FAST EVALUATION OF SYSTEM MATRICES W.R.T. MULTI-TREE COLLECTIONS OF TENSOR PRODUCT REFINABLE BASIS FUNCTIONS

SEBASTIAN KESTLER AND ROB STEVENSON

Abstract. An algorithm is presented that for a local bilinear form evaluates in linear complexity the application of the stiffness matrix w.r.t. a collection of tensor product multiscale basis functions, assuming that this collection has a multi-tree structure. It generalizes an algorithm for sparse-grid index sets [*SIAM J. Sci. Comput.*, 17 (1996), pp. 631-646] by R. Balder and Ch. Zenger, and it finds its application in adaptive tensor product approximation methods.

1. Introduction

For $1 \leq i \leq n$, let $\check{\Psi}_i = \{\check{\psi}_{i,\lambda} : \lambda \in \check{\nabla}_i\}$ and $\hat{\Psi}_i = \{\hat{\psi}_{i,\lambda} : \lambda \in \hat{\nabla}_i\}$ be collections of multi-scale functions, e.g., wavelet bases, multi-level frames, or collections of hierarchical "hat" functions. We assume that the functions from these collections, which we simply will refer to as being wavelets, satisfy standard locality assumptions meaning that the diameter of the support of wavelets on *level* ℓ is of order $2^{-\ell}$. Let $a_i(\cdot,\cdot)$ be a bilinear form that is *local* meaning that $a_i(u,v) = 0$ whenever $|\operatorname{supp} u \cap \operatorname{supp} v| = 0$.

We set $\check{\mathbf{\Psi}} = \{\check{\boldsymbol{\psi}}_{\boldsymbol{\lambda}} = \otimes_{i=1}^n \check{\psi}_{\lambda_i} : \boldsymbol{\lambda} \in \check{\mathbf{\nabla}} := \prod_{i=1}^n \check{\nabla}_i\}$ and similarly $\hat{\mathbf{\Psi}}$, and consider the bilinear form $\mathbf{a}(\cdot,\cdot)$ defined by $\mathbf{a}(\otimes_{i=1}^n u_i, \otimes_{i=1}^n v_i) = \prod_{i=1}^n a_i(u_i,v_i)$. A typical example being $\mathbf{a} = \mathbf{a}_k$ defined by $a_i(u_i,v_i) = \begin{cases} \int_0^1 u_i(x)v_i(x)dx, & i \neq k, \\ \int_0^1 u_i'(x)v_i'(x)dx, & i = k, \end{cases}$ in which case $\sum_{k=1}^n \mathbf{a}_k$ is the bilinear form that results from the variational formulation of Poisson's problem on $(0,1)^n$.

The topic of this paper is to apply, for finite $\check{\mathbf{\Lambda}} \subset \check{\mathbf{\nabla}}$, $\hat{\mathbf{\Lambda}} \subset \hat{\mathbf{\nabla}}$, the "system matrix"

$$\mathbf{a}(\check{\mathbf{\Psi}}|_{\check{\mathbf{\Lambda}}}, \hat{\mathbf{\Psi}}|_{\hat{\mathbf{\Lambda}}}) := [\mathbf{a}(\check{\boldsymbol{\psi}}_{\boldsymbol{\lambda}}, \hat{\boldsymbol{\psi}}_{\boldsymbol{\mu}})]_{\boldsymbol{\lambda}\in\check{\mathbf{\Lambda}},\boldsymbol{\mu}\in\hat{\mathbf{\Lambda}}} \text{ in } \mathcal{O}(\#\check{\mathbf{\Lambda}} + \#\hat{\mathbf{\Lambda}}) \text{ operations.} \tag{1.1}$$

For doing so, it is no real restriction to assume that the collections $\check{\Psi}_i$ and $\hat{\Psi}_i$, and the bilinear forms $a_i(\cdot,\cdot)$ are independent of i. Furthermore, inside this introduction we focus on the simplified case where $(\Psi =)\check{\Psi} = \hat{\Psi}$ and $(\mathbf{\Lambda} =)\check{\mathbf{\Lambda}} = \hat{\mathbf{\Lambda}}$.

Already for $n = 1$, the application of the system matrix in linear complexity cannot be expected for arbitrary $\Lambda \subset \nabla$, because $a(\Psi|_\Lambda, \Psi|_\Lambda)$ is generally not

Date: November 8, 2012.

1991 *Mathematics Subject Classification.* 41A30, 65N30, 65Y20.

Key words and phrases. Tensor product approximation, adaptivity, sparse grids, optimal computational complexity.

The first author has been supported by the Deutsche Forschungsgemeinschaft within the Research Training Group (Graduiertenkolleg) GrK1100 *Modellierung, Analyse und Simulation in der Wirtschaftsmathematik* at the University of Ulm.

sparse due to interactions between wavelets on different levels. Yet, when Λ is the collection of indices of *all* wavelets up to some level ℓ, a solution is provided by the application of a transformation T to a single-scale basis Φ_ℓ on level ℓ. Writing $\Psi|_\Lambda^\top = \Phi_\ell^\top T$, viewing collections of functions as column vectors, we have $a(\Psi|_\Lambda, \Psi|_\Lambda) = T^\top a(\Phi_\ell, \Phi_\ell)T$. By the sparsity of $a(\Phi_\ell, \Phi_\ell)$, and under the assumption that each of its entries can be computed in $\mathcal{O}(1)$ operations, each of these three matrices on the right-hand side can be applied in $\mathcal{O}(\#\Lambda)$ operations, and so can $a(\Psi|_\Lambda, \Psi|_\Lambda)$.

This approach extends to the situation where Λ is a general *tree*, which we define as a set such that for any $\lambda \in \Lambda$ with $|\lambda| > 0$, the support of ψ_λ is covered by the supports of ψ_μ for some $\mu \in \Lambda$ with $|\mu| = |\lambda| - 1$. The argument is that for a tree Λ, there exists a locally finite collection of scaling functions whose span contains $\operatorname{span}\Psi|_\Lambda$, where, thanks to the tree constraint, the representation of the embedding, being a generalization the aforementioned basis transformation T, can be performed in $\mathcal{O}(\#\Lambda)$ operations.

For $n > 1$, we have

$$\mathbf{a}(\boldsymbol{\Psi}|_{\boldsymbol{\Lambda}}, \boldsymbol{\Psi}|_{\boldsymbol{\Lambda}}) = R_{\boldsymbol{\Lambda}}(A\otimes\cdots\otimes A)I_{\boldsymbol{\Lambda}},$$

where $A := a(\Psi, \Psi)$, $I_{\boldsymbol{\Lambda}}$ is the extension operator with zeros of a vector indexed by $\boldsymbol{\Lambda}$ to one indexed by $\boldsymbol{\nabla}$, and $R_{\boldsymbol{\Lambda}}$ denotes its adjoint being the restriction of a vector to its indices in $\boldsymbol{\Lambda}$.

