\partial x}\left(\dfrac{Q^2}{A}\right) + \dfrac{A}{\rho}\dfrac{\partial p}{\partial x} + K_R\dfrac{Q}{A} = 0, \end{cases} \tag{4.15}$$

where A, Q, p denote, respectively, the cross-sectional area, the flux, and the pressure. The coefficient $K_R > 0$ represents the viscous resistance of the flow per unit length of pipe, and ρ is the blood viscosity. Equations (4.15) can be re-written in a more compact form as

$$\frac{\partial \mathbf{Q}}{\partial t} + \frac{\partial \mathbf{G}}{\partial x}(\mathbf{Q}) = \mathbf{B}(\mathbf{Q}), \tag{4.16}$$

where

$$\mathbf{Q} = \begin{bmatrix} A \\ Q \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} Q \\ \dfrac{Q^2}{A} + \displaystyle\int_{A_0}^{A} \frac{a}{\rho}\frac{\partial p}{\partial a} da \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0 \\ -K_R\dfrac{Q}{A} - \dfrac{A}{\rho}\left(\dfrac{\partial p}{\partial A_0}\dfrac{\partial A_0}{\partial x} + \dfrac{\partial p}{\partial \beta}\dfrac{\partial \beta}{\partial x}\right) \end{bmatrix}. \tag{4.17}$$

As the number of unknowns exceeds the number of equations, it is common practice to close the system by providing a constitutive relationship that describes the pressure p as a function of the vessel area A. For instance, by assuming that the wall is instantaneously in equilibrium with the pressure forces acting on it, the following physical law can be derived [76]:

$$p(x,t) = p_0 + \beta(x)\left(\sqrt{\frac{A(x)}{A_0(x)}} - 1\right), \tag{4.18}$$

where

$$\beta(x) = \frac{\sqrt{\pi} h_0 E(x)}{(1-\nu^2)\sqrt{A_0(x)}}, \tag{4.19}$$

being p_0 the external pressure, h_0 and A_0 the vessel thickness and cross-sectional area at the equilibrium state, E the Young modulus, and ν the Poisson ratio, typically set to $1/2$, since the blood vessel walls are practically incompressible. More sophisticated constitutive laws can be derived adding, for example, a model of the viscoelastic behavior of the vessel [125].

Boundary conditions

By recalling that $Q = Au$, where u is the average velocity, system (4.16) can be re-written in non-conservative form as

$$\frac{\partial \mathbf{U}}{\partial t} + \mathbf{H}\frac{\partial \mathbf{U}}{\partial x}(\mathbf{Q}) = \mathbf{f}(\mathbf{Q}), \tag{4.20}$$

where

$$\mathbf{U} = \begin{bmatrix} A \\ u \end{bmatrix}, \quad \mathbf{H} = \begin{bmatrix} u & A \\ \dfrac{c^2}{A} & u \end{bmatrix}, \quad \mathbf{f} = \begin{bmatrix} 0 \\ \dfrac{1}{\rho}\left[K_R u - \dfrac{\partial p}{\partial \beta}\dfrac{\partial \beta}{\partial x} + \dfrac{\partial p}{\partial A_0}\dfrac{\partial A_0}{\partial x}\right] \end{bmatrix}, \tag{4.21}$$

and $c^2 = \dfrac{\beta\sqrt{A}}{2\rho}$. The matrix $\mathbf{H}$ has two real eigenvalues $\lambda_{1,2} = u \pm c$ and corresponding eigenvectors $\mathbf{l}_{1,2}^T = [\pm c/A, 1]$, and characteristic variables $W_{1,2} = u \pm 4c$ (Riemann invariants). Notice that, through the definition of c, the characteristic variables W_1, W_2 can be expressed in terms of the system variables A, u (or Q), and conversely. For the typical values of the physical parameters in the cardiovascular system, the system is strictly hyperbolic and subcritical ($\lambda_1 > 0$, $\lambda_2 < 0$). Therefore, only one condition at each boundary is required.

Typically, at the proximal boundary we prescribe a given velocity (or flux) profile $u_{in}(t)$ or pressure function $p_{in}(t)$ (or area, through the constitutive law (4.18)). If both pieces of information are available, the boundary condition can be encoded in the incoming Riemann invariant W_1 as

$$W_1(x_{in}, t) = W_1\left(u_{in}(t), p_{in}(t)\right). \tag{4.22}$$

At the distal boundary, we will enforce absorbing conditions as $W_2 = 0$, or reflecting conditions $W_2 = -\alpha W_1$, with $0 < \alpha \leq 1$, that model, respectively, a pressure wave exiting the domain, or (partially) reflected.

The boundary conditions enforce one constraint at each boundary. However, when solving the global system, the values of both variables Q (or u) and A need to be

calculated. A compatibility relation to close the boundary problem can be obtained by projecting equation (4.20) onto the outgoing characteristics, which yields

$$\frac{dW_2(x(t),t)}{dt} - \mathbf{l}_2^T \mathbf{f}(\mathbf{U}) = 0 \tag{4.23}$$

at $x = x_{in}$, and analogously for W_1, $\mathbf{l}_1$ at $x = x_{out}$. Equation (4.23) can be approximated by following the characteristics backward. With a first-order discretization in time, we have

$$W_2(x_{in}, t^{n+1}) = W_2(x_{in} - \lambda_2 \Delta t, t^n) + \Delta t \mathbf{l}_2^T \mathbf{f}(\mathbf{U}) \coloneqq W_2^*. \tag{4.24}$$

The identity (4.24) is known as *extrapolation* of the characteristic variable.

Bifurcations modeling

The 1D model of a compliant blood vessel can be extended to represent bifurcations via non-overlapping domain decomposition techniques. If applied repeatedly, this procedure allows modeling the whole arterial tree.

Consider a simple bifurcation where the parent vessel Ω_1 branches out into two daughter vessels Ω_2, Ω_3 (see Figure 4.11, left). At the branching point x^*, area and velocity at the outlet of the parent vessel, and at the inlet of each daughter vessel need to be determined, for a total of six unknowns (see Figure 4.11, center). The corresponding equations are obtained by (i) enforcing the conservation of mass flux through the bifurcation point; (ii) enforcing the continuity of the total pressure $p_T = \rho u^2/2 + \beta \left(\sqrt{A} - \sqrt{A_0}\right)$ across the boundary; (iii) extrapolating the characteristics

as in (4.24). As a result, the system of equations reads as

$$\begin{cases} u_1 A_1 = u_2 A_2 + u_3 A_3 \\ \rho \frac{{u_1}^2}{2} + \beta_1(\sqrt{A_1} - \sqrt{A_{0,1}}) = \rho \frac{u_2^2}{2} + \beta_2(\sqrt{A_2} - \sqrt{A_{0,2}}) \\ \rho \frac{{u_1}^2}{2} + \beta_1(\sqrt{A_1} - \sqrt{A_{0,1}}) = \rho \frac{{u_3}^2}{2} + \beta_3(\sqrt{A_3} - \sqrt{A_{0,3}}), \\ u_1 + 4{A_1}^{1/4}\sqrt{\frac{\beta_1}{2\rho}} = W_{1,1}^* \\ u_2 - 4{A_2}^{1/4}\sqrt{\frac{\beta_2}{2\rho}} = W_{2,2}^* \\ u_3 - 4{A_3}^{1/4}\sqrt{\frac{\beta_3}{2\rho}} = W_{2,3}^* \end{cases} \tag{4.25}$$

where the unknowns are (A_i, u_i), $i = 1, 2, 3$, and the notation $W_{j,i}^*$, $j = 1, 2$, $i = 1, 2, 3$, indicates the j-th Riemann invariant in vessel Ω_i. More in general, denoting the parent vessel by Ω_1, for N_d daughter vessels we can extend conditions (4.25) to

$$\begin{cases} \sum_{i=1}^{N_d+1} A_i u_i = 0 & \\ p_{T,1} = p_{T,i+1}, & i = 1, \ldots, N_d, \\ W_{1,1} = W_{1,1}^\star, \quad W_{2,i+1} = W_{2,i+1}^\star & i = 1, \ldots, N_d. \end{cases} \tag{4.26}$$

Equations (4.25) or (4.26) form a non-linear system that can be solved, for instance, with Newton's method. Note that, in the literature, different formulations can be found, based on variations of the definitions of the quantities involved, or of the constitutive laws. We provide a synopsis of [76, 201, 51, 125, 155, 73, 74, 77] in Tables 4.3-4.4.

Numerical model

We discretize system (4.16) with a second-order Taylor-Galerkin method, which entails expanding the system unknowns with a second-order Taylor expansion in time, and replacing time derivatives with space derivatives. After some manipulations, the

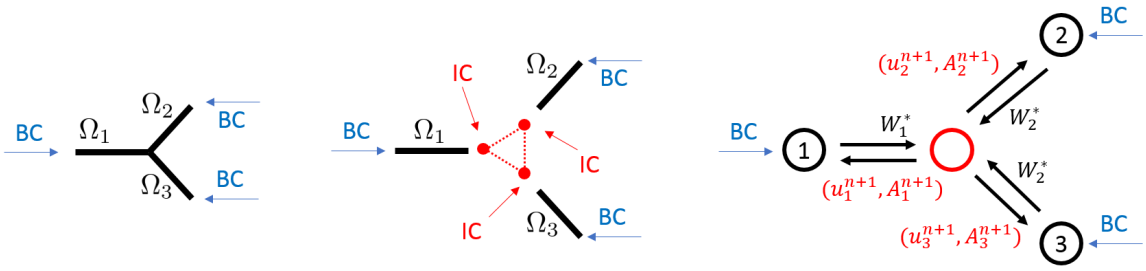


Figure 4.11: 1D bifurcation model (left), domain decomposition (center) and numerical (right) setting. BC and IC stand for boundary and interface conditions, respectively.

final formulation reads

$$\begin{aligned}\mathbf{U}^{n+1} = \mathbf{U}^n &- \Delta t \frac{\partial}{\partial x}\left[\mathbf{G}^n + \frac{\Delta t}{2}\mathbf{G}_U^n\mathbf{B}^n\right] \\ &- \frac{\Delta t^2}{2}\left[\mathbf{B}_U^n\frac{\partial \mathbf{G}^n}{\partial x} - \frac{\partial}{\partial x}\left(\mathbf{G}_U^n\frac{\partial \mathbf{G}^n}{\partial x}\right)\right] \\ &+ \Delta t\left(\mathbf{B}^n + \frac{\Delta t}{2}\mathbf{B}_U^n\mathbf{B}^n\right),\end{aligned} \tag{4.27}$$

were we used the compact notation $(\cdot)^n$ to indicate the evaluation of a certain quantity at time $t^n = n\Delta t$, and $(\cdot)_U$ for the partial derivative $\partial(\cdot)/\partial\mathbf{U}$. Then, the solution to (4.27) is approximated with a classical Finite Elements method. We subdivide the linear vessel Ω into N_e elements of length h, and herein define $\mathbf{V}_h$ as the space of piecewise linear vector functions, and $\mathbf{V}_h^0$ as the space of piecewise linear vector functions that vanish at the boundaries x_{in}, x_{out}. Then, for every $n = 1, \dots, N_t$, we need to find $\mathbf{U}_h^{n+1} \in \mathbf{V}_h$ such that, for every $\psi_h \in \mathbf{V}_h^0$,

$$\begin{aligned}\left(\mathbf{U}_h^{n+1}, \psi_h\right) = \left(\mathbf{U}_h^n, \psi_h\right) &+ \Delta t\left(\mathbf{G}_{LW}^n, \frac{\partial \psi_h}{\partial x}\right) - \frac{\Delta t^2}{2}\left(\mathbf{B}_U^n\frac{\partial \mathbf{G}^n}{\partial x}, \psi_h\right) \\ &- \frac{\Delta t^2}{2}\left(\mathbf{G}_U^n\frac{\partial \mathbf{G}^n}{\partial x}, \frac{\partial \psi_h}{\partial x}\right) + \Delta t\left(\mathbf{B}_{LW}, \psi_h\right),\end{aligned} \tag{4.28}$$

where we defined $\mathbf{G}_{LW} = \mathbf{G} + \Delta t/2\ \mathbf{G}_U\mathbf{B}$, $\mathbf{B}_{LW} = \mathbf{B} + \Delta t/2\ \mathbf{B}_U\mathbf{B}$, and $(\cdot,\cdot)$ denotes the standard $L^2(\Omega)$ inner product. It is important to remark that, in order to

guarantee the stability of the numerical scheme (4.28), the following CFL condition, that relates the space and time discretization steps, needs to be fulfilled:

$$\text{CFL} = \frac{\Delta t}{h} \max\{\lambda_1, \lambda_2\} \leq \frac{1}{\sqrt{3}}. \tag{4.29}$$

Note that, for a Finite Difference solver, the upper bound of the same condition is 1.

4.5.2 DDUQ formulation

1D models of the cardiovascular tree are characterized by several parameters, such as Young modulus, compliance, reference area or pressure for each vessel, that feature variability from patient to patient, or even within the same individual, depending on age or physical conditions (stress/rest). UQ studies have been performed in [201, 51], among others, to assess the sensitivity of the results to input uncertainty.

Since 1D models are computationally inexpensive, the UQ task is affordable even for large networks. However, as for steady problems, our goal is to enable computationally affordable UQ in large networks governed by evolutionary dynamics, while preserving the transverse components dropped by the traditional 1D models. While 1D networks can be tackled monolithically because they are inexpensive, the introduction of 3D TEPEM-FSI solvers will likely benefit from the DDUQ approach, since the independence of the subsystems facilitates a higher level of parallelization.

As a first step, we apply DDUQ to the traditional 1D models. At this stage, our focus is not on parallel performance and computational savings, but rather on benchmarking the method for evolutionary problems, as a validation step before introducing the educated reduced models.

Consider a generic 1D network with N_v vessels and N_b bifurcation points with N_d daughter vessels each. The stochastic network problem is described by the stochastic counterpart of equations (4.16)-(4.26), where the unknowns are now random variables expressed as PCEs (we refer to [201] for a rigorous formulation of the problem). Since 1D bifurcation models are intrinsically formulated via domain decomposition, DDUQ can be naturally applied by identifying the simple vessel and the bifurcation

	CVM [76]	Xiu et al. [201]	Rozza et al. [51]	Blanco et al. [125]
(A, Q)	(10.27)	-	(2)	(1)
$\mathbf{Q}$	(10.29), $(\mathbf{Q}, \mathbf{G}, \mathbf{B})$	-	(6), (U, F, S)	(3), $(\mathbf{U}, \mathbf{F}, \mathbf{S})$ (6), $(\mathbf{U}, \mathbf{H}, \mathbf{B})$
(A, u)	(10.28), $\alpha = 1$	(1)-(2)	-	-
$\mathbf{U}$	(10.31),$(\mathbf{U}, \mathbf{F}, \mathbf{S})$ (10.35),$(\mathbf{U}, \mathbf{H}, \mathbf{f})$	(5),$(\mathbf{U}, \mathbf{H}, 0)$	-	-
β	$\frac{\sqrt{\pi} h_0 E}{(1-\nu^2) A_0}$ (10.22)	$\frac{\sqrt{\pi} h_0 E}{(1-\nu^2) A_0}$ (3)	$\frac{\sqrt{\pi} h_0 E}{(1-\nu^2)\sqrt{A_0}}$ (5)	$\frac{\sqrt{\pi} h_0 E}{(1-\nu^2)\sqrt{A_0}}$ (2)
$p - p_0$	$\beta(\sqrt{A} - \sqrt{A_0})$ (10.21)	$\beta(\sqrt{A} - \sqrt{A_0})$ (3)	$\beta\left(\frac{\sqrt{A}}{\sqrt{A_0}} - 1\right) + \tilde{\psi}(A)$ (4)	$\beta\left(\frac{\sqrt{A}}{\sqrt{A_0}} - 1\right) + \tilde{\psi}(A)$ (2)
c	$A^{1/4}\sqrt{\frac{\beta}{2\rho}}$ (10.35)	$A^{1/4}\sqrt{\frac{\beta}{2\rho}}$ (4)	-	$\left(\frac{A}{A_0}\right)^{1/4}\sqrt{\frac{\beta}{2\rho}}$ (6)
$W_{1,2}$	$u \pm 4c$ (10.37)-(10.38)	$u \pm 4(c - c_0)$ (7)	-	-
A	$\left(\frac{W_1 - W_2}{4}\right)^4 \left(\frac{\rho}{2\beta}\right)^2$ (10.39)	-	-	-
u	$\frac{W_2 + W_1}{2}$ (10.39)	-	-	-
$G(2)$	$\alpha\frac{Q^2}{A} + \int_{A_0}^{A} \frac{a}{\varphi}\frac{\partial \rho}{\partial a} da$ (10.30), $\alpha\frac{Q^2}{A} + \frac{\beta}{3\rho}A^{3/2}$ (old)	-	-	$\alpha\frac{Q^2}{A} + \int_{A_0}^{A} \frac{a}{\rho}\frac{\partial p}{\partial a} da$ (3) $\alpha\frac{Q^2}{A} + \frac{\beta A_0^3}{3\rho}\left(\left(\frac{A}{A_0}\right)^{3/2} - 1\right)$ (3)
W_2^*	$W_2(x_1 - \lambda_2 \Delta t, t^n)$ $+\Delta t \mathbf{l}_2^T \mathbf{B}(\mathbf{Q})$ (10.46)	-	-	

Table 4.3: Synopsis of Euler's equations formulation (I).

	Quarteroni et al. [155]	Formaggia et al. [73]	Lamponi et al. [74]	Formaggia Veneziani [77]
(A,Q)	-	(2)-(3)	(1)-(2)	(1.8)
$\mathbf{Q}$	(9), $(\mathbf{U},\mathbf{F},\mathbf{S})$ (7), $(\mathbf{U},\mathbf{H},\mathbf{B})$	(13), $(\mathbf{U},\mathbf{F},\mathbf{B})$	-	(1.16), $(\mathbf{U},\mathbf{F},\mathbf{B})$ (1.14), $(\mathbf{U},\mathbf{H},\mathbf{S})$
(A,u)	-	-	-	-
$\mathbf{U}$	-	-	-	-
β	$\frac{\sqrt{\pi}h_0E}{(1-\nu^2)}$ (11)	$\sqrt{\pi}h_0E$ $(\nu=0)$ (10)	$\sqrt{\pi}h_0E$ $(\nu=0?)$ (4)	$\frac{\sqrt{\pi}h_0E}{(1-\nu^2)}$ (1.12)
$p-p_0$	$\frac{\beta}{A_0}(\sqrt{A}-\sqrt{A_0})$ (11)	$\frac{\beta}{A_0}(\sqrt{A}-\sqrt{A_0})$ (9)	$\frac{\beta}{A_0}(\sqrt{A}-\sqrt{A_0})$ (4)	$\frac{\beta}{A_0}(\sqrt{A}-\sqrt{A_0})$ (1.12)
c	$A^{1/4}\sqrt{\frac{\beta}{2\rho A_0}}$ (11)	$A^{1/4}\sqrt{\frac{\beta}{2\rho A_0}}$ (12)	$A^{1/4}\sqrt{\frac{\beta}{2\rho A_0}}$ (5)	$A^{1/4}\sqrt{\frac{\beta}{2\rho A_0}}$ (1.17)
$W_{1,2}$	$u\pm 4c$ (32)	$u\pm 4c$ (18)	$u\pm 4(c-c_0)$ (7)	$u\pm 4(c-c_0)$ (1.32)
A	-	$\left(\frac{W_1-W_2}{8}\right)^4\left(\frac{2\rho A_0}{\beta}\right)^2$ (19)	$\left(\frac{W_1-W_2}{8}+c_0\right)^4\left(\frac{2\rho A_0}{\beta}\right)^2$ (8)	$\left(\frac{W_1-W_2}{8}+c_0\right)^4\left(\frac{2\rho A_0}{\beta}\right)^2$ (1.34)
u	-	$\frac{W_2+W_1}{2}$ (19)	$\frac{W_2+W_1}{2}$ (8)	$\frac{W_2+W_1}{2}$ (1.34)
$G(2)$	$\alpha\frac{Q^2}{A}+\int_{A_0}^{A}\frac{a}{\rho}\frac{\partial p}{\partial a}da$ (9)	$\alpha\frac{Q^2}{A}+\int_{0}^{A}\frac{a}{\rho}\frac{\partial p}{\partial a}da$ (14), $\alpha\frac{Q^2}{A}+\frac{\beta}{3\rho A_0}A^{3/2}$ (14)	$\alpha\frac{Q^2}{A}+\int_{A_0}^{A}\frac{a}{\rho}\frac{\partial p}{\partial a}da$	$\alpha\frac{Q^2}{A}+\int_{A_0}^{A}\frac{a}{\rho}\frac{\partial p}{\partial a}da$ (1.16), $\alpha\frac{Q^2}{A}+\frac{\beta}{3\rho A_0}A^{3/2}$ (1.17)
W_2^*	$W_2(x_1-\lambda_2\Delta t,t^n)$ (p.219)	$W_2(x_1-\lambda_2\Delta t,t^n)$ (p.260)	$W_2(x_1-\lambda_2\Delta t,t^n)$ (20)	$W_2(x_1-\lambda_2\Delta t,t^n)$ (1.50)

Table 4.4: Synopsis of Euler's equations formulation (II).

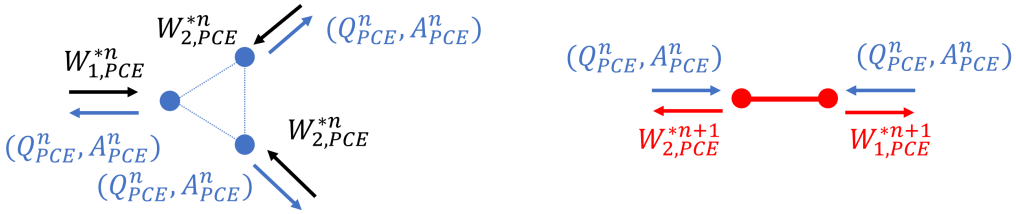


Figure 4.12: Unsteady 1D DDUQ network problem: component representation with inputs/outputs for a bifurcation point with $N_d = 2$ daughter vessels (left) and a simple vessel (right). Note that all the quantities exchanged are the PC coefficients of the unknowns (area A_{PCE}, flow Q_{PCE}, and extrapolated Riemann invariants $W^*_{1,PCE}$, $W^*_{2,PCE}$) at a fixed time instant.

point as two different network components with two and $N_d + 1$ input/output ports, respectively (see Figure 4.12). Since we are dealing with a time-advancing scheme dictated by (4.28)-(4.26) (where now the quantities involved are the PCEs of the quantities involved), at each time step the bifurcation points receive the PC coefficients of the extrapolated Riemann invariants from the previous step, and return the PC coefficients of area and flow at the current time step (see Figure 4.12, left), which are received by the vessel components to compute the current stochastic extrapolated Riemann invariants that will be used as inputs by the bifurcation points at the next step (see Figure 4.12, right). As a result, the time-advancing network solver works as a sort of red-black tree, where bifurcations and vessels correspond to red and black nodes, respectively. The numerical procedure described here is outlined in Algorithm 6.

4.5.3 Numerical results

In order to validate the DDUQ method on unsteady problems, we closely follow the numerical tests performed in [201] on a simple bifurcation and in a 37-vessel network.

Algorithm 6: DDUQ in 1D unsteady networks.

for $n = 1, \dots, N_t$ **do** // Time loop
  Update exogenous inputs and forcing term
  for $n_b = 1, \dots, N_b$ **do** // Loop on bifurcations
    Select input $W^{*n-1}_{1,PCE}$, $W^{*n-1}_{2,PCE}$
    Evaluate N_s samples
    for $n_s = 1, \dots, N_s$ **do**
      Solve problem (4.26) with n_s-th input
    end
    Evaluate output PCE A^n_{PCE}, Q^n_{PCE}
  end
  for $n_v = 1, \dots, N_v$ **do** // Loop on vessels (segments)
    Select input A^n_{PCE}, Q^n_{PCE}
    Evaluate N_s samples
    for $n_s = 1, \dots, N_s$ **do**
      Solve problem (4.28) with n_s-th input
    end
    Evaluate output PCE $W^{*n}_{1,PCE}$, $W^{*n}_{2,PCE}$
  end
end

Simple bifurcation

We consider a simple bifurcation model where parent and daughter vessels are 20cm long, with diameter of 1cm and $1/\sqrt{6}$cm, respectively. The sources of uncertainty are the parameters β_i, $i \in \{1, 2, 3\}$, encoding the mechanical and geometrical properties of the vessel. We assume the random variables to be independent and uniformly distributed in $(-1, 1)$, with first-order PCE

$$\beta_i(\xi) = \langle \beta_i \rangle \left(1 + \sigma_i \xi_i\right), \tag{4.30}$$

where σ_i quantifies the variability of the parameter, and $\langle\cdot\rangle$ denotes the expected value of the random variable. For this specific case, we set $\langle\beta_1\rangle = 32,497\text{g}/(\text{s}^2\ \text{cm}^2)$ and $\langle\beta_{2,3}\rangle = 79,602\text{g}/(\text{s}^2\ \text{cm}^2)$ for the parent and daughter vessels, respectively, and $\sigma_i = 10\%$ for all. The output quantity of interest is the pressure as a function of time, at the inlet and midpoint of the parent vessel, at the bifurcation point, and at the midpoint and outlet of a daughter vessel. Each point is labeled as A, B, C, D, E, in corresponding order. We compare the zero- and first-order moments (mean and standard deviation) of the DDUQ solution with the literature, as well as the sensitivity with respect to the uncertain input parameters, defined in [201] as

$$S_i = \langle\beta_i\rangle \left\langle \frac{\partial p(\boldsymbol{\xi})}{\partial \beta_i(\boldsymbol{\xi})} \right\rangle = \frac{1}{\sigma} \int \frac{\partial p(\boldsymbol{\xi})}{\partial \xi_i} \pi(\boldsymbol{\xi}) d\boldsymbol{\xi}, \tag{4.31}$$

where $\pi(\boldsymbol{\xi})$ is a uniform probability density function.

At the inlet we prescribe the following velocity and area profiles:

$$U_{in}(0^-, t) = U_0 e^{-C(t-t_0)^2}, \qquad A_{in}(0^-, t) = \frac{\pi}{4}, \tag{4.32}$$

with $U_0 = 1\text{cm/s}$, $C = 5000\text{s}^{-2}$, $t_0 = 0.05\text{s}$, through the Riemann invariants, and and absorbing condition on the incoming characteristic. In other words, we set

$$\begin{cases} W_1 = W_1(A_{in}, U_{in}), \quad W_2 = 0 & t < L/c_0 \\ W_1 = 0, \quad W_2 = W_2^* & t \geq L/c_0, \end{cases} \tag{4.33}$$

where $t = L/c_0$ is the time instant when the wave hits the bifurcation point according to [168], and from (4.33) we obtain $A_{in} = A(W_1, W_2)$, $U_{in} = U(W_1, W_2)$. The outflow boundary conditions are set to absorbing or reflecting. Each case is discussed in the following paragraphs.

