Summer 2014

Adaptive wavelet discretization of tensor products in H-Tucker format

Mazen Ali

ADAPTIVE WAVELET DISCRETIZATION OF TENSOR PRODUCTS IN $\mathcal{H}$-TUCKER FORMAT

by

MAZEN ALI

A THESIS

Presented to the Faculty of the Graduate School of

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE IN APPLIED MATHEMATICS

2014

Approved by

Dr. Martin Bohner, Advisor
Dr. Akim Adekpedjou
Dr. Karsten Urban

ABSTRACT

In the work of [34], the solution to a system of coupled parabolic PDEs, modeling the price of a CDO, was approximated numerically. Due to the nature of the problem, the system involved a large number of equations such that the parameters cannot be stored explicitly. The authors combined the data sparse $\mathcal{H}$-Tucker storage format with the Galerkin method to approximate the solution, using wavelets for the space discretization together with time stepping (*Method of Lines*). The aforementioned approximation is of the linear kind, i.e., using a nonadaptive method. In this work, three methods for solving such systems adaptively are presented, together with a convergence and complexity analysis. The best choice of the method among the three, in general, depends on the particular application. It is shown that (quasi-)optimality is not achieved in the classical sense for adaptive methods, since it, in general, relies on the $\mathcal{H}$-Tucker structure.

ACKNOWLEDGMENTS

I would like to say thank you to my advisor, Dr. Martin Bohner, for the support on my thesis and many helpful remarks. I would also like to thank other committee members, Dr. Akim Adekpedjou and Dr. Karsten Urban. Classes in Numerical Mathematics with Dr. Urban inspired me to pursue this field. Special thanks to Dr. John Singler for the many helpful discussions and suggestions. Dr. Singler's classes in Functional Analysis provided me the necessary background to confidently handle the PDE analytic part of this work. Most of all, I would like to thank my parents for giving me the opportunities I have today and supporting me in all my endeavors.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Page

LIST OF ALGORITHMS

Page

LIST OF ABBREVIATIONS

$(x, y)_H$	Standard inner product on H.
$\#$	Cardinality.
$\mathscr{A}^s$	Set of ℓ_2 sequences for which the best N term approximation converges with rate s.
$\bigoplus$	Direct sum.
$\mathrm{clos}_X(\cdot)$	Topological closure operator in (X, τ), where τ is clear from context.
Δ	Laplacian.
$\delta_{i,j}$	Kronecker delta.
$\ell_0(\Lambda)$	Set of sequences in Λ with finite support.
ℓ_τ^ω	Lorentz spaces.
$\ell_p(\Lambda)$	Set of p-summable sequences on Λ.
$\mathscr{F}_{\mathscr{X}}$	Analysis operator on $\mathscr{X}$.
$\mathscr{F}'_{\mathscr{X}}$	Synthesis operator on $\mathscr{X}$.
$\frac{\partial}{\partial t}$	Time derivative in the distributional sense.
$\lambda_{\max}(\cdot)$	Maximum eigenvalue.
$\lambda_{\min}(\cdot)$	Minimum eigenvalue.
$\mathcal{O}(\cdot)$	Landau Big-O notation.
$\langle \cdot, \cdot \rangle$	Duality pairing.

$\operatorname{span}(\cdot)$	Linear span.
$\mathcal{H}$	Set of tensors with $\mathcal{H}$-Tucker representation.
$\mathcal{L}(X, Y)$	Set of bounded linear operators from $B : X \to Y$.
$\mathcal{T}_{r_1,\dots,r_d}(\cdot)$	Tucker truncation of a tensor.
$\mathcal{X}^{(t)}$	t matricization of a tensor.
$\mathbb{N}$	Set of natural numbers.
∇	Gradient.
$\lvert\lvert\lvert\cdot\rvert\rvert\rvert$	Energy norm.
Ψ	Set of wavelets.
$\mathbb{R}$	Set of real numbers.
$\mathbb{R}^{n\times m}$	Set of matrices $\mathbb{R}^{\{1,\dots,n\}\times\{1,\dots,n\}}$.
$\operatorname{singsupp}(\cdot)$	Singular support of a function.
$\lVert\cdot\rVert$	ℓ_2 norm for sequences. Hilbert–Schmidt operator norm for $B : \ell_2 \to \ell_2$. Standard operator norm otherwise.
$\lVert\cdot\rVert_X$	Standard norm on X.
$\operatorname{supp}(\cdot)$	Support of a function.
$\otimes$	Tensor product.
$\times, \mathsf{X}$	Cartesian product for sets, standard product for numbers.
$\operatorname{tr}[\cdot]$	Trace of a matrix.
$\operatorname{vec}(\cdot)$	Vectorization of a tensor.

$\mathbb{Z}$	Set of integers.
$A \lesssim B$	There exists $C > 0$ independent of A or B directly, s.t. $A \leq CB$.
$A \sim B$	$A \lesssim B$ and $A \gtrsim B$.
$H^1(0,T;X)$	Bochner space.
$H_0^1(\Omega)$	Functions in $H^1(\Omega)$ with zero trace.
H^s	Sobolev space W_2^s.
$L_2(0,T;X)$	Bochner space.
$L_p(\Omega)$	Set of p-integrable measurable functions on Ω.
$s_{\max}$	Best possible approximation rate.
u_t	Time derivative in the distributional sense.
W_p^s	Sobolev space of s-weakly differentiable, p-integrable functions.
X'	Dual of X.
x^T	Transpose.
X^Y	Set of functions $\{f : Y \to X\}$. Naturally identified with the Cartesian product.
a.e.	Almost everywhere.
iff	If and only if.
LHS	Left-Hand-Side.
RHS	Right-Hand-Side.
s.p.d.	Symmetric positive definite.

w.l.o.g.	Without loss of generality.
w.r.t.	With respect to.

1. INTRODUCTION

Mathematical modeling is required in various fields for different types of problems, e.g., in engineering, physics, finance or psychology. There are numerous ways of modeling real-world problems mathematically, one of the most common being the Partial Differential Equation (PDE) approach. For instance, the well known Navier-Stokes equation is frequently used to model problems in fluid dynamics or distributions of static pressure. In physics PDEs can be used to model physical quantities obeying conservation laws, e.g., energy. As the various fields evolve, so do the problems and thus the requirements on the modeling process. More realistic and complex models typically lead to more complicated PDEs that require more sophisticated solution methods. For instance, problems in finance are often distinguished by complex dependency structures and an overwhelming number of equations in a system. Therefore a decent model should consider a large number of equations that lead to high-dimensional PDE problems. This results in the need to store and manage large data tensors, and it is a topic on its own. In the work of [41], the methods stemming from two independent fields were combined to solve a Collateralized Debt Obligation (CDO) modeling problem – namely, PDE methods were combined with methods to manage large tensors.

The issue of storing and managing large amount of data is relevant for various other fields of applied mathematics besides PDEs. Consider a tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ of order d that represents a large number of parameters in, e.g., a CDO pricing problem. One can see that the storage requirements grow exponentially with order d. Normally such tensors would be impossible to store on a computer for a realistic number of parameters. This led researchers to develop data sparse storage formats. In most realistic cases, the exact representation of a tensor in such a format is not

possible. For instance, a rank 1 approximation of a tensor takes the form of

$$\mathrm{vec}(\mathcal{X}) \approx f_1 \otimes \cdots \otimes f_d, \quad f_i \in \mathbb{R}^{n_i}.$$

For $\mathcal{X}$ representing a function $f : [0,1]^d \to \mathbb{R}$ sampled on a finite grid, the representation above implies we approximate f by a separable function $\tilde{f} := f_1 \otimes \ldots \otimes f_d$. An improved version of the rank 1 approximation can be found in, e.g., [6, 29]. This format requires much less storage but, unfortunately, efficient and reliable algorithms for this format remain a subtle problem (see, e.g., [1,24]). In that respect, the problem is less subtle for the Tucker decomposition of the form

$$\mathrm{vec}(\mathcal{X}) \approx (U_1 \otimes \cdots \otimes U_d)\,\mathrm{vec}(\mathcal{C}), \quad U_i \in \mathbb{R}^{n_i \times r_i}$$

with so-called *mode frames* U_i and *core tensor* $\mathcal{C}$. Such an approximation can be achieved by the Higher-Order Singular Value Decomposition (HOSVD) as in [14]. The problem here is, however, that the storage requirement for $\mathcal{C}$ grows exponentially with d. In order to combine the advantages of both of the above methods, i.e., low storage requirements and preferable structure for numerical algorithms, various other decompositions have been developed, such as Tensor Train Decomposition (TT) in [33] or the more general Hierarchical Tucker Decomposition ($\mathcal{H}$-Tucker) in [27, 28]. Since then, various software has been developed to implement the mentioned formats. A good introduction to $\mathcal{H}$-Tucker is provided in the manual for an $\mathcal{H}$-Tucker MATLAB toolbox in [35].

The second component that is required to solve the CDO modeling problem are tools for approximating solutions to PDEs. For most PDE problems, closed-form solutions are not available. This requires the use of numerical methods to solve such problems on a computer. The solution of a PDE is a function that typically depends on a time parameter t and a spacial variable y. In most applications $y \in \Omega \subset \mathbb{R}^n$,

where Ω is a bounded domain and $t \in \mathbb{R}^+$. This means the problem is of infinite nature and cannot be solved directly on a computer. Thus, the usual approach is to approximate the solution u by a solution $\tilde{u}$ of a restated finite problem. A direct approach would be to discretize the space and time domains and compute the solution over a finite set of points. This would lead to the family of Finite Difference (FD) methods. A typically better performing approach is to approximate the solution by a solution to the same problem on a finite-dimensional subspace. This subspace is given in terms of its basis functions, whereas the choice of these basis functions, and consequently the spaces, is essential. This family of methods is referred to as Finite Elements Methods (FEM, see, e.g., [47]). Wavelets posses various preferable analytical and numerical properties that made them very popular in many applications, such as image and signal processing, and numerical solution of PDEs (see, e.g., [48] for more details).

Let $\Psi := \{\psi_\lambda : \lambda \in \Lambda\}$ denote the family of such wavelets, where Λ is an index set. Roughly speaking, each wavelet ψ_λ can be characterized by two parameters: position and level. Each wavelet ψ_λ has finite support and thus represents functions in the spanned space locally, in the region where it is supported, hence the position parameter. This is one of the advantages of wavelet analysis in contrast to Fourier analysis: by local changes of the signal, only a few wavelets have to be modified, whereas in Fourier analysis local changes lead, in general, to global adjustments of the analysis (i.e., adjustment of all coefficients). Each wavelet ψ_λ also represents a level of detail. Given a finite family of wavelets, if one wishes to improve the accuracy of the wavelet representation, wavelets on higher levels of detail have to be added to the system. One of the most important properties of a wavelet system Ψ is that it is a stable basis for L_2. Other important properties include (and are not limited to) good approximation properties and compression, i.e., for many functions f, stemming from approximation problems or PDE problems, a few wavelets often suffice to represent

f with high accuracy. To this day extensive research on wavelets has been performed and numerous constructions have been proposed, both for the univariate case and higher dimensions. The choice of the particular construction typically depends on the desired application. Many of the constructions are rather sophisticated. However, once implemented, these can be utilized as "black box" algorithms. A good survey on wavelets and applications for PDEs can be found in [48] and the various references therein.

Roughly speaking, numerical methods for PDEs can be classified into *nonadaptive* and *adaptive* methods. The former one can be viewed as a specific type of linear approximation, while the latter one is a case of nonlinear approximation (see, e.g., [17]). In nonadaptive methods, the solution to a PDE is approximated by solving the problem on a (linear) finite dimensional subspace based on an a-priori error estimator. In adaptive methods, the problem is typically solved with an initial (small) number of degrees of freedom, then, based on an a-posteriori error estimator, certain elements or wavelets are added to the space in order to reduce the error by some magnitude. This process is repeated until a certain tolerance is reached. Depending on the given problem and choice of basis, there exists a best possible approximation rate for the unknown solutions u. The key question is whether this approximation rate can be achieved by nonadaptive methods, or whether an adaptive method is necessary. For sufficiently smooth solutions, methods based on optimized sparse grids have been developed that achieve this rate (see, e.g., [4, 16, 52]). Hence, adaptive methods truly only make sense for nonsmooth solutions. In the context of approximation of solutions to operator equations, a method is optimal if it produces an approximation to any given accuracy in linear storage and computational complexity. As we will see in the sequel, the choice of wavelets as basis is crucial to ensure the aforementioned optimality.

The remainder of this work is structured as follows: in Section 2, a brief overview over the two main tools of this work is given, namely tensor decompositions and wavelets, which should provide a sensible fundament for the sequel. In Section 3, we introduce some approximation theory and discuss the crucial ingredients for an optimal adaptive method. Section 4 reviews the PDE problem from [41]. In [41], a coupled system of parabolic PDEs with a large number of parameters was solved nonadaptively via Galerkin discretization with time-stepping, utilizing the $\mathcal{H}$-Tucker storage format in the process. The goal of this work is to extend the developed method to an adaptive routine. Since the methods proposed in the sequel maintain a relatively general framework, these will be of particular interest when applied to problems with nonsmooth solutions. Specifically, the goal is to

- verify the compressibility of the operators in Section 5,
- formulate a Richardson type adaptive method for the semi-discrete problem in Subsection 6.4,
- formulate an AWGM type adaptive method for the semi-discrete problem in Subsection 6.5,
- and discuss an adaptive approach for the space-time variational formulation of the PDE in Subsection 6.6.

We develop the general form and analyze some important properties of the suggested methods. Section 7 offers some final conclusions.

2. PRELIMINARIES

In this section we briefly discuss the basics of the $\mathcal{H}$-Tucker storage format and wavelets. Both are independent topics on their own. We require the former one to store large tensors, e.g., tensors containing input parameter values for a PDE. Wavelets have many applications and are particularly useful for approximation purposes, e.g., for approximating solutions of differential equations. Combining both frameworks allows one to handle PDEs arising from problems with a huge number of parameters, such as CDO pricing models.

2.1. HTUCKER

The introduction in this subsection mainly follows [41, Section 5], [35] and [27].

2.1.1. Tucker Decomposition. Most introductions to the $\mathcal{H}$-Tucker format begin with a brief review of the Tucker format since $\mathcal{H}$-Tucker is an extension. For this purpose we require the notion of matricization. To shorten notation, define the d-fold product index set as

$$\mathcal{I} := \mathcal{I}_1 \times \ldots \times \mathcal{I}_d, \quad \mathcal{I}_j := \{1, \ldots, n_j\}, \quad j \in \{1, \ldots, d\}.$$

If $\mathcal{X} \in \mathbb{R}^{\mathcal{I}}$, then the tensor $\mathcal{X}$ is said to have order d. The main problem guiding this subsection is that storing $\mathcal{X}$ directly requires storing $n_1 \cdot \ldots \cdot n_d$ elements, which quickly becomes unmanageable on any modern computer, since the storage requirement grows exponentially in d.

Definition 2.1 (Matricization)**.** Let $\mathcal{X} \in \mathbb{R}^{\mathcal{I}}$. Define $t := \{t_1, \ldots, t_k\} \subset \{1, \ldots, d\}$ and $s := \{s_1, \ldots, s_{d-k}\} = \{1, \ldots, d\} \setminus t$. Define the index set

$$\mathcal{I}^{(t)} := \mathcal{I}_{t_1} \times \ldots \times \mathcal{I}_{t_k} \text{ and } \mathcal{I}^{(s)} := \mathcal{I}_{s_1} \times \ldots \times \mathcal{I}_{s_{d-k}}$$

The t-matricization of $\mathcal{X}$ is defined as

$$\mathcal{X}^{(t)} \in \mathbb{R}^{\mathcal{I}^{(t)} \times \mathcal{I}^{(s)}}, \quad \mathcal{X}^{(t)}_{i_t, i_s} = \mathcal{X}_j$$

where $i_t = (i_{t_1}, \ldots, i_{t_k}) \in \mathcal{I}^{(t)}$, $i_s = (i_{s_1}, \ldots, i_{s_{d-k}}) \in \mathcal{I}^{(s)}$ and $j = (i_1, \ldots, i_d) \in \mathcal{I}$.

Note that the order in which the row and column indices are traversed is important in the Definition 2.1. For instance, in [41], a lexicographical order was assumed, and in [35], a reversed lexicographical order was assumed. If followed consistently, any order can be used. From hereon we will assume a lexicographical order. An important special case of the above definition is the matricization w.r.t. a single dimension $t = \{\mu\}$, which is commonly referred to as μ-mode matricization. This allows us to define the notion of *rank* for a tensor.

But first we require the notion of *vectorization* for a tensor.

Definition 2.2 (Vectorization)**.** Let $\mathcal{X} \in \mathbb{R}^{\mathcal{I}}$. The *vectorization* of $\mathcal{X}$ is defined as

$$\text{vec}(\mathcal{X}) := \mathcal{X}^{(\{1,\ldots,d\})}$$

where the columns of $\mathcal{X}$ are traversed in reversed lexicographical order.

A matrix $X \in \mathbb{R}^{n \times m}$ can be represented as a tensor $\mathcal{X} \in \mathbb{R}^{\mathcal{I}^{(t)} \times \mathcal{I}^{(s)}}$ in an obvious manner, and a tensor $\mathcal{X} \in \mathbb{R}^{\mathcal{I}}$ can be represented as a matrix by taking the t-matricization with $t := \{1, \ldots, k\}$ for some $1 \leq k \leq d$. The matricization is in one-to-one correspondence with the original tensor and, hence, we can always identify the

two with each other. Taking this into consideration, for $\mathcal{X}$ being a matrix, Definition 2.2 simply means we transform the matrix $\mathcal{X}$ into a vector $\mathrm{vec}(\mathcal{X})$ by stacking the columns of $\mathcal{X}$ into one long vector.

Definition 2.3 (Tucker Rank, Tucker Truncation)**.** Let $\mathcal{X} \in \mathbb{R}^{\mathcal{I}}$. The tuple $(r_1, \ldots, r_d)$ with $r_\mu = \mathrm{rank}(\mathcal{X}^{(\mu)})$ is called the *Tucker rank* or *multilinear rank* of $\mathcal{X}$. Furthermore, denote the singular value decomposition of $\mathcal{X}^{(\{\mu\})}$ by

$$\mathcal{X}^{(\{\mu\})} = U_{\{\mu\}} \Sigma_{\{\mu\}} V_{\{\mu\}}$$

for $\mu \in \{1, \ldots, d\}$. The matrices U_μ are the so-called *mode frames*. The tensor $\mathcal{X}$ can be written as

$$\mathrm{vec}(\mathcal{X}) = (U_{\{1\}} \otimes \ldots \otimes U_{\{d\}}) \,\mathrm{vec}(\mathcal{C}),$$

where $\mathcal{C}$ is a *core tensor* with

$$\mathrm{vec}(\mathcal{C}) := (U_{\{1\}}^T \otimes \ldots \otimes U_{\{d\}}^T) \,\mathrm{vec}(\mathcal{X}).$$

Let $\tilde{U}_\mu$ be the restriction of the mode frames to the first $1 \leq \tilde{r}_\mu \leq r_\mu$ columns. The *Tucker truncation* is defined as

$$\begin{aligned} \mathcal{T}_{\tilde{r}_1,\ldots,\tilde{r}_d}(\mathcal{X}) &:= (\tilde{U}_{\{1\}} \otimes \ldots \otimes \tilde{U}_{\{d\}}) \,\mathrm{vec}(\tilde{\mathcal{C}}), \\ \mathrm{vec}(\tilde{\mathcal{C}}) &:= (\tilde{U}_{\{1\}}^T \otimes \ldots \otimes \tilde{U}_{\{d\}}^T) \,\mathrm{vec}(\mathcal{X}). \end{aligned}$$

As the following result shows, the approximation $\mathcal{T}_{\tilde{r}_1,\ldots,\tilde{r}_d}(\mathcal{X})$ is nearly optimal, i.e., can be bounded by an absolute multiple of the best approximation. As in the case of matrices or, more generally, linear operators, the error of the approximation

depends on the decay of the singular values. Note that from hereon, we denote

$$\|\mathcal{X}\| := \|\operatorname{vec}(\mathcal{X})\| := \|\operatorname{vec}(\mathcal{X})\|_2.$$

Lemma 2.1. *Let $\mathcal{X} \in \mathbb{R}^{\mathcal{I}}$. The error of the Tucker truncation can be bounded by*

$$\|\mathcal{X} - \mathcal{T}_{r_1,\dots,r_d}(\mathcal{X})\| \leq \sqrt{\sum_{\mu=1}^{d} \sum_{i=r_\mu+1}^{n_\mu} \sigma_{\mu,i}^2} \leq \sqrt{d} \inf_{\mathcal{Y} \in T(r_1,\dots,r_d)} \|\mathcal{X} - \mathcal{Y}\|$$

where $(\sigma_{\mu,i})_i$ are the μ-mode singular values and

$$T(r_1,\dots,r_d) := \{\operatorname{vec}(\mathcal{X}) = (U_{\{1\}} \otimes \dots \otimes U_{\{d\}}) \operatorname{vec}(\mathcal{C}) : \operatorname{rank}(U_{\{\mu\}}) \leq r_\mu, \\ \mu \in \{1,\dots,d\}\}.$$

Proof. See [15, Property 10]. □

Given the above decomposition, in order to (approximately) represent a tensor $\mathcal{X}$ we need to store the mode frames $U_{\{\mu\}}$ and the core tensor $\mathcal{C}$. However, the storage requirement for $\mathcal{C}$ still grows exponentially in d (see, e.g., [27, Definition 3]).

2.1.2. Hierarchical Tucker Decomposition. The $\mathcal{H}$-Tucker decomposition applies similar ideas as the Tucker decomposition, however, the storage requirement is significantly reduced by applying a hierarchical tree structure to store the tensor. More precisely, the storage format is motivated by the following lemma.

Lemma 2.2. *Let $\mathcal{X} \in \mathbb{R}^{\mathcal{I}}$ and $t = t_l \cup t_r$, with $t_l := \{i_l,\dots,i_m\}$ and $t_r := \{i_{m+1},\dots,i_r\}$. Then*

$$\operatorname{span}(\mathcal{X}^{(t)}) \subset \operatorname{span}(\mathcal{X}^{(t_l)} \otimes \mathcal{X}^{(t_r)})$$

where $\operatorname{span}(X)$ *denotes the space spanned by the columns of X.*

Proof. See [35, Lemma 2.1]. □

Lemma 2.2 implies that, given bases for the column spaces, there exists a matrix B_t such that

$$U_t = (U_{t_l} \otimes U_{t_r})B_t, \quad B_t \in \mathbb{R}^{r_{t_l} r_{t_r} \times r_t}.$$

Starting with $\mathcal{X}^{(\{1,\dots,d\})}$ and applying Lemma 2.2 successively, we obtain a tree structure, where t_l denotes the left child node and t_r the right child node. We require some further notation to define the $\mathcal{H}$-Tucker format.

Definition 2.4 (Dimension Tree)**.** A dimension tree T_d is a tree with root $\{1,\dots,d\}$ such that each node $t \in T_d$ is either a leaf and a singleton $t = \{\mu\}$, $\mu \in \{1,\dots,d\}$ or is the union of two disjoint successors $t = t_l \cup t_r$. The set

$$\mathcal{L}(T_d) := \{t \in T_d : t = \{\mu\},\ \mu \in \{1,\dots,d\}\}$$

is called the set of leaves of T_d, and the set

$$\mathcal{I}(T_d) := T_d \setminus \mathcal{L}(T_d)$$

is called the set of inner nodes.

Definition 2.5 (Hierarchical Rank)**.** Let T_d be a dimension tree. The set $(r_t)_{r_t \in T_d}$ is called the *hierarchical rank* of $\mathcal{X} \in \mathbb{R}^{\mathcal{I}}$ if $r_t = \operatorname{rank}(\mathcal{X}^{(t)})$, for all $t \in T_d$.

Finally, we can define the $\mathcal{H}$-Tucker format as follows.

Definition 2.6 ($\mathcal{H}$-Tucker Format)**.** Let T_d be a dimension tree and $(r_t)_{t \in T_d}$ a family of nonnegative integers. Let $(U_t)_{t \in T_d}$ be a family of matrices such that

$$U_t = (U_{t_l} \otimes U_{t_r})B_t, \quad B_t \in \mathbb{R}^{r_{t_l} r_{t_r} \times r_t}$$

for the inner nodes $t \in \mathcal{I}(T_d)$. The matrices $(U_t)_{t\in T_d}$ are called *mode frames* and $(B_t)_{t\in\mathcal{I}(T_d)}$ *transfer tensors*. Then the collection $\big((U_t)_{t\in\mathcal{L}(T_d)}, (B_t)_{t\in\mathcal{I}(T_d)}\big)$ is called the *hierarchical Tucker representation* of the tensor $U_{\{1,\dots,d\}}$. If $\mathcal{X}$ has such a representation, we write $\mathcal{X} \in \mathcal{H}$.

In the representation above, not all leaf nodes are on the highest level. In order to obtain a closed form representation where all leaves are on the highest level, we can replace the mode frames U_t that are not on the highest level by the Kronecker product $U_t = (U_t \otimes 1)I_t$. An example of such an adjusted storage format for a tensor of order $d = 4$ is illustrated in Figure 2.1.

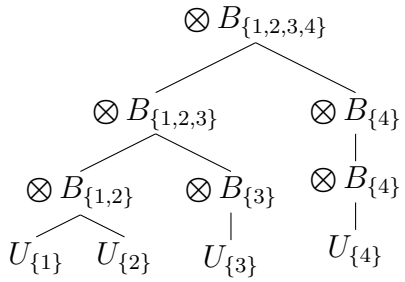

Figure 2.1. Example of an adjusted tree T_d for $d = 4$ (see [41, Figure 5.5]).

In the software, the Kronecker product $U_{t_l} \otimes U_{t_r}$ is never set up explicitly. Instead the following relation is applied to the columns of B_t.

Lemma 2.3. *Let $A \in \mathbb{R}^{m_A\times n_A}$, $B \in \mathbb{R}^{m_B\times n_B}$, $C \in \mathbb{R}^{m_C\times n_C}$ and $X \in \mathbb{R}^{n_B\times n_A}$. Then*

$$(A \otimes B)\,\mathrm{vec}(X) = \mathrm{vec}(C) \Leftrightarrow BXA^T = C.$$

Proof. See, e.g., [49]. □

To avoid repetitive transformation of the columns of the transfer tensors into matrices, the transfer tensors of the form $B_t \in \mathbb{R}^{r_{t_l} r_{t_r}\times r_t}$ are stored in block format

$\mathcal{B}_t \in \mathbb{R}^{r_{t_r} r_t \times r_{t_l}}$ such that

$$\operatorname{vec}((\mathcal{B}_t)_j) = (B_t)_j, \quad (\mathcal{B}_t)_j \in \mathbb{R}^{r_{t_r} \times r_{t_l}},\ j \in \{1, \dots, r_t\}.$$

For $\mathcal{X} \in \mathcal{H}$, the storage requirement is

$$\sum_{t \in \mathcal{I}(T_d)} r_t r_{t_l} r_{t_r} + \sum_{t \in \mathcal{L}(T_d)} n_t r_t \leq (d-1) r^3 + r \sum_{\mu=1}^{d} n_\mu,$$

where $r := \max_{t \in T_d} r_t$. Hence, it does not seem to grow exponentially in d anymore. However, note that, in general, r may vary for different d and therefore it can not be concluded that the rate of growth is linear in d.

Last but not least, in the sequel we require an estimate for the complexity of a matrix-vector product in $\mathcal{H}$-Tucker format which is given by the following lemma.

Lemma 2.4. *Let $\mathcal{I}^1 = m_1 \times \ldots \times m_d$ and $\mathcal{I}^2 = n_1 \times \ldots \times n_d$ be d-fold index sets, $M \in \mathbb{R}^{\mathcal{I}^1 \times \mathcal{I}^2} \cap \mathcal{H}$ a matrix and $X \in \mathbb{R}^{\mathcal{I}^2} \cap \mathcal{H}$ a vector. If both share the same $\mathcal{H}$-Tucker structure, in particular the same dimension tree T_d, then the $\mathcal{H}$-Tucker representation of the matrix-vector product can be computed as*

$$U_t^{MX} = \left[V_t^{(1)} U_t^X, \dots, V_t^{(r_t^M)} U_t^X \right], \quad \forall t \in \mathcal{L}(T_d),$$

where $V_t \in \mathbb{R}^{m_t \times n_t}$ is such that the i-th column of U_t^M satisfies

$$(U_t^M)_i = \operatorname{vec}(V_t^{(i)}).$$

The transfer tensors are computed as

$$\mathcal{B}_t^{MX} = \mathcal{B}_t^M \otimes \mathcal{B}_t^X, \quad \forall t \in \mathcal{I}(T_d),$$

where $\mathcal{B}_t$ is a transfer tensor in block format. The number of arithmetic operations can be bounded by

$$r^M r^X \sum_{k=1}^{d} m_k(2n_k - 1) + (d-1)(r^M)^3(r^X)^3$$

where $r^M := \max_{t \in T_d} r_t^M$ and analogously r^X.