If $\boldsymbol{\Lambda}$ is equal to $\bar{\boldsymbol{\Lambda}} := \{\boldsymbol{\lambda} \in \boldsymbol{\nabla} : |||\boldsymbol{\lambda}|||_\infty \le \ell\}$, being the set of all multi-indices $\boldsymbol{\lambda}$ with, for some level $\ell \in \mathbb{N}_0$, $|||\boldsymbol{\lambda}|||_\infty := \max_i |\lambda_i| \le \ell$, i.e., $\bar{\boldsymbol{\Lambda}}$ corresponds to a *full grid*, then, with Λ denoting the set of $\lambda \in \nabla$ with $|\lambda| \le \ell$, one has

$$R_{\bar{\boldsymbol{\Lambda}}}(A\otimes\cdots\otimes A)I_{\bar{\boldsymbol{\Lambda}}} = a(\Psi|_\Lambda, \Psi|_\Lambda)\otimes\cdots\otimes a(\Psi|_\Lambda, \Psi|_\Lambda).$$

We conclude that the application of $\mathbf{a}(\boldsymbol{\Psi}|_{\boldsymbol{\Lambda}}, \boldsymbol{\Psi}|_{\boldsymbol{\Lambda}})$ can be evaluated in $\mathcal{O}(n\#\boldsymbol{\Lambda}) = \mathcal{O}(\#\boldsymbol{\Lambda})$ operations.

Next, we consider $\boldsymbol{\Lambda}$ to correspond to a *sparse grid*, i.e., for some $\ell \in \mathbb{N}_0$, it is the set of all multi-indices $\boldsymbol{\lambda}$ with $\|\boldsymbol{\lambda}\|_1 := \sum_i |\lambda_i| \le \ell$. For simplicity thinking here of $n = 2$, we write $R_{\boldsymbol{\Lambda}}(A\otimes A)I_{\boldsymbol{\Lambda}} = R_{\boldsymbol{\Lambda}}(A\otimes\mathrm{Id})(\mathrm{Id}\otimes A)I_{\boldsymbol{\Lambda}}$. In view of the subsequent application of $R_{\boldsymbol{\Lambda}}(A\otimes\mathrm{Id})$, we realize that we need the result of the application of $(\mathrm{Id}\otimes A)I_{\boldsymbol{\Lambda}}$ only on some finite subset of $\boldsymbol{\nabla}$. The generally smallest subset that can be selected is the corresponding full grid index set $\bar{\boldsymbol{\Lambda}}$ defined above, i.e., we have $R_{\boldsymbol{\Lambda}}(A\otimes A)I_{\boldsymbol{\Lambda}} = R_{\boldsymbol{\Lambda}}(A\otimes\mathrm{Id})I_{\bar{\boldsymbol{\Lambda}}}R_{\bar{\boldsymbol{\Lambda}}}(\mathrm{Id}\otimes A)I_{\boldsymbol{\Lambda}}$. Unfortunately, the applications of both $R_{\boldsymbol{\Lambda}}(A\otimes\mathrm{Id})I_{\bar{\boldsymbol{\Lambda}}}$ and $R_{\bar{\boldsymbol{\Lambda}}}(\mathrm{Id}\otimes A)I_{\boldsymbol{\Lambda}}$ require $\mathcal{O}(\#\bar{\boldsymbol{\Lambda}})$ operations, where, in the standard setting that $\#\{\lambda \in \nabla : |\lambda| = \ell\} \eqsim 2^\ell$, one has that $\#\boldsymbol{\Lambda} \eqsim \ell 2^\ell$ and $\#\bar{\boldsymbol{\Lambda}} \eqsim 4^\ell$.

(Here and in other places, with $C \eqsim D$ we mean that both $C \lesssim D$ and $C \gtrsim D$, with the first relation meaning that C can be bounded by some absolute multiple of D, and the second one being defined as $D \lesssim C$.)

To solve the above problem, we apply a key idea by Balder and Zenger in [4] for the hierarchical hat functions, that for more general functions was applied later in, e.g., [3, 21, 20]. We split A into the upper block matrix $U = [a(\psi_\lambda, \psi_\mu)]_{|\lambda|\le|\mu|}$ and the strictly lower block matrix $L = [a(\psi_\lambda, \psi_\mu)]_{|\lambda|>|\mu|}$. By definition of U, L, and $\boldsymbol{\Lambda}$, we have

$$(U\otimes\mathrm{Id})I_{\boldsymbol{\Lambda}} = I_{\boldsymbol{\Lambda}}R_{\boldsymbol{\Lambda}}(U\otimes\mathrm{Id})I_{\boldsymbol{\Lambda}}, \quad R_{\boldsymbol{\Lambda}}(L\otimes\mathrm{Id}) = R_{\boldsymbol{\Lambda}}(L\otimes\mathrm{Id})I_{\boldsymbol{\Lambda}}R_{\boldsymbol{\Lambda}},$$

from which we infer that

$$\begin{aligned} R_{\boldsymbol{\Lambda}}(A\otimes A)I_{\boldsymbol{\Lambda}} &= R_{\boldsymbol{\Lambda}}((U+L)\otimes A)I_{\boldsymbol{\Lambda}} \\ &= R_{\boldsymbol{\Lambda}}(L\otimes\mathrm{Id})(\mathrm{Id}\otimes A)I_{\boldsymbol{\Lambda}} + R_{\boldsymbol{\Lambda}}(U\otimes\mathrm{Id})(\mathrm{Id}\otimes A)I_{\boldsymbol{\Lambda}} \\ &= R_{\boldsymbol{\Lambda}}(L\otimes\mathrm{Id})(\mathrm{Id}\otimes A)I_{\boldsymbol{\Lambda}} + R_{\boldsymbol{\Lambda}}(\mathrm{Id}\otimes A)(U\otimes\mathrm{Id})I_{\boldsymbol{\Lambda}} \\ &= R_{\boldsymbol{\Lambda}}(L\otimes\mathrm{Id})I_{\boldsymbol{\Lambda}}\, R_{\boldsymbol{\Lambda}}(\mathrm{Id}\otimes A)I_{\boldsymbol{\Lambda}} + R_{\boldsymbol{\Lambda}}(\mathrm{Id}\otimes A)I_{\boldsymbol{\Lambda}}\, R_{\boldsymbol{\Lambda}}(U\otimes\mathrm{Id})I_{\boldsymbol{\Lambda}}. \end{aligned}$$

Since $\boldsymbol{\Lambda}$, "frozen" in either of its coordinates, is a collection of indices of all wavelets up to some level, $R_{\boldsymbol{\Lambda}}(\mathrm{Id}\otimes A)I_{\boldsymbol{\Lambda}}$, and similarly, $R_{\boldsymbol{\Lambda}}(L\otimes\mathrm{Id})I_{\boldsymbol{\Lambda}}$ and $R_{\boldsymbol{\Lambda}}(U\otimes\mathrm{Id})I_{\boldsymbol{\Lambda}}$, can be applied in $\mathcal{O}(\#\boldsymbol{\Lambda})$ operations, and so can $\mathbf{a}(\boldsymbol{\Psi}|_{\boldsymbol{\Lambda}}, \boldsymbol{\Psi}|_{\boldsymbol{\Lambda}})$.