Absorbing conditions. Absorbing outflow conditions are modeled as $W_1 = W_1^*$, $W_2 = 0$. Before comparing the traditional UQ method with DDUQ, we calibrate the space-time discretization of the underlying deterministic solver by performing a sensitivity study of the stochastic solution to the CFL number (4.29). We tune the

CFL number by fixing the mesh size to $h = 0.25$cm, and by varying the time step Δt to obtain a CFL value that is 10%, 40%, or 90% of the upper bound ($M = 1/\sqrt{3}$). The DDUQ pressure mean value and standard deviation obtained with the three different discretizations as functions of time at the five probe locations (A-E) are shown in Figures 4.13-4.14. While for the expected value the peaks are roughly independent of the CFL number at all locations (see Figure 4.13), the standard deviation is more sensitive to the underlying discretization, with a peak slightly decreasing in value as the CFL number approaches the upper bound (see Figure 4.14). Since the variation is not significant, we perform the remaining simulations with a CFL in the mid range (40% of the upper bound), corresponding to $\Delta t = 5 \cdot 10^{-4}$, and simulate a period of $T = 0.5$s.

Figures 4.15-4.16 show a comparison of the baseline solution from [201] and the DDUQ solution. Although a quantitative comparison is impossible due to lack of numerical data, a visual comparison of the temporal profiles shows good agreement between the two solutions, both for average value and standard deviation. The peaks of the standard deviation happen to be lower in value, and the wave shape does not match perfectly, with lower and sharper dips between peaks (see Figure 4.16). This could be due to the different time steps, mesh sizes, and discretization methods employed here (Taylor-Galerkin) and in [201] (Discontinuous Galerkin), that feature different dispersion/diffusion properties. However, the time instants when the peaks happen are precisely captured.

The temporal pattern of the sensitivity of the solution with respect to the uncertain input parameters is in good agreement with the literature. However, there is a significant mismatch in the values, likely due to a scaling factor not reported in [201]. Therefore, for a more quantitative comparison, we define the quantity

$$R_i = \left| \frac{\max_t S_i(t)}{\max_t S_i^{XS}(t)} - \frac{\min_t S_i(t)}{\min_t S_i^{XS}(t)} \right|, \tag{4.34}$$

as a normalized measure of the mismatch between peaks, where S_i denotes the DDUQ sensitivity and S_i^{XS} is the corresponding value from [201]. Assuming that $S_i = \alpha S_i^{XS}$ for some scaling factor α, we expect $R_i \simeq 0$. In fact, from Table 4.5 the error

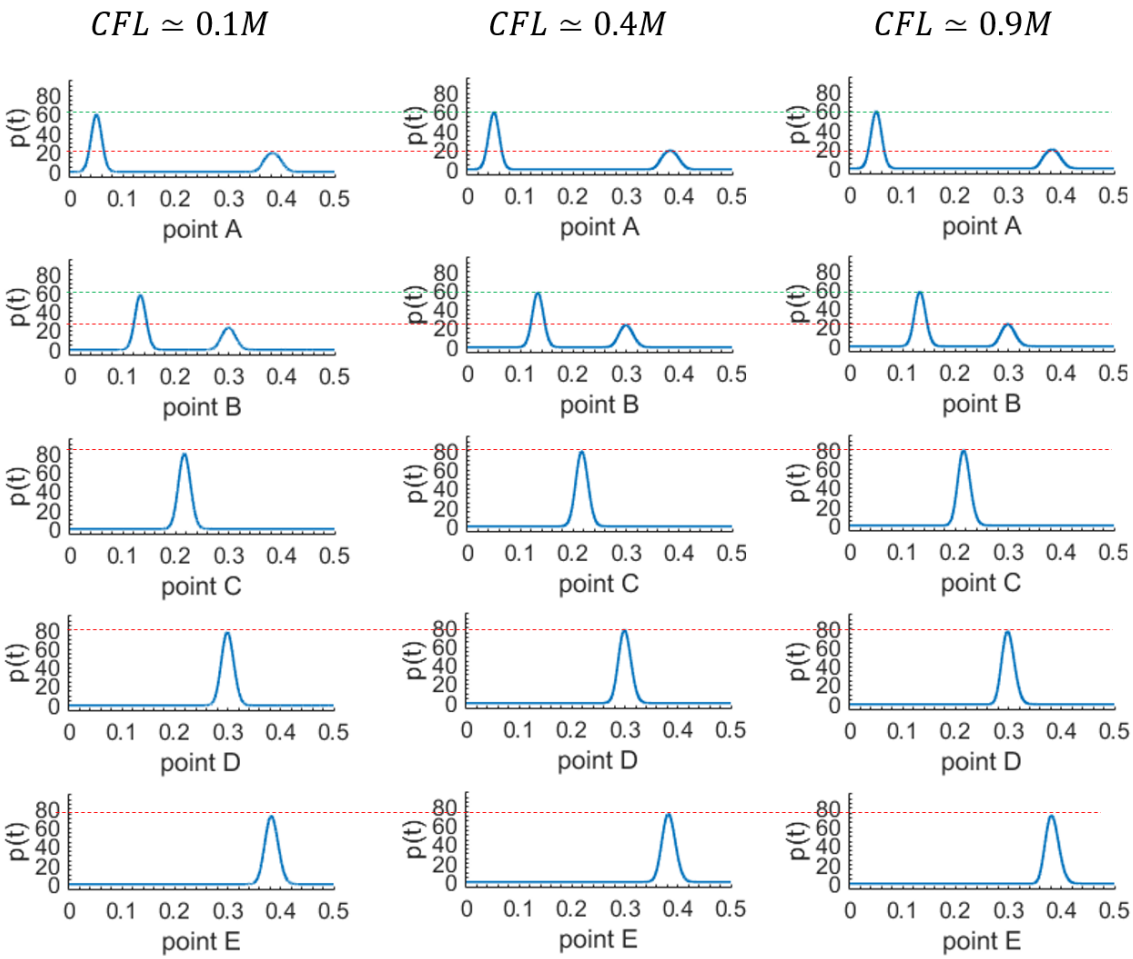


Figure 4.13: Propagating pressure wave in a simple bifurcation with absorbing outflow boundary conditions: mean of DDUQ solution at inflow (first row), parent vessel mid-point (second row), bifurcation point (third row), daughter vessel mid-point (fourth row), and outflow (fifth row) for a CFL number equal to 10% (left), 40% (center), or 90% (right) of the upper bound ($M = 1/\sqrt{3}$).

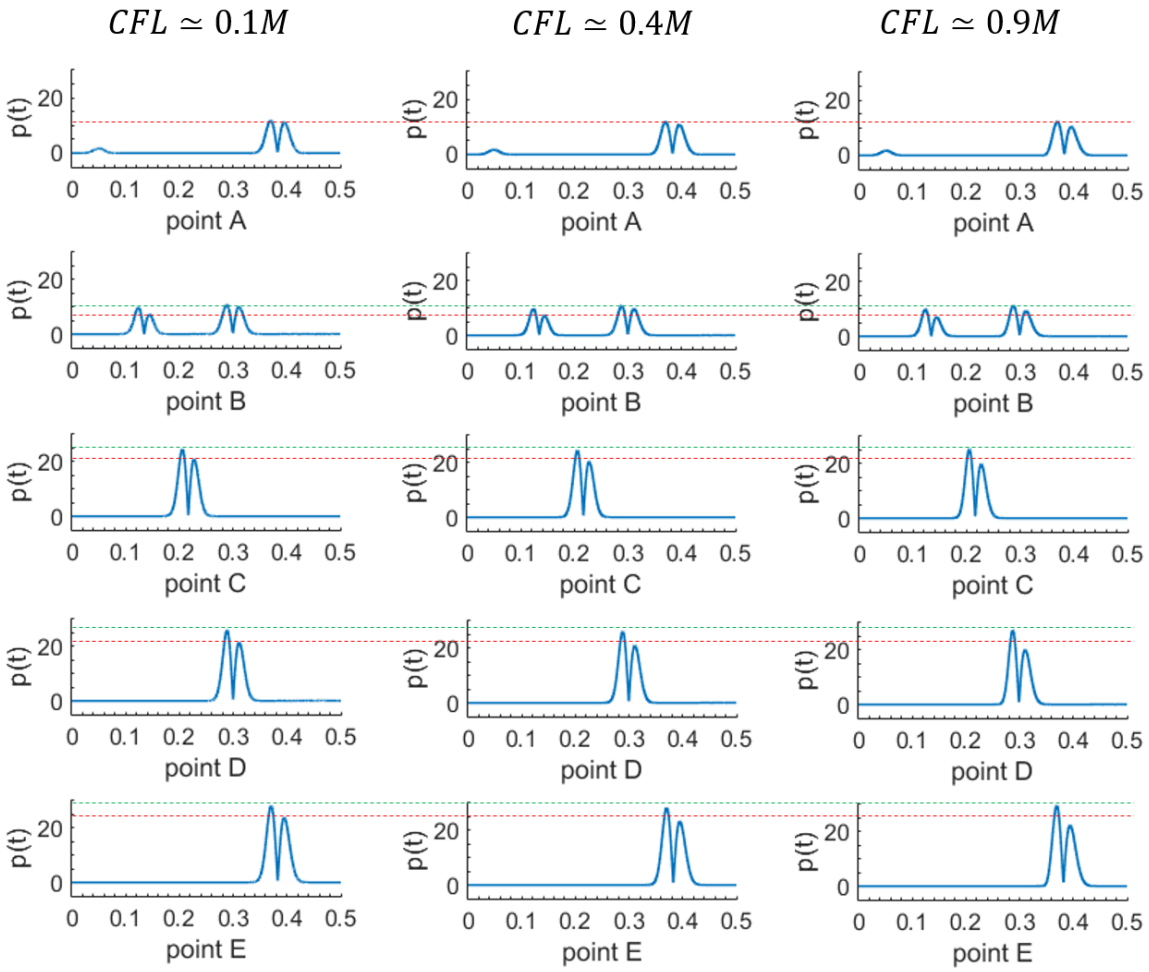


Figure 4.14: Propagating pressure wave in a simple bifurcation with absorbing outflow boundary conditions: standard deviation of DDUQ solution at inflow (first row), parent vessel mid-point (second row), bifurcation point (third row), daughter vessel mid-point (fourth row), and outflow (fifth row) for a CFL number equal to 10% (left), 40% (center), or 90% (right) of the upper bound ($M = 1/\sqrt{3}$).

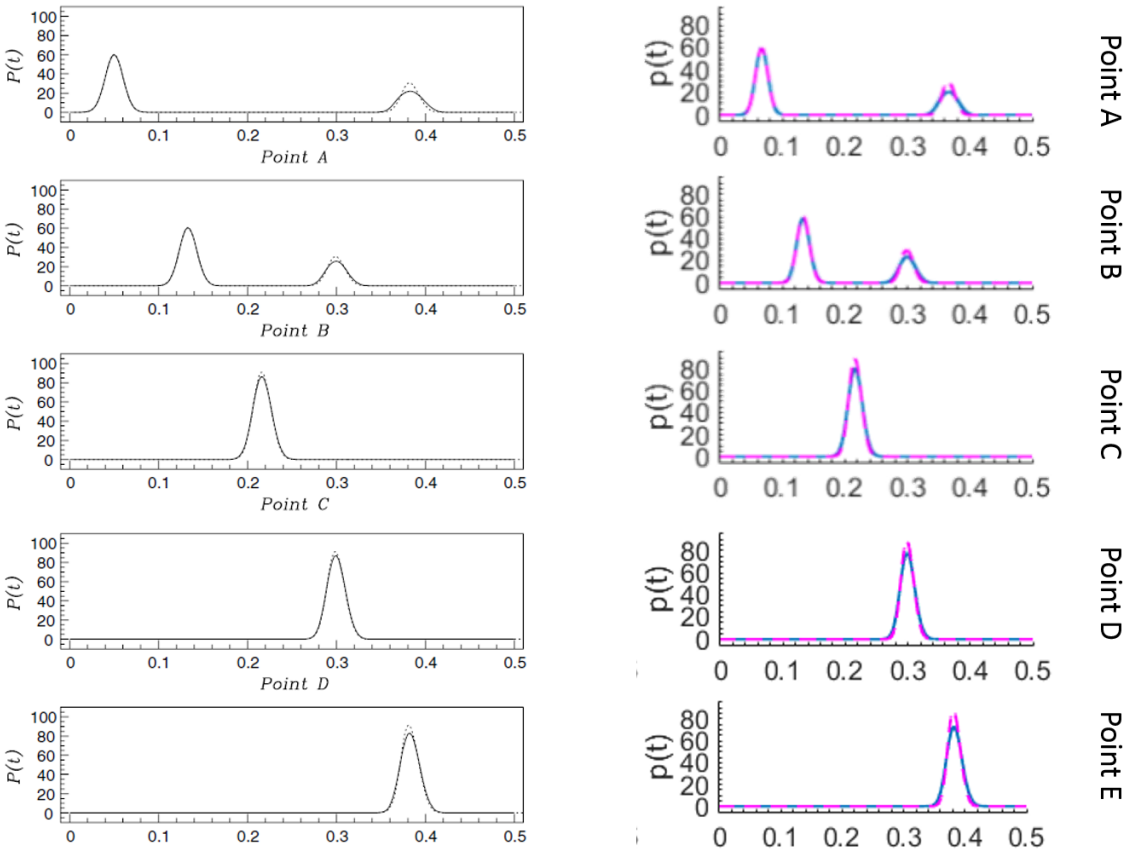


Figure 4.15: Propagating pressure wave in a simple bifurcation with absorbing outflow boundary conditions: mean of baseline solution from [201] (left) and DDUQ solution (right) at inflow (first row), parent vessel mid-point (second row), bifurcation point (third row), daughter vessel mid-point (fourth row), and outflow (fifth row). Deterministic solution is shown in dashed line for comparison.

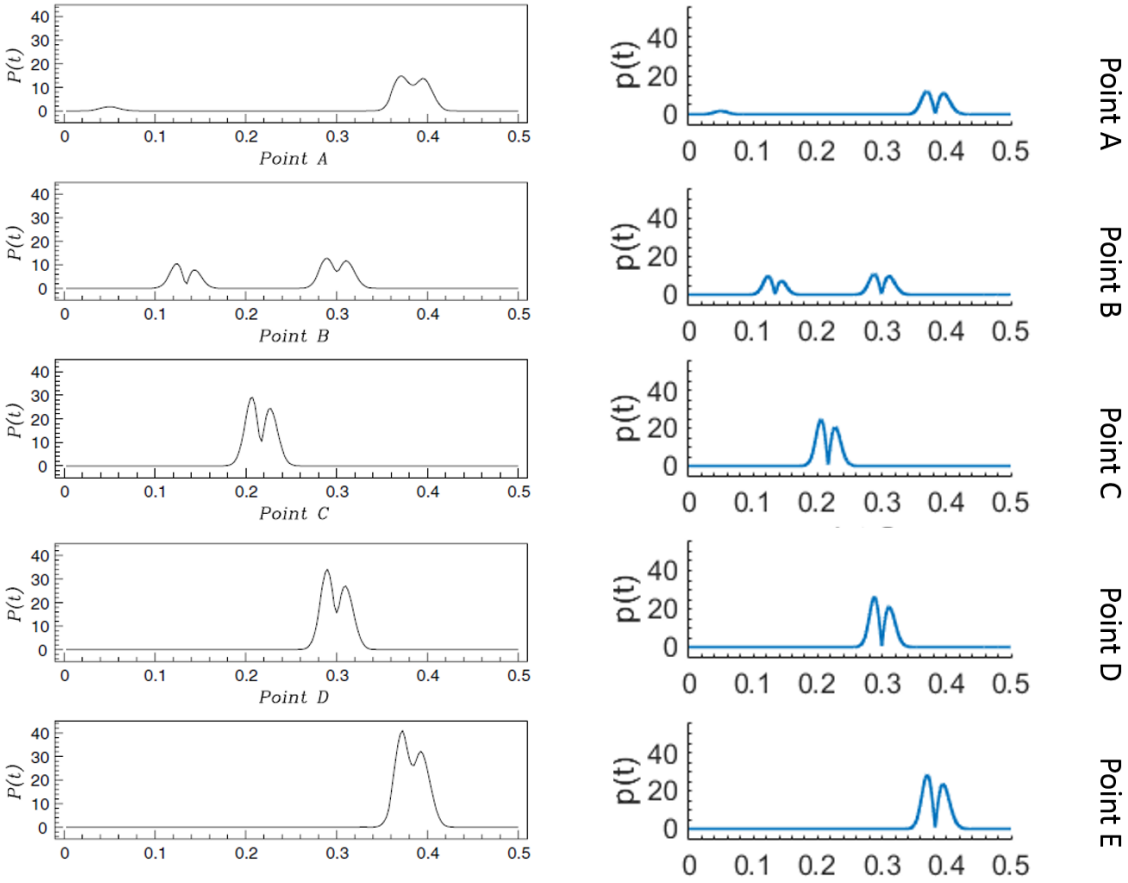


Figure 4.16: Propagating pressure wave in a simple bifurcation with absorbing outflow boundary conditions: standard deviation of baseline solution from [201] (left) and DDUQ solution (right) at inflow (first row), parent vessel mid-point (second row), bifurcation point (third row), daughter vessel mid-point (fourth row), and outflow (fifth row).

	Point B	Point C	Point D	Point E
R_1	0.0952	0.1681	0.0349	0.09997
R_2	N/A	N/A	0.1492	0.0030

Table 4.5: Propagating pressure wave in a simple bifurcation with absorbing outflow boundary conditions: mismatch between peaks of DDUQ sensitivities and the results reported in [201].

ranges between $0.3\% - 17\%$, which is acceptable, considering that the values from the literature were extracted by visual inspection from the figures provided. Moreover, notice that, for this case, not all data is available, since the description of some results in [201] does not match the corresponding figures, and is therefore impossible to extract the data needed for comparison purposes.

Reflecting conditions. Reflecting outflow conditions are modeled as $W_1 = W_1^*$, $W_2 = -\alpha W_1$, with $\alpha = 50\%$. We simulate a period of $T = 1$s, with the same space-time discretization as the previous test. The same considerations as the absorbing case hold for the pressure average and standard deviation: the temporal features are perfectly captured, although the standard deviation profile features deeper and sharper dips, probably due to the choice of the discretization scheme (see Figures 4.17-4.18). A full comparison on the sensitivities in Figure 4.19 shows good agreement in the pattern, with peaks mismatch of $0.08\% - 25\%$, which is acceptable, considering the potential inaccuracy of the reference values, extracted visually from the literature (see Table 4.6).

37-vessel network

We now consider the more realistic case of a network with 37 vessels and 16 bifurcations, built in [129] from an *in vitro* model. At the inlet, the flow profile obtained from experimental data in [129] is prescribed periodically. The outflow boundary

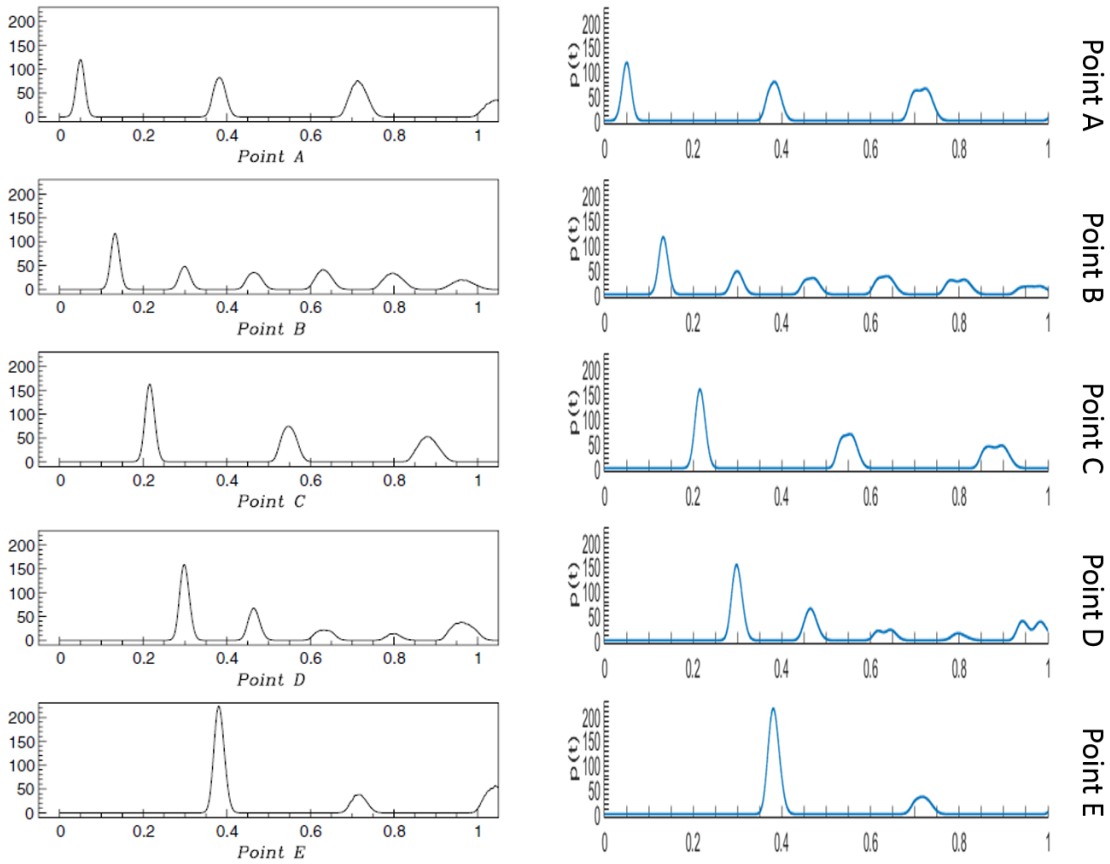


Figure 4.17: Propagating pressure wave in a simple bifurcation with reflecting outflow boundary conditions: mean of baseline solution from [201] (left) and DDUQ solution (right) at inflow (first row), parent vessel mid-point (second row), bifurcation point (third row), daughter vessel mid-point (fourth row), and outflow (fifth row).

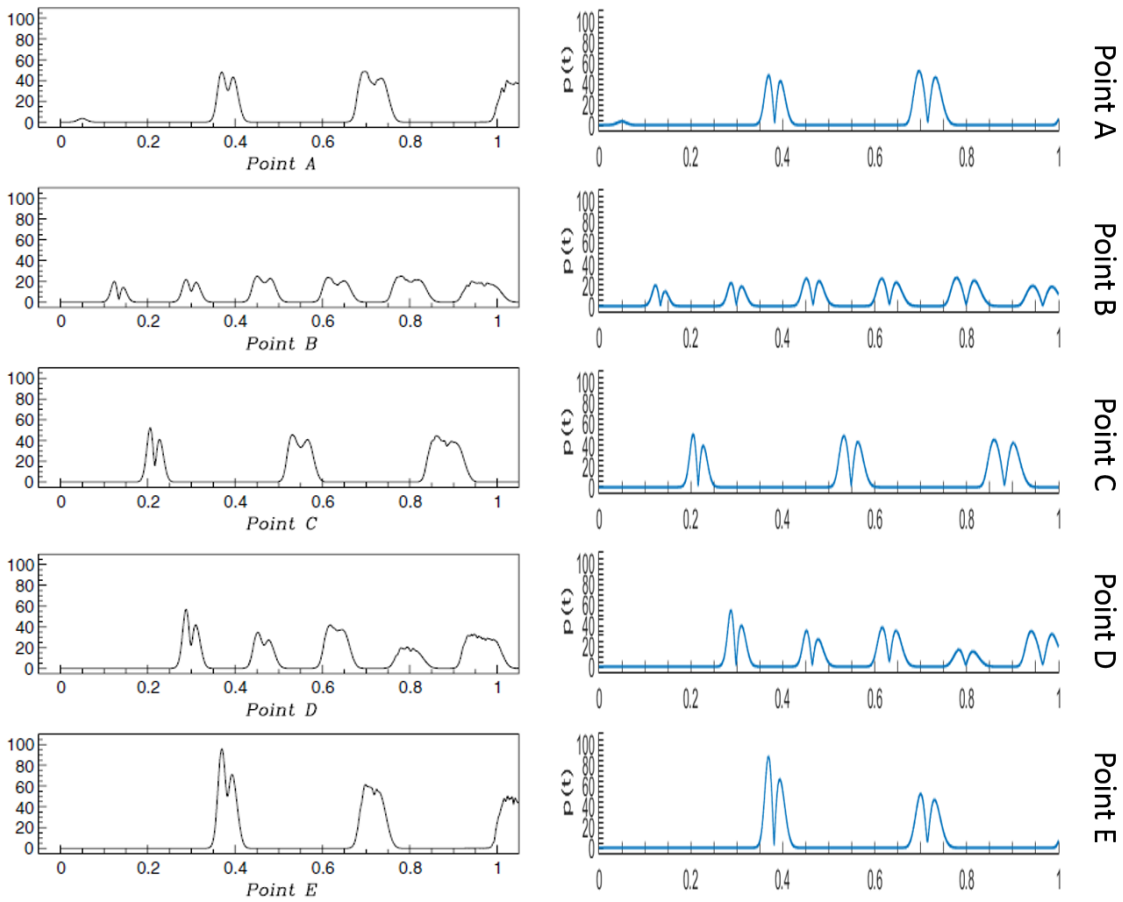


Figure 4.18: Propagating pressure wave in a simple bifurcation with reflecting outflow boundary conditions: standard deviation of baseline solution from [201] (left) and DDUQ solution (right) at inflow (first row), parent vessel mid-point (second row), bifurcation point (third row), daughter vessel mid-point (fourth row), and outflow (fifth row).

