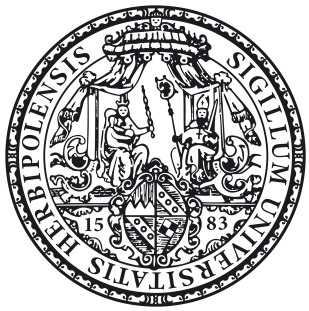
On Multigrid and $\mathcal{H}$-Matrix Methods for Partial Integro-Differential Equations



Dissertation zur Erlangung
des naturwissenschaftliches Doktorgrades
der Julius-Maximilian Universität Würzburg
vorgelegt von

Duncan Kioi Gathungu
aus
Kenia

Würzburg, 2017

Eingereicht am : November 2017
Tag der mündlichen Prüfung : 21.12.17

1. Betreuer: **Prof. Dr. Alfio Borzì**, Universität Würzburg
2. Betreuer: **Prof. Dr. Mario Bebendorf**, Universität Bayreuth

Acknowledgement

As I write this, I am blinking widly at the surrealism of it all. It's with profound gratitude that I acknowledge all who made this feat possible. First and foremost, I would like to thank my supervisor Prof. Dr. Alfio Borzì whose invaluable advice, encouragement, support and guidance enabled realization of this milestone. He meticulously guided me in the scientific field of multigrid methods and optimal control and through him I have grown in leaps and bounds as a researcher.

A special thanks to Prof. Dr. Mario Bebendorf who introduced me to the concept of hierarchical matrices and whose support enabled realization of a part of this thesis. Further, I would like to convey much appreciation to the immediate former and current members of WiReMIX team who in no order of preference include Tim, Melina, Jan, Christian, Tanvir, Dr. Gaviraghi, Dr. Merger, Dr. Wongkaew, Dr. Mohammadi, Dr. Ciaramella, Dr. Sprengel, Dr. Schmidt, Dr. Souvik, Prof. Dr. Griesmaier and Prof. Dr. Hahn.

A special gratitude to Petra and Dr. Schindele who played an instrumental role in enabling me settle down in Würzburg.

Further, I convey much appreciation to my scholarship facilitators, the Kenyan and Germany goverments and last but not least, I thank my family for the spiritual and moral support.
ASANTENI SANA KWA UKARIMU WENU!!

Duncan Kioi Gathungu.

Do the difficult things while they are easy and do the great things while they are small. A journey of a thousand miles must begin with a single step.

Lao Tzu of the Zhou Dynasty ($4^{th} - 6^{th}$ Century BCE)

Erklärung/Declaration

Hiermit erkläre ich, dass ich die eingereichte, Doktorarbeit eigenständig, d.h. insbesondere selbständig und ohne Hilfe einer kommerziellen Promotionsberatung angefertigt und keine anderen als die vor mir angegebenen Hilfsmitteln benutzt habe.

I hereby declare that I executed the thesis independently i.e. in particular, self-prepared and without assistance of a commercial doctorate consultancy and that no sources and tools other than those mentioned have been used.

.....	.....
Ort/Place, Datum/Date	Unterschrift/Signature

Abstract

The main theme of this thesis is the development of multigrid and hierarchical matrix solution procedures with almost linear computational complexity for classes of partial integro-differential problems. An elliptic partial integro-differential equation, a convection-diffusion partial integro-differential equation and a convection-diffusion partial integro-differential optimality system are investigated. In the first part of this work, an efficient multigrid finite-differences scheme for solving an elliptic Fredholm partial integro-differential equation (PIDE) is discussed. This scheme combines a second-order accurate finite difference discretization and a Simpson's quadrature rule to approximate the PIDE problem and a multigrid scheme and a fast multilevel integration method of the Fredholm operator allowing the fast solution of the PIDE problem. Theoretical estimates of second-order accuracy and results of local Fourier analysis of convergence of the proposed multigrid scheme are presented. Results of numerical experiments validate these estimates and demonstrate optimal computational complexity of the proposed framework that includes numerical experiments for elliptic PIDE problems with singular kernels. The experience gained in this part of the work is used for the investigation of convection diffusion partial-integro differential equations in the second part of this thesis. Convection-diffusion PIDE problems are discretized using a finite volume scheme referred to as the Chang and Cooper (CC) scheme and a quadrature rule. Also for this class of PIDE problems and this numerical setting, a stability and accuracy analysis of the CC scheme combined with a Simpson's quadrature rule is presented proving second-order accuracy of the numerical solution. To extend and investigate the proposed approximation and solution strategy to the case of systems of convection-diffusion PIDE, an optimal control problem governed by this model is considered. In this case the research focus is the CC-Simpson's discretization of the optimality system and its solution by the proposed multigrid strategy. Second-order accuracy of the optimization solution is proved and results of local Fourier analysis are presented that provide sharp convergence estimates of the optimal computational complexity of the multigrid-fast integration technique.

While (geometric) multigrid techniques require ad-hoc implementation depending on the structure of the PIDE problem and on the dimensionality of the domain where the problem is considered, the hierarchical matrix framework allows a more general treatment that exploits the algebraic structure of the problem at hand. In this thesis, this framework is extended to the case of combined differential and integral problems considering the case of a convection-diffusion PIDE. In this case, the starting point is the CC discretization of the convection-diffusion operator combined with the trapezoidal quadrature rule. The hierarchical matrix approach exploits the algebraic nature of the hierarchical matrices for blockwise approximations by low-rank matrices of the sparse convection-diffusion approximation and enables data sparse representation of the fully populated matrix where all essential matrix operations are performed with at most logarithmic optimal complexity. The factorization of part of or the whole coefficient matrix is used as a preconditioner to the solution of the PIDE problem using a generalized minimum residual (GMRes) procedure as a solver. Numerical analysis estimates of the accuracy of the finite-volume and trapezoidal rule approximation are presented and combined with estimates of the hierarchical matrix approximation and with the accuracy of the GMRes iterates. Results of numerical experiments are reported that successfully validate the theoretical estimates and the optimal computational complexity of the proposed hierarchical matrix solution procedure. These results include an extension to higher dimensions and an application to the time evolution of the probability density function of a jump diffusion process.

Zusammenfassung

Das Hauptthema dieser Arbeit ist die Entwicklung von Mehrgitter-Verfahren und hierarchischer Matrix-Lösungsverfahren mit nahezu linearer Rechenkomplexität für Klassen von partiellen Integro-Differential-Problemen. Es werden eine elliptische partielle Integro-Differentialgleichung, eine partielle Konvektions-Diffusions-Integro-Differentialgleichung und ein partielles Konvektions-Diffusions-Integro-Differential-Optimalitätssystem untersucht. Im ersten Teil dieser Arbeit wurde ein effizientes Mehrgitter-Finite-Differenzen-Schema zur Lösung einer elliptischen Fredholm partiellen Integro-Differentialgleichungen (PIDE) diskutiert. Dieses Schema kombiniert eine exakte finite Differenzen-Diskretisierung zweiter Ordnung mit einer Quadraturregel von Simpson, um das PIDE-Problem mit einem Mehrgitter-Schema und einer schnellen Multilevel-Integrationsmethode des Fredholm-Operators zu lösen, was eine schnelle Lösung des PIDE-Probleme ermöglicht. Theoretische Abschätzungen der Genauigkeit zweiter Ordnung und Ergebnisse der lokalen Fourier-Analyse der Konvergenz des vorgeschlagenen Mehrgitter-Systems werden präsentiert. Ergebnisse von numerischen Experimenten validieren diese Schätzungen und demonstrieren die optimale rechnerische Komplexität des vorgeschlagenen Frameworks, das numerische Experimente für elliptische PIDE mit singulären Kernen beinhaltet. Die in diesem Teil der Arbeit gewonnenen Erfahrungen werden zur Untersuchung einer partielle Konvektions-Diffusions-Integro-Differentialgleichungen im zweiten Teil verwendet. Konvektions-Diffusions-PIDE-Probleme werden unter Verwendung eines Finite-Volumen-Schemas, das als das Chang- Cooper- (CC-) Schema bezeichnet wird, und einer Quadraturregel diskretisiert. Auch für diese Klasse von PIDE-Problemen und diese numerische Einstellung wird eine Stabilitäts- und Genauigkeitsanalyse des CC-Schemas in Kombination mit einer Quadraturregel von Simpson vorgestellt, die die Genauigkeit der numerischen Lösung zweiter Ordnung beweist. Um die vorgeschlagene Approximations- und Lösungsstrategie auf den Fall von Konvektions-Diffusions-PIDE-Systemen auszudehnen und zu untersuchen, wird ein Optimalsteuerungsproblem mit diesem Modell als Nebenbedingung untersucht. Der Forschungsschwer-

punkt liegt dabei auf der Diskretisierung des Optimalitätssystems durch die CC-Simpson-Lösung und dessen Lösung durch die vorgeschlagene Mehrgitter-Strategie. Die Genauigkeit der optimalen Lösung zweiter Ordnung wird bewiesen und es werden Ergebnisse der lokalen Fourier-Analyse präsentiert, die scharfe Konvergenz-Schätzungen der optimalen Berechnungskomplexität der schnellen Mehrgitter Integrationstechnik liefern.

Während (geometrische) Mehrgitterverfahren je nach Struktur des PIDE-Problems und der Dimensionalität des Gebietes, in dem das Problem berücksichtigt wird, eine Ad-hoc-Implementierung erfordern, ermöglicht das hierarchische Matrix-Framework eine allgemeinere Behandlung, die die algebraische Struktur des Problems nutzt. In dieser Arbeit wird dieses Verfahren auf den Fall kombinierter Differential- und Integralprobleme im Fall einer Konvektions-Diffusions-PIDE erweitert. In diesem Fall ist der Startpunkt die CC-Diskretisierung des Konvektions-Diffusions-Operators in Kombination mit der Trapez-Quadratur-Regel. Der hierarchische Matrixansatz nutzt die algebraische Natur der hierarchischen Matrizen für blockweise Approximationen durch niedrigrangige Matrizen der dünn besetzten Konvektions-Diffusionsmatrix und ermöglicht eine datenarme Darstellung der vollständig besetzten Matrix, bei der alle wesentlichen Matrixoperationen mit höchstens logarithmisch optimaler Komplexität durchgeführt werden. Die Faktorisierung eines Teils oder der gesamten Koeffizientenmatrix wird als Vorbedingung für die Lösung der PIDE-Probleme unter Verwendung eines verallgemeinerten minimalen Restwert-Verfahrens (GMRes) als Löser verwendet.

Eine numerische Analyse der Abschätzungen der Genauigkeit der Finite-Volumen- und Trapezregel-Approximation werden präsentiert und kombiniert mit Abschätzungen der hierarchischen Matrix-Näherung und mit der Genauigkeit der GMRes iterationen kombiniert. Ergebnisse numerischer Experimente werden vorgestellt, die theoretischen Abschätzungen und die optimale rechnerische Komplexität der vorgeschlagenen hierarchischen Matrix Lösungsverfahren erfolgreich validieren. Diese Ergebnisse beinhalten eine Erweiterung auf höhere Dimensionen und eine Anwendung auf die zeitliche Entwicklung der Wahrscheinlichkeitsdichtefunktion des Sprungdiffusionsprozesses.

Contents

List of Tables

List of Figures

List of Algorithms

1. Introduction

Partial integro-differential equations (PIDEs) constitute a class of equations that involve both differential and integral terms [24, 44, 45, 53, 59, 62]. In the past decade, applications of PIDEs in real life problems have sprouted and therefore necessitated research into this particular field. These applications include modelling of jump diffusion processes [42, 45, 50], biological processes [1], computational neuroscience [46] and computational finance [23]. It is noticeable that, in the past, research on integro-differential problems has focused on one-dimensional problems mainly in the framework of ordinary differential equations where the problem of numerically solving PIDEs is already one of the main issues experienced in applications. For this reason, different strategies have been employed to address this problem. The work [53] develops a moving mesh finite-difference method for a PIDE that involves approximating the time dependent mapping of the coordinate transformation by a piecewise quadratic polynomial in space and piecewise linear functions in time. In [63] a one dimensional PIDE with a convolution kernel is solved through conversion of the PIDE to an ordinary differential equation and the use of the inverse Laplace transform. In [59, 70], compact finite-differences for one-dimensional PIDEs are studied. Additional results on high-order schemes for PIDE can be found in [24]. The research in [67] is devoted to an iterated Galerkin method for PIDE in one-dimension; see also [44]. Further, the work [66] considers the numerical solution of linear PIDE using projection methods. The work in [43] investigates a Tau method with Chebychev and Legendre basis to find the numerical solutions of Fredholm integro-differential equations where the differential part is replaced by its operational Tau representation. We remark that the methodologies referred above are designed for one dimensional problems and their complexity for multi-dimensional problems may become prohibitive. A key strategy for solving PIDE problems is to avoid the inversion of the resulting fully populated matrices of order N whose inversion requires $\mathcal{O}(N^3)$ operations, a requirement that is unsustainable when systems with higher degrees of freedom are considered. When the matrix of coefficients arising from a differential operator is sparse, iterative methods can solve the corresponding

linear algebraic problem efficiently with desired accuracy. However, in the case of dense matrices, the lack of sparsity renders operations on these matrices prohibitively expensive and there is the need to explore other solution techniques that provide the desired results with optimal computational complexity.

Among the methods that have been proposed for solving algebraic problems involving dense matrices, we enumerate the solution strategies by using hierarchical matrices ($\mathcal{H}$-matrices) [8, 36] and multigrid methods [20, 33]. However, while these methodologies have been investigated in the case of integral equations, much less is known in the case of partial integro-differential problems.

It is the purpose of this thesis to contribute to this field of research with the development and analysis of a methodology that is appropriate for multi-dimensional PIDE problems and, in doing so, we use numerical schemes that not only guarantee desired accuracy but also fulfil certain computational complexity requirements. In this thesis, we consider different classes of PIDEs and different discretization schemes. In the first part, we start by presenting a second-order accurate fast multigrid scheme to solve elliptic problems of the following form

$$\begin{aligned} -\Delta y(x) + \int_{\Omega} k(x,z)y(z)dz &= f(x) \qquad \text{in} \quad \Omega, \\ y(x) &= 0 \qquad \text{on} \quad \Gamma, \end{aligned} \tag{1.1}$$

where $-\Delta$ represents the minus Laplacian in the domain $\Omega \subset \mathbb{R}^d$, $d = 2$. Our approach is to combine a multigrid scheme for elliptic problems with the multigrid kernel approximation strategy developed in [20]. For this purpose, we discretize our PIDE problem by finite-differences and quadrature rules and analyse the stability and accuracy of the resulting scheme in the case of the minus Laplace operator that is combined with a Fredholm Hilbert-Schmidt integral operator.

It is well-known that a multigrid scheme solves elliptic problems with optimal computational complexity. However, this is in general not true if a straightforward implementation of the integral term is considered. On the other hand, the multigrid kernel approximation strategy proposed in [20] allows approximation of a Fredholm integral term with $\mathcal{O}(h^{2s})$ accuracy while reducing the complexity of its calculation from $\mathcal{O}(N^2)$ to $\mathcal{O}(sN)$, where N is the number of unknowns.

In the second part of this thesis, we consider a PIDE of the form

$$\begin{aligned} -\varepsilon\Delta y(x) + \nabla \cdot (b\, y(x)) + \int_{\Omega} k(x,z)y(z)dz + \lambda y(x) &= f(x) \quad \text{in} \quad \Omega, \\ y(x) &= 0 \qquad \text{on} \quad \Gamma, \end{aligned} \tag{1.2}$$

where ε is the diffusion coefficient, b is the drift coefficient and λ is the coefficient of the reaction term. This PIDE is referred to as the convection-diffusion PIDE. In this case, for the discretization we employ a second-order accurate and positive finite volume scheme referred to as Chang and Cooper (CC) scheme; see, e.g., [2, 22, 30, 54]. For the integral part of our PIDE operator, we focus on a Fredholm integral term with a positive semi-definite Hilbert-Schmidt kernel that we discretize with a fourth-order accurate Simpsons' quadrature rule. These techniques are further developed and analysed in the context of an optimal control problem governed by the convection-diffusion PIDE (1.3) with a distributed control mechanism as follows

$$\begin{cases} \min J(y,u) & := \frac{1}{2}\|y-y_d\|^2_{L^2(\Omega)} + \frac{\nu}{2}\|u\|^2_{L^2(\Omega)} \\ -\varepsilon\Delta y + \nabla\cdot(b\,y) + \mathcal{I}y + \lambda y & = f+u \quad \text{in} \quad \Omega, \\ y & = 0 \qquad \text{on} \quad \Gamma, \\ u & \in U_{ad}. \end{cases} \tag{1.3}$$

Specifically, we consider this distributed control problem with a quadratic cost and an objective function of tracking type. The purpose of this problem is to find u in the set of admissible controls U_{ad} such that the state y of the system modelled by the PIDE is as much as possible (in the L^2 norm) close to the given target function y_d, while the cost of the control $\|u\|^2_{L^2(\Omega)}$ is kept at a minimum. The main result of this part is a computational tool that allows to solve control-constrained convection-diffusion PIDE optimal control problems with $\mathcal{O}(N \log N)$ complexity. The main motivation for considering this optimal control problem is that the solution to (1.3) is characterized by the so-called first-order optimality conditions that correspond to a system of coupled PIDEs, thus providing a benchmark for validating our multigrid strategy with systems of PIDEs.

In addition to our multigrid strategy and in view of applications with different PIDE in high-dimensions, we develop a hierarchical matrix ($\mathcal{H}$-matrix) framework that is able to solve PIDE problems with convection-diffusion differential operators efficiently in higher dimensions. For this reason, in the third part of this thesis, we investigate solution of PIDE problems using the hierarchical matrix approach [8, 36], considering the convection-diffusion PIDE given by (1.2). We discuss a strategy of using $\mathcal{H}$-matrices to obtain solutions of PIDEs with almost linear complexity. We present accuracy estimates for a CC-trapezoidal rule discretization and combine these estimates with the $\mathcal{H}$-matrix approximation estimates. Storage requirements and perfomance are investigated for different parameters. The generalized minimum residual method (GMRes) with $\mathcal{H}$-matrix preconditioning, is used to obtain the solution of the PIDE. In addition, we extend the application of the hierarchical matrices to a higher dimensional convection-diffusion PIDE as well as an application to a jump diffusion process.

This thesis is organized as follows.
In Chapter 2, we start by discussing the model problems to be considered in this work in appropriate functional spaces and we outline the properties of the integral terms of the PIDE model problems. We discuss the existence of unique solutions for these model problems. Further, we discuss the discretization of these PIDE model problems that include elliptic and convection-diffusion PIDEs. We discuss the finite difference framework used for the differential part of the elliptic PIDE (1.1) and the quadrature rule used. Estimates of solution error of the numerical approximation for this problem are given, proving second-order accuracy. Further, we discuss the CC finite volume scheme that is used to discretize the differential part of the convection-diffusion PIDE given by (1.2). An extension to a three dimensions case of the convection-diffusion PIDE is discussed as well.
In Chapter 3, we discuss the multigrid procedure implemented for the classes of PIDE problems mentioned above and for a related optimality system. We outline the working principles of the multigrid method and the main components of the multigrid procedure. We start by discussing the smoothing procedure to be used in the multigrid cycle, then we review the grid transfer operators. We also discuss in detail the fast integration procedure for the integral term of the PIDE, thereafter we summarize the solution procedures with pseudo codes that incorporate the fast integration in the multigrid cycle. Further, by means of local Fourier analysis we obtain sharp convergence estimates for our multigrid algorithms.
In Chapter 4, we outline the hierarchical matrix approach to the solution of convection diffusion PIDE. We give an overview and the main concepts of hierarchical matrix approximation. We illustrate the fundamental arithmetic operations involving the hierarchical matrices. We describe the hierarchical factorization of the hierarchical matrix which is used as a preconditioner in the GMRes iterative solution procedure of the PIDE problem. We prove theoretical error estimates for the numerical solutions using the hierarchical matrix approximation method. These second order error estimates are validated by numerical experiments with PIDE problems in two dimensions. We present storage and computational time requirements of the hierarchical matrix approximation showing an almost linear computational complexity. Further, an extension of our methodology to a three-dimensional PIDE problem and an application to jump diffusion processes are given, where results of numerical experiments also show an almost linear computational complexity of our numerical $\mathcal{H}$-matrix framework.
In Chapter 5, we outline a multigrid framework for solving a convection diffusion PIDE optimal control problem. We discuss the discretzation technique used to approximate the optimality system. Since the discretized optimality system will be solved in the multigrid framework, we give a detailed exposition of the multigrid method for the system with foreknowledge of the description given in the previous chapter and sections. We present results of local Fourier analysis

of our multigrid scheme applied to the optimality system and compare the estimated convergence factors to the corresponding numerical values obtained in the experiments. We perform experiments to validate the theoretical solution error estimates for the state, adjoint and control variables. Results of experiments validate second order error estimates for the state, adjoint and control variables in unconstrained control case while second order error estimates are realized for the state and adjoint and an order $\frac{3}{2}$ for the control in the constrained control case. Further, the empirical multigrid convergence factors obtained in the experiments are shown to confirm the convergence estimates by local Fourier analysis.
A section of conclusion completes the exposition of this thesis.
The results presented in this thesis are partly based on the following publications:

1. D. K. Gathungu and A. Borzì, *Multigrid solution of an elliptic Fredholm partial integro-differential equation with a Hilbert-Schmidt integral operator*, Applied Mathematics (2017), **8**:967-986.

2. D. K. Gathungu and A. Borzì, *A multigrid scheme for solving convection-diffusion-integral optimal control problems*, Computing and Visualization in Science (2017), DOI:10.1007/s00791-017-0285-7.

3. D. K. Gathungu, M. Bebendorf and A. Borzì, *Hierarchical matrices for convection-diffusion partial integro-differential equations*, Submitted for publication.

2. Partial Integro-Differential equations

In this chapter, we discuss our model PIDE problems. We start this chapter stating the existence of solutions to these problems in appropriate functional spaces. In Section 2.2.1, we discuss the discretization of the elliptic PIDE problem by finite differences and quadrature rules and prove the stability and orders of accuracy of the discretization error. In Section 2.2.2, we discuss the discretization of the convection-diffusion PIDE by the CC scheme and prove stability and second-order error estimates of the discretization scheme. In 2.2.3, we discuss the discretization of a three dimensional convection-diffusion PIDE.

2.1 Model Problems

In this section, we define our working PIDE models. We prove existence and uniqueness of solutions to our given PIDE problems, which represent our models of choice for implementation of our numerical framework. We consider real-valued functions defined in a Lipschitz domain, i.e. a bounded and convex open set $\Omega \subset \mathbb{R}^2$ with boundary Γ that is locally a graph of a Lipschitz continuous function. $\bar{\Omega} := \Omega \cap \Gamma$ denotes the closure of Ω.
We consider an elliptic PIDE of the form

$$\begin{aligned} -\Delta y(x) + \int_{\Omega} k(x,z)y(z)dz &= f(x) \qquad \text{in} \quad \Omega \\ y(x) &= 0 \qquad \text{on} \quad \Gamma, \end{aligned} \tag{2.1}$$

where $x, z \in \Omega$. We adopt a notation of the integral operator as $\mathcal{I}$ where

$$\mathcal{I}y(x) = \int_{\Omega} k(x,z)y(z)dz,$$

and furthermore we choose $f \in L^2(\Omega)$.

We consider a symmetric positive semi-definite Hilbert Schmidt kernel $k \in$

$L^2(\Omega \times \Omega)$, such that $\int\int_{\Omega\Omega} \mid k(x,z) \mid^2 dx\, dz < \infty$, and the following holds

$$\int_{\Omega}\int_{\Omega} k(x,z)v(x)v(z)dx\, dz \geq 0, \qquad \text{for all} \quad v \in L^2(\Omega). \tag{2.2}$$

We have the following theorem.

Theorem 1. *Let $k \in L^2(\Omega \times \Omega)$ be a Hilbert Schmidt kernel. The integral operator $\mathcal{I}$ given by*

$$\mathcal{I}\, y(x) = \int_{\Omega} k(x,z)y(z)dz, \qquad x, z \in \Omega$$

defines a bounded mapping of $L^2(\Omega)$ into itself, with the Hilbert Schmidt norm $\|\mathcal{I}\| < \|k\|_{L^2(\Omega\times\Omega)}$.

Proof. From Tonelli's theorem, $\mathcal{I}\, y(x) = \int_{\Omega} k(x,z)y(z)dz$ is a measurable function of x and its L^2-norm can be determined by the *Cauchy-Schwarz inequality*. Let $y \in L^2(\Omega)$, we have

$$\begin{aligned}
\|\mathcal{I}\, y\|^2_{L^2(\Omega)} &= \int_{\Omega} \mid \mathcal{I}\, y(x) \mid^2 dx = \int_{\Omega} \mid \int_{\Omega} k(x,z)y(z)dz \mid^2 dx \\
&\leq \int_{\Omega} \left(\int_{\Omega} \mid k(x,z) \mid^2 dz \right) \left(\int_{\Omega} \mid y(z) \mid^2 dz \right) dx \\
&= \int_{\Omega}\int_{\Omega} \mid k(x,z) \mid^2 \|y\|^2_{L^2(\Omega)} dz dx.
\end{aligned}$$

Hence $\mathcal{I}y \in L^2(\mathbb{R})$.
□

Remark 2. *From Schur's test [39], since the kernel k is a measurable function, it satisfies the following conditions*

$$\xi_1 = \operatorname*{ess\,sup}_{x\in\mathbb{R}} \int_{\Omega} \mid k(x,z) \mid dz < \infty, \qquad \xi_2 = \operatorname*{ess\,sup}_{z\in\mathbb{R}} \int_{\Omega} \mid k(x,z) \mid dx < \infty.$$

Then the integral operator $\mathcal{I}$ defines a bounded mapping and $\| \mathcal{I} \|_2 \leq (\xi_1\xi_2)^{\frac{1}{2}}$.

Now, we state the following theorem.

Theorem 3. *There exists a unique function $y \in H_0^1(\Omega) \cap H^2(\Omega)$ that solves (2.1).*

Proof. On obtaining the variational formulation of (2.1), the proof is as a result of applying Lax-Milgram theorem and the properties of the kernel . □

Next, we define the convection-diffusion PIDE that is considered in the second part of the thesis. It is of the form

$$\begin{aligned} -\varepsilon\Delta y(x) + \nabla\cdot(b\,y(x)) + \mathcal{I}y(x) + \lambda y(x) &= f(x) \quad \text{in} \quad \Omega, \\ y(x) &= 0 \qquad \text{on} \quad \Gamma, \end{aligned} \tag{2.3}$$

where $y\colon \Omega \to \mathbb{R}$ represents the unknown function and Ω is a bounded and convex open set in $\mathbb{R}^2$, with Lipschitz boundary Γ. We choose $f \in L^2(\Omega)$, the diffusion coefficient $\varepsilon > 0$, the linear reaction coefficient $\lambda > 0$, and the drift $b = (b_1, b_2) \in C^1(\bar{\Omega}) \times C^1(\bar{\Omega})$ is a smooth vector-function on $\bar{\Omega}$.
The term

$$\mathcal{I}y(x) = \int_\Omega k(x,z)y(z)dz,$$

with $x, z \in \Omega$, is a Hilbert-Schmidt integral operator with a symmetric positive semi-definite Hilbert-Schmidt kernel $k \in L^2(\Omega \times \Omega)$.

Theorem 4. *There exist a unique function $y \in H_0^1(\Omega) \cap H^2(\Omega)$ that solves (2.3).*

Proof. The proof follows the same reasoning outlined in [27, Chapter 3], by applying Lax-Milgram theorem and the properties of kernel of the integral term. □

With the definition of the PIDE problems under consideration, we discuss the discretization techniques used in this thesis in the following section.

2.2 Discretization methods

We discretize the PIDE problems (2.1) and (2.3) using finite differences and finite volume schemes, respectively, for the differential operators and quadrature rules for the integral terms [35, 47, 60]. For simplicity, we assume that $k \in C(\bar{\Omega} \times \bar{\Omega})$ and $f \in C(\bar{\Omega})$ such that we can evaluate these functions on grid points. Specifically, we consider $\Omega = (a,b) \times (a,b)$ and N is an integer with $N \geq 2$. Let $h = \frac{b-a}{N-1}$ be the mesh size and consider an equidistant grid. We denote the mesh points $x_i = a + (i-1)h,\ i = 1,\dots,N$, and $z_j = a + (j-1)h,\ j = 1,\dots,N$. These grid points define the following grid

$$\Omega_h = \{\mathcal{Z}_{ij} = (x_i, z_j) \in \mathbb{R}^2 : i,j = 2,\dots,N-1\} \cap \Omega.$$

Later, we consider a sequence of nested uniform grids $\{\Omega_{h_\ell}\}_{h_\ell > 0}$, where $N = N_\ell = 2^\ell + 1$ for $\ell \in \mathbb{N}$.

For grid functions v and w defined on Ω_h, we introduce the discrete L^2-scalar

product given by

$$(v,w)_h = h^2 \sum_{\mathcal{Z}\in\Omega_h} v(\mathcal{Z})\, w(\mathcal{Z}) = \sum_{i=2}^{N-1}\sum_{j=2}^{N-1} h^2 v(x_i,z_j)w(x_i,z_j),$$

with associated discrete L_h^2-norm given by $\|v\|_h = \sqrt{(v,v)_h}$. The discrete H^1-norm is given by $\|v\|_{1,h} = \left(\|v\|_h^2 + \sum_{i=1}^{2} \left\|\partial_i^- v\right\|_{L_h^2}\right)^{\frac{1}{2}}$ where $\partial_i^- v$ denotes the backward difference quotient in the x_i direction; see, e.g., [47]. Given continuous functions in Ω are approximated by grid functions defined through their values at the grid points. Thus the right-hand sides of (2.1) and (2.3) in Ω_h are represented by $f_{ij}^h = f(x_i,z_j)$, and similarly for the kernel function.

Further, we introduce the following finite-difference operators. The forward finite-difference operator is given by

$$D_x^+ y(x_i,z_j) \equiv \frac{y(x_{i+1},z_j) - y(x_i,z_j)}{h}. \tag{2.4}$$

The backward finite-difference operator is as follows

$$D_x^- y(x_i,z_j) \equiv \frac{y(x_i,z_j) - y(x_{i-1},z_j)}{h}. \tag{2.5}$$

With these operators, we can define the H_h^1 norm as

$$\|y\|_{1,h} = \left(\|y\|_h^2 + \left\|D_x^- y]\right|_x^2 + \left\|D_z^- y]\right|_z^2\right)^{1/2}.$$

Notice that the bracket] denotes summation up to N in the given direction x, resp. z; see [47]. With this preparation, we have

$$\begin{aligned}\Delta_h y(x_i,z_j) &= D_x^- D_x^+ y(x_i,z_j) + D_z^- D_z^+ y(x_i,z_j)\\ &= \tfrac{y(x_{i+1},z_j)-2y(x_i,z_j)+y(x_{i-1},z_j)}{h^2} + \tfrac{y(x_i,z_{j+1})-2y(x_i,z_j)+y(x_i,z_{j-1})}{h^2}.\end{aligned}$$

which in stencil form is written as

$$\frac{1}{h^2}\begin{bmatrix} 0 & 1 & 0\\ 1 & -4 & 1\\ 0 & 1 & 0\end{bmatrix}.$$

The integral term of the elliptic PIDE is written explicitly as follows

$$\mathcal{I}y(x) = \int_\Omega k(x,z)y(z)dz\,, \qquad x,\ z\in\Omega.$$

The integral term is approximated by quadrature techniques and we focus on the trapezoidal and Simpson's rules.

Using the trapezoidal quadrature rule, we have the following approximation of the integral operator

$$\mathcal{I}^h y(x_{ij}) = h^2 \sum_{l=1}^{N}\sum_{m=1}^{N} r(l,m)\, k(x_{ij}, z_{lm})\, y(z_{lm}), \qquad x, z \in \Omega_h, \tag{2.6}$$

where the $r(l,m) = \tilde{r}(l)\,\tilde{r}(m)$ represent the coefficients of the quadrature rule, which in the trapezoidal case are given by

$$\tilde{r}(l) = \begin{cases} \frac{1}{2} & \text{if } l = 2, l = N-1 \\ 1 & l \bmod 2 = 0\ , l = 1 \dots N. \end{cases} \tag{2.7}$$

Assuming a sufficiently regular integrand, the trapezoidal rule provides a second-order accurate approximation of the integral as follows [61]

$$\left\| \mathcal{I}^h y - \mathcal{I} y \right\|_h = \mathcal{O}(h^2).$$

On using the Simpson's quadrature rule, we also have the following approximation of this integral operator

$$\mathcal{I}^h y(x_{ij}) = h^2 \sum_{l=1}^{N}\sum_{m=1}^{N} r(l,m) k(x_{ij}, z_{lm}) y(z_{lm})\ , x,\ z \in \Omega_h, \tag{2.8}$$

where $r(l,m) = \tilde{r}(l)\tilde{r}(m)$. These coefficients of the Simpson's quadrature rule are given by

$$\tilde{r}(l) = \begin{cases} \frac{1}{3} & if\ l = 1, l = N \\ \frac{4}{3} & l\ mod\ 2 = 0\ , l = 1 \dots N \\ \frac{2}{3} & \text{else.} \end{cases}$$

The Simpson's rule provides a fourth-order accurate approximation of the integral as follows

$$\left\| \mathcal{I}^h y - \mathcal{I} y \right\|_h = \mathcal{O}(h^4).$$

We refer to the formulae (2.6) and (2.8) as the full-kernel (FK) evaluations.
We need the following lemma.

Lemma 5. *The positivity of the Hilbert Schmidt operator stated in (2.2) is preserved after discretization.*

Proof. Consider the following function $v(x) = \sum_{l,m=1}^{N} \tilde{\delta}_\varrho(x - z_{lm}) v_{lm}$ where $\tilde{\delta}_\varrho(x)$ is a suitable approximation of the Dirac delta function by a Gaussian with

variance $\varrho \to 0$. Inserting this function in (2.2) we have

$$\sum_{l,m=1}^{N}\sum_{i,j=1}^{N} v_{lm}v_{ij}\int_{\Omega}\int_{\Omega} k(x,z)\tilde{\delta}_{\varrho}(x-x_{lm})\tilde{\delta}_{\varrho}(z-z_{ij})dx\,dz \geq 0. \tag{2.9}$$

Therefore by continuity, as $\varrho \to 0$ the above integral tends to $k(x_{lm}, z_{ij})$. Thus, we obtain $(\mathcal{I}^h y, y)_h \geq 0$.
□

Next, we discuss the discretization techniques of each class of PIDE.

2.2.1 Discretization of the elliptic PIDE model

With the setting above, the negative Laplacian with homogeneous Dirichlet boundary conditions is approximated by the five-point stencil and is denoted by $-\Delta^h$. We write the finite-difference scheme and quadrature rule approximation of (2.1) as follows

$$-\Delta^h Y + \mathcal{I}^h Y = f_h \qquad \text{in} \quad \Omega_h, \tag{2.10}$$

where $Y = (Y_{ij})$ denotes the numerical approximation to y.

Next, we investigate the stability and accuracy of (2.10). For this purpose, we use the numerical analysis framework in [47]. We denote $\mathcal{A}^h = -\Delta^h + \mathcal{I}^h$.
We need the following lemma, see also [47].

Lemma 6. *Suppose Y is a function defined on $\bar{\Omega}_h$ with $Y = 0$ on the boundary and assume positivity of the Hilbert-Schmidt kernel, then the following holds*

$$\left(\mathcal{A}^h Y, Y\right)_h \geq \sum_{i=1}^{N}\sum_{j=1}^{N-1} h^2 \left|D_x^- Y_{ij}\right|^2 + \sum_{i=1}^{N-1}\sum_{j=1}^{N} h^2 \left|D_z^- Y_{ij}\right|^2.$$

Proof. Using the results of Lemma 5, we have

$$\begin{aligned}
\left(\mathcal{A}^h Y, Y\right)_h &= \left(-D_x^+ D_x^- Y - D_z^+ D_z^- Y + \mathcal{I}^h Y, Y\right)_h, \\
&= (-D_x^+ D_x^- Y, Y)_h + (-D_z^+ D_z^- Y, Y)_h + (\mathcal{I}^h Y, Y)_h, \\
&\geq \sum_{i=1}^{N}\sum_{j=1}^{N-1} h^2 \left|D_x Y_{ij}\right|^2 + \sum_{i=1}^{N-1}\sum_{j=1}^{N} h^2 \left|D_z Y_{ij}\right|^2, \\
&\geq \|D_x^- Y]|_x^2 + \|D_z^- Y]|_z^2.
\end{aligned} \tag{2.11}$$

□

Lemma 7. *Suppose Y is a function defined on $\bar{\Omega}_h$ with $Y = 0$ on the boundary; then there exists a constant ρ_*, which is independent of Y and h, such that the following*

discrete Poincaré-Friedrichs inequality holds

$$\|Y\|_h^2 \leq \rho_* \left(\left\| D_x^- Y \right] \right|_x^2 + \left\| D_z^- Y \right] \right|_z^2 \Big), \tag{2.12}$$

for all such Y; see [47].