Remark 1.1. Since, for general $n > 1$, in the above scheme two recursive calls for $\mathrm{Id}\otimes A\otimes\cdots\otimes A$ have to be made, one verifies that its complexity is $\mathcal{O}(2^n\#\boldsymbol{\Lambda})$. So in high dimensions, one should avoid multiplications with $R_{\boldsymbol{\Lambda}}(A_1\otimes\cdots\otimes A_n)I_{\boldsymbol{\Lambda}}$ for more than one or a few A_i not being truly sparse. For, say, the Poisson problem, this can be realized by applying orthogonal wavelets (cf. [7, 8]) or prewavelets (cf. [16, 11]). This issue, however, is outside the scope of the current paper, and we will ignore the dependency of constants on the space dimension n.

The goal of this paper is to generalize the algorithm for the multiplication with $\mathbf{a}(\check{\boldsymbol{\Psi}}|_{\check{\boldsymbol{\Lambda}}}, \hat{\boldsymbol{\Psi}}|_{\hat{\boldsymbol{\Lambda}}})$ from [4] to the case of $\check{\boldsymbol{\Lambda}}$ and $\hat{\boldsymbol{\Lambda}}$ being *multi-trees*, and to prove that it requires only $\mathcal{O}(\#\check{\boldsymbol{\Lambda}} + \#\hat{\boldsymbol{\Lambda}})$ operations. We define a multi-index set $\boldsymbol{\Lambda}$ to be a multi-tree when "frozen" in any $n-1$ coordinates, it is a *tree* in the remaining coordinate.

The application of this result lies in adaptive tensor product approximation methods, as adaptive sparse grid methods ([2] + references cited there), or adaptive tensor product wavelet Galerkin methods (e.g. [18]). It seems that multi-trees are the most general sets for which (1.1) is realizable (unless, by a special choice of the wavelets, the bi-infinite matrix $a(\check{\boldsymbol{\Psi}}, \hat{\boldsymbol{\Psi}})$ as a whole is sparse, cf. [6]).

For $\check{\boldsymbol{\Psi}} = \hat{\boldsymbol{\Psi}}$ being a hierarchical basis and $\check{\boldsymbol{\Lambda}} = \hat{\boldsymbol{\Lambda}}$, similar results, although described more informally, can be found in [1, 11, 17]. For a hierarchical basis, our condition of $\check{\boldsymbol{\Lambda}} = \hat{\boldsymbol{\Lambda}}$ being a multi-tree is equal to the condition on this index set imposed in these references. The discussion in [11, §3.1.3] about a prewavelet basis learns that the generalization from the hierarchical basis to a general multi-level collection is not trivial.

Our generalization of possibly having different collections $\check{\boldsymbol{\Psi}}$ and $\hat{\boldsymbol{\Psi}}$ and/or different multi-trees $\check{\boldsymbol{\Lambda}}$ and $\hat{\boldsymbol{\Lambda}}$ has obvious applications in Petrov-Galerkin methods. Another application of this generalized setting will be a new residual evaluation scheme, inside an adaptive tensor product wavelet Galerkin method, that is presented in [15].

This paper is organized as follows: In Sect. 2, we present the evaluation schemes for A, U and L restricted to blocks formed by collections of row- and column indices that both are trees. In Sect. 3, these evaluation schemes will be the building blocks for the multiplication with $\mathbf{a}(\check{\boldsymbol{\Psi}}|_{\check{\boldsymbol{\Lambda}}}, \hat{\boldsymbol{\Psi}}|_{\hat{\boldsymbol{\Lambda}}})$ in linear complexity for multi-trees $\check{\boldsymbol{\Lambda}} \subset \check{\boldsymbol{\nabla}}$ and $\hat{\boldsymbol{\Lambda}} \subset \hat{\boldsymbol{\nabla}}$. In Sect. 4, we report on numerical experiments with the adaptive wavelet Galerkin method that confirm this linear complexity.

2. The one factor case

2.1. **Wavelet assumptions.** We consider two collections

$$\breve{\Psi} = \{\breve{\psi}_\lambda : \lambda \in \breve{\nabla}\}, \quad \hat{\Psi} = \{\hat{\psi}_\lambda : \lambda \in \hat{\nabla}\}$$

of functions on some domain $\Omega \subset \mathbb{R}^d$. For $\cup \in \{\breve{},\hat{}\}$, and with $|\lambda| \in \mathbb{N}_0$ denoting the *level* of $\lambda \in \breve{\nabla}$, we assume that the collection $\breve{\Psi}$ consists of local functions in the sense that

$$\sup_{\lambda\in\breve{\nabla}} 2^{|\lambda|} \operatorname{diam} \operatorname{supp} \breve{\psi}_\lambda < \infty, \tag{2.1}$$

$$\sup_{\ell\in\mathbb{N}_0} \sup_{x\in\Omega} \#\{\lambda \in \breve{\nabla} : |\lambda| = \ell \wedge \operatorname{supp} \breve{\psi}_\lambda \cap B(x; 2^{-\ell}) \neq \emptyset\} < \infty. \tag{2.2}$$

We will refer to the functions $\breve{\psi}_\lambda$ as being *wavelets*, although not necessarily they have vanishing moments or other specific wavelet properties.

For $\ell \in \mathbb{N}_0$, and any $\Lambda \subset \breve{\nabla}$, we set $\Lambda_\ell := \{\lambda \in \Lambda : |\lambda| = \ell\}$ and $\Lambda_{\ell\uparrow} := \{\lambda \in \Lambda : |\lambda| \geq \ell\}$. We write $\breve{\Psi}_\ell := \breve{\Psi}|_{\breve{\nabla}_\ell}$, and add the harmless assumption that for any $\ell \in \mathbb{N}_0$,

$$\Omega = \cup_{\breve{\psi}_\lambda \in \breve{\Psi}_\ell} \operatorname{supp} \breve{\psi}_\lambda. \tag{2.3}$$

For $\ell \in \mathbb{N}_0$, we assume a collection $\breve{\Phi}_\ell = \{\breve{\phi}_\lambda : \lambda \in \breve{\Delta}_\ell\}$, whose members will be referred to as being *scaling functions*, with

$$\operatorname{span} \breve{\Phi}_{\ell+1} \supseteq \operatorname{span} \breve{\Phi}_\ell \cup \breve{\Psi}_{\ell+1}, \quad \breve{\Phi}_0 = \breve{\Psi}_0, \tag{2.4}$$

$$\sup_{\ell\in\mathbb{N}_0} \sup_{\lambda\in\breve{\Delta}_\ell} 2^\ell \operatorname{diam} \operatorname{supp} \breve{\phi}_\lambda < \infty, \tag{2.5}$$

$$\sup_{\ell\in\mathbb{N}_0} \sup_{x\in\Omega} \#\{\lambda \in \breve{\Delta}_\ell : \operatorname{supp} \breve{\phi}_\lambda \cap B(x; 2^{-\ell}) \neq \emptyset\} < \infty, \tag{2.6}$$

$$\{\breve{\phi}_\lambda|_\Sigma : \lambda \in \breve{\Delta}_\ell,\ \breve{\phi}_\lambda|_\Sigma \not\equiv 0\} \text{ is independent (for all open } \Sigma \subset \Omega,\ \ell \in \mathbb{N}_0). \tag{2.7}$$

W.l.o.g. we assume that the index sets $\breve{\Delta}_\ell$ for different ℓ are mutually disjoint, and set $\breve{\Phi} := \cup_{\ell\in\mathbb{N}_0} \breve{\Phi}_\ell$ with index set $\breve{\Delta} := \cup_{\ell\in\mathbb{N}_0} \breve{\Delta}_\ell$. For $\lambda \in \breve{\Delta}$, we set $|\lambda| := \ell$ when $\lambda \in \breve{\Delta}_\ell$.