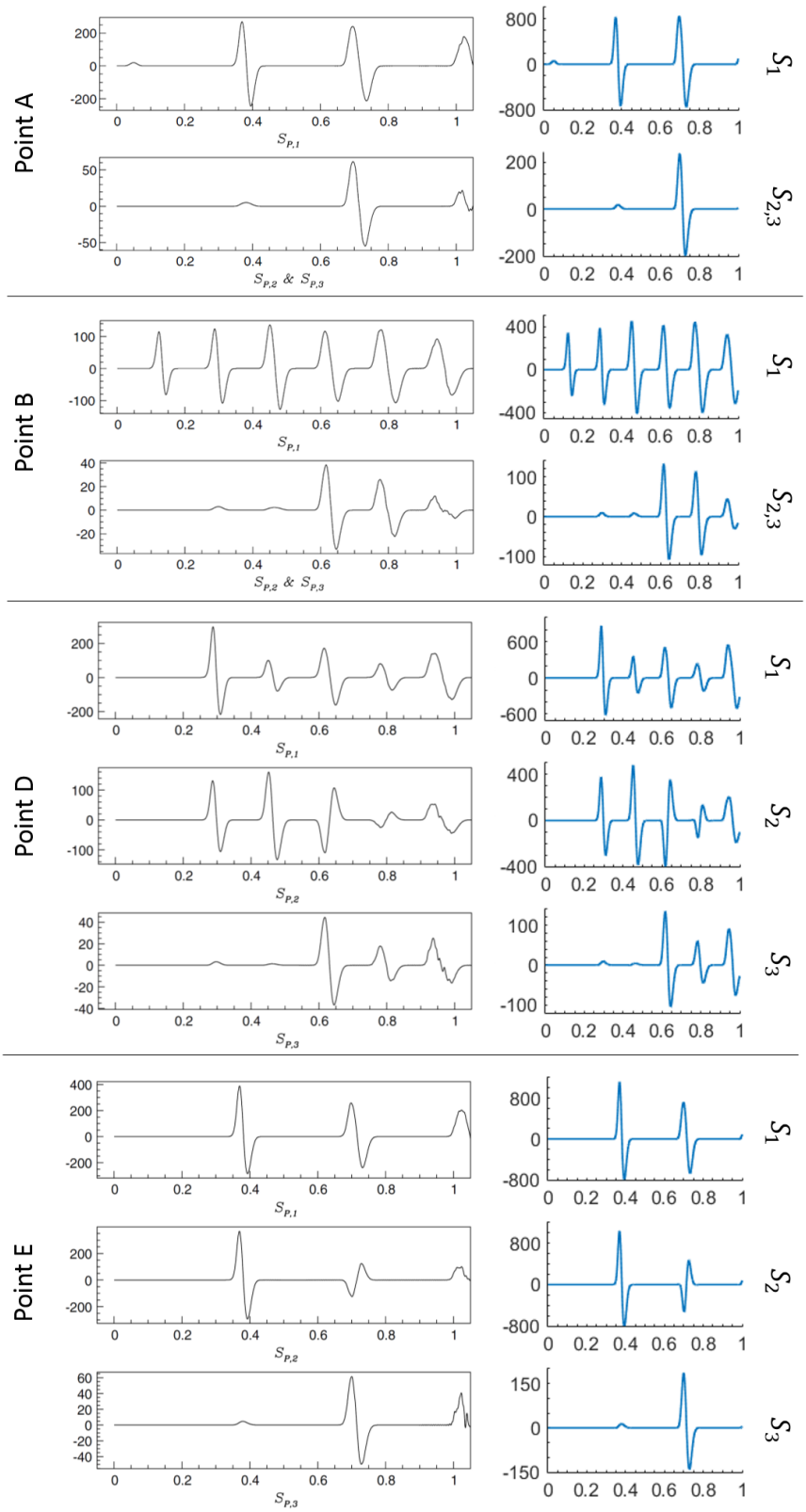


Figure 4.19: Propagating pressure wave in a simple bifurcation with reflecting outflow boundary conditions: baseline sensitivities from [201] (left) and DDUQ sensitivities (right) at different probe locations.

	Point A	Point B	Point D	Point E
R_1	0.0051	0.1427	0.1870	0.0896
R_2	0.1715	0.2482	0.0600	0.1393
R_3	0.1715	0.2482	$8.4 \cdot 10^{-4}$	0.1646

Table 4.6: Propagating pressure wave in a simple bifurcation with reflecting outflow boundary conditions: mismatch between peaks of DDUQ sensitivities and [201].

conditions are described as

$$Q = \frac{p - p_{out}}{R_p}, \tag{4.35}$$

where p is the computed (or measured) pressure at the outlet of the terminal vessel, $p_{out} = 3.2$mmHg is the constant hydrostatic pressure measured experimentally, and R_p is the peripheral resistance, reported in [129, 201]. Due to the large size of the system, we are interested in the solution in 9 output representative vessels, identified in Figure 4.20. The deterministic solution obtained with DDUQ after 10 deterministic cycles is in good agreement with the literature (see Figure 4.21). Note that, as remarked in [129], "due to the simple resistance boundary conditions used in the experiment, the measured pulse waves become less physiological in the more distal vessels. They contain many non-physiological oscillations, in both pressure and the flow, whose frequency is surprisingly well captured by the 1-D model."

For the stochastic test we consider 37 random inputs, corresponding to the parameter β_i in each vessel, represented via first-order PCE as in (4.30). Due to the high dimensionality of the problem, Smolyak sparse quadrature rules are employed [178]. Since 10 stochastic cycles are computationally expensive, we run $N_{det} < 10$ deterministic cycles, and $N_{stoch} = 10 - N_{det}$ stochastic cycles initialized in such a way that the initial expected value is set to the deterministic solution, and the initial standard deviation is 10% of the latter.

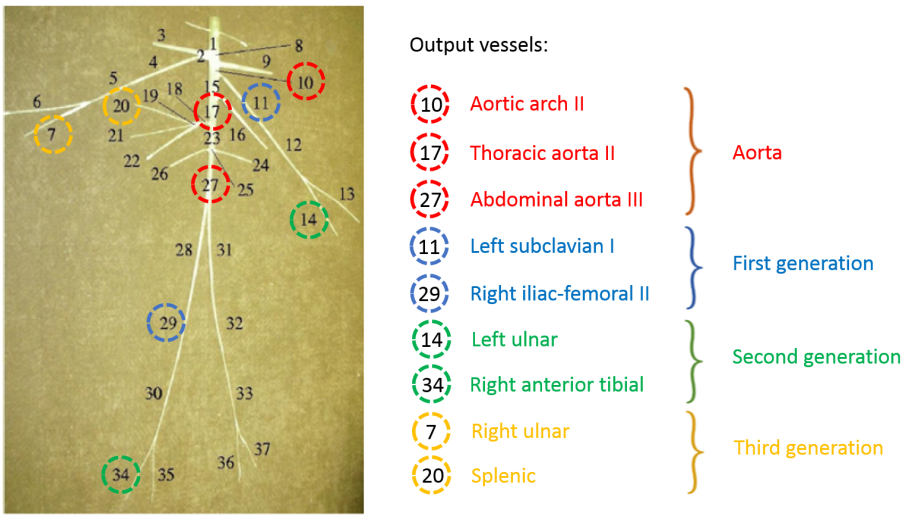


Figure 4.20: 37-vessel network and output probe vessels.

We introduce the notation

$$S_{i,j} = \langle \beta_j \rangle \langle \partial p_i(\xi)/\partial \beta_j(\xi) \rangle, \qquad R_{i,j} = \left| \frac{\max_t S_{i,j}(t)}{\max_t S_{i,j}^{XS}(t)} - \frac{\min_t S_{i,j}(t)}{\min_t S_{i,j}^{XS}(t)} \right| \tag{4.36}$$

to denote the sensitivity of the solution in the i-th vessel with respect to the j-th parameter, and the corresponding peak mismatch. As for the simple bifurcation case, assuming that $S_{i,j} = \alpha S_{i,j}^{XS}$ for some scaling factor α, we expect $R_{i,j} \simeq 0$. Maximum and minimum values of sensitivities are provided in [201], therefore it is possible to compute $R_{i,j}$ precisely. The results for $N_{det} = 9$ and $N_{det} = 7$ are shown in Figure 4.22. The mismatch is significantly high in both cases, likely because one or three stochastic cycles are not enough for the unsteady stochastic problem to got to regime. However, we noticed a drastic reduction passing from an initialization with $N_{det} = 9$ to $N_{det} = 7$. Therefore, we expect R_{ij} to further decrease as long as N_{det} is decreased. Simulations for such cases have been launched, but crashed because of memory issues. In the future we plan to optimize the current code to improve the efficiency of the data structures employed, for a solid successful validation of the method in realistic geometries.

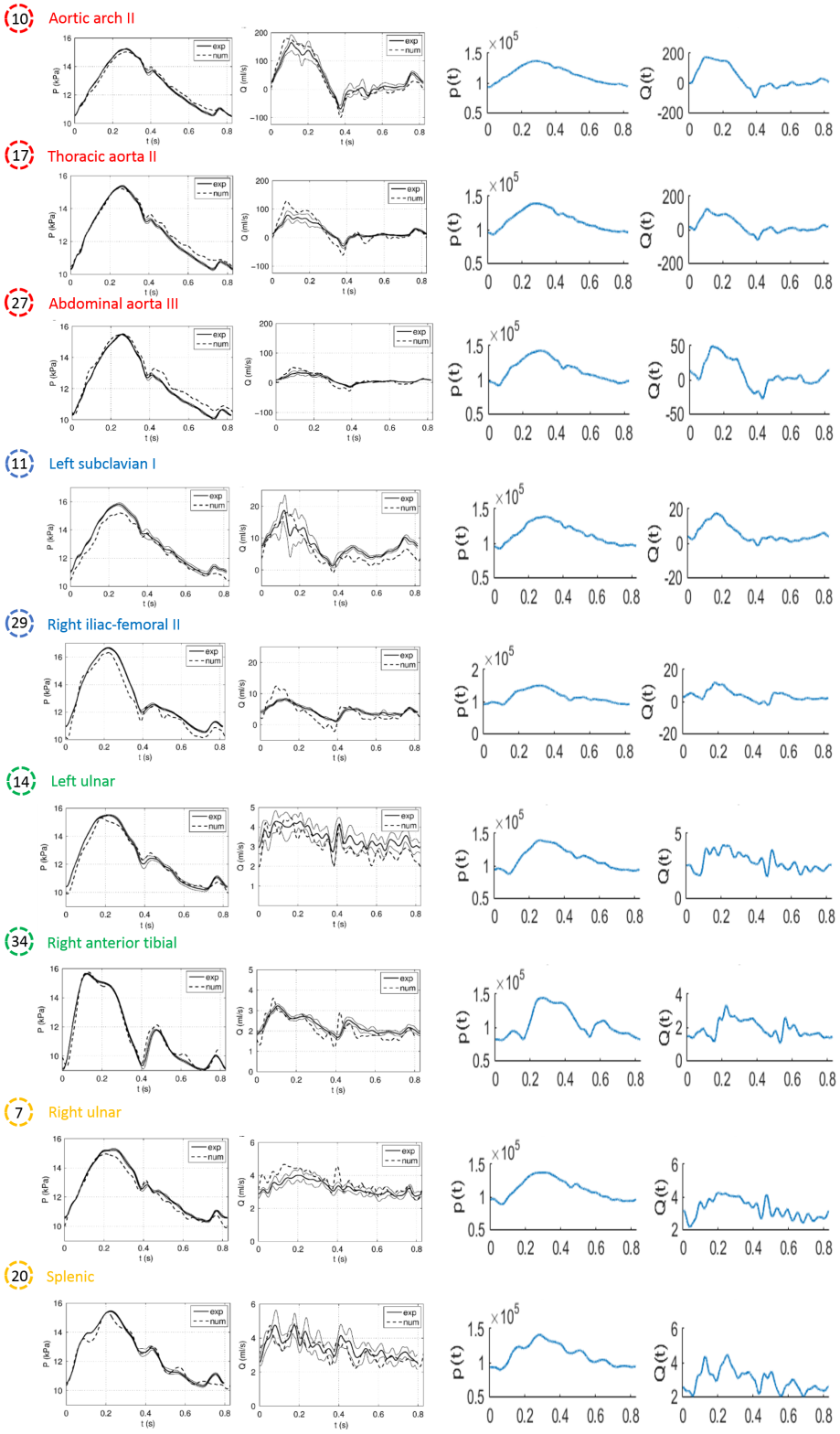


Figure 4.21: Deterministic baseline solution form [201] (left) and DDUQ solution (right) in output probe vessels of 37-vessel network.

R_{ij}	$\partial\beta_{10}$	$\partial\beta_{17}$	$\partial\beta_{27}$	$\partial\beta_{11}$	$\partial\beta_{29}$	$\partial\beta_{14}$	$\partial\beta_{34}$	$\partial\beta_{7}$	$\partial\beta_{20}$
∂p_{10}	3976.98	662.25	630.49	668.02	234.22	559.64	108.62	998.96	175.42
∂p_{17}	1606.98	1643.57	355.65	606.66	152.97	538.95	63.97	923.22	253.92
∂p_{27}	1753.52	760.55	1079.76	717.33	110.79	632.86	90.46	917.23	71.44
∂p_{11}	2014.02	624.61	649.71	402.02	347.07	4.82	66.58	262.89	146.86
∂p_{29}	1009.92	505.68	380.76	576.76	350.39	350.45	52.99	710.43	74.65
∂p_{14}	1175.18	761.86	527.04	104.46	186.68	836.38	63.38	224.36	224.99
∂p_{34}	582.71	494.12	273.78	453.47	35.53	297.05	380.72	580.83	72.99
∂p_{7}	1245.33	803.49	674.66	552.93	293.95	157.07	106.87	1365.38	295.33
∂p_{20}	892.74	812.94	46.34	392.97	79.74	79.42	40.48	432.98	348.11

R_{ij}	$\partial\beta_{10}$	$\partial\beta_{17}$	$\partial\beta_{27}$	$\partial\beta_{11}$	$\partial\beta_{29}$	$\partial\beta_{14}$	$\partial\beta_{34}$	$\partial\beta_{7}$	$\partial\beta_{20}$
∂p_{10}	58.52	78.12	57.14	111.68	31.00	61.04	11.61	83.01	38.80
∂p_{17}	52.25	69.07	44.02	100.15	26.20	66.32	10.70	68.94	41.08
∂p_{27}	33.09	58.02	31.23	97.26	14.48	65.85	5.28	58.22	19.22
∂p_{11}	32.81	58.05	63.60	14.11	40.44	13.93	10.92	35.57	27.52
∂p_{29}	25.64	33.70	17.91	49.13	8.37	30.75	1.57	37.30	15.61
∂p_{14}	26.86	55.37	45.45	20.52	23.89	2.42	7.75	29.07	27.02
∂p_{34}	9.41	32.89	14.92	43.26	2.03	23.25	1.05	22.02	11.15
∂p_{7}	17.09	54.66	48.66	54.02	26.96	19.96	8.59	2.58	32.83
∂p_{20}	28.02	59.27	32.03	65.81	18.00	36.64	9.76	68.56	3.66

Figure 4.22: Mismatch coefficient for the sensitivity peaks in a 37-vessel network with 9 (top) and 7 (bottom) deterministic initialization cycles.

4.6 Final remarks

In this Chapter we proposed a new approach to perform uncertainty quantification in the cardiovascular network that is computationally affordable and reliable. This is achieved by combining the DDUQ approach [45], that employs domain decomposition to propagate uncertainties in large-scale networks scalably, with the TEPEM method for the steady Navier-Stokes equations [127], that allows the modeling of 3D dynamics in a fraction of the time employed for classical FEM strategies. The numerical results show that (i) weak scalability can be achieved for non-linear vector problems in realistic 3D domains; (ii) *educated* reduced models like TEPEM enable rapid and agile subsystem sampling in 3D domains, being computationally more affordable than full high-fidelity models, while comparably accurate, and (iii) provide statistical information about quantities of clinical interest, such as the wall shear stress, that are completely out of reach for the traditional 1D models; (iv) stochastic simulations in patient-specific geometries achieve a good trade-off between computational affordability and accuracy, compared to the traditional models used in the literature. In the future we plan to consolidate the validation of the DDUQ method on unsteady problems, and to model fluid-structure interaction with TEPEM.

Chapter 5

Platform and algorithm effects on computational fluid dynamics

Acknowledgements. This chapter reflects the content of [97], in collaboration with Tiziano Passerini, Umberto Villa, Jaroslaw Slawinsky, Alessandro Veneziani, and Vaidy Sunderam.

5.1 Introduction and Background

Overall performance of HPC resources depends on two interrelated factors: (1) the architecture of the physical resource and (2) its optimal exploitation for the specific problem to solve. In real production settings, performance must be judiciously balanced with cost.

As for point (1), traditionally performance of HPC applications has been measured by a single metric, i.e., time to completion for the particular application at hand, parameterized with respect to problem size and number of processing elements used. Nevertheless, with the advent of cloud computing, the viability of executing parallel applications on the cloud (either through self-assembly or renting a prebuilt cluster) and the actual dollar cost effectiveness of executing HPC applications on different target platforms have become relevant. On the other hand, communication is an issue of paramount concern in the matter of efficiency. On clouds, a great deal of

attention has been devoted to data handling but there has been relatively little focus on interconnection network capabilities. For explicit message passing parallel programs, such as those which make use of MPI, data handling and interconnection network capabilities lead to substantial heterogeneity in communication, with significant impact on performance. In addition, it is worth stressing that most real-life applications we are interested in are not regular or symmetric and thus their MPI process communication graphs are intrinsically unevenly weighted. The most popular software packages for graph partitioning (see, e.g., [82]) employ algorithms that minimize the edge cut or the communication volume as to obtain load balance. Nevertheless, numerical analysis suggests that a suitable percentage of additional work on each processing unit benefits the overall performance, due to a faster convergence of the iterative solver. The interplay between the additional numerical costs on each processor, the advantages induced by the faster convergence of the iterative method and the total communication time (which in turn relates to the specific architecture employed) is not trivial in problems of practical interest.

In this work we tackle these issues by exploring two aspects related to work partitioning.

(a) We re-map the effective topology of the application's interconnection network by managing the allocation of MPI processes to processor cores *before* the execution of the application, so that highly coupled MPI processes are "close," i.e., mapped on cores within a single node. In this way the intra-node communication is maximized and the long-distance inter-node communication is reduced.

(b) We consider well-established methods to associate mathematical formalism to the parallel solution of complex systems of partial differential equations (PDEs). In particular, we resort to *domain decomposition techniques* (DD) to detect the *optimal* splitting of the tasks that minimizes the computational time. This method was historically introduced – well before the advent of parallel computing – to compute manually the solution of PDEs by splitting the process over different subdomains of the region of interest to take advantage of simple geometries (e.g., a L-shape domain

was split into rectangles Fig. 5.2) where simple methods were available. Nowadays, DD is a powerful approach to manage the solution over different computational resources either with or without overlapping of subdomains, depending on the specific problem of interest and the identification of optimal interfaces to minimize inter-node communications.

Our reference application is the solution of problems related to computational hemodynamics, blood flow and solutes like Oxygen. We aim at demonstrating the relevance of all these issues in a realistic context, when dealing with a patient-specific setting to be considered as one out of many similar - but different- cases to be routinely simulated. We use an object oriented C++ library for the solution of PDEs with the finite element method (FEM) called LiFEV ("Library for Finite Elements 5") [4], extensively adopted in several projects of practical interest – see, e.g., [139, 138, 95, 133, 191, 87].

After providing some background on the numerical setting and the formulation of the two classes of problems used to test the performance of IaaS grids and clouds as opposed to local clusters and to experimentally detect the optimal partitioning that guarantees the fastest convergence of the numerical solver and a brief summary of the packages used, in Section 5.2 we discuss our experiences with comparing cost and utility on three typical platform types: (a) Infrastructure as a Service (IaaS) clouds, (b) grids, and (c) on-premise local resources, with a particular focus on process-to-node mapping vis-a-vis efficiency.

In Section 5.3 we consider the work balance in terms of DD and interface handling. We present an automatic procedure to optimize the mapping of the sub-domains to the available processing units based on graph analysis. We first consider a non overlapping strategy, where each domain shares with the others only the interface (e.g., a surface cutting in our case the volume of the artery of interest). However, it is well known that this is not necessarily the best option. In fact, a faster convergence to the desired solution in the iterative-by-subdomain approach can be attained if we allow some overlapping.

In Section 5.4 we test this option in both idealized and real 3D geometries. We show that the detection of the optimal overlapping in real cases – albeit non trivial – has the potential to significantly reduce the computational costs of the entire solution process.

5.1.1 The numerical problem

Computational hemodynamics requires the study of incompressible fluids described by the Navier-Stokes equations (NSE) [150, 66]. From the computational viewpoint, these equations are very challenging, for intrinsic mathematical features (see, e.g., [66]). In our tests NSE - completed by appropriate initial and boundary conditions – are solved for computing blood velocity and pressure in an artery affected by a disease, called *cerebral aneurysm*. The latter consists of an abnormal sac in the artery, inducing non-physiological flow patterns that can lead eventually to rupture of the arterial wall and brain hemorrhage. The application of computational hemodynamics to the study of vascular diseases is time- and cost- sensitive, as it typically entails the generation of large data sets of simulations on patient populations, with the final goal of finding statistical correlations of flow patterns with outcome [47, 139]. Here, in particular, we consider a benchmark problem proposed in the Inaugural CFD Challenge Workshop [199], i. e. the study of blood flow inside a giant brain aneurysm in an internal carotid artery.

The equations are approximated by the Finite Element Method (FEM) combined with backward difference formulas (BDF) to handle the time dependence. With FEM, the solution is approximated by piecewise polynomial functions over subdivisions of the artery, called *elements*. The collection of elements is called *mesh*. This step reduces the partial differential equations to a system of ordinary differential equations in time. The latter is finally solved in selected instants by a second order BDF approximation. At each time step a large sparse (i. e. with the majority of entries of the associated matrix equal to 0) linear system needs to be solved. The more elements are introduced in the computational domain and the more instants

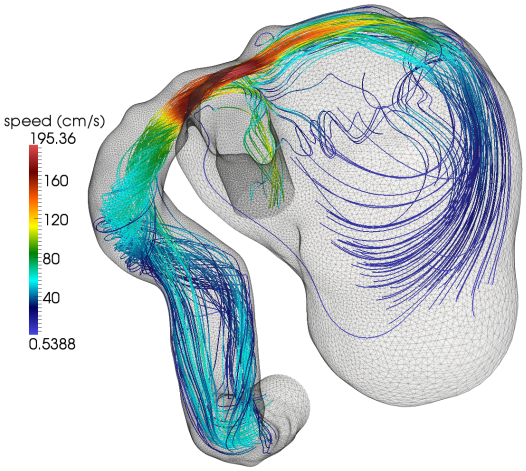


Figure 5.1: Solution of the problem, based on NSE, when $t = 0.28$s. Streamlines of the velocity field, when the flow rate is maximum over the cardiac cycle.

are collocated for the numerical solution, the higher the computational costs of the procedure are and the more accurate the solution is. In particular, here we consider a mesh with 837,154 elements, such that the total number of unknowns in the linear system is 3,162,146. The equations are collocated in 100 instants within the cardiac cycle (i. e. the simulation time step is $0.01s$). A snapshot of the computed solution is shown in Figure 5.1. Although current HPC facilities handle larger problems, these numbers can be considered representative of the size of problems of interest in CACT and SP in computational hemodynamics – as a reasonable trade-off between the accuracy requested by clinical applications and the expected timeline.

5.1.2 Domain decomposition techniques for the solution of Partial Differential Equations

DD techniques provide an important framework to associate mathematical formalism to the parallel solution of a complex PDEs system – see, e.g., [154, 188]. The PDE problem over a region of interest Ω is decomposed in subproblems to be iteratively solved by single processors or clusters up to the fulfillment of a convergence criterion

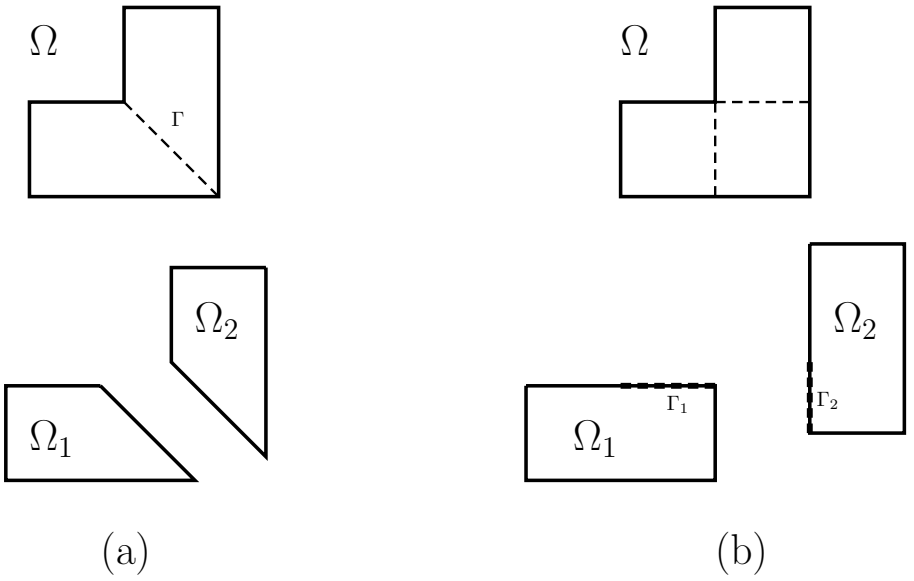


Figure 5.2: Schematic representation of (a) non overlapping and (b) overlapping DD in a L-shaped domain Ω. In the first case conditions on the interface Γ must fulfill compatibility constraints depending on the nature of the PDE for the split-by-subdomain solution to be equivalent to the unsplit one.

stating that the solution found is equivalent to the one of the unsplit system. Each subproblem exchanges information with the neighborhood ones by means of *interface conditions*. In non overlapping splittings, these conditions need to be properly chosen to guarantee that the split-by-subdomain solution is equivalent to the global one. In overlapping partitions, less constraints are required since synchronization conditions for each subdomain are prescribed on different space locations. In fact, each subdomain has its own interfaces. Notice that with overlap the PDE problem is solved multiple times on the overlapping regions, with a potential computational duplication overhead. However, beyond the more freedom when selecting the interface conditions, the iterative solver requires in general a lower number of iterations to converge. We illustrate the difference between the two approaches for a simple problem in Fig. 5.2.

The interplay of (i) additional numerical costs due to the overlap, (ii) efficiency advantages induced by the specific iterative methods and (iii) versatility of the selection of domain interfaces (and the associated conditions) for the communication time, is not trivial in problems of practical interest. Numerical analysis focuses typically on points (i) and (ii) in idealized or simple geometries, while in the present work we assess the performances when the geometry of Ω is nontrivial, following up previous works [58, 99].

To this aim we consider the differential *Advection-Diffusion-Reaction* (ADR) problem

$$-\sum_{i=1}^{3}\frac{\partial}{\partial x_i}\left(\mu\frac{\partial u}{\partial x_i}\right)+\sum_{i=1}^{3}\beta_i\frac{\partial u}{\partial x_i}+\sigma u=f, \tag{5.1}$$

for $(x_1, x_2, x_3) \in \Omega \subset \mathbb{R}^3$ with $\mu > 0$ and σ coefficients for simplicity assumed to be constant. Here the unknown u may represent the density of a species in a region where it diffuses with diffusivity μ, it undergoes to a chemical reaction with rate σ and it is convected in the domain by the vector field $\boldsymbol{\beta} = [\beta_1\ \beta_2\ \beta_3]^T$, that denotes the blood velocity and it is function of the space coordinates x_1, x_2, x_3. When available, it can be prescribed analytically, as we do in the tests in idealized geometries. More in general, it is retrieved by solving the NSE computed as in the previous Sections. The forcing term f is a given function of space too. Hereafter it will be set to 0 for simplicity. We associate with the equations the boundary conditions $u(\Gamma_D) = g(x_1, x_2, x_3)$, $\dfrac{\partial u}{\partial \mathbf{n}}(\Gamma_N) = 0$, where Γ_D and Γ_N are two disjoint portions of the boundary of Ω such that $\Gamma_D \cup \Gamma_N = \partial\Omega$. This is a simplified model of the dynamics of blood solutes like Oxygen in the arteries [156]. Specifically, we do not consider time dependence, since it does not introduce significant changes for the focus of the present paper. The NSE solution is therefore retrieved in a particular instant of the hart beat, the so called *systolic peak*, corresponding to the maximum opening of the ventricular valve.