Proof. See [41, Lemma 5.9]. □

We conclude this subsection with some remarks on the practical implementation of the $\mathcal{H}$-Tucker format. For a comprehensive list of currently available operations with the $\mathcal{H}$-Tucker format see [41, Section 5]. Note that if the $\mathcal{H}$-Tucker format of a tensor $\mathcal{X}$ is not given explicitly, i.e., cannot be derived analytically, computing such a representation is a nontrivial task. Hierarchical SVD is often not practical due to the size of the tensor $\mathcal{X}$. Alternatively, one could apply the *Black Box* algorithm. However, this is a heuristic algorithm, since it only uses certain entries of each matricization. For more details see [41, Section 5.4]. Finally, matrix-vector products with $\mathcal{H}$-Tucker tensors lead to an increase in the hierarchical rank as can be seen in Lemma 2.4. This rapidly leads to unmanageable storage sizes and not practical computation times, and thus the resulting tensor has to be further truncated. Such truncations introduce additional errors, the control over which is particularly important in the context of approximation of solutions to PDEs. However, this issue will not be discussed in further detail in this work.

2.2. WAVELETS

In this subsection a brief introduction to wavelets is provided. The introduction mainly follows [41, Section 4]. Wavelets have many applications, the most common being image processing (see, e.g., [37]) or numerical solution of PDEs (see, e.g., [48]). There are many possible wavelet constructions and the best choice depends on the

problem at hand. To mention a few: biorthogonal B-Spline wavelets as in [11], orthonormal wavelets as in [13], interpolatory wavelets as in [20], semi-orthogonal wavelets (prewavelets) as in [7], noncompactly supported wavelets as in [21], frames as in [45], piecewise polynomial multiwavelets as in [22] or fractal multiwavelets, as in [23]. In the work of [41], L_2-orthonormal multiwavelets from [22] are utilized.

2.2.1. Multiresolution Analysis. Generally, the construction of a wavelet basis starts with a *Multiresolution Analysis* (MRA) given a basis of scaling functions. If not mentioned otherwise, such an MRA is usually assumed to be generated by a single scaling function. Otherwise one speaks of *multiscaling* functions and the resulting wavelets are referred to as *multiwavelets*. Using more than one scaling function leads to a more complicated construction but introduces additional flexibility if one, e.g., wishes to obtain orthonormal bases in higher dimensions. Typically one constructs wavelets bases on the interval and uses tensor products to obtain a basis for higher dimensions, as in the work of [41]. Such bases are satisfactory for PDE problems arising in, e.g., finance, where typically exotic and nonsmooth domains are not an issue. However, there are more sophisticated constructions for the multi-dimensional case that allow for more flexible domains, such as in [12]. For the purpose of introduction, we focus on the 1D case since most properties are preserved for the tensor products bases. At the end of this subsection we collect important properties of wavelets that will be required in the sequel. For the most part of this work, we then assume a general wavelet basis for the general Lipschitz domain[1] $\Omega \subset \mathbb{R}^n$ satisfying these properties, unless required otherwise as in Section 5.

Definition 2.7 ((Orthonormal) MRA)**.** A sequence of spaces $(S_j)_{j\in\mathbb{Z}}$ with $S_j \subset L_2(\mathbb{R})$ for all $j \in \mathbb{Z}$ is called a *Multiresolution Analysis* if for any $j \in \mathbb{Z}$

1. $S_j \subset S_{j+1}$ (nestedness).

[1] I.e., open, nonempty, bounded and connected.

2. $\mathrm{clos}_{L_2(\mathbb{R})}(\bigcup_{j\in\mathbb{Z}} S_j) = L_2(\mathbb{R})$ (density).

3. $\bigcap_{j\in\mathbb{Z}} S_j = \{0\}$ (trivial intersection).

4. $f \in S_j \Leftrightarrow f(2\cdot) \in S_{j+1}$ (scaling).

5. $f \in S_j \Leftrightarrow f(\cdot - k) \in S_j$ for any $k \in \mathbb{Z}$ (translation invariance).

6. There are (scaling) functions $\varphi_1, \ldots, \varphi_n$ such that $\Phi_j := \{\varphi^n_{[j,k]} : k \in \mathbb{Z}\}$ is a uniformly stable basis for S_j where the *multiscaling* functions $\varphi^n_{[j,k]}$ are given by

$$\varphi^n_{[j,k]} := 2^{j/2}\varphi_{((k-1)\bmod n)+1}\left(2^j \cdot - \left(\frac{k-1-((k-1)\bmod n)}{n}\right)\right).$$

An MRA is called *orthonormal* if for all $j, k_1, k_2 \in \mathbb{Z}$

$$(\varphi^n_{[j,k_1]}, \varphi^n_{[j,k_2]})_{L_2} = \delta_{k_1,k_2},$$

where δ denotes the Kronecker delta.

Typically one has a coarsest level, say $j_0 = 0$, such that for all $j < j_0$, $S_j = \{0\}$. Since an MRA is nested, we can write for any $j \in \mathbb{Z}$, $S_{j+1} = S_j \bigoplus W_j$, where $W_j \perp S_j$. The spaces W_j are called *wavelet spaces* or *detail spaces* and are generated by the functions $\psi_1, \ldots, \psi_n$ (*mother wavelets*). The *multiwavelets* are then defined as

$$\psi^n_{[j,k]} := 2^{j/2}\psi_{((k-1)\bmod n)+1}\left(2^j \cdot - \left(\frac{k-1-((k-1)\bmod n)}{n}\right)\right),$$

and the system $\Psi_j := \left\{\psi^n_{[j,k]} : k \in \mathbb{Z}\right\}$ is a basis for W_j. Note that by iterating the orthogonal decomposition, we obtain

$$S_{j+1} = \bigoplus_{k=-\infty}^{j} W_k,$$

and hence, the collection of wavelets on all levels is dense in $L_2(\mathbb{R})$ by the properties of the MRA. The system $\Psi := \bigcup_{j\in\mathbb{Z}} \Psi_j$ is thus a basis for $L_2(\mathbb{R})$.

This is only one possible construction using orthogonal complement spaces of S_j. In general, one could choose W_j to be a complement space with a different "angle", so called *biorthogonal wavelets*. For more details see [48, Section 5].

The MRA gives us a basis for $L_2(\mathbb{R})$. To obtain a basis for $L_2(\Omega)$ for a bounded domain $\Omega \subset \mathbb{R}$, unfortunately, we can not simply restrict the functions in Ψ to Ω. This cut-off procedure would, e.g., lead to an unstable basis and would, in general, destroy orthogonality. In general, there are different approaches to solve this problem and the best choice depends on the properties one desires to preserve from the MRA on $\mathbb{R}$. Perhaps the more common strategy is to leave the inner wavelets on the interval Ω unchanged and modify/add wavelets on the boundary. For more details see [48, Section 8]. We briefly summarize a common strategy for constructing an orthonormal multiwavelet basis for a more general domain $\Omega \subset \mathbb{R}^d$.

1. Construct an MRA on $\mathbb{R}$.
2. Orthogonalize to obtain an orthonormal MRA on $\mathbb{R}$.
3. Construct orthonormal multiwavelets on $\mathbb{R}$.
4. Modify the multiwavelet system to obtain an orthonormal stable basis for $[0, 1]$.
5. Construct an orthonormal stable wavelet basis for the reference domain $\hat{\Omega} = [0, 1]^d$ by taking tensor products and properly scaling.
6. Construct a wavelet basis for a parametric image of $\hat{\Omega}$ by taking an appropriate transformation, i.e., $\Omega_i = F_i(\hat{\Omega})$.
7. Construct a wavelet basis for a more general domain $\bar{\Omega} = \bigcup_{i=1}^{N} \bar{\Omega}_i$ by matching.

2.2.2. Properties of Wavelet Bases. From hereon we will often write a system of functions as $\Psi := \{\psi_\lambda : \lambda \in \Lambda\}$, where Λ is some general index set. For instance, in the MRA from Definition 2.7 $\lambda = (j,k)$. Furthermore, to shorten notation we write

$$\begin{aligned} c^T\Psi &:= \sum_{\lambda\in\Lambda} c_\lambda \psi_\lambda, && c \in \ell_2(\Lambda), \\ (\tilde{\Psi},\Psi)_{L_2} &:= ((\psi_\lambda,\psi_\mu))_{\lambda\in\tilde{\Lambda},\mu\in\Lambda}, \\ (f,\Psi)_{L_2} &:= ((f,\psi_\lambda))_{\lambda\in\Lambda}, && f \in L_2(\Omega). \end{aligned}$$

One of the most important properties of wavelet bases is that they give rise to a Riesz basis.

Definition 2.8 (Riesz Basis)**.** Let H be a Hilbert space. A system Ψ is called a *Riesz basis* for H if every element in H has a unique expansion in Ψ and there are constants $0 < c \le C$ such that for $x \in \ell_2(\Lambda)$, we have

$$c\|x\|_{\ell_2} \le \|x^T\Psi\|_H \le C\|x\|_{\ell_2}.$$

The best possible choices for such constants are called *Riesz constants* and are denoted by c_Ψ, C_Ψ.

Obviously, for an orthonormal system, we get the best possible case $c_\Psi = C_\Psi = 1$. The Riesz constants are an indicator for the stability of the basis. The condition number of a Riesz basis is defined as

$$\kappa(\Psi) := \frac{C_\Psi}{c_\Psi} = \frac{\lambda_{\max}(G)}{\lambda_{\min}(G)},$$

where $G := (\Psi,\Psi)_{L_2}$ is the Gramian of Ψ.

As previously mentioned, for higher dimensions, a basis can be constructed by taking tensor products. The following lemma establishes the relationship between the condition of the component basis and the tensor product basis. A detailed discussion on the underlying theory of tensor product spaces can be found in [36].

Lemma 2.5 (Bases of Product Spaces)**.** *Let H_i be separable Hilbert spaces with Riesz bases Ψ_i for $i = 1, \dots, N$. Then the tensor product basis*

$$\Psi := \bigotimes_{i=1}^{N} \Psi_i$$

is a Riesz basis for $\bigotimes_{i=1}^{N} H_i$ with Riesz constants

$$c_\Psi = \prod_{i=1}^{N} c_{\Psi_i}, \quad C_\Psi = \prod_{i=1}^{N} C_{\Psi_i}.$$

Proof. See [30, Proposition 2.23]. □

For a product domain $\Omega = \bigtimes_{i=1}^{N} \Omega_i$, this lemma gives a basis for $L_2(\Omega)$ since from, e.g., [51, Section 1.6], we know that

$$L_2(\Omega) \cong \bigotimes_{i=1}^{N} L_2(\Omega_i).$$

However, for solving differential equations, we are actually interested in bases for Sobolev spaces $H^s(\Omega)$, and these can not be decomposed into a product space. Instead, by, e.g., [18, Lemma 3.1.9], such spaces can be identified with an intersection of product spaces

$$H^s(\Omega) \cong \bigcap_{i=1}^{N} L_2(\Omega_1) \otimes \dots \otimes L_2(\Omega_{i-1}) \otimes H^s(\Omega_i) \otimes L_2(\Omega_{i+1}) \otimes \dots \otimes L_2(\Omega_N).$$

A basis for such spaces is given by the following lemma.

Lemma 2.6 (Bases for Intersection Spaces)**.** *Let H be a separable Hilbert space with Riesz basis Ψ. Let $H_i \subset H$ be subspaces with Riesz bases*

$$\Psi_i := \left\{ \frac{\psi_\lambda}{c_\lambda^i} : \lambda \in \Lambda \right\}$$

for nonzero c_λ^i. Then the system

$$\mathbf{\Psi} := \left\{ \frac{\psi_\lambda}{\sqrt{\sum_{i=1}^N (c_\lambda^i)^2}} : \lambda \in \Lambda \right\}$$

is a Riesz basis for $\bigcap_{i=1}^N H_i$ with Riesz constants

$$c_{\mathbf{\Psi}} = \min_{i=1,\dots,N} c_{\Psi_i}, \quad C_{\mathbf{\Psi}} = \min_{i=1,\dots,N} C_{\Psi_i}.$$

Proof. See [18, Lemma 3.1.8]. □

Lemma 2.5 and Lemma 2.6 tell us how to construct bases for $H^s(\Omega)$. An important observation at this point is that, in general, the condition of the basis is not uniform w.r.t. the dimension of the domain Ω. This has a lot of unpleasant implications. For instance, for nonorthonormal wavelet bases, the condition number of the differential operator as in [19, Section 2] grows exponentially with the space dimension. This is the main reason for the choice of an orthonormal multiwavelet basis, since then $\kappa(\mathbf{\Psi})$ does not depend on the space dimension.

To conclude, we collect and briefly discuss important wavelet properties required in the sequel. From hereon, we assume a given basis Ψ on the domain[2] $\Omega \subset \mathbb{R}^d$ with the following properties (cf. [30, Section 2.2.3] and [18, Section 5.3]). Note that the resulting tensor product wavelet basis preserves the listed properties.

[2]In practice commonly $d = 1$, and a basis for higher dimensions is constructed by taking tensor products as described above.

Assumption 2.1 (Stability)**.** *Ψ is a wavelet basis for $L_2(\Omega)$, i.e., in particular a Riesz basis, where $\Omega \subset \mathbb{R}^d$. Furthermore, with proper scaling Ψ is a Riesz basis for $H^s(\Omega)$, where s should be deduced from the context.*

Assumption 2.2 (Minimal Level)**.** *We assume a given minimal level $j_0 > -\infty$ for the original MRA and thus the resulting system Ψ.*

Assumption 2.3 (Local Support)**.** *Wavelets in Ψ have compact support that decays exponentially with the level, i.e.,*

$$|\operatorname{supp}(\psi_\lambda)| \lesssim 2^{-|\lambda| d}, \quad \forall \lambda \in \Lambda, \tag{2.1}$$

where $|\lambda|$ denotes the (detail) level of the wavelet ψ_λ.

Assumption 2.4 (Finite Number of Overlaps)**.** *The system Ψ is locally finite, i.e.*

$$\#\left\{\mu \in \Lambda : |\mu| = |\lambda|,\ |\operatorname{supp}(\psi_\mu) \cap \operatorname{supp}(\psi_\lambda)| > 0\right\} \lesssim 1$$

uniformly in $|\lambda|$.

Together with Assumption 2.3, Assumption 2.4 implies for $\lambda \in \Lambda$,

$$\#\left\{\mu \in \Lambda : |\operatorname{supp}(\psi_\mu) \cap \operatorname{supp}(\psi_\lambda)| > 0\right\} \lesssim 2^{\max(|\mu|-|\lambda|,0)d}.$$

Briefly this can be seen as follows. Fix ψ_λ and consider $\psi_{\lambda'}$, where λ' only differs from λ in the level, i.e., $\psi_{\lambda'}$ has the same position but is on a different level. If $\psi_{\lambda'}$ is one level finer than ψ_λ, then, by Assumption 2.3, there will be at most 2^d (actually $\lesssim 2^d$) wavelets on the level $|\lambda'|$ that intersect the support of ψ_λ. Hence, assuming Ψ is locally finite, if the number of overlaps on the level $|\lambda|$ is at most C, then between $|\lambda|$ and $|\lambda'|$ it will be at most $C2^d$. For a coarser level, the number of overlaps can only get smaller. Hence, in general, we get the bound $2^{\max(|\mu|-|\lambda|,0)d}$.

Assumption 2.5 (Piecewise Polynomial)**.** *The wavelets are piecewise polynomial of order* $p+2$.

Assumption 2.6 (Singular Support)**.** *For* $\lambda \in \Lambda$, $\ell \in \mathbb{Z}$

$$\#\{\mu \in \Lambda : |\mu| = \ell,\ \operatorname{dist}(\operatorname{singsupp}(\psi_\lambda), \operatorname{supp}(\psi_\mu)) = 0 \\ \text{or } \operatorname{dist}(\operatorname{singsupp}(\psi_\lambda), \operatorname{supp}(\psi_\mu)) = 0\} \lesssim 2^{\max(\ell-|\lambda|,0)(d-1)}, \tag{2.2}$$

where $\operatorname{singsupp}(\cdot)$ *denotes the singular support of a function, i.e., the complement of the largest open set on which the function is smooth.*

The Assumption 2.6 is fulfilled if $\operatorname{singsupp}(\psi_\lambda)$ is overlapped by a finite number of ψ_μ, bounded uniformly in $|\lambda|$ and $|\mu|$. Since $\operatorname{singsupp}(\cdot)$ is $d-1$ dimensional, for, e.g., $d = 1$ this means that any point on the interval Ω is overlapped by a finite number of wavelets ψ_λ, bounded uniformly in $|\lambda|$ (cf. [19, Section 3]).

Assumption 2.7 (Boundary Wavelets)**.** *The number of boundary wavelets, i.e., wavelets whose support has nonempty intersection with* $\partial\Omega$ *is uniformly bounded.*

Assumption 2.8 (Vanishing Moments)**.** *Inner wavelets, i.e., wavelets whose support has empty intersection with the boundary, have* p *vanishing moments*

$$(x^n, \psi_\lambda)_{L_2} = 0, \quad 0 \le n < p.$$

The higher the order of vanishing moments, the better the compression properties of the wavelets. For $f \in W^s_\infty(\operatorname{supp}(\psi_\lambda)) \cap L_2(\Omega)$, in conjunction with a Whitney type estimate, we get from Assumption 2.8 (cf. [48, Proposition 5.9]) that

$$(f, \psi_\lambda)_{L_2} \lesssim 2^{-|\lambda|(\frac{d}{2}+s)} \|f\|_{W^s_\infty},\ s \in [0, p]. \tag{2.3}$$

Assumption 2.9 (Global Smoothness)**.** *The wavelets are globally in* $C^r(\Omega)$*, where* $p \geq r + 2$*.*

Assumption 2.9 also gives (cf. [42, Section 3])

$$\|\psi_\lambda\|_{W_\infty^s} \lesssim 2^{|\lambda|(\frac{d}{2}+s)}, \quad s \in [0, r+1] \tag{2.4}$$

and, if we integrate only over the part of the domain, where ψ_λ is smooth, then we get

$$\|\psi_\lambda\|_{W_\infty^s} \lesssim 2^{|\lambda|(\frac{d}{2}+s)}, \quad s \geq 0. \tag{2.5}$$

There are several wavelet constructions that satisfy Assumptions 2.1-2.9, including the one used in [41]. For convenience we use the term wavelets, even though, strictly speaking, in [41] multiwavelets have been utilized.

3. ADAPTIVE METHODS

In this section, we give a brief overview on adaptive wavelet methods. PDEs or PIDEs are expressed as operator equations. These operator equations are reformulated as equivalent operator equations in sequence spaces and the solution is then nonlinearly approximated. Thus, adaptive wavelet methods can be viewed as a special case of nonlinear approximations as opposed to linear approximations, e.g., Galerkin methods, where the solutions is approximated by a (linear) finite-dimensional subspace of the solution space. This section is mainly based on [46] and the references therein.

3.1. NONLINEAR APPROXIMATION

The brief introduction to approximation theory here is mainly based on [17, Section 2] and [46, Section 1.3]. Consider a separable Hilbert space $\mathscr{X}$ and a basis for $\mathscr{X}$, Ψ. If we seek to approximate a solution $u \in \mathscr{X}$, in linear approximation we consider a finite subspace as

$$\mathscr{X}_N := \operatorname{span}\{\psi_\lambda : \lambda \in \Lambda_N\}, \quad \#\Lambda = N < \infty.$$

The error of the linear approximation by the finite subspace is measured as

$$E_N(u)_{\mathscr{X}} := \inf_{g \in \mathscr{X}_N} \|u - g\|_{\mathscr{X}}.$$

Since a Hilbert space is strictly convex, there exists a unique $u_N \in \mathscr{X}_N$ that minimizes this error. In the context of numerical methods for PDEs, Galerkin methods (nonadaptive) are linear approximations. Typically, the wavelet index sets Λ_N consist of all wavelets up to a certain level ($\sim \log_2(N)$).

Turning to a different approach, consider the *nonlinear* manifold

$$\Sigma_N := \left\{ u_N \in \mathscr{X} : u = \sum_{\lambda \in I} c_\lambda \psi_\lambda,\ \#I \leq N \right\}.$$

The manifold is indeed nonlinear since for $x, y \in \Sigma_N$, we have $x + y \in \Sigma_{2N}$ in general. An approximation to the solution $u \in \mathscr{X}$ from Σ_N can be thus interpreted as the best approximation to u, given a budget of N terms. This is known as the *best N-term approximation* and clearly belongs to the framework of nonlinear approximation. The error is measured as

$$\rho_N(u) := \inf_{g \in \Sigma_N} \|u - g\|_{\mathscr{X}}.$$

The issue of existence of such approximations is discussed in a general setting in [2]. Clearly such approximations are not unique in general. Think of a function with $N+1$ constant wavelet coefficients – taking any N coefficients results in a best N-term approximation. In the context of numerical methods for PDEs, adaptive methods fall into the framework of nonlinear approximations.

If Ψ is a Riesz basis, then we have the norm equivalence

$$\|c^T \Psi\|_{\mathscr{X}} \sim \|c\|_{\ell_2}, \quad c \in \ell_2(\Lambda).$$

Consider approximating the sequence c by c_N with $\mathrm{supp}(c_N) \leq N$. Again, obviously this approximation is not unique in general. However, it is easily seen that taking c_N to be the largest N coefficients in magnitude[3] is a best possible choice in the sense

$$c_N = \operatorname*{arg\,min}_{\mathrm{supp}(g_N) \leq N} \|c - g_N\|_{\ell_2}.$$

[3]Since $c \in \ell_2(\Lambda)$, the sequence is in particular bounded.

Due to the norm equivalence, we have

$$\rho_N(u) \sim \|u - c_N^T \Psi\|_{\mathscr{X}}.$$

Hence, the approximation $\tilde{u}_N := c_N^T \Psi$ is comparable to the best possible choice u_N. Such best N-term approximations converge, in general, with different rates for different types of sequences. It thus makes sense to analyze classes that characterize those rates more closely. But first, we briefly discuss the *best possible approximation rate*.

Consider a nested sequence $(\Lambda_i)_{i=0}^{\infty}$ of sets with a finite number of degrees of freedom such that

$$\Lambda_0 \subset \Lambda_1 \subset \ldots, \quad \bigcup_{i=0}^{\infty} \Lambda_i = \Lambda.$$

In adaptive methods, the enlargement of a Λ_i to Λ_{i+1} is based on a posteriori error estimator, which is the main difference to nonadaptive methods, where the sequence is fixed from the beginning based on a priori error estimator. For wavelets, the index set Λ_i typically includes all wavelets up to level i. Associated with the space $\mathscr{X}$ and the basis Ψ, there exists a best possible approximation rate $s_{\max}$, i.e., a parameter $s_{\max}$ such that for all sufficiently smooth $u \in \mathscr{X}$

$$\|u - u_{\Lambda_i}\| \lesssim (\#\Lambda_i)^{-s_{\max}},$$

and this rate $s_{\max}$ cannot be improved by additional smoothness assumptions or a different choice of $(\Lambda_i)_i$. A fundamental question is whether this rate of convergence can be achieved by a nonadaptive method or whether an adaptive method is required. In fact, there are possible choices for a finite Λ such that this is achieved (e.g., *optimized sparse grid spaces*). However, the main disadvantage of such nonadaptive methods in contrast to adaptive methods is the high regularity assumptions

on the solution. Hence, adaptive methods truly make sense only for problems with nonsmooth[4] solutions. For more details see, e.g., [30, Section 3.1.2].

Remark 3.1 ([46, Remark 1.1]). Note that there are $u \in \mathscr{X}$ for which any desirable rate can be realized. Think of u having finite representations or being exceptionally close to such functions. An appropriate choice of $(\Lambda_i)_i$ would yield any desirable rate. Those exceptional cases are not taken into consideration (cf. [46, Section 1.1]).

Example 3.1 ([46, Example 1.1]). Let $\mathscr{X} = H^m(\Omega)$, $\Omega \subset \mathbb{R}^d$ be a bounded domain and Ψ a wavelet basis of order $p > m$. Then

$$s_{\max} = \frac{p-m}{d}.$$

For $s \in (0, s_{\max}]$ and $u \in H^{sd+m}(\Omega)$ the rate s is realized. In particular, for $s = s_{\max}$, we require $u \in H^p(\Omega)$. The nested sequence $(\Lambda_i)_i$ is simply chosen as the set of wavelets of level up to i. The result is sharp in the sense that for any $\varepsilon > 0$, there is no choice of $(\Lambda_i)_i$ such that s is realized for all $u \in H^{sn+m-\varepsilon}(\Omega)$. Note that the best possible approximation rate is inversely proportional to the space dimension, i.e., for higher dimensions, the approximation rate deteriorates. This is sometimes referred to as *the curse of dimensionality*.

Example 3.2 ([46, Section 7.2]). Let Ω_i be a domain of dimension d_i and Ψ_i a wavelet basis of order $p_i \geq m$. It is known that a sufficiently smooth function on $\Omega := \Omega_1 \times \ldots \times \Omega_N$ can be approximated in $H^m(\Omega)$ by sparse grid approximations with the best possible rate being

$$s_{\max} = \max_i \frac{p_i - m}{d_i}.$$

[4]Here nonsmooth is meant in the measure of Sobolev spaces. Adaptive methods deal with solutions that are nonsmooth in this measure but smooth in the measure of Besov spaces.

Thus, thinking of $d_1 = \ldots = d_N = 1$ and $p_1 = \ldots = p_N =: p > m$, the aforementioned curse of dimensionality is completely removed in this case. See [4, 16, 52] for details on sparse grid approximations and, e.g., [30, Section 8], [5] for applications.

As previously mentioned, we would like to further investigate the appropriate class of sequences. The class of coefficient sequences for which the best N-term approximation converges with rate $s > 0$ is defined as

$$\mathscr{A}^s := \left\{ c \in \ell_2(\Lambda) : \|c\|_{\mathscr{A}^s} := \sup_{\varepsilon > 0} \varepsilon \cdot [\min\{N \in \mathbb{N}_0 : \|c - c_N\|_{\ell_2} \leq \varepsilon\}^s < \infty \right\}.$$

It is easily verified that for $c \in \mathscr{A}^s$, the smallest N such that $\|c - c_N\|_{\ell_2} \leq \varepsilon$ satisfies

$$N \leq \varepsilon^{-1/s} \|c\|_{\mathscr{A}^s}^{1/s},$$

and this bound is sharp. This bound justifies the following definition of (quasi-) optimality for adaptive methods. From hereon, we use $\ell_0(\Lambda)$ to denote sequences in Λ with finite support.

Definition 3.1 (Quasi-Optimal)**.** An adaptive wavelet method is called *(quasi-)optimal* if for $u = c^T \Psi \in \mathscr{X}$ with $c \in \mathscr{A}^s$, $s \in (0, s_{\max}]$ and a given tolerance $\varepsilon > 0$, the method produces an approximation $v \in \ell_0$ with

$$\|c - v\|_{\ell_2} \leq \varepsilon,$$

and

$$\#\operatorname{supp}(v) \lesssim \varepsilon^{-1/s} \|v\|_{\mathscr{A}^s}^{1/s},$$

at the cost of $\mathcal{O}(\varepsilon^{-1/s} \|v\|_{\mathscr{A}^s}^{1/s})$ arithmetic operations.

Roughly speaking, a (quasi-)optimal method produces a solution for a prescribed tolerance $\varepsilon > 0$, with computational and storage complexity being linear in the output size, i.e., in the number of nonzero coefficients of the approximation v. For $s > s_{\max}$ the class $\mathscr{A}^s$ is, although not empty, not interesting, since it corresponds to the exceptionally "nice" case discussed in Remark 3.1.

3.2. LINEAR OPERATOR EQUATIONS

In this subsection, we consider general operator equations as we can apply the theory to PDEs. The discussion follows mainly [46, Section 2.1]. Note that for notational simplicity, we drop the index set Λ when it is clear from context.