Viewing $\breve{\Psi}_\ell$, $\breve{\Phi}_\ell$ as column vectors, the assumptions we made so far guarantee the existence of matrices $\breve{M}_{\ell,0}$, $\breve{M}_{\ell,1}$ such that

$$\left[\breve{\Phi}_{\ell-1}^\top \quad \breve{\Psi}_\ell^\top\right] = \breve{\Phi}_\ell^\top \left[\breve{M}_{\ell,0} \quad \breve{M}_{\ell,1}\right],$$

where the number of non-zeros per row and column of $\breve{M}_{\ell,0}$ and $\breve{M}_{\ell,1}$ is finite, uniformly in the rows and columns and in $\ell \in \mathbb{N}$.

A subset $\Lambda \subset \breve{\nabla}_{\ell\uparrow}$ is called an *ℓ-tree*, or simply a tree when the value of ℓ is clear from the context, when for any $\lambda \in \Lambda$ with $|\lambda| > \ell$, the support of $\breve{\psi}_\lambda$ is covered by the supports of $\breve{\psi}_\mu$ for some $\mu \in \Lambda$ with $|\mu| = |\lambda| - 1$.

For $\mu \in \breve{\nabla}$, we define

$$\breve{S}_\mu = \{x \in \Omega : \operatorname{dist}(x, \operatorname{supp} \breve{\psi}_\mu) \leq C 2^{-|\mu|}\}$$

for some constant $C \geq 0$ such that for all

$$\mu, \lambda \in \breve{\nabla} \text{ with } |\mu| = |\lambda| - 1 \text{ and } |\operatorname{supp} \breve{\psi}_\lambda \cap \operatorname{supp} \breve{\psi}_\mu| > 0 \Longrightarrow \breve{S}_\mu \supset \breve{S}_\lambda. \tag{2.8}$$

For Haar wavelets or hierarchical "hat" functions, C can be taken equal to 0. The next lemma shows that, thanks to (2.1), a suitable constant C always exists.

Lemma 2.1. *With* $C := \sup_{\lambda\in\breve{\nabla}} 2^{|\lambda|}\operatorname{diam}\operatorname{supp}\breve{\psi}_\lambda$, (2.8) *is satisfied.*

Proof. For μ and λ as in (2.8), and $x \in \breve{S}_\lambda$, $\operatorname{dist}(x, \breve{S}_\mu) \le C2^{-|\lambda|} + \operatorname{diam}\operatorname{supp}\breve{\psi}_\lambda \le C2^{-|\mu|}$. □

On $\operatorname{span}\breve{\Phi} \times \operatorname{span}\hat{\Phi}$, we will consider a bilinear form $a(\cdot,\cdot)$ that is *local* in the sense that for $\breve{\phi}_\lambda \in \breve{\Phi}$, $\hat{\phi}_\mu \in \hat{\Phi}$,

$$a(\breve{\phi}_\lambda, \hat{\phi}_\mu) = 0 \quad \text{whenever } |\operatorname{supp}\breve{\phi}_\lambda \cap \operatorname{supp}\hat{\phi}_\mu| = 0.$$

For collections $\breve{\Sigma} \subset \breve{\Psi} \cup \breve{\Phi}$, $\hat{\Sigma} \subset \hat{\Psi} \cup \hat{\Phi}$, with $a(\breve{\Sigma}, \hat{\Sigma})$ we will mean the matrix $[a(\breve{\sigma}, \hat{\sigma})]_{\breve{\sigma}\in\breve{\Sigma},\hat{\sigma}\in\hat{\Sigma}}$. We set the matrix

$$A := a(\breve{\Psi}, \hat{\Psi}), \tag{2.9}$$

and the "upper block" and "strictly lower block" matrices

$$U = \big[a(\breve{\psi}_\lambda, \hat{\psi}_\mu)\big]_{|\lambda|\le|\mu|}, \quad L = \big[a(\breve{\psi}_\lambda, \hat{\psi}_\mu)\big]_{|\lambda|>|\mu|},$$

so that $A = U + L$.

Remark 2.2. The conditions (2.1), (2.2), (2.5), (2.6), and (2.7) are used to conclude the uniform sparsity of $\breve{M}_{\ell,0}$ and $\breve{M}_{\ell,1}$, i.e., the number of non-zero entries in each of their rows or columns is bounded uniformly in ℓ. They also imply that, for local $a(\cdot,\cdot)$, $a(\breve{\Phi}_\ell, \hat{\Phi}_\ell)$ is uniformly sparse.

In the following exposition, the conditions involving the supports of the "wavelets" and "scaling functions", i.e., (2.1), (2.2), (2.3), (2.5), and (2.6) cannot, however, be simply replaced by the conditions of uniform sparsity of $\breve{M}_{\ell,0}$, $\breve{M}_{\ell,1}$ and $a(\breve{\Phi}_\ell, \hat{\Phi}_\ell)$. Indeed, for example, the concept of a tree, defined in terms of supports, allows us to conclude from $|\operatorname{supp}\breve{\psi}_\lambda \cap \operatorname{supp}\hat{\psi}_\mu| = 0$ for all $\mu \in \Lambda_\ell$, where $\Lambda \subset \hat{\nabla}$ is a tree, that $|\operatorname{supp}\breve{\psi}_\lambda \cap \operatorname{supp}\hat{\psi}_\mu| = 0$, and thus $a(\breve{\psi}_\lambda, \hat{\psi}_\mu) = 0$, for all $\mu \in \Lambda_{\ell\uparrow}$. They also imply that the number of $\lambda \in \breve{\Delta}_{\ell-1}$ with $|\operatorname{supp}\breve{\phi}_\lambda \cap \operatorname{supp}\hat{\psi}_\mu| > 0$ for any $\mu \in \Lambda_\ell$ is $\mathcal{O}(\#\Lambda_\ell)$. These and similar properties will be used extensively.

Remark 2.3. Condition (2.4) allows for $\breve{\Psi}$ to be dependent. In particular, besides multi-level bases, the framework includes multi-level frames (cf. [14]). A bilinear form a on $\operatorname{span}\breve{\Phi} \times \operatorname{span}\hat{\Phi}$ gives rise to a matrix A as defined in (2.9), i.e., a bilinear form on $\ell_0(\breve{\nabla}) \times \ell_0(\hat{\nabla})$, where $\ell_0(\breve{\nabla})$, and similarly $\ell_0(\hat{\nabla})$, denotes the set of vectors indexed by $\breve{\nabla}$ that have finite support. When for either $\cup \in \{\breve{\cup}, \hat{\cup}\}$, $\breve{\Psi}$ is not independent, or $\operatorname{span}\breve{\Psi} \neq \operatorname{span}\breve{\Phi}$, then this relation between a bilinear form and a matrix is not one-to-one. Consequently, in general the matrices U and L cannot be associated to bilinear forms on $\operatorname{span}\breve{\Phi} \times \operatorname{span}\hat{\Phi}$.