Remark 8. *From (2.11) and (2.12), we obtain $\left(-\Delta^h Y + \mathcal{I}^h Y, Y\right)_h \geq \rho_0 \|Y\|_{1,h}^2$, where $\rho_0 = 1/(1+\rho^*)$.*

Theorem 9. *The scheme (2.10) is stable in the sense that $\|Y^*\|_{1,h} \leq \frac{1}{\rho_0} \|f_h\|_h$.*

Proof. We have

$$\begin{aligned} \rho_0 \|Y\|_{1,h}^2 &\leq (-\Delta^h Y + \mathcal{I}^h Y, Y)_h = (f_h, Y)_h \leq |(f_h, Y)_h|, \\ &\leq \|f_h\|_h \|Y\|_h \leq \|f_h\|_h \|Y\|_{1,h}, \end{aligned}$$

hence $\|Y^*\|_{1,h} \leq \frac{1}{\rho_0} \|f_h\|_h$.
□

We conclude this section with the following theorem where we consider quadrature rules that are at least $\mathcal{O}(h^2)$ accurate.

Theorem 10. *Suppose $f \in L^2(\Omega)$ and k is a continuous positive Hilbert Schmidt kernel, and assume that the weak solution y to (2.1) belongs to $C^4(\bar{\Omega})$; then the solution Y to (2.10) approximates y with second-order accuracy as follows*

$$\|y - Y\|_{1,h} \leq c\, h^2,$$

where c is a positive constant independent of h. In particular $\|y - Y\|_h \leq c\, h^2$.

Proof. The proof uses Theorem 9 and the fact that the truncation error of (2.10) is of second order. This proof follows exactly the same reasoning as in Theorem 2.26 in [47]. □

2.2.2 Discretization of the convection-diffusion PIDE model

In this section, we discuss the approximation of the convection-diffusion PIDE problem (2.3). The diffusion and convection terms are discretized using the Chang and Cooper scheme [22, 54], while the integral term is approximated by using a second-order accurate quadrature rule.

The Chang and Cooper (CC) scheme is a second-order accurate, cell-centred finite volume scheme that results in a monotone discrete convection-diffusion operator; see [2, 54] for the numerical analysis of this scheme. The way to formulate the CC scheme is to consider the flux form of the convection and diffusion terms in the state equation as follows

$$\nabla \cdot F = \nabla \cdot [\varepsilon \nabla y - b y],$$

and consider the following scheme for the elementary cell centered at $\mathcal{Z}_{ij} \in \Omega_h$. We have

$$\nabla \cdot F = \frac{1}{h}\left\{\left(F_{i+1/2,j} - F_{i-1/2,j}\right) + \left(F_{i,j+1/2} - F_{i,j-1/2}\right)\right\}, \tag{2.13}$$

where $F_{i+1/2,j}$ and $F_{i,j+1/2}$ represent the fluxes in the ith and jth direction at the cell faces $i+1/2, j$ and $i, j+1/2$, respectively. Now, in order to compute these fluxes, in the CC scheme, a parameter δ is introduced such that the value of y at the cell face $i+1/2, j$ is given by $y_{i+1/2,j} = \delta_i^j y_{i,j} + \left(1-\delta_i^j\right) y_{i+1,j}$. Similarly for the face $i, j+1/2$, we have $y_{i,j+1/2} = \delta_j^i y_{i,j} + \left(1-\delta_j^i\right) y_{i,j+1}$. Using this interpolation of y, we obtain

$$F_{i+1/2,j} = \left[\frac{\varepsilon}{h} - (1-\delta_i^j) b_{i+\frac{1}{2},j}\right] y_{i+1,j} - \left[\frac{\varepsilon}{h} + \delta_i^j b_{i+\frac{1}{2},j}\right] y_{i,j}, \tag{2.14}$$

$$F_{i-1/2,j} = \left[\frac{\varepsilon}{h} - (1-\delta_{i-1}^j) b_{i-\frac{1}{2},j}\right] y_{i,j} - \left[\frac{\varepsilon}{h} + \delta_{i-1}^j b_{i-\frac{1}{2},j}\right] y_{i-1,j}, \tag{2.15}$$

$$F_{i,j+1/2} = \left[\frac{\varepsilon}{h} - (1-\delta_j^i) b_{i,j+\frac{1}{2}}\right] y_{i,j+1} - \left[\frac{\varepsilon}{h} + \delta_j^i b_{i,j+\frac{1}{2}}\right] y_{i,j}, \tag{2.16}$$

$$F_{i,j-1/2} = \left[\frac{\varepsilon}{h} - (1-\delta_{j-1}^i) b_{i,j-\frac{1}{2}}\right] y_{i,j} - \left[\frac{\varepsilon}{h} + \delta_{j-1}^i b_{i,j-\frac{1}{2}}\right] y_{i,j-1}, \tag{2.17}$$

In the CC scheme, the interpolation parameter δ is given by

$$\delta_i^j = \frac{1}{\omega_i^j} - \frac{1}{\exp\left(\omega_i^j\right) - 1}, \quad \omega_i^j = h\frac{b_{i,j}}{\varepsilon},$$
$$\delta_j^i = \frac{1}{\omega_j^i} - \frac{1}{\exp\left(\omega_j^i\right) - 1}, \quad \omega_j^i = h\frac{b_{i,j}}{\varepsilon}.$$

Remark 11. *The CC scheme is a second-order accurate monotone upwinding method for all $\epsilon > 0$ and any smooth function b. Notice that $\delta_i^j \to 1/2$ as $\omega_i^j \to 0$; $\delta_i^j \to 0$ as $\omega_i^j \to +\infty$ and $\delta_i^j \to 1$ as $\omega_i^j \to -\infty$ and likewise in the other direction $\delta_j^i \to 1/2$ as $\omega_j^i \to 0$; $\delta_j^i \to 0$ as $\omega_j^i \to +\infty$ and $\delta_j^i \to 1$ as $\omega_j^i \to -\infty$.*

Now, we combine the CC scheme and the quadrature rule to approximate our governing model as follows (notice that the solution y is zero at the boundary)

$$-\hat{A}_{ij} y_{i+1,j} - \hat{B}_{ij} y_{i-1,j} + \hat{C}_{ij} y_{i,j} - \hat{D}_{ij} y_{i,j+1} - \hat{E}_{ij} y_{i,j-1} + h^2 \sum_{l=2}^{N-1}\sum_{m=2}^{N-1} r(l,m) k(x_{ij}, z_{lm}) y_{l,m} = f_{i,j}, \tag{2.18}$$

where $i, j = 2, \ldots, N-1$. When considering the trapezoidal rule for the integral term approximation, we refer to this scheme as the CCT scheme.

The coefficients in (2.18) are given by

$$\hat{A}_{ij} = \frac{1}{h}\left[\frac{\varepsilon}{h} - (1-\delta_i^j)b_{i+\frac{1}{2},j}\right], \qquad \hat{B}_{ij} = \frac{1}{h}\left[\frac{\varepsilon}{h} + \delta_{i-1}^j b_{i-\frac{1}{2},j}\right],$$

$$\hat{C}_{ij} = \frac{1}{h}\left[\frac{\varepsilon}{h} + \delta_i^j b_{i+\frac{1}{2},j}\right] + \frac{1}{h}\left[\frac{\varepsilon}{h} - (1-\delta_{i-1}^j)b_{i-\frac{1}{2},j}\right] + \frac{1}{h}\left[\frac{\varepsilon}{h} + \delta_j^i b_{i,j+\frac{1}{2}}\right] + \frac{1}{h}\left[\frac{\varepsilon}{h} - (1-\delta_{j-1}^i)b_{i,j-\frac{1}{2}}\right] + \lambda,$$

$$\hat{D}_{ij} = \frac{1}{h}\left[\frac{\varepsilon}{h} - (1-\delta_j^i)b_{i,j+\frac{1}{2}}\right], \qquad \hat{E}_{ij} = \frac{1}{h}\left[\frac{\varepsilon}{h} + \delta_{j-1}^i b_{i,j-\frac{1}{2}}\right].$$

In the following, we denote with $\mathcal{C}^h$ the matrix of coefficients resulting from the CC discretization of the differential part of our PIDE operator. Therefore we have

$$\left(\mathcal{C}^h y_h\right)_{ij} = -\hat{A}_{ij}y_{i+1,j} - \hat{B}_{ij}y_{i-1,j} + \hat{C}_{ij}y_{i,j} - \hat{D}_{ij}y_{i,j+1} - \hat{E}_{ij}y_{i,j-1}. \tag{2.19}$$

With this notation, the algebraic problem given by (2.18) becomes $\left(\mathcal{C}^h + \mathcal{I}^h\right) y_h = f_h$. For ease of notation , we define $\mathcal{A}^h = \mathcal{C}^h + \mathcal{I}^h$ and discuss the solution to

$$\mathcal{A}^h\, y_h = f_h, \tag{2.20}$$

where $\mathcal{A}^h$ is a $(N-1)^2 \times (N-1)^2$ block dense matrix.

Next, we discuss the accuracy and stability of our CCT discretization scheme. For this analysis, we focus on a one dimensional case and notice that its extension to multi dimensions is straightforward. For simplicity, we assume that the PIDE solution $y \in C^4([a,b])$, $f \in C([a,b])$, and that the kernel k is continuous. Since $b \in C^1([a,b])$, we have that the drift coefficient b is Lipschitz continuous, i.e. $|b(x+h) - b(x)| \leq \rho\, h$ where ρ is the Lipschitz constant. We denote with $e_h = y - y_h$ the discretization error in Ω_h, where y represents the solution of the PIDE problem and y_h is its numerical approximation. The operator performing the CC convex combination between cell-centred values is defined as follows

$$E_\delta y_i = (1-\delta_{i-1})y_i + \delta_{i-1}y_{i-1}.$$

Theorem 12. *If $\lambda \geq \frac{\rho}{2}$, then the CCT scheme $\mathcal{A}^h y_h = f_h$ is stable in the sense that there exists a constant $c > 0$ such that the following holds*

$$\|y_h\|_{1,h} \leq \frac{1}{c}\,\|f_h\|_h. \tag{2.21}$$

Proof. The CCT scheme can be written as follows (for simplicity, we drop the

index h in y_h)

$$-\varepsilon D^+D^-y_i + D^+\left(bE_\delta y\right)_i + \left(\mathcal{I}^h y\right)_i + \lambda y_i = f_i, \qquad i = 2,\dots,N-1. \tag{2.22}$$

We have

$$-\varepsilon\, D^+D^-\, y_i = -\frac{1}{h}\left[\frac{\varepsilon}{h}(y_{i+1}-y_i) - \frac{\varepsilon}{h}(y_i - y_{i-1})\right],$$

and

$$\begin{aligned} D^+\left(bE_\delta y\right)_i &= D^+\left((1-\delta_{i-1})b_{i-\frac{1}{2}}y_i + \delta_{i-1}b_{i-\frac{1}{2}}y_{i-1}\right), \\ &= \frac{1}{h}\left((1-\delta_i)b_{i+\frac{1}{2}}y_{i+1} - (1-\delta_{i-1})b_{i-\frac{1}{2}}y_i\right) + \frac{1}{h}\left(\delta_i b_{i+\frac{1}{2}}y_i - \delta_{i-1}b_{i-\frac{1}{2}}y_{i-1}\right). \end{aligned}$$

Taking the L_h^2 inner product of equation (2.22) with y, we obtain

$$-\varepsilon\,\left(D^+D^-y, y\right)_h + \left(D^+\left(bE_\delta y\right), y\right)_h + \lambda\,\left(y,y\right)_h + \left(\mathcal{I}y, y\right)_h = \left(f, y\right)_h.$$

The first term of this equation results in the following [47]

$$-\varepsilon\,\left(D^+D^-y, y\right)_h = \varepsilon\,\|D^-y]|_h^2.$$

The second term results in the following

$$\begin{aligned} \left(D^+\left(bE_\delta y\right), y\right)_h =& \sum_{i=0}^{N}\left((1-\delta_i)b_{i+\frac{1}{2}}y_{i+1}y_i - (1-\delta_{i-1})b_{i-\frac{1}{2}}y_iy_i\right) \\ &+ \left(\delta_i b_{i+\frac{1}{2}}y_iy_i - \delta_{i-1}b_{i-\frac{1}{2}}y_{i-1}y_i\right), \\ =& \sum_{i=1}^{N-1}(1-\delta_i)b_{i+\frac{1}{2}}y_{i+1}y_i - \sum_{i=1}^{N-1}b_{i+\frac{1}{2}}y_iy_i + \sum_{i=1}^{N-1}\delta_{i-1}b_{i-\frac{1}{2}}(y_i)^2 + \sum_{i=1}^{N-1}\delta_i b_{i+\frac{1}{2}}(y_i)^2 \\ &- \sum_{i=1}^{N-1}\delta_{i-1}b_{i-\frac{1}{2}}y_{i-1}y_i. \end{aligned}$$

Because of the homogeneous Dirichlet boundary conditions ($y_0 = 0$, $y_N = 0$) and shifting indices, we obtain the following

$$\begin{aligned} \left(D^+\left(bE_\delta y\right), y\right)_h =& \sum_{i=1}^{N-2}(1-2\delta_i)b_{i+\frac{1}{2}}y_{i+1}y_i + \sum_{i=1}^{N-2}\delta_i b_{i+\frac{1}{2}}y_iy_i + \sum_{i=1}^{N-2}(\delta_i-1)b_{i+\frac{1}{2}}y_{i+1}y_{i+1} \\ &+ \delta_{N-1}b_{N-\frac{1}{2}}(y_{N-1})^2 - (\delta_0-1)\,b_{\frac{1}{2}}(y_1)^2. \end{aligned}$$

Using the Cauchy inequality, we obtain the following inequality

$$\begin{aligned}\left(D^{+}\left(bE_{\delta}y\right),y\right)_{h} &\leq \sum_{i=1}^{N-2}\left[y_{i}^{2}+y_{i+1}^{2}\right]\left(\frac{1-2\delta_{i}}{2}\right)b_{i+\frac{1}{2}}+\sum_{i=1}^{N-2}\delta_{i}b_{i+\frac{1}{2}}y_{i}y_{i}+\sum_{i=1}^{N-2}\left(\delta_{i}-1\right)b_{i+\frac{1}{2}}y_{i+1}y_{i+1}\\ &\quad+\delta_{N-1}b_{N-\frac{1}{2}}\left(y_{N-1}\right)^{2}-\left(\delta_{0}-1\right)b_{\frac{1}{2}}\left(y_{1}\right)^{2}\\ &=\sum_{i=1}^{N-2}\frac{1}{2}b_{i+\frac{1}{2}}y_{i+1}^{2}-\sum_{i=2}^{N-2}\frac{1}{2}b_{i-\frac{1}{2}}y_{i}^{2}\\ &\leq\sum_{i=1}^{N-2}\frac{1}{2}|b_{i+\frac{1}{2}}-b_{i-\frac{1}{2}}||y_{i}|^{2}=\frac{1}{2}\rho\sum_{i=1}^{N-2}|y_{i}|^{2}h=\frac{1}{2}\rho\|y\|_{h}^{2}.\end{aligned}$$

Using the results above and Lemma 5, we have

$$(\mathcal{A}^{h}y,y)_{h}\geq\varepsilon\left\|D^{-}y\right]\|_{h}^{2}+\left(\lambda-\frac{\rho}{2}\right)\|y\|_{h}^{2}.$$

Now, recall the discrete Poincaré-Friedrichs inequality [47], $c_{*}\left\|D^{-}y\right]\|_{h}^{2}\geq\|y\|_{h}^{2}$, where the Poincaré-Friedrichs coefficient is given by $c_{*}=(b-a)^{2}/2$. We obtain

$$(\mathcal{A}^{h}y,y)_{h}\geq\left(\frac{\varepsilon}{c_{*}}+\lambda-\frac{\rho}{2}\right)\|y\|_{h}^{2}.$$

We assume $\lambda-\frac{\rho}{2}\geq 0$ and define the constant $c_{0}=\frac{\varepsilon}{c_{*}}+\lambda-\frac{\rho}{2}$. Hence, we have $(\mathcal{A}^{h}y,y)_{h}\geq c_{0}\|y\|_{h}^{2}$ and $(\mathcal{A}^{h}y,y)_{h}\geq\varepsilon\left\|D^{-}y\right]\|_{h}^{2}$. Thus taking $c=\min\{\varepsilon,c_{0}\}$, we obtain $(\mathcal{A}^{h}y,y)_{h}\geq c\|y\|_{1,h}^{2}$. Therefore, since $\mathcal{A}^{h}y=f$, we have

$$\|y\|_{1,h}^{2}\leq\frac{1}{c}\left(\mathcal{A}^{h}y,y\right)_{h}=\frac{1}{c}\left(f,y\right)_{h}\leq\frac{1}{c}\|f\|_{h}\|y\|_{h}\leq\frac{1}{c}\|f\|_{h}\|y\|_{1,h}.$$

That is, $\|y\|_{1,h}\leq\frac{1}{c}\|f\|_{h}$ as claimed. □

Next, we define the truncation error for the CCT scheme as follows. Let y denote the solution of the continuous PIDE problem, then the truncation error at the grid point indexed by i is given by

$$\Psi_{i}=\left[-\varepsilon\left(D^{+}D^{-}y_{i}-\partial_{xx}^{2}y_{i}\right)\right]+\left[D^{+}(b\,E_{\delta}y)_{i}-\partial_{x}(b\,y)_{i}\right]+\left[\mathcal{I}^{h}y_{i}-\mathcal{I}y_{i}\right],\quad(2.23)$$

We consider the Taylors' expansions for each differential term to determine the order of the truncation error. As seen earlier and applying the Taylors' expan-

sions, we have the following

$$\begin{aligned}
-\varepsilon\, D^+D^-\, y_i &= -\frac{1}{h}\left[\frac{\varepsilon}{h}(y_{i+1}-y_i)-\frac{\varepsilon}{h}(y_i-y_{i-1})\right], \\
&= -\varepsilon\left(\frac{1}{h}\partial_x y_i+\frac{1}{2}\partial_{xx}y_i+\frac{\partial^3}{\partial x^3}y_i+\frac{h^2}{24}\frac{\partial^4}{\partial x^4}y_i\right) \\
&\quad +\varepsilon\left(\frac{1}{h}\partial_x y_i-\frac{1}{2}\partial_{xx}y_i+\frac{h}{6}\frac{\partial^3}{\partial x^3}y_i-\frac{h^2}{24}\frac{\partial^4}{\partial x^4}y_i\right)+\mathcal{O}(h^3).
\end{aligned} \tag{2.24}$$

Hence

$$-\varepsilon\left(D^+D^-y_i-\partial^2_{xx}y_i\right)=-\varepsilon\frac{h^2}{2}\left(\frac{\partial^4}{\partial x^4}y_i+\frac{h^2}{6}\frac{\partial^6}{\partial x^6}y_i+\frac{h^6}{20160}\frac{\partial^8}{\partial x^8}y_i+\ldots+HOD\right), \tag{2.25}$$

where HOD represents higher order terms in the Taylors' expansion.

Similarly for the second term, we employ the Taylors' expansion and we have the following

$$\begin{aligned}
D^+\left(bE_\delta y\right)_i &= \frac{1}{h}\left((1-\delta_i)b_{i+\frac{1}{2}}y_{i+1}-(1-\delta_{i-1})b_{i-\frac{1}{2}}y_i\right)+\frac{1}{h}\left(\delta_i b_{i+\frac{1}{2}}y_i-\delta_{i-1}b_{i-\frac{1}{2}}y_{i-1}\right), \\
&= \frac{1}{h}\left(b_{i+\frac{1}{2}}\left((1-\delta_i)y_{i+1}+\delta_i y_i\right)-b_{i-\frac{1}{2}}\left((1-\delta_{i-1})y_i+\delta_{i-1}y_{i-1}\right)\right), \\
&= \frac{1}{h}y_i\left(b_{i+\frac{1}{2}}-b_{i-\frac{1}{2}}\right)+\frac{\partial}{\partial x}y_i\left((1-\delta_i)b_{i+\frac{1}{2}}+\delta_{i-1}b_{i-\frac{1}{2}}\right) \\
&+\frac{h}{2}\frac{\partial^2}{\partial x^2}y_i\left((1-\delta_i)b_{i+\frac{1}{2}}+\delta_{i-1}b_{i-\frac{1}{2}}\right)+\mathcal{O}(h^2), \\
&= y_i\frac{\partial}{\partial x}b_i+\frac{\partial}{\partial x}y_i b_i\left(1-\delta_i+\delta_{i-1}\right)+\frac{h}{2}\frac{\partial}{\partial x}\left(y_i b_i\right)\left(1-\delta_i-\delta_{i-1}\right) \\
&+\frac{\partial^2}{\partial x^2}y_i b_i\left(1-\delta_i-\delta_{i-1}\right)+\mathcal{O}(h^2).
\end{aligned}$$

Hence we have

$$D^+(b\,E_\delta y)_i-\partial_x(b\,y)_i=\frac{\partial}{\partial x}y_i b_i\left(\delta_{i-1}-\delta_i\right)+\frac{h}{2}\frac{\partial}{\partial x}\left(\frac{\partial}{\partial x}\left(y_i b_i\right)\right)\left(1-\delta_i-\delta_{i-1}\right)+\mathcal{O}(h^2), \tag{2.26}$$

and recall δ_i and ω_i are as defined previously where on expansion they are given as follows

$$\delta_i=\frac{\sum\limits_{m=1}^{\infty}\frac{\omega_i^m}{(m+1)!}}{\sum\limits_{m=1}^{\infty}\frac{\omega_i^m}{(m)!}},$$

and

$$\begin{aligned}\delta_{i-1}-\delta_i &= \frac{\sum_{m=1}^{\infty}\frac{\omega_{i-1}^m}{(m+1)!}}{\sum_{m=1}^{\infty}\frac{\omega_{i-1}^m}{(m)!}} - \frac{\sum_{m=1}^{\infty}\frac{\omega_{i}^m}{(m+1)!}}{\sum_{m=1}^{\infty}\frac{\omega_{i}^m}{(m)!}},\\ &= -\frac{\frac{1}{12}\omega_{i-1}\omega_i(\omega_{i-1}-\omega_i)+\mathcal{O}(h^4)}{\sum_{s=1}^{\infty}\sum_{t=1}^{\infty}\frac{\omega_{i-1}^s\omega_i^t}{t!s!}},\end{aligned}$$

and

$$1-\delta_i-\delta_{i-1} = -\frac{\frac{1}{12}\omega_{i-1}\omega_i(\omega_{i-1}+\omega_i)+\mathcal{O}(h^4)}{\sum_{s=1}^{\infty}\sum_{t=1}^{\infty}\frac{\omega_{i-1}^s\omega_i^t}{t!s!}},$$

$$\omega_{i-1}+\omega_i = h\left(\frac{b_i}{\varepsilon}+\frac{b_i}{\varepsilon}\right)=\mathcal{O}(h),$$

$$1-\delta_i-\delta_{i-1}=\mathcal{O}(h),$$

$$\delta_{i-1}-\delta_i=\mathcal{O}(h^2).$$

With these terms, we can now state the order of the truncation error.

Lemma 13. *If $y \in C^4([a,b])$ and $k \in C^2([a,b]) \times C^2([a,b])$, the truncation error Ψ_i is consistent of order 2 as follows*

$$\|\Psi\|_h = \mathcal{O}(h^2).$$

Proof. The proof is similar to [30, Proposition 1]. □

Theorem 14. *Let y be the solution to the PIDE problem (2.3) and y_h be the solution to the CCT scheme (2.18), if $\lambda \geq \rho/2$, then the following holds*

$$\|y-y_h\|_{1,h} = \mathcal{O}(h^2).$$

Proof. Notice that the PIDE problem and the CCT scheme are linear. Therefore the solution error satisfies the equation $\mathcal{A}^h e_h = \Psi_h$ such that

$$\begin{cases} -\varepsilon\Delta e_h + \nabla\cdot(b\ e_h) + \mathcal{I}(e_h) + \lambda e_h &= \Psi_h,\\ e_h &= 0. \end{cases} \tag{2.27}$$

Then the proof follows from Theorem 12 and Lemma 13. □

2.2.3 A higher dimensional case

In this section, we extend the CC scheme combined with a quadrature rule (Trapezoidal or Simpson's) discretization to the convection-diffusion PIDE problem to higher spatial dimensions. We consider the three dimensional case of equation (2.3). Analogous to the two-dimensional case, we denote the mesh points $x_i = a + (i-1)h,\ i = 1,\dots,N$, $z_j = a + (j-1)h,\ j = 1,\dots,N$, and $r_k = a + (k-1)h,\ k = 1,\dots,N$. These grid points define the following grid

$$\Omega_h = \{\mathcal{Z}_{ijk} = (x_i, z_j, r_k) \in \mathbb{R}^3 : i,j,k = 2,\dots,N-1\} \cap \Omega.$$

We have the three dimensional finite volume discretization of the flux at the faces of the elementary cell centred at $\mathcal{Z}_{ijk} \in \Omega_h$ results in the following

$$\nabla \cdot F = \frac{1}{h}\left\{\left(F_{i+1/2,j,k} - F_{i-1/2,j,k}\right) + \left(F_{i,j+1/2,k} - F_{i,j-1/2,k}\right) + \left(F_{i,j,k+1/2} - F_{i,j,k-1/2}\right)\right\}, \tag{2.28}$$

where $F_{i+1/2,j,k}$, $F_{i,j+1/2,k}$ and $F_{i,j,k+1/2}$ represent the fluxes in the ith, jth and kth direction at the cell faces $i+1/2,j,k$ and $i,j+1/2,k$ and $i,j,k+1/2$ respectively. Now the parameter δ is introduced such that the value of y at the cell face $i+1/2,j,k$ is given by $y_{i+1/2,j,k} = \delta^j_{ik} y_{i,j,k} + \left(1-\delta^j_{ik}\right) y_{i+1,j,k}$. Similarly for the face $i,j+1/2,k$, we have $y_{i,j+1/2,k} = \delta^i_{jk} y_{i,j,k} + \left(1-\delta^i_{jk}\right) y_{i,j+1,k}$. Similarly for the face $i,j,k+1/2$, we have $y_{i,j,k+1/2} = \delta^k_{ij} y_{i,j,k} + \left(1-\delta^k_{ij}\right) y_{i,j,k+1}$. Using this interpolation of y, we obtain

$$F_{i+1/2,j,k} = \left[\frac{\varepsilon}{h} - (1-\delta^{jk}_i) b_{i+\frac{1}{2},j,k}\right] y_{i+1,j,k} - \left[\frac{\varepsilon}{h} + \delta^{jk}_i b_{i+\frac{1}{2},j,k}\right] y_{i,j,k}, \tag{2.29}$$

$$F_{i-1/2,j,k} = \left[\frac{\varepsilon}{h} - (1-\delta^{jk}_{i-1}) b_{i-\frac{1}{2},j,k}\right] y_{i,j,k} - \left[\frac{\varepsilon}{h} + \delta^{jk}_{i-1} b_{i-\frac{1}{2},j,k}\right] y_{i-1,j,k}, \tag{2.30}$$

$$F_{i,j+1/2,k} = \left[\frac{\varepsilon}{h} - (1-\delta^{ik}_j) b_{i,j+\frac{1}{2},k}\right] y_{i,j+1,k} - \left[\frac{\varepsilon}{h} + \delta^{ik}_j b_{i,j+\frac{1}{2},k}\right] y_{i,j,k}, \tag{2.31}$$

$$F_{i,j-1/2,k} = \left[\frac{\varepsilon}{h} - (1-\delta^{ik}_{j-1}) b_{i,j-\frac{1}{2},k}\right] y_{i,j,k} - \left[\frac{\varepsilon}{h} + \delta^{ik}_{j-1} b_{i,j-\frac{1}{2},k}\right] y_{i,j-1,k}, \tag{2.32}$$

$$F_{i,j,k+1/2} = \left[\frac{\varepsilon}{h} - (1-\delta^{ij}_k) b_{i,j,k+\frac{1}{2}}\right] y_{i,j,k+1} - \left[\frac{\varepsilon}{h} + \delta^{ij}_k b_{i,j,k+\frac{1}{2}}\right] y_{i,j,k}, \tag{2.33}$$

$$F_{i,j,k-1/2} = \left[\frac{\varepsilon}{h} - (1-\delta^{ij}_{k-1}) b_{i,j,k-\frac{1}{2}}\right] y_{i,j,k} - \left[\frac{\varepsilon}{h} + \delta^{ij}_{k-1} b_{i,j,k-\frac{1}{2}}\right] y_{i,j,k-1}. \tag{2.34}$$

In the CC scheme, the interpolation parameter δ is given by

$$\delta_i^{jk} = \frac{1}{\omega_i^{jk}} - \frac{1}{\exp\left(\omega_i^{jk}\right) - 1}, \quad \omega_i^{jk} = h\frac{b_{i,j,k}}{\varepsilon},$$
$$\delta_j^{ik} = \frac{1}{\omega_j^{ik}} - \frac{1}{\exp\left(\omega_j^{ik}\right) - 1}, \quad \omega_j^{ik} = h\frac{b_{i,j,k}}{\varepsilon},$$
$$\delta_k^{ij} = \frac{1}{\omega_k^{ij}} - \frac{1}{\exp\left(\omega_k^{ij}\right) - 1}, \quad \omega_k^{ij} = h\frac{b_{i,j,k}}{\varepsilon}.$$

Now, we can combine the CC scheme and a quadrature rule to approximate our governing model as follows (notice that the solution y is zero at the boundary)

$$-\hat{A}_{ijk}y_{i+1,j,k} - \hat{B}_{ijk}y_{i-1,j,k} - \hat{C}_{ijk}y_{i,j+1,k} + \hat{D}_{ijk}y_{i,j,k} - \hat{E}_{ijk}y_{i,j-1,k} - \hat{F}_{ijk}y_{i,j,k+1} - \hat{G}_{ijk}y_{i,j,k-1}$$
$$+ h^3 \sum_{l=2}^{N-1}\sum_{m=2}^{N-1}\sum_{s=2}^{N-1} r(l,m,s)k(x_{ijk}, z_{lms})y_{l,m,s} = f_{i,j,k}, \quad (2.35)$$

where $i,j,k = 2,\ldots,N-1$ and $r(l,m,s) = \tilde{r}(l)\,\tilde{r}(m)\,\tilde{r}(s)$ are the coefficients of the quadrature rule.
The coefficients in (2.35) are given by

$$\hat{A}_{ijk} = \frac{1}{h}\left[\frac{\varepsilon}{h} - (1-\delta_i^{jk})b_{i+\frac{1}{2},j,k}\right], \qquad \hat{B}_{ijk} = \frac{1}{h}\left[\frac{\varepsilon}{h} + \delta_{i-1}^{jk}b_{i-\frac{1}{2},j,k}\right],$$

$$\hat{C}_{ijk} = \frac{1}{h}\left[\frac{\varepsilon}{h} - (1-\delta_j^{ik})b_{i,j+\frac{1}{2},k}\right],$$

$$\hat{D}_{ijk} = \frac{1}{h}\left[\frac{\varepsilon}{h} + \delta_i^{jk}b_{i+\frac{1}{2},j,k}\right] + \frac{1}{h}\left[\frac{\varepsilon}{h} - (1-\delta_{i-1}^{jk})b_{i-\frac{1}{2},j,k}\right] + \frac{1}{h}\left[\frac{\varepsilon}{h} + \delta_j^{ik}b_{i,j+\frac{1}{2},k}\right]$$
$$+ \frac{1}{h}\left[\frac{\varepsilon}{h} - (1-\delta_{k-1}^{ij})b_{i,j,k-\frac{1}{2}}\right] + \frac{1}{h}\left[\frac{\varepsilon}{h} + \delta_k^{ij}b_{i,j,k+\frac{1}{2}}\right] + \frac{1}{h}\left[\frac{\varepsilon}{h} - (1-\delta_{k-1}^{ij})b_{i,j,k-\frac{1}{2}}\right] + \lambda,$$

$$\hat{E}_{ijk} = \frac{1}{h}\left[\frac{\varepsilon}{h} + \delta_{j-1}^{ik}b_{i,j-\frac{1}{2},k}\right],$$

$$\hat{F}_{ijk} = \frac{1}{h}\left[\frac{\varepsilon}{h} - (1-\delta_k^{ij})b_{i,j,k+\frac{1}{2}}\right], \qquad \hat{G}_{ijk} = \frac{1}{h}\left[\frac{\varepsilon}{h} + \delta_{k-1}^{ij}b_{i,j,k-\frac{1}{2}}\right].$$

Analogous to the aforementioned algebraic structure, the three dimensional discretization scheme is written as $\mathcal{A}^h y_h = f_h$.

2.3 Summary and remarks

In this chapter, the model PIDE problems and their discretization were discussed. In Section 2.2.1, the elliptic PIDE was discretized using finite difference scheme for the minus Laplacian operator and the integral operator was discretized using a quadrature rule. In Section 2.2.2, the convection-diffusion PIDE was discretized using a finite volume CC scheme for the differential operator and the integral term was discretized using a quadrature rule. For both problems, stability of the resulting discretization schemes was discussed and second-order accuracy of the corresponding numerical solution was proved. The case of a three-dimensional convection-diffusion PIDE problem was also discussed.

3. Multigrid methods

In this chapter, we discuss the multigrid (MG) strategy for solving the classes of PIDE considered in this thesis. We discuss in detail the fast integration technique combined with a geometric multigrid method. The key components of the MG strategy, i.e. the restriction, prolongation and smoothing procedures are discussed. The convergence analysis of the resulting algorithm is done by means of the local Fourier analysis. Comparison of theoretical convergence estimates with observed convergence factors confirm the optimal computational complexity of the proposed multigrid techniques.

The development of multigrid methods started in the sixties with the work of R. P. Fedorenko [25, 26], considering the Poisson equation in a unit square. This was the onset of extension of the multigrid strategy to solve different partial differential equations. However, full efficiency of the multigrid approach was realized after the works [17] and [33], focusing on linear and non-linear boundary value problems, however extension of the multigrid scheme to solve Fredholm integral equations has been considered in [20]. With the foreknowledge of the application of the MG strategy to partial differential equations and Fredholm integral equations, we extend these methodologies to elliptic PIDEs. In the following section, we discuss the main components of a geometric multigrid scheme.

3.1 The multigrid method for differential problems

In this section, we discuss all components of a (geometric) multigrid method for solving elliptic PDE problems. We start by discussing a standard iterative Gauss-Seidel scheme. Even though this scheme is characterised by global poor convergence rates, it provides rapid damping of high frequency solution errors leaving smooth, longer wave-length errors. The multigrid method is based on two complementary schemes. Appropriate iterative methods are used to reduce the high frequency components of the solution error. For this reason, an iterative scheme such as the Gauss-Seidel method are referred to as smoothers. A coarse grid correction scheme is designed to remove the low frequency error compo-

nents of the solution error. Detailed expositions on the MG strategy can be found in [16, 33, 65, 68]. To illustrate the smoothing procedure, we consider the discretization of a differential problem given by $\mathcal{A}y = f$ in Ω. We define a sequence of grids with mesh sizes $\{h_k\}_{k=0}$ generating a grid hierarchy $\Omega_k := \Omega_{h_k}$ on Ω. The operators and variables on the grid Ω_k are indexed with the level number k, $k = 1, \ldots, l$, where l denotes the finest level and that $h_{k-1} = 2h_k$, h_0 is given. Further, we specialise in the subsequent sections that Ω_h denotes a fine grid and Ω_H denotes the next coarse grid, where $H = 2h$. In a one-dimensional case, where $\Omega = (a, b)$, we denote with $x_j = a + (j-1)h$ the grid points on the grid with mesh size h, $j = 1, \ldots, N$, $h = \frac{b-a}{N-1}$, and with $x_J = a + (J-1)H$ the grid points on the grid with mesh size H, $J = 1, \ldots, \frac{N+1}{2}$.

To outline and analyze the structure of the multigrid method, we consider a numerical approximation of the differential problem. Let us index with k the resulting operators and variables defined on a grid with mesh sizes $h = h_k$, $k = 1, \ldots, l$. On each level we have a problem of the form $\mathcal{A}^k y_k = f_k$ in Ω_k.

A possible choice for solving the system $\mathcal{A}^k y_k = f_k$ is by the use of a linear iterative scheme as follows

$$y_k^{(m)} = S_k(y_k^{(m-1)}, f_k),$$

where m denotes the number of iteration steps starting with an initial given $y_k^{(0)}$. To formulate this iterative method, and considering the classical splitting $\mathcal{A}^k = D_k + L_k + U_k$, where D_k is the diagonal matrix, L_k is the lower triangular matrix and U_k is the upper triangular matrix of $\mathcal{A}^k$ is used. The Gauss-Seidel smoothing procedure S results as follows

$$S_k(y_k^{(m)}, f_k) = M_k y_k^{(m-1)} + N_k f_k, \tag{3.1}$$

with the iteration matrix $M_k = -(D_k + L_k)^{-1} U_k$ and $N_k = (D_k + L_k)^{-1}$.
The iteration matrix is the amplification matrix of the solution error as $e_k^{(m)} := y^* - y_k^{(m)}$, with $y^* = \left(\mathcal{A}^k\right)^{-1} f_k$ being the exact solution. The iteration (3.1) is equivalent to $e_k^{(m)} = M_k e_k^{(m-1)}$. We need the following definitions to discuss the convergence of an iterative scheme.

Theorem 15. *A linear iterative method $S_k(y_k^{(m)}, f_k) = M_k y_k^{(m-1)} + N_k f_k$ converges for all initial values $y_k^{(0)}$ to the solution y^* of $\mathcal{A}^k y_k = f_k$ iff the following spectral radius condition holds*

$$\rho(M_k) < 1.$$

The main purpose of using an iterative procedure in the multigrid algorithm is its ability to smooth the high frequency components of the solution error. To illustrate this fact in the case of the Gauss-Seidel scheme, we consider the simplest

differential problem given by

$$-y''(x) = f(x) \quad \text{in } (a,b),\ y(a) = 0,\ y(b) = 0.$$

The finite-difference discretization of this scheme is given by

$$-\frac{y_{j+1} - 2y_j + y_{j-1}}{h^2} = f_j \quad j = 2, \ldots, N-1,$$

where $y_1 = 0$ and $y_N = 0$.