Let $\mathscr{X}$ and $\mathscr{Y}$ be Hilbert spaces and let $\Psi^{\mathscr{X}}$, $\Psi^{\mathscr{Y}}$ denote the respective bases. Define the *analysis operator* as

$$\mathscr{F}_{\mathscr{X}} : \mathscr{X}' \to \ell_2(\Lambda_{\mathscr{X}}), \quad g \mapsto [g(\psi_\lambda)]_{\lambda \in \Lambda_{\mathscr{X}}}$$

and analogously for $\mathscr{Y}$. We assume the bases for both $\mathscr{X}$ and $\mathscr{Y}$ are Riesz bases which is equivalent to assuming the analysis operator is boundedly invertible. Since $\mathscr{F}_{\mathscr{X}}$ is bounded, we can define the adjoint as

$$\mathscr{F}'_{\mathscr{X}} : (\ell_2(\Lambda_{\mathscr{X}}))' \to \mathscr{X}'', \quad (\mathscr{F}'_{\mathscr{X}} l)(g) = l(\mathscr{F}_{\mathscr{X}}(g)), \quad l \in \ell_2'',\ g \in \mathscr{X}'.$$

Since ℓ_2 is a Hilbert space, by the Riesz Representation Theorem (RRT), it is isometrically isomorph to its dual. Identify $l \in \ell_2'$ with $c \in \ell_2$. Then by RRT, we can write

$$(\mathscr{F}'_{\mathscr{X}} l)(g) = (\mathscr{F}_{\mathscr{X}}(g), c)_{\ell_2} = \sum_{\lambda \in \Lambda_{\mathscr{X}}} c_\lambda g(\psi_\lambda). \tag{3.1}$$

Furthermore, since $\mathscr{X}$ is a Hilbert space, it is reflexive. Fix $g \in \mathscr{X}'$ and let $C : \mathscr{X} \to \mathscr{X}''$ denote the canonical mapping. Identifying g with $x_0 = \sum_{\lambda \in \Lambda_{\mathscr{X}}} d_\lambda \psi_\lambda$, we can write

$$(\mathscr{F}'_{\mathscr{X}} l)(g) = C_{x_0}(g) = g(x_0) = \sum_{\lambda \in \Lambda_{\mathscr{X}}} d_\lambda g(\psi_\lambda),$$

where the last equality follows by linearity and boundedness of g. Comparing with (3.1), since $\Psi^{\mathscr{X}}$ is a basis, we infer

$$x_0 = c^T \Psi^{\mathscr{X}}.$$

Hence, we can identify the adjoint $\mathscr{F}'_{\mathscr{X}}$ with $\mathscr{F}^*_{\mathscr{X}}$ defined as

$$\mathscr{F}^*_{\mathscr{X}} : \ell_2 \to \mathscr{X}, \quad c \mapsto c^T \Psi^{\mathscr{X}}$$

and similarly for $\mathscr{F}_{\mathscr{Y}}$ where both adjoints are boundedly invertible as well. The operator $\mathscr{F}^*_{\mathscr{X}}$ is called the *synthesis operator* and is commonly defined as the adjoint of $\mathscr{F}_{\mathscr{X}}$ (cf. [46, Section 2.1]), although strictly speaking it can only be identified with the adjoint as described above.

We are interested in solving the operator equation

$$Bu = f, \quad B \in \mathcal{L}(\mathscr{X}, \mathscr{Y}'), \quad u \in \mathscr{X}, \quad f \in \mathscr{Y}',$$

where B is boundedly invertible. This problem is equivalent to

$$\boldsymbol{B}\boldsymbol{u} = \boldsymbol{f},$$
$$\boldsymbol{B} := \mathscr{F}_{\mathscr{Y}} B \mathscr{F}^*_{\mathscr{X}} = [(B\psi^{\mathscr{X}}_\mu)(\psi^{\mathscr{Y}}_\lambda)]_{\lambda \in \Lambda_{\mathscr{Y}}, \mu \in \Lambda_{\mathscr{X}}} \in \mathcal{L}(\ell_2(\Lambda_{\mathscr{X}}), \ell_2(\Lambda_{\mathscr{Y}})),$$

$$\boldsymbol{f} := \mathscr{F}_{\mathscr{Y}} f = [f(\psi_\lambda^{\mathscr{Y}})]_{\lambda\in\Lambda_{\mathscr{Y}}} \in \ell_2(\Lambda_{\mathscr{Y}}),$$
$$u = \boldsymbol{u}^T \Psi^{\mathscr{X}},$$

where, since B is boundedly invertible together with the analysis, synthesis operators and their adjoints, $\boldsymbol{B}$ is boundedly invertible as well. Moreover, we get the useful correspondence

$$(\boldsymbol{B}\boldsymbol{v}, \boldsymbol{w})_{\ell_2} = (Bv)(w)$$

for any $\boldsymbol{v},\ \boldsymbol{w} \in \ell_2$, where $v = \boldsymbol{v}^T\Psi^{\mathscr{X}}$ and $w = \boldsymbol{w}^T\Psi^{\mathscr{Y}}$. The Riesz constants are equal to

$$C_{\Psi^{\mathscr{X}}} = \|\mathscr{F}_{\mathscr{X}}\| = \sup_{\|g\|_{\mathscr{X}}=1} \|\mathscr{F}_{\mathscr{X}} g\|_{\ell_2},$$
$$c_{\Psi^{\mathscr{X}}} = \|(\mathscr{F}_{\mathscr{X}})^{-1}\|^{-1} = \inf_{\|g\|_{\mathscr{X}}=1} \|\mathscr{F}_{\mathscr{X}} g\|_{\ell_2}$$

and similarly for $\mathscr{Y}$. By definition of $\boldsymbol{B}$, we infer $\|\boldsymbol{B}\| \le \|B\| C_{\Psi^{\mathscr{X}}} C_{\Psi^{\mathscr{Y}}}$, $\|\boldsymbol{B}^{-1}\| \le \|B^{-1}\|/(c_{\Psi^{\mathscr{X}}} c_{\Psi^{\mathscr{Y}}})$. This in particular shows that the condition number $\kappa(\boldsymbol{B})$ depends on the operator B itself and on the condition of the Riesz bases for $\mathscr{X}$ and $\mathscr{Y}$, which again justifies the choice of a uniformly well conditioned basis, e.g., orthonormal, as mentioned in Subsection 2.2.2.

Example 3.3. As a model example consider the Poisson equation on a bounded domain $\Omega \subset \mathbb{R}^d$

$$\begin{cases} -\Delta u = f & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega. \end{cases}$$

Multiplying by a test function v and integrating by parts, we get the weak form

$$\int_\Omega \nabla u \cdot \nabla v = \int_\Omega fv, \quad \forall v \in H_0^1(\Omega)$$

for some $f \in L_2(\Omega)$. The test and trial spaces are $\mathscr{X} = \mathscr{Y} = H_0^1(\Omega)$. The associated operator equation reads as

$$Bu = f, \quad B \in \mathcal{L}(\mathscr{X}, \mathscr{X}'), \quad (Bu)(v) := \int_\Omega \nabla u \cdot \nabla v,$$
$$f \in X', \quad f(v) := \int_\Omega fv.$$

The operator B is boundedly invertible. The equivalent ℓ_2 problem reads

$$\boldsymbol{Bu} = \boldsymbol{f}, \quad \boldsymbol{B} := \left[\int_\Omega \nabla\psi_\mu \cdot \nabla\psi_\lambda\right]_{\lambda,\mu\in\Lambda}, \quad \boldsymbol{f} = \left[\int_\Omega f\psi_\lambda\right]_{\lambda\in\Lambda}, \quad u = \boldsymbol{u}^T\Psi.$$

3.3. APPLY

According to Definition 3.1, in order to obtain a (quasi-)optimal routine, we require a method that, for a given $\boldsymbol{u} \in \mathscr{A}^s$, produces an approximation to any desired tolerance in linear complexity. Note that the system $\boldsymbol{Bu} = \boldsymbol{f}$ involves a bi-infinite matrix and infinite vectors. Hence, it should be clear that producing an approximation to any desired accuracy is not possible for an arbitrary $\boldsymbol{B}$. Intuitively, since we only know how to deal with finite systems, we have to approximate the operator $\boldsymbol{B}$ by a finite matrix and we have to be able to do so for any desired accuracy. This means that, when truncating $\boldsymbol{B}$ in an appropriate manner, the error should approach 0 as we increase the size of the finite approximation. In other words, $\boldsymbol{B}$ has to be of a certain kind that allows this compression. In fact, we will see that the compression properties are usually guaranteed by the underlying wavelet basis, as wavelets have preferable compression properties that are well known from image processing.

More precisely, we require the availability of a routine for approximate operator application that was coined by [9, Section 6.4] as **APPLY**. Since the publication of the original paper, several modifications of **APPLY** have appeared and thus, even though we briefly mention one of them, in the sequel any **APPLY** routine satisfying the following properties can be utilized. From hereon, we often use the shorthand notation $\|\cdot\|$ without a subscript if the space is clear from the context.

Definition 3.2 ([46, Definition 3.1])**.** For $\bar{s} > 0$, the operator $\boldsymbol{B}$ is called $\bar{s}$-admissible if for any $\varepsilon > 0$, we have an approximate matrix-vector routine

$$\mathbf{APPLY}[\boldsymbol{B}, \boldsymbol{w}, \varepsilon] \to \boldsymbol{w}_\varepsilon$$

such that, for $\boldsymbol{w} \in \ell_0$, $\boldsymbol{w}_\varepsilon \in \ell_0$ and $\|\boldsymbol{B}\boldsymbol{w} - \boldsymbol{w}_\varepsilon\| \leq \varepsilon$, where for any $s \in (0, \bar{s}]$

$$\#\operatorname{supp}(\boldsymbol{w}_\varepsilon) \lesssim \varepsilon^{-1/s} \|\boldsymbol{w}\|_{\mathscr{A}^s}^{1/s}$$

and the number of operations is of the order

$$\mathcal{O}(\varepsilon^{-1/s} \|\boldsymbol{w}\|_{\mathscr{A}^s}^{1/s} + \#\operatorname{supp}(\boldsymbol{w}) + 1).$$

In fact, in order to guarantee optimality, comparing with Definition 3.1, we require $\bar{s} \geq s_{\max}$ (see [46, Section 3.2] for details). As shown in [9, Proposition 3.8], such an $\bar{s}$-admissible operator defines a bounded linear mapping on the approximation class $\mathscr{A}^s$ (cf. also [46, Proposition 3.1]). Intuitively, it is not surprising that $\boldsymbol{B}$ preserves the approximation class, given $\boldsymbol{B}$ itself can be approximated to any desired accuracy.

Proposition 3.1. *Let $\boldsymbol{B}$ be $\bar{s}$-admissible. Then for $s \in (0, \bar{s}]$, $\boldsymbol{B} : \mathscr{A}^s \to \mathscr{A}^s$ is bounded and $\|\boldsymbol{w}_\varepsilon\|_{\mathscr{A}^s} \lesssim \|\boldsymbol{w}\|_{\mathscr{A}^s}$ holds uniformly in ε.*

Proof. The proof can be found in [46, Proposition 3.1]. □

In the work of [9], a specific class of operators was investigated that allowed to define a valid **APPLY** routine. If an operator $\boldsymbol{B}$ belongs to this class, then it can be approximated to any desired accuracy by sparse matrices. Note that the approximation property ensures one part of Definition 3.2, while the "sparse" part ensures that this approximation can actually be performed in linear complexity.

Definition 3.3 ([9, Definition 3.6])**.** An operator $\boldsymbol{B} : \ell_2 \to \ell_2$ is called s^*-compressible if for any $s \in (0, s^*)$, there exists $(\alpha_j)_j, (\beta_j)_j \in \ell_1(\mathbb{N})$ such that the sequence of operators $(\boldsymbol{B}_j)_j$ satisfies

$$\|\boldsymbol{B} - \boldsymbol{B}_j\| \leq \alpha_j 2^{-sj},$$

where $\boldsymbol{B}_j$ is derived from $\boldsymbol{B}$ by replacing all but in the order of $\mathcal{O}(\beta_j 2^j)$ entries by 0.

In practice, an entry of $\boldsymbol{B}$ is not given explicitly but has to be approximated numerically, e.g., by a quadrature rule. Hence, it should be clear that the question whether $\boldsymbol{B}$ is $\bar{s}$-admissible also depends on the computability of the individual entries which motivates the following definition.

Definition 3.4 (Computable)**.** The operator $\boldsymbol{B}$ is s^*-computable if it is s^*-compressible and each entry can be computed in $\mathcal{O}(1)$ operations or, more generally, if each column of $\boldsymbol{B}_j$ can be computed in $\mathcal{O}(2^j)$ operations.

A useful lemma that is often used to show s^*-compressibility and that will be used in the sequel several times is the well known *Schur Lemma* stated below. Typically, the entries of the (preconditioned) operator $\boldsymbol{B}$ have strong decay properties due to the underlying wavelet basis. Choosing an appropriate dropping criteria, one obtains a sparse operator $\boldsymbol{B}_j$ and uses the Schur Lemma to show that the norm of the remainder of the bi-infinite matrix is bounded and approaches 0 as we increase j.

Then one investigates a suitable quadrature rule for the entries such that each entry can be computed in $\mathcal{O}(1)$ operations while preserving the error on the same level. This finally gives s^*-computability.

Lemma 3.1 (Schur Lemma)**.** *Let $\boldsymbol{M} = (m_{\lambda,\mu})_{\lambda,\mu\in\Lambda}$ be an operator such that there exist positive weights $(\omega_\lambda)_{\lambda\in\Lambda}$ and constants $0 < c_r, c_c < \infty$ with*

$$\sum_{\mu\in\Lambda} \omega_\mu |m_{\lambda,\mu}| \leq c_r \omega_\lambda, \quad \forall \lambda \in \Lambda,$$
$$\sum_{\lambda\in\Lambda} \omega_\lambda |m_{\lambda,\mu}| \leq c_c \omega_\mu, \quad \forall \mu \in \Lambda.$$

Then $\|\boldsymbol{M}\|_{\ell_2} \leq \sqrt{c_r c_c}$.

Proof. The proof can be found in a slightly different form in [38, Lemma 8.4]. □

To show that $\boldsymbol{B}$ is $\bar{s}$-admissible, we require an **APPLY** routine satisfying Definition 3.2. Such a routine was presented first in [9], and later an optimized version was proposed in [19]. In fact, for those routines to be valid, all we require is that the operator is s^*-computable and, hence, this readily implies that s^*-computable operators are $\bar{s}$-admissible for any $\bar{s} < s^*$. The idea for an approximate matrix vector routine in [9] is to find an optimal balance between accuracy and computational effort. This is achieved by applying good approximations of the operator $\boldsymbol{B}$ to rough approximations of the vector and rough approximations of $\boldsymbol{B}$ to good approximation of the vector. Suppose we want to approximate $\boldsymbol{B}\boldsymbol{w}$, where $\#\operatorname{supp}(\boldsymbol{w}) = N < \infty$. Let $\boldsymbol{w}_j$ denote the best 2^j-term approximation to $\boldsymbol{w}$. Then compute *bins* as

$$\boldsymbol{w}_0,$$
$$\boldsymbol{w}_j - \boldsymbol{w}_{j-1}, \quad j = 1, \ldots, \lfloor \log_2(N) \rfloor,$$
$$\boldsymbol{w}_j = \boldsymbol{w}, \quad j > \log_2(N).$$

Finally, compute the approximation as

$$\boldsymbol{B}\boldsymbol{w} \approx \boldsymbol{w}^k := \boldsymbol{B}_k \boldsymbol{w}_0 + \sum_{j=0}^{k-1} \boldsymbol{B}_j (\boldsymbol{w}_{k-j} - \boldsymbol{w}_{k-j-1}).$$

This approximation lies in the heart of the **APPLY** routine proposed in [9]. Bigger choices of the parameter k lead to more accurate approximations and more computational effort at the same time. Thus, obviously k will depend on the prescribed tolerance $\varepsilon > 0$. Before we can state an important result concerning the convergence of this routine and possible choices for k, we require the following definition.

Definition 3.5 (Lorentz Spaces, cf. [48, Definition 7.8])**.** Given $\boldsymbol{v} \in \ell_2$, define its *decreasing rearrangement* $\boldsymbol{v}^*$ as follows: for $n \geq 1$, let $\boldsymbol{v}_n^*$ be the n-th largest element of $\boldsymbol{v}$ in magnitude and $\boldsymbol{v}^* := (\boldsymbol{v}_n^*)_{n=1}^{\infty}$. Then for each $0 < \tau < 2$ define

$$|\boldsymbol{v}|_{\ell_\tau^w} := \sup_{n \geq 1} n^{1/\tau} \boldsymbol{v}_n^*$$

and the weak ℓ_τ^w space as

$$\ell_\tau^w := \left\{ \boldsymbol{v} \in \ell_2 : |\boldsymbol{v}|_{\ell_\tau^w} < \infty \right\}.$$

The corresponding norm is defined as

$$\|\boldsymbol{v}\|_{\ell_\tau^w} := |\boldsymbol{v}|_{\ell_\tau^w} + \|\boldsymbol{v}\|_{\ell_2}.$$

Lorentz spaces[5] characterize the decay of coefficients in a sequence. It is thus not surprising that they are closely related to the class $\mathscr{A}^s$, as the following result shows.

[5]The above definition is only a special kind of Lorentz spaces.

Proposition 3.2. *For $s > 0$ define*

$$\tau := \left(s + \frac{1}{2}\right)^{-1}.$$

Then we have

(i) $\boldsymbol{v} \in \mathscr{A}^s$ iff $\boldsymbol{v} \in \ell^w_\tau$. The norm equivalence

$$\|\boldsymbol{v}\|_{\mathscr{A}^s} \sim \|\boldsymbol{v}\|_{\ell^w_\tau}$$

holds with constants depending only on τ for $\tau \searrow 0$.

(ii) If $\boldsymbol{v} \in \ell^w_\tau$, then the error of the best N-term approximation can be bounded as

$$\rho_N(\boldsymbol{v}) \lesssim N^{-s}\|\boldsymbol{v}\|_{\ell^w_\tau}$$

with a constant depending only on τ for $\tau \searrow 0$.

Proof. See [48, Proposition 7.11] and the references therein. □

Finally, we are ready to state the result from [9] on the convergence of **APPLY**.

Proposition 3.3. *Let $\boldsymbol{B}$ be s^*-compressible. For $\boldsymbol{w}^k$, the error estimate*

$$\|\boldsymbol{B}\boldsymbol{w} - \boldsymbol{w}^k\| \lesssim 2^{-ks}\|\boldsymbol{w}\|_{\ell^w_\tau}$$

holds, where $s < s^*$ and

$$s = \frac{1}{\tau} - \frac{1}{2}$$

with a constant depending only on τ for $\tau \searrow 0$.

Proof. The proof can be found in [48, Proposition 7.13]. □

The question remains as to the choice of $k = k(\varepsilon)$. Taking a closer look at the approximation $\boldsymbol{w}^k$, we observe

$$\begin{aligned}\|\boldsymbol{B}\boldsymbol{w} - \boldsymbol{w}^k\| &\leq \|\boldsymbol{B}\|\|\boldsymbol{w} - \boldsymbol{w}_k\| + \|\boldsymbol{B} - \boldsymbol{B}_k\|\|\boldsymbol{w}_0\| \\ &\quad + \sum_{j=0}^{k-1} \|\boldsymbol{B} - \boldsymbol{B}_j\|\|\boldsymbol{w}_{k-j} - \boldsymbol{w}_{k-j-1}\| \\ &\leq \|\boldsymbol{B}\|\|\boldsymbol{w} - \boldsymbol{w}_k\| + \alpha_k 2^{-ks}\|\boldsymbol{w}_0\| + \sum_{j=0}^{k-1} \alpha_j 2^{-js}\|\boldsymbol{w}_{k-j} - \boldsymbol{w}_{k-j-1}\| \\ &=: E_k.\end{aligned}$$

Hence, if we can estimate $\|\boldsymbol{B}\|$ and $(\alpha_j)_j$, then we can ensure $E_k \leq \varepsilon$. More generally, to have an implementable version of **APPLY**, we require knowledge of the sequences $(e_j)_{j\in\mathbb{N}_0}$, $(c_j)_{j\in\mathbb{N}_0}$, where

$$\|\boldsymbol{B} - \boldsymbol{B}_j\| \leq e_j, \quad \|\boldsymbol{B}\| \leq e_0$$

and c_j is the number of nonzero entries in each column of $\boldsymbol{B}_j$, $c_0 = 0$. A more efficient modification of the original **APPLY** routine discussed above was proposed by [19]. Summing up, we get the following result.

Theorem 3.1. *If $\boldsymbol{B}$ is s^*-computable, then it is also $\bar{s}$-admissible for any $\bar{s} < s^*$.*

Proof. The proof follows from the discussion above. For more details see [46, Theorem 3.2]. □

3.4. RICHARDSON ITERATIONS

With the tools introduced above, we are ready to provide a sensible introduction to an adaptive routine for solving operator equations. First we discuss Richardson iterations from [10] and then the Adaptive Wavelet Galerkin Method from [9] in

the next subsection, since both are required in the sequel. The introduction mainly follows [46, Sections 3 and 4].

The Richardson iteration is a fixed point type scheme: we assume that there exists $\alpha \in \mathbb{R}$ such that

$$\|I - \alpha \boldsymbol{B}\| < 1,$$

and, hence, the iteration of the form

$$\boldsymbol{u}^{i+1} = \boldsymbol{u}^i + \alpha(\boldsymbol{f} - \boldsymbol{B}\boldsymbol{u}^i)$$

converges linearly by the Banach Fixed Point Theorem. Under certain conditions, the existence of such a dampening parameter α is guaranteed. We consider the nonsymmetric case for the Lemma 3.2 as the PDE from [41] is nonsymmetric.

Lemma 3.2. *Let $\boldsymbol{B} \in \mathcal{L}(\ell_2, \ell_2)$ and $\boldsymbol{B}_S := \frac{1}{2}(\boldsymbol{B} + \boldsymbol{B}^T)$ be positive definite and boundedly invertible. Then for $\alpha \in (0, 1/(\|\boldsymbol{B}_S\| + \|\boldsymbol{B}_S^{-1}\|^{-1})]$ with $\alpha < 2/(\|\boldsymbol{B}_S^{-1}\|\|\boldsymbol{B}\|)$, we have*

$$\|I - \alpha \boldsymbol{B}\| \leq \sqrt{1 - 2\alpha\|\boldsymbol{B}_S^{-1}\|^{-1} + \alpha^2\|\boldsymbol{B}\|^2} < 1.$$

Proof. The proof can be found in [46, Lemma 3.2]. □

Hence, if we can estimate $\|\boldsymbol{B}\|$, $\|\boldsymbol{B}^{-1}\|$ and from thereon $\|\boldsymbol{B}_S\|$ and $\|\boldsymbol{B}_S^{-1}\|$, then we obtain a linearly convergent iterative scheme for solving $\boldsymbol{B}\boldsymbol{u} = \boldsymbol{f}$. However, this scheme is still not implementable, since $\boldsymbol{f}$, $\boldsymbol{u}$ are generally infinite vectors and $\boldsymbol{B}$ is a bi-infinite matrix. Thus, we require further approximations. Approximations for $\boldsymbol{B}\boldsymbol{u}$ were discussed in the previous subsection. As for $\boldsymbol{f}$, the approximation generally depends on the specific RHS $\boldsymbol{f}$ at hand. In a more general setting, we will assume

the availability of a routine for the RHS with the following properties. For specific approximations of the RHS see, e.g., [19, Section 5] or [30, Section 4.6.3].

Assumption 3.1. *The routine* $\mathbf{RHS}[\boldsymbol{f}, \varepsilon] \to \boldsymbol{f}_\varepsilon$ *satisfies*

$$\|\boldsymbol{f} - \boldsymbol{f}_\varepsilon\| \leq \varepsilon, \quad \#\operatorname{supp}(\boldsymbol{f}_\varepsilon) \lesssim \min\{N : \|\boldsymbol{f} - \boldsymbol{f}_N\| \leq \varepsilon\}$$

at the cost of $\mathcal{O}(\#\operatorname{supp}(\boldsymbol{f}_\varepsilon) + 1)$ *arithmetic operations.*

At first sight, the Assumption 3.1 does not fit directly into the definition of (quasi-)optimality as given in Definition 3.1, which is the ultimate goal for an adaptive wavelet algorithm. However, as the following statement shows, it is in fact sufficient, given an appropriate $\boldsymbol{B}$.

Proposition 3.4. *Let* $\boldsymbol{B}$ *be* $\bar{s}$*-admissible and* $\boldsymbol{u} \in \mathscr{A}^s$ *for* $s \in (0, \bar{s}]$*. The* $\boldsymbol{f}_\varepsilon$ *satisfies* $\#\operatorname{supp}(\boldsymbol{f}_\varepsilon) \lesssim \varepsilon^{-1/s}\|\boldsymbol{u}\|_{\mathscr{A}^s}^{1/s}$*, and the number of operations is of the order*

$$\varepsilon^{-1/s}\|\boldsymbol{u}\|_{\mathscr{A}^s}^{1/s} + 1.$$

Proof. The proof can be found in [46, Corollary 3.1]. □

Technically, we could already formulate a first Richardson iteration. However, numerical experiments showed that such a routine is not optimal (see, e.g., [48, Section 7.6.3]) in the sense that it does not converge with a rate comparable to the best N-term approximation. The iterations produce a lot of negligible coefficients that do not contribute significantly to the accuracy of the approximation but substantially increase computational effort. A remedy would be to apply coarsening of the iterands to restore the balance between accuracy and computational effort, while keeping the error on the same level. A specific implementation of such a routine can be found in, e.g., [46, Section 3.3]. For the sequel, we assume the availability of the following.

Assumption 3.2. *The routine* **COARSE**$[\boldsymbol{w}, \varepsilon] \to \boldsymbol{w}_\varepsilon$ *for* $\boldsymbol{w} \in \ell_0$ *satisfies*

$$\|\boldsymbol{w} - \boldsymbol{w}_\varepsilon\| \le \varepsilon, \quad \#\operatorname{supp}(\boldsymbol{w}_\varepsilon) \lesssim \min\{N : \|\boldsymbol{w} - \boldsymbol{w}_N\| \le \varepsilon\}$$

with the number of operations in the order of

$$\mathcal{O}(\#\operatorname{supp}(\boldsymbol{w}_\varepsilon) + \max(\varepsilon^{-1}\|\boldsymbol{w}\|, 1)).$$

Although **COARSE** introduces an additional error, the following statement shows that it preserves the error on the same level while maintaining a number of nonzero terms comparable to the best N-term approximation and guarantees a uniform bound on the $\|\cdot\|_{\mathscr{A}^s}$-norm (which is not the case without **COARSE**).

Proposition 3.5. *Let* $\zeta > 1$ *and* $s > 0$. *Then for any* $\varepsilon > 0$, $\boldsymbol{v} \in \mathscr{A}^s$ *and* $\boldsymbol{w} \in \ell_0$ *with*

$$\|\boldsymbol{v} - \boldsymbol{w}\| \le \varepsilon$$

for $\boldsymbol{w}_{\zeta\varepsilon} := \mathbf{COARSE}[\boldsymbol{w}, \zeta\varepsilon]$ *we have*

$$\#\operatorname{supp}(\boldsymbol{w}_{\zeta\varepsilon}) \lesssim \varepsilon^{-1/s}\|\boldsymbol{v}\|_{\mathscr{A}^s}^{1/s}, \quad \|\boldsymbol{w}_{\zeta\varepsilon}\|_{\mathscr{A}^s} \lesssim \|\boldsymbol{v}\|_{\mathscr{A}^s}, \quad \|\boldsymbol{v} - \boldsymbol{w}_{\zeta\varepsilon}\| \le (1+\zeta)\varepsilon.$$

Proof. The proof can be found in [46, Proposition 3.2] or in more detail in [8, Theorem 4.9.1]. □

Finally, an implementable version of the Richardson iteration is formulated in Algorithm 3.1 (cf. [46, Section 3]).

Note that this routine can be further improved by an inner iteration as in [48, Algorithm 7.3]. Moreover, one can think of applying more advanced iterative schemes,

Algorithm 3.1 RICH

Input: $\varepsilon > 0$, $\varepsilon_0 \geq \|\boldsymbol{u}\|$, $\theta \leq \frac{1}{2}$, $K \in \mathbb{N}$ and $\rho < 1$ s.t. $\|I - \alpha \boldsymbol{B}\| \leq \rho$, $2\rho^K < \theta$
Output: $\boldsymbol{u}_\varepsilon$ with $\|\boldsymbol{u}_\varepsilon - \boldsymbol{u}\| \leq \varepsilon$

```
i := 0, u^0 := 0
while ε_i > ε do
    i := i + 1
    ε_i := 2ρ^K ε_{i−1}/θ
    y^{i,0} := u^{i−1}
    for j = 1 to K do
        δ := ρ^j ε_{i−1}/2αK
        y^{i,j} := y^{i,j−1} + α (RHS[f; δ] − APPLY[B; y^{i,j−1}; δ])
    end for
    u^i := COARSE[(1 − θ)ε_i, y^{i,K}]
end while
return u_ε := u^i
```

such as *Krylov subspace* methods. For Richardson iterations the following result gives (quasi-)optimality.

Theorem 3.2. *The output of* **RICH** *satisfies* $\|\boldsymbol{u} - \boldsymbol{u}_\varepsilon\| \leq \varepsilon$. *If* $\boldsymbol{B}$ *is* $\bar{s}$*-admissible with* $\bar{s} \geq s_{\max}$ *and* $\varepsilon < \varepsilon_0 \lesssim \|\boldsymbol{u}\|$, *then* **RICH** *is (quasi-)optimal.*

Proof. The proof can be found in [46, Theorem 3.1]. □

3.5. ADAPTIVE WAVELET GALERKIN METHOD

In alternative to the iterative scheme, one can consider using an *Adaptive Wavelet Galerkin Method* (AWGM) from [9] to approximate the solution of an operator equation. In contrast to **RICH**, AWGM does not require coarsening of the iterands and thus numerically outperforms the iterative Richardson procedure (see, e.g., [26]). However, unlike the more general Richardson iterations, AWGM is limited to the elliptic case. The idea behind AWGM is very similar to the Adaptive Finite Elements Method: there the solution is projected to a finite subspace Λ_i representing a finite mesh, then an error estimator is used to mark elements for refinement and

the equation is solved on the refined mesh $\Lambda_{i+1} \supset \Lambda_i$. The loop terminates when a certain criterion is fulfilled, e.g., when the residual is small enough. As is common in the literature, AWGM is introduced in an idealized setting first and then converted to an implementable routine. The introduction mainly follows [46, Section 4].