Remark 2.4. The setting we discussed so far also applies to the case that $\breve{\Psi}$ and $\breve{\Phi}$ are collections of *functionals*, and $a(\breve{\phi}_\lambda, \hat{\phi}_\mu)$ reads as the duality pairing $\breve{\phi}_\lambda(\hat{\phi}_\mu)$. Then, on all places $\operatorname{supp}\breve{\psi}_\lambda$ and $\operatorname{supp}\breve{\phi}_\lambda$ should be read as the distributional support. In that case, the condition $\Omega = \cup_{\breve{\psi}_\lambda\in\breve{\Psi}_\ell} \operatorname{supp}\breve{\psi}_\lambda$ from (2.3) may impose a real restriction, which, however, can be circumvented, for $\breve{\Psi} \cup \breve{\Phi}$ being a collection of either functions or functionals, by realizing that on all places $\operatorname{supp}\breve{\psi}_\lambda$ and $\operatorname{supp}\breve{\phi}_\lambda$ can be read as a set that *contains* the support of $\breve{\psi}_\lambda$ or $\breve{\phi}_\lambda$.

2.2. **The application of $A|_{\check\Lambda\times\hat\Lambda}$, $U|_{\check\Lambda\times\hat\Lambda}$, and $L|_{\check\Lambda\times\hat\Lambda}$ for trees $\check\Lambda\subset\check\nabla$ and $\hat\Lambda\subset\hat\nabla$.** The matrix-vector multiplication routines that we are going to present will only require the evaluation of $a(\check\phi_\lambda,\hat\phi_\mu)$ for $|\lambda|=|\mu|$. Our operation counts will be valid under the assumption that such an evaluation requires $\mathcal{O}(1)$ operations.

Our first algorithm concerns the application of $A|_{\check\Lambda\times\hat\Lambda}=a(\check\Psi|_{\check\Lambda},\hat\Psi|_{\hat\Lambda})$ for trees $\check\Lambda\subset\check\nabla$ and $\hat\Lambda\subset\hat\nabla$. Since the algorithm will be recursive in the coarsest level, it has to accept more general subsets of $\check\Phi\cup\check\Psi$ and $\hat\Phi\cup\hat\Psi$ than only $\check\Psi|_{\check\Lambda}$ and $\hat\Psi|_{\hat\Lambda}$ for trees $\check\Lambda\subset\check\nabla$ and $\hat\Lambda\subset\hat\nabla$.

`eval`$A(\ell,\check\Pi,\check\Lambda,\hat\Pi,\hat\Lambda,\vec d,\vec c)$
% *Input:* $\ell\in\mathbb{N}$, $\check\Pi\subset\check\Delta_{\ell-1}$, $\hat\Pi\subset\hat\Delta_{\ell-1}$, *$\ell$-trees* $\check\Lambda\subset\check\nabla_{\ell\uparrow}$ *and* $\hat\Lambda\subset\hat\nabla_{\ell\uparrow}$, *all finite,*
% $\vec d:=(d_\lambda)_{\lambda\in\hat\Pi},\vec c:=(c_\lambda)_{\lambda\in\hat\Lambda}\subset\mathbb{R}$.
% *Output: With* $v:=\vec d^{\,\top}\hat\Phi|_{\hat\Pi}+\vec c^{\,\top}\hat\Psi|_{\hat\Lambda}$, *the concatenation of the vectors inside*
% *the dashed boxes gives*
% $a(\check\Phi|_{\check\Pi},v)$, $a(\check\Psi|_{\check\Lambda},v)$.
% *Set*
% $\check\Pi^{(1)}:=\{\lambda\in\check\Pi:|\operatorname{supp}\check\phi_\lambda\cap\operatorname{supp}\hat\psi_\mu|=0\ \forall\mu\in\hat\Lambda_\ell\}$, $\check\Pi^{(2)}:=\check\Pi\setminus\check\Pi^{(1)}$,
% $\hat\Pi^{(1)}:=\{\lambda\in\hat\Pi:|\operatorname{supp}\hat\phi_\lambda\cap\operatorname{supp}\check\psi_\mu|=0\ \forall\mu\in\check\Lambda_\ell\}$, $\hat\Pi^{(2)}:=\hat\Pi\setminus\hat\Pi^{(1)}$,
% $\vec d^{\,(1)}:=\vec d|_{\hat\Pi^{(1)}}$, *and* $\vec d^{\,(2)}:=\vec d|_{\hat\Pi^{(2)}}$.
% *Let* $\underline{\check\Pi}\subset\check\Delta_\ell$, $\underline{\hat\Pi}\subset\hat\Delta_\ell$ *be the smallest collections with*
% $\operatorname{span}\check\Phi|_{\underline{\check\Pi}}\supset\operatorname{span}\check\Phi|_{\check\Pi^{(2)}}\cup\check\Psi_{\check\Lambda_\ell}$, *or* $\operatorname{span}\hat\Phi|_{\underline{\hat\Pi}}\supset\operatorname{span}\hat\Phi|_{\hat\Pi^{(2)}}\cup\hat\Psi_{\hat\Lambda_\ell}$.

`if` $\check\Pi\cup\check\Lambda\neq\emptyset$ `and` $\hat\Pi\cup\hat\Lambda\neq\emptyset$ `then`
  compute $\boxed{a(\check\Phi|_{\check\Pi^{(1)}},\hat\Phi|_{\hat\Pi})\vec d}$
  $\vec{\underline d}:=\big(\hat M_{\ell,0}\vec d^{\,(2)}+\hat M_{\ell,1}\vec c|_{\hat\Lambda_\ell}\big)|_{\underline{\hat\Pi}}$
  `eval`$A(\ell+1,\underline{\check\Pi},\check\Lambda_{\ell+1\uparrow},\underline{\hat\Pi},\hat\Lambda_{\ell+1\uparrow},\vec{\underline d},\vec c|_{\hat\Lambda_{\ell+1\uparrow}})$
  % With $\underline v:=\vec{\underline d}^{\,\top}\hat\Phi|_{\underline{\hat\Pi}}+\vec c|_{\hat\Lambda_{\ell+1\uparrow}}^{\top}\hat\Psi|_{\hat\Lambda_{\ell+1\uparrow}}$, it returns $a(\check\Phi|_{\underline{\check\Pi}},\underline v)$, $\boxed{a(\check\Psi|_{\check\Lambda_{\ell+1\uparrow}},\underline v)}$.
  compute $\boxed{\big(\check M_{\ell,0}^{\top}a(\check\Phi|_{\underline{\check\Pi}},\underline v)\big)|_{\check\Pi^{(2)}}+a(\check\Phi|_{\check\Pi^{(2)}},\hat\Phi|_{\hat\Pi^{(1)}})\vec d^{\,(1)}}$
  compute $\boxed{\big(\check M_{\ell,1}^{\top}a(\check\Phi|_{\underline{\check\Pi}},\underline v)\big)|_{\check\Lambda_\ell}}$
`endif`

Theorem 2.5. *A call of* `eval`A *yields the output as specified in its definition, at the cost of* $\mathcal{O}(\#\check\Pi+\#\check\Lambda+\#\hat\Pi+\#\hat\Lambda)$ *operations.*

Remark 2.6. Recalling that $\check\Phi_0=\check\Psi_0$ and $\hat\Phi_0=\hat\Psi_0$, for finite trees $\check\Lambda\subset\check\nabla$ and $\hat\Lambda\subset\hat\nabla$, `eval`$A$ shows in particular how to apply $A|_{\check\Lambda\times\hat\Lambda}$ in $\mathcal{O}(\#\check\Lambda+\#\hat\Lambda)$ operations.