The Gauss-Seidel scheme applied to this problem results in

$$e_{j+1}^m - 2e_j^{m+1} + e_{j-1}^{m+1} = 0.$$

Now, in the LFA framework we represent the error as $e_j^m = \sum_\theta E_\theta^m e^{i\theta j}$. Therefore we have

$$\mu_\theta = \left| \frac{E_\theta^{m+1}}{E_\theta^m} \right| = \frac{e^{i\theta}}{2 - e^{-i\theta}}, \quad \theta \in (-\pi, \pi],$$

which means that the amplitude of the solution error with frequency θ is reduced by a factor μ_θ by a Gauss-Seidel sweep. In particular, as we discuss in detail in Section 3.3, we obtain that $\mu_\theta < 0.5$ for the high frequencies $\frac{\pi}{2} \leq |\theta| \leq \pi$ and independently of the mesh sizes. It is this property that we refer to as the smoothing property.

A second essential component of the MG method is the transfer operators. The transfer operators are used to transfer smooth functions between different grids. These operators involve transfer from fine grids to coarse grids and from coarse grids back to fine grids. They are interpolation and restriction operators.

An interpolation operator, $\mathbb{I}_H^h$, transfers a function y to the finer grid Ω_h knowing only its values on the coarse grid Ω_H. To construct finite difference coarse-to-fine transfer operators, we consider the Lagrange interpolation given as follows

$$L_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{N} \frac{x - x_j}{x_i - x_j},$$

$$P(x) = \sum_{i=0}^{N} y_i L_i(x),$$

$$|y(x) - P(x)| \leq (N+1)!^{-1} \prod_{i=0} |x - x_i| \max_{\xi \in [a,b]} |y^{(N+1)}(\xi)|,$$

where $P(x)$ denotes a polynomial of degree $(N-1)$ that approximates the function y, knowing the values y_i on N discrete points. Notice that every element of

the product $\prod_{i=0}|x - x_i|$ is a multiple of the mesh size h and so the error of the interpolation is proportional to h^N. That is $y(x) = P(x) + \mathcal{O}(h^N)$. We consider the symmetric and asymmetric Langrange interpolation of the second and the fourth order. We assume nested grids, and therefore the coarse grid points coincide with the fine grid points. The values of these coarse grid points are assigned directly on the corresponding finer grid points and the values of the in-between points are calculated by computing the Lagrange coefficients of the interpolation schemes.

The second order interpolation corresponds to linear interpolation. The scheme is defined by

$$y(x_i + h) = \frac{1}{2}\left(y(x_i) + y(x_i + H)\right) \text{ or } \mathbb{I}_H^h := \frac{1}{2}\begin{bmatrix} 1 & 1 \end{bmatrix}.$$

For higher orders, more grid points are accessed and the function is approximated by a polynomial of higher degree. The interpolation scheme of the fourth order in stencil form is given by

$$\mathbb{I}_H^h := \frac{1}{16}\begin{bmatrix} -1 & 9 & 9 & -1 \end{bmatrix}. \tag{3.2}$$

The scheme equation (3.2) is symmetric and needs two grid points on both sides of the interpolated grid point. All the grid points can be used except the ones next to the boundary. To approximate the grid points next to the boundary, we need a scheme that accesses one grid point on one side and three other grid points on the other side as illustrated by Figure 3.1. This is referred to as the asymmetric interpolation and in stencil form is given by

$$\mathbb{I}_H^h := \frac{1}{16}\begin{bmatrix} 5 & 15 & -5 & 1 \end{bmatrix}. \tag{3.3}$$

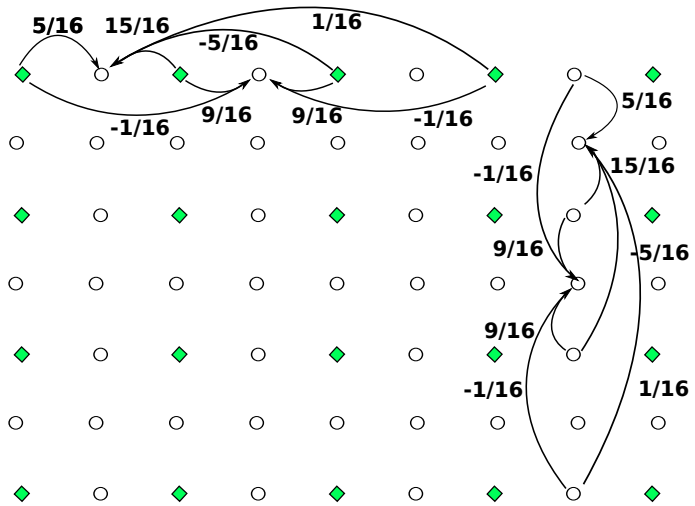


Figure 3.1: Illustration of the Fourth order interpolation on a mesh grid.

Next, we discuss the restriction operator, $\mathbb{I}_h^H$. The restriction operator maps fine grid functions to coarse grid functions. In some cases it is defined as the adjoint of the interpolation operator and we assume that $\mathbb{I}_h^H = c\left(\mathbb{I}_H^h\right)^T$ for a constant $c > 0$. This assumption holds for $\mathbb{I}_h^H$ being full-weighting restriction and $\mathbb{I}_H^h$ being a linear interpolation and $c = \left(\frac{h}{H}\right)^{dim}$ with dim being the spacial dimension.

The simplest restriction operator is the direct injection, which transfers only the values of the nested points as shown in Figure 3.2.

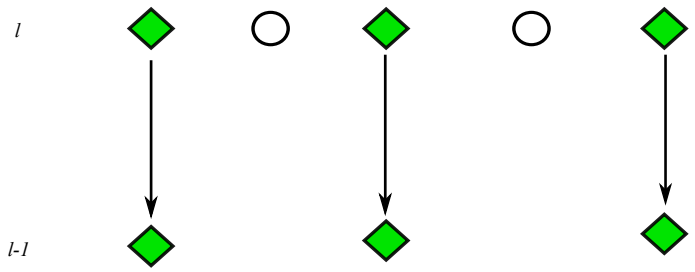


Figure 3.2: Illustration of straight injection for levels l and $l-1$.

The commonly used restriction is the weighted restriction, that is directly connected to the linear interpolation and it is given in stencil form by $(dim = 1)$

$$\mathbb{I}_h^H = \frac{1}{2}\left(\mathbb{I}_H^h\right)^T := \frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \end{bmatrix}. \tag{3.4}$$

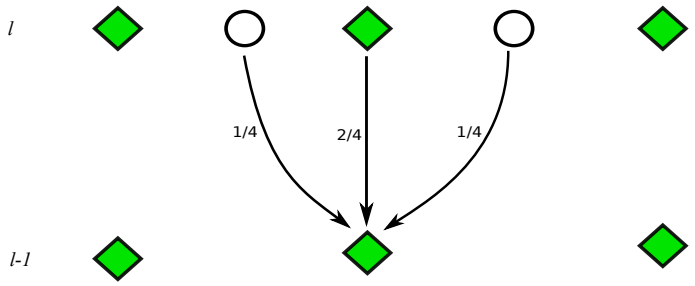


Figure 3.3: Second order restriction for levels l and $l-1$.

The fourth order restriction, accesses more grid points and it is the adjoint of the fourth order interpolation scheme and it is given in stencil form by

$$\mathbb{I}_h^H = \frac{1}{2}\left(\mathbb{I}_H^h\right)^T := \frac{1}{32}\begin{bmatrix} -1 & 0 & 9 & 16 & 9 & 0 & -1 \end{bmatrix} \tag{3.5}$$

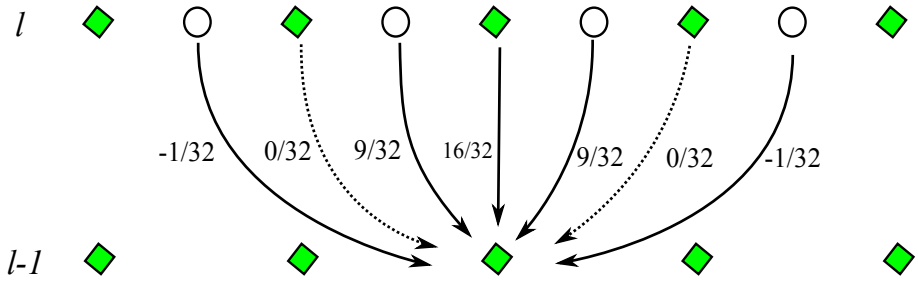


Figure 3.4: Fourth order restriction for levels l and $l-1$.

Notice that the second and the fourth order transfer operators are used interchangeably to optimize the performance of the algorithms.

Next, we combine the smoothing procedure and the transfer operators, to construct the multigrid cycle.
In general, a multigrid algorithm consists of the following four elements;

1. Smoothing of the high frequency error components via an iterative method;
2. Approximation of the smooth errors on the coarse grid;
3. Recursive application of (1) and (2) on the sequence of grids as illustrated in Figure 3.5, until a given stopping criterion is satisfied,
4. A coarse grid correction procedure.

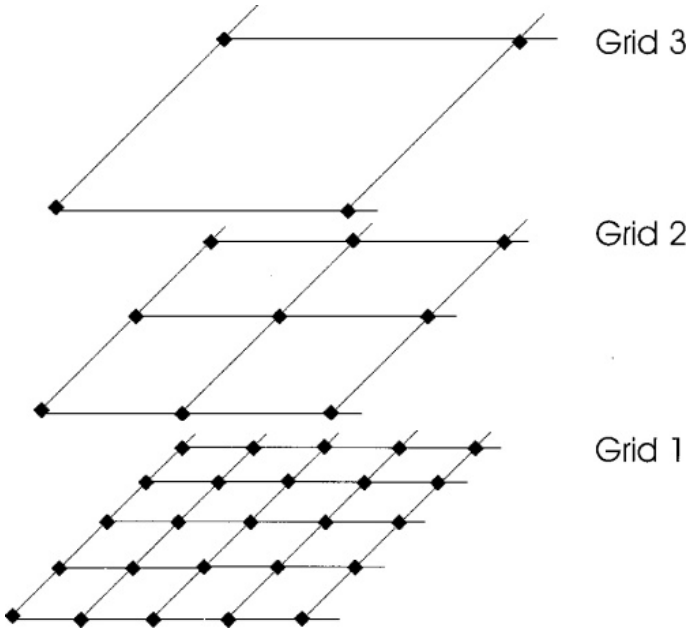


Figure 3.5: A set of nested grids with different mesh sizes. Source: S. Botello [1].

[1] https://www.osapublishing.org

Our multigrid solution procedure for solving the discrete PIDE problem (2.10) is based on the full approximation storage (FAS) framework [17,33,65,68] and the multigrid fast integration technique presented in [20]. We focus on the nonlinear FAS framework in view of future applications (nonlinear problems, differential inequalities).

To illustrate our FAS multigrid strategy, we first focus on the two-grid case, which involves the fine grid Ω_h and the coarse grid Ω_H, where $H = 2h$.

In Ω_h, we consider the discretized PIDE equation (2.10) as follows

$$\mathcal{A}^h y_h = f_h, \tag{3.6}$$

where y_h denotes the solution to this problem.

The main idea of any multigrid strategy for solving (3.6) is to combine a basic iterative method that is efficient in reducing short-wavelength errors of the approximate solution to (3.6), with a coarse-grid correction of the fine-grid long-wavelength solution's errors that is obtained solving a coarse problem.

We denote the smoothing scheme with S. Specifically, when S is applied to (3.6), with a starting approximation y_h^{m-1}, it results in $y_h^m = S(y_h^{m-1}, f_h)$. The smoothing property is such that the solution error $e_h^m = y_h - y_h^m$ has smaller higher-frequency modes than the error $e_h^{m-1} = y_h - y_h^{m-1}$.

In the multigrid solution process, starting with an initial approximation y_h^0 and applying S to (3.6) m_1-times, we obtain the approximate solution $\tilde{y}_h = y_h^{m_1}$.

Now, the desired (smooth) correction e_h to $\tilde{y}_h$, that is required to obtain the exact solution, is defined by $\mathcal{A}^h(\tilde{y}_h + e_h) = f_h$. Equivalently, this correction can be defined as the solution to

$$\mathcal{A}^h(\tilde{y}_h + e_h) = r_h + \mathcal{A}^h \tilde{y}_h \ , \tag{3.7}$$

where $r_h = f_h - \mathcal{A}^h \tilde{y}_h$ is the residual associated to $\tilde{y}_h$.

Next, notice that the structure of $\mathcal{A}^h$ and the smoothness of the error function allow to represent (3.7) on the coarse grid Ω_H. On this grid, $\tilde{y}_h + e_h$ is represented in terms of coarse variables as follows

$$y_H = \hat{\mathbb{I}}_h^H \tilde{y}_h + e_H \ , \tag{3.8}$$

where $\hat{\mathbb{I}}_h^H \tilde{y}_h$ represents the restriction of $\tilde{y}_h$ to the coarse grid by means of the direct injection operator denoted with $\hat{\mathbb{I}}_h^H$.

With this preparation, it appears natural to approximate (3.7) on the coarse grid as follows

$$\mathcal{A}^H y_H = \mathbb{I}_h^H (f_h - \mathcal{A}^h \tilde{y}_h) + \mathcal{A}^H \hat{\mathbb{I}}_h^H \tilde{y}_h. \tag{3.9}$$

Notice that this equation can be re-written as $\mathcal{A}^H y_H = \mathbb{I}_h^H f_h + \beta_h^H$ where $\beta_h^H =$

$\mathcal{A}^H \hat{\mathbb{I}}_h^H \tilde{y}^h - \mathbb{I}_h^H \mathcal{A}^h \tilde{y}_h$. The term β_h^H is the so-called fine-to-coarse defect correction.

Now, suppose to solve (3.9) to obtain y_H. Then we can compute $e_H = y_H - \hat{\mathbb{I}}_h^H \tilde{y}_h$, which represents the coarse-grid approximation to e_h. Notice that while y_H and $\hat{\mathbb{I}}_h^H \tilde{y}_h$ need not to be smooth, their difference is expected to be smooth by construction, and therefore it can be accurately interpolated on the fine grid to obtain an approximation to e_h that is used to correct $\tilde{y}_h$. This procedure defines the following coarse-grid correction step

$$y_h = \tilde{y}_h + \mathbb{I}_H^h (y_H - \hat{\mathbb{I}}_h^H \tilde{y}_h). \tag{3.10}$$

In order to damp the high-frequency errors that may arise through the coarse-grid correction, a post-smoothing is applied. In Algorithm 1, we summarize the FAS multigrid strategy.

Algorithm 1 Full approximation storage (FAS) scheme

Input: $y_k^{(0)}$, $k = l$.

1. If $k = 1$ solve $\mathcal{A}^k y_k = f_k$ exactly.
2. Pre-smoothing steps: $y_k^{(m)} = S_k(y_k^{(m-1)}, f_k),\ m = 1, \ldots, m_1$;
3. Residual computation: $r_k = f_k - \mathcal{A}^k y_k^{(m_1)}$;
4. Restriction of the residual: $r_{k-1} = \mathbb{I}_k^{k-1} r^k$;
5. Set $y_{k-1} = \hat{\mathbb{I}}_k^{k-1} y_k^{(m_1)}$;
6. Set $f_{k-1} = r_{k-1} + \mathcal{A}^{k-1} y_{k-1}$;
7. Call γ times the FAS scheme to solve $\mathcal{A}^{k-1} y_{k-1} = f_{k-1}$;
8. Coarse-grid correction : $y_k^{(m_1+1)} = y_k^{(m_1)} + \mathbb{I}_{k-1}^k \left(y_{k-1} - \hat{\mathbb{I}}_k^{k-1} y_k^{(m_1)} \right)$;
9. Post-smoothing steps: $y_k^{(m)} = S_k(y_k^{(m-1)}, f_k),\ m = m_1 + 2, \ldots, m_1 + m_2 + 1$;
10. End

3.2 Multigrid fast integration technique

In the fast integration technique, partial kernel evaluation technique is used to accelerate the approximation of the integral term. For the evaluation of the integral term $\mathcal{I}y(x)$ on each of the N grid points will require $\mathcal{O}(N^2)$ operations. To reduce the complexity of this integration procedure, we perform part of the integration on coarser grids in such a way that minimal error is added of atmost the order of

the original fine grid discretization error where we exploit the smoothness of the kernel k. This is done by replacing some values of the k by interpolations from the coarser grids.

We illustrate the advantage of the partial kernel evaluation over the full kernel evaluation in the Figure 3.6. We consider the discretization on different grids with level k and N_l discrete points per dimension, dim and denoting l as the finest grid level and k as the coarsest grid level with $k \leq l$. Evaluating each and every discrete point, we get a total of $\left(2^l+1\right)^{2\,dim}$ entries. Evaluation on the coarse grid with level k we get $\left(2^k+1\right)^{2\,dim}$ entries. Depending on the depth, $depth = l-k$ where $k \leq l$, we have to evaluate

$$\frac{N_k^{2\,dim}}{N_l^{2\,dim}} = \frac{\left(2^k+1\right)^{2\,dim}}{\left(2^{k+depth}+1\right)^{2\,dim}} \approx \left(\frac{1}{4}\right)^{depth\cdot dim}, \tag{3.11}$$

of the former $N_l^{2\,dim}$ points. For example for $dim = 2$ and considering a depth=2 for $l = 5$ and $k = 3$, for the full kernel approximation we have 1.18×10^6 evaluations need to be performed and for the partial kernel evaluation we have only 6561 evaluations to be performed. It is evident that this improves the computational time. Loss of accuracy as a result of use of coarser grids is compensated by use of higher order transfer operators.

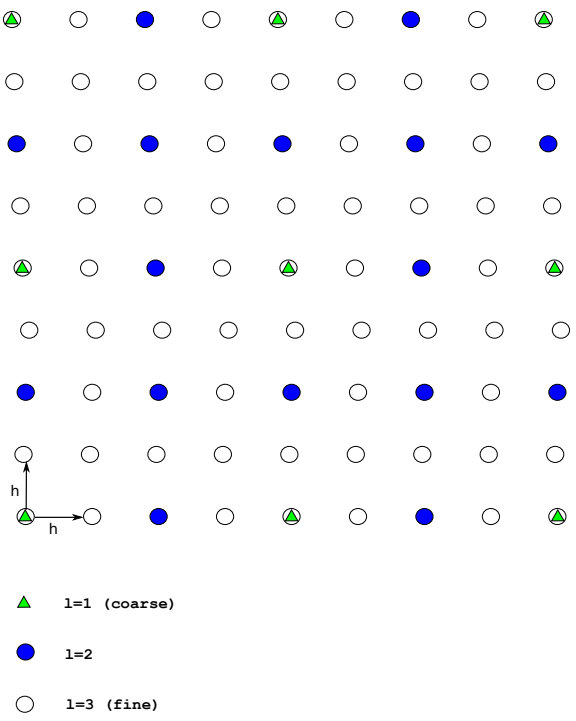


Figure 3.6: Kernel discretization on different grids Full kernel: Evaluation on $l = 3$ (81 points)
Fast integration: Evaluation on $l = 1$ (9 points) and interpolation to the other 72 points

The fast integration strategy [20, 52] aims at performing integration mostly on coarser grids and to interpolate the resulting integral function to the original fine grid where this function is required.

Now, suppose that the kernel $k(x,z)$ and y are sufficiently smooth (For the case of singular kernels see [20]). In Ω_h, the integral $\mathcal{I}y(x) = \int_\Omega k(x,z)y(z)dz$ is approximated by $(\mathcal{I}^h y)_i = h\sum_j k^{hh}_{i,j} y_j$. On the other hand, in the strategy of [20], the kernel is approximated by $\bar{k}^{hh}_{i,j} = \left[\bar{\mathbb{I}}^h_H k^{hH}_{i,\cdot}\right]_j$, where the interpolation operator $\bar{\mathbb{I}}^h_H$ may be equal to $\mathbb{I}^h_H$. With this setting, we have

$$(\mathcal{I}^h y_h)_i \approx (\bar{\mathcal{I}}^h y_h)_i = h\sum_j \bar{k}^{hh}_{i,j} y_{h,j} = h\sum_j \left[\mathbb{I}^h_H k^{hH}_{i,\cdot}\right]_j y_{h,j}$$

$$= h\sum_j k^{hH}_{i,J}\left[\left(\mathbb{I}^h_H\right)^T y_h\right]_J = H\sum_j k^{hH}_{i,J} y_{H,J} \qquad \text{where } y_H = \mathbb{I}^H_h y_h,$$

where y_H is obtained by coarsening of y_h. In particular, using straight injection and the full-weighted restriction we have $y_{H,J} = y_{h,2J-1}$. Now, we go a step further and consider the coarse integral function $(\mathcal{I}^H y_H)_I = H\sum k^{HH}_{I,J} y_{H,J}$. This function is evaluated on the coarse grid and, from the calculation above, it is clear that it is equal to $(\bar{\mathcal{I}}^h y_h)_i$ for all $i = 2I-1$. Therefore we obtain the following approximation to the integral function on the fine grid

$$\mathcal{I}^h y_h \approx \mathbb{I}^h_H(\mathcal{I}^H y_H) \qquad \text{where} \qquad y_H = \mathbb{I}^H_h y_h. \tag{3.12}$$

In one dimension, the summation complexity on the coarse grid is of order $\mathcal{O}(N^2/2)$ operations, which may still be large. However, assuming that the kernel is sufficiently smooth and using the fact that the coarse-grid summation has the same structure of the fine-grid summation, the coarsening-summation procedure just described can be applied recursively, until a grid is reached with $\mathcal{O}(\sqrt{N})$ grid points. On this grid the summation is then actually performed, requiring $\mathcal{O}(N)$ operations. Further, the computational effort of the restriction $\mathbb{I}^H_h y_h$ and of the interpolation $\mathbb{I}^h_H(\mathcal{I}^H y_H)$ is $\mathcal{O}(2pN)$, where p is the order of interpolation ($p = 2$ for linear interpolation). Therefore the order of total work required to obtain the (approximated) summation is $\mathcal{O}(N)$ operations. Moreover, we notice that using the full integration procedure with Simpson's rule on the grid with mesh size $\bar{H}$ and using fourth order transfer operators, we obtain an approximation to the integral of order $\mathcal{O}(\bar{H}^4)$ that corresponds to the accuracy $\mathcal{O}(h^2)$, since

$$\bar{H}^4 \approx \left(\frac{1}{\sqrt{N}}\right)^4 = \frac{1}{N^2} = h^2.$$

We summarize the fast integration (FI) technique in Algorithm 2, where we perform full-kernel evaluation when a level $k = l - d$, with given depth d, is reached.

Algorithm 2 Fast integration (FI) method.

Input: y_k, $k = l, d$.

1. If $size(y_k) \approx \sqrt{size(y_l)}$ (or $k = l - d$) then perform FK evaluation, $(\mathcal{I}^k y_k)_I = h_k^{dim} \sum k_{i,j}^{kk} y_{k,j}$.
2. Restriction: $y_{k-1} = \mathbb{I}_k^{k-1} y_k$;
3. Call the FI scheme to compute $(\mathcal{I}^{k-1} y_{k-1})$;
4. Interpolation: $\mathbb{I}_{k-1}^k (\mathcal{I}^{k-1} y_{k-1})$
5. End

Our approach is to implement a Gauss-Seidel step for the differential operator, without updating the integral part of our PIDE operator. It can be appropriately called a Gauss-Seidel-Picard (GSP) iteration, where the integral is evaluated using the FI scheme before the Gauss-Seidel step starts.

In the one-dimensional case of (2.1), our smoothing scheme is given by Algorithm 3.

Algorithm 3 Gauss-Seidel-Picard (GSP) scheme

Input: $y_k^{(0)}$; $\mathcal{I}^k y_k^{(0)}$.

1. for $i = 2 : N_k - 1$
2. Compute the dynamic residual:

$$R_{k,i} = \left(f_{k,i} + \frac{1}{h_k^2} \left(y_{k,i+1}^{(0)} - 2y_{k,i}^{(0)} + y_{k,i-1} \right) - (\mathcal{I}^k y_k^{(0)})_i \right)$$

3. Compute: $y_{k,i} = y_{k,i}^{(0)} + \left(\frac{h^2}{2} \right) R_{k,i}$
4. end

Our multigrid scheme for PIDE problems is given in Algorithm 4. Notice that this algorithm describes one cycle of the multigrid procedure that is repeated many times until a convergence criterion is satisfied. In Algorithm 4, the parameter γ is called the cycle index and it is the number of times the same multigrid procedure is applied to the coarse level. A V-cycle occurs when $\gamma = 1$ and a W-cycle results when $\gamma = 2$.

Algorithm 4 Full approximation storage fast integration (FAS-FI) scheme

Input: $y_k^{(0)}$, $k = l$.

1. If $k = 1$ solve $\mathcal{A}^k y_k = f_k$ exactly.
2. Perform fast integration to approx. evaluate $\mathcal{I}^k y_k^{(0)}$;
3. Pre-smoothing steps: $y_k^{(m)} = S_k(y_k^{(m-1)}, f_k),\ m = 1, \ldots, m_1$;
4. Perform fast integration to approx. evaluate $\mathcal{I}^k y_k^{(m_1)}$;
5. Residual computation: $r_k = f_k - \mathcal{A}^k y_k^{(m_1)}$;
6. Restriction of the residual: $r_{k-1} = \mathbb{I}_k^{k-1} r^k$;
7. Set $y_{k-1} = \hat{\mathbb{I}}_k^{k-1} y_k^{(m_1)}$;
8. Perform fast integration to approx. evaluate $\mathcal{I}^k y_{k-1}$;
9. Set $f_{k-1} = r_{k-1} + \mathcal{A}^{k-1} y_{k-1}$;
10. Call γ times the FAS scheme to solve $\mathcal{A}^{k-1} y_{k-1} = f_{k-1}$;
11. Coarse-grid correction : $y_k^{(m_1+1)} = y_k^{(m_1)} + \mathbb{I}_{k-1}^k \left(y_{k-1} - \hat{\mathbb{I}}_k^{k-1} y_k^{(m_1)} \right)$;
12. Perform fast integration to approx. evaluate $\mathcal{I}^k y_k^{(m_1+1)}$;
13. Post-smoothing steps: $y_k^{(m)} = S_k(y_k^{(m-1)}, f_k),\ m = m_1 + 2, \ldots, m_1 + m_2 + 1$;
14. End

Since Algorithm 4, represent one multigrid cycle, in the following Algorithm 5, we summarize the multigrid iterative procedure, where we use a given residual tolerance as the stopping criterion and $Num_{maxiter}$ being the maximum number of V-cycles to be performed.

Algorithm 5 Multigrid iterative procedure.

Input: $y^{(0)}$, residual tolerance, k, l, d, $Num_{maxiter}$, m_1 and m_2, V-cycle=1.

1. **while** stopping criterion is not satisfied and V-cycle $\leq Num_{maxiter}$**do**
2. call FAS-FI procedure
3. V-cycle=V-cycle+1
4. **endwhile**

3.3 Local Fourier analysis

In this section, we investigate the convergence of the two-grid version of our FAS-FI multigrid solution procedure using local Fourier analysis (LFA) [17, 19, 65, 69]. From the early day of multigrid development, A. Brandt introduced a tool for analyzing a multigrid process, the so called *local mode analysis* or the *local Fourier analysis* . It has proved to be beneficial in developing, analysing and debugging multigrid algorithms. In its fundamental form, it is used to analyze quantitatively the smoothing procedures. This smoothing analysis allows judgement to be made as to whether a given iterative scheme is suitable as a smoothing routine within a multigrid framework.

In order to ease notation, we consider a one-dimensional case and use h and H indices to denote variables on the fine and coarse grids, respectively. For the LFA investigation, we assume that the kernel of the integral term is translational invariant in the sense that $k(x,z) = k(|x-z|)$ and require that $k(|x-z|)$ decays rapidly to zero as $|x-z|$ becomes large. With these assumptions the stencil of our PIDE operator can be cast in the standard LFA framework. However, treating the fast-kernel evaluation in this framework proves to be too cumbersome. On the other hand, numerical experiments show that, apart of the different complexity, the convergence of our multigrid scheme with FI and with FK evaluation are very similar. Therefore we analyze our two-grid scheme with the latter procedure.

We apply the local Fourier analysis to the two-grid operator for $\mathcal{A}^h y_h = f_h$. It is given by

$$TG_h^H = S_h^{m_2} \left[\mathbb{I}_h - \mathbb{I}_H^h \left(\mathcal{A}^H\right)^{-1} \mathbb{I}_h^H \mathcal{A}^h \right] S_h^{m_1}, \tag{3.13}$$

where m_1 pre- and m_2 post-smoothing steps are considered. The coarse grid operator is given by $CG_h^H = \left[\mathbb{I}_h - \mathbb{I}_H^h \left(\mathcal{A}^H\right)^{-1} \mathbb{I}_h^H \mathcal{A}^h \right]$. Thus for the coefficient of the θ Fourier mode in the Fourier space, this action is represented by so called Fourier symbol. The Fourier symbol corresponding to (3.13) is given by

$$\widehat{TG}_h^H(\theta) = \widehat{S}_h^{m_2}(\theta) \left[\widehat{\mathbb{I}}_h - \widehat{\mathbb{I}}_H^h(\theta) \left(\widehat{\mathcal{A}}^H(2\theta)\right)^{-1} \widehat{\mathbb{I}}_h^H(\theta) \widehat{\mathcal{A}}^h(\theta) \right] \widehat{S}_h^{m_1}(\theta). \tag{3.14}$$

The local Fourier analysis considers infinite grids, $G_h = \{jh,\, j \in \mathbb{Z}\}$, and therefore the influence of boundary conditions is not taken into account. Nevertheless, LFA is able to provide sharp estimates of multigrid convergence factors. This analysis is based on the function basis

$$\phi_h(\theta, x) = \mathrm{e}^{i\theta x/h}, \qquad \theta \in (-\pi, \pi].$$

For any low frequency $\theta^0 \in [-\pi/2, \pi/2)$, we consider the high frequency mode

given by

$$\theta^1 = \theta^0 - \text{signum}\left(\theta^0\right)\pi. \tag{3.15}$$

We have $\phi_h(\theta^0,\cdot) = \phi_h(\theta^1,\cdot)$ for $\theta^0 \in [-\pi/2,\pi/2)$ and $x \in G_H$. We also have $\phi_h(\theta,x) = \phi_H(2\theta^0,x)$ on G_H for $\theta = \theta^0$ and $\theta = \theta^1$.

The two components $\phi_h(\theta^0,\cdot)$ and $\phi_h(\theta^1,\cdot)$ are called harmonics. For a given $\theta^0 \in [-\pi/2,\pi/2)$, the two-dimensional space of harmonics is defined by

$$E_h^\theta = span[\phi_h(\theta^\alpha,\cdot) : \alpha \in \{0,1\}].$$

For each θ and a translational invariant kernel, we assume that the space E_h^θ is invariant under the action of TG_h^H. In fact in our case, the stencil of the discrete PIDE operator is defined by constant coefficients that do not depend on the choice of origin of the infinite grid. Now, we study the action of TG_h^H on the following function

$$\psi(x_j) = \sum_{\alpha,\theta} \mathcal{A}_\theta^\alpha \, \phi_h(\theta^\alpha, x_j), \qquad x_j \in G_h.$$

Specifically, we determine how the coefficients $\mathcal{A}_\theta^\alpha$, $\theta \in [-\pi/2,\pi/2)$ and $\alpha = 0,1$ are transformed under the action of the two-grid operator. This requires to calculate the Fourier symbols of the components that enter in the construction of this operator.

First, we derive the Fourier symbol of our smoothing operator. For this purpose, we introduce the following Fourier representation of the solution error before and after one smoothing step. This analysis quantitatively analyzes the reduction of high frequency error components on the fine grid on assumption of an ideal situation where

1. The smoothing does not affect the low frequencies;
2. Low frequencies are approximated well on the coarse grid;
3. There is no iteraction between high and low frequencies error components.

With these assumptions, an estimate of the convergence factor is possible. We drop the index α as we assume invariance of E_h^θ under the action of the smoothing operator. Now, the solution errors after the m, and $m+1$ iterations are given by

$$e_j^m = \sum_\theta \mathrm{E}_\theta^m \mathrm{e}^{i\theta j} \quad \text{and} \quad e_j^{m+1} = \sum_\theta \mathrm{E}_\theta^{m+1} \mathrm{e}^{i\theta j}, \tag{3.16}$$

where E_θ^m and E_θ^{m+1} denote the error amplitude after m and $m+1$ iterations of the smoother. Notice that $e^{m+1} = S_h\, e^m$, and thus for the coefficient of the θ Fourier mode in the Fourier space, this action is represented by $\mathrm{E}_\theta^{m+1} = \hat{S}_h(\theta)\,\mathrm{E}_\theta^m$, where $\hat{S}_h(\theta)$ is the so called Fourier symbol of S_h [65].

Now, consider the following point-wise definition of our GSP iteration applied to our discretized elliptic PIDE (2.1) on G_h. We have

$$\frac{2}{h^2}e_j^{m+1} - \frac{1}{h^2}e_{j-1}^{m+1} = \frac{1}{h^2}e_{j+1}^{m} - h\sum_{l=-\infty}^{\infty} k_{jl}e_l^m. \tag{3.17}$$

Notice that, we consider full kernel evaluation on infinite grids $l = -\infty$ to $l = \infty$. At this point, recall that $k(x,z) = k(|x-z|)$ and introduce the index $n = j - l$. Since we assume that the element $k_{jl} = k_{|j-l|} = k_{|n|}$ becomes very small as n becomes large, we truncate the sum in (3.17) and consider the following equation

$$\frac{2}{h^2}e_j^{m+1} - \frac{1}{h^2}e_{j-1}^{m+1} = \frac{1}{h^2}e_{j+1}^{m} - h\sum_{n=-L}^{L} k_{|n|}e_{j-n}^m. \tag{3.18}$$

where we assume that the partial sum provides a sufficiently accurate approximation of the integral term on G_h. Specifically, assuming that $k(|x-z|) = \exp(-|x-z|^2)$ and requiring that the $k_{|n|} = \mathcal{O}(10^{-16})$ (double precision machine epsilon) for $|n| > L$, one should choose $L \propto 1/h$. However, in practice, a much smaller L results in accurate LFA estimates.

Next, in (3.18), we insert (3.16) and obtain

$$\frac{2}{h^2}\sum_\theta \mathrm{E}_\theta^{m+1}\mathrm{e}^{i\theta j} - \frac{1}{h^2}\sum_\theta \mathrm{E}_\theta^{m+1}\mathrm{e}^{i\theta(j-1)} = \frac{1}{h^2}\sum_\theta \mathrm{E}_\theta^{m}\mathrm{e}^{i\theta(j+1)} - h\sum_{n=-L}^{L} k_{|n|}\sum_\theta \mathrm{E}_\theta^m \mathrm{e}^{i\theta(j-n)}, \tag{3.19}$$

which can be re-written as follows

$$\sum_\theta \mathrm{E}_\theta^{m+1}\left(\frac{2}{h^2} - \frac{1}{h^2}\mathrm{e}^{-i\theta}\right)\mathrm{e}^{i\theta j} = \sum_\theta \mathrm{E}_\theta^{m}\left(\frac{1}{h^2}\mathrm{e}^{i\theta} - h\sum_{n=-L}^{L} k_{|n|}\mathrm{e}^{-i\theta n}\right)\mathrm{e}^{i\theta j}. \tag{3.20}$$

Now, comparing the coefficients of equal frequency modes on both sides of (3.19), we obtain

$$\hat{S}_h(\theta) := \frac{\mathrm{E}_\theta^{m+1}}{\mathrm{E}_\theta^{m}} = \frac{e^{i\theta} - h^3\sum\limits_{n=-L}^{L} k_{|n|}e^{-i\theta n}}{2 - \mathrm{e}^{-i\theta}}. \tag{3.21}$$

Therefore an appropriate estimate of the smoothing factor of our GSP scheme is given by

$$\mu_{GSP} \;=\; \max_{\frac{\pi}{2}\le|\theta|\le\pi}\left(\left|\hat{S}_h(\theta)\right|\right). \tag{3.22}$$

With this definition, we obtain a smoothing factor of our GSP scheme given by $\mu_{GSP} = 0.447$ for all mesh sizes as illustrated in the Figure 3.7.

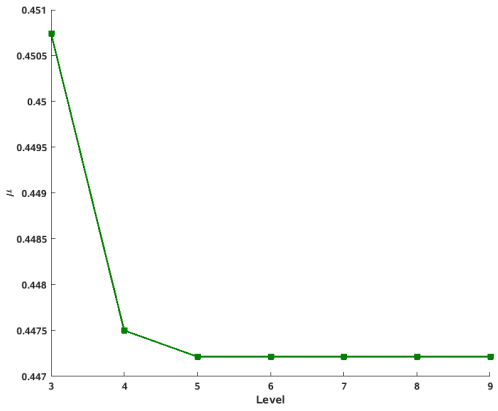


Figure 3.7: Smoothing factor μ_{GSP} for different mesh sizes $h_k,\ k = 3, \ldots, l$.