Crucial for the convergence analysis of AWGM is the availability of an equivalent energy norm. For this purpose, we assume $\boldsymbol{B}$ is s.p.d. and is boundedly invertible. Note that if $\boldsymbol{B}$ is not symmetric and positive definite, then we can apply the scheme to the normal equation $\boldsymbol{B}^T\boldsymbol{B}\boldsymbol{u} = \boldsymbol{B}^T\boldsymbol{f}$. In fact, a closer inspection of the convergence analysis reveals that symmetry of $\boldsymbol{B}$ is not crucial. If $\boldsymbol{B}$ is coercive but not symmetric, then the induced bilinear form is not an inner product anymore and the associated (pseudo-)norm is not a norm, since only the relaxed version $\|x+y\| \lesssim \|x\| + \|y\|$ of the triangle inequality holds. However, the convergence analysis does not depend on this and is still valid in the setting. We will come back to the norm equivalences in Subsection 6.5.

Define on $\ell_2(\Lambda)$

$$|||\cdot||| := \sqrt{(\boldsymbol{B}\cdot,\cdot)_{\ell_2}}.$$

Define restrictions of operators to subsets $\mathcal{J} \subset \Lambda$ where, in practice, $\mathcal{J}$ will be finite. Define $\ell_2(\mathcal{J}) \subset \ell_2(\Lambda)$ as the set of those vectors in $\ell_2(\Lambda)$, that have supports in $\mathcal{J}$. Let $E_{\mathcal{J}}$ denote the trivial embedding of $\ell_2(\mathcal{J})$ into $\ell_2(\Lambda)$ and $R_{\mathcal{J}}$ its Hilbert adjoint. Hence, $E_{\mathcal{J}}$ extends by filling entries in positions $j \in \Lambda\setminus\mathcal{J}$ with zeros, and $R_{\mathcal{J}}$ restricts by dropping entries outside $\mathcal{J}$. Define the restricted operator onto $\ell_2(\mathcal{J})$ as

$$\boldsymbol{B}_{\mathcal{J}} := R_{\mathcal{J}}\boldsymbol{B}E_{\mathcal{J}}.$$

The solution to $\boldsymbol{B}_{\Lambda_i}\boldsymbol{u}_{\Lambda_i} = R_{\Lambda_i}\boldsymbol{f}$ is known as the *Galerkin approximation* and is the best approximation w.r.t. $|||\cdot|||$. The idea of AWGM is to compute the Galerkin

solution on (a finite) Λ_i, then enlarge Λ_i to Λ_{i+1} such that the solution on Λ_{i+1} satisfies $|||\boldsymbol{u} - \boldsymbol{u}_{\Lambda_{i+1}}||| \leq \rho |||\boldsymbol{u} - \boldsymbol{u}_{\Lambda_i}|||$ for some $\rho < 1$. Furthermore, this enlargement should be minimal. The following result shows that, if the enlargement captures the bulk of the residuum, also referred to as the *bulk criterion*, then the AWGM converges.

Lemma 3.3. *Let $\mu \in (0, 1]$, $\boldsymbol{w} \in \ell_2(\Lambda_i)$ and $\Lambda_i \subset \Lambda_{i+1} \subset \Lambda$ such that*

$$\|R_{\Lambda_{i+1}}(\boldsymbol{f} - \boldsymbol{B}\boldsymbol{w})\| \geq \mu \|\boldsymbol{f} - \boldsymbol{B}\boldsymbol{w}\|. \tag{3.2}$$

Then for the Galerkin solution $\boldsymbol{u}_{\Lambda_{i+1}}$, we have

$$|||\boldsymbol{u} - \boldsymbol{u}_{\Lambda_{i+1}}||| \leq \sqrt{1 - \beta^2} |||\boldsymbol{u} - \boldsymbol{w}|||$$

where

$$\beta := \mu \cdot \kappa(\boldsymbol{B})^{-\frac{1}{2}}.$$

Proof. The proof can be found in [46, Lemma 4.1]. □

Taking μ sufficiently small, e.g., $0 < \mu < \kappa(\boldsymbol{B})^{-1/2}$, we get $\beta < \kappa(\boldsymbol{B})^{-1} \leq 1$ and thus a convergent scheme. Moreover, this particular choice of μ ensures the cardinality of Λ_{i+1} is controlled, as the following result shows.

Lemma 3.4. *For μ as above and for the minimal Λ_{i+1} satisfying* (3.2)*, we have*

$$\#(\Lambda_{i+1} \setminus \Lambda_i) \leq \min \left\{ N : |||\boldsymbol{u} - \boldsymbol{u}_N||| \leq \sqrt{1 - \bar{\beta}^2} |||\boldsymbol{u} - \boldsymbol{w}||| \right\},$$

where

$$\bar{\beta} := \mu \cdot \kappa(\boldsymbol{B})^{\frac{1}{2}} < 1.$$

Proof. The proof can be found in [46, Lemma 4.2]. □

In conclusion, if we choose a minimal sequence $(\Lambda_i)_i$, where in each iteration we capture the bulk of the residuum and compute the Galerkin approximation, then we get a convergent solving procedure for the operator equation. The following result offers a summary.

Proposition 3.6. *For the sequence of Galerkin approximations $(\boldsymbol{u}_i)_i$ produced as discussed above and the choice of μ as above we have*

$$|||\boldsymbol{u} - \boldsymbol{u}_{\Lambda_i}||| \leq (1-\beta^2)^{i/2}|||\boldsymbol{u}|||.$$

If additionally $\boldsymbol{u} \in \mathscr{A}^s$ for some $s > 0$, then

$$\#\operatorname{supp}(\boldsymbol{u}_{\Lambda_i}) \lesssim \|\boldsymbol{u} - \boldsymbol{u}_{\Lambda_{i-1}}\|^{-1/s}\|\boldsymbol{u}\|_{\mathscr{A}^s}^{1/s}.$$

Proof. The proof can be found in [46, Proposition 4.1]. □

Of course, this procedure is not practically implementable. In order to obtain an implementable routine, three issues have to be addressed:

- The approximation of the, in general, infinite residual.
- Enlargement of Λ_i to Λ_{i+1} based on the computed (approximate) residual.
- Galerkin solution on Λ_{i+1}.

Given appropriate solutions to all of the above issues, the following proposition shows that the results from the idealized procedure extend to this setting.

Proposition 3.7. *Let $\delta \in (0, \alpha), \gamma > 0$ be constants such that $\mu := \frac{\alpha+\delta}{1-\delta} < \kappa(\boldsymbol{B})^{-1/2}$ and $\gamma < \frac{(1-\delta)(\alpha-\delta)}{1+\delta}\kappa(\boldsymbol{B})^{-1}$. Let $\boldsymbol{r} \in \ell_2(\Lambda)$ (approximate residual) be such that*

$$\|\boldsymbol{f} - \boldsymbol{B}\boldsymbol{w} - \boldsymbol{r}\| \leq \delta\|\boldsymbol{r}\|.$$

Let $\Lambda_i \subset \Lambda_{i+1} \subset \Lambda$ be such that (bulk criterion)

$$\|R_{\Lambda_{i+1}} \boldsymbol{r}\| \geq \alpha \|\boldsymbol{r}\|$$

and such that $\#(\Lambda_{i+1} \setminus \Lambda_i)$ is minimal up to some absolute multiple. Let $\bar{\boldsymbol{w}} \in \ell_2(\Lambda_{i+1})$ be an approximation to $\boldsymbol{u}_{\Lambda_{i+1}}$ such that

$$\|R_{\Lambda_{i+1}} \boldsymbol{f} - \boldsymbol{B}_{\Lambda_{i+1}} \bar{\boldsymbol{w}}\| \leq \gamma \|\boldsymbol{r}\|.$$

Then

$$|||\boldsymbol{u} - \bar{\boldsymbol{w}}||| \leq \rho |||\boldsymbol{u} - \boldsymbol{w}|||,$$

where

$$\rho := \sqrt{1 - \left(\frac{\alpha - \delta}{1 + \delta}\right)^2 \kappa(\boldsymbol{B})^{-1} + \frac{\gamma^2}{(1-\delta)^2} \kappa(\boldsymbol{B})} < 1$$

and

$$\#(\Lambda_{i+1} \setminus \Lambda_i) \lesssim \min \left\{ N : |||\boldsymbol{u} - \boldsymbol{u}_N||| \leq \sqrt{1 - \bar{\beta}} |||\boldsymbol{u} - \boldsymbol{w}||| \right\}.$$

Proof. The proof can be found in [46, Proposition 4.2]. □

The first issue, concerning approximating the infinite residual, can be solved using the routines **RHS** and **APPLY** as described in the previous subsections. The second issue, concerning expanding Λ_i to Λ_{i+1} is solved by adding to Λ_i those indices from Λ for which the entries of the approximate finitely supported residual are largest in magnitude. This is achieved by the routine **EXPAND**, and the enlarged index set is indeed minimal (see [46, Proposition 4.3]). As for the Galerkin approximation, as an

example of a routine satisfying the requirements of the Proposition 3.7, the Richardson iteration can be utilized (see [46, Proposition 4.4]). One can, however, easily think of other iterative methods to compute the Galerkin approximation. From hereon, we denote this step of AGWM as **GALERKIN**. Thus, we have the ingredients for a valid AWGM routine. We conclude this subsection by stating an **AWGM** routine in Algorithm 3.2 as in [46, Section 4] and the corresponding optimality result.

Algorithm 3.2 AWGM

Input: $\varepsilon, \varepsilon_{-1} > 0$, $\alpha, \delta, \gamma, \theta$ with $\delta \in (0, \alpha)$, $\frac{\alpha+\delta}{1-\delta} < \kappa(B)^{-1/2}$, $\theta > 0$ and $\gamma \in (0, \frac{(1-\delta)(\alpha-\delta)}{1+\delta}\kappa(B)^{-1})$

Output: $\boldsymbol{u}_\varepsilon$ with $\|\boldsymbol{f} - \boldsymbol{B}\boldsymbol{u}_\varepsilon\| \leq \varepsilon$

$i := 0, \boldsymbol{u}^i := 0, \Lambda_i := \emptyset$

loop

$\zeta := \theta\varepsilon_{i-1}$

repeat

$\zeta := \zeta/2$, $\boldsymbol{r}^i := \mathbf{RHS}[\boldsymbol{f}; \zeta/2] - \mathbf{APPLY}[\boldsymbol{B}; \boldsymbol{u}^i; \zeta/2]$

if $\varepsilon_i := \|\boldsymbol{r}^i\| + \zeta \leq \varepsilon$ **then**

$\boldsymbol{u}_\varepsilon := \boldsymbol{u}^i$ stop

end if

until $\zeta \leq \delta\|\boldsymbol{r}^i\|$

$\Lambda_{i+1} := \mathbf{EXPAND}[\Lambda_i; \boldsymbol{r}^i; \alpha]$

$\boldsymbol{u}^{i+1} := \mathbf{GALERKIN}[\Lambda_{i+1}; \boldsymbol{u}^i; \varepsilon_i; \gamma\|\boldsymbol{r}^i\|]$

end loop

return $\boldsymbol{u}_\varepsilon := \boldsymbol{u}^{i+1}$

Theorem 3.3. *The output of* **AWGM** *satisfies* $\|\boldsymbol{f} - \boldsymbol{B}\boldsymbol{u}_\varepsilon\| \leq \varepsilon$. *If* $\boldsymbol{B}$ *is* $\bar{s}$*-admissible for* $\bar{s} \geq s_{\max}$, *then* **AWGM** *is (quasi-)optimal.*

Proof. The proof can be found in [46, Theorem 4.1]. □

4. PDE FOR THE CDO PROBLEM

In this section, we introduce the PDE from [41]. The PDE arises from a stochastic model of a CDO, where the underlying process is an Itō-Process. In such models, typically, the conditional expectation can be equivalently expressed through a PDE. Since the portfolio states are expressed as a Markov chain with a very large number of states, the resulting PDE contains a high-dimensional part, i.e., a huge number of parameters. Under appropriate assumptions, this part can be separated from the wavelet components such that we can apply $\mathcal{H}$-Tucker to store and manage the high-dimensional part.

4.1. VARIATIONAL FORMULATION

The value of the CDO portfolio is described by the function $u = (u^0, \dots, u^J)^T$ where $\mathcal{J} := \{0, \dots, J\}$ are the states of the Markov chain. The function satisfies a system of parabolic PDEs, i.e., for each $j \in \mathcal{J}$ we have

$$\begin{cases} u_t^j(t,y) &= r(t,y)u^j(t,y) - \alpha^T(t)\nabla u^j(t,y) - \dfrac{1}{2}\operatorname{tr}\left[\beta(t)\beta(t)^T H_{u^j}(t,y)\right] \\ &\quad - \displaystyle\sum_{k:k\neq j} d^{j,k}(t,y)(a^{j,k}(t,y) + u^k(t,y) - u^j(t,y)) - c^j(t,y), \\ u(T,y) &= u_T(y) := (a^0(y), \dots, a^J(y))^T. \end{cases} \tag{4.1}$$

As usual, since we are interested in the weak solution and due to homogenization, we can assume homogeneous Dirichlet boundary conditions on the domain of interest $\Omega \subset \mathbb{R}^d$. The parameters are (see [41, Section 6]):

- $r(t,y)$ is the market interest rate.

- $\alpha(t) \in \mathbb{R}^d$ is the drift vector and $\beta(t) \in \mathbb{R}^{d\times d}$ is the volatility matrix stemming from the market process

$$dY(t) = \alpha(t)dt + \beta(t)dW(t),$$

  where $W(t)$ is a d-dimensional Wiener process. The space variable $y \in \mathbb{R}^d$ describes the current market situation.

- $H_u(t, y)$ is the Hessian of u.

- $d^{j,k}(t, y)$ are the transition intensities.

- $a^{j,k}(t, y)$ are the recovery payments.

- $c^j(t, y)$ are the CDO payments.

- $a^j(y)$ are the final payments at maturity.

As can be inferred from [41, Theorem 6.1 and Remark 6.1], the weak form of this PDE is well posed. However, due to the aforementioned high-dimensional structure, i.e., due to the huge number of $\lambda^{j,k}$, it is not possible to solve the PDE numerically, given the current state of numerical methods and storage capacities. For this purpose, we require the following assumption that, as we will see later in Subsection 4.2, allows us to express the semi-discrete problem as a tensor problem, where one side of the tensor product is compatible with the $\mathcal{H}$-Tucker structure, while the other represents the wavelet part.

Assumption 4.1. *Assume that the coefficients have an affine decomposition as*

$$d^{j,k}(t, y) = \tilde{d}^{j,k}(t)h_d(y), \qquad a^{j,k}(t, y) = \tilde{a}^{j,k}(t)h_a(y),$$
$$c^j(t, y) = \tilde{c}^j(t)h_c(y), \qquad a^j(y) = \tilde{a}^j h_{a(T)}(y).$$

In the following, notation of the form $L_2(0,T;X)$ or $H^1(0,T;X)$, where X is some Banach space, is used to denote Bochner spaces[6].

Lemma 4.1. *Let Assumption 4.1 hold. Then for* (4.1) *the variational formulation in space reads*

$$\begin{cases} (u_t, v)_{L_2} + a(u,v) = (f,v)_{L_2}, \\ u(T) = u_T := (a^0, \dots, a^J)^T \end{cases} \tag{4.2}$$

for almost all $t \in [0,T]$ *and* $f \in L_2(0,T;(L_2(\Omega))^{J+1})$. *The test space is* $\mathscr{X} := L_2(0,T;(H_0^1(\Omega))^{J+1}) \cap H^1(0,T;(L_2(\Omega))^{J+1})$ *and the trial space is* $\mathscr{Y} := H_0^1(\Omega)^{J+1}$. *The bilinear form* $a(\cdot,\cdot)$ *is given by*

$$\begin{aligned} a(u,v) := &- \int_\Omega r(t,y) u^T(t,y) v(y) \mathrm{d}y + \int_\Omega \alpha(t)^T \nabla u(t,y) v(y) \mathrm{d}y \\ &- \frac{1}{2} \int_\Omega \operatorname{tr} \left[\left(\frac{\partial}{\partial y} u(t,y) \right)^T \beta(t) \beta(t)^T \nabla v(y) \right] \mathrm{d}y + \int_\Omega h_d(y) u^T(t,y) D^T(y) v(y) \mathrm{d}y \end{aligned}$$

and

$$f(t,y) = -\tilde{c}(t) h_c(y) - \Gamma(t) h_d(y) h_a(y)$$

with $\tilde{c}(t) := (\tilde{c}^0(t), \dots, \tilde{c}^J(t))^T$

$$\Gamma(t) := \left(\sum_{k:k\neq 0} \tilde{d}^{0,k}(t) \tilde{a}^{0,k}(t), \dots, \sum_{k:k\neq J} \tilde{d}^{J,k}(t) \tilde{a}^{J,k}(t) \right)^T$$

[6] See [25, Section 5.9.2].

and $D(t) = (D_{i,j}(t))_{j,i\in\mathcal{J}}$ *with*

$$D_{i,j}(t) := \begin{cases} \tilde{d}^{i,j} & \text{if } i \neq j, \\ -\sum_{k:k\neq j} \tilde{d}^{i,k}(t) & \text{if } i = j. \end{cases}$$

Proof. See [41, Lemma 6.1]. □

Remark 4.1. Note that if $u \in H^1(0,T;X)$ for a Banach space X, then u has a continuous representative $u^* \in C(0,T;X)$ such that the statement $u(T) = u_T$ makes sense. For more details, we refer to [25, Section 5.9.2, Theorem 2].

Theorem 4.1. *If* $\beta(t)\beta(t)^T$ *has full rank and* $\Omega \subset \mathbb{R}^d$ *is a Lipschitz domain, then the problem* (4.2) *is well posed.*

Proof. See [41, Theorem 6.1 and Remark 6.1]. □

4.2. DISCRETIZATION

A common method to solve time-dependent PDEs is to discretize in space and time separately. This is often referred to as *Method of Lines*. In particular, as in [41, Section 6.2], we want to use the θ-time stepping scheme for the PDE in (4.2), and it will be required in the sequel as well. Discretizing in space first, let $u^j(t,y) = \sum_{\lambda\in\Lambda} x^j_\lambda(t)\psi_\lambda(y)$. Note that, given a basis for H^1_0, the basis for the Cartesian product can be simply taken as the canonical basis of the form $e^j_\lambda := (\psi_\lambda \delta_{i,j})_i$ where δ denotes the Kronecker delta. Analogously, we can use the canonical basis as test functions.

Thus, for each $j \in \mathcal{J}$, we get the semi-discrete problem

$$\begin{aligned}
&\frac{\partial}{\partial t}\sum_{\mu\in\Lambda} x_\mu^j(t)(\psi_\mu,\psi_\lambda)_{L_2}\\
&=\sum_{\mu\in\Lambda} x_\mu^j(t)(r(t,\cdot)\psi_\mu,\psi_\lambda)_{L_2}-\sum_{\mu\in\Lambda} x_\mu^j(t)\sum_{i=1}^{d}(\alpha(t))_i(D^i\psi_\mu,\psi_\lambda)_{L_2}\\
&\quad+\frac{1}{2}\sum_{\mu\in\Lambda} x_\mu^j(t)\sum_{i,k=1}^{d}(\beta(t)\beta(t)^T)_{i,k}(D^i\psi_\mu,D^k\psi_\lambda)_{L_2}-\tilde{c}^j(t)(h_c,\psi_\lambda)_{L_2}\\
&\quad-\sum_{k:k\neq j}\Big[\tilde{d}^{j,k}(t)\tilde{a}^{j,k}(t)(h_dh_a,\psi_\lambda)_{L_2}+\tilde{d}^{j,k}(t)\sum_{\mu\in\Lambda} x_\mu^k(t)(h_d\psi_\mu,\psi_\lambda)_{L_2}\\
&\quad-\tilde{d}^{j,k}(t)\sum_{\mu\in\Lambda} x_\mu^j(t)(h_d\psi_\mu,\psi_\lambda)\Big]
\end{aligned}$$

for each $\lambda \in \Lambda$. Hence, in matrix notation with $x^j(t) := (x_\lambda^j(t))_{\lambda\in\Lambda}$ the system for each $j \in \mathcal{J}$ reads as

$$\begin{aligned}
\frac{\partial}{\partial t}M_1x^j(t) =&A(t)x^j(t)-\sum_{k:k\neq j}\tilde{d}^{j,k}(t)\left(M_2(t)(x^k(t)-x^j(t))+\tilde{a}^{j,k}(t)F_1\right)\\
&-\tilde{c}^j(t)F_2,
\end{aligned}\tag{4.3}$$

where

$$\begin{aligned}
M_1 :=&\,((\psi_\mu,\psi_\lambda)_{L_2})_{\lambda,\mu\in\Lambda}\,,\\
M_2(t) :=&\,(h_d\psi_\mu,\psi_\lambda)_{\lambda,\mu\in\Lambda}\,,\\
A(t) :=&\Big((r(t,\cdot)\psi_\mu,\psi_\lambda)_{L_2}-\sum_{i=1}^{d}(\alpha(t))_i(D^i\psi_\mu,\psi_\lambda)_{L_2}\\
&+\frac{1}{2}\sum_{i,k=1}^{d}(\beta(t)\beta(t)^T)_{i,k}(D^i\psi_\mu,D^k\psi_\lambda)_{L_2}\Big)_{\lambda,\mu\in\Lambda},\\
F_1 :=&\,((h_dh_a,\psi_\lambda)_{L_2})_{\lambda\in\Lambda}\,,\\
F_2 =&\,((h_c,\psi_\lambda)_{L_2})_{\lambda\in\Lambda}\,.
\end{aligned}$$

Thus, for the entire system the semi-discrete problem reads

$$\left(\frac{\partial}{\partial t}(I \otimes M_1) - [I \otimes A(t) - D(t) \otimes M_2(t)]\right) x(t) = b(t) \otimes F_1 - \tilde{c}(t) \otimes F_2 \quad (4.4)$$

where

$$I \in \mathbb{R}^{(J+1)\times(J+1)},$$
$$b(t) = -\left(\sum_{k:k\neq j} \tilde{d}^{j,k}(t)\tilde{a}^{j,k}(t)\right)_{j\in\mathcal{J}}$$

and the terminal condition is given by

$$x(T) := \tilde{a} \otimes F_3,$$
$$F_3 := \left((h_{a(T)}, \psi_\lambda)_{L_2}\right)_{\lambda\in\Lambda}$$

with $\tilde{a} := (\tilde{a}^0, \ldots, \tilde{a}^J)^T$. For more details, see [41, Theorem 6.2].

Remark 4.2. The expression "$\frac{\partial}{\partial t}u(t) = \sum_{\lambda\in\Lambda} x'(t)\psi_\lambda$" that was implicitly used above should be interpreted as follows. Let the solution u be in $\mathscr{X} = L_2(0,T;H_0^1(\Omega)) \cap H^1(0,T;L_2(\Omega))$, as in the problem above. Then $u_t(t) \in L_2(\Omega)$, and one can easily show $u_t(t) = \sum_{\lambda\in\Lambda} d_\lambda(t)\psi_\lambda$. By definition of the weak time derivative of u and using properties of the Bochner integral[7]

$$\begin{aligned}\int_0^T u_t(t)v\mathrm{d}t &= \sum_{\lambda\in\Lambda}\psi_\lambda \int_0^T d_\lambda(t)v(t)\mathrm{d}t \\ &= -\int_0^T u(t)v'\mathrm{d}t = -\sum_{\lambda\in\Lambda}\psi_\lambda\int_0^T x_\lambda(t)v'(t)\mathrm{d}t,\end{aligned}$$
$$\int_0^T d_\lambda(t)v(t)\mathrm{d}t = -\int_0^T x_\lambda(t)v'(t)\mathrm{d}t, \quad \forall\lambda\in\Lambda$$

[7]For details on integration theory on general measure spaces, we refer to [40, Sections 17–19].

for a test function $v \in H_0^1((0,T))$. In fact, $x_\lambda \in H^1((0,T))$ and $u_t(t) = \sum_{\lambda\in\Lambda} x'_\lambda(t)\psi_\lambda$. Moreover, since $x_\lambda \in H^1((0,T))$, x has an absolutely continuous representative that is differentiable almost everywhere. Thus, applying finite differences in time makes sense. For a comprehensive analysis of the time-stepping scheme, see, e.g., [47, Section 1].

5. APPROXIMATING DIFFERENTIAL OPERATORS

The method of lines, as described in Subsection 4.2, is a standard method for solving time-dependent boundary value problems. The aim of an adaptive wavelet method is to achieve an approximation to the solution of a prescribed tolerance while keeping computational and storage costs at a minimum. In other words, the aim is to achieve (quasi-)optimality as in Definition 3.1. Due to the nature of time-stepping schemes and to our best knowledge, no such optimality results have been established for time-stepping schemes in this context. On the other hand, as can be seen in Section 3, the analysis concerning (quasi-)optimality of adaptive methods was conducted in a rather general framework, i.e., for general operator equations. Thus, intuitively, these results should naturally extend to parabolic problems, where the operator arises from the space-time variatonal formulations. Indeed, an approach proposed by [43] for solving a general parabolic problem was shown to be (quasi-)optimal. The method introduces no penalty in complexity due to the use of tensor-product bases. Efficient approximate residual evaluation based on multi-trees was presented in [32] and an adaptive method for periodic problems thereafter [31].

The results in [43] are rather technical, and we will thus refrain from discussing them here in detail. Instead, in this section, we will focus on properties of operators of the form as they arise in the semi-discrete problem in (4.3), ignoring for now that these only represent operator equations for a *fixed* t, and are thus only one component of the entire differential operator. This will suffice to illustrate main ideas and wavelet bases properties leading to compressiblity and (quasi-)optimality. Furthermore, we will require the compressibility results in the sequel for solving the semi-discrete problem adaptively. Given an appropriate basis, the estimates can be

used in combination with results from [43] to show (quasi-)optimality for solving the normal equations. For more details, we refer the reader to the aforementioned paper.

5.1. COMPRESSIBILITY

As previously mentioned, compression of operators is usually guaranteed by the properties of the underlying wavelets. In this subsection, we take a closer look to confirm this expectation. To show compressibility, we will proceed as follows: first we show compressiblity in 1D, then we use the result to get compressiblity for the tensor product. Since the operator in (4.3) consists of three parts, we state results that apply to each part individually, and the overall compression then trivially follows. In this section, we are thus interested in operator equations, i.e.

$$\boldsymbol{B}\boldsymbol{u} = \boldsymbol{f},$$

where, however, $\boldsymbol{B}$ and $\boldsymbol{f}$ will be of a more specific form, as will be discussed below. Furthermore, since we will be using tensor product bases later on, we consider operators that arise by discretizing differential operators using tensor product wavelet bases. Note that from hereon, we solely concentrate on compressiblity issues and assume the underlying problem is well posed. For the particular problem in (4.2), existence, uniqueness and stability were shown in [41].

For the 1D compression, we can use the results from [42]. [42, Lemma 3.1] is restated here in its general form for a bounded domain $\Omega \subset \mathbb{R}^d$, and for illustration purposes, the proof is included in more detail. First we need the following notation: for $\lambda, \mu \in \Lambda$ with $\operatorname{supp}(\psi_\lambda) \cap \operatorname{supp}(\psi_\mu) \neq \emptyset$ and $\operatorname{supp}(\psi_\lambda) \cap \partial\Omega = \emptyset$ for $|\lambda| \geq |\mu|$ (and

respectively for $|\lambda| < |\mu|$), define the indicator $i(\lambda,\mu) \in \{0,1\}$ by setting

$$i(\lambda,\mu) := \begin{cases} \operatorname{dist}(\operatorname{singsupp}(\psi_\mu), \operatorname{supp}(\psi_\lambda)) > 0 & \text{when } |\mu| \le |\lambda| \\ \operatorname{dist}(\operatorname{singsupp}(\psi_\lambda), \operatorname{supp}(\psi_\mu)) > 0 & \text{when } |\mu| > |\lambda|. \end{cases}$$

The notation is introduced in order to exploit the fact that many wavelets constructions, in particular those discussed in the previous section, are piecewise polynomials and have vanishing moments. This will allow to use Assumptions 2.8 and 2.9 for the case $i(\mu,\lambda) = 0$. Moreover, note that if the coefficient function is a polynomial itself, then the entire entry vanishes in case $i(\mu,\lambda) = 0$. Vanishing moments do not hold in general for boundary wavelets (see [19]), i.e., wavelets that intersect the boundary. These are the wavelets that are added to the basis system in order to ensure boundary adaptivness. However, as long as the number of such wavelets is uniformly bounded, this does not influence the compressiblity results.

For the next result we are going to need particularly the Assumptions 2.3, 2.8 and 2.9.