To take advantage of domain decomposition, we split the domain Ω into two overlapping subdomains Ω_1 and Ω_2, such that $\Omega_1 \cap \Omega_2 = \Omega_o$ and $\Omega_1 \cup \Omega_2 = \Omega$. Let us denote by Γ_j the interfaces between the two subdomains ($j = 1, 2$), that is the portion

of the boundary of Ω_j that is not also boundary of Ω, in short $\Gamma_j \equiv \partial\Omega_j \setminus (\partial\Omega_j \cap \partial\Omega)$. The solution of the problem in each subdomain will be denoted by $u_j(x_1, x_2, x_3)$. We reformulate the original problem in an iterative fashion. Given an initial guess $u_j^{(0)}$ (typically $= 0$), we solve on each subdomain for $k = 1, 2, \ldots$

$$-\sum_{i=1}^{3} \frac{\partial}{\partial x_i}\left(\mu \frac{\partial u_j^{(k)}}{\partial x_i}\right) + \sum_{i=1}^{3} \beta_i \frac{\partial u_j^{(k)}}{\partial x_i} + \sigma u_j^{(k)} = f \qquad \text{in } \Omega_j, j = 1, 2 \tag{5.2}$$

with boundary conditions

$$u_j^{(k)}(\Gamma_D \cap \partial\Omega_j) = g(x_1, x_2, x_3), \quad \frac{\partial u_j^{(k)}}{\partial \mathbf{n}}(\Gamma_N \cap \partial\Omega_j) = 0, \quad u_j^{(k)}(\Gamma_j) = u_{\widehat{j}}^{(k-1)}(\Gamma_j), \tag{5.3}$$

(where $\widehat{j} = 2$ for $j = 1$ and $\widehat{j} = 1$ for $j = 2$) up to the fulfillment of the convergence condition to check that the solution in the overlapping region is not changing significantly along the iterations.

At each iteration we solve two independent problems in each subdomain, while the communication by subdomain occurs in the latter of boundary conditions (5.3). The convergence of the iterative scheme depends in general on the size of the overlapping region. In fact, if the overlapping is 100 % of Ω, convergence is trivially guaranteed as at the first iteration (5.2-5.3) we are solving (twice) the unsplit problem. On the other hand, if the overlapping reduces to a volume-zero region, convergence is not guaranteed, as in general the juxtaposition of the two problems does not coincide with the original problem (if the interface conditions are chosen properly).

The one presented here is the so called *additive* formulation of the overlapping DD method, where the two subdomain problems can be solved simultaneously – as opposed to the *multiplicative* version, where one subdomain can be solved only when the problem on the other subdomain is completed. In the multiplicative formulation a faster convergence is guaranteed in terms of number of iterations (about one half of the additive scheme), but the algorithm has an intrinsically sequential structure. From now on, we refer only to the additive algorithm.

The selection of the interfaces Γ_j has the only constraint to guarantee a non empty overlapping. The optimal selection is the result of the trade-off between the compu-

tational cost of each subproblem and the reduction of the communication between processors. This will be investigated in Sect. 5.4 - see also [99].

5.1.3 Packages used by the numerical solver

For a more detailed description of the implementation of the numerical solver we refer to [173, 99]. We report here the complete list of required packages:

- LiFEV library [4], for the formulation of the algebraic counterparts to differential problems; this library is the direct dependency for our solver application;

- Third-party scientific libraries: (1) Trilinos [163] for the solution of linear systems (data structures and algorithms); (2) ParMETIS [82], used for mesh partitioning; ad hoc MATLAB scripts were prepared to add an overlapping region to an existing non-overlapping partition. (3) SuiteSparse [6], as a support library extending the capabilities of Trilinos; (4) BLAS/LAPACK libraries (generic or vendor-specific implementations); (5) NetGen [114] for generating the mesh to partition.

- General-purpose and communication libraries: (1) Boost C++ libraries [2] 1.44 or above, mainly used for memory management (smart pointers); (2) HDF5 [187], for the storage of large data on file; (3) MPI libraries (e.g., Open MPI);

- Compilers: C++ compiler (e.g., GCC version 4 or above); [optional] Fortran compiler, compatible with C++;

- Deployment tools: (1) GNU make; (2) Autotools; (3) CMake (version 2.8 or above).

5.2 CFD Experiences on clouds, grids and on-premise resources

Grids and especially clouds present real opportunities for CFD applications to execute on platforms other than their home environments [80]. Nevertheless, applications often continue to be executed only on the default "home" platform, even if other viable and better options are present, since executing the application on different target platforms may require a non-trivial amount of re-building effort (even if the actual application source code is untouched). Message passing parallel programs are a staple modality of numerical simulations and computational analyses. In addition to the parallel framework (e.g., MPI), codes depend on various other auxiliary components: scientific and mathematical libraries, header files, particular compiler options and flags. These parameters (or subsets thereof) are quite specific to a particular *target platform*. In the ADAPT project at Emory [172], we investigated the feasibility and ease of deploying classes of applications on target platforms other than those on which they normally execute. As a benchmark test, we have experimented LiFEV [4] whose home environment is a 128-core cluster, and ported it on other computational platforms: clusters, grids and Amazon's EC2 cloud. We conduct both a detailed comparison of platforms based on *utility* of the computational task to the user, function of the wait time and the cost, and further analyze strategies for process mapping when interconnection networks are heterogeneous.[1]

User-oriented performance analysis has recently been applied to research on HPC and Grid scheduling strategies. The value that users associate with a completed job is modeled as a *utility function*, with a generally non-trivial dependence on time [116]. The importance of a job to a user can be seen as a function of time, combining an index for the importance of the results and the user sensitivity to delay. It has been shown that a proper job scheduling strategy can significantly increase the performance of HPC systems, measured as the aggregate utility of their users [50, 52].

[1]Preliminary results from these exercises were presented in conference papers [173, 175].

Several works in the literature discuss an extension to this scenario, in which heterogeneous resources can be discovered and assembled from an arbitrary set of providers. In this case, the *utility* for the user may be defined based on a more detailed analysis of user-specific requirements. For instance, requirements may include the features of the physical resources (memory, processor speed, presence of GPU), presence of installed software or availability of specific services. It is then possible to discriminate between resource providers based on their ability to satisfy the requirements, in full or in part (*partial utility*) [169]. The evaluation of the utility function can be done at runtime, to decide whether or not to dynamically re-distribute resources to obtain an optimal "quality of execution," i. e., an optimal trade off between resource savings and performance degradation [170].

In our approach, the platforms are considered as interchangeable – after the conditioning process. We discuss the effort required to provision each platform with an environment adequate to sustain the user's task. Finally, we identify a basic set of user requirements (minimal cost and minimal execution time of the task) to define a user-based ranking of the tested architectures.

5.2.1 Heterogeneous Target Platforms

In our study, we compared five heterogeneous computational platforms supporting the parallel hemodynamics simulation. As the starting point for our analyses, we selected the in-house computing cluster `puma`[2] constituting a computational test bed for the LiFEV developer team. As a second platform, we used a larger compute cluster called `ellipse`, provided on a fee-for-use basis within our university. The third platform was the HPC supercomputer `lonestar` made available to the U. S. research community by Texas Advanced Computing Center. Next, we evaluated the usability of on-demand resources. The first such platform was `rockhopper` [5] offered as a part of the Penguin's On-Demand HPC Cloud Service [3] and the second platform was the IaaS cloud provided by Amazon's Elastic Compute Cloud (EC2)

[2]This is the "home" environment where the application is run by default.

	puma	ellipse	lonestar	rockhopper	ec2
type	cluster	cluster/grid	grid	cloud cluster	IaaS Cloud
cores	2x2	2x2	2x6	4x12	4x4
RAM	8GB	4GB	24GB	2.5GB/slot	66GB
network	SDR IB	1GbE	QDR IB	QDR IB	10GbE
storage	NFS	NFS	Lustre	Lustre	local fs
support	full	very limited	limited	online	none
OS	Rocks 5.1	CentOS 4.8	CentOS 5.5	CentOS 5.6	AMI 12.03
access	user space	user space	user space	user space	privileged
MPI	Open MPI	none	MVAPICH2	Open MPI	none

Table 5.1: Specification of a single node of the test architectures.

service. From the rich EC2 resource offerings, we picked the most powerful instances `cc2.8xlarge` from *Cluster Compute* (referred to as `ec2` in the following).

The five platforms are heterogeneous in many respects: they differ in hardware configuration, availability (measured as wait-time before execution), access modality (privileged vs. unprivileged user), storage (e.g., size of user disk space and presence of a shared file system), build (e.g., the compilers and system tools availability), computational aggregation (e.g., presence of configured MPI environment), and execution (e.g., interactive shell). Table 5.1 collects the main features of the chosen targets. We refer to [173, 175] for further details.

5.2.2 Metrics

We aim to compare different hardware platforms with respect to the execution of the same task, evaluating several different metrics.

As previously mentioned, hemodynamics applications are both time- and cost-sensitive. It is worth noting that optimizing these two aspects separately leads in general to conflicting strategies, as it is often the case that the most expensive

hardware resource provides the result in the shortest time, as we will see later on. We therefore consider both traditional metrics ("time to completion" and "cost per simulation"), and a user-specific combination of these two, corresponding to the "perceived cost" of the computational experiment.

Time to completion This is the wall clock time from program launch to final exit. In the mainstream HPC community, in which it is a primary focus, it is not common to include the queue waiting time. In terms of utility in the sense adopted in this work, queue time is certainly important but it is highly variable and, in fact, our platforms presented few queue delays compared to the execution time of the application. We decided to exclude queue time in our analysis for the sake of simplicity and uniformity, especially since "on-demand" IaaS clouds are practically characterized by zero waiting time.

Cost per simulation The overall cost for the execution of the job depends mainly on the unit cost of the hardware resource (cost per core-hour), its pricing policy (by core or by node, by hour or prorated), and on the execution wall-clock time. Other factors, that we consider negligible relative to the former (for our application), are the size of occupied storage and/or volume of data staged in and out. A ranking of the different platforms based on the metric *cost per simulation* is shown in Table 5.2. For each platform, we reported in the table the cost of the cheapest use case.

Utility function The *utility function* expresses the job's value to a user, as a function of time. This has a user-specific, complex dependency on several parameters, including expenditures, time to completion, and significance of the task. Following [52, 108], we consider a simple linear utility function with customizable maximum

rank	price per simulation [$]	target	# of MPI proc.
1	$3.53	`ec2-1`	16
2	$4.76	`ec2-2`	16
3	$5.31	`ellipse`	4
4	$5.70	`puma`	16
5	$7.27	`ec2-4`	64
6	$9.52	`lonestar`	8
7	$13.09	`ec2-8`	32
8	$20.99	`rockhopper`	16
9	$22.59	`ec2-16`	32

Table 5.2: Cost of the benchmarked architectures

(starting) value and slope, as shown in figure 5.3. The equation reads

$$U(t) = \begin{cases} U_{max} & \text{if } t \leq T^* \\ U_{max}\left(\dfrac{T_0 - t}{T_0 - T^*}\right) & \text{if } T^* < t \leq T_0 \\ 0 & \text{if } t > T_0. \end{cases}$$

U_{max} is a measure of the *importance* of the job to the user, and we assume that it can be given a monetary value, as the price that the user would be willing to pay for the simulation. T^* is the expected completion time, which can be estimated in several ways. We use a simple averaging method defined in section 5.2.3, based on the performance of the available platforms. T_0 is the user-defined time at which the utility is zero, while the distance $(T_0 - T^*)$ is a measure of the user's *delay tolerance* and can be measured as a multiple of the expected completion time T^*.

With this formulation, we assume that there is no loss of value during the expected duration of the job (when $t \leq T^*$). An extension of the model could take into account the decrease in the utility function during runtime, reflecting the fact that faster runtime is valuable to users [116].

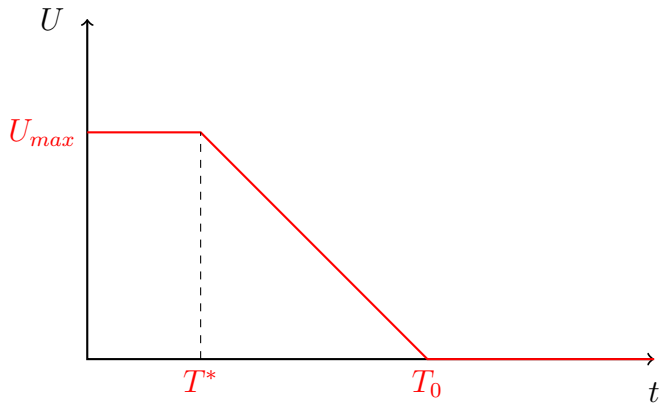


Figure 5.3: The considered utility function. U_{max} is a measure of the *importance* of the job to the user, T^* is the expected completion time, T_0 is the time at which the utility is zero [52].

5.2.3 Experimental Results

Our experiments on different architectures yielded interesting results. This discussion centers on cost and utility.

Performance, scaling and time to completion

In our study we tested the selected platforms executing a fixed-size simulation (over 3.1M unknowns) with varying numbers of processors, i.e., a strong scalability benchmark. All tested clusters allowed the reservation of computing resources by specifying the number of processes (or slots) used by the parallel job. However, in the case of the Amazon EC2 cloud, we needed to set the execution policy: we assumed that each `ec2` instance can host a maximum of 16 processes (as they have 16 physical cores) and we decided to map the MPI processes onto the physical nodes in round-robin fashion. As Amazon charges users on the basis of running instances, we decided to optimize the cost of the benchmark by testing small assemblies of `ec2` first, and then to increase the number of nodes in the assembly by powers of 2. For this reason, we present several configurations of cloud instances; we label such separate assemblies

as ec2-i, where i is the number of ec2 nodes.

The application repeats the same set of operations in each simulated time frame (in our case corresponding to 0.01s intervals). For each considered hardware platform, the time required to compute a single frame was observed to be constant during the course of the simulation. We, therefore, use the average computing time for a single frame as a proxy for the performance of the hardware resource. This facilitates a side-by-side comparison of all platforms, including cases when the simulation could not be completed due to cluster usage policies (e.g., ellipse limits the job execution time to 12 hours so for jobs that spanned small numbers of cores only a fraction of the entire simulation could be done).

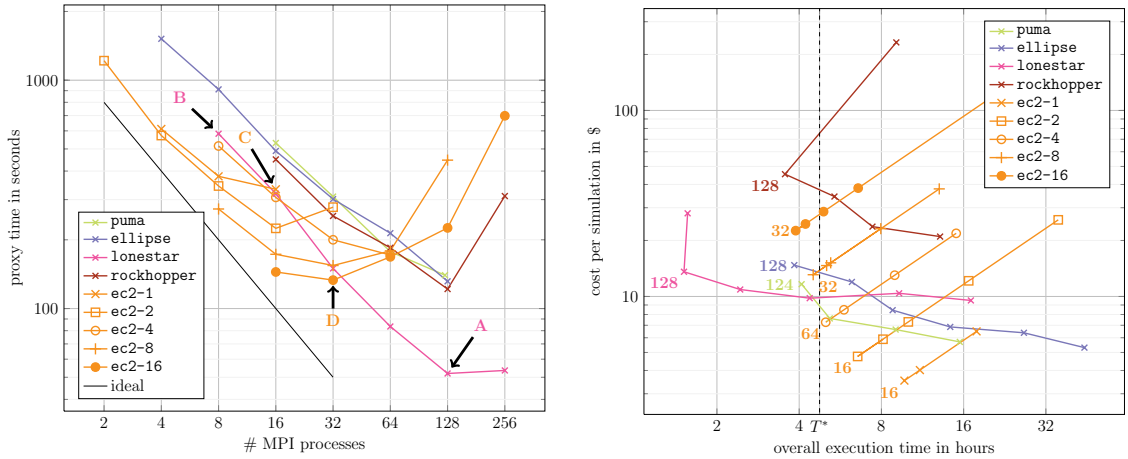
The graph in Figure 5.4a shows a comparison of the performances of the different platforms, as a function of the number of computing cores. The clusters puma and ellipse, the grid lonestar and the cloud cluster rockhopper achieve good strong scaling up to 128 computing cores, while they show a significant decrease in performance for larger numbers of cores. In particular, Point **A** in the figure corresponds to the fastest execution case in our experiment, that is running the simulation with 128 computing cores on lonestar. If this metric is used to represent utility or value to the user, it is clear that when using more than 32 cores, lonestar is the best platform.

ec2 resources scale less well. ec2-1 achieves good scaling only in the range 4-8 cores, ec2-2 up to 16 cores, ec2-4 up to 32, ec2-8 up to 16 cores, while ec2-16 does not achieve strong scaling in any range. Point **C** in Figure 5.4a corresponds to the case when the simulation was sustained by 16 computing cores on a single ec2 instance. It is worth noting that the time to completion in this case matches the time to completion obtained using 16 computing cores on lonestar. Most significantly, the time to completion required by ec2-1 when using 8 computing cores is lower than the time required by lonestar with the same number of computing cores. This result suggests that one of the advantages of IaaS clouds is the availability of powerful hardware configurations (both in terms of memory and CPU clock speed), that can

match and outperform the computing nodes provided by standard grid resources. This finding is in agreement with previous reports. A study [157] pointed out that when an EC2 user reserves an entire computing node (this happens in our case using `cc2.8xlarge` instances) the impact of virtualization is negligible since processor cores are not shared among users (see also [110, 195]). This results in performance comparable to "bare metal" hardware. As predicted by Iosup and coworkers [107], this is a significant advantage of *Cluster Compute* instances over former offerings by EC2, that were suffering from performance degradation due to concurrency of multiple users or applications using the same processor. On the other hand, the performance of `ec2` platforms seems to be sensitive to overload of the instances, as shown by the poor strong scaling achieved by `ec2-1` when all of the 16 available computing cores are used for the execution of the simulation.

Point **D** corresponds to the fastest execution on `ec2` resources, that is running the simulation with 32 computing cores using `ec2-16`. In this case, we launched 16 EC2 instances and allocated 2 computing cores on each instance in a round robin fashion. The loss of performance of `ec2-16` as the number of cores per instance increases suggests that *when requiring a relatively large number of instances, the physical connectivity of the nodes may become an issue, and the timings seem to be dominated by communication overheads.* The severe impact of network latency and bandwidth on EC2 performance is a known issue, especially for large assemblies of instances [110, 195, 107]. In terms of utility, therefore, individual EC2 nodes offer high performance but when communication across nodes or racks is involved, this platform is less attractive.

Based on the metric *time to completion* we can rank the different resources – Table 5.3 shows the wall clock times for the fastest run on each platform. The grid `lonestar` is by far the fastest resource, while `ec2` is generally the slowest. However, one of the solutions provided by `ec2` (namely `ec2-16`) matches the performance of the clusters `ellipse` and `puma` using a smaller amount of computational nodes (16 for `ec2-16` and 32 for the clusters `ellipse` and `puma`) and for a monetary cost which



(a) The average computation time per simulated time step (proxy), for the benchmarked architectures

(b) The labels indicate the number of MPI processes in the fastest run. $T^* = 4h\ 44m$ corresponds to the average value among the fastest runs.

Figure 5.4: The average computation time per simulated time step (a) and the relation between the cost and time of the simulation (b).

is about twice as much the estimated operational cost of the in-house computing facilities (renting one node on `ec2` is about four times the operational cost of our in-house facilities). This extra cost comes, however, with big advantages: `ec2-16` is an on-demand resource and it is immediately available, whereas in-house computing facilities are shared among many users and therefore waiting times to obtain the needed resources may be extremely long. This result further demonstrates one of the strengths of on-demand resources as compared to on-premise resources, i.e., `ec2` can count on a more efficient hardware configuration. The cloud cluster `rockhopper` performs best among the tested on-demand resources and better than the tested on-premise resources. However, it is still significantly slower than the tested HPC cluster.

rank	time to completion [s]	target	# of MPI proc.
1	1h 31m	`lonestar`	128
2	3h 33m	`rockhopper`	128
3	3h 50m	`ellipse`	128
4	3h 53m	`ec2-16`	32
5	4h 05m	`puma`	124*
6	4h 30m	`ec2-8`	32
7	5h 00m	`ec2-4`	64
8	6h 33m	`ec2-2`	16
9	9h 43m	`ec2-1`	16

Table 5.3: The performance ranking of the hardware resources for the fastest run based on the metric *time to completion*.
* One node is permanently down.

Utility function

Ideally, users desire to minimize both simulation cost and time to completion but these objectives compete with each other. This is confirmed by our tests where the cheapest resource, namely `ec2-1`, was also the slowest one. In Figure 5.4b we present how the cost per simulation relates to the time to completion for the different architectures. The closest points of the graphs to the origin of the axes represent execution cases that minimize both metrics. Clearly, the decision on which architecture to prefer cannot be made based on a single attribute. The general trend of these characteristics shows an increase in the cost per simulation with the decreasing time to completion. A remarkable exception is `ec2`, for which cost increases with time to completion. In fact, on this platform slower executions achieved with few computing cores are actually more expensive due to the policy requiring the reservation of 16-core instances.

To define a ranking of the tested platforms based on a user-centric performance

analysis we evaluate the utility function defined in Section 5.2.2. We consider three user profiles,

Case 1. The job has high priority, and the user has little delay tolerance;

Case 2. The job has average priority, and the user has average delay tolerance;

Case 3. The job has low priority, and the user has large delay tolerance.

Referring for the sake of example to the results of our benchmark, we assume that the value of a simulation to the user (i.e., the cost the user would be willing to pay) is in the range between $3.53 (low) and $22.59 (high). More precisely, we assume that a job with low priority has a value to the user equal to the average cost of the simulation over the tested architectures, i.e., $10.31. We assign double this value to a high priority job ($20.62) while an average priority job will have an intermediate value between the previous two ($15.465). We further assume that for all the user profiles the expected time to completion T^* is the average value of the times measured on the different architectures (cf. table 5.3), i.e., $T^* = 4\text{h } 44\text{m}$. A user with an average delay tolerance is represented by a utility function that remains non-negative for a runtime up to twice the expected value (i.e., $T_0 = 2T^*$). A user with large delay tolerance accepts twice as much delay ($T_0 = 3T^*$), while a user with small delay tolerance accepts half as much (i.e., $T_0 = 1.5T^*$).

We plot in Figure 5.5 the user-specific utility functions together with the graphs shown in Figure 5.4b. As discussed in previous sections, each platform was tested in several use cases (varying the number of computing cores); a case is considered *useful* to the user if it is represented by a point on the cost/time plot located below the graph of the user's utility function. For the sake of example, we reported on the plot the points corresponding to the cases discussed in detail in the previous sections. Point **A** corresponds to the fastest execution of the simulation in our experiment, obtained when using 128 cores on `lonestar`. This case is *useful* to user profiles 1 and 2, for which the *importance* of the simulation is greater than the actual cost. User profile 3 would not consider this case *useful* due to its high cost.

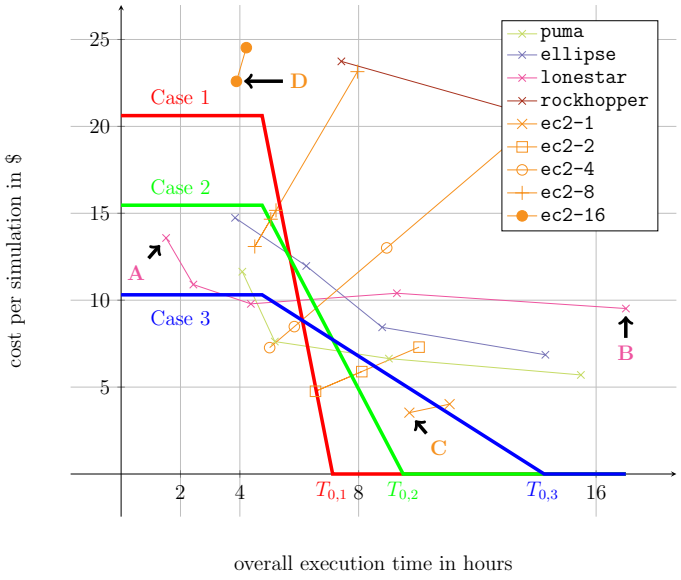


Figure 5.5: Evaluating the cost/time characteristics of the different platforms against the user-specific utility function. $T_{0,1}$, $T_{0,2}$ and $T_{0,3}$ are the times at which the utility function is zero for user profiles 1, 2 and 3, respectively.

Despite the cost being relatively lower, the use case of `lonestar` represented by point **B** (8 computing cores) is not *useful* for any user profile, because for all of the profiles the time to completion of the simulation exceeds the time T_0 for which the utility function is zero. The use case of `ec2-1` corresponding to the cheapest execution in our experiment (16 computing cores) is represented by point **C**; because of the long time to completion, this use case is only *useful* to user profile 3. The fastest execution achieved on `ec2` resources (2 computing cores on each instance of `ec2-16`) is represented by point **D**. This use case has a cost exceeding the maximum value of the utility function for all the user profiles, so it is *useful* to none of them.

In our experiment, a variety of platforms can meet the requirements of user profile 1. Fast and expensive architectures (e.g., `lonestar`) can be chosen in alternative to slower and cheaper ones (e.g., `ec2`). However, because of a small delay tolerance, a cheap option (`ec2-2`) has to be ruled out, being penalized by high execution times.

The second user profile has the largest pool of *useful* choices, including the cheaper (and slower) `ec2-2`. For user profile 3, because of the low priority assigned by the user to the job, most of the fastest options (`lonestar`, `ellipse`) have to be discarded. On the other hand, the on-premise cluster `puma` and some of Amazon's instances (most significantly the very cheap `ec2-1`) can meet the user's requests.

According to this model, one of the on-demand resources (`rockhopper`) is not *useful* to any of the considered user profiles. Amazon's diverse offering allows instead this service to be competitive for a wide range of user profiles, being able to provide reasonably small execution times (`ec2-8`) or extremely cheap solutions (`ec2-1`). On-premise resources do not perform well compared to HPC machines, being significantly slower, and in most cases they are also outperformed by cheaper on-demand resources. As a result, they are competitive only in specific execution cases (i.e., with the proper choice of the number of computing cores). Finally, `lonestar` is a very strong competitor in the first two user scenarios (average to high job priority), while its performance is matched and outperformed both by on-demand and on-premise resources in the third scenario (low job priority and high delay tolerance).

Notably, the on-demand cutting edge offering by Amazon EC2 has the advantage of availability. In fact, our analysis does not consider queue waiting times that may diminish the attractiveness of shorter execution time on grid resources. This feature would make the IaaS choice even more convenient. Moreover, the cost per simulation on the resources offered by Amazon can be optimized with a proper scheduling policy that takes into account the specific pricing policy of Amazon (per-node rather than per-core). Furthermore, if cost needs to be minimized, it is possible to select cheaper Amazon instances such as `cc2.4xlarge`.