Next, in order to investigate the two-grid convergence factor, we construct the Fourier symbol of the two-grid operator. For this purpose, we derive the Fourier symbol for $\mathcal{A}^h = -\Delta^h + \mathcal{I}^h$, applied to a generic vector v with jth component given by $v_j = \sum_\theta V_\theta e^{i\theta j}$. We have

$$\mathcal{A}^h v|_j = \sum_\theta V_\theta \left(-\frac{e^{i\theta(j+1)} - 2e^{i\theta j} + e^{i\theta(j-1)}}{h^2} + h \sum_{n=-L}^{L} k_{|n|} e^{i\theta(j-n)} \right).$$

Therefore we obtain

$$\hat{\mathcal{A}}^h(\theta) = -\frac{2(1-\cos(\theta))}{h^2} + h \sum_{n=-L}^{L} k_{|n|} e^{-i\theta n}. \tag{3.23}$$

Now, recall that on the fine grid, we distinguish on the two harmonics. Therefore, we have the following operator symbols acting on the vector of the two harmonics

$$\widehat{\mathcal{A}}^h(\theta) = \begin{bmatrix} \hat{\mathcal{A}}^h(\theta^0) & 0 \\ 0 & \hat{\mathcal{A}}^h(\theta^1) \end{bmatrix} \quad \text{and} \quad \widehat{S}_h(\theta) = \begin{bmatrix} \hat{S}_h(\theta^0) & 0 \\ 0 & \hat{S}_h(\theta^1) \end{bmatrix}.$$

On the coarse grid, we have the following

$$\widehat{\mathcal{A}}^H(2\theta) = -\frac{2(1-\cos(2\theta))}{H^2} + H \sum_{n=-\frac{L}{2}}^{\frac{L}{2}} k_{|n|} e^{-i(2\theta)n}. \tag{3.24}$$

For the restriction operator since we use the $4th$-order operator, we have the fol-

lowing [69]

$$\begin{aligned} \mathbb{I}_h^H e^{i\theta j} &= \frac{-e^{i\theta(j-3)} + 9e^{i\theta(j-1)} + 16e^{i\theta j} + 9e^{i\theta(j+1)} - e^{i\theta(j+3)}}{32}, \\ &= \left(\frac{-e^{-3i\theta} + 9e^{-i\theta} + 16 + 9e^{i\theta} - e^{3i\theta}}{32}\right) e^{i\theta j}, \\ &= \left(\frac{18\cos(\theta) + 16 - 2\cos(3\theta)}{32}\right) e^{i\theta j}. \end{aligned}$$

Hence

$$\widehat{\mathbb{I}}_h^H(\theta) = \begin{bmatrix} \frac{18\cos(\theta^0)+16-2\cos(3\theta^0)}{32} & \frac{18\cos(\theta^1)+16-2\cos(3\theta^1)}{32} \end{bmatrix}.$$

For the interpolation operator, we obtain

$$\begin{aligned} \mathbb{I}_H^h e^{i\theta j} &= \frac{-e^{i\theta(j-3)} + 9e^{i\theta(j-1)} + 16e^{i\theta(j)} + 9e^{i\theta(j+1)} - e^{i\theta(j+3)}}{16}, \\ &= \left(\frac{18\cos(\theta) + 16 - 2\cos(3\theta)}{16}\right) e^{i\theta j}. \end{aligned}$$

Hence

$$\widehat{\mathbb{I}}_H^h(\theta) = \begin{bmatrix} \frac{18\cos(\theta^0)+16-2\cos(3\theta^0)}{16} \\ \frac{18\cos(\theta^1)+16-2\cos(3\theta^1)}{16} \end{bmatrix}.$$

Now, we are able to compute the two-grid convergence factor as follows

$$\eta\left(TG_h^H\right) = \sup\left\{\rho\left(\widehat{TG}_h^H(\theta)\right) : \theta \in [-\pi/2, \pi/2)\right\}, \tag{3.25}$$

where ρ denotes the spectral radius of the 2×2 matrix $\widehat{TG}_h^H(\theta)$.

In Table 3.1, we report the values of the two-grid convergence factor given by (3.25) for different numbers of pre- and post-smoothing steps, m_1, m_2. These values are computed by inspection of the function $\rho\left(\widehat{TG}_h^H(\theta)\right)$, which is evaluated using MATLAB to compute the eigenvalues of the matrix $\widehat{TG}_h^H(\theta)$ on a fine grid of θ values, $-\frac{\pi}{2} \leq \theta \leq \frac{\pi}{2}$.

Further in the same table, we compare these values with the value of the observed convergence factor given by

$$\rho = \frac{\left\|r_h^{m+1}\right\|_{L_h^2}}{\left\|r_h^m\right\|_{L_h^2}}.$$

This numerical convergence factor represents the asymptotic ratio of reduction of the L^2-norm of the residual between two multigrid cycles. These calculations refer to the choice $k(|x-z|) = \exp(-|x-z|^2)$. As shown in Table 3.1, the LFA

estimates of the multigrid convergence factor are accurate. (The same values of η_{TG} are obtained with L ranging from 20 to 400).

Table 3.1: Estimated and observed multigrid convergence factors.

m_1, m_2	1	2	3	4
η_{TG}	2.000e-1	4.000e-2	8.000e-3	1.600e-3
ρ	1.347e-1	1.353e-2	9.757e-3	8.653e-3

3.4 Numerical experiments

In this section, we present results of numerical experiments to validate our FAS-FI multigrid strategy and the LFA theoretical estimates. We demonstrate that our FAS-FI scheme has $\mathcal{O}(M \log M)$ computational complexity, where $M = N^2$ denotes the total number of grid points on the finest grid and provides second-order accurate solutions.

Our first purpose is to validate our accuracy estimates for the discretization scheme used. For this purpose, we consider an elliptic PIDE problem with a Gaussian convolution kernel in two dimensions as follows

$$-\Delta y(x) + \int_{\Omega}\int_{\Omega} k(x,z)y(z)dz = f(x), \tag{3.26}$$

where $\Omega = (-1,1) \times (-1,1)$, $x = (x_1, x_2)$, $z = (z_1, z_2)$ and the Gaussian kernel $k(x,z) = \exp\left(-\frac{(x_1-z_1)^2+(x_2-z_2)^2}{2}\right)$.

To investigate the order of accuracy of the discretization scheme, we construct an exact solution to (3.26) by choosing $y(x_1, x_2) = \exp\left(-\frac{x_1^2+x_2^2}{2}\right)$. With this choice, the right-hand side of (3.26) is given by

$$f(x_1,x_2) = 2\exp\left(-\frac{x_1^2+x_2^2}{2}\right) - (x_1^2+x_2^2)\exp^{\left(-\frac{x_1^2+x_2^2}{2}\right)} + \frac{\pi}{4}\exp^{\left(-\left(\frac{x_1^2}{4}+\frac{x_2^2}{4}\right)\right)} \times \left[\operatorname{erf}\left(\frac{x_2}{2}\right) + \operatorname{erf}\left(\frac{2-x_2}{2}\right)\right]\left[\operatorname{erf}\left(1-\frac{x_1}{2}\right) + \operatorname{erf}\left(1+\frac{x_1}{2}\right)\right].$$

The Dirichlet boundary is also given by the chosen y.

Using the exact solution above, we can validate the accuracy of our finite-differences and Simpson's quadrature schemes. In Table 3.2, we report the values of the norm of the solution errors on different grids. We obtain second-order accuracy as predicted.

M	$\|y - Y\|_h$	order of accuracy
9×9	$8.60e-4$	1.99
17×17	$2.16e-4$	2.00
33×33	$5.41e-5$	2.00
65×65	$1.35e-5$	2.00
129×129	$3.38e-6$	

Table 3.2: L_2-norm error using full kernel approximation.

Next, we investigate the FI scheme. For this purpose, we consider the integral term in (3.26), and compute the norm $\|\mathcal{I}^h y - \mathcal{I} y\|_h$. Notice that we can evaluate $\mathcal{I} y$ exactly, while $\mathcal{I}^h y$ is computed using the full-kernel (FK) evaluation formula (2.8) and the FI technique involving different depths, $d = l - k$. For $d = 0$ the FI scheme performs FK evaluation.

In Tables 3.3 and 3.4, we report the values of $\|\mathcal{I}^h y - \mathcal{I} y\|_h$ and the CPU times corresponding to different working levels l and different depths. Because we use a fourth-order quadrature formula, we can see an increase of accuracy of a factor 16 by halving the mesh size and using the FK scheme. On the other hand, increasing the depth of the FI scheme, this scaling factor deteriorates. However, since the truncation error corresponding to the Laplace operator is of second-order, the reduction of accuracy due to the use of the FI scheme with the fourth-order quadrature does not affect the overall solution accuracy of the PIDE problem as shown in Table 3.5.

M		$d=0$	$d=1$	$d=2$	$d=3$	$d=4$	$d=5$
9×9	FK	$3.78e-5$					
	FI	$3.78e-5$	$2.12e-3$				
17×17	FK	$2.54e-6$					
	FI	$2.54e-6$	$7.51e-5$	$2.29e-3$			
33×33	FK	$1.64e-7$					
	FI	$1.64e-7$	$3.89e-6$	$7.01e-5$	$2.35e-3$		
65×65	FK	$1.04e-8$					
	FI	$1.04e-8$	$2.42e-7$	$3.19e-6$	$7.01e-5$	$2.38e-3$	
129×129	FK	$6.56e-10$					
	FI	$6.56e-10$	$1.53e-8$	$1.94e-7$	$3.11e-6$	$7.05e-5$	$2.39e-3$

Table 3.3: Solution errors for 2D integral evaluation using 4^{th}-order interpolation and different depths.

M		$d=0$	$d=1$	$d=2$	$d=3$	$d=4$	$d=5$
9×9	FK	0.15					
	FI	0.21	0.06				
17×17	FK	1.99					
	FI	2.23	0.26	0.06			
33×33	FK	30.22					
	FI	31.08	2.23	0.22	0.06		
65×65	FK	462.31					
	FI	469.61	30.32	2.21	0.22	0.06	
129×129	FK	7367.35					
	FI	7299.60	460.91	30.47	2.21	0.23	0.07

Table 3.4: CPU time (secs.) for 2D integral evaluation using 4^{th}-order interpolation and different depths.

Next, we validate our FAS-FI solution procedure in solving our PIDE problem. One main issue is how the accuracy of the solution obtained with the FAS-FI scheme is affected by the approximation of the integral due to the FI procedure. For this purpose, in Table 3.5, we compare the norm of the solution errors obtained with a FAS scheme with FK calculation and with our FAS scheme including the FI technique. We see a moderate degradation of the quality of the numerical solution while increasing the depth. On the other hand, we notice that a second-accurate solution is obtained by choosing d corresponding to the first before the coarsest grid.

M		$d=0$	$d=1$	$d=2$	$d=3$	$d=4$	$d=5$
9×9	FK	$8.60e-4$					
	FI	$8.60e-4$	$8.39e-4$				
17×17	FK	$2.16e-4$					
	FI	$2.16e-4$	$2.17e-4$	$2.06e-4$			
33×33	FK	$5.41e-5$					
	FI	$5.41e-5$	$5.41e-5$	$5.51e-5$	$8.84e-5$		
65×65	FK	$1.35e-5$					
	FI	$1.35e-5$	$1.35e-5$	$1.36e-5$	$1.46e-5$	$8.64e-5$	
129×129	FK	$3.38e-6$					
	FI	$3.38e-6$	$3.36e-6$	$3.37e-6$	$3.45e-6$	$4.60e-6$	$8.88e-5$

Table 3.5: Solution errors of FAS solution with FK and FI integral evaluation after 5 V-cycles.

For the same experiments as in Table 3.5, we show large speed up in computational time in Table 3.6. Further, in Figure 3.9, we demonstrate that the computational complexity of our multigrid procedure is $\mathcal{O}(M\log M)$ and $M=N^2$ is the total number of grid points. In Figure 3.8, we depict the convergence history of the norm of the residuals at a given working level using different numbers of pre-

and post-smoothing steps, $m = 1, \ldots, 10$, $m = m_1 + m_2$, and 5 V-cycle iterations.

M		$d = 0$	$d = 1$	$d = 2$	$d = 3$	$d = 4$	$d = 5$
9×9	FK	0.98					
	FI	0.95	**0.53**				
17×17	FK	10.49					
	FI	10.36	3.27	**1.72**			
33×33	FK	162.14					
	FI	161.41	44.33	15.78	**8.63**		
65×65	FK	2568.89					
	FI	2574.11	731.88	224.03	84.37	**47.75**	
129×129	FK	41970.15					
	FI	41939.82	11237.26	3578.64	1170.72	451.10	**263.13**

Table 3.6: CPU time (secs.) of FAS-FI solution with 5 V-cycles. In bold are the values of CPU time actually involved in the multigrid solution scheme.

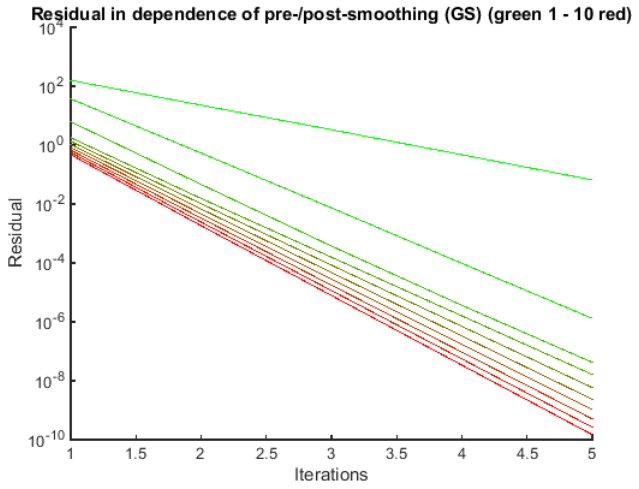


Figure 3.8: Convergence history of the FAS-FI scheme with different $m = m_1 + m_2$, $m =$ 1 (green) to $m = 10$ (red) along 5 V-cycles of FAS; $l = 8,\ d = 3$.

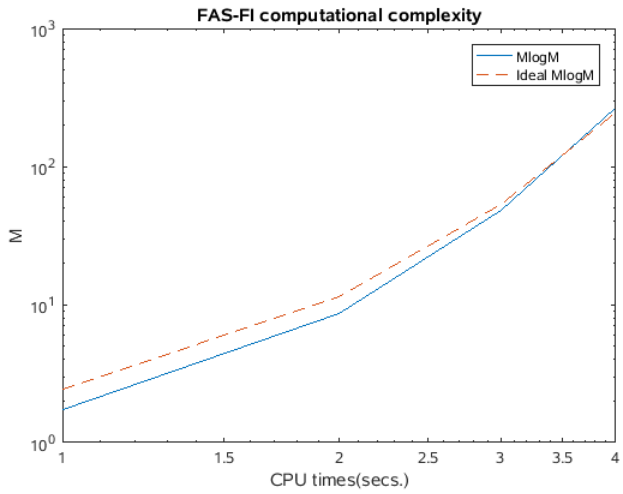


Figure 3.9: Computational complexity of the FAS-FI method; $M = N^2$.

We complete this section presenting results of experiments with a singular kernel. We consider an elliptic PIDE in one dimension given by

$$-\Delta y(x) + \int_{\Omega} \log|x - z|\, y(z)dz = f(x)$$

where $f(x) = 1$ for $x \in \Omega := (-1, 1)$. We further assume homogeneous Dirichlet boundary conditions for y.

We implement the FAS-FI scheme for this PIDE problem whose integral term has a singular kernel with one isolated singularity. On the singularity point, we cannot evaluate the kernel directly. However, we can estimate the integral using its values on neighbouring points. If the singularity is on one x_i of the grid, we use local averaging $k(x_i) \approx \frac{1}{2}\left(k\left(x_{i-\frac{1}{2}}\right) + k\left(x_{i+\frac{1}{2}}\right)\right)$. In Figure 3.11, we depict the observed multigrid computational complexity when solving the singular kernel problem and see that complexity appears to match or even improve on the typical estimate $\mathcal{O}(M \log M)$. Further, in Figure 3.11 the convergence history of the multigrid scheme with different pre- and post-smoothing schemes applied to the singular kernel case is presented.

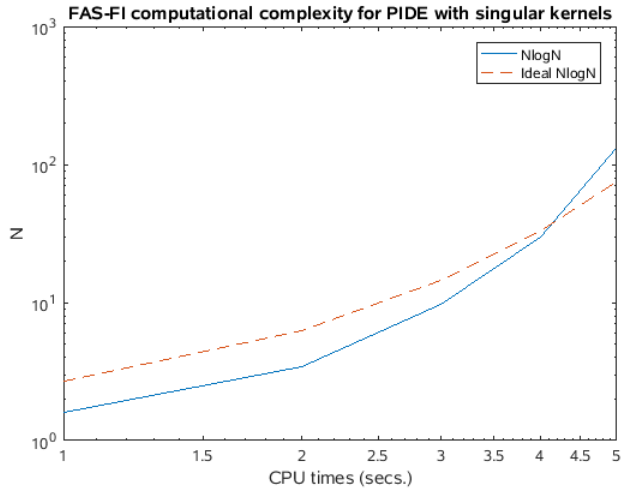


Figure 3.10: Computational complexity of the FAS-FI scheme for the PIDE with a singular kernel.

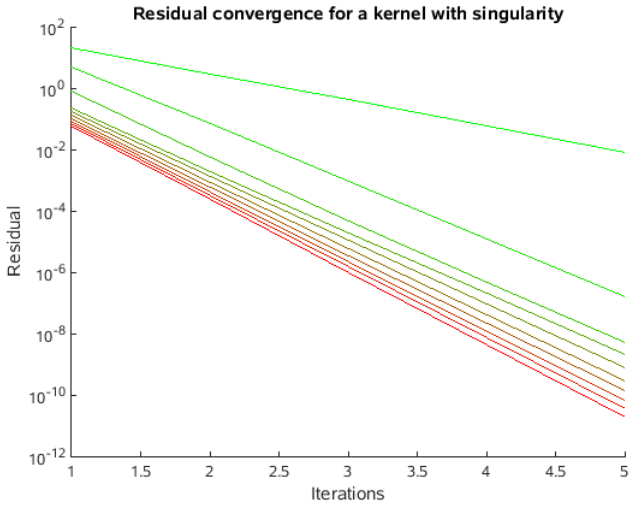


Figure 3.11: Convergence history of the FAS-FI scheme for a PIDE with singular kernel.

3.5 Summary and remarks

In this chapter, a multigrid framework for solving elliptic PIDEs was discussed. The proposed method combines a geometric multigrid scheme for elliptic problems with a multigrid fast integration technique. By means of local Fourier analysis, convergence estimates for the proposed multigrid scheme were obtained. Numerical experiments validated these theoretical estimates. It was shown that optimal computational complexity is obtained when using our multigrid procedure for solving elliptic PIDEs.

4. Hierarchical matrices

In this chapter, we discuss a hierarchical matrix strategy to solve a convection diffusion PIDE problem. We give a detailed description of the method and investigate memory requirements, computational complexity and accuracy of the solutions. We use the hierarchical matrix approach for solving convection-diffusion PIDE problems in three-dimensions and include an application to a time evolution jump diffusion process.

Many different discretization schemes for PIDE problems give rise to algebraic systems with dense coefficient matrices, and dealing with these dense matrices of order N, requires $\mathcal{O}(N^2)$ storage space without compression. Matrix operations involving dense matrices are computationally expensive: for an $N \times N$ dense matrix, computational complexity of matrix-matrix multiplication maybe up to $\mathcal{O}(N^3)$. To overcome this complexity requirement, the panel clustering methods [34, 38] and multipole methods [32] were developed in the 1980s. Extension of the panel clustering method was considered by W. Hackbusch, who then started the development of the hierarchical matrix ($\mathcal{H}$-matrix) method which adopts a hierarchical structure. A hierarchical matrix is a matrix whose block index set has been hierarchically partitioned and whose resulting matrix blocks are given in factored form whenever the rank of such a block matrix is significantly smaller than its size. This strategy uses a data sparse representation to approximate fully populated matrices in certain low rank matrix blocks. This data sparse representaion of $\mathcal{H}$-matrices has a tree structure called the block structure. Hierarchical matrices have enabled more efficient matrix operations obtained from the discretization of elliptic PDEs, integral formulations and now to the solution of classes of PIDEs. In addition to these applications, $\mathcal{H}$- matrices have been used as efficient preconditioners for iterative processes. In [12, 13, 49] $\mathcal{H}$-LU preconditioners are used in the solution of convection diffusion equations while in [5, 6] expositions on the use of $\mathcal{H}$-LU preconditioners are given. In [8, 36], the $\mathcal{H}$-matrix approach is discussed. In $\mathcal{H}$-matrix method, the treatment of fully populated matrices meets the restrictions on storage and arithmetics to nearly optimal complexity $\mathcal{O}\left(N\left(\log_2 N\right)^{\gamma}\right)$ for some small γ. $\mathcal{H}$-matrices are based on the fact that,

typical kernel functions are smooth apart from the diagonal, a notion referred to as asymptotic smoothness [18]. In this chapter, we discuss a strategy for using $\mathcal{H}$-matrices to obtain solutions of PIDEs with optimal computational complexity.

We consider a convection diffusion PIDE problem of the form

$$\begin{aligned} -\varepsilon\Delta y(x) + \nabla\cdot(b\, y(x)) + \mathcal{I}y(x) + \lambda y(x) &= f(x), \quad \text{in} \quad \Omega, \\ y(x) &= 0 \qquad\quad \text{on} \quad \Gamma. \end{aligned} \tag{4.1}$$

This model problem is discretized using the CC scheme outlined in Section 2.2.2 and the analysis of this discretization scheme is revisited to incorporate the hierarchical approximation estimates.

4.1 Construction of Hierarchical matrices

In this section, we discuss the construction of a $\mathcal{H}$-matrix and start by defining the required terminology.

Definition 16. *Let $m, n \in \mathbb{N}$ and $A \in \mathbb{R}^{m\times n}$. If the matrix A is multiplied by any vector $x \in \mathbb{R}^n$, the range of A is obtained i.e. $Im(A) := \{Ax \in \mathbb{R}^m,\ x \in \mathbb{R}^n\}$. Now, the rank of a matrix $A \in \mathbb{R}^{m\times n}$ is the dimension of the image of A i.e.*

$$rank(A) := dim(Im(A)).$$

Corollary 17. *The rank of $A \in \mathbb{R}^{m\times n}$ is equal to the number of non-singular values of A,*

$$rank(A) = |\sigma_i| 0 < \sigma_i \in \Sigma(A)|. \tag{4.2}$$

If A is a square matrix $(m = n)$, then the rank is also equal to the number of non-zero eigenvalues

$$rank(A) = |\lambda_i| 0 \neq \lambda_i \in \Lambda(A)|. \tag{4.3}$$

where λ is the eigen value and Λ is the spectrum(set of all eigenvalues) of A.

We define a set of matrices $A \in \mathbb{R}^{m\times n}$ having q linearly independent rows or columns by $\mathbb{R}_q^{m\times n} := \{A \in \mathbb{R}^{m\times n} : \text{rank}A \leq q\}$.

Definition 18. *A matrix $A \in \mathbb{R}_q^{m\times n}$ is a matrix of low rank if*

$$q(m+n) < m\cdot n.$$

Low-rank matrices will always be represented in outer-product form while entrywise representation will be used for the other matrices.

The closest matrix in $\mathbb{R}_q^{m\times n}$ to a given matrix from $\mathbb{R}^{m\times n}$, $m \geq n$ can be obtained from the singular value decomposition (SVD), $A = U\Sigma V^H$ with $U^H U = I_n = V^H V$ and the diagonal matrix $\Sigma \in \mathbb{R}^{n\times n}$ with the entries $\sigma_1 \geq \ldots \geq \sigma_n \geq 0$.

Theorem 19 (cf. [8]). *Let the SVD $A = U\Sigma V^H$ of $A \in \mathbb{R}^{m\times n}$, $m \geq n$ be given. Then for $q \in \mathbb{N}$ satisfying $q \leq n$ it holds that*

$$\min_{M\in\mathbb{C}_q^{m\times n}} \|A - M\|_F = \|A - A_q\|_F = \|\Sigma - \Sigma_q\|_F,$$

where $A_q := U\Sigma_q V^H \in \mathbb{C}_q^{m\times n}$ and $\Sigma_q := diag\left(\sigma_1, \ldots, \sigma_q, 0, \ldots, 0\right) \in \mathbb{R}^{n\times n}$.

Through inspecting the singular values of A, information on the approximation error, in terms of the Frobenius norm $\|A - A_q\|_F^2 = \sum_{l=q+1}^{n} \sigma_l^2$ can be obtained. This is possible if either the maximum rank q of the approximant is prescribed or the relative accuracy $\epsilon > 0$ is prescribed.

A hierarchical matrix is a matrix whose block index set has been hierarchically partitioned and whose resulting matrix blocks are given in factored form whenever the rank of such a block matrix is significantly smaller than its size. In order to discuss the construction of the $\mathcal{H}$-matrix, we define the following; see, e.g., [8, 13, 36].

Let $I, J \subset \mathbb{N}$ be row and column index sets and let $\mathcal{A}^h = \left(a_{ij}\right)_{i\in I, j\in J} \in \mathbb{R}^{I\times J} \approxeq \mathbb{R}^{\#I,\#J}$, where $\#I$ denotes the cardinality of elements of the set I. A subset $P \subset \mathcal{P}(I \times J)$ of the set of subsets of $I \times J$ is a partition if

$$I \times J = \bigcup_{b\in P} b,$$

and if

$$b_1 \cap b_2 = \emptyset,$$

implying $b_1 = b_2$ for all $b_1, b_2 \in P$ where $b \in P$ are referred to as index blocks or blocks. The hierarchical partitioning is a set of several nested partitions and a $\mathcal{H}$-matrix block partition is a partition consisting of subsets of a hierarchical partitioning.

The distinction of determining whether matrix blocks of an $\mathcal{H}$ matrix are to be represented in factored form or by full matrices is based on the admissibility condition. This admissibility condition determines whether a matrix $t \times s$ can be approximated by a low rank matrix.

Definition 20. *For two sets of indices t and s, let $X_t = \{x_i | i \in t\}$ and $X_s = \{x_j | j \in s\}$. On assumption that the kernel is asymptotically smooth [37], the admissibility condition is given by*

$$\min\left\{\operatorname{diam}\left(X_t\right), \operatorname{diam}\left(X_s\right)\right\} \leq \eta \operatorname{dist}\left(X_t, X_s\right),\ 0 < \eta < 1, \tag{4.4}$$

where

$$\operatorname{diam}(X) = \max_{x_i, x_j \in X} |x_i - x_j|, \quad \operatorname{dist}(x_i, x_j) = \min_{x_i\in X_t, x_j\in X_s} |x_i - x_j|.$$

Hence the hierarchical block partition $b \in P_{I\times J}$ for index sets I and J is admissible or small if the condition is satisfied i.e. the cardinalities $|t|$ and $|s|$ of t and s satisfy $\min\{|t|,|s|\} \leq n_{min}$ with a given minimal dimension $n_{min} \in \mathbb{N}$ otherwise it is inadmissible.

The admissible matrix block $A|_{t\times s}$ can be represented as $A|_{t\times s} = UV^H$ which is referred to as outer-product form, [8]. If $u_i, v_i, i, \ldots, q$ denote columns of u and v respectively, then $A = UV^H$ is equivalent to $A = \sum_{i=1}^{q} u_i v_i^H$. These vectors $u_i, v_i, i = 1, \ldots, q$ require $q(m+n)$ units of storage.

In order to partition a set of matrix indices $I \times J$ hierarchically into sub-blocks, a rule is required to subdivide the index sets I and J. This leads to the so-called cluster trees. With these definitions, we discuss the construction of the cluster tree. Cluster trees are hierarchies of partitions of I and J. For each representation, we assume that the number n of basis functions is a power of 2

$$n = 2^p \quad \text{for} \quad p \in \mathbb{N}.$$

The candidates t, s for the construction of the partition of $I \times J$ are stored in the cluster tree, T_I. The root of the tree T_I is the index set $I_1^{(0)} := \{0, \ldots, n-1\}$.
The two successors of $I_1^{(0)}$ are given by

$$I_1^{(1)} := \left\{0, \ldots, \frac{n}{2} - 1\right\} \text{ and } I_2^{(1)} := \left\{\frac{n}{2}, \ldots, n-1\right\},$$

while the two successors of $I_1^{(1)}$ are given by

$$I_1^{(2)} := \left\{0, \ldots, \frac{n}{4} - 1\right\} \text{ and } I_2^{(2)} := \left\{\frac{n}{4}, \ldots, \frac{n}{2} - 1\right\}$$

and the two successors of $I_2^{(1)}$ are as follows

$$I_3^{(2)} := \left\{\frac{n}{2}, \ldots, 3\frac{n}{4} - 1\right\} \text{ and } I_4^{(2)} := \left\{3\frac{n}{4}, \ldots, n-1\right\}.$$

Each subsequent node t with more than n_{min} indices has exactly two successors, where n_{min} is the minimum block size. Nodes with not more than n_{min} indices are referred to as *leaves*. The parameter n_{min}, controls the depth of the tree. For $n_{min} = 1$ gives the maximum depth. The set of leaves of the cluster tree T_I is denoted by $\mathcal{L}(T_1)$. The number of leaves $\mathcal{L}(T_1)$ is bounded by $|I|/n_{min}$ provided that $|t| \geq n_{min}$, where $|I|$ is the cardinality of I [8]. Ideally a tree T_I is called an index cluster tree if and only if the following properties are satisfied;

1. The root of T_I is the index set I.

2. Each node $s_l \in T_I$ is either a leaf or an internal node with children $S(s_l)$.

3. The children of the same parent are pairwise disjoint i.e.

$$\forall j_1^{l+1}, j_2^{l+1} \in S(i_l) \text{ and} j_1^{l+1} \neq j_2^{l+1} \text{ then } j_1^{l+1} \cap j_2^{l+1} = \emptyset.$$

4. The parent node $S_l = \bigcup_{t^{l+1} \in S(s_l)} t^{l+1}$.

5. $\mathcal{L}(T_I)$ forms a partition of I.

Figure 4.1 illustrates the structure of the cluster tree where $I_1 = \{0, 1, 2, 3\}$.

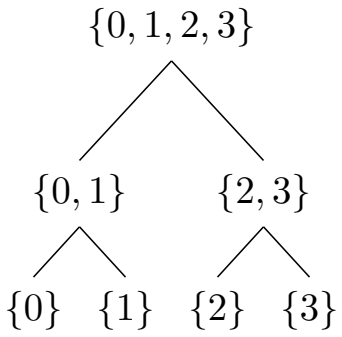


Figure 4.1: Illustration of the Cluster tree, T_I.

A given cluster tree along with the admissibility condition allows canonical construction of a block cluster tree. Let the cluster tree T_I be given, we define the block cluster tree $T_{I\times J}$ by the $root(T) := I \times J$ and each block $t \times s \in T$ has the set of successors

$$S(t \times s) = \begin{cases} \emptyset & \text{if } t \times s \text{ is admissible,} \\ \emptyset & \text{if } \min\{\#t, \#s\} \leq n_{min}, \\ \{t' \times s' | t' \in S(t), s' \in S(s)\} & \text{otherwise} \end{cases}$$

The cluster trees can be used to derive a hierarchy of block partitions of the $I \times J$ corresponding to the matrix, the block cluster tree. The leaves of this tree form a partition of $I \times J$. Let T_I and T_J be cluster trees for the index sets I and J. The finite tree T is a block cluster tree for T_I and T_J if the following conditions hold

1. $root(T) = (root(T_I), root(T_J))$.

2. Each node $b \in T$ has the form $b = (t, s)$ for the clusters $t \in T_I$ and $s \in T_J$.

3. The label of the node $b = (t, s) \in T$ is given by $\hat{b} = \hat{t} \times \hat{s} \subseteq I \times J$.

Remark 21. *1. For practical purposes we use $n_{min} > 1$, because the outer product representation does not pay off for small blocks.*

2. The number of blocks $N_{block}(P) = 3n - 2$, [36].

3. For a cardinality balanced tree T_I as illustrated in Figure 4.2 , the number of nodes for $n_{min} = 1$ is $\#T_I = 2n - 1$, [11]. From [8, Lemma 1.4.1], this estimate shows that the complexity of storing the cluster tree is linear.

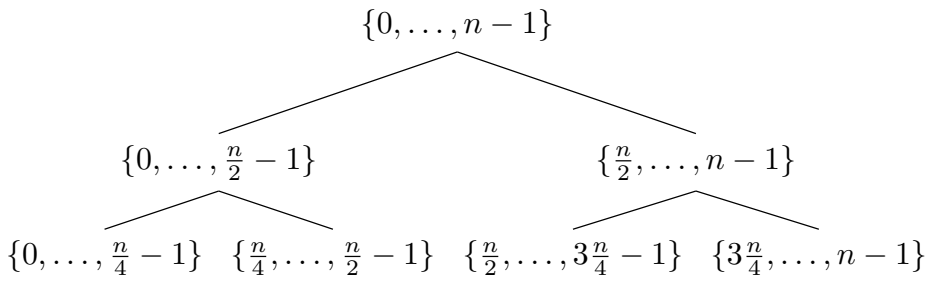


Figure 4.2: A balanced cluster tree.

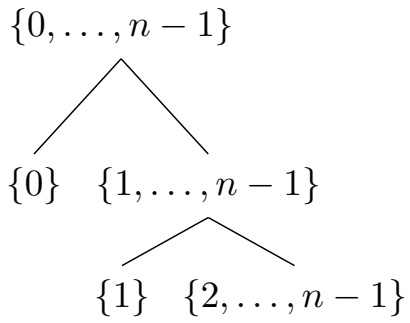


Figure 4.3: An unbalanced cluster tree.

Each block $t \times s$ in the tree $T_{I\times J}$ can be

1. A leaf.
2. Not a leaf , then the block is decomposed into its sons $t' \times s'$ with $t' \in S(t)$ and $s' \in S(s)$.

Figure 4.4 illustrates the structure of a cluster tree where $I_1 \times J_1 = \{0,1,2,3\} \times \{0,1,2,3\}$.

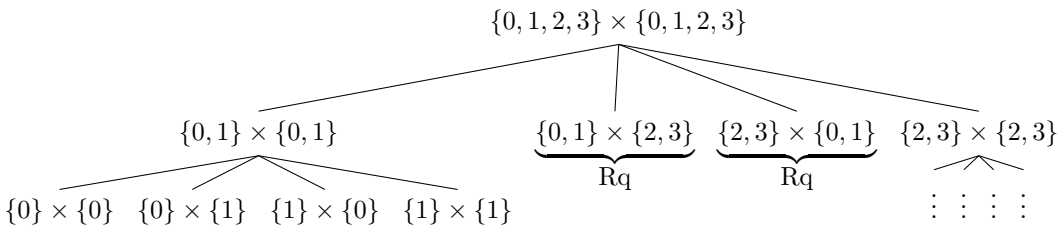


Figure 4.4: Illustration of the block cluster tree,$T_{I\times J}$

In the following Algorithm 6, we summarize the procedure for obtaining a block cluster tree $T_{I\times J}$.

Algorithm 6 Construction of the block cluster tree ,$T_{I\times J}$

1. **if** (t, s) is admissible **then**
$$S(t \times s) := \emptyset$$
2. **else**
$$S(t \times s) := \left\{ t' \times s' |\ t' \in S(t) \text{ and } s' \in S(s) \right\}$$
3. **for** $t' \in S(t)$ and $s' \in S(s)$ **go to** 1
4. **end for**
5. **end if**

Definition 22. *$\mathcal{H}-$matrix. Let $n_{min} \in \mathbb{N}_0$. Let P be a partition of the index set $I \times J$. Let $q : P \to \mathbb{N}_0$ be a mapping that assigns a rank $q(b)$ to each block $b = b_1 \times b_2 \in P$. The set of $\mathcal{H}$-matrices induced by the partition P with a minimum block size n_{min} is defined by*

$$\mathcal{H}(P, q) := \left\{ A \in \mathbb{R}^{I\times J} | \forall t \times s \in P : rank\left(A|_{t\times s}\right) \leq q(t \times s) \text{ or } \min\{\#t, \#s\} \leq n_{min} \right\}. \tag{4.5}$$

The matrix block $A|_{t\times s}$ is admissible if $b \in P$ is admissible.

4.2 $\mathcal{H}$-matrix arithmetics

$\mathcal{H}$-matrices can be stored in a cheap and efficient way with almost linear amount of storage. It is important to know whether the same holds for arithmetic operations involving them. The arithmetic operations of $\mathcal{H}$-matrices involve two strategies, on one hand one can fix the blockwise rank of the intermediate and the final results, while on the other hand one can fix the approximation error. We

review the arithmetic operations primarily used in this work, detailed expositions can be found in [8,36] and references therein.

Consider a $\mathcal{H}$-matrix having the following block structure

$$A = \begin{bmatrix} A_{t_1\times t_1} & A_{t_1\times t_2} \\ A_{t_2\times t_1} & A_{t_2\times t_2} \end{bmatrix}.$$

If $x = \begin{bmatrix} x_{t_1}^T & x_{t_2}^T \end{bmatrix}^T$ is a partition in the same row way, then one can split the $\mathcal{H}$-matrix-vector product into four smaller $\mathcal{H}$-matrix vector products

$$Ax = \begin{bmatrix} A_{t_1\times t_1}x_{t_1} & A_{t_1\times t_2}x_{t_2} \\ A_{t_2\times t_1}x_{t_1} & A_{t_2\times t_2}x_{t_2} \end{bmatrix}.$$

This is repeated down to the leaves through recursion. In the leaves, we have the form $w_t = w_t + A|_{t,s}x_t$ where $w \in \mathbb{R}^n$. If (t,s) is inadmissible then this is a standard matrix vector product. If (t,s) is admissible we can use the low rank factorization $w_t = w_t + UV^T x_t$ to reduce the complexity of the operation. We summarize this procedure in the following Algorithm 7.