Lemma 5.1. *Let $\Omega \subset \mathbb{R}^d$ be a bounded domain. Let $\alpha, \beta \in \mathbb{N}^d$ be multi-indices and D^α the canonical differential of order $|\alpha| = \sum_{i=1}^n \alpha_i$. Assume w.l.o.g. $|\beta| \le |\alpha| \le r+1$ and $|\mu| \le |\lambda|$, where $\mu, \lambda \in \Lambda$. Let the preconditioned entry $a_{\lambda,\mu}^{(\alpha,\beta)}$ be defined as*

$$a_{\lambda,\mu}^{(\alpha,\beta)} := 2^{-|\lambda||\alpha|-|\mu||\beta|} \int_\Omega g D^\alpha \psi_\mu D^\beta \psi_\lambda. \tag{5.1}$$

If $g \in W_\infty^{p+|\beta|}$, then

$$|a_{\lambda,\mu}^{(\alpha,\beta)}| \lesssim \begin{cases} 2^{-\left||\lambda|-|\mu|\right|(\frac{d}{2}+p+|\beta|)} \|g\|_{W_\infty^{p+|\beta|}} & \text{if } i(\mu,\lambda) = 0,\ |\alpha+\beta| \le r+1 \\ 2^{-\left||\lambda|-|\mu|\right|(\frac{d}{2}+r+1-|\alpha|)} \|g\|_{W_\infty^{r+1-|\alpha|}} & \text{otherwise,} \end{cases} \tag{5.2}$$

where we use the norms over the set $\operatorname{supp}(\psi_\mu) \cap \operatorname{supp}(\psi_\lambda)$.

Remark 5.1. Furthermore, in Lemma 5.1 we require that the wavelet system is boundary adapted with Dirichlet boundary, i.e., in this setting we need $D^\eta \psi_\lambda = 0$ on $\partial\Omega$ for all $\eta \leq \beta$ with $\eta \neq \beta$. Note, however, that in the second, better estimate we require that ψ_λ is an inner wavelet, i.e., not stemming from the boundary condition. Otherwise the vanishing moment property is not guaranteed. If ψ_λ is an inner wavelet, then it vanishes on $\partial\Omega$ as well.

Proof of Lemma 5.1. Consider **first** the case $|\alpha + \beta| \geq r + 1$. Then choose $\gamma \leq \beta$ (componentwise) such that $|\gamma + \alpha| = r + 1$. Integrating by parts and using the Dirichlet boundary conditions, we get

$$a_{\lambda,\mu}^{(\alpha,\beta)} = 2^{-|\lambda||\alpha|-|\mu||\beta|} \int_\Omega (-1)^{|\gamma|} D^\gamma (g D^\alpha \psi_\mu) D^{\beta-\gamma} \psi_\lambda.$$

Now observe when applying the Leibniz differentiation rule to $D^\gamma(gD^\alpha\psi_\mu)$, we obtain a sum of mixed derivatives of order up to $r+1-|\alpha|$ for g and $r+1$ for ψ_μ. Thus, we can bound the resulting sum by $\|g\|_{W_\infty^{r+1-|\alpha|}} \cdot \|\psi_\mu\|_{W_\infty^{r+1}}$. For $\|\psi_\mu\|_{W_\infty^{r+1}}$ and $\|\psi_\lambda\|_{W_\infty^{|\beta-\gamma|}}$, we use the property (2.4). As for the measure of the common support, we use the support decay property (2.1) and thus get

$$|a_{\lambda,\mu}^{(\alpha,\beta)}| \lesssim 2^{-|\lambda||\alpha|-|\mu||\beta|} \|g\|_{W_\infty^{r+1-|\alpha|}} 2^{|\mu|(\frac{d}{2}+r+1)} 2^{-|\lambda|d} 2^{|\lambda|(\frac{1}{2}+|\beta-\gamma|)}.$$

Next notice that, since $\beta \geq \gamma$, we get

$$\begin{aligned} |\beta - \gamma| &= \sum_{k=1}^d (\beta_k - \gamma_k) = \sum_{k=1}^d (\beta_k + \alpha_k) - (\alpha_k + \gamma_k) \\ &= \sum_{k=1}^d (\beta_k + \alpha_k) - \sum_{k=1}^d (\alpha_k + \gamma_k) = |\beta + \alpha| - |\alpha + \gamma| = |\beta| + |\alpha| - (r+1). \end{aligned}$$

Summing all together gives

$$|a_{\lambda,\mu}^{(\alpha,\beta)}| \lesssim \|g\|_{W_\infty^{r+1-|\alpha|}} 2^{(|\mu|-|\lambda|)(\frac{d}{2}+r+1-|\alpha|)}.$$

Second, consider the case $|\beta+\alpha| \leq r+1$. Integrating again by parts and using the boundary conditions, we get

$$a_{\lambda,\mu}^{(\alpha,\beta)} = 2^{-|\lambda||\alpha|-|\mu||\beta|} \int_\Omega (-1)^{|\beta|} D^\beta (g D^\alpha \psi_\mu) \psi_\lambda.$$

As in the first case, for g we get derivatives of order up to $r+1-|\alpha|$ and $r+1$ for ψ_μ. Furthermore, we can use the estimate (2.3) for $s := r+1-|\alpha|-|\beta| \leq r+1 \leq p$ and thus get

$$\begin{aligned} |a_{\lambda,\mu}^{(\alpha,\beta)}| &\lesssim 2^{-|\lambda||\alpha|-|\mu||\beta|} \|g\|_{W_\infty^{r+1-|\alpha|}} 2^{|\mu|(\frac{d}{2}+r+1)} 2^{-|\lambda|(\frac{d}{2}+r+1-|\alpha|-|\beta|)} \\ &\lesssim \|g\|_{W_\infty^{r+1-|\alpha|}} 2^{(|\mu|-|\lambda|)(\frac{d}{2}+r+1-|\alpha|)}. \end{aligned}$$

Third, consider the case $i(\mu,\lambda) = 0$ and $|\alpha+\beta| \leq r+1$. Thus, we can use the estimate (2.5) for ψ_μ. With the rest being as before and using $s := p$ for ψ_λ, we get

$$\begin{aligned} |a_{\lambda,\mu}^{(\alpha,\beta)}| &\lesssim 2^{-|\lambda||\alpha|-|\mu||\beta|} \|g\|_{W_\infty^{|\beta|+p}} 2^{|\mu|(\frac{d}{2}+|\alpha|+|\beta|+p)} 2^{-|\lambda|(\frac{d}{2}+p)} \\ &\lesssim \|g\|_{W_\infty^{|\beta|+p}} 2^{(|\mu|-|\lambda|)(\frac{d}{2}+p+|\beta|)}. \end{aligned}$$

This proves the estimate (5.2). □

Next we take a look at the bi-infinite matrix consisting of entries that are of the form as in (5.1) where the wavelets are n-th tensor products of wavelets each in d_m dimensions. Normally one would have $d_m = 1$ for all m. However, the general setting of this result also allows for wavelets in more than one dimension that are

not tensor-products of wavelets on the interval (e.g., [12]), thus also leaving more flexibility w.r.t. the choice of the product domain.

The following theorem is stated as in [42, Theorem 4.1]. To accommodate for the tensor product setting, we require the following notation:

- $\boldsymbol{\Omega} := \bigtimes_{m=1}^{n} \Omega_m$,
- $\boldsymbol{p} := (p_1, \ldots, p_n)$,
- $\boldsymbol{r} := (r_1, \ldots, r_n)$,
- $\boldsymbol{d} := (d_1, \ldots, d_n)$,
- $\boldsymbol{\Lambda} := \bigtimes_{m=1}^{n} \Lambda_m$,
- $|\boldsymbol{\alpha}| := (|\alpha_1|, \ldots, |\alpha_n|)$,
- For $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}$, $|\boldsymbol{\lambda}| := (|\lambda_1|, \ldots, |\lambda_n|)$,
- $\max(\boldsymbol{\alpha}, \boldsymbol{\beta}) := (\max(\alpha_1, \beta_1), \ldots, \max(\alpha_n, \beta_n))$,
- $\boldsymbol{i(\mu, \lambda)} := (i_1(\mu_1, \lambda_1), \ldots, i_n(\mu_n, \lambda_n))$,
- $\boldsymbol{z^{(i)}} := (z_1^{(i_1)}, \ldots, z_n^{(i_n)}) \in \mathbb{R}^n$ where

$$z^{(i)} := \begin{cases} p + \min |\alpha_m|, |\beta_m| & \text{for } i = 0, \\ r + \frac{3}{2} - \max(|\alpha_m|, |\beta_m|) & \text{for } i = 1, \end{cases}$$

- $\boldsymbol{D^{\alpha}} := \bigotimes_{m=1}^{n} D_m^{\alpha_m}$.

Theorem 5.1. *Let* $|\boldsymbol{\alpha}|, |\boldsymbol{\beta}| \leq \boldsymbol{r + 1}$. *Suppose* $\boldsymbol{D^{\gamma} g} \in L_\infty(\boldsymbol{\Omega})$ *for all* $|\boldsymbol{\gamma}| \leq p + \min(|\boldsymbol{\alpha}|, |\boldsymbol{\beta}|)$. *Define the bi-infinite matrix* $\boldsymbol{A}$ *by*

$$(\boldsymbol{A})_{\boldsymbol{\mu},\boldsymbol{\lambda}} := 2^{-|\boldsymbol{\mu}|\cdot|\boldsymbol{\alpha}|-|\boldsymbol{\lambda}|\cdot|\boldsymbol{\beta}|} \int_{\boldsymbol{\Omega}} \boldsymbol{g} \boldsymbol{D^{\alpha}} \left(\bigotimes_{m=1}^{n} \psi_{\mu_m}^{(m)} \right) \boldsymbol{D^{\beta}} \left(\bigotimes_{m=1}^{n} \psi_{\lambda_m}^{(m)} \right), \tag{5.3}$$

where $\cdot$ denotes the inner product on $\mathbb{R}^n$. For any $j \in \mathbb{N}$, we define the compressed matrix $\boldsymbol{A}_j$ by dropping the entries of $\boldsymbol{A}$ if

$$||\boldsymbol{\lambda}| - |\boldsymbol{\mu}|| \cdot \boldsymbol{z}^{\boldsymbol{i}(\boldsymbol{\mu},\boldsymbol{\lambda})} > j. \tag{5.4}$$

Then the resulting error can be bounded by

$$\|\boldsymbol{A} - \boldsymbol{A}_j\| \lesssim j^{n-1} 2^{-j}$$

where we use the norm on $\ell_2(\boldsymbol{\Lambda})$. The number of nonzero entries in each row and column of $\boldsymbol{A}_j$ is of order $\mathcal{O}(j^{n-1}2^{j/s^*})$, where

$$\frac{1}{s^*} := \max_{1\le m\le n} \max\left(\frac{d_m}{p + \min(|\alpha_m|, |\beta_m|)}, \frac{d_m - 1}{r + \frac{3}{2} - \max(|\alpha_m|, |\beta_m|)}\right). \tag{5.5}$$

Proof. We first consider a partition of $\boldsymbol{A}$ into blocks as $\boldsymbol{A} = \sum_{\boldsymbol{i}\in\{0,1\}^n} \boldsymbol{A}^{(\boldsymbol{i})}$, where $\boldsymbol{A}^{(\boldsymbol{i})}$ is the block of all nonzero entries with $\boldsymbol{i}(\boldsymbol{\mu}, \boldsymbol{\lambda}) = \boldsymbol{i}$. Note that there are exactly 2^n such partitions. Hence, we can estimate the error and the number of nonzero entries for each partition, and the same bounds will hold for the entire matrix. By the dropping rule in (5.4), the matrix $\boldsymbol{A}_j$ consists of blocks $\boldsymbol{A}^{(\boldsymbol{i})}_{\boldsymbol{l},\boldsymbol{l}'} = (\boldsymbol{A}^{(\boldsymbol{i})}_{\boldsymbol{\mu},\boldsymbol{\lambda}})_{|\boldsymbol{\mu}|=\boldsymbol{l},|\boldsymbol{\lambda}|=\boldsymbol{l}'}$, where $|\boldsymbol{l}' - \boldsymbol{l}| \cdot \boldsymbol{z}^{(\boldsymbol{i})} \le j$, and we first consider those blocks.

Consider a given dimension m from the tensor product. To estimate the number of nonzero entries, we consider the cases $i_m = 0$ and $i_m = 1$. Note that this is sufficient since the wavelets are tensor products here, i.e., if for a single m a wavelet vanishes, then so does the entire entry in the matrix. If $i_m = 0$, then the number of nonvanishing wavelets can be simply bounded by $2^{\max(l'_m - l_m, 0)d_m}$ as in (2.2). Otherwise, if $i_m = 1$, then, due to the local support of wavelets and the piece-wise smoothness, the number of such wavelets is uniformly bounded if $d_m = 1$, and bounded in general by $2^{\max(l'_m - l_m, 0)(d_m - 1)}$ since the singular support of a wavelet is $(d_m - 1)$-dimensional.

All together we thus can bound the number of nonzero entries in each row of $\boldsymbol{A}_{\boldsymbol{l},\boldsymbol{l}'}^{(\boldsymbol{i})}$ or column of $\boldsymbol{A}_{\boldsymbol{l}',\boldsymbol{l}}^{(\boldsymbol{i})}$ by $2^{\max(\boldsymbol{l}'-\boldsymbol{l},0)\cdot(\boldsymbol{d}-\boldsymbol{i})}$. The parameter s^* was defined in (5.5) in such a way that it is the largest number such that $\boldsymbol{d}-\boldsymbol{i} \leq \frac{\boldsymbol{z}^{(\boldsymbol{i})}}{s^*}$. Counting the number of nonzero elements and considering the dropping rule (5.4), we get

$$\#(\boldsymbol{A}_j)_{\boldsymbol{\mu},\cdot} \lesssim \sum_{\{\boldsymbol{l}':0\leq(\boldsymbol{l}'-|\boldsymbol{\mu}|)\cdot\boldsymbol{z}^{(\boldsymbol{i})}\leq j\}} 2^{(\boldsymbol{l}'-|\boldsymbol{\mu}|)\cdot\boldsymbol{z}^{(\boldsymbol{i})}/s^*} \lesssim \sum_{k=0}^{j} 2^{k/s^*}k^{n-1} \lesssim j^{n-1}2^{j/s^*},$$

where the constant in general depends on $\boldsymbol{z}^{(\boldsymbol{i})}$. Thus the number of nonzero elements in $\boldsymbol{A}_j$ in each row and column is of order $\mathcal{O}(j^{n-1}2^{j/s^*})$.

Now we study the error bound. Recall that each entry of $\boldsymbol{A}$ is an integral over tensor products except for the possibly nonseparable $\boldsymbol{g}$. Since we assumed $\boldsymbol{g}$ to be sufficiently smooth and all of its derivatives essentially bounded, by applying the estimate from Lemma 5.1 and a tensor-product argument, we get

$$\begin{aligned}|(\boldsymbol{A})_{\boldsymbol{\mu},\boldsymbol{\lambda}}| &= \left|2^{-|\boldsymbol{\mu}|\cdot|\boldsymbol{\alpha}|-|\boldsymbol{\lambda}|\cdot|\boldsymbol{\beta}|}\int_{\boldsymbol{\Omega}} \boldsymbol{g}\boldsymbol{D}^{\boldsymbol{\alpha}}\bigotimes_{m=1}^{n}\psi_{\mu_m}^{(m)}\cdot\boldsymbol{D}^{\boldsymbol{\beta}}\bigotimes_{m=1}^{n}\psi_{\lambda_m}^{(m)}\right| \\ &\lesssim \max_{|\boldsymbol{\gamma}|\leq\boldsymbol{z}^{(\boldsymbol{i}(\boldsymbol{\mu},\boldsymbol{\lambda}))}-\frac{\boldsymbol{i}(\boldsymbol{\mu},\boldsymbol{\lambda})}{2}} \|\boldsymbol{D}^{\boldsymbol{\gamma}}\boldsymbol{g}\|_{L_\infty} 2^{||\boldsymbol{\mu}|-|\boldsymbol{\lambda}||\cdot(\frac{\boldsymbol{d}}{2}+\boldsymbol{z}^{(\boldsymbol{i}(\boldsymbol{\mu},\boldsymbol{\lambda}))}-\frac{\boldsymbol{i}(\boldsymbol{\mu},\boldsymbol{\lambda})}{2})}.\end{aligned}$$

Note that the indices here are merely a compact version of (5.2). We apply the Schur Lemma, Lemma 3.1. For any $\boldsymbol{\mu}\in\boldsymbol{\Lambda}$ with $|\boldsymbol{\mu}|=\boldsymbol{l}$, we have

$$\sum_{\{\boldsymbol{\lambda}:|\boldsymbol{\lambda}|=\boldsymbol{l}'\}} |(\boldsymbol{A})_{\boldsymbol{\mu},\boldsymbol{\lambda}}| \lesssim 2^{-|\boldsymbol{l}'-\boldsymbol{l}|\cdot(\frac{\boldsymbol{d}}{2}+\boldsymbol{z}^{(\boldsymbol{i})}-\frac{\boldsymbol{i}}{2})}2^{\max(\boldsymbol{l}'-\boldsymbol{l},0)\cdot(\boldsymbol{d}-\boldsymbol{i})},$$

and similarly for any $\boldsymbol{\lambda}\in\boldsymbol{\Lambda}$ with $|\boldsymbol{\lambda}|=\boldsymbol{l}'$, we have

$$\sum_{\{\boldsymbol{\mu}:|\boldsymbol{\mu}|=\boldsymbol{l}\}} |(\boldsymbol{A})_{\boldsymbol{\mu},\boldsymbol{\lambda}}| \lesssim 2^{-|\boldsymbol{l}'-\boldsymbol{l}|\cdot(\frac{\boldsymbol{d}}{2}+\boldsymbol{z}^{(\boldsymbol{i})}-\frac{\boldsymbol{i}}{2})}2^{\max(\boldsymbol{l}-\boldsymbol{l}',0)\cdot(\boldsymbol{d}-\boldsymbol{i})}.$$

Thus, all together, we get with the Schur Lemma

$$\begin{aligned}\|A^{(\boldsymbol{i})}_{\boldsymbol{l},\boldsymbol{l}'}\| &\lesssim 2^{-2|\boldsymbol{l}'-\boldsymbol{l}|\cdot(\frac{\boldsymbol{d}}{2}+\boldsymbol{z}^{(\boldsymbol{i})}-\frac{\boldsymbol{i}}{2})}2^{\max(\boldsymbol{l}-\boldsymbol{l}',0)\cdot(\boldsymbol{d}-\boldsymbol{i})}2^{\max(\boldsymbol{l}'-\boldsymbol{l},0)\cdot(\boldsymbol{d}-\boldsymbol{i})}\\ &= 2^{-|\boldsymbol{l}'-\boldsymbol{l}|\cdot(\boldsymbol{d}+\boldsymbol{z}^{(\boldsymbol{i})}-\boldsymbol{i})}2^{|\boldsymbol{l}'-\boldsymbol{l}|(\boldsymbol{d}-\boldsymbol{i})} = 2^{-|\boldsymbol{l}'-\boldsymbol{l}|\cdot\boldsymbol{z}^{(\boldsymbol{i})}}.\end{aligned}$$

Applying a straight forward bound and some simple combinatorics, we get

$$\begin{aligned}\|\boldsymbol{A}^{(\boldsymbol{i})}-\boldsymbol{A}^{(\boldsymbol{i})}_j\|^2 &= \sum_{\{\boldsymbol{l},\boldsymbol{l}':|\boldsymbol{l}'-\boldsymbol{l}|\cdot\boldsymbol{z}^{(\boldsymbol{i})}>j\}}\|A^{(\boldsymbol{i})}_{\boldsymbol{l},\boldsymbol{l}'}\|^2\\ &\le \left(\max_{\boldsymbol{l}}\sum_{\{\boldsymbol{l}':|\boldsymbol{l}'-\boldsymbol{l}|\cdot\boldsymbol{z}^{(\boldsymbol{i})}>j\}}\|\boldsymbol{A}^{(\boldsymbol{i})}_{\boldsymbol{l},\boldsymbol{l}'}\|\right)\left(\max_{\boldsymbol{l}'}\sum_{\{\boldsymbol{l}:|\boldsymbol{l}'-\boldsymbol{l}|\cdot\boldsymbol{z}^{(\boldsymbol{i})}>j\}}\|\boldsymbol{A}^{(\boldsymbol{i})}_{\boldsymbol{l},\boldsymbol{l}'}\|\right)\\ &\lesssim \left(\max_{\boldsymbol{l}}\sum_{k=j}^{\infty}2^{-k}k^{n-1}\right)\left(\max_{\boldsymbol{l}'}\sum_{k=j}^{\infty}2^{-k}k^{n-1}\right)\\ &= \left(2^{-j}j^{n-1}\sum_{k=j}^{\infty}2^{-k+j}\left(\frac{k}{j}\right)^{n-1}\right)^2\\ &\lesssim (j^{n-1}2^{-j})^2.\end{aligned}$$

Note here that max is appropriate since as $|\boldsymbol{l}'-\boldsymbol{l}|$ gets large, the sum gets smaller and hence the range of possible $\boldsymbol{l}$ is bounded. □

Theorem 5.1 directly implies s^*-compressibility, which can be easily seen as follows (cf. Definition in 3.3).

Corollary 5.1. *$\boldsymbol{A}$ is s^*-compressible for s^* as defined in* (5.5).

Proof. Replace the compression rule in (5.4) by

$$||\boldsymbol{\lambda}|-|\boldsymbol{\mu}||\cdot\boldsymbol{z}^{\boldsymbol{i}(\boldsymbol{\mu},\boldsymbol{\lambda})} > \varepsilon s^* j$$

where $0 < \varepsilon < 1$. Then the number of nonzero entries in each row and column of $\boldsymbol{A}_j$ can be estimated by

$$\#(\boldsymbol{A}_j)_{\boldsymbol{\mu},\cdot} \lesssim (\varepsilon s^* j)^{n-1} 2^{\varepsilon s^* j / s^*} = (\varepsilon s^* j)^{n-1} 2^{\varepsilon j} = (\varepsilon s^* j)^{n-1} 2^{(\varepsilon-1)j} 2^j = \beta_j 2^j$$

where $\beta_j := (\varepsilon s^* j)^{n-1} 2^{(\varepsilon-1)j} \in \ell_1(\mathbb{N})$. The resulting compression error is bounded as

$$\|\boldsymbol{A} - \boldsymbol{A}_j\| \lesssim (\varepsilon s^* j)^{n-1} 2^{-\varepsilon s^* j}.$$

Now let $0 < \delta < 1$ and observe that

$$(\varepsilon s^* j)^{n-1} 2^{-\varepsilon s^* j} = (\varepsilon s^* j)^{n-1} 2^{s^* \varepsilon j(\delta-1)} 2^{-\varepsilon\delta s^* j} = \alpha_j 2^{-\varepsilon\delta s^* j}$$

where $\alpha_j := (\varepsilon s^* j)^{n-1} 2^{s^* \varepsilon j(\delta-1)} \in \ell_1(\mathbb{N})$. By setting $s := \varepsilon\delta s^*$ and since $0 < \varepsilon < 1$ and $0 < \delta < 1$ were chosen arbitrarily, the result holds for all $0 < s < s^*$. □

For illustration purposes we apply all of the above results to an operator of the form of the semi-discrete problem (4.4), i.e., $d_m = 1$ for all m. Recall that due to Assumption 4.1, the high-dimensional part of the problem was separated from the wavelet part. We switch to $\Psi := \{\psi_\lambda : \lambda \in \Lambda\}$ to denote the wavelet system for $\mathbb{R}^n$

$$\boldsymbol{A}(t) := \Bigg((r(t,\cdot)\psi_\mu, \psi_\lambda)_{L_2(\Omega)} - \sum_{i=1}^{n} (\alpha(t))_i \left(D^{\alpha_i}\psi_\mu, \psi_\lambda\right)_{L_2(\Omega)}, \\ + \sum_{i,j=1}^{n} \frac{(\beta(t)\beta(t)^T)_{i,j}}{2} \left(D^{\alpha_i}\psi_\mu, D^{\alpha_j}\psi_\lambda\right)_{L_2(\Omega)} \Bigg)_{\lambda,\mu\in\Lambda}.$$

The operator $\boldsymbol{A}(t)$ is s^*-compressible according to Theorem 5.1 and Corollary 5.1, where $s^* = p + 1$ with p being the order of vanishing moments of the underlying wavelets system. The preconditioning for an entry of $\boldsymbol{A}(t)$ can be taken as $2^{-|\lambda|-|\mu|}$.

The compression rule can be reduced to

$$z^{(i)} := \begin{cases} p+1 & \text{for } i = 0, \\ r+\frac{1}{2} & \text{for } i = 1, \end{cases}$$

where r is the global smoothness of the wavelet system.

It is clear that Theorem 5.1 applies to a wide range of operators. However, for the particular operator $\boldsymbol{A}(t)$, combining these results with the work in [19], we can obtain a much better estimate for the sparsity, s^* and a simpler compression criterion (see also [30, Section 4.4]). To be precise, we obtain better estimates for the second and third terms of $\boldsymbol{A}(t)$. As for the other terms, same improvements apply if in the particular setting $r(t,\cdot)$ and h_d are constant. To illustrate the origin of better estimates, the results are stated here with a short proof sketch, since many details follow analogously to Theorem 5.1.

Theorem 5.2. *Let the operator $\boldsymbol{B}$ be defined as*

$$\boldsymbol{B} := \left(\boldsymbol{a}(\boldsymbol{\psi}_{\boldsymbol{\lambda}}, \boldsymbol{\psi}_{\boldsymbol{\mu}})\right)_{\boldsymbol{\lambda},\boldsymbol{\mu}\in\boldsymbol{\Lambda}}$$

where

$$\boldsymbol{a}(\boldsymbol{\psi}_{\boldsymbol{\lambda}}, \boldsymbol{\psi}_{\boldsymbol{\mu}}) = 2^{-|\boldsymbol{\lambda}|-|\boldsymbol{\mu}|} \int_\Omega D^\alpha \left(\bigotimes_{m=1}^n \psi_{\lambda_m}\right) D^\beta \left(\bigotimes_{m=1}^n \psi_{\mu_m}\right)$$

with $\Omega \subset \mathbb{R}^d$ being a bounded product domain and $|\alpha| = |\beta| = 1$. The wavelets ψ_λ on the interval are assumed to have $p \geq 3$ vanishing moments. Then $\boldsymbol{B}$ is s^*-compressible with $s^* = \infty$. We derive the matrix $\boldsymbol{B}_j$ from $\boldsymbol{B}$ by dropping all entries for which

$$\| |\boldsymbol{\lambda}| - |\boldsymbol{\mu}| \|_\infty > j.$$

Then the number of nonzero entries in each row or column of $\boldsymbol{B}_j$ *is of the order* $\mathcal{O}(nj)$*, and the resulting error can be bounded by*

$$\|\boldsymbol{B} - \boldsymbol{B}_j\| \lesssim 2^{-(r+\frac{1}{2})j}.$$

Proof. Consider first the case $d = 1$ and let w.l.o.g. $|\lambda| \geq |\mu|$. If $i(\mu, \lambda) = 0$, then, using homogeneous boundary conditions,

$$\int_\Omega \psi'_\mu \psi'_\lambda \mathrm{d}x = -\int_\Omega \psi_\mu \psi''_\lambda \mathrm{d}x = 0$$

where the last equality is due to the vanishing moment property. If $i(\mu, \lambda) = 1$, then the entry does not vanish in general, but the number of such entries is uniformly bounded in $|\lambda|$ and $|\mu|$. This implies the number of nonzero entries in each row or column of B_j is $\mathcal{O}(j)$. The entry bound can be shown similarly to Lemma 5.1, and the overall error bound can be shown as in Theorem 5.1 applying the Schur Lemma. For the general case $d > 1$, due to the tensor product structure, the number of nonzero entries in each row or column of $\boldsymbol{B}_j$ is $\mathcal{O}(nj)$ and the error bound is again shown as in Theorem 5.1.

The estimates imply $s^* = \infty$ which can be seen as follows (cf. [30, Theorem 4.14]). Replace j in the dropping criterion by 2^j. The number of nonzero entries is now $\mathcal{O}(n2^j)$. For any $s > 0$, $js \leq C(s) + (r + 1/2)2^j$ for an appropriate choice of $C(s)$. This gives

$$\|\boldsymbol{B} - \boldsymbol{B}_j\| \lesssim 2^{-(r+\frac{1}{2})2^j} \leq 2^{C(s)} 2^{-js}.$$

The fact that this implies Definition 3.3 can be shown in a similar fashion as in Corollary 5.1 by adjusting the dropping criterion. □

Note that the estimate for s^* in (5.5) comes from the bound on the number of nonzero entries. This implies the result of Theorem 5.2 can be viewed as a limiting case of Theorem 5.1 where as $d_m \to 0$, $s^* \to \infty$.