5.3 Adaptive mapping of parallel components on physical resources

Communication plays a major role in the performance of parallel architectures, especially when a pondered load balance is hardly achievable, due to the intrinsic irregularity and asymmetry of the physical phenomena under examination. IaaS clouds are particularly sensitive to such an issue, as an effect of the general purpose nature of the interconnection networks. Hence, improving the layout of the tasks of a parallel application on a particular hardware architecture is an attractive research subject as it may increase the performance of the application without requiring modifications to the source code. Rubik [7] is a software toolkit that applies simple geometry transformations (e.g., splits, tilts) to the Cartesian task topology of an application, altering its mapping to the Cartesian network topology. Thanks to the tasks shuffling, the underlying hardware may better support MPI collective operations by utilizing more hardware links while avoiding excessive latency or congestion [28]. Our solution follows a similar approach: we design the layout of MPI tasks before we execute the application. However, as our hemodynamic CFD code mainly uses point-to-point communication and has no statically defined communication topology we need to analyze a data exchange graph for a particular execution use case in order to optimize the tasks mapping.

A successful mapping strategy has to consider also the properties of the network backend. Eliminating unnecessary network hops may improve the overall latency and lead to better performance of the executed application. The project described in [183] considers the homogenous, multilevel IB network and offers an improved MPI implementation that exploits the network topology to increase intra-node communication and reducing the long distance inter-node communication. While this is an end point also for our project, we do not force a different MPI framework implementation. Moreover, even though currently we consider a simple, single hop network topology, we propose methods that can be extended to different scenarios. In

particular, when considering wider networks, a more aggressive planning of the mapping can be applied, aggregating machines from even geographically separated data centers to provide the computational platform for the distributed applications. The possibility of the inter-cloud aggregation and the performance of such conglomerate were evaluated by two of the authors in [174].

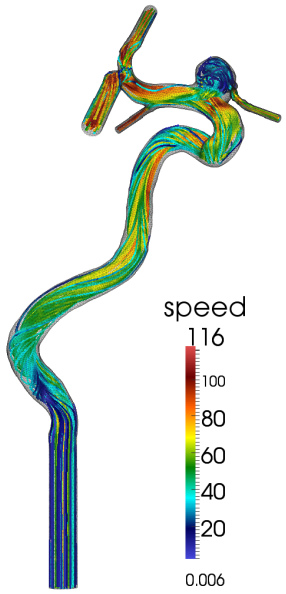
5.3.1 Test case

We consider here a subject-specific arterial geometry, extracted from medical images acquired and processed during the multi-center research project Aneurisk [13]. This kind of geometries are available for download through the web portal AneuriskWeb (`http://ecm2.mathcs.emory.edu/aneuriskweb`). To compute blood velocity and pressure in the subject specific geometry we solve numerically the NSE equations, using LiFEV. We simulate blood motion under pulsatile flow conditions, representing the pumping action of the heart. Blood is described as a Newtonian fluid with density 1 g/cm3 and dynamic viscosity 0.035 dyn/cm2. For the sake of the analyses presented here, we limit our simulation to a short time interval (0.10 seconds), solving the discretized NS equations at 10 instants (i.e., the simulation time step is $0.01s$). A snapshot of the computed solution is shown in Fig. 5.6a.

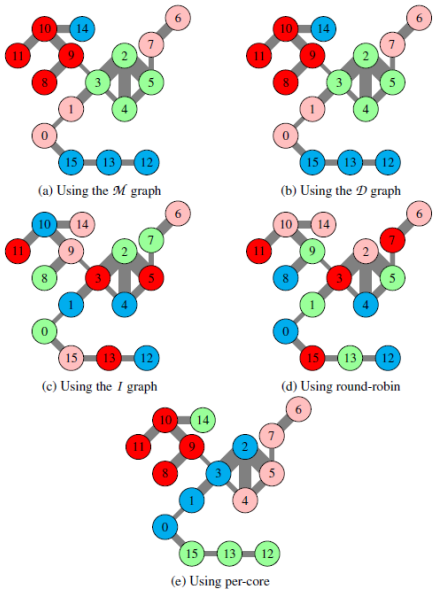
5.3.2 Offline mesh partitioning

The global mesh consists of the set of all elements, faces, edges and vertices in the tessellation. To each of those entities a unique identifier (global id) is assigned. We will denote by N_{el}, N_f, N_{ed}, N_v the total number of elements, faces, edges, and vertices in the global mesh. The topology of the mesh is described by the relationship between different geometric entities and can be expressed in table format (connectivity tables). For instance, the element-to-face table B_0, with size $N_{el} \times N_f$ is such that

$$(B_0)_{ij} = \begin{cases} 1 & \text{if face } j \text{ belongs to element } i \\ 0 & \text{otherwise.} \end{cases}$$



(a) Solution of Navier Stokes Equations for blood flow in an aneurysmatic vessel, for t = 0.05s. Streamlines of the velocity field colored by the blood speed.



(b) Different mappings of the coarse mesh for four 4-way nodes. Each color represents a part for a single host. Edge thickness represents the number of common vertices between parts. In this case, the partitioning using M and D are the same.

Figure 5.6

We have similar definitions for the face-to-edge table B_1, with size $N_f \times N_{ed}$, and the edge-to-vertex table B_2, with size $N_{ed} \times N_v$. Other connectivity tables can be obtained by composition of these.

In the parallel application, the computational domain may be partitioned by subdomains so that each process takes care of only a subset of the global mesh. In this section we consider non overlapping partitions and we refer to these subsets as "local" meshes. The splitting is achieved through the use of graph partitioning algorithms, such as those implemented in the libraries ParMETIS or Scotch, guaranteeing a proper load balancing among processes. The load is measured as the number

of mesh elements assigned to each process. When local meshes are not overlapping, each element belongs to one and only one process; however some faces, edges, and vertices are shared among two or more local meshes (interface entities). In any case, high quality partitionings should minimize the edgecut or the number of connections between disjoint partitions. This property is valuable to reduce the communication between processes necessary to synchronize interface unknowns.

For large scale simulations, mesh partitioning is a highly memory intensive operation due to the size of the global mesh and it is usually performed offline on dedicated machines since many times memory on computational nodes is a limiting resource. In more detail, in [172] the following strategy for mesh partitioning was considered.

1. The element adjacency graph A is built from the topological information stored by the mesh as $A = B_0 \wedge B_0^T$. Here and in the following we denote by the symbol $\wedge$ the Boolean multiplication operator between tables. The element adjacency graph is an unweighted symmetric graph such that two elements of the mesh are connected by a link if they share a common face.

2. The element adjacency graph A is partitioned in n_p connected components by using the recursive bisection multilevel partitioning algorithm implemented in ParMETIS, where n_p is the number of desired processes to run the simulation. The result of the partitioning algorithm is a vector p of integer numbers of length N_{el}, in which the value of entry i ($0 \leq i \leq N_{el} - 1$) specifies the partition element i was assigned to.

3. The global mesh is split according to the partitions of the elements induced by p. By introducing the Boolean table P, of size $n_p \times N_{el}$,

$$(P)_{ij} = \begin{cases} 1 & \text{if } p[j] == i \\ 0 & \text{otherwise} \end{cases}$$

the local mesh corresponding to process i is associated to its own set of elements, faces, edges, vertices evaluating the non-zeros entries of the i-th row of the

matrices $P_f = P \wedge B_0$, $P_{ed} = P_f \wedge B_1$, $P_v = P_{ed} \wedge B_2$, respectively. The above matrices are also used to define proper mappings between the local meshes and the original global mesh, while local connectivities tables B_i^l are obtained by extracting the appropriate rows and columns from the global tables B_i.

4. Finally, the partition connectivity graph M is computed. This is used to estimate the communication volume due to synchronization of the variables associated to interface entities. In particular M_{ij} is proportional to the number of variables shared by processor i and j, i.e., to the number of shared faces, edges and vertices. Thus, we have $M = \alpha_f P_f P_f^T + \alpha_{ed} P_{ed} P_{ed}^T + \alpha P_v P_v^T$, where $\alpha_f, \alpha_{ed}, \alpha_v$ are constant values expressing the number of unknowns associated to each face, edge, and vertex, respectively. These constants depend only on the polynomial degree of the finite element basis.

5.3.3 Evaluation procedure and results

To determine the performance of the different process placement scenarios, all simulation configurations were tested, i.e., three mesh resolutions, from 8 to 128 MPI processes, for three target architectures, using five different placement strategies (total 45 benchmark tests). The first three allocation techniques were formed using the information in the communication graph D, M, and an additional inverted communication graph I defined as $I_{ij} = M - M_{ij}$, where M is the maximal entry in M. We will refer to these strategies as *data*, *part* and *invert*, respectively. The partitioning of these graphs was performed using the `gpart` tool from the Scotch 6.0 software package, that implements graph k-way partitioning heuristics. As a result, we obtained three clusterings C_D, C_M, and C_I , such that each cluster included the same number of parts equal to the number of processing units available in a single node of the target machine and the graph cut-sets were minimized. C_D gave us the process placement optimized to the actual communication statistics. C_M was the mapping optimized with respect to the partition connectivity graph M that is a by-product

of the partitioning algorithm and does not require any additional work. However, given the strong correlation between D and M, similar results were expected in these two cases. Finally, using C_I, the worst placement choice was expected.

The second group of allocation strategies were methods commonly used to execute parallel applications with the OpenMPI: by-node round-robin (*rr*) and by-cores (*pcore*) placements. The first allocation strategy places processes one per node, cycling by node in a round-robin fashion, while the second uses all CPU cores on one node before moving to the next node (Fig. 5.6b). All tests were repeated twice and the data were averaged. Figure 5.7 shows the execution times for different MPI process placements for all configurations relative to the maximal execution time for that configuration (i.e., a chart bar having execution time 1 represents the least effective process placement in the configuration). A detailed description of the results can be found in [172]: a brief summary follows.

As expected, the execution times for different MPI process placements for all configurations demonstrate that deliberate MPI process placement significantly influences the overall performance of the application. In specific situations, the worst mapping in the configuration is almost one order of magnitude slower than the fastest. The standard *pcore* placement mapping is well-suited for processing the CFD application implemented with the LiFEV library. The reason for this behavior has its source in the implementation of the ParMETIS partitioning used by the application. ParMETIS uses recursive bisection, which matches the common 2^i-way multiprocessing architecture of contemporary computers. However, this allocation may be improved by the resource-oriented *part* placement, especially for larger numbers of hosts performing the computation. Moreover, to design such enhanced placement no extra computations – for instance evaluating communication statistics – are needed. The by-product information regarding the pairwise shared mesh entities from the partitioner phase can be utilized instead. As a result, this shows that it is possible to adapt performance of parallel MPI applications by communication-aware placement of their MPI processes if the computation structure, input geometry as well as

target architecture and network wiring is known and it may be done automatically by ADAPT.

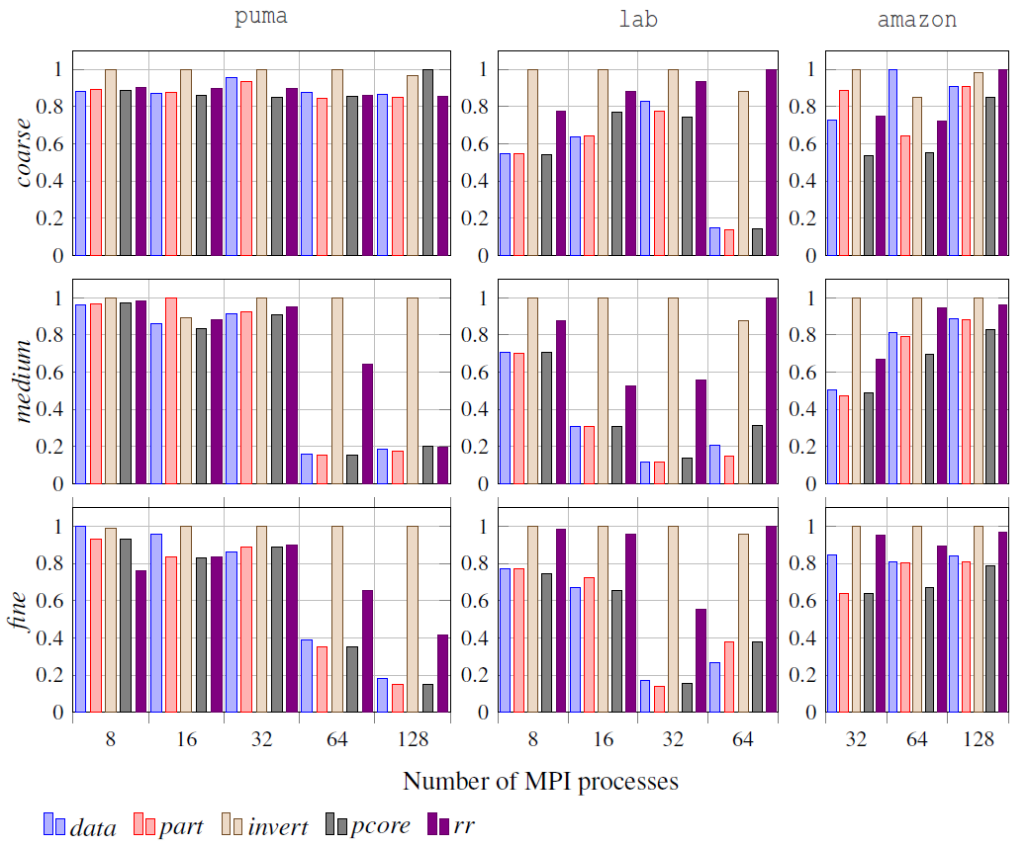


Figure 5.7: Relative execution times for all simulation configurations. Each value represents the speed-up of the mappings in relation to the less efficient mapping for the configuration.

5.4 Experimental optimization of parallel 3D overlapping domain decomposition schemes

In the previous section we dealt with process placement. Here we notice an additional opportunity to improve the overall performances on the algorithmic side by resorting to DD techniques. As pointed out, DD methodology relies on the segregated solution of the problem of interest on each subdomain and the iterative synchronization of the partitioned solutions. The advantage of non overlapping splittings is that each local problem is smaller, so we may expect a faster solution of each local solver. On the other hand, the final solution is the result of the iterative synchronization and the number of iterations depends in general on the shape of each subdomains, the interface conditions and the way the subdomains are synchronized. In this respect, the introduction of overlap introduces generally some advantages, since the number of iterations can be reduced. In addition, the interfaces can be positioned in a more flexible way, so to reduce the communication times. We investigate these aspects in a series of tests with different geometries.

In our tests we consider only the iterative-by-subdomain solution in the computational time. Meshing, partitioning and matrix assembly are not included in this analysis, since they are off-line costs that do not depend on the specific solution procedure. The time $T_{it}^{(k)}$ of each iteration (k) is computed as the maximum of the two parallel subdomain solution times $T_j^{(k)}$,

$$T_{it}^{(k)} = \max_{j=1,2} T_j^{(k)}.$$

The single processor time is given by the time for solving the linear system added by the communication time to read from the other processor the last of conditions (5.3), $T_j^{(k)} = T_{j,sol}^{(k)} + T_{j,com}^{(k)}$. For this particular problem, the computational cost per iteration is constant (denoted by $\overline{T_{sol} + T_{com}}$) , so we get

$$T = \sum_{k=1}^{N_{it}} T_{it}^{(k)} \approx N_{it}(\overline{T_{sol} + T_{com}}).$$

If we denote by p the percentage of overlap in the domain splitting (i.e., the ratio of the volume of the intersection of the domains to the total volume of the geometry), theory of overlapping DD proves that N_{it} *decreases* with p, T_{sol} increases with p while T_{com} depends on the position of the interfaces (precisely on the number of vertexes of the mesh on the interface), so it may change with p in an unpredictable way for a complicated geometry. We therefore expect that the value p has a major impact on the solver performances depending on the different geometries. It is worth noting that the total cost is a function of the mesh size N too. In this case, both factors N_{it} and $\overline{T_{sol} + T_{com}}$ get larger with h, as a price to pay to the improvement of the accuracy of the approximated solution achieved in this way.

In [99] we have presented several test cases in both academic and nontrivial geometries for a symmetric diffusion reaction problem (i.e., for $\beta_1 = \beta_2 = \beta_3 = 0$). The results point out that a small overlap, symmetric with respect to the non overlapping partition originally determined by ParMETIS, guarantees the best performances in terms of computational time. Here we investigate the same test cases in the more general cases of ADR, when the presence of the "directional" term weighed by $\beta_1, \beta_2, \beta_3$ is expected to have an impact on the detection of the optimal overlap.

We considered the following test cases.

Idealized geometries

(1) *Cylinder* – We consider a cylinder of length $L = 6cm$ and radius $R = 0.5cm$. The coefficients μ and σ are set as in [99] and the convective field has been selected to be constant throughout the domain. We use five meshes with different level of refinement, for each of the sizes of the overlap. We comment only the simulations we ran on the fine and very fine meshes as these are the cases of practical interest.

(2) *Idealized Aneurysm* – We consider an idealized representation of a cerebral aneurysm where a torus with radius 2cm is merged with a sphere of radius 0.5cm, representing the sac of the aneurysm. This test emphasizes the role of communication time. In fact a splitting with an interface intersecting the sac has more vertices than with interfaces involving only the artery. Overlapping DD allows to manage the

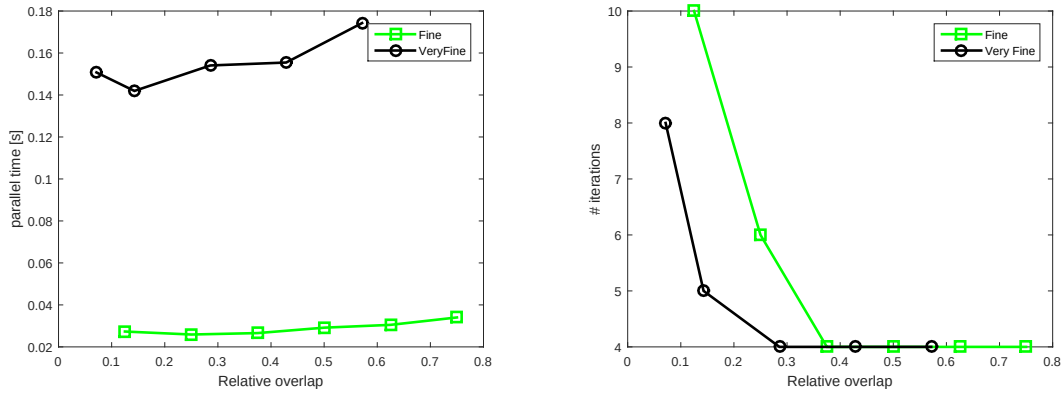
location of the interfaces so to avoid many vertices on the interface yet preserving workload balance between the subdomains. For more details, see [99]. The convective field has been selected to be tangential with respect to the centerline of the torus and constant in modulus.

Real geometry We consider the real morphology reported in Fig. 5.1. The convective field $\boldsymbol{\beta}$ is given by the solution represented there.

5.4.1 Numerical results

Cylinder Figure 5.8a shows the parallel running time as a function of p. The varying dependence of number of iterations and cost per iteration on p results in a convex behavior of the computational time. This behavior is expected, since for small p the high number of iterations dominates the cost, while beyond a certain value it does not decrease any longer, while the cost per processor increases. When compared with the similar tests presented in [99], we notice that the performances are not significantly affected by the presence of the convective field $\boldsymbol{\beta}$. Precisely, the optimal size of the overlap is achieved at the same percentage as the one obtained for the problem with $\boldsymbol{\beta} = \mathbf{0}$, around 25% and 15% for a fine and very fine mesh, respectively. In fact the simplicity of the domain makes the presence of the convective field not relevant for the DD iterations. Nevertheless, it is remarkable that the solution of the problem on the finest mesh (black line) takes fewer (or the same) iterations than those needed for a coarser grid (Figure 5.8b). Indeed, a higher concentration of nodes on the interfaces allows a more detailed exchange of information between the two partitions through the enforcement of the interface conditions and, consequently, it boosts the convergence speed by reducing the number of iterations.

Idealized Aneurysm Figure 5.9d shows that the curve related to the very fine mesh features a minimum at a fraction of overlap of $\sim 30\%$, like in the advection-free case [99], i.e., the trend of the estimated parallel time is still invariant with respect to the presence of the convective field.

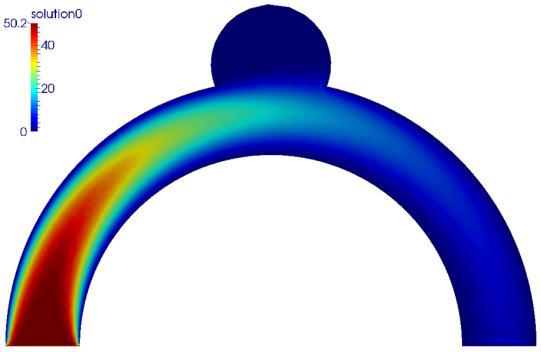


(a) Parallel time as a function of the overlap. (b) Number of iterations of the parallel solver.

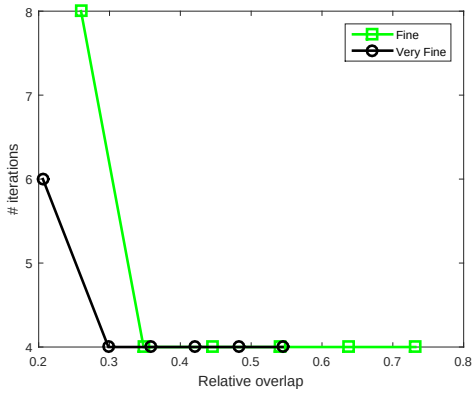
Figure 5.8: Parallel time performed as a function of p for two levels of refinement of the mesh for the solution of an ADR problem on a cylinder (a). Corresponding number of iterations for fine ($\square$) and very fine ($\circ$) meshes (b).

On the contrary, it is interesting to notice that if we do not have a physical convection ($\boldsymbol{\beta} = \mathbf{0}$) the minimum of the curve for a coarser grid happens at a larger size of the overlap (45% vs. 35%, see Figure 5.9c). Numerical performances are here explained by physical arguments. In fact the solute concentration u at the inflow is convected through the domain, determining a rapid exchange of information that accelerates the convergence by subdomains (see Figure 5.9a-5.9b).

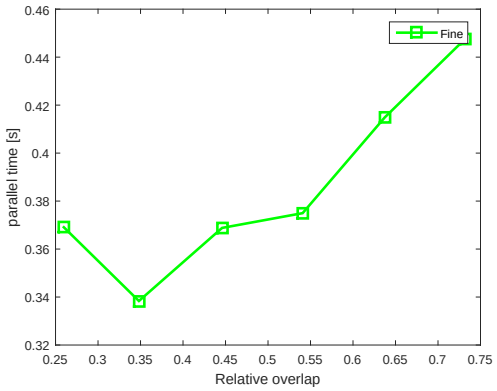
Real Aneurysm In this case the geometry is twisted and the convective field reproduces the real blood flow into the vessel. As we can see from Figures 5.10a-5.10b, the solute at the inflow section is convected through the vessel and it stagnates into the aneurysmatic sac. Table 5.4 shows the increment of degrees of freedom when the overlapping is extended. This number is proportional to the computational cost required for solving each subdomain. On the other hand, for the different partitions we have a different number of nodes at the interface, depending on the position of the two cuts. For instance, passing from 25% to 40% the number of nodes at the interface of partition 1 decreases. As the optimal trade-off between the reduction



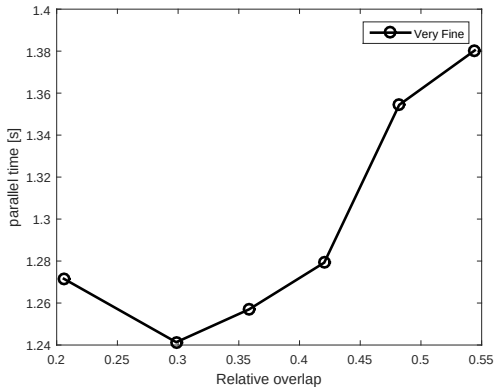
(a) Solution on a slice of the domain.



(b) Number of iterations for a fine (□) and very fine (○) mesh.

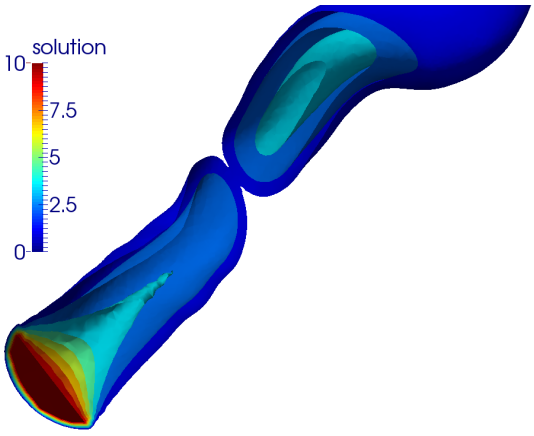


(c) Parallel time for a fine mesh.

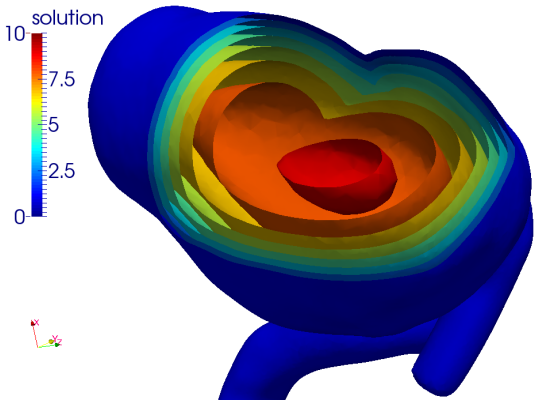


(d) Parallel time for a very fine mesh.

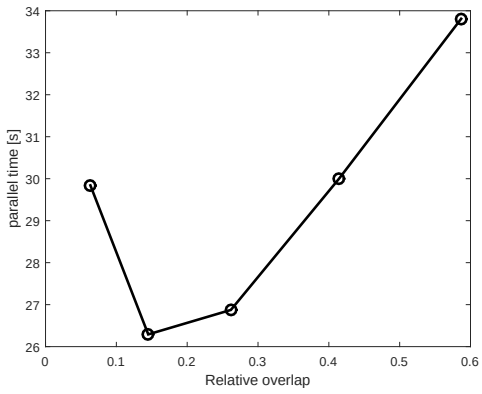
Figure 5.9: Solution to an ADR problem on an idealized aneurysm (a) and number of iterations (b). Parallel time performed as a function of p for a fine (c) and very fine (d) mesh.



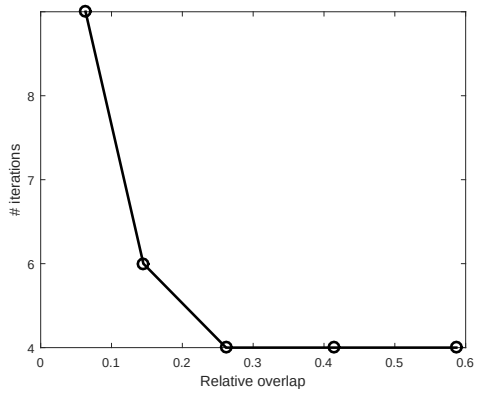
(a) Contour surfaces at inflow.



(b) Contour surfaces in the aneurysm.



(c) Parallel time as a function of the overlap.



(d) Number of iterations.