Algorithm 7 $\mathcal{H}$-matrix vector multiplication

A matrix vector multiplication where $A \in \mathcal{H}(T_{I\times I})$ and $x \in \mathbb{R}^n$
Output: $Ax = w_t \in \mathbb{R}^n$

1. $w_t = 0$;
2. **for all** $t \times s \in \mathcal{L}(\mathcal{T}_{\mathcal{I}\times\mathcal{I}})$ **do**
   - Compute $z = V^T x_t$; Computational cost is $\mathcal{O}(qn)$
   - Compute $w_t = Uz$; Computational cost is $\mathcal{O}(qm)$
3. $w_t = w_t + UV^T x_t$; Overall computational cost is $\mathcal{O}(q(m+n))$.
4. **end for**

Next, we discuss the addition of two hierarchical matrices. The sum of two hierarchical matrices $A, B \in \mathcal{H}(T_{I\times J}, q)$ is usually in $\mathcal{H}(T_{I\times J}, 2q)$. Since A and B have the same structure, it suffices to add the two submatrices in the leaves. In the inadmissible leaves, we have to add dense matrices as follows

$$\begin{aligned} A + B &= \begin{bmatrix} A_{t_1\times t_1} & A_{t_1\times t_2} \\ A_{t_2\times t_1} & A_{t_2\times t_2} \end{bmatrix} + \begin{bmatrix} B_{t_1\times t_1} & B_{t_1\times t_2} \\ B_{t_2\times t_1} & B_{t_2\times t_2} \end{bmatrix}, \\ &= \begin{bmatrix} A_{t_1\times t_1} + B_{t_1\times t_1} & A_{t_1\times t_2} + B_{t_1\times t_2} \\ A_{t_2\times t_1} + B_{t_2\times t_1} & A_{t_2\times t_2} + B_{t_2\times t_2} \end{bmatrix}. \end{aligned}$$

In the case of approximate addition of two matrices that already contain an error due to discretization, it is desirable to get a result of blockwise rank q as well. Truncation step should be performed after addition at a cost of one addition but a resulting matrix leads to cheaper additions as the block-wise ranks are smaller.

Now, we discuss the inversion of hierarchical matrices. Inversion of $\mathcal{H}$-matrices is a delicate issue, first discussed in [37]. Different approaches have been discussed. In [8], the property that each matrix $A \in \mathcal{H}(T_{I\times J}, q)$ can be subdivided according to its block cluster tree is exploited (for brevity we drop the $t's$).

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}.$$

The exact inverse of A is given by

$$A^{-1} = \begin{bmatrix} A_{11}^{-1} + A_{11}^{-1}A_{12}S^{-1}A_{21}A_{11}^{-1} & -A_{11}^{-1}A_{12}S^{-1} \\ -S^{-1}A_{21}A_{11}^{-1} & S^{-1} \end{bmatrix}, \tag{4.6}$$

where S denotes the Schur's complement $S := A_{22} - A_{21}A_{11}^{-1}A_{12}$ of A_{11} in A. The $\mathcal{H}$-matrix inverse of A is computed by replacing the multiplications and additions in (4.6) by $\mathcal{H}$-matrix versions. The complexity of the computation of the $\mathcal{H}$-inverse is determined by the cost of $\mathcal{H}$-matrix multiplication.

4.3 Adaptive cross approximation

The adaptive cross approximation (ACA), involves finding the low-rank approximant from a few of the original matrix entries of admissible blocks [8, 58]. The whole matrix need not be built beforehand and the respective matrix entries are computed on demand. The rank of the approximation is chosen adaptively. The SVD scheme would find the lowest rank that is required for a given accuracy.

We focus on a single block $A \in \mathbb{R}^{m\times n}$. Essential is to decompose the block A as follows

$$A = U_q V_q^T + R_q,$$

where $U_q \in \mathbb{R}^{m\times q}$,$V_q \in \mathbb{R}^{n\times q}$ and $R_q \in \mathbb{R}^{m\times n}$ is the approximation error. If the norm of R_q is small, then A is approximated by a matrix of rank at most q. Starting from $R_0 := A$, find a non-zero pivot in R_{q-1} say (i_q, j_q) and subtract a scaled outer product of the i_q-th row and j_q-th column. We have

$$R_q := R_{q-1} - \left[(R_{q-1})_{i_q j_q}\right]^{-1} (R_{q-1})_{1:m,j_q} (R_{q-1})_{i_q,1:n},$$

where $\left(R_{q-1}\right)_{i,1:n}$ and $\left(R_{q-1}\right)_{1:m,j}$ are the i-th row and j-th column of R_{q-1} respectively. j_q should be chosen as the maximum element in the modulus of the i_q-th row as follows

$$|\left(R_{q-1}\right)_{i_q j_q}| = \max_{j=1,\ldots,n} |\left(R_{q-1}\right)_{i_q j}|.$$

Since in the q-th step only entries in the j_q-th column and the i_q-th column of R_{q-1} are used to compute R_q, there is no need to build the whole matrix R_{q-1}. The algorithm produces the vectors $u_l \in \mathbb{R}^m$ and $v_l \in \mathbb{R}^n$, $l = 1, \ldots, q$, from which the approximant S_q can be formed as follows

$$S_q = \sum_{l=1}^{q} u_l v_l^T.$$

In Algorithm 8, we summarize the ACA scheme. Notice that Algorithm 8 stops at the qth step after a given relative accuracy $\epsilon > 0$ has been reached.

Algorithm 8 Adaptive cross approximation (ACA)

Input: Individual entries a_{ij} of the matrix $A \in \mathbb{R}^{m\times n}$, having an adaptive rank $q < \min\{m, n\}$, $R_0 := A$ and $q := 0$.
Output: Factors $u = \left(u_1, \ldots, u_q\right) \in \mathbb{R}^{m\times q}$, $v = \left(v_1, \ldots, v_q\right) \in \mathbb{R}^{n\times q}$ that yields $A \approx UV^H$.

1. $q = 1$.
2. **while** stopping criterion is not satisfied **do**.
3. Choose $\left(i_q, j_k\right) = \arg\max_{i,j} |\left(R_{q-1}\right)_{i,j}|$;

$$u_q = R_{q-1} e_{j_q}; \quad v_q = R_{q-1}^H e_{i_q};$$

$$\gamma_q = \left(\left(R_{q-1}\right)_{i_q, j_q}\right)^{-1};$$

$$R_q = R_{q-1} - \gamma_q u_q v_q^H;$$

4. $q = q + 1$.
5. **endwhile**.

A $\mathcal{H}$-matrix approximation of a fully populated matrix $\mathcal{A}^h \in \mathbb{R}^{I\times J}$ for finite index sets I and J is done first by obtaining block partitions of the matrix index set $I \times J$ and replacing each block $b = t \times s \subset I \times J$ of this partitioning with a matrix of low-rank $q(b)$. If this rank $q(b)$ is small compared to the number of indices in t and s then storage requirements are very lower than an approximated full matrix.

From (4.1), the differential part on discretization gives a sparse matrix while

the integral term yields a fully populated matrix. We consider seperate approaches to the construction of $\mathcal{H}$-matrices of the two parts, $\mathcal{C}^{\mathcal{H}}$ and $\mathcal{I}^{\mathcal{H}}$. We convert the sparse matrix to compressed row storage (CRS) format (see the Appendix A.3) and convert to $\mathcal{H}$-matrix afterwards. The $\mathcal{H}$-matrix of the integral operator is implemented directly, where the block entries are computed once the admissibility condition has been satisfied. Thereafter we add the two $\mathcal{H}$-matrices where addition of the $\mathcal{H}$-matrix from the sparse matrix only affects the diagonal entries of the $\mathcal{H}$-matrix from the fully populated matrices. With the coefficient matrix from (2.20) stored in the $\mathcal{H}$-matrix format, we discuss the strategy to decompose and obtain the solution of the system of linear equations (2.20).

The gain in storage and computational cost for $\mathcal{H}$-matrices arithmetic operations, when the matrix $\mathcal{A}^h$ is approximated by the $\mathcal{H}$-matrix $\mathcal{A}^{\mathcal{H}} = \mathcal{C}^{\mathcal{H}} + \mathcal{I}^{\mathcal{H}}$, originate from the low-rank representations of the admissible blocks. These operations can be performed with almost linear complexity i.e. $\mathcal{O}(qN \log N)$.

Further, storage gains can be achieved through coarsening techniques. We discuss the agglomeration of low-rank blocks. This is a procedure of unifying neighbouring blocks to a single one with the aim of saving memory, for details see [8]. The coarsening technique that can be applied to the whole $\mathcal{H}$-matrix or a sub-matrix as shown in the Figure 4.5.

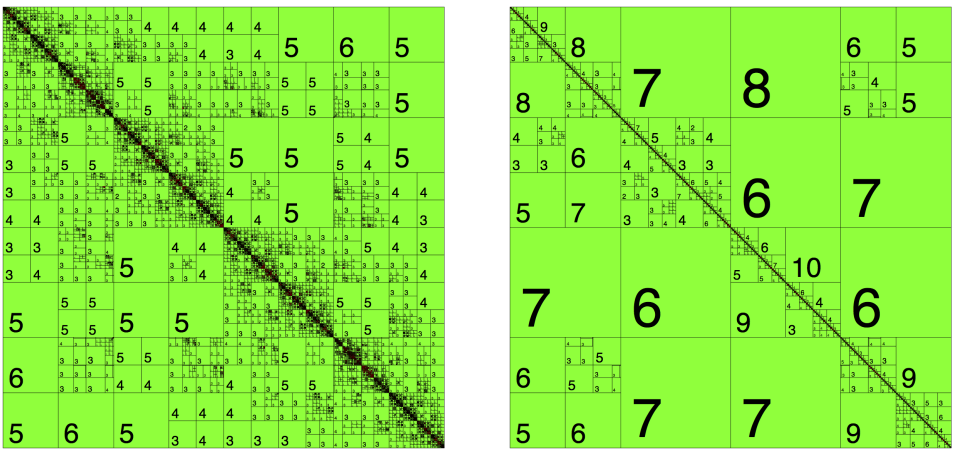


Figure 4.5: $\mathcal{A}^{\mathcal{H}}$ before and after agglomeration for the meshsize 128×128 with their rank distribution.

The integral term in equation (2.3) is the bottleneck of the efficient solution of the convection diffusion PIDE. In this regard, to address the error due to the $\mathcal{H}$-matrix approximation $\mathcal{I}^{\mathcal{H}}$, we first discuss the efficient $\mathcal{H}$-matrix approximation of the kernel that leads to an efficient approximation of the whole system. The

approximation error of the kernel $k - \tilde{k}$ affects the size of the matrix error $A_{ts} - \tilde{A}_{ts}$. We focus on the admissible block b assuming the block $b = t \times s$ satisfies the admissibility condition.

Theorem 23. *Let $b = t \times s$, $t \subset I$ and $s \subset J$, assuming that*

$$|k(x_1, x_2) - \tilde{k}(x_1, x_2)| \leq \epsilon'', \text{ for all } x_1 \in X_t, x_2 \in X_s,$$

then

$$|a_{ij} - \tilde{a}_{ij}| \leq \epsilon'', \ i \in t, j \in s,$$

and

$$\|A_b - \tilde{A}_b\|_F \leq \sqrt{|t||s|}\epsilon''.$$

Proof. For entrywise estimates for the kernel of the integral

$$|a_{ij} - \tilde{a}_{ij}| \leq \epsilon''.$$

Considering the Frobenius norm, we have

$$\begin{aligned} \|A_b - \tilde{A}_b\|_F^2 &= \sum_{i \in t} \sum_{j \in s} |k - \tilde{k}|^2 \\ &= \sum_{i \in t} \sum_{j \in s} |a_{ij} - \tilde{a}_{ij}|^2 \leq |t||s|\epsilon''^2 \\ \|A_b - \tilde{A}_b\|_F &= \sqrt{|t||s|}\epsilon''. \end{aligned}$$

□

Notice that we construct $\mathcal{H}$-matrix approximations of $\mathcal{C}^h$ and $\mathcal{I}^h$, separately. Thus we obtain $\mathcal{C}^{\mathcal{H}}$ and $\mathcal{I}^{\mathcal{H}}$, and $\mathcal{A}^{\mathcal{H}} = \mathcal{C}^{\mathcal{H}} + \mathcal{I}^{\mathcal{H}}$. From the Theorem 23, we have that there exist positive constants c' and c'' such that $\|\mathcal{C}^h - \mathcal{C}^{\mathcal{H}}\|_F \leq c'\,\epsilon'$ and $\|\mathcal{I}^h - \mathcal{I}^{\mathcal{H}}\|_F \leq c''\,\epsilon''$. Therefore we have $\|\mathcal{A}^h - \mathcal{A}^{\mathcal{H}}\|_F \leq \epsilon$ where $\epsilon = c'\,\epsilon' + c''\,\epsilon''$. Next, we discuss the $\mathcal{H}$-matrix solution error estimates.

4.4 $\mathcal{H}$-matrix approximation analysis

In this section, we discuss the $\mathcal{H}$-matrix approximation property where we revisit solution error estimates from Section 2.2.2. We anticipate some results that we use to estimate the solution error resulting from the $\mathcal{H}$-matrix approximation.

In our approach, the CCT matrix $\mathcal{A}^h$ is being approximated by the $\mathcal{H}$-matrix $\mathcal{A}^{\mathcal{H}}$ and, as stated in Theorem 23, for a given accuracy ϵ it is possible to construct $\mathcal{A}^{\mathcal{H}}$ such that the following holds

$$\|\mathcal{A}^h - \mathcal{A}^{\mathcal{H}}\|_F \leq \epsilon,$$

where $\|\cdot\|_F$ denotes the Frobenius matrix.

Now, let y_h be the solution to the CCT scheme $\mathcal{A}^h y_h = f_h$ and denote with $y_h^{\mathcal{H}}$ the solution to $\mathcal{A}^{\mathcal{H}} y_h^{\mathcal{H}} = f_h$, and given that the approximation $\mathcal{A}^{\mathcal{H}} = \mathcal{A}^h + \mathcal{E}$, $\|\mathcal{E}\|_F = \epsilon$ and hence $\left(\mathcal{A}^h + \mathcal{E}\right) y_h^{\mathcal{H}} = f_h$. Further $\mathcal{A}^h \left(y_h - y_h^{\mathcal{H}}\right) = \mathcal{E} y_h^{\mathcal{H}}$ and hence utilising the stability of $\mathcal{A}^h$ there exists a constant $\beta > 0$ such that the following holds [55]

$$\|y_h - y_h^{\mathcal{H}}\|_h \leq \beta\, \epsilon, \tag{4.7}$$

where β depends on the data.

Further notice that the problem $\mathcal{A}^{\mathcal{H}} y_h^{\mathcal{H}} = f_h$ will be solved by an iterative scheme that stops whenever a tolerance on the norm of the residual, $r_h = f_h - \mathcal{A}^{\mathcal{H}} \tilde{y}_h^{\mathcal{H}}$, is satisfied, i.e. $\|r_h\| < \text{tol}$. This means that the iterative procedure provides the solution $\tilde{y}_h^{\mathcal{H}}$ such that $\mathcal{A}^{\mathcal{H}} \tilde{y}_h^{\mathcal{H}} = f_h - r_h$. Thus, we also have $\mathcal{A}^{\mathcal{H}} \left(y_h^{\mathcal{H}} - \tilde{y}_h^{\mathcal{H}}\right) = r_h$. Since there is no guarantee on the stability of $\mathcal{A}^{\mathcal{H}}$, we use $\mathcal{A}^{\mathcal{H}} = \mathcal{A}^h + \mathcal{E}$ and obtain the following estimate

$$\left(\mathcal{A}^h + \mathcal{E}\right) \left(y_h^{\mathcal{H}} - \tilde{y}_h^{\mathcal{H}}\right) = r_h,$$

and from the Perturbation Lemma [55, Chapter 2], we have the following

$$\begin{aligned} \|y_h^{\mathcal{H}} - \tilde{y}_h^{\mathcal{H}}\| &\leq \| \left(\mathcal{A}^h + \mathcal{E}\right)^{-1} \| \, \|r_h\|, \\ &\leq \frac{\| \left(\mathcal{A}^h\right)^{-1} \|}{1 - \| \left(\mathcal{A}^h\right)^{-1} \| \|\mathcal{E}\|} \, \text{tol}, \\ &\leq \beta' \, \text{tol}, \end{aligned} \tag{4.8}$$

where $\beta' = \frac{\|\left(\mathcal{A}^h\right)^{-1}\|}{1 - \|\left(\mathcal{A}^h\right)^{-1}\| \|\mathcal{E}\|}$.

Notice that, using the generalised minimum residue (GMRes) method [48,57], with a precondition of accuracy δ, we have $\|r_h\|_h \leq c_G\, \delta^\ell \, \|f_h\|_h$, where $c_G > 0$ is independent of h, and ℓ is the index of the ℓ-th GMRes iterate; see, e.g., [8, Chapter 2].

Now, we can collect our results and prove the following theorem.

Theorem 24. *Let $y \in C^4([a,b])$ be the solution to the PIDE problem (2.3) with $k \in C^2([a,b]) \times C^2([a,b])$ and $\lambda \geq \rho/2$, and $\tilde{y}_h^{\mathcal{H}}$ be the approximate solution to the corresponding $\mathcal{H}$-matrix problem $\mathcal{A}^{\mathcal{H}} y_h^{\mathcal{H}} = f_h$ obtained after $\ell \geq 1$ iterations of the preconditioned GMRes scheme, then the resulting solution satisfies the following estimate*

$$\|y - \tilde{y}_h^{\mathcal{H}}\|_h \leq c_1\, h^2 + c_2\, \epsilon + c_3 \left(\delta^\ell \beta'\right), \tag{4.9}$$

for positive constants c_1, c_2, c_3 depending on the data and independent of h.

Proof. Let y_h be the solution to $\mathcal{A}^h y_h = f_h$ and $y_h^{\mathcal{H}}$ be the solution to $\mathcal{A}^{\mathcal{H}} y_h^{\mathcal{H}} = f_h$. Consider the following estimate

$$\begin{aligned}\|y - \tilde{y}_h^{\mathcal{H}}\|_h &= \|y - y_h + y_h - \tilde{y}_h^{\mathcal{H}} + y_h^{\mathcal{H}} - y_h^{\mathcal{H}}\|_h \\ &\leq \|y - y_h\|_h + \|y_h - y_h^{\mathcal{H}}\|_h + \|y_h^{\mathcal{H}} - \tilde{y}_h^{\mathcal{H}}\|_h.\end{aligned}$$

For the first term, we have the estimate of Theorem 14. The second term is estimated by (4.7). The third term is estimated by (4.8) and the convergence property of the preconditioned GMRes algorithm. Thus the theorem is proved. □

4.5 $\mathcal{H}$-LU factorization

In this section, we consider the matrix $\mathcal{A}^{\mathcal{H}}$ to construct a suitable LU factorization. Even though the computational effort of hierachical matrix inversion is relatively high, it can be done in almost linear complexity provide that the blockwise rank behaves well. Hierarchical LU decomposition [6, 7, 9] offer a more significantly efficient alternative. An approximate hierarchical LU (HLU) decomposition is defined as follows

$$\mathcal{A}^{\mathcal{H}} \approx L_{\mathcal{H}} U_{\mathcal{H}},$$

where the lower and upper triangular matrices $L_{\mathcal{H}}$ and $U_{\mathcal{H}}$, respectively, are stored in $\mathcal{H}$-matrix format. See Figure 4.6. The storage of $L_{\mathcal{H}}$ and $U_{\mathcal{H}}$ together is the same as the storage of the matrix $\mathcal{A}^{\mathcal{H}}$. The HLU decomposition provides a more significant strategy to reduce the computational effort than computing the $\mathcal{H}$-matrix inverse. The hierarchical block structure of the block $A|_{t\times t} \in T_I \backslash \mathcal{L}(T_I)$ is exploited.

That is

$$A_{t\times t} = \begin{bmatrix} A_{t_1\times t_1} & A_{t_1\times t_2} \\ A_{t_2\times t_1} & A_{t_2\times t_2} \end{bmatrix} = \begin{bmatrix} L_{t_1\times t_1} & \\ L_{t_2\times t_1} & L_{t_2\times t_2} \end{bmatrix} \begin{bmatrix} U_{t_1\times t_1} & U_{t_1\times t_2} \\ & U_{t_2\times t_2} \end{bmatrix}, \tag{4.10}$$

where $t_1, t_2 \in T_I$ are the sons of t in T_I. This LU decomposition on the block $A_{t\times t}$ leads to the following problems on the sons of $t \times t$:

(a) Evaluate $L_{t_1\times t_1}$ and $U_{t_1\times t_1}$ from $L_{t_1\times t_1} U_{t_1\times t_1} = A_{t_1\times t_1}$;

(b) Evaluate $U_{t_1\times t_1}$ from $L_{t_1\times t_1} U_{t_1\times t_2} = A_{t_1\times t_2}$;

(c) Evaluate $L_{t_2\times t_1}$ from $L_{t_2\times t_1} U_{t_1\times t_1} = A_{t_2\times t_1}$;

(d) Evaluate $L_{t_2\times t_2}$ and $U_{t_2\times t_2}$ from $L_{t_2\times t_2} U_{t_2\times t_2} = A_{t_2\times t_2} - L_{t_2\times t_1} U_{t_1\times t_2}$.

Two scenarios are likely to occur, if a block $t \times t \in \mathcal{L}(T_{I\times I})$ is a leaf, the usual pivoted LU decomposition is used, otherwise for (a) and (d), two LU decompositions of half the size are evaluated. To solve (b), a recursive block forward substitution is used, see [8]. Similarly, (c) can be solved by recursive block forward substitution. An approximate LU decomposition of $A \in \mathcal{H}(T_{I\times I}, q)$ can be obtained to a prescribed order of accuracy if the blockwise rank can be guaranteed to be logarithmically bounded. We summarize the HLU factorization in the following Algorithm 9.

Algorithm 9 $\mathcal{H}$-LU Factorization

1. **If** $A_{t_1\times t_1}$ or $A_{t_2\times t_2}$ is a full matrix leaf, use full matrix LU factorization;
2. **Otherwise** apply recursively HLU factorization to $A_{t_1\times t_1}$ to obtain $L_{t_1\times t_1}$ and $U_{t_1\times t_1}$;
3. Solve $A_{t_1\times t_2} = L_{t_1\times t_1} U_{t_1\times t_2}$ with $L_{t_1\times t_1}$ to obtain $U_{t_1\times t_2}$;
4. Solve $A_{t_2\times t_1} = L_{t_2\times t_1} U_{t_1\times t_1}$ with $U_{t_1\times t_1}$ to obtain $L_{t_2\times t_1}$;
5. Apply HLU factorization recursively to $L_{t_2\times t_2} U_{t_2\times t_2} = A_{t_2\times t_2} - L_{t_2\times t_1} U_{t_1\times t_2}$ to get $L_{t_2\times t_2}$ and $U_{t_2\times t_2}$.

From [8, Th. 4.3.3], we have that there exist $L_{\mathcal{H}}$ and $U_{\mathcal{H}}$ such that

$$\|\mathcal{A}^{\mathcal{H}} - L_{\mathcal{H}} U_{\mathcal{H}}\|_2 \leq \mathrm{cond}(\mathcal{A}^{\mathcal{H}})\,\delta,$$

where $\|\cdot\|_2$ denotes the matrix spectral norm.
Therefore we can use the HLU decomposition to construct the preconditioner

$$C_{pre} = (U_{\mathcal{H}})^{-1} (L_{\mathcal{H}})^{-1}.$$

Since we consider a convection-diffusion PIDE, the aim of preconditioning is to obtain a spectrum of $C_{pre}\mathcal{A}^{\mathcal{H}}$ which is clustered away form the origin. From the estimate above, we have

$$\|I - C_{pre}\,\mathcal{A}^{\mathcal{H}}\|_2 \leq \tilde{c}\ \mathrm{cond}(\mathcal{A}^{\mathcal{H}})\,\delta.$$

Notice that using C_{pre} to precondition the GMRes scheme to solve $\mathcal{A}^{\mathcal{H}} y = f_h$, for convergence, the numerical range

$$\mathcal{F}\left(C_{pre}\mathcal{A}^{\mathcal{H}}\right) : \{x^T C_{pre}\mathcal{A}^{\mathcal{H}} x \colon x \in \mathbb{C}^n, \|x\|_2 = 1\}$$

of $C_{pre}\mathcal{A}^{\mathcal{H}}$ is important, see [31]. For the ℓ-th iterate, provided that $\mathcal{F}\left(C_{pre}\mathcal{A}^{\mathcal{H}}\right) \subset \mathcal{B}_\delta(1)$ where $\mathcal{B}_\delta(1)$ is a disc centered 1 with radius δ, then we have the following

estimate

$$|x^T C_{pre} \mathcal{A}^{\mathcal{H}} x - 1| = |x^T \left(C_{pre} \mathcal{A}^{\mathcal{H}} - I \right) x_1| \leq \|I - C_{pre}\, \mathcal{A}^{\mathcal{H}}\|_2 \leq \tilde{c}\, \mathrm{cond}(\mathcal{A}^{\mathcal{H}})\, \delta.$$

Hence, we have

$$\|f_h - \mathcal{A}^{\mathcal{H}} y^{(\ell)}\|_h \leq c_G\, \delta^{\ell}\, \|f_h\|_h,$$

where ℓ denotes the ℓ-th GMRes iterate and $y^{(\ell)}$ denotes the corresponding solution; see, e.g., [8, Chapter 2].

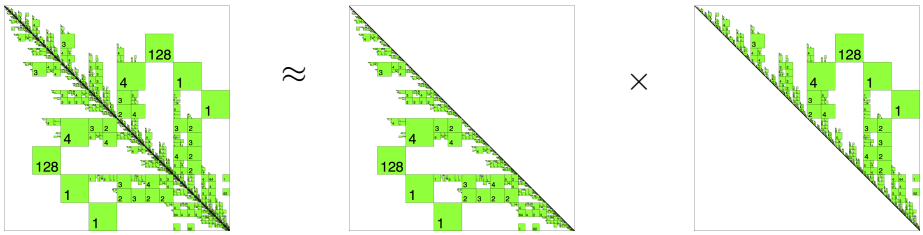


Figure 4.6: Illustration of HLU decomposition of $\mathcal{C}^{\mathcal{H}}$ for the mesh size 257×257.

4.6 Numerical experiments

In this section, we present results of numerical experiments to investigate the computational performance of our $\mathcal{H}$-matrix solution procedure for our convection diffusion PIDE problem (2.3) defined on a unit square domain $\Omega = (0,1) \times (0,1)$. We consider a drift term given by $b(x,z) = (x_1 + x_2, z_1 + z_2)$. We choose $\varepsilon = 2$ and $\lambda = 1$, and the kernel of the integral term is given by

$$k(x,z) = \exp\left(- \left((x_1 - z_1)^2 + (x_2 - z_2)^2 \right) \right),$$

where $x = (x_1, x_2)$ and $z = (z_1, z_2)$.

All numerical tests discussed in this section are performed on an Intel core i5 2.4GHz processor with 4GB core memory. For the $\mathcal{H}$-matrices calculation, we use the C++ library 'Another software library on Hierarchical matrices for Elliptic Differential equations' (AHMED) with no parallelization.

In the first set of experiments, we validate the performance of the $\mathcal{H}$-matrix approach in terms of the memory data usage, the compression rate, the memory increment and the results from the agglomeration of the matrices of the approximation of the kernel of the integral term. In this calculation, we choose a prescribed accuracy for the $\mathcal{H}$-matrix approximation given by $\epsilon = 10^{-4}$, the minimum block size $n_{min} = 15$ and η_{int} is the cluster parameter for the approximation of the kernel and the integration procedure, while η_{pre} is cluster parameter

for the preconditioning procedure. They are chosen differently in a bid to balance between computational requirements and the accuracy of the solutions. For the HLU preconditioning, we set the accuracy $\delta = 0.1$. The compression rate is computed as follows

$$Compression\ rate\,(\%) = \frac{Memory\ consumption}{Original\ memory} \times 100,$$

where the memory consumption corresponds to the storage of the $\mathcal{H}$ matrix, while the original memory refers to the storage requirement of the CCT matrix of coefficients. The same rate can be defined for the single components of the PIDE operator.

In the Tables 4.1 and 4.2, we report results of the computational effort for the $\mathcal{H}$-matrix approximation of the Fredholm integral operator for different η_{int}. We see that the $\mathcal{H}$-matrix approach has an almost linear storage requirement and optimal computational complexity as depicted in column five of the tables. In the Tables 4.3 and 4.4, similar results are presented for the case where the agglomeration procedure is used for different values of η_{int}. For the agglomeration procedure, an almost linear storage requirement and optimal computational complexity is realized. This can be seen in the fifth columns of the tables.

Table 4.1: Computational requirements for the kernel approximation with $\eta_{int} = 0.8$.

$N \times N$	Memory(MB)	Time(sec)	Compression rate (%)	Increment ratio of Memory	Leaves	$\left\|\mathcal{I}^{\mathcal{H}}\right\|_F$
33×33	3.00	0.03	85.11	-	526	0.7121
65×65	12.63	0.15	21.01	4.21	3790	0.7380
129×129	58.22	0.66	5.87	4.61	20767	0.7510
257×257	262.11	3.05	1.63	4.50	93532	0.7564
513×513	1150.06	14.48	0.44	4.39	403474	0.7607
1025×1025	4899.92	71.19	0.12	4.26	1701871	0.7612

Table 4.2: Computational requirements for the kernel approximation with $\eta_{int} = 3.0$.

$N \times N$	Memory(MB)	Time(sec)	Compression rate (%)	Increment ratio of Memory	Leaves	$\|\mathcal{I}^{\mathcal{H}}\|_F$
33×33	0.67	0.01	18.86	-	130	0.7004
65×65	2.91	0.05	4.85	4.34	610	0.7381
129×129	12.62	0.17	1.27	4.34	2719	0.7364
257×257	53.73	0.74	0.33	4.26	11512	0.7408
513×513	232.79	3.55	0.09	4.33	46186	0.7608
1025×1025	973.04	15.00	0.02	4.18	189331	0.7610

Table 4.3: Computational performance of the agglomeration procedure with $\eta_{int} = 0.8$.

$N \times N$	Memory(MB)	Time(sec)	Compression rate (%)	Increment ratio of Memory
33×33	1.06	0.19	30.19	-
65×65	4.76	0.79	7.92	4.49
129×129	19.84	1.48	1.99	4.17
257×257	94.18	6.03	0.58	4.75
513×513	378.77	34.18	0.15	4.02
1025×1025	1498.23	361.09	0.04	3.96

Table 4.4: Computational performance of the agglomeration procedure with $\eta_{int} = 3.0$.

$N \times N$	Memory(MB)	Time(sec)	Compression rate (%)	Increment ratio of Memory
33×33	0.63	0.01	17.92	-
65×65	2.79	0.03	4.64	4.43
129×129	12.07	0.11	1.22	4.33
257×257	50.25	0.52	0.31	4.16
513×513	215.55	2.38	0.08	4.29
1025×1025	894.66	11.61	0.02	4.15

Next, we apply the $\mathcal{H}$-matrices framework to implement a HLU preconditioned GMRes [13, 48] and perform iterations until the given residual tolerance is reached. Our GMRes algorithm, (see the Appendix A.2), uses the Arnoldi method (Gram-Schmidt process of orthonomalization) to construct the orthonormal basis of the Krylov space; see [4, 57] for all details. As an initial guess for the numerical solution of our PIDE problem, we choose the zero function. With this setting, we solve our PIDE problem assuming an exact solution given by

$y(x_1, x_2) = \sin(2\pi x_1)\sin(2\pi x_2)$ with which we determine the source term f by substitution in the continuous model.

In Table 4.5, we report on the L_2-norm of the solution error for different sizes of our problem. We see that a second-order of accuracy of the numerical solution is reached. This result suggests that the leading error in the estimate (4.9) of Theorem 24 is related to the CCT discretization.

Table 4.5: L_2-norm of solution errors.

$N \times N$	$\|y - \tilde{y}_h^{\mathcal{H}}\|_{L_h^2}$
33×33	1.041e-2
65×65	2.283e-3
129×129	5.634e-4
257×257	1.267e-4
513×513	3.055e-5
1025×1025	7.248e-6

For the experiments of Table 4.5, the GMRes algorithm has a required tolerance of $tol = 10^{-10}$. For this experiment, we report in Tables 4.6 and 4.7 the corresponding memory requirements and the computational time for preconditioning for different values of η_{pre}. Notice that the number of iterations (#It) and the increment ratio of the memory of the preconditioned GMRes algorithm for solving our PIDE problem demonstrate optimal complexity of our solution procedure. Notice that a balance between the parameters $\eta_{int}, \eta_{pre}, \epsilon$ and δ leads to improved computational complexity.

Table 4.6: Computational effort for computing the $\mathcal{H}$-matrix solution to the CCT PIDE problem with $\eta_{int} = 0.8, \epsilon = 1e{-}4$ and $\eta_{pre} = 0.7$.

$N \times N$	Precond: Memory(MB)	Precond: Time(sec)	Increment ratio of Memory	#It	GMRes: time(sec)
33×33	0.95	0.02	-	4	0.002
65×65	4.85	0.22	5.11	4	0.01
129×129	25.29	1.53	5.21	4	0.07
257×257	135.91	9.36	5.37	5	0.40
513×513	735.96	65.51	5.41	7	2.70

Table 4.7: Computational effort for computing the $\mathcal{H}$-matrix solution to the CCT PIDE problem with $\eta_{int} = 0.8$, $\epsilon = 1e{-}4$ and $\eta_{pre} = 0.3$.

$N \times N$	Precond: Memory(MB)	Precond: Time(sec)	Increment ratio of Memory	#It	GMRes: time(sec)
33×33	0.95	0.02	-	4	0.002
65×65	5.41	0.36	5.69	4	0.03
129×129	29.71	3.35	5.49	5	0.13
257×257	161.51	23.55	5.43	7	0.86
513×513	807.51	140.91	4.99	9	6.19

We conclude our series of experiments on two-dimensional PIDE problems showing a comparison of computational performance of our $\mathcal{H}$-matrices method with our multigrid approach [28, 29]. For this purpose, we solve our convection-diffusion PIDE problem (4.1) with the setting given above and the function f corresponding to the chosen exact solution $y(x,z) = \sin(2\pi x)\sin(2\pi z)$. Thus in Table 4.8, we present results of the accuracy of the numerical solution obtained with the $\mathcal{H}$-matrices and multigrid procedures corresponding to different mesh sizes.

Notice that this comparison should take into account the different implementation 'languages' for the two methods: C++ for the $\mathcal{H}$-matrices scheme; MATLAB for the multigrid scheme.

Table 4.8: Computational time (secs) and L_2-norm of solution errors of a $\mathcal{H}$-matrix method and Multigrid method.

$N \times N$	$\mathcal{H}$-matrices: (C++) Time(sec)	$\|y - \tilde{y}_h^{\mathcal{H}}\|_{L_h^2}$	MG V-cycle: (MATLAB) Time(sec)	$\|y - y_h\|_{L_h^2}$
17×17	0.03	6.92e-2	1.35	6.62e-3
33×33	0.27	1.04e-2	6.95	1.64e-3
65×65	1.11	2.28e-3	40.44	4.10e-4
129×129	4.28	5.63e-4	260.71	1.03e-4

Next, we extend the $\mathcal{H}$-matrix solution to a three-dimensions (3D) case of the convection-diffusion PIDE problem (4.1). The discretization is as aforementioned, where we use the CC scheme and the trapezoidal rule. We adopt a similar procedure to the $\mathcal{H}$-matrix of the two-dimension case of (4.1). We consider a unit cuboid domain and choose the drift term to be $b(x,z,r) = (x_1 + x_2 + x_3, z_1 + z_2 + z_3, r_1 + r_2 + r_3)$, $\varepsilon = 2$, and $\lambda = 1$ and the integral kernel is given by

$$k(x,z) = \exp\left(-\left((x_1 - z_1)^2 + (x_2 - z_2)^2 + (x_3 - z_3)^2\right)\right),$$

where $x = (x_1, x_2, x_3)$ and $z = (z_1, z_2, z_3)$.

In Tables 4.9 and 4.10, we report results of the computational effort for the $\mathcal{H}$-matrix approximation of the Fredholm integral operator for different values of η_{int}. We see that the $\mathcal{H}$-matrices approach has an almost linear storage requirement and optimal computational complexity as depicted by the increment ratio of the memory . In Tables 4.11 and 4.12, similar results are presented for the case where the agglomeration procedure is used with different values of η_{int} .

Table 4.9: Computational requirements for the 3D kernel approximation with $\eta_{int} = 2.0$ and $\epsilon = 1e-2$.

$N \times N \times N$	Memory (MB)	Time (sec)	Compression rate (%)	Increment ratio: Memory	Leaves	$\|\mathcal{I}^{\mathcal{H}}\|_F$
$17 \times 17 \times 17$	7.99	0.13	18.38	-	3586	0.5368
$33 \times 33 \times 33$	73.80	1.11	2.18	9.24	20983	0.6002
$65 \times 65 \times 65$	643.16	11.31	0.27	8.71	106864	0.6339
$129 \times 129 \times 129$	5409.16	113.16	0.03	8.41	660547	0.6509

Table 4.10: Computational requirements for the 3D kernel approximation with $\eta_{int} = 3.0$ and $\epsilon = 1e-2$.

$N \times N \times N$	Memory (MB)	Time (sec)	Compression rate (%)	Increment ratio: Memory	Leaves	$\|\mathcal{I}^{\mathcal{H}}\|_F$
$17 \times 17 \times 17$	3.43	0.07	7.90	-	1348	0.5362
$33 \times 33 \times 33$	42.78	0.66	1.26	12.47	10039	0.6020
$65 \times 65 \times 65$	371.20	6.46	0.16	8.68	52234	0.6364
$129 \times 129 \times 129$	3016.89	58.95	0.02	8.13	327757	0.6505

Table 4.11: Computational performance for the agglomeration procedure with $\eta_{int} = 2.0$ and $\epsilon = 1e-2$.