Another observation that can be made at this point is that the results would hold for polynomial coefficients as well. In summary, we thus obtain for PDE operators of the form as in (4.4) that these are s^*-compressible by Theorem 5.1. Furthermore, in case $r(t, y)$ is constant in space or, more generally, polynomial, we obtain by Theorem 5.2 that $s^* = \infty$ and the number of nonzero entries in each row or column is of linear complexity.

5.2. RIGHT-HAND SIDE

Last but not least the approximation of the RHS should be considered. Recall from Subsection 3.2 the general form of the RHS arising from an operator equation is

$$\boldsymbol{f} = (\langle f, \psi_\lambda \rangle)_{\lambda \in \Lambda},$$

where $\langle \cdot, \cdot \rangle$ denotes the standard duality pairing. Thus, $\boldsymbol{f} \in \ell_2(\Lambda)$ since Ψ is a Riesz Basis and is, in general, an infinitely supported vector. The question is thus whether and how we can approximate $\boldsymbol{f}$ by a finitely supported vector to any desired accuracy. Note that the given $\boldsymbol{f}$ is the RHS of the equation

$$\boldsymbol{B}\boldsymbol{u} = \boldsymbol{f},$$

where $\boldsymbol{B}$ is s^*-compressible and $\boldsymbol{u} \in \mathscr{A}^s$, so that $\boldsymbol{f} \in \mathscr{A}^s$. This follows from the fact that $\boldsymbol{B} : \mathscr{A}^s \to \mathscr{A}^s$ is a bounded linear operator, which itself is implied by s^*-compressibility for $s^* > s$ (cf. Proposition 3.1). Thus, $\boldsymbol{f}$ is in the appropriate class in the sense that its best N-term approximation converges.

For our particular PDE, the portion of RHS to be approximated is of the form

$$\boldsymbol{f} = ((f, \psi_\lambda)_{L_2})_{\lambda \in \Lambda}\,.$$

Proceeding as in [19, Section 5], for the estimate assume that $\operatorname{supp}(\psi_\lambda)$ has empty intersection with the boundary and that $f \in W^p_\infty$. Then, by using the fact that ψ_λ has p vanishing moments and the estimate in (2.3), we get for the 1D case

$$\left|2^{-|\lambda|}\int_\Omega f\psi_\lambda\right| \lesssim 2^{-(\frac{3}{2}+p)|\lambda|}\|f\|_{W^p_\infty},$$

and generally for the n-dimensional case by a tensor product argument

$$\left|2^{-|\boldsymbol{\lambda}|}\int_\Omega f\boldsymbol{\psi_\lambda}\right| \lesssim 2^{-(\frac{3}{2}+p)|\boldsymbol{\lambda}|}\|f\|_{W^p_\infty}.$$

Given this estimate, consider an approximation to $\boldsymbol{f}$ by dropping all entries outside the set

$$\boldsymbol{\Lambda}_\ell := \{\boldsymbol{\lambda} \in \boldsymbol{\Lambda} : |\boldsymbol{\lambda}| \le \ell\}\,.$$

Note that $\#\boldsymbol{\Lambda}_\ell \sim 2^\ell$ due to the fact $\#\{\lambda \in \Lambda : |\lambda| \le l\} \sim 2^l$. The approximation error is thus bounded by

$$\|\boldsymbol{f} - \boldsymbol{f}_\ell\| \lesssim \sqrt{\sum_{k>\ell} 2^k 2^{-2k(\frac{3}{2}+p)}} \lesssim 2^{-(1+p)\ell}.$$

Hence, with a support length N, the error should behave as $\mathcal{O}(N^{-(1+p)})$.

In general, replace p by $p(\lambda)$ where $p(\lambda) = 0$ if $\mathrm{supp}(\psi_\lambda)$ intersects the boundary and $p(\lambda) = p$ otherwise. The dropping criterion reads now

$$\sum_{k=1}^{n} \left(\frac{3}{2} + p(\lambda_k)\right) |\lambda_k| > \left(\frac{3}{2} + p\right) \ell.$$

As previously mentioned, we assume that the number of boundary wavelets is uniformly bounded for the chosen wavelet construction (cf. Assumption 2.7). This implies

$$\#\left\{\lambda \in \Lambda : |\lambda| \leq l \text{ and } p(\lambda) = p, \text{ or } |\lambda| \leq l \text{ and } p(\lambda) = 0\right\} \sim 2^l + (l+1) \sim 2^l.$$

Similarly we obtain an approximation to $\boldsymbol{f}$ of order $\mathcal{O}(N^{-(1+p)})$. Note that $s_{\max} \leq p - 1 \leq p + 1$ which means this approximation is better than the best possible rate (cf. Assumption 3.1).

Note that since the goal is an adaptive scheme, it would be of more interest to consider nonsmooth f. Here one could apply the approach discussed in [30, Section 4.6.3] for approximating such f under appropriate assumptions. This is subject to further investigation and will not be discussed here.

5.3. QUASI-OPTIMALITY

Thus far we have discussed the approximation of the RHS and the differential operator in the equation of the form

$$\boldsymbol{B}\boldsymbol{u} = \boldsymbol{f}.$$

It remains to discuss (quasi-)optimality as defined in Definition 3.1. This can be demonstrated analogously to [46, Section 5]. Recall that for (quasi-)optimality, we require that we can produce an approximation to the solution $\boldsymbol{u}$ up to any desired tolerance, with storage and computational complexity that grows linearly in the out-

put size. Furthermore, to be more precise, given the following discussion, (quasi-)optimality is guaranteed for the *normal equation*, since $\boldsymbol{B}$ itself will not be s.p.d.

First, as mentioned in Section 3, in order to have optimality, we require that $\boldsymbol{B}$ is s-admissible for $s \geq s_{\max}$, where $s_{\max}$ is the best possible approximation rate for $\boldsymbol{u}$. Since, as discussed in Subsection 3.3, s^*-computability implies s-admissibility, it suffices to verify that $\boldsymbol{B}$ is s^*-computable for $s^* \geq s_{\max}$.

We consider an operator defined by entries as in (5.3), or, more generally,

$$(\boldsymbol{B})_{\mu,\lambda} := \sum_{|\alpha|,|\beta|\leq m} 2^{-(|\mu|+|\lambda|)m} \int_\Omega g_{\alpha,\beta} D^\alpha \psi_\mu D^\beta \psi_\lambda, \tag{5.6}$$

where $m \leq r+1$, with r being the global order of smoothness of the wavelet basis and $\Omega \subset \mathbb{R}^n$. We require that the wavelet basis possesses the approximation property as in Assumption 2.8 which gives the Whitney-type estimate

$$\inf_{v_l \in \operatorname{span}\{\psi_\lambda : |\lambda| \leq l\}} \|u - v_l\|_{H^m} \lesssim 2^{-(p-m)l} \|u\|_{H^p}, \quad u \in H^p(\Omega) \cap H_0^m(\Omega),$$

where the last condition is due to wavelets with homogeneous Dirichlet boundary of order m and $p > m$ is the order of vanishing moments. Due to the fact that

$$\#\{\lambda \in \Lambda : |\lambda| \leq l\} \sim 2^{nl},$$

then rewriting

$$2^{-(p-m)l} \|u\|_{H^p} = 2^{nl\left(-\frac{(p-m)}{n}\right)} \|u\|_{H^p},$$

we observe that

$$s_{\max} \leq \frac{p-m}{n}.$$

In case of a tensor product basis, as in our problem, similarly the best approximation rate is known to be

$$s_{\max} = \max_l \frac{p_l - t}{n_l},$$

for approximating sufficiently smooth functions in $H^t(\Omega)$, where $\Omega \subset \mathbb{R}^n$ is a product domain. For more details, see [46, Section 7.2] and references therein. For most applications, as in the work of [41], $n_l = 1$ and $p_l = p$, for all l. Thus $s_{\max} = p - t$ which also shows that the so called "curse of dimensionality" in the context of approximation rates – the rate is inversely proportional with the space dimension – is removed for tensor bases. Comparing this rate with s^* from (5.5), since obviously $p + t \geq p - t$, we easily verify that $s^* \geq s_{\max}$.

Furthermore, we require s^*-computability of $\boldsymbol{B}$ to apply Theorem 3.3 that ensures that the resulting **AWGM** is (quasi-)optimal. Hence, we require suitable quadrature rules to compute the significant entries of $\boldsymbol{B}$ that keep the error on the same level and require $\mathcal{O}(1)$ operations per entry of a row or column of $\boldsymbol{B}$. For operators as defined in (5.6), this can be achieved by applying a product composite quadrature rule, keeping s^*-computability for the same value of s^* as in the s^*-compressiblity estimate. This was shown in, e.g., [46, Section 5.2] or [42, Section 6]. This approach, in general, still requires computing entries that involve wavelets on largely different levels. A more efficient approach, avoiding such computations, can be realized through Trees, as discussed in [46, Section 5.3] or [31].

Last but not least, we need to verify the availability of $\mathbf{RHS}[\boldsymbol{f}, \varepsilon]$ as assumed in Assumption 3.1. In the previous subsection we described an approximation for a sufficiently smooth $\boldsymbol{f}$. Furthermore, as in the case of $\boldsymbol{B}$, we require a suitable quadrature rule for approximating the entries, keeping the error on the same level and requiring $\mathcal{O}(1)$ operations per entry. As was shown in [19] and [42], by taking

again a product composite quadrature rule, the error is maintained on the same level and the number of operations can be bounded by some absolute multiple of $\#\operatorname{supp}(\boldsymbol{f}_\varepsilon)$. As was stated in Proposition 3.4, this shows that for a given tolerance $\varepsilon > 0$, we obtain an approximation

$$\|\boldsymbol{f} - \boldsymbol{f}_\varepsilon\| \leq \varepsilon,$$

where the number of operations is bounded by some absolute multiple of

$$\varepsilon^{-1/s}\|u\|_{\mathscr{A}^s}^{1/s} + 1,$$

and the storage requirement can also be bounded as

$$\#\operatorname{supp}(\boldsymbol{f}_\varepsilon) \lesssim \varepsilon^{-1/s}\|u\|_{\mathscr{A}^s}^{1/s}.$$

All together this ensures the (quasi-)optimality of the **AWGM**. See [46] for more details.

6. $\mathcal{H}$-TUCKER-APPLY

In this section, we are concerned with adaptive schemes to solve PDEs of the form as in (4.1). There are several possibilities to approach this issue.

An initial ansatz would be to tackle the tensor problem as derived in (4.4). One could think of applying an iterative solver to the problem, e.g., BiCGSTAB as in the work of [41], coupled with approximate residual evaluations via **APPLY** and **RHS**, which would result in a scheme similar to **AWGM**. In terms of the current state of software, this would also be the easiest to implement approach. This method will be discussed in Subsection 6.5. Firstly, due to the nature of time-stepping, no (quasi-)optimality can be guaranteed, e.g., storage requirement and computational complexity will generally depend on the number of time-steps, and Definition 3.1 is not directly applicable to this situation.

Alternatively, one could apply Richardson iterations, as discussed in Subsection 6.4. This yields a convergent scheme for the semi-discrete problem, applied to the operators directly, since these are positive definite. The iteration method is applicable to a wider range of operator equations, as opposed to **AWGM** (see [10]). The same statement concerning (quasi-)optimality as above holds in this situation as well.

And lastly, one could consider a different operator equation, by reformulating (4.2) in a space-time variational form. Hence, we would simply have an operator equation of the form

$$\boldsymbol{B}\boldsymbol{u} = \boldsymbol{f}.$$

From a theoretical point of view, this would be the "nicest" approach. Using results from previous works (e.g., [43]) and considering the discussion in Section 5, we get that the **AWGM** applied to the normal equations is near to (quasi-)optimal. By

"near to", we mean that due to evaluations of the form Dx, where D and x are given in $\mathcal{H}$-Tucker format, we will not obtain (quasi-)optimality in the classical sense, since the complexity will generally depend on the high-dimensional part D, the input vector x and the structure of the $\mathcal{H}$-Tucker format of both[8]. Assuming we can separate the high-dimensional part, this approach would also be generally implementable. However, considering the current state of software, this would be by far the most costly method to implement.

At this point, we would like to emphasize that this section provides three proposals for an adaptive scheme to solve operator equations as in (4.2) with some initial analysis. However, the discussion is by no means complete. In order to fully comprehend the quantitative properties of all three of the proposed schemes, a more thorough analysis and extensive numerical testing would be required. For instance, an issue that is ignored in the following discussion is that matrix-vector operations on the $\mathcal{H}$-Tucker tensors that are heavily used in, e.g., linear solvers, necessarily lead to truncations of the $\mathcal{H}$-Tucker storage format. These truncations introduce an additional error, and a careful analysis of the introduced error w.r.t. the error of the wavelet approximation of the solution is important for the practical implementation. This is, however, outside the scope of this work.

6.1. GENERAL KRONECKER PRODUCTS

In this subsection, we briefly describe the implementation of matrix-vector products with general Kronecker products as implemented in the current software and stated in [41, Section 5.5]. This tells us exactly what kind of operations with tensors in $\mathcal{H}$-Tucker are currently permitted.

[8]Note that this remark applies to all of the three approaches mentioned here.

Consider the Kronecker sum

$$M := H_1 \otimes A_1 + H_2 \otimes A_2.$$

Apply M to a vector x, where $x = x_1 \otimes x_2$ such that $H_i x_1$ and $A_i x_2$ both make sense for $i \in \{1, 2\}$. Then

$$Mx = H_1 x_1 \otimes A_1 x_2 + H_2 x_1 \otimes A_2 x_2.$$

We think of H_i being in $\mathcal{H}$-Tucker format, A_i being simple matrices, e.g., simply stored as an array or, in the FLENS library, as GeneralMatrix. More importantly, for operators arising from PDEs, frequently A_i will be sparse. Sparse matrices cannot be incorporated well into the $\mathcal{H}$-Tucker format, given the current state of software, since the $\mathcal{H}$-Tucker implementation is designed to store dense matrices. In a naive approach, we would have to densify A_i which is highly inefficient. A remedy suggested in [41, Section 5.5] is to apply binary operator tree structure, justified by the above equation. This can be implemented using expression templates in C++ ([50, Chapter 18]). In a nutshell, the matrix M is not set up explicitly and instead the class HTuckerClosure stores a reference to the left and right operands together with the operation given by a template parameter. In each operand, the (possibly sparse) matrices A_i are not converted to $\mathcal{H}$-Tucker but rather stored as references to the original matrices. Explicit evaluations are only performed when we compute Mx and the results are stored in the same format. For instance, an initialization of the matrix $M := A \otimes I_1 + I_2 \otimes B$, where $A \in \mathcal{H}$ and B is a general dense matrix, is given in Figure 6.1 (cf. [41, Listing 5.38]).

The general procedure for computing Mx is as follows.

1. Compute $T_i := H_i x_1$ by the matrix-vector product in $\mathcal{H}$-Tucker format.

```
HTuckerTree<double> A(5);
flens::GeneralMatrix<FullStorage<double, ColMajor> > C(7,7);
MatrixTensor<flens::GeneralMatrix<FullStorage<double, ColMajor> > >
    B(1);
C.setMatrix(1,C);
DimensionIndex index1(1);
index1=7;
DimensionIndex index2(5,4);
IdentityTensor I1(index1);
IdentityTensor I2(index2);

HTuckerClosure<OpAdd,
        HTuckerClosure<OpTensor,
                HTuckerClosure<OpMat, HTuckerTree<double>,
                    HTuckerTree<double> >,
                IdentityTensor>,
        HTuckerClosure<OpTensor,
                IdenityTensor,
                MatrixTensor<flens::GeneralMatrix<FullStorage<double
                    , Colmajor> > >
        >
> M=A*I1+I2*B;
```

Figure 6.1. Initialization of a general Kronecker product matrix.

2. Compute $N_i := A_i x_2$ by the standard matrix-vector product.

3. Concatenate $T_i \otimes N_i$, where again N_i is not converted to $\mathcal{H}$-Tucker but rather stored as a reference.

4. Add to get the end result $T_1 \otimes N_1 + T_2 \otimes N_2$ in $\mathcal{H}$-Tucker format.

6.2. SEMI-DISCRETE PROBLEM

In this subsection, we take a closer look at the semi-discrete problem (4.3) and, in particular, the arising operators. The discussion is essentially analogous to semi-discrete problems arising in FEM. For more details, e.g., stability of the semi-discrete problem, we refer to [47], or, stability of the θ-scheme can be found in, e.g., [39].

Consider a general parabolic Dirichlet BVP with initial data

$$\begin{cases} u_t + Lu = f, & (t,x) \in (0,T] \times \Omega \\ u(0,x) = u_0(x), & x \in \Omega, \end{cases} \tag{6.1}$$

for an elliptic L. The weak form reads as

$$\begin{cases} (u_t, v) + a(t; u, v) = (f, v), & v \in V \subset \mathscr{X}, \\ u(0) = u_0, \end{cases} \tag{6.2}$$

where we assume homogeneous boundary data and the same test and trial Hilbert space $\mathscr{X}$ (Galerkin method). One can think of $\mathscr{X}$ being a Sobolev space, or, for our purposes, a vector-valued Sobolev space. Discretizing in space by writing $u(t) = \sum_{\lambda \in \Lambda} x(t)\psi_\lambda$, we get the semi-discrete problem

$$\mathcal{M}x'(t) + \mathcal{A}(t)x(t) = F(t).$$

Here $\mathcal{M}$ is the *mass* matrix and $\mathcal{A}$ is the *stiffness* matrix. The operator $\mathcal{A}(t)$ inherits the properties of the bilinear form. In particular, if the bilinear form $a(t; \cdot, \cdot)$ is positive definite, then so is $\mathcal{A}(t)$. The mass matrix $\mathcal{M}$ is always positive definite, since it is a Gramian. Hence, the semi-discrete problem has a unique solution. To obtain a fully-discrete problem, apply a θ-scheme for the time discretization

$$\begin{aligned} \mathcal{M}\frac{x(t+\Delta t) - x(t)}{\Delta t} =& \theta(F(t+\Delta t) - \mathcal{A}(t+\Delta t)x(t+\Delta t)) \\ &+ (1-\theta)(F(t) - \mathcal{A}(t)x(t)), \end{aligned} \tag{6.3}$$

i.e.,

$$(\mathcal{M} + \Delta t\theta\mathcal{A}(t+\Delta t))\, x(t+\Delta t) = (\mathcal{M} - \Delta t(1-\theta)\mathcal{A}(t))x(t) \\ + \Delta t(\theta F(t+\Delta t) + (1-\theta)F(t)),$$

where for simplicity we assumed a uniform time grid. Note again that the operator on the LHS, namely $\mathcal{M}+\Delta t\theta\mathcal{A}(t+\Delta t)$, is positive definite. For $\theta \in [0,1]$, the scheme is A-stable. See [39] for a detailed analysis.

If we replace the initial condition by a terminal condition, then we get the problem

$$\begin{cases} -(u_t, v) + \bar{a}(t; u, v) = (f, v), \qquad v \in \mathscr{X}, \\ u(T) = u_T. \end{cases}$$

Note that the minus in front of the time derivative is important for the problem to be well posed or, equivalently, without the minus sign, we would require $-\bar{a}(\cdot,\cdot)$ to be weakly coercive, instead of $\bar{a}(\cdot,\cdot)$. By introducing the time transformation $v(T-t) = u(t)$ for $t \le T$, the above problem is equivalent to the IVP (6.2). In the case $V \subset (H^1(\Omega))^{J+1}$, for the IVP to be well posed, it suffices when the bilinear form $a(t;\cdot,\cdot)$ is continuous and weakly coercive, i.e., satisfies the Gårding inequality

$$\exists \gamma \ge 0,\ \alpha > 0 :\ a(t; v, v) + \gamma\|v\|_{L_2}^2 \ge \alpha\|v\|_V^2, \quad \forall v \in V. \tag{6.4}$$

This has been shown for the particular PDE in (4.2), see [41, Theorem 6.1]. If (6.4) is satisfied, then we can assume w.l.o.g. that $\gamma = 0$, i.e., the bilinear form is coercive. This can be seen as in [39, Section 11.1.1] by applying the transformation $u_\gamma := e^{-\gamma}u$.

Then

$$\frac{\partial}{\partial t}u_\gamma = e^{-\gamma t}u_t - \gamma u_\gamma,$$
$$Lu_\gamma = e^{-\gamma t}Lu,$$

which implies (6.1) is equivalent to

$$\begin{cases} \frac{\partial}{\partial t}u_\gamma + Lu_\gamma + \gamma u_\gamma = e^{-\gamma t}f, \\ u_\gamma(0,x) = u_0. \end{cases}$$

The associated bilinear form is

$$a_\gamma(u,v) = a(u,v) + \gamma(u,v),$$

and, if $a(\cdot,\cdot)$ is weakly coercive, then

$$a_\gamma(v,v) \geq \alpha\|v\|_V^2 - \gamma\|v\|_{L_2} + \gamma\|v\|_{L_2},$$

and hence, from hereon, we will assume $a(\cdot,\cdot)$ is coercive. This gives in particular for our problem that the bilinear form is positive definite and, hence, so is the associated discrete operator $\mathcal{A}$. To be more precise, the semi-discrete problem for our PDE system reads

$$\mathcal{M}x'(t) - \mathcal{A}(t)x(t) = F(t),$$

where, comparing with (4.3)

$$\begin{aligned} \mathcal{M} &= I \otimes M_1, \\ \mathcal{A}(t) &= I \otimes A(t) - D(t) \otimes M_2(t), \\ F(t) &= b(t) \otimes F_1 - \tilde{c}(t) \otimes F_2. \end{aligned} \tag{6.5}$$

The fact that this is indeed the discretized operator associated with the bilinear form can be seen by testing against $v_{\lambda,j}$ with

$$v_{\lambda,j} \in H_0^1(\Omega)^{J+1}, \quad (v_{\lambda,j})_k := \begin{cases} \psi_\lambda, & \text{if } k = j \\ 0, & \text{otherwise.} \end{cases}$$

For details of this derivation, see, e.g., Section 4 or [41].

6.3. PRECONDITIONING

As was discussed in the Section 5, the operators we obtain are s^*-compressible *post* scaling. Here we briefly elaborate on the topic of preconditioning. In the sequel, we will assume the operators have been already properly preconditioned.

There are several possibilities for preconditioning. Generally, as described in [41, Section 6.4.1], for an equation of the form $Bx = f$, where B is of the form

$$B = I \otimes A + D \otimes M,$$

one could think of preconditioning the wavelet part only, i.e., using a preconditioner of the form $I \otimes P^{-1}$, where P is the wavelet preconditioner. If A and M require the same preconditioner, this approach would work. The preconditioned system is of the

form

$$(I \otimes P^{-1}AP^{-1} + D \otimes P^{-1}MP^{-1})\bar{x} = f_1 \otimes P^{-1}f_2, \quad \bar{x} = x_1 \otimes Px_2. \tag{6.6}$$

This is basically equivalent to incorporating preconditioning into the basis, i.e., constructing a Riesz basis of the form $\Psi := \left\{2^{-|\lambda|}\psi_\lambda : \lambda \in \Lambda\right\}$ (cf. [19]). However, consider the case $M = I$. The preconditioned operator reads

$$I \otimes P^{-1}AP^{-1} + D \otimes P^{-1}P^{-1}.$$

Clearly this results in an ill-conditioned operator. Furthermore, even if $M \neq I$, recall that in the original version part of the operator is a mass-matrix that, using orthonormal multi-wavelets as in [41], is in fact an identity operator. Hence, we have to apply a preconditioner to the entire operator. For instance, a basic preconditioner would be the Jacobi preconditioner, i.e., a diagonal matrix P with diagonal entries $P_{i,i} = \left(\sqrt{B_{i,i}}\right)^{-1}$. We would require the $\mathcal{H}$-Tucker representation of such a preconditioner, and this can be achieved as desribed in [41, Section 6.4.1]. One could consider implementing more sophisticated preconditioners. A good survey on preconditioning can be found in [3]. Note, however, that, in general, implementing a preconditioner in $\mathcal{H}$-Tucker format is a nontrivial task.

Unfortunately, for our purposes, this is not sufficient. Adaptive wavelet methods rely specifically on the compressibility of the wavelet operators. Hence, we require the more specific preconditioning as in (6.6). For of our particular PDE, as discussed in Subsection 6.2, this would suffice to obtain compressible operators. In a more general setting, for the purpose of different row and column scaling, let us consider preconditioning the wavelet matrices using (possibly different) left and right preconditioners,

P_1 and P_2 respectively. Recall the semi-discrete problem for $j \in \mathcal{J}$ as in (4.4),

$$\frac{\partial}{\partial t} M_1 x^j(t) = A(t)x^j(t) - \sum_{k:k\neq j} \tilde{d}^{j,k}(t) \left(M_2(t)(x^k(t) - x^i(t)) + \tilde{a}^{j,k} F_1 \right) - \tilde{c}^j(t) F_2.$$

The preconditioned semi-discrete problem reads

$$\begin{aligned} \frac{\partial}{\partial t}(I \otimes P_1^{-1} M_1 P_2^{-1})\bar{x}(t) - \left(I \otimes P_1^{-1} A(t) P_2^{-1} - D(t) \otimes P_1^{-1} M_2(t) P_2^{-1} \right) \bar{x}(t) \\ = b(t) \otimes P_1^{-1} F_1 - \tilde{c}(t) \otimes P_1^{-1} F_2(t), \\ \bar{x}(t) = (I \otimes P_2) x(t). \end{aligned}$$

For $P_1 \neq I$, the system will have an ill-conditioned mass matrix. A simple way out would be to apply another preconditioner, however, now to the entire system. E.g., we could apply the Jacobi preconditioner mentioned above. In practice, preconditioning the wavelet matrices will not cost computational effort since the matrix-matrix and matrix-vector products are not setup explicitly. Computational effort would be required for recovering the original solution vector, i.e., solving $\bar{x}(t) = (I \otimes P_2)x(t)$. This cost should be negligible given a "simple" choice of P_2, e.g., if P is a diagonal operator. Hence, the only computationally expensive part in this preconditioning would be setting up the preconditioner for the entire system in $\mathcal{H}$-Tucker format. In particular, this would require calls of the Black Box algorithm. It is expected that these costs are negligible in comparison to the entire adaptive routine, although this is not precisely clear at this point.

6.4. RICHARDSON ITERATIONS

In this subsection, we consider equations of the form

$$(I \otimes A + D \otimes M)x = f$$

as they arise in (6.5). As discussed in Subsection 6.2, the operator on the LHS is positive definite, which will allow to use Richardson iterations in order to approximately solve the infinite-dimensional problem. In this subsection, we check the availability of the routines required for the approximate iterations.

In the above equation, $I \in \mathbb{R}^{(J+1)\times(J+1)}$ is an identity matrix, $D \in \mathbb{R}^{(J+1)\times(J+1)}$ can be viewed as the matrix containing the transition intensities, i.e., this will be the part of the operator requiring $\mathcal{H}$-Tucker storage format. The operators A and M are bi-infinite matrices representing the wavelet part of the operator, and f is the RHS which will be usually obtained from the previous iteration. In order for this matrix-vector product to be implementable, as discussed in Subsection 6.1, we have to assume that x is given or can be factored as

$$x = x_1 \otimes x_2,$$

where x_1 is of $\mathcal{H}$-Tucker format compatible with D and x_2 is compatible with the wavelet matrices A and M. Then we can write the above equation as

$$(I \otimes A + D \otimes M)x = x_1 \otimes Ax_2 + Dx_1 \otimes Mx_2 = f,$$

which can be implemented as discussed in Subsection 6.1. Note that, strictly speaking, A and M are not matrices and thus, e.g., $I \otimes A$ is not a Kronecker product between matrices. More precisely, one should consider linear mappings

$$\begin{aligned} I &: \mathbb{R}^{(J+1)\times(J+1)} \to \mathbb{R}^{(J+1)\times(J+1)}, \\ A &: \ell_2(\Lambda) \to \ell_2(\Lambda), \\ I \otimes A &: \mathbb{R}^{(J+1)\times(J+1)} \otimes \ell_2(\Lambda) \to \mathbb{R}^{(J+1)\times(J+1)} \otimes \ell_2(\Lambda), \end{aligned}$$

where the latter is the unique mapping satisfying

$$(I \otimes A)(v_1 \otimes v_2) = Iv_1 \otimes Av_2,$$

and I, A are representations of the linear mappings w.r.t. the canonical bases. With an appropriate choice of basis, one can, loosely speaking, view $I \otimes A$ as the Kronecker product of I with a bi-infinite matrix A, keeping in mind that this is based upon the interpretation of $I \otimes A$ as a linear mapping. For more details, see, e.g., [36].

Firstly, we would at least require $(I \otimes A + D \otimes M)$ to be s^*-compressible. This is, however, a trivial consequence of s^*-compressiblity of A and M.

Proposition 6.1. *Let D be the intensity matrix as defined in* (4.4) *and A_ε an approximation to A such that*

$$\|A - A_\varepsilon\| \leq \varepsilon$$

and the number of nonzero entries in each row or column of A_ε is of order $\mathcal{O}(g(\varepsilon))$. Similarly define M_ε. Then

$$\|(I \otimes A + D \otimes M) - (I \otimes A_\varepsilon + D \otimes M_\varepsilon)\| \lesssim \varepsilon,$$

and the number of nonzero entries in each row or column is of order $\mathcal{O}(g(\varepsilon))$ as well.