Proof. By definition of $\underline{\check\Pi}$, we have

$$\#\underline{\check\Pi}\lesssim\#\check\Lambda_\ell+\#\check\Pi^{(2)}\lesssim\#\check\Lambda_\ell+\#\hat\Lambda_\ell,$$

and analogously $\#\underline{\hat\Pi}\lesssim\#\hat\Lambda_\ell+\#\check\Lambda_\ell$. We infer that after sufficiently many recursive calls of `eval`A either of the current sets $\check\Pi\cup\check\Lambda$ or $\hat\Pi\cup\hat\Lambda$ will be empty, in which case that call returns the empty set as required.

Next, we observe that the desired output of `eval`A can be decomposed as follows

$$a(\check\Phi|_{\check\Pi},v)=\boxed{a(\check\Phi|_{\check\Pi^{(1)}},v)}+\boxed{a(\check\Phi|_{\check\Pi^{(2)}},v)},\ a(\check\Psi|_{\check\Lambda},v)=\boxed{a(\check\Psi|_{\check\Lambda_{\ell+1\uparrow}},v)}+\boxed{a(\check\Psi|_{\check\Lambda_\ell},v)}.$$

By definition of $\check{\Pi}^{(1)}$, and from $\hat{\Lambda}$ being a tree, one has

$$\boxed{a(\check{\Phi}|_{\check{\Pi}^{(1)}}, v)} = a(\check{\Phi}|_{\check{\Pi}^{(1)}}, \vec{d}^{\top}\hat{\Phi}|_{\hat{\Pi}}) = \boxed{a(\check{\Phi}|_{\check{\Pi}^{(1)}}, \hat{\Phi}|_{\hat{\Pi}})\vec{d}}.$$

The definition of $\vec{d}$ and $\underline{\hat{\Pi}}$ shows that

$$\underline{v} = \underline{\vec{d}}^{\top}\hat{\Phi}|_{\underline{\hat{\Pi}}} + \vec{c}|^{\top}_{\hat{\Lambda}_{\ell+1\uparrow}}\hat{\Psi}|_{\hat{\Lambda}_{\ell+1\uparrow}} = (\vec{d}^{(2)})^{\top}\hat{\Phi}|_{\hat{\Pi}^{(2)}} + \vec{c}^{\top}\hat{\Psi}|_{\hat{\Lambda}} = v - (\vec{d}^{(1)})^{\top}\hat{\Phi}|_{\hat{\Pi}^{(1)}}.$$

By induction, the recursive call yields

$$a(\check{\Phi}|_{\underline{\check{\Pi}}}, \underline{v}), \quad \boxed{a(\check{\Psi}|_{\check{\Lambda}_{\ell+1\uparrow}}, \underline{v})} = \boxed{a(\check{\Psi}|_{\check{\Lambda}_{\ell+1\uparrow}}, v)},$$

where the latter equality follows by definition of $\hat{\Pi}^{(1)}$, and from $\check{\Lambda}$ being a tree. The definition of $\underline{\check{\Pi}}$ shows that

$$\check{\Phi}|_{\check{\Pi}^{(2)}} = (\check{M}^{\top}_{\ell,0}\check{\Phi}|_{\underline{\check{\Pi}}})|_{\check{\Pi}^{(2)}}, \quad \check{\Psi}|_{\check{\Lambda}_\ell} = (\check{M}^{\top}_{\ell,1}\check{\Phi}|_{\underline{\check{\Pi}}})|_{\check{\Lambda}_\ell}.$$

We conclude that

$$\begin{aligned}\boxed{a(\check{\Phi}|_{\check{\Pi}^{(2)}}, v)} &= a(\check{\Phi}|_{\check{\Pi}^{(2)}}, \underline{v}) + a(\check{\Phi}|_{\check{\Pi}^{(2)}}, (\vec{d}^{(1)})^{\top}\hat{\Phi}|_{\hat{\Pi}^{(1)}}) \\ &= \boxed{\big(\check{M}^{\top}_{\ell,0}a(\check{\Phi}|_{\underline{\check{\Pi}}}, \underline{v})\big)|_{\check{\Pi}^{(2)}} + a(\check{\Phi}|_{\check{\Pi}^{(2)}}, \hat{\Phi}|_{\hat{\Pi}^{(1)}})\vec{d}^{(1)}}.\end{aligned}$$

From $a(\check{\Psi}|_{\check{\Lambda}_\ell}, \hat{\Phi}|_{\hat{\Pi}^{(1)}}) = 0$ by definition of $\hat{\Pi}^{(1)}$, we have

$$\boxed{a(\check{\Psi}|_{\check{\Lambda}_\ell}, v)} = a(\check{\Psi}|_{\check{\Lambda}_\ell}, \underline{v}) = \boxed{\big(\check{M}^{\top}_{\ell,1}a(\check{\Phi}|_{\underline{\check{\Pi}}}, \underline{v})\big)|_{\check{\Lambda}_\ell}}.$$

Since $\check{\Pi} = \check{\Pi}^{(1)} \cup \check{\Pi}^{(2)}$ and $\check{\Lambda} = \check{\Lambda}_\ell \cup \check{\Lambda}_{\ell+1\uparrow}$, we conclude the first statement of the theorem.

From the assumptions on the collections $\check{\Phi}$, $\check{\Psi}$, $\hat{\Phi}$, and $\hat{\Psi}$, and their consequences on the sparsity of the matrices $\check{M}_{\ell,0}$, $\check{M}_{\ell,1}$, $\hat{M}_{\ell,0}$, and $\hat{M}_{\ell,1}$, one easily infers that the total cost of the evaluations of the statements in `eval`A is $\mathcal{O}(\#\check{\Pi}+\#\check{\Lambda}_\ell+\#\hat{\Pi}+\#\hat{\Lambda}_\ell)$ plus the cost of the recursive call. Using $\#\underline{\check{\Pi}} + \#\underline{\hat{\Pi}} \lesssim \#\check{\Lambda}_\ell + \#\hat{\Lambda}_\ell$ and induction, we conclude the second statement of the theorem. □

The following two algorithms concern the application of $U|_{\check{\Lambda}\times\hat{\Lambda}}$ and $L|_{\check{\Lambda}\times\hat{\Lambda}}$ for trees $\check{\Lambda} \subset \check{\nabla}$ and $\hat{\Lambda} \subset \hat{\nabla}$. Again, because the algorithms will be recursive in the coarsest level, they shall be able to perform somewhat more general tasks.