Figure 5.10: Solution to an ADR problem on a real aneurysmatic vessel (a-b). Parallel time performed as a function of p for a very fine mesh (c) and number of iterations (d).

% overlap	DoF0	DoF1	InterNodes0	InterNodes1
5%	86,265	85,096	2046	1491
15%	93,255	89,009	3358	1721
25%	104,465	93,791	5068	1947
40%	118,632	97,760	6161	1584
60%	135,228	101,729	6897	1646

Table 5.4: Number of nodes of each partition (DoF0, DoF1) and total number of nodes on the interfaces (InterNodes0, InterNodes1) for different levels of overlap on a very fine mesh for Test 5.

of the iterations attained by a larger overlapping (that however does not improve after 25%), the increased computational cost per subdomain and the communication cost, results indicates 15%. This actually yields a well balanced load for the two subdomains, and a total number of interface degrees of freedom of about 5,000, even if the total number of iterations is 6 vs the minimum of 4 reached with 25% overlapping or more (see Figure 5.10c-5.10d).

5.5 Conclusions

High Performance Computing (HPC) quantification of dynamics traditionally described more in empirical qualitative terms is expected to bring strong improvements for understanding and optimizing processes with a major impact on industry and society. A well established example is medicine and cardiovascular sciences in particular. Numerical analysis of patient-specific settings is becoming a consolidated tool for clinical routine. This allows to improve the level of knowledge available to medical doctors thanks to mathematical models and numerical tools that compute quantities difficult or impossible to measure and overall to enhance the reliability of measures.

However, the intrinsic complexity of the dynamics of interest – reflected by complicated systems of Partial Differential Equations – the constraining timelines of the clinical routine as well as the large volumes of patients typically needed by clinical trials rise formidable challenges in terms of computational resources. Traditional local clusters may be not adequate to afford the computational requests and alternative solutions like grid/cloud resources on demand need to be deeply evaluated. The performance evaluation of these resources is however much more complex for the variety of user needs and availability scenarios that may present. A general recipe for the identification of the optimal strategy is currently out of reach. Nevertheless, in this paper we aim at presenting the results of years of experience in a vital environment like Emory University, where mathematicians and computer scientists routinely assist medical doctors in their daily activity. We focused on a real problem, such as hemodynamics in patient-specific settings and presented extensive results on different platforms. We propose a way for measuring the performances under realistic scenarios. Comparing execution time and cost of the application on on-premise and on-demand targets, we found some evidence to support the claim that IaaS resources may be utilized for scientific CFD simulations possibly at lower cost than incurred locally. In particular, our test with Amazon's spot-request feature coupled with availability of cutting edge resources (16-core nodes, 60GB RAM) suggests that small on-demand assemblies may be a viable alternative to local clusters. It is crucial that IaaS's provide resources immediately while local and grid resources are often subject to long queue wait times – an aspect that might offset any additional expense. Furthermore, while a modern local computing cluster with an efficient interconnection network will outperform an on-demand assembly (which is highly vulnerable to network performance), the cloud solution might be useful when cost needs to be minimized.

Among clusters, grids and cloud platforms there is a tremendous variation in communication performance. In order to reduce the heterogeneity due to data handling and interconnection network capabilities, we analyzed performance variations and

process placement strategies for a parallel CFD application based on the finite element library LiFEV. The communication profile for this parallel application depends greatly on the partitioning of the mesh representing the physical geometry of the input. Such communication imbalance invites exploration of the possible mappings of the parallel tasks onto diversely performing networks of processors. As parallel target platforms universally present heterogeneous inter-process communication capabilities when nodes are multicore, performance advantages are possible through process placement that exploit this knowledge. We studied five process placement strategies: three of them use problem-related information and the others are typical OpenMPI process allocations. We found that the standard *pcore* placement mapping is well-suited for processing our CFD application. However, we showed that this allocation may be improved by our *part* placement, especially for larger numbers of hosts performing the computation. As a result, we showed that it is possible to adapt performance of parallel MPI applications by communication-aware placement of their MPI processes if the computation structure, input geometry as well as target architecture and network wiring is known and it may be done automatically by ADAPT.

As a complementary approach, we discuss in detail the optimal splitting of a problem of interest with different mathematical techniques. The introduction of overlap in the partition of problems featuring complex morphology gives more freedom in the optimal selection of interfaces and consequently may outperform more traditional nonoverlapping approaches. Different aspects have competitive dynamics resulting in a nontrivial optimization. The dependence of the number of iterations on the iterative-by-subdomain method decreases with the overlap, while the cost of the solution on each subdomain increases. The communication time depends on the location of the interface. Our results in realistic geometries point out the efficacy of an appropriate selection of overlapping to reduce costs in a parallel computing setting. In general, a small amount of overlap results in a good trade-off of all the competitive mechanisms affecting the total computational time. This shows that a

more thoughtful positioning of the interfaces can benefit the overall computing performance, at a small additional computational cost on each processing unit. This complements discussion of sections 5.2 and 5.3, highlighting the trade-offs that can be achieved on different types of parallel platforms.

This paper has highlighted three dimensions of hemodynamic simulations on clusters, grids and clouds, both algorithmic- and platform-specific. While no universal conclusions can be drawn, our work proposes indications to the identification of protocols for hemodynamics computations in outsourcing that we think are needed – and progressively will be more requested in the next future – by clinical applications. We plan to include overlapping partitions more extensively in the current activities to have a more solid experience on the identification of optimal location of the interfaces and in general of the workbalance.

Acknowledgments

The authors gratefully acknowledge support by the Computational and Life Sciences (CLS) Strategic Initiative of Emory University. This research was supported in part by US National Science Foundation Grant OCI-1124418, OCI-1053575, DMS-1419060, DMS-1412963, Fondazione Cariplo "iCardioCloud," URC Emory Grant 2015 "Numerical methods and flows at moderate reynolds number in left ventricular assisted devices."

Chapter 6

Conclusions

This work has been motivated by the vision of Mathematics as a tool to empower and support clinicians in diagnosis/prognosis, surgical planning, and virtual surgery. For this to happen, several challenges need to be tackled, such as the large number of patients that need to be processed, the uncertainty in the numerical results due to missing patient-specific data and model assumptions, and privacy and efficiency concerns when scientists and clinicians outsource *in silico* experiments to reduce system and operating expenses of local computational resources. We addressed these issues by employing a variety of mathematical and computational techniques – parallel computing, reduced order modelling, and uncertainty quantification among others - designed for but not limited to medical applications.

To reduce the computational costs, an educated reduced model (HiMod) for patient-specific geometries can be tuned to be either as accurate as 3D high-fidelity models, or as computationally cheap as reduced 1D models. We resorted to a genuine polar coordinate spectral discretization, which seems more natural for domains like the ones occurring in computational hemodynamics. The selection of appropriate basis functions is troublesome both for the scalar case, and - even more - for the vector case, where additional analytical, numerical, and physical constraints need to be met. Although, from a mathematical viewpoint, guaranteeing desirable properties such as well-posedness, well-conditioning, orthogonality, regularity, and the fulfillment of boundary conditions simultaneously is extremely challenging, from a practical viewpoint it can be unnecessary. For instance, a discontinuity in the solution localized

around the pole may be irrelevant if the goal is approximating the dynamics in proximity of the walls efficiently. This confirms the hierarchical model reduction as a competitive method with great potential to radically change the clinical practice by making *quantitative* data securely and quickly accessible to the clinicians for more informed decisions, thus reducing in a broader perspective unnecessary hospitalizations.

To quantify the uncertainty at an affordable computational cost, we proposed a domain decomposition approach that solves the stochastic problem only locally, at the subsystem level, significantly fostering parallelism and enabling accurate and relatively inexpensive stochastic solves in large-scale networks. The numerical tests showed excellent strong and weak scalability properties for different types of solvers, and bounded errors regardless of the network size (i.e., truncation errors do not grow uncontrollably with the problem size). In order to improve the performance of the solver, we designed new multigrid methods tailored to network problems. While coarsening the PCE polynomial order does not provide significant improvements, combining network components to create coarser networks considerably accelerates the software convergence, even more than the traditional algebraic multigrid method.

To improve the accuracy of uncertainty quantification in the cardiovascular network at an affordable computational cost, we combined a Cartesian version of the HiMod solver (TEPEM) with the DDUQ method. While the traditional methods that can be found in the literature for quantifying uncertainties in large-scale problems suffer from the intensive computational cost associated with high-fidelity models, or from inaccuracy due to the reduced 1D models, DDUQ and TEPEM feature the perfect capabilities to "fill the gap" between feasibility and reliability: while, on one hand, the DDUQ method provides an effective way to quantify the model uncertainties by promoting the independence of the subsystems, on the other hand, the TEPEM provides an effective way to solve the local problems maintaining accuracy and reduced computational burden. By embedding the TEPEM solver into the DDUQ framework, we were able to show excellent scaling for 3D non-linear vector problems, and

to provide relevant statistical information about quantities of clinical interest, such as the wall shear stress, with a level of detail currently out of reach for the traditional one-dimensional models, and in a fraction of the computational time demanded by full three-dimensional approaches.

To optimize the execution on parallel architectures, we analyzed performance variations and process placement strategies for parallel CFD applications, that are consistently affected by communication imbalance. We showed that it is possible to adapt performance of parallel MPI applications by communication-aware placement of the MPI processes if the computation structure, input geometry, target architecture, and network wiring is known. Moreover, we compared different hardware platforms with respect to the execution of the same task, based on time to completion, monetary cost, and user's priorities. From a mathematical viewpoint, we detect the optimal domain splitting that minimizes the computational time via domain decomposition techniques. Our results in realistic geometries show that a small amount of overlap provides a good trade-off of all the competitive mechanisms affecting the total computational time (number of iterations to convergence vs. local computational cost).

As future developments, we would like to incorporate fluid-structure interaction models in the hierarchical reduced-order solvers, for a more realistic representation of the physics of the phenomenon, which will require the enforcement of non-Dirichlet boundary conditions. From an analytical viewpoint, the fulfillment of the inf-sup condition in the reduced spaces needs to be rigorously proved in a general framework, while from an algebraic viewpoint, the design of *ad-hoc* preconditioners based on the hierarchical structure of the HiMod matrix is still to be investigated. The computation of quantities of clinical interest, such as the fractional flow reserve (FFR), will need to be endowed with confidence intervals in order to make the numerical results more reliable and usable by clinicians.

Appendix

7.1 Bottom-Up basis functions

Possible choices for the basis functions different from Zernike and parity-restricted Chebyshev polynomials according to a bottom-up approach are detailed here, with a short motivation about their discard. Figure 7.1 (left) shows the performance of different basis sets in approximating the function $f(\hat{r}) = \cos(\frac{\pi}{2}\hat{r})$ on the interval $(0, 1)$. The relative error is defined as $e = \|f - f_{approx}\|_{L^2_w(\Omega)} / \|f\|_{L^2_w(\Omega)}$, where $w(\hat{r}) = \hat{r}$ and f_{approx} denotes the approximation of f via the truncation of (2.9).

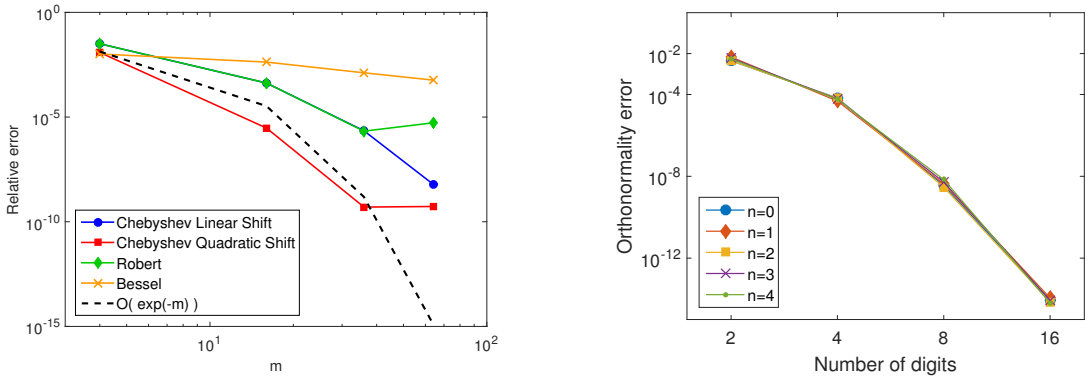


Figure 7.1: Left: Relative error for the approximation of the function $f(\hat{r}) = \cos\left(\frac{\pi}{2}\hat{r}\right)$ with Chebyshev polynomials with linear (●) and quadratic (■) shift, Robert (⧫) and Bessel (×) functions. An exponential decay is shown by the dashed line. Right: Orthonormality error associated with the basis functions $J_n(\lambda_j \hat{r})$, for $n = 0, 1, \ldots, 4$, $j = 1, 2, \ldots, 8$, as a function of the numerical precision expressed in number of digits.

Bessel functions: $J_n(\hat{\lambda}_j\hat{r})$. The modal coefficients of Bessel series asymptotically behave like $1/j^3$ [89, 36]. For this reason Bessel functions are expected to be a bad choice for function approximation. In particular, Figure 7.1 (left) shows that even a large number of modes is unable to guarantee a desired accuracy on the relative error. Moreover, this basis is highly sensitive to numerical precision. Figure 7.1 (right) shows that the basis functions lose orthonormality as the accuracy of the roots $\hat{\lambda}_j$ decreases. The orthonormality error associated with the basis functions $J_n(\hat{\lambda}_j\hat{r})$, for $n = 0, 1, \ldots, 4$, $j = 1, 2, \ldots, 8$, is measured via the Frobenius norm of the matrix $M_n - I$, where M_n is the mass matrix associated with the normalized Bessel functions of order n, with components $[M_n]_{jl} = \|J_n(\hat{\lambda}_j\hat{r})\|^{-1}_{L^2_w(0,1)}\|J_n(\hat{\lambda}_l\hat{r})\|^{-1}_{L^2_w(0,1)} \int_0^1 J_n(\hat{\lambda}_j\hat{r})J_n(\hat{\lambda}_l\hat{r})\hat{r}d\hat{r}$, and I is the identity matrix.

Polar Robert functions: $\hat{r}^j T_n(\hat{r})$. As addressed in [35, 37], this basis is extremely ill-conditioned. Indeed, in proximity of $\hat{r} = 1$, the linear independence of the low-degree basis functions is compromised, since the variation of these functions in this region is so slow that they are asymptotically equivalent [37]. These features make the polar Robert functions a basis unsuitable in most cases.

Shifted-Chebyshev polynomials with linear shift: $T_n(2\hat{r} - 1)$. The grid $\{\hat{r}_i\}_{i=0}^N$ constituted by the roots of the Shifted-Chebyshev polynomial of order $(N+1)$ has points clustered near both $\hat{r} = 0$ and $\hat{r} = 1$. This property makes this grid ideal to solve large gradients near the origin, but less suited to a generic dependence of the function at hand on the radial coordinate.

7.2 HiMod coefficients for the Advection-Diffusion-Reaction Equations

We denote the inverse transpose of the strain gradient tensor by

$$\hat{\mathbb{F}}^{-T} = \frac{\partial \hat{\boldsymbol{\psi}}}{\partial \mathbf{z}} \circ \hat{\boldsymbol{\psi}}^{-1} = \begin{bmatrix} 1 & -\dfrac{\hat{r}}{R}\dfrac{\partial R}{\partial \hat{x}} & 0 \\ 0 & \dfrac{1}{R} & 0 \\ 0 & -\dfrac{1}{R^2}\dfrac{\partial R}{\partial \hat{\vartheta}} & \dfrac{1}{R} \end{bmatrix} = \begin{bmatrix} 1 & \hat{D}_r & \hat{D}_\vartheta \\ 0 & \hat{J}_r & \hat{D}_{r\vartheta} \\ 0 & \hat{D}_{\vartheta r} & \hat{J}_\vartheta \end{bmatrix}, \tag{7.1}$$

where the subscripts r and ϑ refer to the radial and angular component, while $\hat{r}$ and $\hat{\vartheta}$ denote the polar coordinates in the reference domain. Notice that, if we assume the physical radius R to be constant, matrix (7.1) features a diagonal pattern, being $\hat{D}_r = \hat{D}_\vartheta = \hat{D}_{r\vartheta} = \hat{D}_{\vartheta r} = 0$ and $\hat{J}_r = \hat{J}_\vartheta = R^{-1}$.

The coefficients of the HiMod formulation (2.13) for the ADR problem (2.11) with viscosity μ, convective field $\mathbf{b} = [b_x, b_r, b_\vartheta]^T$, reaction coefficient σ and forcing term f are given by

$$\begin{aligned}
\mathfrak{a}_{kj} = \int_{\hat{\gamma}} \Bigg(& \mu(\hat{D}_r^2 + \hat{D}_{\vartheta r}^2 + \hat{J}_r^2)\frac{\partial \hat{\varphi}_k}{\partial \hat{r}}\frac{\partial \hat{\varphi}_j}{\partial \hat{r}} + \mu(\hat{D}_r\hat{D}_\vartheta + \hat{J}_r\hat{D}_{r\vartheta} + \hat{D}_{\vartheta r}\hat{J}_\vartheta)\frac{1}{\hat{r}}\frac{\partial \hat{\varphi}_k}{\partial \hat{r}}\frac{\partial \hat{\varphi}_j}{\partial \hat{\vartheta}} \\
& + \mu(\hat{D}_r\hat{D}_\vartheta + \hat{D}_{r\vartheta}\hat{J}_r + \hat{J}_\vartheta\hat{D}_{\vartheta r})\frac{1}{\hat{r}}\frac{\partial \hat{\varphi}_k}{\partial \hat{\vartheta}}\frac{\partial \hat{\varphi}_j}{\partial \hat{r}} + \mu\left(\hat{D}_\vartheta^2 + \hat{D}_{r\vartheta}^2 + \hat{J}_\vartheta^2\right)\frac{1}{\hat{r}^2}\frac{\partial \hat{\varphi}_k}{\partial \hat{\vartheta}}\frac{\partial \hat{\varphi}_j}{\partial \hat{\vartheta}} \\
& + (b_x\hat{D}_r + b_r\hat{J}_r + b_\vartheta\hat{D}_{\vartheta r})\frac{\partial \hat{\varphi}_k}{\partial \hat{r}}\hat{\varphi}_j + (b_x\hat{D}_\vartheta + b_r\hat{D}_{r\vartheta} + b_\vartheta\hat{J}_\vartheta)\frac{1}{\hat{r}}\frac{\partial \hat{\varphi}_k}{\partial \hat{\vartheta}}\hat{\varphi}_j + \sigma\hat{\varphi}_k\hat{\varphi}_j\Bigg)\hat{J}d\hat{\gamma}, \\
\mathfrak{b}_{kj} = \int_{\hat{\gamma}} \mu\left(\hat{D}_r\frac{\partial \hat{\varphi}_k}{\partial \hat{r}}\hat{\varphi}_j + \frac{\hat{D}_\vartheta}{\hat{r}}\frac{\partial \hat{\varphi}_k}{\partial \hat{\vartheta}}\hat{\varphi}_j\right)\hat{J}d\hat{\gamma}, \qquad & \mathfrak{d}_{kj} = \int_{\hat{\gamma}} \mu\hat{\varphi}_k\hat{\varphi}_j\hat{J}d\hat{\gamma}, \\
\mathfrak{c}_{kj} = \int_{\hat{\gamma}} \left(\mu\hat{D}_r\hat{\varphi}_k\frac{\partial \hat{\varphi}_j}{\partial \hat{r}} + \mu\frac{\hat{D}_\vartheta}{\hat{r}}\hat{\varphi}_k\frac{\partial \hat{\varphi}_j}{\partial \hat{\vartheta}} + b_x\hat{\varphi}_k\hat{\varphi}_j\right)\hat{J}d\hat{\gamma}, \qquad & \mathfrak{f}_j = \int_{\hat{\gamma}} f(\hat{\boldsymbol{\psi}}^{-1}(\hat{z}))\hat{\varphi}_j\hat{J}d\hat{\gamma},
\end{aligned}$$

being $\hat{J} = |\det(\hat{\mathbb{F}}^{-T})|$, and with $d\hat{\gamma} = \hat{r}d\hat{r}d\hat{\vartheta}$. The diagonal pattern of matrix $\hat{\mathbb{F}}^{-T}$

for the cylindrical setting with a constant radius yields some simplifications, so that

$$\mathfrak{a}_{kj} = \int_{\hat{\gamma}} \left(\frac{\mu}{R^2} \frac{\partial \hat{\varphi}_k}{\partial \hat{r}} \frac{\partial \hat{\varphi}_j}{\partial \hat{r}} + \frac{\mu}{R^2 \hat{r}^2} \frac{\partial \hat{\varphi}_k}{\partial \hat{\vartheta}} \frac{\partial \hat{\varphi}_j}{\partial \hat{\vartheta}} + \frac{b_r}{R} \frac{\partial \hat{\varphi}_k}{\partial \hat{r}} \hat{\varphi}_j + \frac{b_\vartheta}{R\hat{r}} \frac{\partial \hat{\varphi}_k}{\partial \hat{\vartheta}} \hat{\varphi}_j + \sigma \hat{\varphi}_k \hat{\varphi}_j \right) \hat{J} d\hat{\gamma},$$

$$\mathfrak{b}_{kj} = 0, \quad \mathfrak{c}_{kj} = \int_{\hat{\gamma}} b_x \hat{\varphi}_k \hat{\varphi}_j \hat{J} d\hat{\gamma}, \quad \mathfrak{d}_{kj} = \int_{\hat{\gamma}} \mu \hat{\varphi}_k \hat{\varphi}_j \hat{J} d\hat{\gamma}, \quad \mathfrak{f}_j = \int_{\hat{\gamma}} f(\hat{\boldsymbol{\psi}}^{-1}(\hat{z})) \hat{\varphi}_j \hat{J} d\hat{\gamma}.$$

7.3 HiMod coefficients for the Navier-Stokes equations

The HiMod formulation of the Navier-Stokes equations (2.17) reads as:

For all $k = 1, \dots, m_u$, $w = 1, \dots, m_p$, $i = 1, \dots, N_{h,u}$, $q = 1, \dots, N_{h,p}$, find $u_{x,k,i}$, $u_{r,k,i}$, $u_{\vartheta,k,i}$, $p_{w,q}$ such that, $\forall b \in \{x, r, \vartheta, p\}$, $\forall j = 1, \dots, \{m_u, m_p\}$, $\forall l = 1, \dots, \{N_{h,u}, N_{h,p}\}$[2]

$$\sum_{a \in \{x,r,\vartheta\}} \sum_{k=1}^{m_u} \sum_{w=1}^{m_p} \int_{\hat{\Omega}_{1D}} \Bigg\{ \sum_{i=1}^{N_{h,u}} \left[\mathfrak{a}_{ab,kj} \zeta_{a,i} \zeta_{b,l} + \mathfrak{b}_{ab,kj} \zeta_{a,i} \zeta'_{b,l} + \mathfrak{c}_{ab,kj} \zeta'_{a,i} \zeta_{b,l} + \mathfrak{d}_{ab,kj} \zeta'_{a,i} \zeta'_{b,l} \right] u_{a,k,i} + \sum_{q=1}^{N_{h,p}} \left[\mathfrak{a}_{pb,wj} \zeta_{p,q} \zeta_{b,l} + \mathfrak{b}_{pb,wj} \zeta_{p,q} \zeta'_{b,l} \right] p_{w,q} \Bigg\} d\hat{x} = \int_{\hat{\Omega}_{1D}} \int_{\hat{\gamma}} \hat{f} \hat{\varphi}_{b,j} \zeta_{b,l} \hat{J} d\hat{\gamma} d\hat{x}, \tag{7.2}$$

where $\hat{J} = |\det(\hat{\mathbb{F}}^{-T})|$, $d\hat{\gamma} = \hat{r} d\hat{r} d\hat{\vartheta}$, and the coefficients $\mathfrak{a}_{ab,kj}$, $\mathfrak{a}_{pb,wj}$, $\mathfrak{b}_{ab,kj}$, $\mathfrak{b}_{pb,wj}$, $\mathfrak{c}_{ab,kj}$, $\mathfrak{d}_{ab,kj}$ collect the contribution of the transverse dynamics.