Proof. By the discussion on the representation of $I \otimes A$ from above, we easily get

$$\|I \otimes A - I \otimes A_\varepsilon\| \leq (J+1)\|A - A_\varepsilon\| \leq (J+1)\varepsilon.$$

Since D is a finite matrix, we can set

$$C := \max\left\{\max_i \sum_j \tilde{d}_{i,j},\ \max_j \sum_i \tilde{d}_{i,j}\right\}$$

and obtain for the remaining term

$$\|D \otimes M - D \otimes M_\varepsilon\| \leq C(J+1)^2 \|M - M_\varepsilon\| \leq C(J+1)^2 \varepsilon.$$

By counting, the number of nonzero terms in each row or column of $I \otimes A_\varepsilon$ and $D \otimes M_\varepsilon$ can be bounded by some absolute multiple of $(J+1)g(\varepsilon)$. □

Note that we only require the error estimate in Proposition 6.1, since $I \otimes A_\varepsilon + D \otimes M_\varepsilon$ is not set up explicitly. Both for an iterative solver and for the approximate residuals, we require the availability of a matrix-vector product Bx that can be computed to any desired accuracy. Since the matrices representing the differential part are s^*-compressible, we can easily obtain such a routine by combining **APPLY** with the method described in Subsection 6.1. We will refer to this routine as **HT-APPLY**. Note that D is given in $\mathcal{H}$-Tucker format and the result of the performed computation is stored in $\mathcal{H}$-Tucker format as well, using operations permitted by the current state of software (see [41]). Given a nontrivial $x \neq 0$, we can state the routine as in Algorithm 6.1.

Algorithm 6.1 HT-APPLY

Input: $B := (I \otimes A + D \otimes M)$, $x = x_1 \otimes x_2 \in \ell_0(\Lambda)$, $\varepsilon > 0$
Output: w_ε with $\|Bx - w_\varepsilon\| \leq \varepsilon$

1: Compute $R_1 := Dx_1$ in HTucker format
2: Compute norms $\delta_1 := \|Dx_1\|$ and $\delta_2 := \|x_1\|$
3: Compute $R_2 := \mathbf{APPLY}\,[M; x_2; \varepsilon/2\delta_1]$
4: Compute $L_1 := \mathbf{APPLY}\,[A; x_2; \varepsilon/2\delta_2]$
5: Concatenate $L := x_1 \otimes L_1$ and $R := R_1 \otimes R_2$
6: **return** $L + R$

We collect some straightforward properties of Algorithm 6.1.

Proposition 6.2. *For an s^*-admissible A and M, the output of* **HT-APPLY** *satisfies*

$$\|Bx - w_\varepsilon\| \leq \varepsilon,$$
$$\#\operatorname{supp}(w_\varepsilon) \lesssim (J+1)\varepsilon^{-1/s}\|x_2\|_{\mathscr{A}^s}^{1/s}$$

for $x \in \ell_0(\Lambda)$ and any $s < s^*$. The number of arithmetic operations is bounded by an absolute multiple of

$$\varepsilon^{-1/s}\|x_2\|_{\mathscr{A}^s}^{1/s} + \#\operatorname{supp}(x_2) + 1 + g(D, x_1),$$

where $g(D, x_1)$ is given in [41, Lemma 5.9] by

$$g(D, x_1) := r^D r^{x_1} \sum_{k=1}^{d} m_k(2n_k - 1) + (d-1)(r^D)^3(r^{x_1})^3.$$

The parameters in the g are defined as follows:

- *d is the order of the $\mathcal{H}$-Tucker tensor used to store the vector x_1 and the row and column order of the $\mathcal{H}$-Tucker tensor used to store D.*
- *$r^D := \max_{t \in T_d} r_t^D$ and $r^{x_1} := \max_{t \in T_d} r_t^{x_1}$ are the maximal $\mathcal{H}$-Tucker ranks, where T_d is the $\mathcal{H}$-Tucker tree of D and x_1.*

Proof. For the error observe that

$$\|x_1 \otimes Ax_2 - x_1 \otimes L_1\|^2 = \left\| \begin{pmatrix} (x_1)_0(Ax_2 - L_1) \\ \vdots \\ (x_1)_J(Ax_2 - L_1) \end{pmatrix} \right\|^2 \leq \varepsilon^2/4.$$

Similarly

$$\|Dx_1 \otimes Mx_2 - Dx_1 \otimes R_2\|^2 \leq \varepsilon^2/4,$$

which gives the first claim. By the same arguments and using the bound from **APPLY**, we get the bound on $\#\operatorname{supp}(w_\varepsilon)$. Finally, counting the number of operations and considering that the number of operations required for the matrix-vector product in line 1 is $\mathcal{O}(g(D, x_1))$, we get the last claim. □

To make notation compact, let $B := (I \otimes A + D \otimes M)$. We have the necessary ingredients for the Richardson iteration of the form

$$x^{i+1} = x^i + \alpha(f - Bx^i).$$

This is a fixed point type iteration, where convergence is guaranteed by the Banach Fixed Point Theorem. Thus, convergence relies on the existence of an $\alpha \in \mathbb{R}$ with

$$\|I - \alpha B\| < 1,$$

where I is the identity operator. If B is positive definite, then this is guaranteed by, e.g., Lemma 3.2 for an appropriate choice of α. We require one last adjustment for the routine **COARSE** (see Algorithm 6.2) before we can formulate the inexact Richardson iteration.

Algorithm 6.2 HT-COARSE

Input: $x = x_1 \otimes x_2 \in \ell_0$, $x \neq 0$, $\varepsilon > 0$
Output: x_ε with $\|x - x_\varepsilon\| \leq \varepsilon$
return $x_\varepsilon := \mathbf{COARSE}\,[x_2; \varepsilon/\|x_1\|]$

Proposition 6.3. *The output of the routine* **HT-COARSE** *satisfies*

$$\|x - x_\varepsilon\| \leq \varepsilon,$$
$$\#\operatorname{supp}(x_\varepsilon) \lesssim (J+1)\min\left\{N : \|x_2 - x_2^N\| \leq \varepsilon\right\},$$

and the number of operations is of the order

$$\#\operatorname{supp}(x_2) + \max\left\{\log(\varepsilon^{-1}\|x_2\|), 1\right\}.$$

Proof. The proof follows directly from the properties of **COARSE**. □

Recall the RHS f contains a term of the form $I \otimes A + D \otimes M$ and $b \otimes F$. The former can be approximated to any desired accuracy by above. The remaining term can be viewed as

$$b \otimes F = \begin{pmatrix} b_0 F \\ \vdots \\ b_J F \end{pmatrix},$$

and hence, by the discussion in Subsection 5.2, can also be approximated to any desired accuracy. The adjustment to the routine **RHS** (see 6.3) are analogous to those above.

Algorithm 6.3 HT-RHS

Input: $\varepsilon > 0$, F
Output: f_ε with $\|b \otimes F - f_\varepsilon\| \leq \varepsilon$
  $F_\varepsilon := \mathbf{RHS}\,[F; \varepsilon/\|b\|]$
  Concatenate $f_\varepsilon := b \otimes F_\varepsilon$
  return f_ε

Once more, we collect the properties of this algorithm.

Proposition 6.4. *The output of* **HT-RHS** *satisfies*

$$\|b \otimes F - f_\varepsilon\| \leq \varepsilon.$$

If B is s^*-admissible and for $x = x_1 \otimes x_2$, $x_2 \in \mathscr{A}^s$ with $0 < s \leq s^*$, then

$$\#\operatorname{supp}(f_\varepsilon) \lesssim (J+1)\varepsilon^{-1/s}\|x_2\|_{\mathscr{A}^s},$$

and the number of operations by the call of **HT-RHS** *can be bounded by*

$$\varepsilon^{-1/s}\|x_2\|_{\mathscr{A}^s} + 1.$$

Proof. The proof follows by Proposition 3.4 and straight-forward computations. □

Remark 6.1. Note that, strictly speaking, approximating RHS involves not just F, but an application of **HT-APPLY** as well, thus we would apply half a given tolerance, i.e., $\varepsilon/2$ to both routines. Moreover, since **HT-RICH** would in general include a call to **HT-APPLY**, this would have to be accounted for in the arithmetic complexity.

Together with routines for the RHS and approximate matrix vector products, we can formulate the Richardson iteration.

For the convergence of the scheme we require that B is boundedly invertible and positive definite. This can be seen by the following lemma. We call an operator $B : \mathcal{L}(\ell_2(\Lambda), \ell_2(\Lambda))$ *coercive* if

$$(Bx, x)_{\ell_2} \geq \alpha\|x\|^2_{\ell_2}$$

for some $\alpha > 0$ and all $x \in \ell_2$.

Algorithm 6.4 HT-RICH

Input: $\varepsilon > 0$, $\varepsilon_0 \geq \|x\|$, $\theta \leq \frac{1}{2}$, $K \in \mathbb{N}$ and $\rho < 1$ s.t. $\|I - \alpha B\| \leq \rho$, $2\rho^K < \theta$
Output: x_ε with $\|x_\varepsilon - x\| \leq \varepsilon$

```
i := 0, x^0 := 0
while ε_i > ε do
    i := i + 1
    ε_i := 2ρ^K ε_{i−1}/θ
    y^{i,0} := x^{i−1}
    for j = 1 to K do
        δ := ρ^j ε_{i−1}/2αK
        y^{i,j} := y^{i,j−1} + α (HT-RHS[f; δ] − HT-APPLY[B; y^{i,j−1}; δ])
    end for
    x^i := HT-COARSE[(1 − θ)ε_i, y^{i,K}]
end while
return x_ε := x^i
```

Lemma 6.1. *Let Ψ be a wavelet basis satisfying the usual properties discussed in Subsection 2.2 with global smoothness parameter $r > \frac{n}{2} - 2$. Let $B := \mathcal{M} + \Delta t\theta\mathcal{A}$ be an operator arising in a semi-discrete problem* (6.3) *for a weakly coercive and continuous bilinear form $a(\cdot,\cdot)$. Furthermore, let*

$$\begin{cases} 0 \leq \Delta t\theta < \frac{c_\Psi}{\gamma C_\Psi - \alpha c_\Psi}, & \text{if } \gamma C_\Psi - \alpha c_\Psi > 0, \\ \Delta t\theta \geq 0, & \text{otherwise,} \end{cases}$$

where $\alpha > 0$ and $\gamma \geq 0$ are constants from (6.4)*; and C_Ψ, c_Ψ are upper and lower Riesz constants. Then B is bounded and coercive, i.e., in particular positive definite and boundedly invertible.*

Proof. First we show boundedness. The boundedness of $\mathcal{A}$ is directly implied by the continuity of the associated bilinear form. As for $\mathcal{M}$, we can apply the Schur Lemma, Lemma 3.1. Let $\Omega \subset \mathbb{R}^n$ and r denote the global smoothness of the underlying

wavelets. Take weights $\omega_\lambda := 1$. Furthermore, let

$$U(\lambda, \ell) := \{\mu \in \Lambda : |\mu| = \ell,\ (\psi_\lambda, \psi_\mu) \neq 0\}.$$

Note that $U(\lambda, \ell)$ contains only those indices on level ℓ that have intersecting support with λ. Furthermore, the wavelets are piecewise polynomial and, hence, due to vanishing moments, only those entries are nonzero, where the singular support of one wavelet intersects the support of the other wavelet ($i(\lambda, \mu) = 1$). This together with Assumption 2.6 implies $\#U(\lambda, \ell) \lesssim 2^{\max(\ell - |\lambda|, 0)(n-1)}$. By applying estimates (2.3) and (2.4), we get

$$\begin{aligned}\sum_{\mu \in \Lambda} (\psi_\lambda, \psi_\mu) = \sum_{\ell \in \mathbb{Z}} \sum_{\mu \in U(\lambda, \ell)} (\psi_\lambda, \psi_\mu) &\lesssim \sum_{\ell = |\lambda|}^{\infty} 2^{(\ell - |\lambda|)(n-1)} 2^{(\frac{n}{2} + s)(|\lambda| - \ell)} \\ &= \sum_{\ell = |\lambda|}^{\infty} 2^{(\ell - |\lambda|)\left(\frac{n}{2} - (1+s)\right)} \lesssim 1.\end{aligned}$$

for any $\frac{n}{2} - 1 < s \leq r + 1$. By symmetry, we get the same for the row sum. Hence, $\|\mathcal{M}\| \lesssim 1$.

We now show coercivity. Let $\gamma \geq 0$ and $\alpha > 0$ be the constants from (6.4). Then for any $x \in \ell_2(\Lambda)$,

$$\begin{aligned}(\mathcal{A}x, x) = a(x^T\Psi, x^T\Psi) &\geq \alpha \|x^T\Psi\|_V^2 - \gamma \|x^T\Psi\|_{L_2}^2 \\ &\geq \alpha \|x^T\Psi\|_{L_2}^2 - \gamma \|x^T\Psi\|_{L_2}^2 \geq \alpha c_\Psi \|x\|_{\ell_2}^2 - \gamma C_\Psi \|x\|_{\ell_2}^2,\end{aligned}$$

where c_Ψ and C_Ψ are lower and upper Riesz constants. Similarly for $\mathcal{M}$,

$$(\mathcal{M}x, x) = \|x^T\Psi\|_{L_2}^2 \geq c_\Psi \|x\|_{\ell_2}^2,$$

and thus for the operator B,

$$(Bx, x) \geq \|x\|^2(c_\Psi + \Delta t\theta(\alpha c_\Psi - \gamma C_\Psi)).$$

For $\alpha c_\Psi - \gamma C_\Psi < 0$, we get the restriction on the step-size

$$c_\Psi + \Delta t\theta(\alpha c_\Psi - \gamma C_\Psi) > 0,$$

i.e,

$$0 \leq \Delta t\theta < \frac{c_\Psi}{\gamma C_\Psi - \alpha c_\Psi}.$$

This completes the proof. □

Remark 6.2. For notational simplicity we considered here a general wavelet basis on $\Omega \subset \mathbb{R}^n$. For tensor-product wavelet bases, we get more specific estimates, and the proof is analogous to the one above.

Proposition 6.5. *Let* $B_S := \frac{1}{2}(B + B^T)$ *and* $\alpha \in (0, 1/(\|B_S\| + \|B_S^{-1}\|^{-1})]$, $\alpha < 2/(\|B_S\|\|B_S^{-1}\|)$. *For* $x_2 \in \mathscr{A}^s$ *with some* $s > 0$ *and an* $\bar{s}$*-admissible* B *with* $s \leq \bar{s}$, *the output of* **HT-RICH** *satisfies*

$$\|x - x_\varepsilon\| \leq \varepsilon,$$
$$\#\operatorname{supp}(x_\varepsilon) \lesssim (J+1)\varepsilon^{-1}\|x_2\|_{\mathscr{A}^s},$$

and the number of operations is of the order

$$\varepsilon^{-1/s}\|x_2\|_{\mathscr{A}^s}^{1/s} + g(D, x_1),$$

where $g(D, x_1)$ *is defined in Proposition 6.2.*

Proof. For the error bound apply, Lemma 6.1. For the support estimate and computational complexity, apply Propositions 6.3 and 6.2. The remainder of the proof is as in [46, Theorem 3.1]. □

Summing up, in this subsection we proposed a scheme for an adaptive solving procedure based on a Richardson iteration for a PDE problem of the form (4.1). The routine converges but is not (quasi-)optimal in the classical sense. This is due to the $\mathcal{H}$-Tucker part D and x_1. Computational complexity in general depends on the size, and, more importantly, the storage structure. Unlike the size of the input vector, the storage structure is, in general, independent of the approximation properties in the sense of the space $\mathscr{A}^s$, and, hence, cannot be estimated from above by the $\mathscr{A}^s$ norm. Moreover, taking into account the time-discretization, (quasi-)optimality cannot be guaranteed for the entire fully discrete problem regardless of the $\mathcal{H}$-Tucker part.

6.5. SEMI-DISCRETE AWGM

In this subsection, we propose an **AWGM** method to solve the semi-discrete problem at each time step. Although this routine will have similar convergence and complexity properties as **HT-RICH**, we expect it to perform quantitatively better since it does not require coarsening of the iterands. On the other hand, Richardson iterations apply to a wider range of operator equations, while **AWGM** applies solely to the elliptic case, i.e., s.p.d. operators arising in Galerkin problems.

Essential for the convergence analysis is the availability of an equivalent energy norm with some properties. Note that for a norm, it suffices when the operator B is positive definite and, hence, also invertible. However, to obtain the necessary norm equivalences as in [46, Section 4], we require more than positive definiteness. At the same time, closer inspection of the convergence analysis shows that symmetry is not necessary for the real case. We summarize the required assumptions and norm

equivalences in the following. Note that as before, we always implicitly assume the underlying field is $\mathbb{K} = \mathbb{R}$.

Lemma 6.2. *Let $B : \mathcal{L}(\ell_2(\Lambda), \ell_2(\Lambda))$ be a coercive operator. Define the bilinear form $b(\cdot, \cdot)$ on $\ell_2(\Lambda)$ by*

$$b(x, y) := (Bx, y)_{\ell_2},$$

and $|||x||| := \sqrt{b(x, x)}$. Then the following equivalences hold for all $x \in \ell_2(\Lambda)$

(i) $\|B^{-1}\|^{-1}\|x\| \leq \|Bx\| \leq \|B\|\|x\|$.

(ii) $\|B^{-1}\|^{-1/2}\|x\| \leq |||x||| \leq \|B\|^{1/2}\|x\|$.

(iii) $\|B^{-1}\|^{-1/2}|||x||| \leq \|Bx\| \leq \|B\|^{1/2}|||x|||$.

Proof. (i) Using Lax–Milgram for nonsymmetric coercive bilinear forms, we get that B is invertible and the inverse is bounded by $\|B^{-1}\| \leq \frac{1}{\alpha}$, where α is the coercivity constant. This is equivalent to B being bounded from below since

$$\|x\| = \|B^{-1}Bx\| \leq \|B^{-1}\|\|Bx\|,$$

which gives the LHS of the inequality. The RHS is a trivial consequence of the definition of the operator norm $\|B\|$.

(ii) Define α^* by

$$\alpha^* := \sup\left\{\alpha > 0 : (Bx, x) \geq \alpha\|x\|^2,\ x \in \ell_2(\Lambda)\right\}.$$

Since B is coercive, clearly $0 < \alpha^* < \infty$. From (i), we have $\|B^{-1}\| \leq 1/\alpha^*$. By definition of α^* and, applying same arguments as in Lax–Milgram, $\alpha^* \geq$

$\|B^{-1}\|^{-1}$. This gives us the LHS of the inequality

$$|||x|||^2 = (Bx, x) \geq \|B^{-1}\|^{-1}\|x\|^2.$$

The RHS is obtained by a simple application of Cauchy–Schwarz.

(iii) The inverse B^{-1} is coercive as well, since

$$(B^{-1}x, x) = (B^{-1}By, By) = (y, By) \gtrsim \|y\|^2 = \|B^{-1}x\|^2 \gtrsim \|x\|^2.$$

Hence, by applying similar arguments as in (ii), we get

$$|||x|||^2 = (x, Bx) = (B^{-1}Bx, Bx) \gtrsim \|B\|^{-1}\|Bx\|^2,$$

which gives the RHS of the inequality. The LHS follows again by an application of Cauchy–Schwarz to the above.

The proof is complete. □

Remark 6.3. If the bilinear form $b(\cdot, \cdot)$ is not symmetric, then $|||\cdot|||$ is not a norm, since only a "relaxed" version of the triangle inequality holds, i.e.,

$$|||x + y||| \leq \sqrt{\kappa(B)}(|||x||| + |||y|||),$$

which follows from the proven equivalences. Symmetry is crucial for the Cauchy–Schwarz inequality for the associated inner product $b(\cdot, \cdot)$. Obviously, other norm axioms hold even for nonsymmetric $b(\cdot, \cdot)$. Moreover, inspecting the convergence analysis in [46, Section 4], we do not require symmetry.

$B_{\mathcal{J}}$ inherits the properties of B and is in particular invertible and positive definite. Note that $\|R_{\mathcal{J}}x\| \leq \|x\|$ where the bound is sharp, and $\|E_{\mathcal{J}}x\| = \|x\|$, i.e., $\|R_{\mathcal{J}}\| = \|E_{\mathcal{J}}\| = 1$. We infer $\|B_{\mathcal{J}}\| \leq \|B\|$ and $\|B_{\mathcal{J}}^{-1}\| \leq \|B^{-1}\|$. This, in particular,

implies that the condition number $\kappa(B_{\mathcal{J}}) = \|B_{\mathcal{J}}\|\|B_{\mathcal{J}}^{-1}\|$ is uniformly bounded by $\kappa(B)$.

Moreover, $\|B_{\mathcal{J}}\| = \|B\|$ on $\ell_2(\mathcal{J})$, i.e., if we restrict B to elements in $\ell_2(\Lambda)$ with support in $\mathcal{J}$, then, the norm equivalences from Lemma 6.2 apply to $B_{\mathcal{J}}$ as well.

We have the required tools to formulate a solving procedure based on an approximate residual evaluation. Note that since we are considering a system of PDEs, the procedure **EXPAND** has to be slightly modified. For a given finite index set Λ_i, the residual will have the structure

$$r = \begin{pmatrix} r_0 \\ \vdots \\ r_J \end{pmatrix}.$$

This suggests a slight modification of **EXPAND** (see Algorithm 6.5).

Algorithm 6.5 SYS-EXPAND

Input: Finite $\Lambda_i \subset \Lambda$, $r \in \ell_0(\Lambda)$, $\alpha \in [0,1]$
Output: $\Lambda_{i+1} \supset \Lambda_i$

1: **for** $k = 0$ to J **do**
2: $\quad \bar{r}_k := \textbf{COARSE}\left[r|_{\Lambda\setminus\Lambda_i}; \sqrt{1-\alpha^2}\|r_k\|\right]$
3: **end for**
4: **return** $\Lambda_{i+1} := \Lambda_i \cup \bigcup_{k=1}^{J} \operatorname{supp}(\bar{r}_k)$

Similar to [46, Proposition 4.3], one can show that this **SYS-EXPAND** captures the bulk of the residuum.

Proposition 6.6. $\Lambda_i \subset \Lambda_{i+1} :=$**SYS-EXPAND**$[\Lambda_i; r; \alpha]$ *satisfies* $\|R_{\Lambda_{i+1}} r\| \geq \alpha\|r\|$ *and*

$$\#(\Lambda_{i+1} \setminus \Lambda_i) \lesssim \min\left\{\#(\bar{\Lambda} \setminus \Lambda_i) : \|R_{\bar{\Lambda}} r\| \geq \alpha\|r\|\right\},$$

where the number of operations is of the order $\mathcal{O}(\#\Lambda_i + \#\operatorname{supp}(r) + 1)$.

Proof. Let $\bar{r}$ be defined by

$$\bar{r} := \begin{pmatrix} \bar{r}_0 \\ \vdots \\ \bar{r}_J \end{pmatrix}.$$

Then

$$\begin{aligned} \|r - R_{\Lambda_{i+1}} r\|^2 = \| \, r|_{\Lambda\setminus\Lambda_{i+1}} \, \|^2 \leq \| \, r|_{\Lambda\setminus\Lambda_i} - \bar{r}\|^2 &= \sum_{k=1}^{J} (1-\alpha^2)\|r_k\|^2 \\ &= (1-\alpha^2)\|r\|^2. \end{aligned}$$

This is, however, equivalent to $\|r - R_{\Lambda_{i+1}} r\|^2 \leq (1-\alpha^2)\|r\|^2$, i.e.,

$$\begin{aligned} \alpha^2\|r\|^2 &\leq \|r\|^2 - \|r - R_{\Lambda_{i+1}} r\|^2 = \|r\|^2 - \|r_{\Lambda\setminus\Lambda_{i+1}}\|^2 \\ &= \sum_{k=1}^{J}\sum_{\lambda\in\Lambda}(r_k)_\lambda^2 - \sum_{k=1}^{J}\sum_{\lambda\in\Lambda\setminus\Lambda_{i+1}}(r_k)_\lambda^2 = \sum_{k=1}^{J}\sum_{\lambda\in\Lambda_{i+1}}(r_k)_\lambda^2 = \|R_{\Lambda_{i+1}} r\|^2, \end{aligned}$$

which gives the first claim.

The arithmetic complexity is implied by the properties of the routine **COARSE**, being the only computationally expensive part of **SYS-EXPAND**. Minimality is implied by **COARSE** by observing

$$\begin{aligned} \#(\Lambda_{i+1}\setminus\Lambda_i) &= \#\operatorname{supp}(\bar{r}) \\ &\lesssim \min\left\{\#\bar{\Lambda} : \bar{\Lambda}\in\Lambda\setminus\Lambda_i,\ \| \, r|_{\Lambda\setminus\Lambda_i} - R_{\bar{\Lambda}}(r|_{\Lambda\setminus\Lambda_i})\| \leq \sqrt{1-\alpha^2}\|r\|\right\} \\ &= \min\left\{\#\bar{\Lambda} : \bar{\Lambda}\in\Lambda\setminus\Lambda_i,\ \|R_{\Lambda_i\cup\bar{\Lambda}} r\| \geq \alpha\|r\|\right\}, \end{aligned}$$

where for the last equality we used the same arguments as above. □

Next, after having refined from Λ_i to Λ_{i+1}, we need to solve the system on Λ_{i+1}. The step corresponds to **GALERKIN** in Algorithm 3.2. As discussed in Subsection 3.5, the (idealized) Galerkin solution on Λ_{i+1} corresponds to the best approximation to the solutions on Λ_{i+1} and w.r.t. the energy (pseudo-) norm $|||\cdot|||$. To compute an approximate Galerkin solution, we will use an iterative solver, e.g., BiCGSTAB as in the work of [41]. In general, given an appropriate iterative solver, we assume the availability of a **GALERKIN** routine that, for a given target tolerance ε and an initial guess $x^0_{\Lambda_{i+1}}$ with $\|R_{\Lambda_{i+1}} f - B_{\Lambda_{i+1}} x^0_{\Lambda_{i+1}}\| \leq \delta$, produces a Galerkin solution on Λ_{i+1} with the property

$$\|R_{\Lambda_{i+1}} f - B_{\Lambda_{i+1}} x_{\Lambda_{i+1}}\| \leq \varepsilon.$$

For instance, Galerkin can be implemented applying Richardson iterations (cf. Subsection 6.4 and [46, Section 4.2]). In that case, the cost of one call can be bounded by

$$\eta(\delta/\varepsilon)(\delta^{-1/s}\|x_2\|^{1/s}_{\mathscr{A}^s} + \delta^{-1/s}\|(x_2)_{\Lambda_i}\| + \#\Lambda_{i+1} + 1 + g(D, x_1)),$$

where δ is the initial error, ε is the target accuracy, $g(D, x_1)$ is defined in Proposition 6.2 and $\eta \geq 1$ is a nondecreasing function. Alternatively, a more efficient routine can be constructed by using a *defect correction* principle. See [46, Section 4.2] for more details.

We can formulate the **HT-AWGM** procedure. Let F denote here the entire RHS.

Algorithm 6.6 HT-AWGM

Input: $\varepsilon, \varepsilon_{-1} > 0$, $\alpha, \delta, \gamma, \theta$ with $\delta \in (0, \alpha)$, $\frac{\alpha+\delta}{1-\delta} < \kappa(B)^{-1/2}$, $\theta > 0$ and $\gamma \in (0, \frac{(1-\delta)(\alpha-\delta)}{1+\delta}\kappa(B)^{-1})$

Output: x_ε with $\|f - Bx_\varepsilon\| \leq \varepsilon$

$i := 0, x^i := 0, \Lambda_i := \emptyset$

loop

$\zeta := \theta\varepsilon_{i-1}$

repeat

$\zeta := \zeta/2$, $r^i :=$ **HT-RHS**$[f; \zeta/2]$ − **HT-APPLY**$[B; x^i; \zeta/2]$

if $\varepsilon_i := \|r^i\| + \zeta \leq \varepsilon$ **then**

$x_\varepsilon := x^i$ stop

end if

until $\zeta \leq \delta\|r^i\|$

$\Lambda_{i+1} :=$ **SYS-EXPAND**$[\Lambda_i; r^i; \alpha]$

$x^{i+1} :=$ **GALERKIN**$[\Lambda_{i+1}; x^i; \varepsilon_i; \gamma\|r^i\|]$

end loop

return $x_\varepsilon := x^{i+1}$

Using norm equivalence (i) from Lemma 6.2, we get

$$\|B^{-1}\|^{-1}\|x - x_\varepsilon\| \leq \|f - Bx_\varepsilon\|.$$

Using (ii), we get

$$\|x - x_\varepsilon\| \geq \|B\|^{-1/2}|||x - x_\varepsilon|||,$$

and, using (iii), we get

$$\|x - x_\varepsilon\| \geq \|B\|^{-1}\|f - Bx_\varepsilon\|.$$

Overall

$$\|B\|^{-1}\|f - Bx_\varepsilon\| \leq \|x - x_\varepsilon\| \leq \|B^{-1}\|\|f - Bx_\varepsilon\|.$$

This justifies using the residuum as an error estimator.