`eval`$U(\ell, \check{\Pi}, \check{\Lambda}, \hat{\Pi}, \hat{\Lambda}, \vec{d}, \vec{c})$
% *Input:* $\ell \in \mathbb{N}$, $\check{\Pi} \subset \check{\Delta}_{\ell-1}$, $\hat{\Pi} \subset \hat{\Delta}_{\ell-1}$, *$\ell$-trees* $\check{\Lambda} \subset \check{\nabla}_{\ell\uparrow}$ *and* $\hat{\Lambda} \subset \hat{\nabla}_{\ell\uparrow}$, *all finite,*
% $\vec{d} := (d_\lambda)_{\lambda\in\hat{\Pi}}$, $\vec{c} := (c_\lambda)_{\lambda\in\hat{\Lambda}} \subset \mathbb{R}$.
% *Output: With* $v := \vec{d}^{\top}\hat{\Phi}|_{\hat{\Pi}} + \vec{c}^{\top}\hat{\Psi}|_{\hat{\Lambda}}$, *the concatenation of the vectors inside*
% *the dashed boxes gives*
% $a(\check{\Phi}|_{\check{\Pi}}, v)$, $U|_{\check{\Lambda}\times\hat{\Lambda}}\vec{c}$.
% *Let* $\check{\Pi}^{(1)} := \{\lambda \in \check{\Pi} : |\operatorname{supp}\check{\phi}_\lambda \cap \operatorname{supp}\hat{\psi}_\mu| = 0 \ \forall \mu \in \hat{\Lambda}_\ell\}$, $\check{\Pi}^{(2)} := \check{\Pi} \setminus \check{\Pi}^{(1)}$,
% *and let* $\underline{\check{\Pi}} \subset \check{\Delta}_\ell$, $\underline{\hat{\Pi}} \subset \hat{\Delta}_\ell$ *be the smallest collections with*
% $\operatorname{span}\check{\Phi}|_{\underline{\check{\Pi}}} \supset \operatorname{span}\check{\Phi}|_{\check{\Pi}^{(2)}} \cup \check{\Psi}_{\check{\Lambda}_\ell}$, *or* $\operatorname{span}\hat{\Phi}|_{\underline{\hat{\Pi}}} \supset \operatorname{span}\hat{\Psi}_{\hat{\Lambda}_\ell}$.

`if` $\check{\Pi} \cup \check{\Lambda} \neq \emptyset$ `and` $\hat{\Pi} \cup \hat{\Lambda} \neq \emptyset$ `then`
compute $\boxed{a(\check{\Phi}|_{\check{\Pi}^{(1)}}, \hat{\Phi}|_{\hat{\Pi}})\vec{d}}$

$\underline{\vec{d}} := \big(\hat{M}_{\ell,1}\vec{c}|_{\hat{\Lambda}_\ell}\big)|_{\underline{\hat{\Pi}}}$
`eval`$U(\ell+1, \underline{\check{\Pi}}, \check{\Lambda}_{\ell+1\uparrow}, \underline{\hat{\Pi}}, \hat{\Lambda}_{\ell+1\uparrow}, \underline{\vec{d}}, \vec{c}|_{\hat{\Lambda}_{\ell+1\uparrow}})$
% With $\underline{v} := \underline{\vec{d}}^\top \hat{\Phi}|_{\underline{\hat{\Pi}}} + \vec{c}|_{\hat{\Lambda}_{\ell+1\uparrow}}^\top \hat{\Psi}|_{\hat{\Lambda}_{\ell+1\uparrow}}$, it returns $a(\check{\Phi}|_{\underline{\check{\Pi}}}, \underline{v})$, $U|_{\check{\Lambda}_{\ell+1\uparrow}\times\hat{\Lambda}_{\ell+1\uparrow}}\vec{c}|_{\hat{\Lambda}_{\ell+1\uparrow}}$.
compute $\big(\check{M}_{\ell,0}^\top a(\check{\Phi}|_{\underline{\check{\Pi}}}, \underline{v})\big)|_{\check{\Pi}^{(2)}} + a(\check{\Phi}|_{\check{\Pi}^{(2)}}, \hat{\Phi}|_{\hat{\Pi}})\vec{d}$
compute $\big(\check{M}_{\ell,1}^\top a(\check{\Phi}|_{\underline{\check{\Pi}}}, \underline{v})\big)|_{\check{\Lambda}_\ell}$
`endif`

Theorem 2.7. *A call of* `eval`U *yields the output as specified in its definition, at the cost of* $\mathcal{O}(\#\hat{\Pi} + \#\hat{\Lambda})$ *operations.*

Remark 2.8. For finite trees $\check{\Lambda} \subset \check{\nabla}$ and $\hat{\Lambda} \subset \hat{\nabla}$, `eval`$U$ shows in particular how to apply $U|_{\check{\Lambda}\times\hat{\Lambda}}$ in $\mathcal{O}(\#\check{\Lambda} + \#\hat{\Lambda})$ operations.

Proof. From $\#\underline{\hat{\Pi}} \lesssim \#\hat{\Lambda}_\ell$, we infer that after sufficiently many recursive calls of `eval`U either of the current sets $\check{\Pi} \cup \check{\Lambda}$ or $\hat{\Pi} \cup \hat{\Lambda}$ will be empty, in which case that call returns the empty set as required.

The desired output of `eval`U can be decomposed as follows:

$$a(\check{\Phi}|_{\check{\Pi}}, v) = \boxed{a(\check{\Phi}|_{\check{\Pi}^{(1)}}, v)} + \boxed{a(\check{\Phi}|_{\check{\Pi}^{(2)}}, v)},\quad U|_{\check{\Lambda}\times\hat{\Lambda}}\vec{c} = \boxed{U|_{\check{\Lambda}_\ell\times\hat{\Lambda}}\vec{c}} + \boxed{U|_{\check{\Lambda}_{\ell+1\uparrow}\times\hat{\Lambda}}\vec{c}}.$$

By definition of $\check{\Pi}^{(1)}$, and from $\hat{\Lambda}$ being a tree, one has

$$\boxed{a(\check{\Phi}|_{\check{\Pi}^{(1)}}, v)} = a(\check{\Phi}|_{\check{\Pi}^{(1)}}, \vec{d}^\top \hat{\Phi}|_{\hat{\Pi}}) = \boxed{a(\check{\Phi}|_{\check{\Pi}^{(1)}}, \hat{\Phi}|_{\hat{\Pi}})\vec{d}}.$$

The definition of $\underline{\vec{d}}$ and $\underline{\hat{\Pi}}$ shows that

$$\underline{v} = \underline{\vec{d}}^\top \hat{\Phi}|_{\underline{\hat{\Pi}}} + \vec{c}|_{\hat{\Lambda}_{\ell+1\uparrow}}^\top \hat{\Psi}|_{\hat{\Lambda}_{\ell+1\uparrow}} = \vec{c}^\top \hat{\Psi}|_{\hat{\Lambda}} = v - \vec{d}^\top \hat{\Phi}|_{\hat{\Pi}}.$$

By induction, the recursive call yields

$$a(\check{\Phi}|_{\underline{\check{\Pi}}}, \underline{v}),\quad \boxed{U|_{\check{\Lambda}_{\ell+1\uparrow}\times\hat{\Lambda}_{\ell+1\uparrow}}\vec{c}|_{\hat{\Lambda}_{\ell+1\uparrow}}} = \boxed{U|_{\check{\Lambda}_{\ell+1\uparrow}\times\hat{\Lambda}}\vec{c}},$$

the latter equality by definition of U. The definition of $\underline{\check{\Pi}}$ shows that

$$\check{\Phi}|_{\check{\Pi}^{(2)}} = (\check{M}_{\ell,0}^\top \check{\Phi}|_{\underline{\check{\Pi}}})|_{\check{\Pi}^{(2)}},\quad \check{\Psi}|_{\check{\Lambda}_\ell} = (\check{M}_{\ell,1}^\top \check{\Phi}|_{\underline{\check{\Pi}}})|_{\check{\Lambda}_\ell}.$$