$N \times N \times N$	Memory (MB)	Time (sec)	Compression rate (%)	Increment ratio: Memory
$17 \times 17 \times 17$	2.77	0.23	6.37	-
$33 \times 33 \times 33$	32.44	0.98	0.96	8.60
$65 \times 65 \times 65$	274.21	7.19	0.11	8.45
$129 \times 129 \times 129$	2167.63	111.67	0.01	7.91

Table 4.12: Computational performance for the agglomeration procedure with $\eta_{int} = 3.0$ and $\epsilon = 1e-2$.

$N \times N \times N$	Memory (MB)	Time (sec)	Compression rate (%)	Increment ratio: Memory
$17 \times 17 \times 17$	2.14	0.06	4.93	-
$33 \times 33 \times 33$	26.77	0.37	0.79	12.49
$65 \times 65 \times 65$	245.02	2.29	0.10	9.15
$129 \times 129 \times 129$	1774.57	44.78	0.01	7.24

In Table 4.13, we report results on the computational performance of the $\mathcal{H}$ matrices scheme. It is noticeable that the computational complexity improves on refinement of the grid.

Table 4.13: Computational effort for computing the $\mathcal{H}$-matrix solution to the three-dimensional CCT PIDE problem with $\eta_{int} = 2.0$, $\epsilon = 10^{-2}$, $\delta = 0.1$ and $\eta_{pre} = 0.7$.

$N \times N \times N$	Precond: Memory(MB)	Precond: Time(sec)	#It	GMRes: time(sec)
$9 \times 9 \times 9$	0.31	0.01	6	0.002
$17 \times 17 \times 17$	6.17	0.59	9	0.04
$33 \times 33 \times 33$	88.84	16.18	15	0.76
$65 \times 65 \times 65$	1128.83	318.61	26	16.10

Next, we discuss results of numerical experiments where we use our CCT $\mathcal{H}$-matrix scheme to solve the time evolution of the probability density function (PDF) of a stochastic jump-diffusion process. We refer to [3, 30] for all details concerning these processes and the formulation of the following Fokker-Planck-Kolmogorov problem

$$\begin{aligned} \partial_t y &= \varepsilon \Delta y - \nabla \cdot (b\, y) + \lambda\, \mathcal{I} y - \lambda\, y + f && \text{in } \mathcal{Q} = \Omega \times (0, T), \\ y(x, 0) &= y_0(x) && \text{in } \Omega \times \{t = 0\}, \\ y(x, t) &= 0 && \text{on } \Gamma \times (0, T]. \end{aligned} \tag{4.11}$$

Notice that zero Dirichlet boundary conditions correspond to absorbing barriers for the stochastic process. In the model (4.11) with $f = 0$, the function y represents the PDF of a jump-diffusion process with drift b and dispersion $\sqrt{2\varepsilon}$. In this model, λ denotes the rate of the time events of the compound Poisson process and the kernel $k(x, z) = k(x - z)$ represents the PDF of the sizes of its jumps. We add the source-term f to easily define an exact solution for validating our solution procedure.

Now, we consider the semi-discretization in time of (4.11) by an implicit Euler scheme on a time grid with time step $\delta t = \frac{T}{M}$, where M is a positive integer. Let

denote each time step $t^n = n\delta t,\ n = 0, 1, \ldots, M$. With this setting, one can easily verify that at the time step t^n the following boundary value problem needs to be solved

$$\begin{aligned} -\varepsilon\Delta y^n + \nabla\cdot(b\, y^n) - \lambda\, \mathcal{I} y^n + \left(\lambda + \frac{1}{\delta t}\right) y^n &= \frac{1}{\delta t} y^{n-1} + f, \qquad \text{in} \quad \Omega, \\ y &= 0, \qquad \text{on} \quad \Gamma, \end{aligned}$$

where the function y^{n-1} denotes the solution at the previous time step. Notice that this problem has a structure similar to (4.1), the only difference being the sign in front of the integral. However, the integral operator is bounded and therefore, with appropriate adaptation, we can use the error estimates in Theorem 14 and the numerical analysis tools in [47] to prove that the Euler-CCT solution to (4.11) has an accuracy of $\mathcal{O}(h^2 + \delta t)$.

We perform experiments with (4.11) approximated by our Euler-implicit CCT $\mathcal{H}$-matrices scheme. We consider a unit square domain and choose the drift term to be $b(x, z, t) = (x_1 + x_2, z_1 + z_2)$, $\varepsilon = 2$ and $\lambda = 1$, further we take $T = 1$. The kernel is as in the previous experiment with a normalisation coefficient equal to $1/(\pi\sqrt{2})$. To validate our algorithm, we assume an exact solution given by $y(x, z, t) = \sin(2\pi x)\sin(2\pi z)\sin(2\pi t)$. Certainly, this function cannot represent a PDF, however it is appropriate to test the accuracy of the solution and facilitates the computation of the corresponding source term.

In Figure 4.7, we present the values of the discrete $L^2(\mathcal{Q})$-norm of the solution error for different meshes with subsequent relative refinement of $h^2 + \delta t$ that scales by a factor of four. We see that the numerical accuracy improves by refinement as predicted. In Figure 4.8, we depict the computational time and memory requirements of the HLU preconditioner and of the GMRes solver for solving our time dependent PIDE problem.

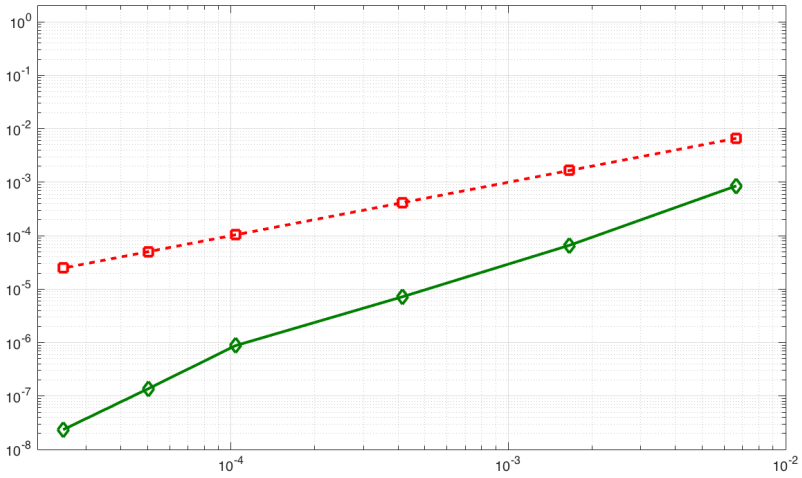


Figure 4.7: Logarithmic plot of the $L_2(\mathcal{Q})$-norm of the solution error (Green and $\diamond$) against $h^2 + \delta t$ depicted in (Red and $\square$).

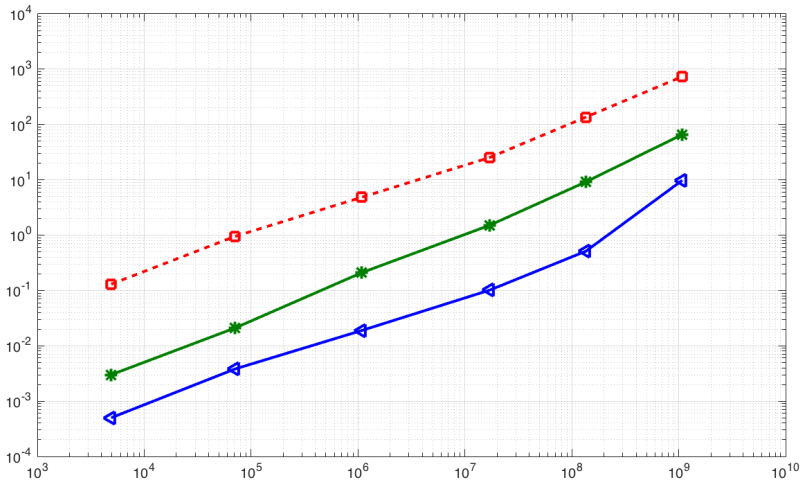


Figure 4.8: Computational requirements for the $\mathcal{H}$-matrices GMRes scheme. It is plotted: the precondition time (sec; Green and $*$); precondition storage (MB; Red and $\square$), and GMRes time (sec; Blue and $\triangle$).

4.7 Summary and remarks

A hierarchical matrix approach to the solution of convection-diffusion partial integro-differential problems was discussed. The convection-diffusion differential operator was discretized by using a second-order positive finite-volume scheme, while the integral term was approximated by the trapezoidal quadrature rule, thus resulting in a sparse matrix and a fully populated matrix, respectively. For

solving this problem, a $\mathcal{H}$-matrix approximation of these two operators was computed separately and assembled together to define the corresponding $\mathcal{H}$-matrix problem. Further, the $\mathcal{H}$-matrix framework was used to develop a LU preconditioner which was used by a GMRes scheme. An application of the proposed methodology to a three dimensional problem was considered as well and results included. Numerical analysis estimates of the accuracy of the finite-volume discretization and trapezoidal rule approximation were combined with estimates of the $\mathcal{H}$-matrix approximation and with the accuracy of the GMRes iterates.

Results of numerical experiments were presented that successfully validated the theoretical estimates and the optimal computational complexity of the proposed $\mathcal{H}$-matrix approach also in the three-dimensional case and in a time dependent setting.

5. PIDE Optimal control problems

Since the pioneering works of J.L. Lions [51], numerous results concerning the formulation and analysis of optimal control problems governed by partial differential equations (PDEs) have been published ; see, e.g. [16, 41, 64] and references therein. However, during these five decades, much less has been done for the investigation of control problems where the differential constraint is given by a partial-integro differential equation. One reason for this situation may be the difficulty in numerically solving the resulting optimality systems, since the presence of a Fredholm integral term results in non-sparse algebraic problems that are difficult to solve. On the other hand, in the recent past, multigrid techniques have been developed [16] that allow to solve different PDE-based optimal control problems with optimal computational complexity. Moreover, in a seperate research field, efforts have been put in developing multigrid techniques that solve multidimensional integral problems with optimal complexity [20, 33, 52]. The purpose of this chapter is to discuss the extension of our multigrid and fast integration strategy to solve an optimal control problem governed by a convection-diffusion PIDE model. The choice of this problem is motivated by the fact that its solution leads to consider a coupled system of PIDEs that provide a benchmark to validate our multigrid strategy in this case.

We consider a control u that is chosen among a family of admissible controls U_{ad}, while the state of the system y that depends on the control is given by the solution of the following PIDE problem

$$\begin{aligned} -\varepsilon\Delta y + \nabla\cdot(b\,y) + \mathcal{I}y + \lambda y = f + u, \quad &\text{in} \quad \Omega, \\ y = 0 \qquad\qquad\quad &\text{on} \quad \Gamma, \\ u \in U_{ad}. \end{aligned}$$

In this framework, the purpose of the control is modelled by the function $J : Y \times U \to \mathrm{R}_+ \cup \{0\}$ to be minimized. J is referred to as the objective function and

we consider an objective function of the tracking type given by

$$J(y,u) := \frac{1}{2}\|y - y_d\|^2_{L^2(\Omega)} + \frac{\nu}{2}\|u\|^2_{L^2(\Omega)}, \tag{5.1}$$

where $y_d \in L^2(\Omega)$ is the desired state and U_{ad} is the set of admissible controls. The term $\frac{1}{2}\|y - y_d\|^2_{L^2(\Omega)}$ is referred to as the tracking term. The parameter $\nu > 0$ is used to weight the L^2-norm of the control. The main focus of our work is the development of a computational tool that allows to solve the optimality system of this control-constrained convection-diffusion PIDE optimal control problem.

5.1 A PIDE optimal control problem

We consider an optimal control problem given by

$$\begin{cases} \min\ J(y,u) \\ -\varepsilon\Delta y + \nabla\cdot(b\,y) + \mathcal{I}y + \lambda y &= f + u, \quad \text{in} \quad \Omega, \\ y &= 0 \quad \text{on} \quad \Gamma, \\ u &\in U_{ad}, \end{cases} \tag{5.2}$$

where Ω is a convex, bounded and open set in $\mathbb{R}^2$, with Lipschitz boundary Γ. The integral term

$$\mathcal{I}y(x) = \int_\Omega k(x,z)y(z)dz,$$

with $x, z \in \Omega$, is a Hilbert-Schmidt integral operator with a symmetric positive semi-definite Hilbert-Schmidt kernel $k \in L^2(\Omega\times\Omega)$. We chose $f \in L^2(\Omega)$, the diffusion coefficient $\varepsilon > 0$, $\lambda > 0$, and the drift $b = (b_1, b_2)$ is a smooth vector-function in $\bar{\Omega} = \Omega \cup \Gamma$.

Further, we chose a bounded closed convex set of admissible controls in $L^2(\Omega)$ given by

$$U_{ad} = \left\{ u \in L^2(\Omega) |\ \underline{u} \le u \le \overline{u} \quad \text{a.e. in } \Omega \right\}, \tag{5.3}$$

where, $\underline{u} < \overline{u}$ and $\underline{u}, \overline{u} \in \mathbb{R}$.

In (5.2), we consider the minimization of the quadratic cost functional of tracking type given by (5.1).

We denote the dependence of the state y on the control u by $y = y(u)$ and the map $u \to y(u)$ from $L_2(\Omega)$ to $H^1_0(\Omega) \cap H^2(\Omega)$ is affine and continuous. We denote the first derivative of y with respect to u in the direction δu by $y'(\delta u)$. The

reduced cost functional $\hat{J}$ is given by

$$\hat{J}(u) = J(y(u), u). \tag{5.4}$$

The mapping $u \to \hat{J}(u)$ is Frechét differentiable and the second derivative is given by

$$\hat{J}''(u)(\delta u, \delta u) = \left\| y'(\delta u) \right\|^2_{L^2(\Omega)} + \nu \left\| \delta u \right\|^2_{L^2(\Omega)}.$$

It follows that $u \to \hat{J}(u)$ is uniformly convex if $\nu > 0$. This implies the existence of a unique solution u^* to (5.2) and this solution is characterised by $\hat{J}'(u^*)(\delta u) \geq 0$.

To formally obtain the necessary optimality conditions, we use the Lagrange approach [51, 64] and consider the following Lagrange function

$$\mathcal{L}(y, u, p) = J(y, u) + \int_\Omega \left[-\varepsilon \Delta y + \nabla \cdot (by) + \mathcal{I} y + \lambda y - u - f \right] p \, dx, \tag{5.5}$$

where p represents the adjoint variable.

Taking the variations with respect to y, u, and p, one obtains the first-order optimality conditions for (5.2) as follows

$$\begin{aligned} -\varepsilon \Delta y + \nabla \cdot (by) + \mathcal{I} y + \lambda y &= f + u && \text{in} \quad \Omega, \\ y &= 0 && \text{on} \quad \Gamma, \\ -\varepsilon \Delta p - b \nabla p + \mathcal{I} p + \lambda p &= -(y - y_d) && \text{in} \quad \Omega, \\ p &= 0 && \text{on} \quad \Gamma, \\ (\nu u - p, v - u)_{L^2(\Omega)} &\geq 0 && \forall v \in U_{ad}. \end{aligned} \tag{5.6}$$

The first equation is called the state equation, the second equation is called the adjoint equation, which has the same structure as the state equation, henceforth for a given $y, y_d \in L^2(\Omega)$ there exists unique $p \in H_0^1(\Omega) \cap H^2(\Omega)$; see Section 2.1. The inequality in (5.6) is referred to as the optimality condition. This variational inequality can be conveniently reformulated as illustrated in the following theorem.

Theorem 25 (cf. [64]). *Let $PU_{ad} : L_2(\Omega) \to U_{ad}$ be a pointwise projection*

$$PU_{ad}(u) := \max\left\{ \underline{u}, \min\left\{ u, \overline{u} \right\} \right\}. \tag{5.7}$$

If $\nu > 0$, then $(\nu u - p, v - u)_{L^2(\Omega)} \geq 0$ is equivalent to the equation

$$u = PU_{ad}\left\{ \frac{1}{\nu} \, p(u) \right\},$$

where $p \in H_0^1(\Omega) \cap H^2(\Omega)$ represents the weak solution to the adjoint equation corre-

sponding to $y = y(u)$.

Lemma 26. *The reduced functional* $\hat{J}(u)$ *is differentiable and its derivative is given by*

$$\hat{J}'(u)(\delta u) = (\nabla \hat{J}(u), \delta u)_{L^2(\Omega)} = (\nu u - p, \delta u)_{L^2(\Omega)}$$

where $\nabla \hat{J}(u)$ *denotes the* $L^2(\Omega)$ *reduced gradient and* p *is as in Theorem 25.*

Next, we discuss the discretization of the optimality system.

5.2 Discretization of the PIDE optimality system

In this section, we discuss the approximation of the optimality system (5.6). The diffusion and convection terms in the state and adjoint equations are discretized using the Chang and Cooper scheme and its transpose while the integral functions are approximated by using the Simpson's quadrature rule. Setting $h = \frac{b-a}{N-1}$ be the mesh size and consider an equidistant grid. We denote the mesh points $x_i = a + (i-1)h,\ i = 1, \ldots, N$, and $z_j = a + (j-1)h,\ j = 1, \ldots, N$. These grid points define the following grid

$$\Omega_h = \{\mathcal{Z}_{ij} = (x_i, z_j) \in \mathbb{R}^2 : i, j = 2, \ldots, N-1\} \cap \Omega.$$

Further, the functions in the spaces $L^2(\Omega)$ and $H^1(\Omega)$ are approximated by the grid functions defined through their mean values with respect to the elementary cells $\left[x_1 - \frac{h}{2}, x_1 + \frac{h}{2}\right] \times \left[x_2 - \frac{h}{2}, x_2 + \frac{h}{2}\right]$. With this setting, we have two restriction operators: the operator R_h denotes a mapping $R_h : H^1(\Omega) \to H_h^1$ and $\tilde{R}_h : L^2(\Omega) \to L_h^2$; see [14, 35, 47] for more details.

The governing model is approximated as follows (notice that the solution y is zero at the boundary)

$$-\hat{A}_{ij}y_{i+1,j} - \hat{B}_{ij}y_{i-1,j} + \hat{C}_{ij}y_{i,j} - \hat{D}_{ij}y_{i,j+1} - \hat{E}_{ij}y_{i,j-1} + h^2 \sum_{l=2}^{N-1}\sum_{m=2}^{N-1} r(l,m)k(z_{ij}, z_{lm})y(z_{lm}) = f_{i,j} + u_{i,j}, \quad (5.8)$$

where $i, j = 2, \ldots, N-1$, and the coefficients are given by

$$\hat{A}_{ij} = \frac{1}{h}\left[\frac{\varepsilon}{h} - (1 - \delta_i^j)b_1\right], \qquad \hat{B}_{ij} = \frac{1}{h}\left[\frac{\varepsilon}{h} + \delta_{i-1}^j b_1\right],$$

$$\hat{C}_{ij} = \left(\frac{1}{h}\left(\left[\frac{\varepsilon}{h} + \delta_i^j b_1\right] + \left[\frac{\varepsilon}{h} - (1 - \delta_{i-1}^j)b_1\right] + \left[\frac{\varepsilon}{h} + \delta_j^i b_2\right] + \left[\frac{\varepsilon}{h} - (1 - \delta_{j-1}^i)b_2\right]\right) + \lambda\right),$$

$$\hat{D}_{ij} = \frac{1}{h}\left[\frac{\varepsilon}{h} - (1 - \delta_j^i)b_2\right], \qquad \hat{E}_{ij} = \frac{1}{h}\left[\frac{\varepsilon}{h} + \delta_{j-1}^i b_2\right].$$

The algebraic problem given by (5.8) and including homogeneous Dirichlet boundary conditions can be formulated in the following matrix notation

$$\mathcal{A}^h y_h = f_h + u_h, \tag{5.9}$$

where $\mathcal{A}^h$ is a $(N-1)^2 \times (N-1)^2$ block dense matrix, and y_h, u_h, and f_h represent the numerical approximation of y, u, and f on Ω_h.

Next, we discuss the approximation of the adjoint equation. Analogous to the state equation, we can combine the CC scheme and the Simpson's quadrature rule to approximate our adjoint equation as follows

$$\begin{aligned} -\hat{a}_{ij}p_{i+1,j} - \hat{b}_{ij}p_{i-1,j} + \hat{c}_{ij}p_{i,j} - \hat{d}_{ij}p_{i,j+1} - \hat{e}_{ij}p_{i,j-1} + h^2 \sum_{l=2}^{N-1}\sum_{m=2}^{N-1} r(l,m)k(z_{ij},z_{lm})p(z_{lm}) \\ = -\left(y_{i,j} - y_{d_{i,j}}\right), \end{aligned} \tag{5.10}$$

where $i,j = 2,\ldots,N-1$, and the coefficients are given by

$$\hat{a}_{ij} = \frac{1}{h}\left[\frac{\varepsilon}{h} + (1-\delta_i^j)b_1\right], \qquad \hat{b}_{ij} = \frac{1}{h}\left[\frac{\varepsilon}{h} - \delta_{i-1}^j b_1\right],$$

$$\hat{c}_{ij} = \left(\frac{1}{h}\left(\left[\frac{\varepsilon}{h} - \delta_i^j b_1\right] + \left[\frac{\varepsilon}{h} + (1-\delta_{i-1}^j)b_1\right] + \left[\frac{\varepsilon}{h} - \delta_j^i b_2\right] + \left[\frac{\varepsilon}{h} + (1-\delta_{j-1}^i)b_2\right]\right) + \lambda\right),$$

$$\hat{d}_{ij} = \frac{1}{h}\left[\frac{\varepsilon}{h} + (1-\delta_j^i)b_2\right], \qquad \hat{e}_{ij} = \frac{1}{h}\left[\frac{\varepsilon}{h} - \delta_{j-1}^i b_2\right].$$

In [2,56], it is shown that the transpose of the CC scheme for the convection-diffusion operator appearing in our governing model (5.2) provides a suitable approximation of the corresponding adjoint equation. The discretization of the adjoint equation is a delicate issue since it directly influences the accuracy of the optimization gradient. We follow the discretize-before-optimize strategy (see, e.g., [16]) for the derivation of the discrete adjoint equation starting from the discrete Lagrange function. This derivation is discussed in detail in [2]. The result above is a direct result of considering the Lagrange function (5.5) in discrete form and performing discrete integration by parts with respect to the flux increment. Therefore starting from $(\nabla F, p)$, we have

$$\begin{aligned} \sum_{i,j=2}^{N-1} \left(F_{i+1/2,j} - F_{i-1/2,j}\right) p_{i,j} = \\ \sum_{i,j=1}^{N-1} \left(W_{i+1/2,j}y_{i+1,j} - T_{i+1/2,j}y_{i,j} + W_{i-1/2,j}y_{i,j} - T_{i-1/2,j}y_{i-1,j}\right) p_{i,j}, \end{aligned} \tag{5.11}$$

and likewise in the j-th direction where

$$W_{i+1/2,j} = \frac{\varepsilon}{h} - (1-\delta_i^j) b_{i+\frac{1}{2},j} \text{ and } W_{i-1/2,j} = \frac{\varepsilon}{h} - (1-\delta_{i-1}^j) b_{i-\frac{1}{2},j},$$

and

$$T_{i+1/2,j} = \frac{\varepsilon}{h} + \delta_i^j b_{i+\frac{1}{2},j} \text{ and } T_{i-1/2,j} = \frac{\varepsilon}{h} + \delta_{i-1}^j b_{i-\frac{1}{2},j}.$$

We recast the summation to collect the terms y_{ij} as follows

$$\sum_{i,j=1}^{N-1} W_{i+1/2} y_{i+1,j} p_{i,j} \to \sum_{i,j=2}^{N} W_{i-1/2} y_{i,j} p_{i-1,j}, \tag{5.12}$$

and

$$\sum_{i,j=1}^{N-1} T_{i-1/2,j} y_{i-1,j} p_{i,j} \to \sum_{i,j=2}^{N} T_{i+1/2,j} y_{i,j} p_{i+1,j}, \tag{5.13}$$

On simplification and incorporating the zero boundary conditions we have the following

$$\begin{aligned} &\sum_{i,j=2}^{N-1} \left(F_{i+1/2,j} - F_{i-1/2,j}\right) p_{i,j} \\ &\qquad = \sum_{i,j=2}^{N-1} \left(W_{i-1/2} p_{i-1,j} - T_{i+1/2,j} p_{i,j} - W_{i-1/2} p_{i,j} + T_{i+1/2,j} p_{i+1,j}\right) y_{i,j}, \end{aligned} \tag{5.14}$$

and likewise in the j-th direction.
Now taking the derivative with respect to y_{ij}, we obtain that the discrete adjoint equation is the transpose of the state equation discretized with the CC scheme.

Summarizing, the approximation of the adjoint equation results in the following linear algebraic problem

$$\mathcal{A}^{h,T} p_h = g_h, \tag{5.15}$$

where $\mathcal{A}^{h,T}$ is a $(N-1)^2 \times (N-1)^2$ block dense matrix and T denotes transpose and $g_h = -(y_h - \tilde{R}_h y_d)$ as discussed below.

In the discretize-before-optimize framework, we can start by formulating the following discrete optimal control problem

$$\begin{cases} \min \frac{1}{2} \left\| y_h - \tilde{R}_h y_d \right\|_{L_h^2}^2 + \frac{\nu}{2} \left\| u_h \right\|_{L_h^2}^2, \\ -\varepsilon \Delta_h y_h + \nabla_h (b \cdot y_h) + \mathcal{I}_h y_h + \lambda y_h = u_h + \tilde{R}_h f, \end{cases} \tag{5.16}$$

where $u_h \in U_{adh} = U_{ad} \cap L_h^2$.

Now, let u_h^* denote the unique optimal solution to (5.16) and $y_h^* = y_h(u_h^*)$ is obtained by solving the discretized PIDE model (5.8). Thus, the optimality system

for (5.16) is given by

$$\begin{aligned} \mathcal{A}^h y_h &= u_h + \tilde{R}_h f, \\ \mathcal{A}^{h,T} p_h &= \tilde{R}_h y_d - y_h, \\ (\nu u_h - p_h, v_h - u_h) &\geq 0, \qquad \forall v_h \in U_{adh}. \end{aligned} \tag{5.17}$$

We conclude this section with remarks concerning the accuracy of the optimal solution to (5.17). Notice that no results are available that apply directly to our case, since we use the finite-volume CC scheme and not the much studied cases using the finite-element method [41,71], and our model has also an integral term. However, our optimal control formulation and our discretization framework have many similarities with [10]. For this reason and based on previous experience [14], we may argue that in our case, for solutions where the constraints on the control are active, an accuracy of the optimal control of order $\frac{3}{2}$ in the L_h^2-norm can be expected.

On the other hand, if we assume that the control constraints are not active, then we can show that a second-order accurate solution is obtained with our approximation scheme. For this purpose, notice that in the case of inactive control constraints, the optimal solution satisfies $\nu u_h = p_h$ and we can eliminate u_h in the state equation in (5.17). Thus, we obtain the following optimality system

$$\begin{aligned} \mathcal{A}^h y_h - p_h/\nu &= \tilde{R}_h f, \\ \mathcal{A}^{h,T} p_h + y_h &= \tilde{R}_h y_d. \end{aligned} \tag{5.18}$$

Now, we can rearrange the ν factor and introduce a vector notation such that (5.18) can be written as $\tilde{\mathcal{A}}^h \Psi_h = \zeta_h$, with $\Psi_h = (y_h, p_h)$, where the matrix $\tilde{\mathcal{A}}_h$ and the right-hand side ζ_h of the system are given by

$$\tilde{\mathcal{A}}^h = \begin{pmatrix} \nu\,\mathcal{A}^h & -\mathrm{I}_h \\ \mathrm{I}_h & \mathcal{A}^{h,T} \end{pmatrix}, \qquad \zeta_h = \begin{pmatrix} \nu \tilde{R}_h f \\ \tilde{R}_h y_d \end{pmatrix}, \tag{5.19}$$

where I_h is the identity operator.

Next, notice that the solution error corresponding to Ψ_h, which we denote with $e_\Psi = (e_y, e_p)^T$, satisfies the error equation $\tilde{\mathcal{A}}^h e_\Psi = \varphi_h$, where $\varphi_h = (\varphi_y, \varphi_p)^T$ represents the truncation error of our CC-Simpson's-quadrature discretization of the optimality system. Further, we recall two properties of our problem discussed in [2,54,56]. The first property is that our CC convection-diffusion operator with homogeneous Dirichlet boundary conditions is positive definite. This fact combined with a positive semi-definite Hilbert-Schmidt kernel means that there exists

a positive constant $\hat{c} > 0$ such that

$$(\mathcal{A}^h v, v)_h \geq \hat{c}\, \|v\|^2_{L^2_h}, \qquad \forall v \in L^2_h, \tag{5.20}$$

where $\hat{c}$ depends on ϵ, b, λ and on the domain Ω. The same estimate holds for $\mathcal{A}^{h,T}$.

The second property that we need to recall is that assuming sufficient regular solutions, our approximation scheme is second-order consistent, which means that there exists a positive constant $c' > 0$ such that $\|\varphi_h\|_{L^2_h} \leq c'\, h^2$.

Now, based on the coercivity property (5.20), it follows that $\tilde{\mathcal{A}}_h$ has a bounded inverse and, similarly to [15], we obtain

$$\|y^*_h - R_h y^*\|_{L^2_h} + \|u^*_h - R_h u^*\|_{L^2_h} + \|p^*_h - R_h p^*\|_{L^2_h} \leq \tilde{c}\, h^2, \tag{5.21}$$

where $\tilde{c} > 0$ depends on the problem's parameters and does not depend on h.

In the section of numerical experiments, we validate successfully this estimate for the control-unconstrained case. Moreover, we demonstrate that, in the case of active control constraints, a control with accuracy of order $O(h^{3/2})$ is obtained.

5.3 A multigrid scheme for PIDE optimality systems

In this section, we extend our FAS-FI method (see Chapter 3) to solve the optimality system (5.17). The choice of a nonlinear multigrid scheme is motivated by the need of imposing constraints on the control variable at all grid levels. We start considering only two nested grids corresponding to the fine grid Ω_h and the coarse grid Ω_H, $H = 2h$.

We choose different inter-grid transfer operators for the FAS scheme and for the FI method. For the former, we use the $2nd$-order full-weight restriction and linear interpolation. We denote these operators with $\mathbb{I}^H_h$ and $\mathbb{I}^h_H$, respectively. This choice is consistent with the FAS practice for solving elliptic problems. In the FI procedure, we utilize $4th$-order restriction and interpolation operators [33, 40] and denote them with $\tilde{\mathbb{I}}^H_h$ and $\tilde{\mathbb{I}}^h_H$, respectively. This choice results from the discussion in [20, 52] and in Section 3.1 and is consistent with the higher-order approximation of the Simpson scheme. Notice that for grid points next to the boundary, an asymmetric fourth-order interpolation formula is used [65, 69].

Now, we recall the nonlinear multigrid FAS [16, 21, 65, 68] framework for the case of our optimality system. In this framework, the coarse grid problem is

constructed on Ω_H as follows

$$-\varepsilon\Delta_H y_H + \nabla_H(b \cdot y_H) + \mathbb{I}_h^H \mathcal{I}^h y_h + \lambda y_H = u_H + \mathbb{I}_h^H f_h + \tau(y)_h^H, \tag{5.22}$$

$$-\varepsilon\Delta_H p_H - b \cdot \nabla_H(p_H) + \mathbb{I}_h^H \mathcal{I}^h p_h + \lambda p_H = \mathbb{I}_h^H y_{dh} + y_H + \tau(p)_h^H, \tag{5.23}$$

$$(\nu u_H - p_H) \cdot (v_H - u_H) \geq 0 \quad \forall v_H \in U_{adH}. \tag{5.24}$$

where $\tau(y)_h^H$ and $\tau(p)_h^H$ are the fine -to-coarse grid defect corrections defined by

$$\tau(y)_h^H = -\varepsilon\Delta_H \mathbb{I}_h^H y_h + \nabla_H(b \cdot \mathbb{I}_h^H y_h) + \mathbb{I}_h^H \mathcal{I}^h y_h + \lambda \mathbb{I}_h^H y_h - \mathbb{I}_h^H \left(-\varepsilon\Delta_h y_h + \nabla_h(b \cdot y_h) + \mathcal{I}^h y_h + \lambda y_h\right), \tag{5.25}$$

$$\tau(p)_h^H = -\varepsilon\Delta_H \mathbb{I}_h^H p_h - b.\nabla_H(\mathbb{I}_h^H p_h) + \mathbb{I}_h^H \mathcal{I}^h p_h + \lambda \mathbb{I}_h^H p_h - \mathbb{I}_h^H \left(-\varepsilon\Delta_h p_h - b.\nabla_h(p_h) + \mathcal{I}^h p_h + \lambda p_h\right). \tag{5.26}$$

Once the coarse grid problem (5.22-5.26) is solved, thus obtaining (y_H, p_H), we have the coarse-grid corrections as follows

$$y_h^{new} = y_h + \mathbb{I}_H^h \left(y_H - \mathbb{I}_h^H y_h\right), \tag{5.27}$$

$$p_h^{new} = p_h + \mathbb{I}_H^h \left(p_H - \mathbb{I}_h^H p_h\right). \tag{5.28}$$

If the solution error on the finer grid is well damped by a smoothing scheme, then the grid Ω_H should provide sufficient resolution of the error of Ψ_h and hence $\Psi_H - \mathbb{I}_h^H \Psi_h$ should be a good approximation to this error. Denoting the number of pre- and post-smoothing iterations with m_1 and m_2 respectively, the multigrid FAS-(m_1, m_2) cycle algorithm is implemented recursively as outlined in the Algorithm 11. The FI procedure is used to evaluate the integrals $\mathcal{I}^h y_h$ and $\mathcal{I}^h p_h$. As outlined earlier, this method aims at performing integration mostly on coarser grids and to interpolate the resulting integral function to the original fine grid where the integral is required. See the discussion in Section 3.2 and Algorithm 2. Next, we discuss the smoothing iteration used in our multigrid scheme. As in [16], our approach involves solving the state, the adjoint and the control variables simultaneously in the multigrid process by implementing a collective update of the optimization variables. This procedure aims at realizing tight coupling in the optimality system along the hierarchy of grids such that it is robust with respect to the choice of the value of the optimization parameter, ν.

In order to illustrate our smoothing procedure, we consider the discrete opti-

mality system at $z_{ij} \in \Omega_h$ and without control constraints. We have

$$\begin{aligned} -\hat{A}_{ij}y_{i+1,j} - \hat{B}_{ij}y_{i-1,j} + \hat{C}_{ij}y_{i,j} - \hat{D}_{ij}y_{i,j+1} - \hat{E}_{ij}y_{i,j-1} + (\mathcal{I}^h y_h)_{ij} - u_{i,j} &= f_{i,j} \\ -\hat{a}_{ij}p_{i+1,j} - \hat{b}_{ij}p_{i-1,j} + \hat{c}_{ij}p_{i,j} - \hat{d}_{ij}p_{i,j+1} - \hat{e}_{ij}p_{i,j-1} + (\mathcal{I}^h p_h)_{ij} + y_{i,j} &= y_{d_{i,j}} \\ \nu u_{ij} - p_{ij} &= 0. \end{aligned} \tag{5.29}$$

Now, we use the first two equations to obtain the maps $y_{ij} = y_{ij}(u_{ij})$ and $p_{ij} = p_{ij}(u_{ij})$. The latter is replaced in the third equation to obtain an equation for u_{ij}. Notice that the values of the variables on the neighbouring points are considered as constants in this calculation. We obtain

$$y_{ij}(u_{ij}) := \frac{u_{i,j} - \bar{A}}{\hat{C}_{ij}}, \tag{5.30}$$

and

$$p_{ij}(u_{ij}) := \frac{-y_{i,j}(u_{ij}) - \bar{B}}{\hat{c}_{ij}}. \tag{5.31}$$

where

$$\begin{aligned} \bar{A} &= -\hat{A}_{ij}y_{i+1,j} - \hat{B}_{ij}y_{i-1,j} - \hat{D}_{ij}y_{i,j+1} - \hat{E}_{ij}y_{i,j-1} + (\mathcal{I}^h y_h)_{ij} - f_{i,j}, \\ \bar{B} &= -\hat{a}_{ij}p_{i+1,j} - \hat{b}_{ij}p_{i-1,j} - \hat{d}_{ij}p_{i,j+1} - \hat{e}_{ij}p_{i,j-1} + (\mathcal{I}^h p_h)_{ij} - y_{d_{i,j}}. \end{aligned} \tag{5.32}$$

Now, substituting (5.30) into (5.31) and requiring that $\nu u_{ij} - p_{ij}(u_{ij}) = 0$, we obtain

$$\tilde{u}_{i,j} = \frac{1}{(1 + \hat{c}_{ij}\hat{C}_{ij}\nu)} \left(\bar{A} - \hat{C}_{ij}\bar{B}\right). \tag{5.33}$$

This $\tilde{u}_{i,j}$ represents the update value for the control in the control-unconstrained case. In the presence of control constraints, this value needs to be projected in the admissible set of controls. In our case, we have

$$u_{i,j} = \begin{cases} \overline{u}_{i,j} & \text{if} \quad \tilde{u}_{i,j} > \bar{u}_{i,j}, \\ \tilde{u}_{i,j} & \text{if} \quad \underline{u}_{i,j} \leq \tilde{u}_{i,j} \leq \overline{u}_{i,j}, \\ \underline{u}_{i,j} & \text{if} \quad \tilde{u}_{i,j} < \underline{u}_{i,j}. \end{cases} \tag{5.34}$$

With this new values of $u_{i,j}$, new values for $y_{i,j}$ and $p_{i,j}$ are obtained by (5.30) and (5.31). This update step can be applied once on all grid points in any order to define a smoothing iteration sweep. During a sweep, the integral function is not being updated.