For the sake of simplicity, the following notation is adopted:

$$\mathscr{J}_{ab,cd}^{\alpha\beta,\gamma\delta}(f_1, f_2, \dots; \eta_1, \eta_2, \dots) = \int_{\hat{\gamma}} f_1(\hat{J}_r, \hat{J}_\vartheta, \hat{D}_r, \hat{D}_{\vartheta r}, b_x, b_r, b_\vartheta) f_2(\hat{J}_r, \hat{J}_\vartheta, \hat{D}_r, \hat{D}_{\vartheta r}, b_x, b_r, b_\vartheta) \dots \dots \varphi_{ac}{}^{(\alpha,\beta)}(\hat{r}, \hat{\vartheta}) \varphi_{bd}{}^{(\gamma,\delta)}(\hat{r}, \hat{\vartheta}) \eta_1 \eta_2 \dots \hat{J} d\hat{\gamma}, \tag{7.3}$$

[2] If $b = p$, the indices j and l run up to m_p and $N_{h,p}$, respectively, and up to m_u and $N_{h,u}$ otherwise.

where $a, b \in \{x, r, \vartheta; p\}$ refer to the axial, radial, angular component of the modal function or to the pressure modal basis, and $c, d \in \{1, \ldots, m_u \text{ or } m_p\}$ are the corresponding modal indices. The superscripts $\{\alpha, \beta, \gamma, \delta\} \in \{0, 1\}$ take into account the differentiation applied to the modal basis. In particular, when α or γ (β or δ) are set to 1, the corresponding modal function is differentiated with respect to the radial (angular) variable. Finally, f_i is a function of $\hat{x}, \hat{r}, \hat{\vartheta}$ through $\hat{J}_r, \hat{J}_\vartheta, \hat{D}_r, \hat{D}_{\vartheta r}, b_x, b_r, b_\vartheta$, being $\{b_x, b_r, b_\vartheta\}$ the terms coming from the linearization of the non-linear term, and $\eta_i \in \{\alpha, \nu, 2\nu, \pm 1\}$ are constant parameters. For instance,

$$\mathfrak{a}_{\vartheta x,sj} = \mathscr{I}^{10,01}_{\vartheta x,sj}\left(\hat{D}_r, \frac{\hat{J}_\vartheta}{\hat{r}}; \nu\right) = \int_{\hat{\gamma}} \left(\nu \hat{D}_r \frac{\hat{J}_\vartheta}{\hat{r}} {\varphi_{\vartheta s}}^{(1,0)}(\hat{r}, \hat{\vartheta}) {\varphi_{xj}}^{(0,1)}(\hat{r}, \hat{\vartheta})\right) \hat{J} d\hat{\gamma}.$$

The explicit expression for all the coefficients is provided below.

$$\begin{aligned}
\mathfrak{a}_{xx,kj} = \;& \mathscr{I}^{00,00}_{xx,kj}(1;\alpha) + \mathscr{I}^{01,00}_{xx,kj}\left(\frac{\hat{J}_\vartheta}{r}, b_\vartheta; 1\right) + \mathscr{I}^{01,01}_{xx,kj}\left(\frac{\hat{J}^2_\vartheta}{\hat{r}^2}; \nu\right) + \mathscr{I}^{01,10}_{xx,kj}\left(\frac{\hat{J}_\vartheta}{r}, \hat{D}_{\vartheta r}; \nu\right) + \\
& \mathscr{I}^{10,00}_{xx,kj}\left(\hat{D}_{\vartheta r}, b_\vartheta; 1\right) + \mathscr{I}^{10,00}_{xx,kj}\left(\hat{J}_r, b_r; 1\right) + \mathscr{I}^{10,00}_{xx,kj}\left(\hat{D}_r, b_x; 1\right) + \\
& \mathscr{I}^{10,01}_{xx,kj}\left(\frac{\hat{J}_\vartheta}{r}, \hat{D}_{\vartheta r}; \nu\right) + \mathscr{I}^{10,10}_{xx,kj}\left(\hat{D}^2_r; 2\nu\right) + \mathscr{I}^{10,10}_{xx,kj}\left(\hat{J}^2_r; \nu\right) + \mathscr{I}^{10,10}_{xx,kj}\left(\hat{D}^2_{\vartheta r}; \nu\right), \\
\mathfrak{a}_{xr,kl} = \;& \mathscr{I}^{10,10}_{xr,kl}\left(\hat{D}_r, \hat{J}_r; \nu\right), \qquad \mathfrak{a}_{x\vartheta,kz} = \mathscr{I}^{01,10}_{x\vartheta,kz}\left(\hat{D}_r, \frac{\hat{J}_\vartheta}{\hat{r}}; \nu\right) + \mathscr{I}^{10,10}_{x\vartheta,kz}\left(\hat{D}_r, \hat{D}_{\vartheta r}; \nu\right), \\
\mathfrak{a}_{rr,hl} = \;& \mathscr{I}^{00,00}_{rr,hl}(1;\alpha) + \mathscr{I}^{00,00}_{rr,hl}\left(\frac{\hat{J}^2_\vartheta}{\hat{r}^2}; 2\nu\right) + \mathscr{I}^{01,00}_{rr,hl}\left(\frac{\hat{J}_\vartheta}{r}, b_\vartheta; 1\right) + \mathscr{I}^{01,01}_{rr,hl}\left(\frac{\hat{J}^2_\vartheta}{\hat{r}^2}; \nu\right) + \\
& \mathscr{I}^{01,10}_{rr,hl}\left(\frac{\hat{J}_\vartheta}{r}, \hat{D}_{\vartheta r}; \nu\right) + \mathscr{I}^{10,00}_{rr,hl}\left(\hat{D}_r, b_x; 1\right) + \mathscr{I}^{10,00}_{rr,hl}\left(\hat{J}_r, b_r; 1\right) + \\
& \mathscr{I}^{10,00}_{rr,hl}\left(\hat{D}_{\vartheta r}, b_\vartheta; 1\right) + \mathscr{I}^{10,01}_{rr,hl}\left(\frac{\hat{J}_\vartheta}{r}, \hat{D}_{\vartheta r}; \nu\right) + \mathscr{I}^{10,10}_{rr,hl}\left(\hat{D}^2_r; \nu\right) + \\
& \mathscr{I}^{10,10}_{rr,hl}\left(\hat{D}^2_{\vartheta r}; \nu\right) + \mathscr{I}^{10,10}_{rr,hl}\left(\hat{J}^2_r; 2\nu\right),
\end{aligned}$$

$$
\begin{aligned}
\mathfrak{a}_{r\vartheta,hz} = \;& \mathscr{I}^{00,00}_{r\vartheta,hz}\left(\frac{\hat{J}_\vartheta}{r}, b_\vartheta; 1\right) + \mathscr{I}^{00,01}_{r\vartheta,hz}\left(\frac{\hat{J}^2_\vartheta}{\hat{r}^2}; 2\nu\right) + \mathscr{I}^{00,10}_{r\vartheta,hz}\left(\frac{\hat{J}_\vartheta}{\hat{r}}, \hat{D}_{\vartheta r}; 2\nu\right) + \\
& \mathscr{I}^{01,00}_{r\vartheta,hz}\left(\frac{\hat{J}^2_\vartheta}{\hat{r}^2}; -\nu\right) + \mathscr{I}^{01,10}_{r\vartheta,hz}\left(\hat{J}_r, \frac{\hat{J}_\vartheta}{\hat{r}}; \nu\right) + \\
& \mathscr{I}^{10,00}_{r\vartheta,hz}\left(\frac{\hat{J}_\vartheta}{\hat{r}}, \hat{D}_{\vartheta r}; -\nu\right) + \mathscr{I}^{10,10}_{r\vartheta,hz}\left(\hat{J}_r, \hat{D}_{\vartheta r}; \nu\right), \\
\mathfrak{a}_{rx,hj} = \;& \mathscr{I}^{10,10}_{rx,hj}\left(\hat{J}_r, \hat{D}_r; \nu\right), \qquad \mathfrak{a}_{rp,hi} = \mathscr{I}^{00,00}_{rp,hi}\left(-\frac{\hat{J}_\vartheta}{\hat{r}}; 1\right) + \mathscr{I}^{10,00}_{rp,hi}\left(-\hat{J}_r; 1\right), \\
\mathfrak{a}_{\vartheta\vartheta,sz} = \;& \mathscr{I}^{00,00}_{\vartheta\vartheta,sz}(1; \alpha) + \mathscr{I}^{00,00}_{\vartheta\vartheta,sz}\left(\frac{\hat{J}^2_\vartheta}{\hat{r}^2}; \nu\right) + \mathscr{I}^{00,10}_{\vartheta\vartheta,sz}\left(\hat{J}_r, \frac{\hat{J}_\vartheta}{\hat{r}}; -\nu\right) + \mathscr{I}^{01,00}_{\vartheta\vartheta,sz}\left(\frac{\hat{J}_\vartheta}{r}, b_\vartheta; 1\right) + \\
& \mathscr{I}^{01,01}_{\vartheta\vartheta,sz}\left(\frac{\hat{J}^2_\vartheta}{\hat{r}^2}; 2\nu\right) + \mathscr{I}^{01,10}_{\vartheta\vartheta,sz}\left(\frac{\hat{J}_\vartheta}{\hat{r}}, \hat{D}_{\vartheta r}; 2\nu\right) + \mathscr{I}^{10,00}_{\vartheta\vartheta,sz}\left(\hat{J}_r, \frac{\hat{J}_\vartheta}{\hat{r}}; -\nu\right) + \\
& \mathscr{I}^{10,00}_{\vartheta\vartheta,sz}\left(\hat{D}_r, b_x; 1\right) + \mathscr{I}^{10,00}_{\vartheta\vartheta,sz}\left(\hat{J}_r, b_r; 1\right) + \mathscr{I}^{10,00}_{\vartheta\vartheta,sz}\left(\hat{D}_{\vartheta r}, b_\vartheta; 1\right) + \\
& \mathscr{I}^{10,10}_{\vartheta\vartheta,sz}\left(\hat{D}^2_r; \nu\right) + \mathscr{I}^{10,10}_{\vartheta\vartheta,sz}\left(\hat{J}^2_r; \nu\right) + \mathscr{I}^{10,10}_{\vartheta\vartheta,sz}\left(\hat{D}^2_{\vartheta r}; 2\nu\right), \\
\mathfrak{a}_{\vartheta r,sl} = \;& \mathscr{I}^{00,00}_{\vartheta r,sl}\left(\frac{\hat{J}_\vartheta}{r}, b_\vartheta; -1\right) + \mathscr{I}^{00,01}_{\vartheta r,sl}\left(\frac{\hat{J}^2_\vartheta}{\hat{r}^2}; -\nu\right) + \mathscr{I}^{00,10}_{\vartheta r,sl}\left(\frac{\hat{J}_\vartheta}{r}, \hat{D}_{\vartheta r}; -\nu\right) + \\
& \mathscr{I}^{01,00}_{\vartheta r,sl}\left(\frac{\hat{J}^2_\vartheta}{\hat{r}^2}; 2\nu\right) + \mathscr{I}^{10,00}_{\vartheta r,sl}\left(\frac{\hat{J}_\vartheta}{r}, \hat{D}_{\vartheta r}; 2\nu\right) + \\
& \mathscr{I}^{10,01}_{\vartheta r,sl}\left(\hat{J}_r, \frac{\hat{J}_\vartheta}{\hat{r}}; \nu\right) + \mathscr{I}^{10,10}_{\vartheta r,sl}\left(\hat{J}_r, \hat{D}_{\vartheta r}; \nu\right), \\
\mathfrak{a}_{\vartheta x,sj} = \;& \mathscr{I}^{10,01}_{\vartheta x,sj}\left(\hat{D}_r, \frac{\hat{J}_\vartheta}{\hat{r}}; \nu\right) + \mathscr{I}^{10,10}_{\vartheta x,sj}\left(\hat{D}_{\vartheta r}, \hat{D}_r; \nu\right), \\
\mathfrak{a}_{p\vartheta,wz} = \;& \mathscr{I}^{00,01}_{p\vartheta,wz}\left(\frac{\hat{J}_\vartheta}{\hat{r}}; -1\right) + \mathscr{I}^{00,10}_{p\vartheta,wz}\left(\hat{D}_{\vartheta r}; -1\right), \\
\mathfrak{a}_{pr,wl} = \;& \mathscr{I}^{00,10}_{pr,wl}\left(\hat{J}_r; -1\right) + \mathscr{I}^{00,00}_{pr,wl}\left(\frac{\hat{J}_\vartheta}{\hat{r}}; -1\right), \qquad \mathfrak{a}_{xp,ki} = \mathscr{I}^{10,00}_{xp,ki}\left(\hat{D}_r; -1\right), \\
\mathfrak{a}_{px,wj} = \;& \mathscr{I}^{00,10}_{px,wj}\left(\hat{D}_r; -1\right), \qquad \mathfrak{a}_{\vartheta p,si} = \mathscr{I}^{01,00}_{\vartheta p,si}\left(\frac{\hat{J}_\vartheta}{\hat{r}}; -1\right) + \mathscr{I}^{10,00}_{\vartheta p,si}\left(\frac{\hat{D}_{\vartheta r}}{\hat{r}}; -1\right);
\end{aligned}
$$

$$\mathfrak{b}_{xx,kj} = \mathscr{I}^{10,00}_{xx,kj}\left(\hat{D}_r; 2\nu\right), \quad \mathfrak{b}_{xr,kl} = \mathscr{I}^{10,00}_{xr,kl}\left(\hat{J}_r; \nu\right), \quad \mathfrak{b}_{px,wj} = \mathscr{I}^{00,00}_{px,wj}\left(1; -1\right),$$
$$\mathfrak{b}_{xp,ki} = \mathscr{I}^{00,00}_{xp,ki}\left(1; -1\right), \qquad \mathfrak{b}_{rr,hl} = \mathscr{I}^{10,00}_{rr,hl}\left(\hat{D}_r; \nu\right), \quad \mathfrak{b}_{\vartheta\vartheta,sz} = \mathscr{I}^{10,00}_{\vartheta\vartheta,sz}\left(\hat{D}_r; \nu\right),$$
$$\mathfrak{b}_{x\vartheta,kz} = \mathscr{I}^{01,00}_{x\vartheta,kz}\left(\frac{\hat{J}_\vartheta}{\hat{r}}; \nu\right) + \mathscr{I}^{10,00}_{x\vartheta,kz}\left(\hat{D}_{\vartheta r}; \nu\right);$$

$$\mathfrak{c}_{xx,kj} = \mathscr{I}^{00,00}_{xx,kj}\left(b_x; 1\right) + \mathscr{I}^{00,10}_{xx,kj}\left(\hat{D}_r; 2\nu\right), \qquad \mathfrak{c}_{rr,rl} = \mathscr{I}^{00,00}_{rr,hl}\left(b_x; 1\right) + \mathscr{I}^{00,10}_{rr,hl}\left(\hat{D}_r; \nu\right),$$
$$\mathfrak{c}_{\vartheta\vartheta,sz} = \mathscr{I}^{00,00}_{\vartheta\vartheta,sz}\left(b_x; 1\right) + \mathscr{I}^{00,10}_{\vartheta\vartheta,sz}\left(\hat{D}_r; \nu\right), \qquad \mathfrak{c}_{rx,hj} = \mathscr{I}^{00,10}_{rx,hj}\left(\hat{J}_r; \nu\right),$$
$$\mathfrak{c}_{\vartheta x,sj} = \mathscr{I}^{00,01}_{\vartheta x,sj}\left(\frac{\hat{J}_\vartheta}{\hat{r}}; \nu\right) + \mathscr{I}^{00,10}_{\vartheta x,sj}\left(\hat{D}_{\vartheta r}; \nu\right),$$
$$\mathfrak{d}_{rr,hl} = \mathscr{I}^{00,00}_{rr,hl}\left(1; \nu\right), \quad \mathfrak{d}_{\vartheta\vartheta,sz} = \mathscr{I}^{00,00}_{\vartheta\vartheta,sz}\left(1; \nu\right), \quad \mathfrak{d}_{xx,kj} = \mathscr{I}^{00,00}_{xx,kj}\left(1; 2\nu\right).$$

Bibliography

[1] HeartFlow. https://www.heartflow.com/.

[2] Boost C++ Libraries. http://www.boost.org, 2012.

[3] HPC Cloud Service, Penguin Computing. http://www.penguincomputing.com/Services/HPCCloud, 2012.

[4] LifeV Project. http://www.lifev.org, 2012.

[5] Penguin Computing On Demand / Indiana University. https://podiu.penguincomputing.com/, 2012.

[6] SuiteSparse. http://www.cise.ufl.edu/research/sparse/SuiteSparse/, 2012.

[7] Performance Analysis and Visualization at Exascale (PAVE). https://computation.llnl.gov/project/performance-analysis-through-visualization/, 2014.

[8] HiMod Project. http://himod.mathcs.emory.edu/, 2017.

[9] M Aletti, S Perotto, and A Veneziani. Educated bases for the HiMod reduction of advection-diffusion-reaction problems with general boundary conditions. Technical report, MOX Report No. 37/2015.

[10] MS Allen and RJ Kuether. Substructuring with nonlinear subcomponents: a nonlinear normal mode perspective. In *Topics in Experimental Dynamics Substructuring and Wind Turbine Dynamics, Volume 2*, pages 109–121. Springer, 2012.

[11] S Amaral, D Allaire, and K Willcox. A decomposition-based approach to uncertainty analysis of feed-forward multicomponent systems. *International Journal for Numerical Methods in Engineering*, 100(13):982–1005, 2014.

[12] DG Anderson. Iterative procedures for nonlinear integral equations. *Journal of the ACM (JACM)*, 12(4):547–560, 1965.

[13] L Antiga, T Passerini, M Piccinelli, and A Veneziani. Aneurisk web, ecm2.mathcs.emory.edu/aneuriskweb, 2011.

[14] L Antiga, M Piccinelli, L Botti, B Ene-Iordache, A Remuzzi, and DA Steinman. An image-based modeling framework for patient-specific computational hemodynamics. *Medical & biological engineering & computing*, 46(11):1097, 2008.

[15] L Antiga and DA Steinman. Vmtk: vascular modeling toolkit. *VMTK, San Francisco, CA, accessed Apr*, 27:2015, 2006.

[16] C Anukal and M Sankaran. First-order approximation methods in reliability-based design optimization. *Journal of Mechanical Design*, 127(5):851–857, 2005.

[17] M Arnst, R Ghanem, E Phipps, and J Red-Horse. Reduced chaos expansions with random coefficientsin reduced-dimensional stochastic modeling of coupled problems. *International Journal for Numerical Methods in Engineering*, 97(5):352–376, 2014.

[18] R Askey and JA Wilson. *Some basic hypergeometric orthogonal polynomials that generalize Jacobi polynomials*, volume 319. American Mathematical Soc., 1985.

[19] K Atkinson and O Hansen. A spectral method for the eigenvalue problem for elliptic equations. *Electronic Transactions on Numerical Analysis*, 37:386–412, 2010.

[20] F Auricchio, M Conti, A Lefieux, S Morganti, A Reali, F Sardanelli, F Secchi, S Trimarchi, and A Veneziani. Patient-specific analysis of post-operative aortic hemodynamics: a focus on thoracic endovascular repair (tevar). *Computational Mechanics*, 54(4):943–953, 2014.

[21] F Auteri and L Quartapelle. Spectral elliptic solvers in a finite cylinder. *Communications in Computational Physics*, 5(2-4):426–441, 2009.

[22] M Azaiez, J Shen, C Xu, and Q Zhuang. A Laguerre–Legendre spectral method for the Stokes problem in a semi-infinite channel. *SIAM Journal on Numerical Analysis*, 47(1):271–292, 2008.

[23] A Barone. Parallel and Multilevel Techniques for Hierarchical Model Reduction. Master's thesis, Politecnico di Milano, Italy, 2014.

[24] GK Batchelor. *An Introduction to Fluid Dynamics*. Cambridge University Press, Cambridge, 2000.

[25] KJ Bathe. The inf–sup condition and its evaluation for mixed finite element methods. *Computers & structures*, 79(2):243–252, 2001.

[26] O Beckwith, S Perotto, and A Veneziani. Inf-sup stability of hierarchical model reduction. In preparation.

[27] C Bernardi, M Dauge, Y Maday, and M Azaïez. *Spectral Methods for Axisymmetric Domains*, volume 3. Gauthier-Villars Paris, 1999.

[28] A Bhatele, T Gamblin, SH Langer, PT Bremer, EW Draeger, B Hamann, KE Isaacs, AG Landge, J Levine, V Pascucci, et al. Mapping applications with collectives over sub-communicators on torus networks. In *High Performance Computing, Networking, Storage and Analysis (SC), 2012 International Conference for*, pages 1–11. IEEE, 2012.

[29] HM Blackburn and JM Lopez. Modulated rotating waves in an enclosed swirling flow. *Journal of Fluid Mechanics*, 465:33–58, 2002.

[30] PJ Blanco, LA Mansilla Alvarez, and RA Feijóo. Hybrid element-based approximation for the Navier–Stokes equations in pipe-like domains. *Computer Methods in Applied Mechanics and Engineering*, 283:971–993, 2015.

[31] PJ Blanco, LA Mansilla Alvarez, and RA Feijóo. Hybrid element-based approximation for the Navier–Stokes equations in pipe-like domains. 283:971–993, 2015.

[32] PJ Blanco, SM Watanabe, MARF Passos, PA Lemos, and RA Feijóo. An anatomically detailed arterial network model for one-dimensional computational hemodynamics. 62(2):736–753, 2014.

[33] PJ Blanco, SM Watanabe, RAB Queiroz, PR Trenhago, LG Fernandes, and RA Feijóo. Trends in the computational modeling and numerical simulation of the cardiovascular system. In RA Feijóo, A Ziviani, and PJ Blanco, editors, *Scientific Computing Applied to Medicine and Healthcare*, chapter 2, pages 29–77. National Institute of Science and Technology in Medicine Assisted by Scientific Computing and National Laboratory for Scientific Computing, Petrópolis, RJ, 2012.

[34] D Boffi, F Brezzi, M Fortin, et al. *Mixed Finite Flement Methods and Applications*, volume 44. Springer Berlin, Heidelberg, 2013.

[35] JP Boyd. *Chebyshev and Fourier Spectral Methods*. Dover Publications, New York, second edition, 2001.

[36] JP Boyd and N Flyer. Compatibility conditions for time-dependent partial differential equations and the rate of convergence of Chebyshev and Fourier spectral methods. *Computer Methods in Applied Mechanics and Engineering*, 175(3-4):281–309, 1999.

[37] JP Boyd and F Yu. Comparing seven spectral methods for interpolation and for solving the Poisson equation in a disk: Zernike polynomials, Logan–Shepp ridge polynomials, Chebyshev–Fourier series, cylindrical Robert functions, Bessel–Fourier expansions, square-to-disk conformal mapping and radial basis functions. *Journal of Computational Physics*, 230(4):1408–1438, 2011.

[38] S Brenner and R Scott. *The Mathematical Theory of Finite Element Methods*, volume 15. Springer-Verlag New York, 2008.

[39] C Brezinski. Convergence acceleration during the 20th century. In *Numerical Analysis: Historical Developments in the 20th Century*, pages 113–133. Gulf Professional Publishing, 2001.

[40] H Brezis. *Functional Analysis, Sobolev Spaces and Partial Differential Equations*. Universitext. Springer New York, 2011.

[41] F Brezzi. On the existence, uniqueness and approximation of saddle-point problems arising from Lagrangian multipliers. *ESAIM: Mathematical Modelling and Numerical Analysis - Modélisation Mathématique et Analyse Numérique*, 8(R2):129–151, 1974.

[42] WL Briggs, VE Henson, and SF McCormick. *A multigrid tutorial*. SIAM, 2000.

[43] CA Bulant. *Computational models for the geometric and functional assessment of the coronary circulation*. PhD thesis, Laboratório Nacional de Computação Científica - LNCC, Petrópolis - Brazil, 2017.

[44] C Canuto, MY Hussaini, A Quarteroni, and TA Zang. *Spectral Methods - Fundamentals in Single Domains*. Scientific Computation. Springer-Verlag, Berlin, Heidelberg, 2006.

[45] KT Carlberg, M Khalil, K Sargsyan, and S Guzzetti. Uncertainty propagation in large-scale networks via domain decomposition. In preparation.

[46] KT Carlberg, M Khalil, R Tuminaro, and S Guzzetti. Multigrid methods for uncertainty propagation in large-scale networks via overlapping domain decomposition. In preparation.

[47] JR Cebral, F Mut, J Weir, and C Putman. Quantitative characterization of the hemodynamic environment in ruptured and unruptured brain aneurysms. *American Journal of Neuroradiology*, 32(1):145–151, 2011.

[48] D Chapelle and KJ Bathe. The inf-sup test. *Computers & structures*, 47(4-5):537–545, 1993.

[49] P Chaturani and RN Pralhad. Blood flow in tapered tubes with biorheological applications. *Biorheology*, 22(4):303–314, 1985.

[50] G Cheliotis, C Kenyon, R Buyya, and A Melbourne. Grid economics: 10 lessons from finance. *GRIDS Lab and IBM Research Zurich, Melbourne, Tech. Rep*, 2003.

[51] P Chen, A Quarteroni, and G Rozza. Simulation-based uncertainty quantification of human arterial network hemodynamics. *International Journal for Numerical Methods in Biomedical Engineering*, 29(6):698–721, 2013.

[52] BN Chun and DE Culler. User-centric performance analysis of market-based cluster batch schedulers. In *2002 2nd International Symposium on Cluster Computing and the Grid*, pages 30–30. IEEE/ACM, 2002.

[53] PG Constantine, ET Phipps, and TM Wildey. Efficient uncertainty propagation for network multiphysics systems. *International Journal for Numerical Methods in Engineering*, 99(3):183–202, 2014.

[54] A Coşkun, C Chen, PH Stone, and CL Feldman. Computational fluid dynamics tools can be used to predict the progression of coronary artery disease. *Physica A: Statistical Mechanics and its Applications*, 362(1):182–190, 2006.

[55] MA Costa and DI Simon. Molecular basis of restenosis and drug-eluting stents. *Circulation*, 111(17):2257–2273, 2005.

[56] R Courant and D Hilbert. *Methods of Mathematical Physics*, volume I. John Wiley & Sons, Inc., New York, 1966.

[57] R Craig and M Bampton. Coupling of substructures for dynamic analyses. *AIAA journal*, 6(7):1313–1319, 1968.

[58] D Darjany, B Englert, and EH Kim. Implementing Overlapping Domain Decomposition Methods on a Virtual Parallel Machine. In Geyong Min, Beniamino Di Martino, Laurence T. Yang, Minyi Guo, and Gudula Rünger, editors, *Frontiers of High Performance Computing and Networking – ISPA 2006 Workshops*, volume 4331 of *Lecture Notes in Computer Science*, pages 717–727. Springer Berlin Heidelberg, 2006.

[59] HF Davis. *Fourier Series and Orthogonal Functions*. Dover Publications, New York, 2012.

[60] DA de Zélicourt, CM Haggerty, KS Sundareswaran, BS Whited, JR Rossignac, KR Kanter, JW Gaynor, TL Spray, F Sotiropoulos, MA Fogel, and AP Yoganathan. Individualized computer-based surgical planning to address pulmonary arteriovenous malformations in patients with a single ventricle with an interrupted inferior vena cava and azygous continuation. *J. Thorac. Cardiovasc. Surg.*, 141(5):1170–1177, May 2011.

[61] B Debusschere, K Sargsyan, C Safta, and K Chowdhary. Uncertainty quantification toolkit (uqtk). *Handbook of Uncertainty Quantification*, pages 1–21, 2016.

[62] S Deparis. *Numerical Analysis of Axisymmetric Flows and Methods for Fluid-Structure Interaction Arising in Blood Flow Simulation*. PhD thesis, École Polytechnique Fédérale de Lausanne, 2004.

[63] X Du and W Chen. Collaborative reliability analysis under the framework of multidisciplinary systems design. *Optimization and Engineering*, 6(1):63–84, 2005.

[64] F Durst. *Fluid Mechanics: An Introduction to the Theory of Fluid Flows*. Springer Berlin Heidelberg, 2008.

[65] AP Dwivedi, TS Pal, and L Rakesh. Micropolar fluid model for blood flow through a small tapered tube. *Indian Journal of Technology*, 20:295–299, 1982.

[66] HC Elman, DJ Silvester, and AJ Wathen. *Finite elements and fast iterative solvers: with applications in incompressible fluid dynamics*. Oxford University Press, USA, 2005.

[67] JF Epperson. On the runge example. *The American Mathematical Monthly*, 94(4):329–341, 1987.

[68] A Ern and JL Guermond. *Theory and Practice of Finite Elements*, volume 159. Springer New York, 2013.

[69] A Ern, S Perotto, and A Veneziani. Hierarchical model reduction for advection-diffusion-reaction problems. In K Kunish, O Günther, and O Steinbach, editors, *Numerical Mathematics and Advanced Applications*, pages 703–710. Springer, Berlin, Heidelberg, 2008.

[70] B Everitt. The cambridge dictionary of statistics cambridge university press. *Cambridge, UK*, 1998.

[71] H Fang and Y Saad. Two classes of multisecant methods for nonlinear acceleration. *Numerical Linear Algebra with Applications*, 16(3):197–221, 2009.

[72] L Formaggia, JF Gerbeau, F Nobile, and A Quarteroni. Numerical treatment of defective boundary conditions for the navier–stokes equations. 40(1):376–401, 2002.

[73] L Formaggia, D Lamponi, and A Quarteroni. One-dimensional models for blood flow in arteries. *Journal of engineering mathematics*, 47(3-4):251–276, 2003.

[74] L Formaggia, D Lamponi, M Tuveri, and A Veneziani. Numerical modeling of 1d arterial networks coupled with a lumped parameters description of the heart. *Computer Methods in Biomechanics and Biomedical Engineering*, 9(5):273–288, 2006.

[75] L Formaggia, A Quarteroni, and A Veneziani, editors. *Cardiovascular Mathematics*, volume 1 of *M&SA*. Springer, Italy, 2009.