Proposition 6.7. *The output of* **HT-AWGM** *satisfies*

$$\|f - Bx_\varepsilon\| \leq \varepsilon.$$

If $x_2 \in \mathscr{A}^s$ *for some* $s > 0$*, then*

$$\#\operatorname{supp}(x_\varepsilon) \lesssim (J+1)\varepsilon^{-1/s}\|x_2\|_{\mathscr{A}^s}.$$

If B *is* s^**-admissible for* $s^* \geq s$ *and* $\varepsilon \lesssim \varepsilon_{-1} \sim \|f\|$*, then the number of operations for a call of* **HT-AWGM** *can be bounded by*

$$\varepsilon^{-1/s}\|x_2\|_{\mathscr{A}^s}^{1/s} + g(D, x_1),$$

where $g(D, x_1)$ *is defined in Proposition 6.2.*

Proof. Apply [46, Theorem 4.1] with Propositions from this subsection and Subsection 6.4. □

Summing up, as in Subsection 6.4, we get a convergent adaptive routine with the same properties but better (expected) quantitative performance than **HT-RICH**. For a fixed point in time, the routine is still not (quasi-)optimal, even though rather close

to it. Overall, the routine for the entire fully discrete problem is not (quasi-)optimal regardless of the $\mathcal{H}$-Tucker part.

6.6. SPACE-TIME APPROACH

In this subsection, we consider solving the PDE by first considering the variational formulation in space-time. The advantage of this approach is that due to the work in [43], as opposed to time stepping, we get (quasi-)optimality. More precisely, this will get us the "closest" to (quasi-)optimality, i.e., if we ignore the $\mathcal{H}$-Tucker part in the complexity estimates.

The weak space-time form of the PDE in (4.1) is obtained by further integrating the weak space-form over time and including the terminal condition

$$\begin{aligned} b(u,v) &:= \int_0^T [(u_t(t), v_1)_{L_2} + a(u(t), v_1)]\mathrm{d}t + (u(T), v_2)_{L_2} = f(v), \\ f(v) &:= \int_0^T (f(t), v_1)_{L_2}\mathrm{d}t + (u_T, v_2)_{L_2}. \end{aligned}$$

Note that here, for vector-valued functions, the inner product is defined as a sum of the vector components, i.e.,

$$(u,v)_{L_2} = \sum_{j=1}^{J+1} (u^j, v^j)_{L_2}.$$

The above is just one possibility to include the terminal or initial condition (see, e.g., [44]). For instance, similar to the boundary conditions, one could choose the homogenization approach and concentrate on solving equations with homogeneous terminal/initial conditions. This approach can be viewed as posing unnecessary restrictions on the terminal/initial functions. Another approach would be to apply partial integration w.r.t. time and apply test functions that vanish on one of the boundaries, e.g., at $t = 0$ for terminal conditions and $t = T$ for initial conditions. In

the above, as in the work of [43], multipliers are used to incorporate the terminal into the equation.

Recall that in the derivation of the weak form in Subsection 4.1, it was assumed that the coefficients admit to an affine decomposition w.r.t. space and time, i.e.,

$$d^{j,k}(t,y) = \tilde{d}^{j,k}(t)h_d(y),$$

and analogously for the other coefficients. This assumption is not necessary from a theoretical point of view, i.e., the problem is well posed independent of the assumption above. However, it is crucial for splitting the $\mathcal{H}$-Tucker part from the wavelet part, which itself is crucial for an implementable numerical solution. It is thus not surprising that in the weak space-time formulation we require the coefficients to uniformly depend on space and time, i.e.

$$d^{j,k}(t,y) = \tilde{d}^{j,k}h_d(t,y),$$

and analogously for the other coefficients.

For convenience, we summarize all results and derivations in the following.

Proposition 6.8. *The weak space-time formulation of* (4.1) *reads*

$$\begin{aligned} b(u,v) &= f(v), \\ b(u,v) &:= \int_0^T [(u_t(t),v_1)_{L_2} + a(u(t),v_1)]\mathrm{d}t + (u(T),v_2)_{L_2}, \\ f(v) &:= \int_0^T (f(t),v_1)_{L_2}\mathrm{d}t + (u_T,v_2)_{L_2}, \end{aligned} \tag{6.7}$$

or, as an operator equation

$$\begin{aligned} &Bu = f, \\ &B \in \mathcal{L}(\mathscr{X}, \mathscr{Y}'),\ (Bu)(v) = b(u,v), \\ &f \in \mathscr{Y}'. \end{aligned}$$

The trial space is

$$\mathscr{X} := L_2(0,T;(H_0^1(\Omega))^{J+1}) \cap H^1(0,T;(L_2(\Omega))^{J+1}),$$

and the test space is

$$\mathscr{Y} := \mathscr{X} \times (L_2(\Omega))^{J+1}.$$

For (6.7)*, we have the following:*

(i) The operator B is boundedly invertible.

(ii) Let $\Psi^{\mathscr{X}} := \{\sigma_\xi : \xi \in \Xi\}$ be a Riesz basis for $L_2(0,T;H_0^1(\Omega)) \cap H^1(0,T;L_2(\Omega))$ and $\Psi^{L_2} := \{\psi_\lambda : \lambda \in \Lambda\}$ a Riesz basis for $L_2(\Omega)$. The equivalent ℓ_2-sequence space problem reads

$$\boldsymbol{B}\boldsymbol{u} = \boldsymbol{f},$$

where the operator $\boldsymbol{B} \in \mathcal{L}(\ell_2(\Lambda^\Xi), \ell_2(\Lambda^{\Xi\times\Lambda}))$ is boundedly invertible and can be viewed as

$$\boldsymbol{B} := \begin{pmatrix} \boldsymbol{B}_1 \\ \boldsymbol{B}_2 \end{pmatrix},$$

$$
\begin{aligned}
\boldsymbol{B}_1 &:= I \otimes \boldsymbol{M}_1 - [I \otimes \boldsymbol{A} - \boldsymbol{N}]\,,\ I \in \mathbb{R}^{(J+1)\times(J+1)},\\
\boldsymbol{M}_1 &:= \left(\int_0^T \left(D^t \sigma_\nu(t), \sigma_\xi(t) \right)_{L_2} \mathrm{d}t \right)_{\xi,\nu\in\Xi},\\
\boldsymbol{A} &:= \left(\int_0^T (r(t)\sigma_\nu(t), \sigma_\xi(t))_{L_2} \mathrm{d}t - \sum_{i=1}^{n} \int_0^T (\alpha(t))_i (D^i \sigma_\nu(t), \sigma_\xi(t))_{L_2} \mathrm{d}t, \right.\\
&\quad \left. + \frac{1}{2} \sum_{i,j=1}^{n} \int_0^T (\beta(t)\beta^T(t))_{i,j}) (D^i \sigma_\nu(t), D^j \sigma_\xi(t))_{L_2} \mathrm{d}t \right)_{\xi,\nu\in\Xi},\\
\boldsymbol{N} &:= (\boldsymbol{N}_{i,j})_{i,j\in\mathcal{J}}\,,\\
\boldsymbol{N}_{i,j} &:= \begin{cases} \boldsymbol{M}_2^{i,j} & \text{if } i \neq j, \\ -\sum_{k:k\neq i} \boldsymbol{M}_2^{i,k} & \text{otherwise,} \end{cases}\\
\boldsymbol{M}_2^{i,j} &:= \left(\int_0^T (d^{i,j}\sigma_\nu(t), \sigma_\xi(t))_{L_2} \mathrm{d}t \right)_{\xi,\nu\in\Xi},\\
\boldsymbol{B}_2 &:= I \otimes \boldsymbol{C},\ I \in \mathbb{R}^{(J+1)\times(J+1)},\\
\boldsymbol{C} &:= ((\sigma_\xi(T), \psi_\lambda)_{L_2})_{\lambda\in\Lambda,\xi\in\Xi}\,,\\
\boldsymbol{f} &:= \begin{pmatrix} \boldsymbol{f}_1 - \boldsymbol{f}_2 \\ \boldsymbol{f}_3 \end{pmatrix},\\
\boldsymbol{f}_1 &:= \left(-\sum_{k:k\neq j} \left(\int_0^T (d^{j,k}(t) a^{j,k}(t), \sigma_\xi)_{L_2} \mathrm{d}t \right)_{\xi\in\Xi} \right)_{j\in\mathcal{J}},\\
\boldsymbol{f}_2 &:= \left(\left(\int_0^T (c^j(t), \sigma_\xi(t))_{L_2} \mathrm{d}t \right)_{\xi\in\Xi} \right)_{j\in\mathcal{J}},\\
\boldsymbol{f}_3 &:= \left(\left((u_T^j, \psi_\lambda)_{L_2} \right)_{\lambda\in\Lambda} \right)_{j\in\mathcal{J}}.
\end{aligned}
$$

The solution vector should be interpreted as

$$
\boldsymbol{u} = \begin{pmatrix} \boldsymbol{u}^0 \\ \vdots \\ \boldsymbol{u}^{J+1} \end{pmatrix}, \quad u = \boldsymbol{u}^T \Psi^{\mathscr{X}}.
$$

(iii) If the coefficients are affine in space and time, i.e., Assumption 4.1 holds, then, keeping the notation as in (4.4)*, the simplifications*

$$\boldsymbol{N} = \int_0^T D(t) \otimes M_2 \mathrm{d}t, \quad \boldsymbol{f}_1 = \int_0^T b(t) \otimes F_1 \mathrm{d}t,$$
$$\boldsymbol{f}_2 = \int_0^T \tilde{c}(t) \otimes F_2 \mathrm{d}t, \qquad \boldsymbol{f}_3 = \tilde{a} \otimes ((h_T, \psi_\Lambda)_{L_2})_{\lambda \in \Lambda}$$

can be made, where $\int_0^T \mathrm{d}t$ *should be interpreted component-wise.*

(iv) Finally, if we assume

$$d^{j,k}(t,y) = \tilde{d}^{j,k} h_d(t,y),$$
$$a^{j,k}(t,y) = \tilde{a}^{j,k} h_a(t,y),$$
$$c^j(t,y) = \tilde{c}^j h_c(t,y),$$

then we can simplify

$$\boldsymbol{N} = D \otimes \boldsymbol{M}_2, \quad \boldsymbol{M}_2 := \left(\int_0^T (h_d \sigma_\nu, \sigma_\xi)_{L_2} \right)_{\xi,\nu \in \Xi},$$
$$\boldsymbol{f}_1 = b \otimes \boldsymbol{F}_1, \qquad \boldsymbol{F}_1 := \left(\int_0^T (h_d h_a, \sigma_\xi)_{L_2} \right)_{\xi \in \Xi},$$
$$\boldsymbol{f}_2 = \tilde{c} \otimes \boldsymbol{F}_2, \qquad \boldsymbol{F}_2 := \left(\int_0^T (h_c, \sigma_\xi) \right)_{\xi \in \Xi}.$$

Proof. For (i) see [43, Theorem 5.1].

The operator $\boldsymbol{B}$ is boundedly invertible by the arguments in Subsection 3.2. As a basis for the vector-valued function in a Bochner Space u, we consider the canonical basis, i.e., the j-th canonical vectors are $e_\xi^j := (\sigma_\xi \delta_{i,j})_{i \in \mathcal{J}}$. Expand each u^j as $u^j = \sum_{\nu \in \Xi} c_\nu^j \sigma_\nu$ and multiply by a test function σ_ξ. Hence, for each $j \in \mathcal{J}$, we get

by integrating over time

$$\begin{aligned}\sum_{\nu\in\Xi} c_\nu^j \Bigg[&\int_0^T (D^t\sigma_\nu(t),\sigma_\xi(t))_{L_2}\mathrm{d}t - \int_0^T (r(t)\sigma_\nu(t),\sigma_\xi(t)_{L_2}\mathrm{d}t \\ &+ \sum_{k=1}^n \int_0^T (\alpha(t))_k (D^k\sigma_\nu(t),\sigma_\xi(t))_{L_2}\mathrm{d}t \\ &- \frac{1}{2}\sum_{i,k=1}^n \int_0^T (\beta(t)\beta(t)^T)_{i,j}(D^i\sigma_\nu(t),D^k\sigma_\xi(t))_{L_2}\mathrm{d}t\Bigg] \\ &+ \sum_{k:k\neq j}\sum_{\nu\in\Xi}(c_\nu^k - c_\nu^j)\int_0^T (d^{j,k}(t)\sigma_\nu(t),\sigma_\xi(t))_{L_2}\mathrm{d}t \\ &= -\sum_{k:k\neq j}\int_0^T (d^{j,k}(t)a^{j,k}(t),\sigma_\xi(t))_{L_2}\mathrm{d}t - \int_0^T (c^j(t),\sigma_\xi(t))_{L_2}\mathrm{d}t.\end{aligned}$$

Moreover, note that by Lemma 2.3,

$$I\otimes \boldsymbol{A}\boldsymbol{u} = \begin{pmatrix} \boldsymbol{A}\boldsymbol{u}^0 \\ \vdots \\ \boldsymbol{A}\boldsymbol{u}^J \end{pmatrix}.$$

By the same lemma

$$\boldsymbol{N}\boldsymbol{u} = \left(\sum_{k=1}^J \boldsymbol{N}_{j,k}\boldsymbol{u}^k\right)_{j\in\mathcal{J}} = \left(\sum_{k:k\neq j} M_2^{j,k}(\boldsymbol{u}^k - \boldsymbol{u}^j)\right)_{j\in\mathcal{J}},$$

which proves the representation of $\boldsymbol{B}_1$, $\boldsymbol{f}_1$ and $\boldsymbol{f}_2$. The representation of the incorporated terminal condition follows analogously.

Parts (iii) and (iv) are simply special cases of the above. □

As basis for the Bochner Space $\mathscr{X}$ for our purposes, we can utilize the tensor product basis as in [43], where the best possible rate $s_{\max}$ is derived as well. In Proposition 6.8, it can be seen that the only implementable case is (iv). Since now we have an operator that is not s.p.d., different test and trial spaces, i.e., a *Petrov-*

Galerkin problem, and different index sets for the right and left-hand sides, we cannot apply **AWGM** directly anymore (see, e.g., [31, Section 6.2]). A remedy would be to apply **AWGM** to the normal equation

$$\boldsymbol{B}^T\boldsymbol{B}\boldsymbol{u} = \boldsymbol{B}^T\boldsymbol{f}.$$

Following the work of [43], it can be shown that $\boldsymbol{B}^T\boldsymbol{B}$ is s^*-compressible and the **AWGM** applied to the normal equation is (quasi-)optimal. More precisely, for our problem, we get "close" to (quasi-)optimality if $\boldsymbol{u} = \boldsymbol{u}_1 \otimes \boldsymbol{u}_2$ with $\boldsymbol{u}_2 \in \mathscr{A}^s$. In fact, for case (iv), the solution is of a tensor product form. Generally, consider the operator equation

$$\boldsymbol{B}\boldsymbol{u} = (\boldsymbol{B}_1 \otimes \boldsymbol{B}_2)\boldsymbol{u} = \boldsymbol{f} = \boldsymbol{f}_1 \otimes \boldsymbol{f}_2,$$

where B is a bounded, linear and invertible operator and $\boldsymbol{B}_i$ is compatible with $\boldsymbol{f}_i$ for $i \in \{1, 2\}$. Then $\boldsymbol{B}^{-1} = \boldsymbol{B}_1^{-1} \otimes \boldsymbol{B}_2^{-1}$ and thus

$$\boldsymbol{u} = \boldsymbol{B}_1^{-1}\boldsymbol{f}_1 \otimes \boldsymbol{B}_2^{-1}\boldsymbol{f}_2 =: \boldsymbol{u}_1 \otimes \boldsymbol{u}_2.$$

We conclude this section by summarizing the discussion above.

Proposition 6.9. *The output of* **HT-AWGM** *applied to the normal equations satisfies*

$$\|\boldsymbol{f} - \boldsymbol{B}\boldsymbol{u}_\varepsilon\| \leq \varepsilon.$$

If $\boldsymbol{u}_2 \in \mathscr{A}^s$*, then*

$$\#\operatorname{supp}(\boldsymbol{u}_\varepsilon) \lesssim (J+1)\varepsilon^{-1/s}\|\boldsymbol{u}_2\|_{\mathscr{A}^s}^{1/s}.$$

If additionally $\boldsymbol{B}$ *is* s^**-admissible for* $s^* \geq s$*, then the number of arithmetic operations is of the order*

$$\mathcal{O}(\varepsilon^{-1/s}\|\boldsymbol{u}_2\|_{\mathscr{A}^s}^{1/s} + g(D, \boldsymbol{u}_1)).$$

Proof. See [43, Theorem 4.13] and use similar arguments as in Subsection 6.5. □

7. CONCLUSIONS

In this thesis, results from [41] on solving a coupled system of parabolic PDEs with a "high-dimensional" structure were extended to an adaptive setting. Given an appropriate wavelet basis, the differential operators were verified to have the expected compression properties which allows us to approximate them by sparse matrices. This itself is a necessary requirement for an adaptive method. First, two adaptive methods were proposed for the operator equation arising from a semi-discrete problem by the method of lines. For the particular PDE system from the CDO model, both methods were shown to converge and have same complexity properties. **HT-AWGM** is expected to perform quantitatively better since it does not require coarsening of the iterands. However, **HT-RICH** applies to a wider range of operators and must be considered for more general problems. Both methods rely on the assumption that the coefficients of the PDE admit to an affine decomposition in time and space. (Quasi-)optimality is not guaranteed for any of the two methods, due to the nature of time stepping and the $\mathcal{H}$-Tucker structure. Finally, a third approach based on the weak space time formulation was proposed. Applying known results, this approach was shown to converge and to be the closest one being (quasi-)optimal out of the three suggested routines. (Quasi-)optimality is not guaranteed, since computational effort generally depends on the structure of the $\mathcal{H}$-Tucker storage format which in itself is independent of the input vector. This approach relies on the assumption that the PDE coefficients depend uniformly on space and time.

This work provides merely an initial analysis of solving a coupled system of PDEs in $\mathcal{H}$-Tucker format. For future research, numerical testing has to be performed in order to verify and improve the results presented here. For instance, since the differential operator consists of parts that require different preconditioning, the system

is preconditioned twice. It would be of interest to investigate more efficient methods of preconditioning. Another issue, that was not investigated here, is the interaction of tensor truncations with the adaptive routine. It is unclear at this point whether and how the truncation error can be controlled such that the overall approximation accuracy is ensured. Moreover, sharp estimates for the operator norms of discretized differential operators are required for an efficient implementation of the presented routines.

A slightly different problem would be to develop a format for the approximate application of general differential operators in $\mathcal{H}$-Tucker format, i.e., in particular of nontensor form. Due to the nature of the format, it is unclear at this point whether it is at all possible to apply such operators efficiently. Finally, in the work of [41] a very specific structure of the $\mathcal{H}$-Tucker tensor was considered that allowed to represent it exactly. For general tensor structures, one would either have to use HOSVD or the Black Box algorithm. The former is impractical due to the size of the matricizations and the latter is a heuristic approach and is not always reliable. Developing an efficient and reliable approximation routine for general tensor structures would allow to design software applicable to a variety of realistic CDO models.

BIBLIOGRAPHY

[1] Acar, E., Dunlavy, D. M., and Kolda, T. G. A scalable optimization approach for fitting canonical tensor decompositions. *J. Chemometr. 25*, 2 (2011), 67–86.

[2] Bechler, P. Existence of the best n-term approximants for structured dictionaries. *Arch. Math. 99*, 1 (2012), 61–70.

[3] Benzi, M. Preconditioning techniques for large linear systems: a survey. *J. Comput. Phys. 182*, 2 (2002), 418–477.

[4] Bungartz, H.-J., and Griebel, M. Sparse grids. *Acta Numer. 13* (2004), 147–269.

[5] Bungartz, H.-J., Heinecke, A., Pflüger, D., and Schraufstetter, S. Option pricing with a direct adaptive sparse grid approach. *J. Comput. Appl. Math. 236*, 15 (2012), 3741–3750.

[6] Carroll, J. D., and Chang, J.-J. Analysis of individual differences in multidimensional scaling via an n-way generalization of eckart-young decomposition. *Psychometrika 35*, 3 (1970), 283–319.

[7] Chui, C. K., and Villiers, J. D. Spline-wavelets with arbitrary knots on a bounded interval: Orthogonal decomposition and computational algorithms. *Commun. Appl. Anal. 2*, 4 (1998), 457–486.

[8] Cohen, A. *Numerical analysis of wavelet methods*, vol. 32 of *Studies in Mathematics and its Applications*. North-Holland Publishing Co., Amsterdam, 2003.

[9] Cohen, A., Dahmen, W., and DeVore, R. Adaptive wavelet methods for elliptic operator equations: convergence rates. *Math. Comp. 70*, 233 (2001), 27–75.

[10] Cohen, A., Dahmen, W., and DeVore, R. Adaptive wavelet methods. II. Beyond the elliptic case. *Found. Comput. Math. 2*, 3 (2002), 203–245.

[11] Cohen, A., Daubechies, I., and Feauveau, J.-C. Biorthogonal bases of compactly supported wavelets. *Comm. Pure Appl. Math. 45*, 5 (1992), 485–560.

[12] Colthurst, T. *Multidimensional wavelets*. PhD thesis, Massachusetts Institute of Technology, 1997.

[13] Daubechies, I. Orthonormal bases of compactly supported wavelets. II. Variations on a theme. *SIAM J. Math. Anal. 24*, 2 (1993), 499–519.

[14] De Lathauwer, L., De Moor, B., and Vandewalle, J. A multilinear singular value decomposition. *SIAM J. Matrix Anal. A. 21*, 4 (2000), 1253–1278.

[15] De Lathauwer, L., De Moor, B., and Vandewalle, J. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl. 21*, 4 (2000), 1253–1278.

[16] DeVore, R., Konyagin, S., and Temlyakov, V. Hyperbolic wavelet approximation. *Constr. Approx. 14*, 1 (1998), 1–26.

[17] DeVore, R. A. Nonlinear approximation. *Acta Numer. 7* (1998), 51–150.

[18] Dijkema, T. J. *Adaptive tensor product wavelet methods for solving PDEs*. PhD thesis, Universiteit Utrecht, 2009.

[19] Dijkema, T. J., Schwab, C., and Stevenson, R. An adaptive wavelet method for solving high-dimensional elliptic pdes. *Constr. Approx. 30*, 3 (2009), 423–455.

[20] Donoho, D. L. Interpolating wavelet transforms. *Preprint, Department of Statistics, Stanford University 2*, 3 (1992).

[21] Donoho, D. L. Nonlinear solution of linear inverse problems by wavelet–vaguelette decomposition. *Appl. Comput. Harmon. A. 2*, 2 (1995), 101–126.

[22] Donovan, G. C., Geronimo, J. S., and Hardin, D. P. Intertwining multiresolution analyses and the construction of piecewise-polynomial wavelets. *SIAM J. Math. Anal. 27*, 6 (1996), 1791–1815.

[23] Donovan, G. C., Geronimo, J. S., Hardin, D. P., and Massopust, P. R. Construction of orthogonal wavelets using fractal interpolation functions. *SIAM J. Math. Anal. 27*, 4 (1996), 1158–1192.

[24] Espig, M., and Hackbusch, W. A regularized Newton method for the efficient approximation of tensors represented in the canonical tensor format. *Numer. Math. 122*, 3 (2012), 489–525.

[25] Evans, L. C. *Partial differential equations*, second ed., vol. 19 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, RI, 2010.

[26] Gantumur, T., Harbrecht, H., and Stevenson, R. An optimal adaptive wavelet method without coarsening of the iterands. *Math. Comput. 76*, 258 (2007), 615–629.

[27] Grasedyck, L. Hierarchical singular value decomposition of tensors. *SIAM J. Matrix Anal. Appl. 31*, 4 (2009/10), 2029–2054.

[28] Hackbusch, W., and Kühn, S. A new scheme for the tensor representation. *J. Fourier Anal. Appl. 15*, 5 (2009), 706–722.

[29] Harshman, R. Foundations of the parafac procedure: Models and conditions for an "explanatory" multi-modal factor analysis. *UCLA Working Papers in Phonetics 16* (1970).

[30] Kestler, S. *On theaAdaptive tensor product wavelet Galerkin method with applications in finance*. PhD thesis, Universität Ulm, 2013.

[31] Kestler, S., Steih, K., and Urban, K. An efficient space-time adaptive wavelet galerkin method for time-periodic parabolic partial differential equations. *arXiv preprint arXiv:1401.5782* (2014).

[32] Kestler, S., and Stevenson, R. An Efficient Approximate Residual Evaluation in the Adaptive Tensor Product Wavelet Method. *J. Sci. Comput. 57*, 3 (2013), 439–463.

[33] Khoromskij, B. N., and Oseledets, I. V. QTT approximation of elliptic solution operators in higher dimensions. *Russian J. Numer. Anal. Math. Modelling 26*, 3 (2011), 303–322.

[34] Kiesel, R., Rupp, A., and Urban, K. Valuation of structured financial products by adaptive multiwavelet methods in high dimensions. *Preprint, Department of Numerical Mathematics, University of Ulm* (2014).

[35] Kressner, D., and Tobler, C. htucker–a matlab toolbox for tensors in hierarchical tucker format. *Mathicse, EPF Lausanne, Preprint* (2012).

[36] Light, W. A., and Cheney, E. W. *Approximation theory in tensor product spaces*, vol. 1169 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 1985.

[37] Mallat, S. *A wavelet tour of signal processing*, third ed. Elsevier/Academic Press, Amsterdam, 2009. The sparse way, With contributions from Gabriel Peyré.

[38] Meyer, Y., and Coifman, R. *Wavelets: Calderón-Zygmund and multilinear operators*, vol. 48. Cambridge University Press, 2000.

[39] Quarteroni, A., and Valli, A. *Numerical approximation of partial differential equations*, vol. 23 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, 1994.

[40] Royden, H. L. *Real analysis*, third ed. Macmillan Publishing Company, New York, 1988.

[41] Rupp, A. *High dimensional wavelet methods for structured financial products*. PhD thesis, Universität Ulm, 2013.

[42] Schwab, C., and Stevenson, R. Adaptive wavelet algorithms for elliptic pdes on product domains. *Math. Comput. 77*, 261 (2008), 71–92.

[43] Schwab, C., and Stevenson, R. Space-time adaptive wavelet methods for parabolic evolution problems. *Math. Comput. 78*, 267 (2009), 1293–1318.

[44] Schwab, C., and Suli, E. Adaptive galerkin approximation algorithms for partial differential equations in infinite dimensions. Tech. rep., Hausdorff Institute for Mathematics, Bonn, 2011.

[45] Stevenson, R. Adaptive solution of operator equations using wavelet frames. *SIAM J. Numer. Anal. 41*, 3 (2003), 1074–1100.

[46] Stevenson, R. Adaptive wavelet methods for solving operator equations: an overview. In *Multiscale, nonlinear and adaptive approximation*. Springer, Berlin, 2009, pp. 543–597.

[47] Thomée, V. *Galerkin finite element methods for parabolic problems*, second ed., vol. 25 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, 2006.

[48] Urban, K. *Wavelet methods for elliptic partial differential equations*. Numerical Mathematics and Scientific Computation. Oxford University Press, Oxford, 2009.

[49] Van Loan, C. F. The ubiquitous Kronecker product. *J. Comput. Appl. Math. 123*, 1-2 (2000), 85–100. Numerical analysis 2000, Vol. III. Linear algebra.

[50] Vandevoorde, D., and Josuttis, N. M. *C++ templates: the complete guide*. Addison-Wesley Professional, 2002.

[51] Weidmann, J. *Lineare Operatoren in Hilberträumen 1: Grundlagen*, vol. 1. Springer, 2000.

[52] Zenger, C. Sparse grids. In *Parallel algorithms for partial differential equations (Kiel, 1990)*, vol. 31 of *Notes Numer. Fluid Mech.* Vieweg, Braunschweig, 1991, pp. 241–251.

VITA

Mazen Ali was born in Sana′a, Yemen on June 13, 1990. He graduated from the Russian Embassy School 2007 in Sana'a, Yemen. After finishing High School, he left for studies to Germany. Upon completing classes in a German language school in Bonn, he started his studies in Economics and Business Administration at the Goethe University in Frankfurt on the Main in February, 2008. During his last year of Bachelor studies, he completed additional classes in Mathematics at the Faculty of Computer Science and Mathematics, Goethe University. He graduated from Goethe University with a Bachelor of Science in Economics and Business Administration in Summer 2012. Immediately after his graduation, he started his studies in Master of Finance with major in financial mathematics at the University of Ulm, Germany. During his second semester at the University of Ulm, he worked for the Department of Numerical Mathematics as a graduate assistant in coding. During his second year of master studies, he transferred in Fall 2013 to Missouri University of Science and Technology to do his Master's in Applied Mathematics. During this time as a graduate student, he was employed by the Department of Mathematics and Statistics as a graduate teaching assistant. Mazen Ali received his Master of Science in Applied Mathematics from Missouri University of Science and Technology in July 2014.