We summarize this collective smoothing procedure with the following Algorithm 10.

Algorithm 10 Collective smoothing and fast integration (CSMA-FI) procedure

Input: y_{ij}^0,p_{ij}^0,u_{ij}^0, f_{ij}^0, z_{ij}^0 .

1. Call the FI procedure to compute $\mathcal{I}^h y_h^0$ and $\mathcal{I}^h p_h^0$
2. Compute $\bar{A}$ and $\bar{B}$ according to (5.32) and determine u_{ij} by (5.33) and (5.34).
3. Compute y_{ij} by (5.30) and p_{ij} by (5.31)
4. End

Now, we can define our FAS-FI scheme for solving (5.17) with the smoothing procedure just described. In Algorithm 11, the FI procedure is integrated in the FAS scheme to evaluate the Fredholm integral terms at any working level. In this algorithm, the smoothing procedure on the grid level k is denoted by $\mathcal{S}_k$ and $\mathcal{S}_k^m$ is the smoothing operator applied m-times. Notice that Algorithm 11 defines one cycle of the FAS-FI scheme. This cycle is repeated many times until a convergence criterion is met.

Algorithm 11 Multigrid FAS-FI (m_1, m_2) cycle

Input: $\Psi_k^{(0)}$, $k = l$.

1. If $k = 1$ solve exactly $\tilde{\mathcal{A}}_k \Psi_k = \zeta_k$.
2. Perform fast integration to obtain $\mathcal{I}^k y_k^{(0)}$ and $\mathcal{I}^k p_k^{(0)}$ (Algorithm 2).
3. Pre-smoothing. Compute $\Psi_k^{(m_1)}$ with
$$\Psi_k^{(\ell)} = \mathcal{S}_k(\Psi_k^{(\ell-1)}, \zeta_k)$$
for $\ell = 1, \ldots, m_1$; (Algorithm 10).
4. Perform fast integration to obtain $\mathcal{I}^k y_k^{(m_1)}$ and $\mathcal{I}^k p_k^{(m_1)}$ (Algorithm 2).
5. Compute the residue : $r_k = \left(\zeta_k - \tilde{\mathcal{A}}_k \Psi_k^{(m_1)}\right)$;
6. Restriction of the residue: $r_{k-1} = \mathbb{I}_k^{k-1} r_k$;
7. Set $\Psi_{k-1} = \mathbb{I}_k^{k-1} \Psi_k^{(m_1)}$;
8. Set $\zeta_{k-1} = r_{k-1} + \tilde{\mathcal{A}}_{k-1} \Psi_{k-1}$;
9. Call γ times the FAS-FI scheme to solve $\tilde{\mathcal{A}}_{k-1} \Psi_{k-1} = \zeta_{k-1}$;
10. Coarse grid correction: $\Psi_k^{(m_1+1)} = \Psi_k^{(m_1)} + \mathbb{I}_{k-1}^k \left(\Psi_{k-1} - \mathbb{I}_k^{k-1} \Psi_k^{(m_1)}\right)$;
11. Perform fast integration to obtain $\mathcal{I}^k y_k^{(m_1+1)}$ and $\mathcal{I}^k p_k^{(m_1+1)}$ (Algorithm 2).
12. Post smoothing: Compute $\Psi_k^{(m_1+m_2+1)}$ with
$$\Psi_k^{(\ell)} = \mathcal{S}_k(\Psi_k^{(\ell-1)}, \zeta_k)$$
for $\ell = m_1 + 2, \ldots, m_1 + m_2 + 1$ (Algorithm 10).
13. End

5.4 Local Fourier analysis

In this section, we investigate the convergence properties of our FAS-FI scheme for solving the optimality system (5.17) by local Fourier analysis (LFA). Specifically, we extend the LFA presented in [15] to our multigrid scheme for solving a PIDE optimality system. In order to ease notation, we consider a one-dimensional case and use h and H to denote the variables on the fine and coarse grids, respectively. For the purpose of LFA analysis, we assume that the kernel of the inte-

gral term is translational invariant, that is, $k(x,z) = k(|x-z|)$ and require that $k(|x-z|)$ decays rapidly to zero as $|x-z|$ becomes large. The first condition is needed to obtain a translational invariant stencil of the PIDE operator, as usual in the LFA framework; the second condition is required to have a well-posed integral operator on the infinite grid.

Now, recall the family of operators given by (5.19). Notice that $\tilde{\mathcal{A}}^h$ can be decomposed as $\tilde{\mathcal{A}}^h = \tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,+} - \tilde{\mathcal{A}}^{h,-}$. To analyse the CSMA full kernel scheme, we define

$$\begin{aligned}
\tilde{\mathcal{A}}^{h,+} &= \begin{bmatrix} -\nu\hat{A} + \nu h \sum_{l=-\infty}^{\infty} k_{jl} & 0 \\ 0 & -\hat{a} + h \sum_{l=-\infty}^{\infty} k_{jl} \end{bmatrix}, \\
\tilde{\mathcal{A}}^{h,-} &= \begin{bmatrix} -\nu\hat{C} & 0 \\ 0 & -\hat{c} \end{bmatrix}, \\
\tilde{\mathcal{D}}^{h} &= \begin{bmatrix} \nu\hat{B} & -\mathrm{I}_h \\ \mathrm{I}_h & \hat{b} \end{bmatrix}.
\end{aligned} \tag{5.35}$$

With this choice, lexicographic CSMA sweeps are given by

$$\left(\tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,+}\right)\Psi^{(1)} - \tilde{\mathcal{A}}^{h,-}\Psi^{(0)} = \zeta_h \quad \text{and} \quad \left(\tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,-}\right)\Psi^{(2)} - \tilde{\mathcal{A}}^{h,+}\Psi^{(1)} = \zeta_h. \tag{5.36}$$

From equations (5.36), we can deduce that

$$\Psi^{(1)} = \left(\tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,+}\right)^{-1}\left[\zeta_h + \tilde{\mathcal{A}}^{h,-}\Psi^{(0)}\right] \quad \text{and} \quad \Psi^{(2)} = \left(\mathcal{D}^h - \tilde{\mathcal{A}}^{h,-}\right)^{-1}\left[\zeta_h + \tilde{\mathcal{A}}^{h,+}\Psi^{(1)}\right] \tag{5.37}$$

It follows that

$$\begin{aligned}
\Psi^{(1)} &= \left(\tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,+}\right)^{-1}\left[\zeta_h + \tilde{\mathcal{A}}^{h,-}\Psi^{(0)}\right] \\
&= \left(\tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,+}\right)^{-1}\zeta_h + \left(\tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,+}\right)^{-1}\left(-\tilde{\mathcal{A}}^h + \tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,+}\right)\Psi^{(0)} \\
&= \left(\tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,+}\right)^{-1}\zeta_h + \left(\mathrm{I} - \left(\tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,+}\right)^{-1}\tilde{\mathcal{A}}^h\right)\Psi^{(0)}
\end{aligned} \tag{5.38}$$

Subsequently it follows that

$$\begin{aligned}
\Psi^{(2)} &= \left(\tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,-}\right)^{-1}\left[\zeta_h + \tilde{\mathcal{A}}^{h,+}\Psi^{(1)}\right] \\
&= \left(\tilde{\mathcal{D}}_h - \tilde{\mathcal{A}}^{h,-}\right)^{-1}\left[\zeta_h + \tilde{\mathcal{A}}^{h,+}\left(\left(\tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,+}\right)^{-1}\zeta_h + \left(\mathrm{I} - \left(\tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,+}\right)^{-1}\tilde{\mathcal{A}}^h\right)\Psi^{(0)}\right)\right]
\end{aligned} \tag{5.39}$$

Using $\tilde{\mathcal{A}}^{h,+} = \tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^h - \tilde{\mathcal{A}}^{h,-}$, we get the following equation

$$\Psi^{(2)} = \left(\tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,-}\right)^{-1} \zeta_h + \left(\mathrm{I} - \left(\tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,-}\right)^{-1}\right) \times \left[\left(\tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,+}\right)^{-1} \zeta_h + \Psi^{(0)} - \left(\tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,+}\right)^{-1} \tilde{\mathcal{A}}^h \Psi^{(0)}\right]$$

Utilizing the relation $\tilde{\mathcal{D}}^h = \left(\tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,-}\right) + \left(\tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,+}\right) - \tilde{\mathcal{A}}^h$, we deduce that the resulting iteration is given by

$$\Psi^{(2)} = \Psi^{(0)} + \Lambda^h \left[\zeta_h - \tilde{\mathcal{A}}^h \Psi^{(0)}\right] \quad \text{and} \quad \Lambda^h = \left(\tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,+}\right)^{-1} \tilde{\mathcal{D}}^h \left(\tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,-}\right)^{-1} \tag{5.40}$$

Now, recall the setting for the local Fourier analysis given in Section 3.3. We consider the function basis

$$\phi_h(\theta, x) = \mathbf{a}\, e^{i\theta x/h}, \qquad \theta \in (-\pi, \pi],$$

where $\mathbf{a} = (1,1)^T$. For any low frequency $\theta^0 \in [-\pi/2, \pi/2)$, we consider

$$\theta^1 = \theta^0 - \text{signum}\left(\theta^0\right) \pi. \tag{5.41}$$

We have $\phi_h(\theta^0, \cdot) = \phi_h(\theta^1, \cdot)$ for $\theta^0 \in [-\pi/2, \pi/2)$ and $x \in G_H$. We have $\phi_h(\theta, x) = \phi_H(2\theta^0, x)$ on G_H for $\theta = \theta^0$ and $\theta = \theta^1$.

The two components $\phi_h(\theta^0, \cdot)$ and $\phi_h(\theta^1, \cdot)$ are called harmonics. For a given $\theta^0 \in [-\pi/2, \pi/2)$, the space of harmonics is defined by $E_h^\theta = span[\phi_h(\theta^\alpha, \cdot) : \alpha \in \{0,1\}]$. For each θ and a translational invariant kernel, the space $E_h^\theta \times E_h^\theta$ is invariant under the action of TG_h^H given by

$$TG_h^H = \mathcal{S}_h^{m_2} \left[\mathbb{I}_h - \mathbb{I}_H^h \left(\left(\tilde{\mathcal{A}}^H\right)^{-1}\right) \mathbb{I}_h^H \tilde{\mathcal{A}}^h\right] \mathcal{S}_h^{m_1}, \tag{5.42}$$

where the coarse-grid operator is given by $CG_h^H = \mathbb{I}_h - \mathbb{I}_H^h \left(\left(\tilde{\mathcal{A}}^H\right)^{-1}\right) \mathbb{I}_h^H \tilde{\mathcal{A}}^h$ on an couple $(\psi_y, \psi_p) \in E_h^\theta \times E_h^\theta$ where

$$\psi_y = \sum_{\alpha,\theta} Y_\theta^\alpha \phi_h(\theta^\alpha, x) \quad \text{and} \quad \psi_p = \sum_{\alpha,\theta} P_\theta^\alpha \phi_h(\theta^\alpha, x).$$

We analyze how the vector of coefficients $\left(Y^0, Y^1, P^0, P^1\right)$ are transformed if the two-grid iteration (5.42) is applied to $\left(\psi_y, \psi_p\right)$. To do this, we study the action of TG_h^H on the following function

$$\psi(x_j) = \sum_{\alpha,\theta} \hat{E}_\theta^\alpha \phi_h(\theta^\alpha, x_j), \qquad x_j \in G_h.$$

where $\hat{E}_\theta^\alpha = \begin{pmatrix} Y_\theta^\alpha \\ P_\theta^\alpha \end{pmatrix}$. Specifically, we determine how the amplitudes $\hat{E}_\theta^\alpha$, $\theta \in [-\frac{\pi}{2}, \frac{\pi}{2})$ and $\alpha = 0, 1$, are transformed under the action of the two-grid operator. This requires to calculate the Fourier symbols of the components that enter in the construction of the TG operator.

Notice that the symbols of the discrete operators $\left(\tilde{\mathcal{D}}^h - \tilde{\mathcal{A}}^{h,+}\right)$ and $\tilde{\mathcal{A}}^{h,-}$ given in (5.35) appear to depend on j. This is actually not the case since a translational invariant kernel $k(x,z) = k(|x-z|)$ is considered. In fact, consider the integral term $h \sum_{l=-\infty}^{\infty} k_{jl} e_l^m$ and introduce the index $n = j - l$. Further, because the element $k_{jl} = k_{|j-l|} = k_{|n|}$ becomes small as n becomes large, we truncate the sum in the integral on the infinite grid and replace it with the following $h \sum_{n=-L}^{L} k_{|n|} e_{j-n}^m$ where we assume that the partial sum provides a sufficiently accurate approximation of the integral term on G_h. Specifically, assuming that $k(|x-z|) = \exp(-|x-z|^2)$ and requiring that the $k_{|n|} = \mathcal{O}(10^{-16})$ (double precision machine epsilon) for $|n| > L$, one should choose $L \propto 1/h$. However, in practice, a much smaller L also results in accurate LFA estimates.

Now, we proceed with our analysis as follows. For the Fourier symbol of the forward CSMA scheme that we use in our implementation, we have

$$\overline{\left(\hat{\mathcal{D}}^h - \hat{\mathcal{A}}^{h,+}\right)}(\theta) = \begin{bmatrix} \nu\hat{B} + \nu\hat{A}e^{i\theta} - \nu h \sum_{n=-L}^{L} k_{|n|} e^{-i\theta n} & -\mathrm{I}_h \\ \mathrm{I}_h & \hat{b} + \hat{a}e^{\theta i} - h \sum_{n=-L}^{L} k_{|n|} e^{-i\theta n} \end{bmatrix}, \tag{5.43}$$

and

$$\overline{\left(\hat{\mathcal{A}}^{h,-}\right)}(\theta) = \begin{bmatrix} -\nu\hat{C}e^{-i\theta} & 0 \\ 0 & -\hat{c}e^{-i\theta} \end{bmatrix}. \tag{5.44}$$

Hence, we define

$$\overline{\mathcal{S}_h^+}(\theta) = \left(\overline{\left(\hat{\mathcal{D}}^h - \hat{\mathcal{A}}^{h,+}\right)}(\theta)\right)^{-1} \overline{\left(\hat{\mathcal{A}}^{h,-}\right)}(\theta).$$

In the one-grid LFA setting, the smoothing factor of this iteration sweep is defined as follows

$$\mu = \mu\left(\mathcal{S}_h^+\right) = \max_{\frac{\pi}{2} \le |\theta| \le \pi} \left\{ |\rho\left(\overline{\mathcal{S}_h^+}(\theta)\right)| \right\},$$

where ρ represents the spectral radius. In the same way one defines the Fourier symbol of the backward CSMA scheme, $\overline{\mathcal{S}_h^-}(\theta) = \left(\overline{\left(\hat{\mathcal{D}}^h - \hat{\mathcal{A}}^{h,-}\right)}(\theta)\right)^{-1} \overline{\left(\hat{\mathcal{A}}^{h,+}\right)}(\theta)$ as follows

$$\overline{\left(\hat{\mathcal{D}}^h - \hat{\mathcal{A}}^{h,-}\right)}(\theta) = \begin{bmatrix} \nu\hat{B} + \nu\hat{C}e^{i\theta} & -\mathrm{I}_h \\ \mathrm{I}_h & \hat{b} + \hat{c}e^{\theta i} \end{bmatrix}, \tag{5.45}$$

and

$$\overline{(\hat{\mathcal{A}}^{h,+})}(\theta) = \begin{bmatrix} -\nu\hat{A}e^{-i\theta} + \nu h \sum_{n=-L}^{L} k_{|n|}e^{-i\theta n} & 0 \\ 0 & -\hat{a}e^{-i\theta} + \nu h \sum_{n=-L}^{L} k_{|n|}e^{-i\theta n} \end{bmatrix}. \quad (5.46)$$

To determine the Fourier symbols of the TG scheme, we recall the following theorem [15,65].

Theorem 27. *Under the assumption that the components in (5.42) are linear and that $(\hat{\mathcal{A}}^H)^{-1}$ exists, the coarse grid operator CG_h^H is represented on E_h^θ by a 4×4 matrix $\widehat{CG}_h^H(\theta)$,*

$$\widehat{CG}_h^H(\theta) = \left[\hat{\mathbb{I}}_h - \hat{\mathbb{I}}_H^h(\theta)\left(\hat{\mathcal{A}}^H(2\theta)\right)^{-1}\hat{\mathbb{I}}_h^H(\theta)\hat{\mathcal{A}}^h(\theta)\right],$$

for each $\theta \in [-\pi/2, \pi/2)$. $\hat{\mathbb{I}}_h$ and $\hat{\mathcal{A}}^h(\theta)$ are 4×4 matrices, $\hat{\mathbb{I}}_H^h(\theta)$ is 4×2 matrix, $\hat{\mathbb{I}}_h^H(\theta)$ is 2×4 matrix and $\hat{\mathcal{A}}^H(2\theta)$ is a 2×2 matrix. If the space $E_h^\theta \times E_h^\theta$ is invariant under the smoothing operator $\mathcal{S}_h$ i.e $\mathcal{S}_h\colon E_h^\theta \times E_h^\theta \to E_h^\theta \times E_h^\theta$ for all $\theta \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right)^2$, TG_h^H is represented by 4×4 matrix on E_h^θ by

$$\widehat{TG}_h^H(\theta) = \hat{\mathcal{S}}_h(\theta)^{m_2}\widehat{CG}_h^H(\theta)\hat{\mathcal{S}}_h(\theta)^{m_1}.$$

Next, in order to apply this theorem, we construct the symbols of the operators in explicit form. The symbol of the coarse-grid operator $\mathcal{A}_H$ is given by

$$\hat{\mathcal{A}}^H(2\theta) = \begin{bmatrix} -\nu\hat{A}e^{i2\theta} + \nu\hat{B} - \nu\hat{C}e^{-i2\theta} + \nu h \sum_{n=-\frac{L}{2}}^{\frac{L}{2}} k_{|n|}\, e^{-i(2\theta)n} & -1 \\ 1 & -\hat{a}e^{i2\theta} + \hat{b} - \hat{c}e^{-i2\theta} + h \sum_{n=-\frac{L}{2}}^{\frac{L}{2}} k_{|n|}\, e^{-i(2\theta)n} \end{bmatrix}.$$

The symbol $\hat{\mathcal{A}}^h(\theta)$ of the fine-grid operator $\tilde{\mathcal{A}}^h$ is given by

$$\hat{\mathcal{A}}^h(\theta) = \begin{bmatrix} \hat{\mathcal{A}}^h(\theta^0) & 0 \\ 0 & \hat{\mathcal{A}}^h(\theta^1) \end{bmatrix},$$

where

$$\hat{\mathcal{A}}^h(\theta) = \begin{bmatrix} -\nu\hat{A}e^{i\theta} + \nu\hat{B} - \nu\hat{C}e^{-i\theta} + \nu h \sum_{n=-L}^{L} k_{|n|}e^{-i\theta n} & -1 \\ 1 & -\hat{a}e^{i\theta} + \hat{b} - \hat{c}e^{-i\theta} + h \sum_{n=-L}^{L} k_{|n|}e^{-i\theta n} \end{bmatrix}.$$

The symbol $\hat{\mathbb{I}}_h^H(\theta)$ of the restriction operator $\mathbb{I}_h^H$ is given by

$$\begin{bmatrix} \frac{\cos(\theta^0)+1}{2} & \frac{\cos(2\theta^1)+1}{2} & 0 & 0 \\ 0 & 0 & \frac{\cos(\theta^0)+1}{2} & \frac{\cos(2\theta^1)+1}{2} \end{bmatrix}.$$

The symbol $\hat{\mathbb{I}}_H^h(\theta)$ of the prolongation operator $\mathbb{I}_H^h$ is given by

$$\hat{\mathbb{I}}_H^h(\theta) = \left(\hat{\mathbb{I}}_h^H(\theta)\right)^T.$$

For the smoothing iteration $\mathcal{S}_h$, we have

$$\widehat{\mathcal{S}}_h(\theta) = \begin{bmatrix} \hat{\mathcal{S}}_h(\theta^0) & 0 \\ 0 & \hat{\mathcal{S}}_h(\theta^1) \end{bmatrix}, \tag{5.47}$$

In the LFA framework, the convergence factor is given by

$$\eta\left(TG_h^H\right) = \sup\left\{\rho\left(\widehat{TG}_h^H(\theta)\right) : \theta \in [-\pi/2, \pi/2)\right\}. \tag{5.48}$$

A simpler but less accurate estimate of the convergence factor can be obtained assuming an ideal coarse-grid correction which completely annihilates the low-frequency error components and leaves the high-frequency error components unchanged. In this case, define the projection operator $\mathcal{Q}_h^H$ on E_h^θ as

$$\widehat{\mathcal{Q}}_h^H(\theta) = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \qquad \text{for} \quad [-\pi/2, \pi/2).$$

Hence the TG convergence factor is solely determined by the smoothing property of $\mathcal{S}_h$ as follows

$$\eta_{ideal}^{m_1+m_2} = \sup\left\{ \sqrt[m_1+m_2]{\left|\rho\left(\hat{\mathcal{Q}}_h^H(\theta)\hat{\mathcal{S}}_h(\theta)^{m_1+m_2}\right)\right|} : \theta \in [-\pi/2, \pi/2)\right\}. \tag{5.49}$$

Notice that in the two-grid LFA setting, the smoothing factor is defined as follows

$$\mu = \sup\left\{\left|\rho\left(\hat{\mathcal{Q}}_h^H(\theta)\hat{\mathcal{S}}_h(\theta)\right)\right| : \theta \in [-\pi/2, \pi/2)\right\}. \tag{5.50}$$

In Table 5.1, we report results of LFA estimates using (5.48) and (5.49). These estimates appear accurate compared to the results of the numerical experiments presented in the next section, where we choose $m_1 = 2$ and $m_2 = 2$ for the pre- and post-smoothing steps, respectively.

Table 5.1: LFA Estimates of convergence factors.

(m_1, m_2)	$\eta_{ideal}^{m_1+m_2}$	$\eta\left(TG_h^H\right)$
$(1,1)$	1.6e-1	7.1e-2
$(2,2)$	2.1e-2	1.2e-2
$(3,3)$	4.2e-3	1.9e-3
$(4,4)$	6.8e-4	3.0e-4
$(5,5)$	1.1e-4	4.9e-5

5.5 Numerical experiments

In this section, we present results of numerical experiments to validate our multigrid framework for solving the convection-diffusion PIDE optimality system.

We report values of the tracking norm $\|y - y_d\|_{L^2(\Omega)}$ depending on the cost of the control and the convergence rates obtained by our FAS-FI algorithm. For all numerical experiments, we consider a unit square domain $\Omega = (0,1)^2 \subset \mathbb{R}^2$. We consider the drift term to be $b(x,z) = (x_1 + x_2, z_1 + z_2)$, $\varepsilon = 0.5$ and the kernel of the integral as $k(x,z) = \exp(-|x-z|^2)$. In all the experiments, we use $m_1 = m_2 = 2$ smoothing steps.

To have an exact solution that is required to validate our solution accuracy estimate, we assume the following state and adjoint solutions and adapt the right-hand sides of the state and adjoint equations (i.e. f and y_d) correspondingly

$$y(x,z) = \sin(2\pi x)\sin(2\pi z), \tag{5.51}$$

$$p(x,z) = \nu \sin(2\pi x)\sin(2\pi z), \tag{5.52}$$

$$u(x,z) = \frac{1}{\nu} p(x,z). \tag{5.53}$$

The solutions (5.51)-(5.53) represent the exact solutions for the unconstrained case that we consider in the first experiment whose results are presented in Table 5.2. We can see in this table that the state, adjoint and the control solutions have $\mathcal{O}(h^2)$ order of accuracy.

Table 5.2: $L^2(\Omega)$ norm of solution error; ν=1e-2. Control-unconstrained case.

Mesh	$\|y - y_h\|_{L^2(\Omega)}$	Order	$\|p - p_h\|_{L^2(\Omega)}$	Order	$\|u - u_h\|_{L^2(\Omega)}$	Order
17×17	6.48e-3	-	6.49e-5	-	6.49e-3	-
33×33	1.61e-3	2.01	1.63e-5	1.99	1.63e-3	1.99
65×65	4.02e-4	2.00	4.22e-6	1.95	4.21e-4	1.95
129×129	1.00e-4	2.00	1.06e-6	1.99	1.07e-4	1.97

For the constrained test cases, the exact solutions are defined as follows

$$y(x,z) = \sin(2\pi x)\sin(2\pi z), \tag{5.54}$$
$$p(x,z) = \nu \sin(2\pi x)\sin(2\pi z), \tag{5.55}$$
$$u(x,z) = \max\left\{-0.5, \min\left\{1, \frac{1}{\nu}p(x,z)\right\}\right\}. \tag{5.56}$$

In the second set of experiments, we validate the accuracy of solutions for the constrained case. The corresponding results are reported in Table 5.3 showing that $\mathcal{O}(h^2)$ order of accuracy is obtained for both the state and adjoint solutions, while we obtain $\mathcal{O}(h^{3/2})$ order of accuracy for the control function.

Table 5.3: $L^2(\Omega)$ norm of solution error; ν=1e-2. Control-constrained case.

Mesh	$\|y - y_h\|_{L^2(\Omega)}$	Order	$\|p - p_h\|_{L^2(\Omega)}$	Order	$\|u - u_h\|_{L^2(\Omega)}$	Order
17×17	6.48e-3	-	6.49e-5	-	4.92e-2	-
33×33	1.61e-3	2.01	1.63e-5	1.99	1.65e-2	1.58
65×65	4.02e-4	2.00	4.22e-6	1.95	5.67e-3	1.54
129×129	1.01e-4	2.00	1.06e-6	1.99	1.98e-3	1.52

Next, we validate the LFA convergence factor estimate. For this purpose, we compare the estimates in Table 5.1 with observed convergence factors given by asymptotic values of the following ratios of reduction of the L^2-norm of the residuals of the state and adjoint equations between two consecutive (N and $N+1$) multigrid cycles

$$\rho(y) = \lim_N \frac{\left\|r_h(y)^{N+1}\right\|_{L_h^2}}{\left\|r_h(y)^{N}\right\|_{L_h^2}} \quad \text{and} \quad \rho(p) = \lim_N \frac{\left\|r_h(p)^{N+1}\right\|_{L_h^2}}{\left\|r_h(p)^{N}\right\|_{L_h^2}}. \tag{5.57}$$

Next, comparing the theoretical estimates of Table 5.1 with observed convergence factors reported in Table 5.4, we see that the LFA estimates are very accurate by refining the mesh and for smaller values of ν. We investigate the optimization performance of our optimal control problem in the next series of experiments, we first consider an attainable target state of the form

$$y_d(x,z) = \sin(2\pi x)\sin(2\pi z).$$

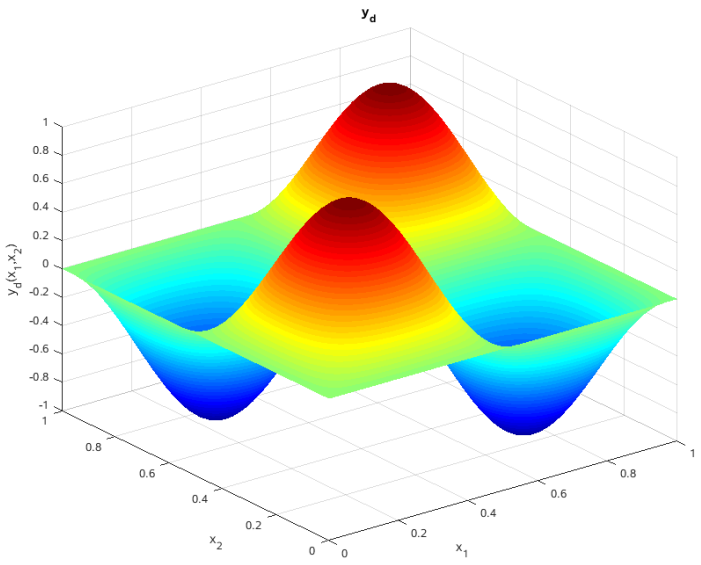


Figure 5.1: Attainable state, y_d

Case 1: Attainable target and no control constraints. In this case, the convergence behaviour of the multigrid procedure is almost independent of the value of the cost ν and of the mesh size. The convergence factors obtained are reported in Table 5.4. Also in this table, we report the tracking errors that become smaller for smaller values of ν, as expected.

Table 5.4: Case 1: observed convergence factors and tracking errors.

Mesh	$\rho(y),\ \rho(p)$	$\|y - y_d\|_{L^2(\Omega)}$
	$\nu = 10^{-4}$	
65×65	0.24, 0.24	2.9944e-1
129×129	0.09, 0.08	4.1321e-1
	$\nu = 10^{-8}$	
65×65	0.03, 0.21	1.0954e-4
129×129	0.03, 0.01	4.4515e-4

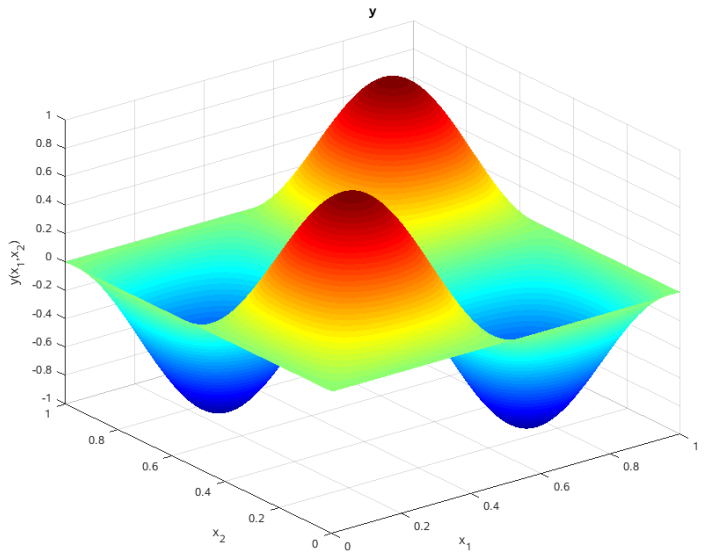


Figure 5.2: Case 1: The state y for $\nu = 10^{-8}$ and 129×129 mesh.

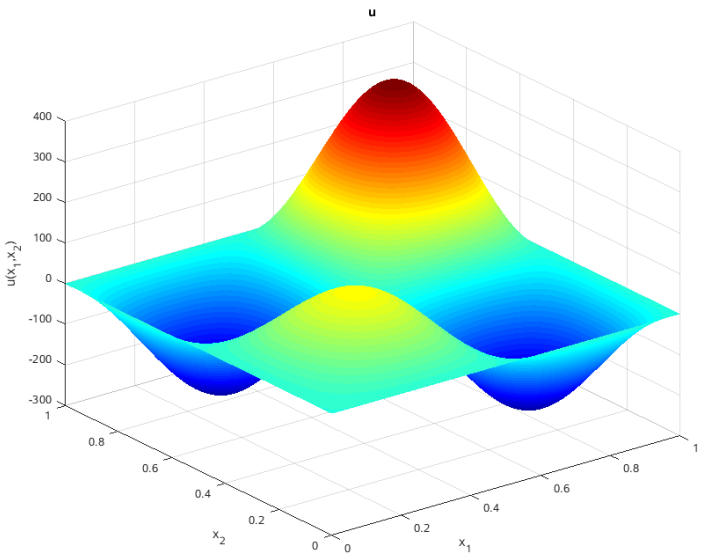


Figure 5.3: Case 1: The control u for $\nu = 10^{-8}$ and 129×129 mesh.

Case 2: Attainable target and control constraints given by $\underline{u} = -10$ and $\overline{u} = 10$. The results obtained for this case are reported in Table 5.5. Notice that similar convergence performance as in Case 1 is obtained, especially by considering finer meshes. In this case, the constraints are active in most of the portions of the domain and thus we cannot obtain a considerable improvement in the tracking error. The Figures 5.4 and 5.5 depict the state and control solutions for $\nu = 10^{-8}$.

Table 5.5: Case 2: observed convergence factors and tracking errors.

Mesh	$\rho(y),\ \rho(p)$	$\|y - y_d\|_{L^2(\Omega)}$
	$\nu = 10^{-4}$	
65×65	0.16, 0.07	4.2845e-1
129×129	0.44, 0.44	4.6431e-1
	$\nu = 10^{-8}$	
65×65	0.13, 0.21	4.2764e-1
129×129	0.04, 0.17	4.6307e-1

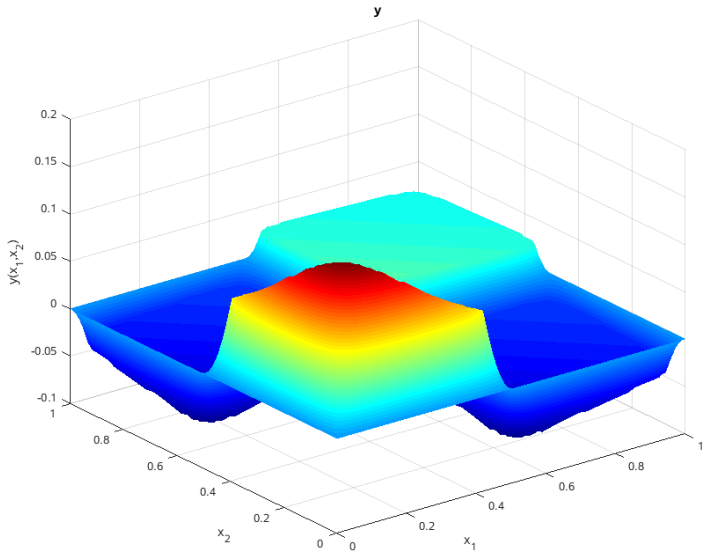


Figure 5.4: Case 2: The state y for $\nu = 10^{-8}$ and 129×129 mesh.

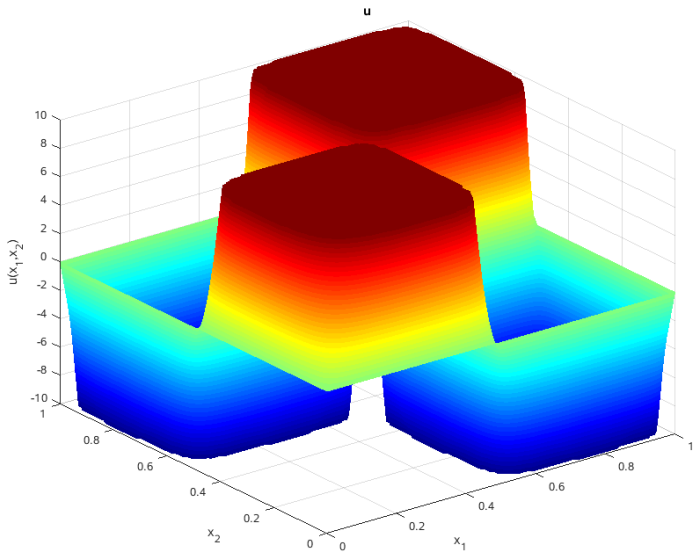


Figure 5.5: Case 2: The control u for $\nu = 10^{-8}$ and 129×129 mesh.

In next experiment, we consider an unattainable target state of the following form

$$y_d(x_1, x_2) = \begin{cases} 2 & \text{on } (0.25, 0.75) \times (0.25, 0.75) \\ 1 & \text{otherwise.} \end{cases}$$

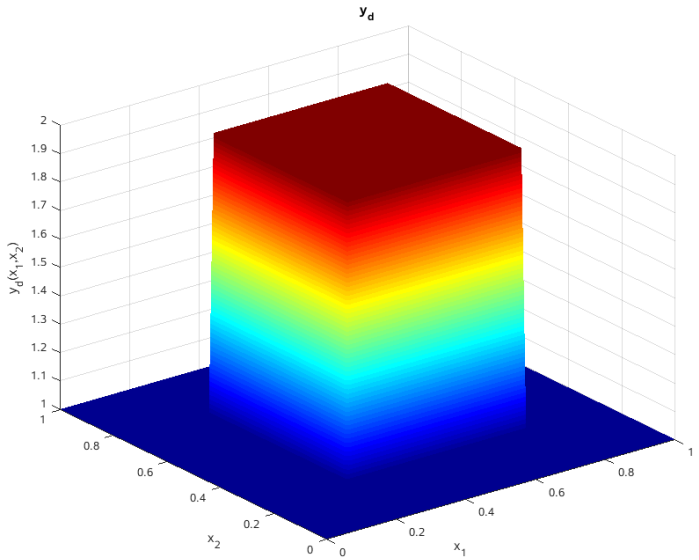


Figure 5.6: Unattainable state, y_d

Case 3: Unattainable target and no control constraints. The results for this case are reported in Table 5.6. The multigrid convergence factors show indepen-

dence on the mesh size but a weak dependence on ν. This is reported in Table 5.6. Changing the cost of the control leads to no change in the tracking norm as the target state is unattainable but provides information on the multigrid convergence factors.

The Figures 5.7 and 5.8 depict the state and control solutions for $\nu = 10^{-8}$. Notice that the presence of convection in the differential operator results in a control function with a slope.

Table 5.6: Case 3: observed convergence factors and tracking errors.