[76] L Formaggia, A Quarteroni, and A Veneziani. Multiscale models of the vascular system. In L Formaggia, A Quarteroni, and A Veneziani, editors, *Cardiovascular Mathematics: Modeling and Simulation of the Circulatory System*, volume 1 of *MS&A - Modeling Simulation and Applications*, chapter 11, pages 395–446. Springer Milan, 2009.

[77] L Formaggia and A Veneziani. Reduced and multiscale models for the human cardiovascular system. *Lecture notes VKI lecture series*, 7, 2003.

[78] L Formaggia, A Veneziani, and C Vergara. A new approach to numerical solution of defective boundary value problems in incompressible fluid dynamics. *SIAM Journal on Numerical Analysis*, 46(6):2769–2794, 2008.

[79] L Formaggia, A Veneziani, and C Vergara. Flow rate boundary problems for an incompressible fluid in deformable domains: formulations and solution methods. *Computer Methods in Applied Mechanics and Engineering*, 199(9-12):677–688, 2010.

[80] I Foster, Y Zhao, I Raicu, and S Lu. Cloud computing and grid computing 360-degree compared. In *2008 Grid Computing Environments Workshop (GCE '08)*, pages 1–10. IEEE, 2008.

[81] LM Friedman, C Furberg, DL DeMets, DM Reboussin, CB Granger, et al. *Fundamentals of clinical trials*, volume 4. Springer, 2010.

[82] K George and K Vipin. MeTis: Unstructured Graph Partitioning and Sparse Matrix Ordering System, Version 4.0. `http://www.cs.umn.edu/~metis`, 2009.

[83] RG Ghanem and PD Spanos. Spectral stochastic finite-element formulation for reliability analysis. *Journal of Engineering Mechanics*, 117(10):2351–2372, 1991.

[84] RG Ghanem and PD Spanos. *Stochastic finite elements: a spectral approach*. Courier Corporation, 2003.

[85] GD Giannoglou, JV Soulis, TM Farmakis, DM Farmakis, and GE Louridas. Haemodynamic factors and the important role of local low static pressure in coronary wall thickening. *IJC*, 86(1):27–40, 2002.

[86] BD Gogas, SB King, LH Timmins, T Passerini, M Piccinelli, A Veneziani, S Kim, DS Molony, DP Giddens, PW Serruys, et al. Biomechanical assessment of fully bioresorbable devices. *JACC: Cardiovascular Interventions*, 6(7):760–761, 2013.

[87] BD Gogas, L Timmins, T Passerini, M Piccinelli, S Kim, D Molony, A Veneziani, D Giddens, S King, and H Samady. Biomechanical assessment

of bioresorbable devices. *JACC: Cardiovascular Interventions*, 6(7):760–761, 2013.

[88] BD Gogas, B Yang, T Passerini, A Veneziani, M Piccinelli, G Esposito, E Rasoul-Arzrumly, M Awad, G Mekonnen, OY Hung, et al. Computational fluid dynamics applied to virtually deployed drug-eluting coronary bioresorbable scaffolds: Clinical translations derived from a proof-of-concept. *Global cardiology science & practice*, 2014(4):428, 2014.

[89] D Gottlieb and SA Orszag. *Numerical Analysis of Spectral Methods: Theory and Applications*, volume 26 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. Society for Industrial and Applied Mathematics, 1977.

[90] R Gould. Graph theory. Menlo Park, CA: Benjamin-Cummings, 1988.

[91] L Grinberg, T Anor, E Cheever, JR Madsen, and GE Karniadakis. Simulation of the human intracranial arterial tree. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 367(1896):2371–2386, 2009.

[92] L Grinberg, T Anor, JR Madsen, A Yakhot, and GE Karniadakis. Large-scale simulation of the human arterial tree. *CEPP*, 36:194–205, 2009.

[93] XS Gu, JE Renaud, and CL Penninger. Implicit uncertainty propagation for robust collaborative optimization. *Journal of Mechanical Design*, 128(4):1001–1013, 2006.

[94] S Guzzetti. Hierarchical Model Reduction for Incompressible Flows in Cylindrical Domains. Master's thesis, Politecnico di Milano, Italy, 2014.

[95] S Guzzetti. Hierarchical model reduction for the incompressible navier-stokes equations, 2014.

[96] S Guzzetti, LA Mansilla Alvarez, PJ Blanco, KT Carlberg, and A Veneziani. Reduced models for uncertainty quantification in large-scale networks via domain decomposition. In preparation.

[97] S Guzzetti, T Passerini, J Slawinski, U Villa, A Veneziani, and V Sunderam. Platform and algorithm effects on computational fluid dynamics applications in life sciences. *Future Generation Computer Systems*, 67:382–396, 2017.

[98] S Guzzetti, S Perotto, and A Veneziani. Hierarchical model reduction for incompressible fluids in pipes. *International Journal for Numerical Methods in Engineering*, 114(5):469–500, 2018.

[99] S Guzzetti, V Sunderam, and A Veneziani. Experimental optimization of parallel 3d overlapping domain decomposition schemes, 2015. to appear in Proceedings of 11th International Conference on Parallel Processing and Applied Mathematics.

[100] CM Haggerty, DA de Zélicourt, M Restrepo, J Rossignac, TL Spray, KR Kanter, MA Fogel, and AP Yoganathan. Comparing pre- and post-operative Fontan hemodynamic simulations: Implications for the reliability of surgical planning. *Ann Biomed Eng*, Jul 2012.

[101] JF Hale, DA McDonald, and JR Womersley. Velocity profiles of oscillating arterial flow, with some calculations of viscous drag and the Reynolds number. *The Journal of Physiology*, 128(3):629–640, 1955.

[102] TP Hamilton and P Pulay. Direct inversion in the iterative subspace (diis) optimization of open-shell, excited-state, and small multiconfiguration scf wave functions. *The Journal of chemical physics*, 84(10):5728–5734, 1986.

[103] TV How and RA Black. Pressure losses in non-Newtonian flow through rigid wall tapered tubes. *Biorheology*, 24(3):337–351, 1987.

[104] TJR Hughes and J Lubliner. On the one-dimensional theory of blood flow in the larger vessels. 18:161–170, 1973.

[105] WC Hurty. Dynamic analysis of structural systems using component modes. *AIAA journal*, 3(4):678–685, 1965.

[106] DBP Huynh, DJ Knezevic, and AT Patera. A static condensation reduced basis element method: approximation and a posteriori error estimation. *ESAIM: Mathematical Modelling and Numerical Analysis*, 47(1):213–251, 2013.

[107] A Iosup, S Ostermann, MN Yigitbasi, R Prodan, T Fahringer, and DHJ Epema. Performance Analysis of Cloud Computing Services for Many-Tasks Scientific Computing. *IEEE Transactions on Parallel and Distributed Systems*, 22(6):931–945, 2011.

[108] DE Irwin, LE Grit, and JS Chase. Balancing Risk and Reward in a Market-Based Task Service. In *2004 13th International Symposium on High Performance Distributed Computing (HPDC '04)*. IEEE, June 2004.

[109] E Isaacson and HB Keller. *Analysis of numerical methods*. Courier Corporation, 2012.

[110] KR Jackson, L Ramakrishnan, K Muriki, S Canon, S Cholia, J Shalf, HJ Wasserman, and NJ Wright. Performance Analysis of High Performance Computing Applications on the Amazon Web Services Cloud. In *CLOUDCOM '10: Proceedings of the 2010 IEEE Second International Conference on Cloud Computing Technology and Science*. IEEE Computer Society, November 2010.

[111] K Jbilou and H Sadok. Vector extrapolation methods. applications and numerical comparison. *Journal of Computational and Applied Mathematics*, 122(1-2):149–165, 2000.

[112] K Jbilou and H Sadok. Matrix polynomial and epsilon-type extrapolation methods with applications. *Numerical Algorithms*, 68(1):107–119, 2015.

[113] J Jiang, R Tuminaro, and KT Carlberg. Acceleration strategies for uncertainty propagation via overlapping domain decomposition. In preparation.

[114] S Joachim, G Hannes, and G Robert. NETGEN - automatic mesh generator. `http://www.hpfem.jku.at/netgen`, 2012.

[115] DN Ku. Blood flow in arteries. *Annual review of fluid mechanics*, 29(1):399–434, 1997.

[116] CB Lee and AE Snavely. Precise and realistic utility functions for user-centric performance analysis of schedulers. In *2007 16th International Symposium on High Performance Distributed Computing (HPDC '07)*, pages 107–116. ACM, 2007.

[117] A Leonard and A Wray. A new numerical method for the simulation of three-dimensional flow in a pipe. In *Eighth International Conference on Numerical Methods in Fluid Dynamics*, pages 335–342. Springer Berlin, Heidelberg, 1982.

[118] BC Lesieutre, A Pinar, and S Roy. Power system extreme event detection: The vulnerability frontier. In *Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008)*, pages 184–184. IEEE, 2008.

[119] RJ LeVeque. *Numerical Methods for Conservation Laws*. Lectures in Mathematics ETH Zürich, Department of Mathematics Research Institute of Mathematics. Birkhäuser Basel, 1992.

[120] Q Liao and K Willcox. A domain decomposition approach for uncertainty analysis. *SIAM Journal on Scientific Computing*, 37(1):A103–A133, 2015.

[121] PW Livermore, CA Jones, and SJ Worland. Spectral radial basis functions for full sphere computations. *Journal of Computational Physics*, 227(2):1209–1224, 2007.

[122] JM Lopez, F Marques, and J Shen. An efficient spectral-projection method for the Navier–Stokes equations in cylindrical geometries: II. three-dimensional cases. *Journal of Computational Physics*, 176(2):384–401, 2002.

[123] JM Lopez and J Shen. An efficient spectral-projection method for the Navier–Stokes equations in cylindrical geometries: I. axisymmetric cases. *Journal of Computational Physics*, 139(2):308–326, 1998.

[124] Y Maday and EM Rønquist. A reduced-basis element method. *Journal of scientific computing*, 17(1-4):447–459, 2002.

[125] ACI Malossi, PJ Blanco, and S Deparis. A two-level time step technique for the partitioned solution of one-dimensional arterial networks. *Computer Methods in Applied Mechanics and Engineering*, 237:212–226, 2012.

[126] LA Mansilla Alvarez. *An effective numerical technique for pipe-like domains and its application in computational hemodynamics*. PhD thesis, Laboratório Nacional de Computação Científica - LNCC, Petrópolis - Brazil, 2018.

[127] LA Mansilla Alvarez, PJ Blanco, C Bulant, E Dari, A Veneziani, and R Feijóo. Transversally enriched pipe element method (TEPEM): An effective numerical approach for blood flow modeling. *International Journal for Numerical Methods in Biomedical Engineering*, 33(4), 2017.

[128] LA Mansilla Alvarez, PJ Blanco, CA Bulant, and RA Feijóo. Towards fast hemodynamic simulations in large-scale circulatory networks. *Computer Methods in Applied Mechanics and Engineering*, 344:734–765, 2019.

[129] KS Matthys, J Alastruey, J Peiró, AW Khir, P Segers, PR Verdonck, KH Parker, and SJ Sherwin. Pulse wave propagation in a model human arterial network: assessment of 1-d numerical simulations against in vitro measurements. *Journal of biomechanics*, 40(15):3476–3486, 2007.

[130] DW Matula, G Marble, and JD Isaacson. Graph coloring algorithms. In *Graph theory and computing*, pages 109–122. Elsevier, 1972.

[131] A Migliavacca, G Pennati, G Dubini, R Fumero, R Pietrabissa, G Urcelay, EL Bove, TY Hsia, and MR De Laval. Modeling of the norwood circulation: effects of shunt size, vascular resistances, and heart rate. *American Journal of Physiopathology*, 280(5):H2076–H2086, May 2001.

[132] MP Mignolet, A Przekop, SA Rizzi, and SM Spottswood. A review of indirect/non-intrusive reduced order modeling of nonlinear geometric structures. *Journal of Sound and Vibration*, 332(10):2437–2460, 2013.

[133] L Mirabella, C Haggerty, Passerini T, M Piccinelli, PJ Del Nido, A Veneziani, and AP Yoganathan. Treatment planning for a tcpc test case: a numerical investigation under rigid and moving wall assumptions. *Int J Num Meth Biomed Eng*, 29(2):197–216, 2013.

[134] D Olson, P Bochev, M Luskin, and AV Shapeev. Development of an optimization-based atomistic-to-continuum coupling method. In *International Conference on Large-Scale Scientific Computing*, pages 33–44. Springer, 2013.

[135] D Olson, PB Bochev, M Luskin, and AV Shapeev. An optimization-based atomistic-to-continuum coupling method. *SIAM Journal on Numerical Analysis*, 52(4):2183–2204, 2014.

[136] M Park. Matamg: Matlab algebraic multigrid toolbox. `https://github.com/parkmh/MATAMG`, 2009.

[137] ML Parks, PB Bochev, and RB Lehoucq. Connecting atomistic-to-continuum coupling and domain decomposition. *Multiscale Modeling & Simulation*, 7(1):362–380, 2008.

[138] T Passerini, A Quaini, U Villa, A Veneziani, and S Canic. Validation of an open source framework for the simulation of blood flow in rigid and deformable vessels. *Int J Num Meth Biomed Eng*, 29(11):1192–1213, 2013.

[139] T Passerini, LM Sangalli, S Vantini, M Piccinelli, S Bacigaluppi, L Antiga, E Boccardi, P Secchi, and A Veneziani. An integrated statistical investigation of internal carotid arteries of patients affected by cerebral aneurysms. *Cardiovascular Engineering and Technology*, 3(1):26–40, 2012.

[140] J Peiró and A Veneziani. Reduced Models of the Cardiovascular System. In L Formaggia, A Quarteroni, and A Veneziani, editors, *Cardiovascular Mathematics*, pages 347–394. Springer-Verlag Mailand, 2009.

[141] S Perotto. A survey of hierarchical model (Hi-Mod) reduction methods for elliptic problems. In S.R. Idelsohn, editor, *Numerical Simulations of Coupled Problems in Engineering*, volume 33, pages 217–241. Springer Cham, 2014.

[142] S Perotto. Hierarchical Model (Hi-Mod) reduction in non-rectilinear domains. In J Erhel, MJ Gander, L Halpern, G Pichot, T Sassi, and O Widlund, editors, *Domain Decomposition Methods in Science and Engineering XXI*, pages 477–485. Springer International Publishing, Cham, 2014.

[143] S Perotto, A Ern, and A Veneziani. Hierarchical local model reduction for elliptic problems: a domain decomposition approach. *Multiscale Modeling & Simulation*, 8(4):1102–1127, 2010.

[144] S Perotto, A Reali, P Rusconi, and A Veneziani. HIGAMod: A Hierarchical IsoGeometric Approach for MODel reduction in curved pipes. *Computers & Fluids*, 142:21–29, 2017.

[145] S Perotto and A Veneziani. Coupled model and grid adaptivity in hierarchical reduction of elliptic problems. *Journal of Scientific Computing*, 60(3):505–536, 2014.

[146] S Perotto and A Zilio. Hierarchical model reduction: three different approaches. In A. Cangiani, R.L. Davidchack, E. Georgoulis, A.N. Gorban, J. Levesley, and M.V. Tretyakov, editors, *Numerical Mathematics and Advanced Applications 2011*, pages 851–859. Springer Berlin, Heidelberg, 2013.

[147] S Perotto and A Zilio. Space–time adaptive hierarchical model reduction for parabolic equations. *Advanced Modeling and Simulation in Engineering Sciences*, 2(1):25, 2015.

[148] M Piccinelli, A Veneziani, DA Steinman, A Remuzzi, and L Antiga. A framework for geometric analysis of vascular structures: application to cerebral aneurysms. *IEEE Trans Med Imaging*, 28(28):1141–55, 2009.

[149] N Poussineau. *Réduction Variationnelle d'un Couplage Fluide-Structure: Application à l'Hémodynamique*. PhD thesis, Paris 6, 2007.

[150] L. Quartapelle. *Numerical solution of the incompressible Navier-Stokes equations*, volume 113. Birkhauser Basel, 1993.

[151] A Quarteroni and L Formaggia. Mathematical modelling and numerical simulation of the cardiovascular system. *Handbook of numerical analysis*, 12:3–127, 2004.

[152] A Quarteroni, R Sacco, and F Saleri. *Numerical mathematics*. Text in Applied Mathematics. Springer, New York, 2007.

[153] A Quarteroni and A Valli. *Numerical Approximation of Partial Differential Equations*, volume 23 of *Springer Series in Computational Mathematics*. Springer-Verlag Berlin Heidelberg, 1994.

[154] A Quarteroni and A Valli. Domain decomposition methods for partial differential equations. Technical report, Oxford University Press, 1999.

[155] A Quarteroni, A Veneziani, and C Vergara. Geometric multiscale modeling of the cardiovascular system, between theory and practice. *Computer Methods in Applied Mechanics and Engineering*, 302:193–252, 2016.

[156] A Quarteroni, A Veneziani, and P Zunino. Mathematical and numerical modeling of solute dynamics in blood flow and arterial walls. *SIAM Journal on Numerical Analysis*, 39(5):1488–1511, 2002.

[157] JJ Rehr, FD Vila, JP Gardner, L Svec, and M Prange. Scientific Computing in the Cloud. *Computing in Science and Engineering*, 12(3):34–43, 2010.

[158] B Rummler. The eigenfunctions of the Stokes operator in special domains. I. *ZAMM-Journal of Applied Mathematics and Mechanics*, 77(8):619–627, 1997.

[159] T Sakai and LG Redekopp. An application of one-sided Jacobi polynomials for spectral modeling of vector fields in polar coordinates. *Journal of Computational Physics*, 228(18):7069–7085, 2009.

[160] S Salsa. *Partial Differential Equations in Action: From Modelling to Theory*. UNITEXT. Springer International Publishing, 3rd edition, 2016.

[161] RS Salzar, MJ Thubrikar, and RT Eppink. Pressure-induced mechanical stress in the carotid artery bifurcation: a possible correlation to atherosclerosis. *Journal of biomechanics*, 28(11):1333–1340, 1995.

[162] H Samady, P Eshtehardi, MC. McDaniel, J Suo, SS Dhawan, C Maynard, LH Timmins, AA Quyyumi, and DP Giddens. Coronary artery wall shear stress is associated with progression and transformation of atherosclerotic plaque and arterial remodeling in patients with coronary artery disease. *Circulation*, 124(7):779–788, 2011.

[163] Sandia National Laboratories. The Trilinos Project. `http://trilinos.sandia.gov`, 2012.

[164] DS Sankar and K Hemalatha. Non-linear mathematical models for blood flow through tapered tubes. *Applied Mathematics and Computation*, 188(1):567–582, 2007.

[165] S Sankaran and AL Marsden. A stochastic collocation method for uncertainty quantification and propagation in cardiovascular simulations. *Journal of Biomechanical Engineering*, 133(3):031001, 2011.

[166] DE Schiavazzi, G Arbia, C Baker, AM Hlavacek, TY Hsia, AL Marsden, and IE Vignon-Clementel. Uncertainty quantification in virtual surgery hemodynamics predictions for single ventricle palliation. *International Journal for Numerical Methods in Biomedical Engineering*, 32(3), 2016.

[167] J Shen. Efficient spectral-Galerkin methods III: Polar and cylindrical geometries. *SIAM Journal on Scientific Computing*, 18(6):1583–1604, 1997.

[168] SJ Sherwin, V Franke, J Peiró, and K Parker. One-dimensional modelling of a vascular network in space-time variables. *Journal of Engineering Mathematics*, 47(3-4):217–250, 2003.

[169] JN Silva, P Ferreira, and L Veiga. Service and resource discovery in cycle-sharing environments with a utility algebra. *Parallel & Distributed Processing (IPDPS), 2010 IEEE International Symposium on*, pages 1–11, 2010.

[170] J Simão and L Veiga. QoE-JVM: An Adaptive and Resource-Aware Java Runtime for Cloud Computing - Springer. *On the Move to Meaningful Internet Systems: OTM*, 2012.

[171] T Simmermacher, T Paez, A Urbina, F Bitsie, D Gregory, B Resor, and DJ Segalman. Probabilistic modeling of localized nonlinearities using component mode synthesis. In *Proceedings of the 22nd International Modal Analysis Conference (IMAC-XXII)*, pages 26–29, 2004.

[172] J Slawinski. *Adaptive Approaches to Utility Computing for Scientific Applications*. PhD thesis, Emory University, 2014.

[173] J Slawinski, T Passerini, U Villa, A Veneziani, and V Sunderam. Experiences with target-platform heterogeneity in clouds, grids, and on-premises resources. In *2012 26th International Parallel and Distributed Processing Symposium (IPDPS-HCW)*, pages 41–52. IEEE, 2012.

[174] J Slawinski, M Slawinska, and V Sunderam. Unibus-managed execution of scientific applications on aggregated clouds. In *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pages 518–521. IEEE Computer Society, 2010.

[175] J Slawinski, U Villa, T Passerini, A Veneziani, and V Sunderam. Issues in communication heterogeneity for message-passing concurrent computing. In *Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2013 IEEE 27th International*, pages 93–102. IEEE, 2013.

[176] RC Smith. *Uncertainty quantification: theory, implementation, and applications*, volume 12. Siam, 2013.

[177] AV Smolensky, S Clement, T Passerini, M Piccinelli, A Veneziani, JN Oshinski, and WR Taylor. Potential hemodynamic mechanisms for gender differences in aaa formation. In *ASME 2012 Summer Bioengineering Conference*, pages 11–12. American Society of Mechanical Engineers, 2012.

[178] SA Smolyak. Quadrature and interpolation formulas for tensor products of certain classes of functions. In *Doklady Akademii Nauk*, volume 148, pages 1042–1045. Russian Academy of Sciences, 1963.

[179] C Soize and C Farhat. A nonparametric probabilistic approach for quantifying uncertainties in low-dimensional and high-dimensional nonlinear models.

International Journal for Numerical Methods in Engineering, 109(6):837–888, 2017.

[180] MR Spiegel. *Theory and Problems of Fourier Analysis with Applications to Boundary Value Problems*. Schaum's Outline Series. McGraw-Hill, New York, 1974.

[181] EM Stein and R Shakarchi. *Fourier Analysis: An Introduction*, volume 1 of *Princeton Lectures in Analysis*. Princeton Unversity Press, Princeton, New Jersey, 2003.

[182] SFC Stewart, EG Paterson, GW Burgreen, P Hariharan, M Giarra, V Reddy, SW Day, KB Manning, S Deutsch, MR Berman, MR Myers, and RA Malinauskas. Assessment of CFD Performance in Simulations of an Idealized Medical Device: Results of FDA's First Computational Interlaboratory Study. *Cardiovascular Engineering and Technology*, 3(2):139–160, February 2012.

[183] H Subramoni, S Potluri, K Kandalla, B Barth, J Vienne, J Keasler, K Tomko, K Schulz, A Moody, and DK Panda. Design of a scalable infiniband topology service to enable network-topology-aware placement of processes. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, page 70. IEEE Computer Society Press, 2012.

[184] CA Taylor, T JR Hughes, and CK Zarins. Finite element modeling of blood flow in arteries. *Computer Methods in Applied Mechanics and Engineering*, 158(1-2):155–196, 1998.

[185] CA Taylor, TJR Hughes, and CK Zarins. Finite element modeling of blood flow in arteries. 158(1-2):155–196, 1998.

[186] CA Taylor and DA Steinman. Image-based modeling of blood flow and vessel wall dynamics: applications, methods and future directions. *Annals Biomed Eng*, 38(3):1188–1203, 2010.

[187] The HDF Group. Hierarchical data format version 5. `http://www.hdfgroup.org/HDF5`, 2012.

[188] A Toselli and O Widlund. *Domain Decomposition Methods: Algorithms and Theory*, volume 34 of *Springer Series in Computational Mathematics*. Springer Berlin Heidelberg, 2005.

[189] JS Tran, DE Schiavazzi, AM Kahn, and AL Marsden. Uncertainty quantification of simulated biomechanical stimuli in coronary artery bypass grafts. *Computer Methods in Applied Mechanics and Engineering*, 345:402–428, 2019.

[190] GHW van Bogerijen, F Auricchio, M Conti, A Lefieux, A Reali, A Veneziani, JL Tolenaar, FL Moll, V Rampoldi, and S Trimarchi. Aortic hemodynamics after thoracic endovascular aortic repair, with particular attention to the bird-beak configuration. *Journal of Endovascular Therapy*, 21(6):791–802, 2014.

[191] GHW van Bogerijen, F Auricchio, M Conti, A Lefieux, A Reali, A Veneziani, JL Tolenaar, M Moll, V Rampoldi, and S Trimarchi. Aortic hemodynamics after thoracic endovascular aortic repair with the role of bird-beak. *J Endov Therapy*, 21:791–802, 2014.

[192] A Veneziani. *Mathematical and Numerical Modeling of Blood Flow Problems*. PhD thesis, University of Milan, Italy, 1998.

[193] A Veneziani. *Stent design and improvement: a matter of Mathematics too*, chapter Chapter 4. A. Gruenztig Center, Emory University, 2015.

[194] IE Vignon-Clementel, CA Figueroa, KE Jansen, and CA Taylor. Outflow boundary conditions for three-dimensional finite element modeling of blood flow and pressure waves in arteries. 195:3776–3996, 2006.

[195] G Wang and TSE Ng. The Impact of Virtualization on Network Performance of Amazon EC2 Data Center. In *Proc IEEE INFOCOM 2010*, pages 1–9, 2010.

[196] T Washio and CW Oosterlee. Krylov subspace acceleration for nonlinear multigrid schemes. *Electronic Transactions on Numerical Analysis*, 6(271-290):3–1, 1997.

[197] N Westerhof, JW Lankhaar, and BE Westerhof. The arterial windkessel. *Medical & Biological Engineering & Computing*, 47(2):131–141, 2009.

[198] JR Womersley. Method for the calculation of velocity, rate of flow and viscous drag in arteries when the pressure gradient is known. *The Journal of Physiology*, 127(3):553–563, 1955.

[199] Inaugural CFD Challenge Workshop. The asme 2012 summer bioengineering conference. `www.asmeconferences.org/SBC2012/InauguralCFDWorkshop.cfm`, 2012.

[200] D Xiu and GE Karniadakis. The wiener–askey polynomial chaos for stochastic differential equations. *SIAM journal on scientific computing*, 24(2):619–644, 2002.

[201] D Xiu and SJ Sherwin. Parametric uncertainty analysis of pulse wave propagation in a model of a human arterial network. *Journal of Computational Physics*, 226(2):1385–1407, 2007.

[202] B Yang, BD Gogas, G Esposito, O Hung, ER Arzrumly, M Piccinelli, S King, D Giddens, A Veneziani, and H Samady. Novel in-human four dimensional calculation of a coronary bioresorbable scaffold using optical coherence tomography images and blood flow simulations. *Journal of the American College of Cardiology*, 65(10S), 2015.

[203] S Zhang and J Jin. *Computation of Special Functions*. John Wiley and Sons, Inc., New York, 1996.

[204] OC Zienkiewicz and RL Taylor. *The finite element method*, volume 3. McGraw-hill London, 1977.