Mesh	$\rho(y),\ \rho(p)$	$\|y - y_d\|_{L^2(\Omega)}$
	$\nu = 10^{-4}$	
65×65	0.25, 0.25	3.8242e-1
129×129	0.13, 0.08	4.8502e-1
	$\nu = 10^{-8}$	
65×65	0.01, 0.03	1.9173e-1
129×129	0.04, 0.01	1.3265e-1

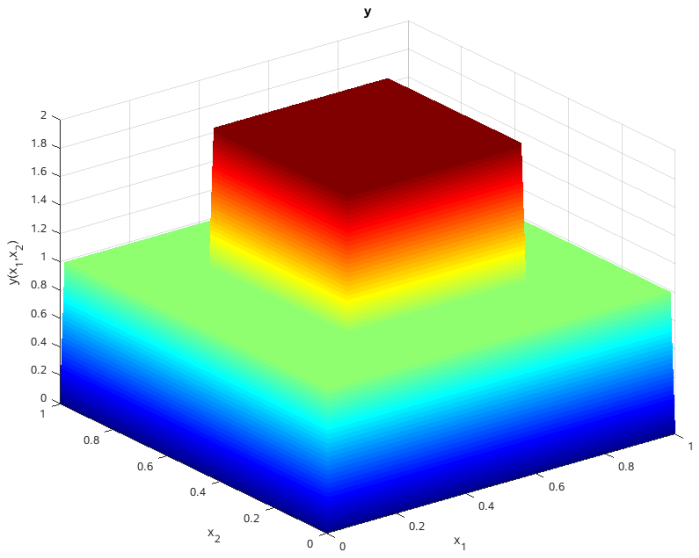


Figure 5.7: Case 3: The state y for $\nu = 10^{-8}$ and 129×129 mesh.

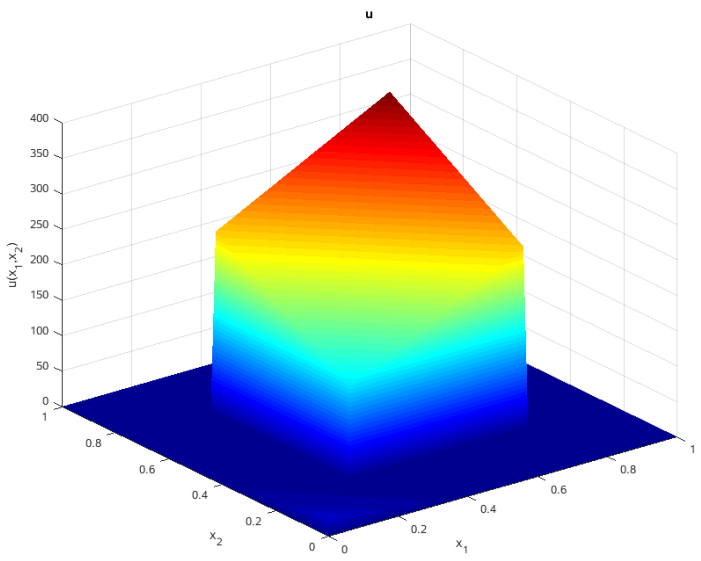


Figure 5.8: Case 3: The control u for $\nu = 10^{-8}$ and 129×129 mesh.

Case 4: Unattainable target and control constraints given by $\underline{u} = -100$ and $\overline{u} = 100$. The results for this case are reported in Table 5.7 and show a similar convergence performance of our multigrid scheme as in Case 3.

Table 5.7: Case 4: observed convergence factors and tracking errors.

Mesh	$\rho(y),\ \rho(p)$	$\|y - y_d\|_{L^2(\Omega)}$
	$\nu = 10^{-4}$	
65×65	0.09, 0.04	3.8242e-1
129×129	0.45, 0.45	4.8502e-1
	$\nu = 10^{-8}$	
65×65	0.01, 0.03	2.3600e-1
129×129	0.03, 0.01	3.3356e-1

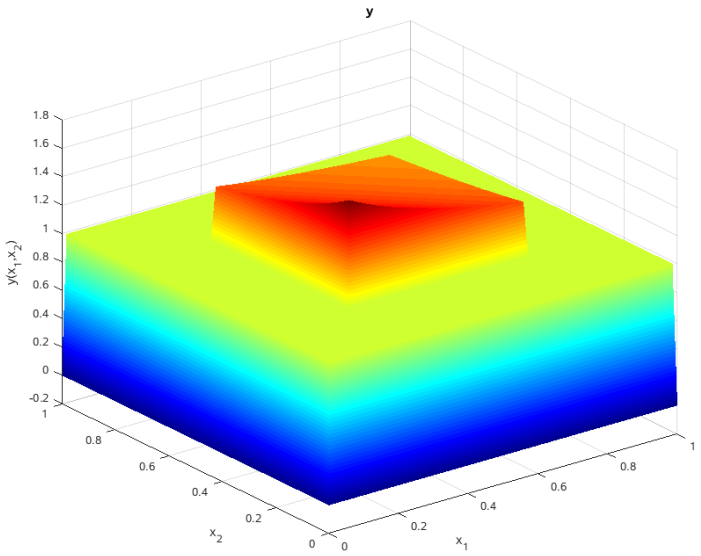


Figure 5.9: Case 4: The state y for $\nu = 10^{-8}$ and 129×129 mesh.

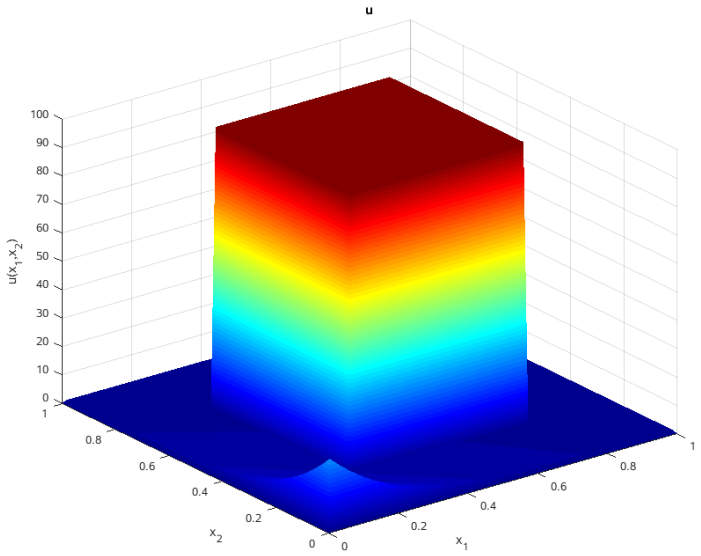


Figure 5.10: Case 4: The control u for $\nu = 10^{-8}$ and 129×129 mesh.

5.6 Summary and remarks

In this chapter, a fast multigrid method for solving an optimal control problem governed by a convection-diffusion partial-integro differential equation was investigated. This method combines a FAS multigrid scheme for elliptic optimality systems with a multigrid fast-integration technique for the efficient evaluation

of the Fredholm integral terms. The optimality system was approximated by the Chang-Cooper finite volume scheme and Simpson quadrature. Existence of optimal controls was proved together with second-order accuracy of the corresponding numerical solution. Numerical experiments for both constrained and unconstrained control problems were discussed to validate the convergence performance of the multigrid scheme. The convergence properties of the proposed multigrid scheme was analyzed by local Fourier analysis and successfully validated by results of numerical experiments.

6. Conclusion

In this thesis, multigrid and hierarchical matrix solution procedures for classes of partial integro-differential problems were discussed. An elliptic partial integro-differential equation, a convection-diffusion partial integro-differential equation and a convection-diffusion partial integro-differential optimality system were considered. In the first part of this work, an efficient multigrid finite-differences scheme for solving an elliptic Fredholm partial integro-differential equation (PIDE) was discussed. This scheme combines a second-order accurate finite difference discretization and a Simpson's quadrature rule to approximate the PIDE problem and a multigrid scheme and a fast multilevel integration method of the Fredholm operator allowing the fast solution of the PIDE problem. Theoretical estimates of second-order accuracy and results of local Fourier analysis of convergence of the multigrid scheme were presented. Results of numerical experiments validated the estimates and demonstrated optimal computational complexity of the framework that included numerical experiments for elliptic PIDE problems with singular kernels. In the second part of this work, a convection-diffusion partial-integro differential equations was considered. The problem was discretized using a finite volume scheme referred to as the Chang and Cooper (CC) scheme and a quadrature rule. Results of stability and accuracy analysis of the CC scheme combined with a Simpson's quadrature rule were presented and second-order accuracy of the numerical solution was proved. The solution strategy to the case of systems of convection-diffusion PIDE where an optimal control problem governed by this model was discussed. In this case, the research focussed on the CC-Simpson's discretization of the optimality system and its solution by the multigrid strategy. Second-order accuracy of the optimization solution was proved and results of local Fourier analysis were presented that provided sharp convergence estimates of the optimal computational complexity of the multigrid-fast integration technique.

In the third part of this work, a hierarchical matrix solution framework was discussed. In this framework, the case of a convection-diffusion PIDE was considered. The CC discretization of the convection-diffusion operator combined

with the trapezoidal quadrature rule was used. Due to the ability of the hierarchical matrix to allow data sparse representation of the fully populated matrix, essential matrix operations were performed with at most logarithmic optimal complexity. In addition, preconditioners to the solution of the PIDE using a generalized minimum residual (GMRes) procedure as a solver were used. Numerical analysis estimates of the accuracy of the finite-volume and trapezoidal rule approximation were presented and combined with estimates of the hierarchical matrix approximation and with the accuracy of the GMRes iterates. Results of numerical experiments were presented that successfully validated the theoretical estimates and the optimal computational complexity of the hierarchical matrix solution procedure. The numerical experiments validated second-order accuracy of the numerical solution. Results of extension of hierarchical matrix method to a three dimension case of the convection-diffusion PIDE and an application to the time evolution of the probability density function of a jump diffusion process were presented. For the three dimensional convection-diffusion PIDE problem and the application to jump-diffusion process, almost linear computational complexity was realized.

A. Appendix

A.1 Codes

Together with this work, we include a CD-ROM that contains the MATLAB® codes implemented for the solution of the classes of PIDE problems considered in this work. We list the name of the routine and its function and give directions on how to run them. We list them in three tranche. The first tranche includes the script used to solve the elliptic PIDE. The second tranche includes the scripts used to solve the convection diffusion PIDE and the PIDE optimal control problem. The last tranche explains the functionality of the AHMED C++ library for the implementation of the Hierarchical matrices framework. In addition, for some of the graphics in this thesis, we have used Inkscape software.

Elliptic PIDE

To run the elliptic PIDE solver, necessary changes are made to choose the desired functions and grid sizes and typing in the MATLAB command window

```
»Test
```

The following list describes the function of each and every script in the folder for elliptic PIDE.

`Test.m`	Main script to run
`EXACT.m`	Solves the elliptic PIDE exactly
`fas_scheme.m`	Multigrid scheme for the elliptic PIDE
`funcf.m`	Right hand side of the state equation
`funcg.m`	Assumed exact solution of the elliptic PIDE
`GAUSS.m`	Gauss-Seidel smoother
`FIXIT.m`	Gauss Picard smoother
`integrate.m`	Integration at all points of the integral term in the elliptic PIDE
`integratev.m`	Integration at certain points of the integral term in the elliptic PIDE

`interpolation.m`	Prolongation routines to interpolate the elliptic PIDE variables
`kernel`	Determines the kernel function to be used
`Keval`	Computes the functional values of the kernel
`LU.m`	Computes the differential operator values of the elliptic PIDE
`prolong.m`	2*nd*-order prolongation of the elliptic PIDE variables
`prolong4.m`	4*th*-order prolongation of the elliptic PIDE variables
`r.m`	Computes the coefficients of the quadrature rule
`residual.m`	Computes the residual of the elliptic PIDE
`restrict4.m`	4*th*-order restriction of the elliptic PIDE variables
`restriction.m`	2*nd*-order restriction of the elliptic PIDE variables
`restriction_adj.m`	2*nd*-order restriction of the elliptic PIDEvariables
`vcycle.m`	For the V-cycle

Convection-diffusion PIDE optimization

To run the convection-diffusion PIDE optimization solver, necessary changes are made to choose the desired functions and grid sizes and typing in the MATLAB command window

```
»Test
```

The following list describes the function of each and every script in the folder for optimal control of the convection diffusion PIDE.

`Test.m`	Main script to run
`B_1_adj.m`	Drift coefficient for the adjoint equation in x direction
`B_2_adj.m`	Drift coefficient for the adjoint equation in z direction
`B_1.m`	Drift coefficient for the state equation in x direction
`B_2.m`	Drift coefficient for the state equation in z direction
`C_1.m`	Diffusion coefficient of the state equation in x direction
`C_2.m`	Diffusion coefficient of the state equation in z direction
`C_1_adj.m`	Diffusion coefficient of the adjoint equation in x direction
`C_2_adj.m`	Diffusion coefficient of the adjoint equation in z direction
`coefficients_adj.m`	Computes the CC coefficients of the adjoint equation
`coefficients.m`	Computes the CC coefficients of the state equation
`CSMA.m`	Collective smoothing multigrid approach routine
`delta_1.m`	CC scheme interpolation for the state equation in x direction
`delta_2.m`	CC scheme interpolation for the state equation in z direction

`delta_1_adj.m`	CC scheme interpolation for the adjoint in the x direction
`delta_2_adj.m`	CC scheme interpolation for the adjoint in the z direction
`EXACT.m`	Solves the state equation exactly
`EXACT_adj.m`	Solves the adjoint equation exactly
`fas_schemeCOMBINED.m`	Multigrid scheme for the optimality system
`funcf.m`	Right hand side of the state equation
`funcg.m`	Assumed exact solution of the state equation
`funcf_adj.m`	Right hand side of the adjoint equation
`funcg_adj.m`	Assumed exact solution of the adjoint equation
`integrate.m`	Integration at all points of $\mathcal{I}y$ in the state equation
`integratev.m`	Integration at certain points of $\mathcal{I}y$ in the state equation
`integrate_adj.m`	Full integration at all points of $\mathcal{I}p$ in the adjoint equation
`integratev_adj.m`	Integration at certain points of $\mathcal{I}p$ in the adjoint equation
`interpolation_adj.m`	Prolongation routines to interpolate the adjoint variables
`interpolation.m`	Prolongation routines to interpolate the state variables
`kernel`	Determines the kernel function to be used
`Keval`	Computes the functional values of the kernel
`LU_adj.m`	Computes the differential values of the adjoint equation
`LU.m`	Computes the differential values of the state equation
`prolong_adj.m`	2*nd*-order prolongation of the adjoint equation variables
`prolong.m`	2*nd*-order prolongation of the state equation variables
`prolong4_adj.m`	4*th*-order prolongation of the adjoint equation variables
`prolong4.m`	4*th*-order prolongation of the state equation variables
`r.m`	Computes the coefficients of the quadrature rule
`residual_adj.m`	Computes the residual of the adjoint equation
`residual.m`	Computes the residual of the state equation
`restrict4_adj.m`	4*th*-order restriction of the adjoint equation variables
`restrict4.m`	4*th*-order restriction of the state equation variables
`restriction.m`	2*nd*-order restriction of the state equation variables
`restriction_adj.m`	2*nd*-order restriction of the adjoint equation variables
`vcycleCOMBINED.m`	For the overall V-cycle

Hierarchical matrices

For the solutions of the hierarchical matrices approximations, we use the Another software library on Hierarchical matrices for Elliptic Differential equations (AHMED), a C++ sofware library. Compiling of AHMED requires the following;

1. C/C++ compiler.

2. BLAS/LAPACK obtained from (`http://www.netlib.org/LAPACK`).
3. METIS, a graph partitioning software obtained from (`http://glaros.dtc.umn.edu/gkhome/metis/metis/overview`).
4. QUARK obtained from (`http://icl.cs.utk.edu/quark/`).
5. MPI (Message Passing Interface) for parallelization in AHMED.

Installation and usage of the library is documented in the `"README.txt"` files in the library. The library is used on permission from Prof. Dr. Mario Bebendorf of Universität Bayreuth, Germany.

A.2 Generalized minimal residual method

The generalized minimal residual (GMRes) method is an iterative method used to compute the numerical solution of non-symmetric system of linear equations. This method relies on the construction of an orthonormal basis of the Kyrlov space. This method uses the Arnoldi's method to generate the orthonormal basis of the Kyrlov subspace. Consider the linear algebraic problem

$$\mathcal{A}y = f,$$

with $\mathcal{A} \in \mathbb{R}^{N\times N}$ is an invertible matrix and $y, f \in \mathbb{R}^N$.

Definition 28. *The Kyrlov subspace of dimension k is defined as*

$$K_k(\mathcal{A}, r_0) = span\{r_0, \mathcal{A}r_0, \ldots, \mathcal{A}^{k-1}r_0\}.$$

If x_0 is the initial guess of the iterative procedure and $r_0 = f - \mathcal{A}x_0$ is the initial residual vector, one has to solve

$$\min_{y\in y_0+K_k(\mathcal{A},r_0)} \|\mathcal{A}y - f\|_2.$$

Due to the properties of power iteration, vectors become linearly dependent and thus the methods relying on the Krylov subspaces involve orthogonalisation schemes, e.g., Lancosz iteration for Hermitian matrices and Arnoldi iteration for more general matrices. This solution is done efficiently if one has an orthonormal basis of $K_k(\mathcal{A}, r_0)$. Let $\{v_1, v_2, \ldots, v_k\}$ be such a basis, one can by placing each basis vector as a column , construct an orthogonal matrix matrix $v_k \in \mathbb{R}^{n\times k}$. Then each vector $y^k \in y_0^0 + K_k(\mathcal{A}, r_0)$ can be written as $y^k = y_0 + v_k z$ for $z \in \mathbb{R}^k$. Hence

$$\|f - \mathcal{A}y\|_2 = \|f - \mathcal{A}\left(y^0 + v_k z\right)\|_2 = \|f - \mathcal{A}y^0 - \mathcal{A}v_k z\|_2.$$

Since $r_0 = f - \mathcal{A}y^0$, we get

$$\min_{y \in y_0 + K_k(\mathcal{A}, r_0)} \|\mathcal{A}y - f\|_2 = \min \|r_0 - \mathcal{A}v_k z\|_2.$$

z is chosen without restrictions from $\mathbb{R}^k$. The GMRes method needs an orthonormal basis of the Krylov space to construct the matrix v_k. In order to build an orthogonal basis $\{v_1, v_2, \ldots, v_k\}$ for $K_k(\mathcal{A}, r_0)$, we use the Arnoldi's method. The construction of this basis and how it is stored is what marks the difference between different methods. We start by summarizing the Arnoldi method in the Algorithm 12.

Algorithm 12 Gram-Schmdt implementation (Arnoldi method)

1. Let $v_1 = \frac{r_0}{\|r_0\|_2}$;
2. **for** $m = 1, 2, \ldots$ **do**
3. Set
$$h_{i,m} = (\mathcal{A}v_m)^T v_i, \; i = 1, 2, \ldots, m;$$
$$\hat{v}_{m+1} = \mathcal{A}v_m - \sum_{i=1}^{m} h_{i,m} v_i;$$
$$h_{m+1,m} = \|\hat{v}_{m+1}\|_2.$$
4. If $h_{m+1,m} = 0$, then stop, otherwise set
$$v_{m+1} = \frac{\hat{v}_{m+1}}{h_{m+1,m}}$$
5. **endfor**.

With this notation we have the following relation

$$\mathcal{A}v_k = v_{k+1}\bar{H}_k,$$

where $\bar{H}_k \in \mathbb{R}^{(k+1)\times k}$ is an upper Hessenberg matrix with one extra row inserted at the bottom which contains one entry.

$$\bar{H}_k = \begin{pmatrix} h_{1,1} & h_{1,2} & \ldots & h_{1,k} \\ h_{2,1} & h_{2,2} & \ldots & h_{2,k} \\ 0 & h_{3,2} & \ldots & h_{3,k} \\ 0 & 0 & \ldots & h_{4,k} \\ \vdots & \vdots & \ddots & \vdots \\ & & & h_{k+1,k} \end{pmatrix} \tag{A.1}$$

(A.1) can then be used and setting $r_0 = \beta v_1$;

$$\|r_0 - \mathcal{A}v_k z\|_2 = \|r_0 - v_{k+1}\bar{H}_k z\|_2 = \|v_{k+1}\left(\beta e_1 - \bar{H}_k z\right)\|_2 \underset{*}{=} \|\beta e_1 - \bar{H}_k z\|_2.$$

where the last equality $(*)$ holds true if v_k is an orthogonal transformation. Thus we only need to compute the solution of the following problem

$$\min_{z\in\mathbb{R}^k} \|\beta e_1 - \bar{H}_k z\|_2. \tag{A.2}$$

The following algorithm details the implementation of the GMRes method .

Algorithm 13 GMRes method using the Gram-Schmidt orthogonalisation

1. Choose y_0 to compute $r_0 = f - \mathcal{A}y_0$ and $v_1 = \frac{r_0}{\|r_0\|_2}$;
2. **for** $j = 1, 2, \ldots$ **do**
   (a) Compute and save $\mathcal{A}v_j$ as to only compute it once;
   (b) Set $h_{i,j} = \langle \mathcal{A}v_j, v_i\rangle$, $i = 1, 2, \ldots, j$;
   (c) Set
   $$\hat{v}_{j+1} = \mathcal{A}v_j - \sum_{i=1}^{j} h_{i,j} v_i;$$
   (d) Set
   $$h_{j+1,j} = \|\hat{v}_{j+1}\|_2;$$
   (e) Set
   $$v_{j+1} = \frac{\hat{v}_{j+1}}{h_{j+1,j}};$$
3. **endfor**.
4. Form an appropriate solution $y_k = y_0 + v_k z_k$, where z_k minimizes (A.2).

The cost of full orthogonalisation of the Krylov subspace $K_k(\mathcal{A}, r_0)$ becomes important when $K_k(\mathcal{A}, r_0)$ reaches a certain size, hence to avoid this disadvantage, the GMRes method is restated or incomplete orthogonalisation is applied. At each iteration of GMRes requires storage of size N more. The number of vectors requiring storage increases. When $\mathcal{A}$ is large, this presents a computational challenge. As a remedy, GMRes is restarted every m iterations with y equal to the new iterate y_m and r_0 equal to $f - \mathcal{A}y_m$. The GMRes iteration will always converge in atmost m iteration steps and this convergence is monotonic since $\|r_{n+1}\| \leq \|r_n\|$. The minimization over a larger subspace allows a smaller residual norm to be achieved and of interest is when the algorithm converges within a specified tolerance in n iterations where $n << m$.

A.3 Compressed Row Storage (CRS)

Compressed Row and Column storage formats are the most general compression strategies of storage of matrices. In application, no assumptions about the sparsity structure of the matrix are made and any zero elements are not stored. In the entire thesis, we use the compressed row storage (CRS). We remark that, despite not storing zero elements, they need an indirect step for addressing every single scale operation in the matrix-vector product or when using pre-conditioners to solve.

The CRS format stores nonzero elements of the matrix rows in contiguous memory locations. Assuming we have a nonsymmetric sparse matrix $\mathcal{M}$, we create 3 vectors: One for floating point numbers val, the other two for the integers $(column\_indices, row\_pointer)$.

The val vector stores the value of the non-zero elements of the matrix $\mathcal{M}$ as they are traversed in the row-wise manner. That is if

$$val(k) = m_{ij} \text{ and } column\_indices(k) = j.$$

The $row\_pointer$ vector stores the locations in the val vector that start a row, that is , if $val(k) = m_{ij}$, then $row\_pointer(i) \leq k < row\_pointer(i+1)$.

By convention, we define $row\_pointer(N+1) = nnz + 1$, where nnz is the number of non-zeros in the matrix $\mathcal{M}$.

The storage savings for this approach is significant. Instead of storing N^2 elements, we need only $2nnz + N + 1$ storage locations.

We include an example to illustrate this approach. Consider a nonsymmetric matrix $\mathcal{M}$ defined by

$$\mathcal{M} = \begin{pmatrix} 10 & 0 & 0 & 0 & -2 & 0 \\ 3 & 9 & 0 & 0 & 0 & 3 \\ 0 & 7 & 8 & 7 & 0 & 0 \\ 3 & 0 & 8 & 7 & 5 & 0 \\ 0 & 8 & 0 & 9 & 9 & 13 \\ 0 & 4 & 0 & 0 & 2 & -1 \end{pmatrix}.$$

The CRS format of the matrix is then specified by the arrays

$$\{val, column\_indices, row\_pointer\},$$

given as follows

val	10	-2	3	9	3	7	8	7	3...9	13	4	2	-1
$column\_pointer$	1	5	1	2	6	2	3	4	1...5	6	2	5	6

row_pointer	1	3	6	9	13	17	20

If the matrix $\mathcal{M}$ is symmetric, we need only to store the upper (or lower) triangular portion of the matrix. The trade-off is a more complicated algorithm with a somewhat different pattern of data access.

Bibliography

[1] N. J. AMSTRONG, K. J. PAINTER, AND J. A. SHERATT, *A continuum approach to modelling cell-cell adhension*, Journal of Theoretical Biology, 243 (2006), pp. 98–113.

[2] M. ANNUNZIATO AND A. BORZÌ, *A Fokker-Planck control framework for multi-dimensional stochastic processes*, Journal of Computational and Applied Mathematics, 237 (2013), pp. 487–507.

[3] D. APPLEBAUM, *Lévy Processes and Stochastic Calculus*, Cambridge Studies in Advanced Mathematics, Cambridge University Press, 2004.

[4] E. AYACHOUR, *A fast implementation for GMRES method*, Journal of Computational and Applied Mathematics, 159 (2003), pp. 269 – 283.

[5] M. BEBENDORF, *Hierarchical LU decomposition-based preconditioners for BEM*, Computing, 74 (2005), pp. 225–247.

[6] M. BEBENDORF, *Approximate inverse preconditioning of finite element discretizations of elliptic operators with nonsmooth coefficients*, SIAM Journal on Matrix Analysis and Applications, 27 (2006), pp. 909–929.

[7] ——, *Why finite element discretizations can be factored by triangular hierarchical matrices*, SIAM Journal on Numerical Analysis, 45 (2007), pp. 1472–1494.

[8] M. BEBENDORF, *Hierarchical Matrices: A Means to Efficiently Solve Elliptic Boundary Value Problems*, Lecture Notes in Computational Science and Engineering, Springer Berlin Heidelberg, 2008.

[9] M. BEBENDORF AND W. HACKBUSCH, *Existence of H-matrix approximants to the inverse FE-matrix of elliptic operators with L^∞ -coefficients*, Numerische Mathematik, 95 (2003), pp. 1–28.

[10] R. BECKER AND B. VEXLER, *Optimal control of the convection-diffusion equation using stabilized finite element methods*, Numerische Mathematik, 106 (2007), pp. 349–367.

[11] S. BÖRM, L. GRASEDYCK, AND W. HACKBUSCH, *Introduction to hierarchical matrices with applications*, Engineering analysis with boundary elements, 27 (2003), pp. 405–422.

[12] S. L. BORNE, *Hierarchical-matrices for convection-diffusion problems with constant convection*, Computing, 70 (2003), pp. 261–274.

[13] S. L. BORNE AND L. GRASEDYCK, *H-matrix preconditioners in convection-dominated problems*, SIAM Journal on Matrix Analysis and Applications, 27 (2006), pp. 1172–1183.

[14] A. BORZÌ AND K. KUNISCH, *A multigrid scheme for elliptic constrained optimal control problems*, Computational Optimization and Applications, 31 (2005), pp. 309–333.

[15] A. BORZÌ, K. KUNISCH, AND D. Y. KWAK, *Accuracy and convergence properties of the finite difference multigrid solution of an optimal control optimality system*, SIAM J. Control Optim., 41 (2002), pp. 1477–1497.

[16] A. BORZÌ AND V. SCHULZ, *Computational Optimization of Systems Governed by Partial Differential Equation*, Society for Industrial and Applied Mathematics, 2012.

[17] A. BRANDT, *Multi-level adaptive solutions to boundary-value problems*, Mathematics of Computation, 31 (1977), pp. 333–390.

[18] ——, *Multilevel computations of integral transforms and particle interactions with oscillatory kernels*, Computer Physics Communications, 65 (1991), pp. 24 – 38.

[19] A. BRANDT AND O. LIVNE, *Multigrid Techniques: 1984 Guide with Applications to Fluid Dynamics, Revised Edition*, Classics in Applied Mathematics, Society for Industrial and Applied Mathematics, 2011.

[20] A. BRANDT AND A. LUBRECHT, *Multilevel matrix multiplication and fast solution of integral equations*, J. Comp. Phys., 90 (1990), pp. 348–370.

[21] W. BRIGGS, V. E. HENSON, AND S. MCCORMICK, *A Multigrid Tutorial*, no. ISBN: 978-0-898714-62-3, Society for Industrial and Applied Mathematics, 2nd edition ed., 1999.

[22] J. S. CHANG AND G. COOPER, *A practical difference scheme for Fokker-Planck equation*, Journal of Computational Physics, 6 (1970), pp. 1–16.

[23] C. CHIARELLA, B. KANG, AND G. MEYER, *The Numerical Solution of the American Option Pricing Problem: Finite Difference and Transform Approaches*, World Scientific Publishing Company, 2014.

[24] E.LIZ AND J. J. NIETO, *Boundary value problems for second order integro-differential equations of Fredholm type*, Journal of Computational and Applied Mathematics, 72 (1996), pp. 215–225.

[25] R. FEDORENKO, *The rate of convergence of an iterative process*, USSR Computational Math. and Math. Physics, 1 (1962), p. 1092.

[26] ——, *A relaxation method for solving elliptic difference equations.*, USSR Computational Math. and Math. Physics, 4 (1962), p. 227.

[27] M. GARRONI AND J. MENALDI, *Second Order Elliptic Integro-Differential Problems*, Chapman & Hall/CRC Research Notes in Mathematics Series, Taylor & Francis, 2002.

[28] D. K. GATHUNGU AND A. BORZÌ, *A multigrid scheme for solving convection-diffusion-integral optimal control problems*, Computing and Visualization in Science, (2017).

[29] ——, *Multigrid solution of an elliptic Fredholm partial integro-differential equation with a Hilbert-Schmidt integral operator*, Applied Mathematics, 8 (2017), pp. 967–986.

[30] B. GAVIRAGHI, M. ANNUNZIATO, AND A. BORZÌ, *Analysis of splitting methods for solving a partial integro-differential Fokker-Planck equation*, Applied Mathematics and Computation, 294 (2017), pp. 1–17.

[31] A. GREENBAUM, *Iterative Methods for Solving Linear Systems*, Society for Industrial and Applied Mathematics, 1997.

[32] L. GREENGARD AND V. ROKHLIN, *A new version of the fast multipole method for the laplace equation in three dimensions*, Acta numerica, 6 (1997), pp. 229–269.

[33] W. HACKBUSCH, *Multi-grid Methods and Applications*, Springer , Berlin, 1985.

[34] W. HACKBUSCH, *The panel clustering algorithm*, in MAFELAP, 1990, pp. 339–348.

[35] W. HACKBUSCH, *Elliptic Differential Equations: Theory and Numerical Treatment*, Computational Mathematics Series, Springer, 1992.

[36] W. HACKBUSCH, *Hierarchical Matrices: Algorithms and Analysis*, Springer Series in Computational Mathematics, Springer Berlin Heidelberg, 2015.

[37] W. HACKBUSCH AND B. KHOROMSKIJ, *A sparse H-matrix arithmetic: general complexity estimates*, Journal of Computational and Applied Mathematics,

125 (2000), pp. 479 – 501. Numerical Analysis 2000. Vol. VI: Ordinary Differential Equations and Integral Equations.

[38] W. HACKBUSCH AND Z. P. NOWAK, *On the fast matrix multiplication in the boundary element method by panel clustering*, Numerische Mathematik, 54 (1989), pp. 463–491.

[39] C. E. HELI, *Real Analysis II*, Georgia Institut of Technology, GA 30332-0160, Atlanta, 2008.

[40] P. HEMKER, *On the order of prolongations and restrictions in multigrid procedures*, Journal of Computational and Applied Mathematics, 32 (1990), pp. 423–429.

[41] M. HINZE, R. PINNAU, M. ULBRICH, AND S. ULBRICH, *Optimization with PDE Constraints*, Mathematical Modelling: Theory and Applications, Springer Netherlands, 2010.

[42] A. HIRSA AND S. N. NEFTCI, *An Introduction to the Mathematics of Financial Derivatives*, Academic Press, 2013.

[43] S. HOSSEINI AND S. SHAHMORAD, *Tau numerical solution of Fredholm integro-differential equations with arbitrary polynimial bases*, Applied Mathematical Modelling, 27 (2003), pp. 145–154.

[44] Q. HU, *Interpolation correction for collocation solutions of Fredholm integro-differential equations*, Mathematics of Computation, 67 (1998), pp. 987–999.

[45] A. ITKIN, *Efficient solution of backward jump-diffusion partial integro-differential equations with splitting and matrix exponentials*, The Journal of Computational Finance, 19 (2016), pp. 29–70.

[46] V. K. JIRSA AND H. HAKEN, *Field theory of electromagnetic brain activity*, Phys. Rev. Lett., 77 (1996), pp. 960–963.

[47] B. S. JOVANOVIĆ AND E. SÜLI, *Analysis of Finite Difference Schemes*, Springer series in Computational Mathematics, 2014.

[48] D. Y. KWAK, *A preconditioned GMRES method*, Applied Mathematics and Computation, 85 (1997), pp. 201 – 208.

[49] S. LE BORNE, L. GRASEDYCK, AND R. KRIEMANN, *Domain-decomposition based H-LU preconditioners*, Domain decomposition methods in science and engineering XVI, 55 (2007), pp. 667–674.

[50] H. LI, L. DI, A. WARE, AND G. YUAN, *The applications of partial integro differential equations related to adaptive wavelet collocation methods for viscosity solutions to jump-diffusion models*, Applied Mathematics and Computation, 246 (2014), pp. 316–335.

[51] J. LIONS, *Optimal Control of Systems Governed by Partial Pifferential Equations*, Grundlehren der mathematischen Wissenschaften, Springer-Verlag, 1971.

[52] T. LUBRECHT AND C. VENNER, *Multi-Level Methods in Lubrication*, Elsevier, 2000.

[53] J. MA, Y. JIANG, AND K. XIANG, *On a moving mesh method for solving partial integro-differential equations*, Journal of Computational Mathematics, 27 (2009), pp. 713–728.

[54] M. MOHAMMADI AND A. BORZÌ, *Analysis of the Chang-Cooper discretization scheme for a class of Fokker-Planck equations*, Journal of Numerical Mathematics, 23 (2015), pp. 271–288.

[55] J. ORTEGA, *Numerical Analysis: A Second Course*, Computer Science and Applied Mathematics, Academic Press, 1972.

[56] S. ROY, M. ANNUNZIATO, AND A. BORZÌ, *A Fokker-Planck feedback control-constrained approach for modelling crowd motion*, Journal of Computational and Theoretical Transport, 45 (2016), pp. 442–458.

[57] Y. SAAD AND M. H. SCHULTZ, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Stat. Comput., 7 (1986), pp. 856–869.

[58] A. K. SAIBABA, S. AMBIKASARAN, J. YUE LI, P. K. KITANIDIS, AND E. F. DARVE, *Application of hierarchical matrices to linear inverse problems in geostatistics*, Oil and Gas Science and Technology-Revue de l'IFP-Institut Francais du Petrole, 67 (2012), p. 857.

[59] A. SOLIMAN, A. EL-ASYED, AND M. EL-AZAB, *On the numerical solution of partial integro-differential equations*, Mathematical Sciences Letters, 1 (2012), pp. 71–80.

[60] J. STOER AND R. BULIRSCH, *Introduction to Numerical Analysis*, Texts in applied mathematics, Springer, New York, 2002.

[61] E. SÜLI AND D. MAYERS, *An Introduction to Numerical Analysis*, Cambridge University Press, 2003.

[62] T. TANG, *A finite difference scheme for partial integro-differential equation with a weakly singular kernel*, Applied Numerical Mathematics, 375 (1993), pp. 309–319.

[63] J. THORWE AND S. BHALEKAR, *Solving partial integro differential equation using laplace tansform method*, American Journal of Computational and Applied Mathematics, 2 (2012), pp. 101–104.

[64] F. TRÖLTZSCH, *Optimal Control of Partial Differential Equations: Theory, Methods, and Applications*, Graduate studies in mathematics, American Mathematical Society, 2010.

[65] U. TROTTENBERG, C. OSTERLEE, AND A. SCHÜLLER, *Multigrid*, Elsevier Academic Press, 2001.

[66] W. VOLK., *The numerical solutions of linear integrodifferential equations by projection methods*, Journal of Integral Equations, 9 (1985), pp. 171–190.

[67] W. VOLK, *The iterated Garlekin method for linear integro-differential equations*, Journal of Computational and Applied Mathematics, 21 (1988), pp. 63–74.

[68] P. WESSELING, *An Introduction to Multigrid Methods*, no. ISBN 0 471 93083 0, John Wiley and Sons, 1992.

[69] R. WIENANDS AND W. JOPPICH, *Practical Fourier Analysis for Multigrid methods*, vol. 4, Chapman and Hall/CRC Press, 2005.

[70] J. ZHAO AND R. M. CORLESS, *Compact finite difference method for integro-differential equations*, Applied Mathematics and Computation, 177 (2006), pp. 271–288.

[71] Z. J. ZHOU AND N. YAN, *A survey of numerical methods for convection-diffusion optimal control problems*, Journal of Numerical Mathematics, 22 (2013), pp. 61–85.