We conclude that

$$\begin{aligned}\boxed{a(\check{\Phi}|_{\check{\Pi}^{(2)}}, v)} &= a(\check{\Phi}|_{\check{\Pi}^{(2)}}, \underline{v}) + a(\check{\Phi}|_{\check{\Pi}^{(2)}}, \vec{d}^\top \hat{\Phi}|_{\hat{\Pi}}) \\ &= \boxed{\big(\check{M}_{\ell,0}^\top a(\check{\Phi}|_{\underline{\check{\Pi}}}, \underline{v})\big)|_{\check{\Pi}^{(2)}} + a(\check{\Phi}|_{\check{\Pi}^{(2)}}, \hat{\Phi}|_{\hat{\Pi}})\vec{d}}.\end{aligned}$$

By definition of U, we have

$$\boxed{U|_{\check{\Lambda}_\ell\times\hat{\Lambda}}\vec{c}} = a(\check{\Psi}|_{\check{\Lambda}_\ell}, \underline{v}) = \boxed{\big(\check{M}_{\ell,1}^\top a(\check{\Phi}|_{\underline{\check{\Pi}}}, \underline{v})\big)|_{\check{\Lambda}_\ell}}.$$

Since $\check{\Pi} = \check{\Pi}^{(1)} \cup \check{\Pi}^{(2)}$ and $\check{\Lambda} = \check{\Lambda}_\ell \cup \check{\Lambda}_{\ell+1\uparrow}$, we conclude the first statement of the theorem.

The computations of $a(\check{\Phi}|_{\check{\Pi}^{(1)}}, v)$, $\underline{\vec{d}}$, and $a(\check{\Phi}|_{\check{\Pi}^{(2)}}, v)$ take the order of $\#\hat{\Pi}$, $\#\hat{\Lambda}_\ell$, and $\#\check{\Pi}^{(2)} \lesssim \#\hat{\Lambda}_\ell$ operations. The induction hypothesis is that the recursive call takes the order of $\#\underline{\hat{\Pi}} + \#\hat{\Lambda}_{\ell+1\uparrow}$ operations.

Above bounds show in particular that $\#a(\check{\Phi}|_{\check{\Pi}}, v) \lesssim \#\hat{\Pi} + \#\hat{\Lambda}_\ell$, and so similarly, $\#a(\check{\Phi}|_{\underline{\check{\Pi}}}, \underline{v}) \lesssim \#\underline{\hat{\Pi}} + \#\hat{\Lambda}_{\ell+1}$. From this we conclude that the computation of $U|_{\check{\Lambda}_{\ell+1\uparrow}\times\hat{\Lambda}_{\ell+1\uparrow}}\vec{c}|_{\hat{\Lambda}_{\ell+1\uparrow}}$ takes the order of $\#\underline{\hat{\Pi}} + \#\hat{\Lambda}_{\ell+1}$ operations. Using that $\#\underline{\hat{\Pi}} \lesssim \#\hat{\Lambda}_\ell$, we now infer the second statement of the theorem. □

`eval`$L(\ell, \check{\Lambda}, \hat{\Pi}, \hat{\Lambda}, \vec{d}, \vec{c})$
% *Input:* $\ell \in \mathbb{N}$, $\hat{\Pi} \subset \hat{\Delta}_{\ell-1}$, *trees* $\check{\Lambda} \subset \check{\nabla}_{\ell\uparrow}$ *and* $\hat{\Lambda} \subset \hat{\nabla}_{\ell\uparrow}$, *all finite,*
% $\vec{d} := (d_\lambda)_{\lambda\in\hat{\Pi}}, \vec{c} := (c_\lambda)_{\lambda\in\hat{\Lambda}} \subset \mathbb{R}$.
% *Output: The concatenation of the vectors inside the dashed boxes gives*
% $a(\check{\Psi}|_{\check{\Lambda}}, \hat{\Phi}|_{\hat{\Pi}})\vec{d} + L|_{\check{\Lambda}\times\hat{\Lambda}}\vec{c}$.
% *Let* $\hat{\Pi}^{(1)} := \{\lambda \in \hat{\Pi} : |\operatorname{supp}\hat{\phi}_\lambda \cap \operatorname{supp}\check{\psi}_\mu| = 0 \ \forall \mu \in \check{\Lambda}_\ell\}$, $\hat{\Pi}^{(2)} := \hat{\Pi} \setminus \hat{\Pi}^{(1)}$,
% *and* $\vec{d}^{(2)} := \vec{d}|_{\hat{\Pi}^{(2)}}$.
% *Let* $\underline{\check{\Pi}} \subset \check{\Delta}_\ell$, $\underline{\hat{\Pi}}, \underline{\hat{\Pi}} \subset \hat{\Delta}_\ell$ *be the smallest collections with*
% $\operatorname{span}\check{\Phi}|_{\underline{\check{\Pi}}} \supset \operatorname{span}\check{\Psi}|_{\check{\Lambda}_\ell}$, $\operatorname{span}\hat{\Phi}|_{\underline{\hat{\Pi}}} \supset \operatorname{span}\hat{\Phi}|_{\hat{\Pi}^{(2)}}$, *or* $\operatorname{span}\hat{\Phi}|_{\underline{\hat{\Pi}}} \supset \operatorname{span}\hat{\Phi}|_{\hat{\Pi}^{(2)}} \cup \hat{\Psi}_{\hat{\Lambda}_\ell}$.

`if` $\check{\Lambda} \neq \emptyset$ `and` $\hat{\Pi} \cup \hat{\Lambda} \neq \emptyset$ `then`
  $\underline{\vec{d}} := \hat{M}_{\ell,0}\vec{d}^{(2)}$
  compute $\boxed{(\check{M}_{\ell,1}^\top a(\check{\Phi}|_{\underline{\check{\Pi}}}, \hat{\Phi}|_{\underline{\hat{\Pi}}})\underline{\vec{d}})|_{\check{\Lambda}_\ell}}$
  $\underline{\vec{d}} := \underline{\vec{d}} + \hat{M}_{\ell,1}\vec{c}|_{\hat{\Lambda}_\ell}$
  `eval`$L(\ell+1, \check{\Lambda}_{\ell+1\uparrow}, \underline{\hat{\Pi}}, \hat{\Lambda}_{\ell+1\uparrow}, \underline{\vec{d}}, \vec{c}|_{\hat{\Lambda}_{\ell+1\downarrow}})$
  % It returns $\boxed{a(\check{\Psi}|_{\check{\Lambda}_{\ell+1\uparrow}}, \hat{\Phi}|_{\underline{\hat{\Pi}}})\underline{\vec{d}} + L|_{\check{\Lambda}_{\ell+1\uparrow}\times\hat{\Lambda}_{\ell+1\uparrow}}\vec{c}|_{\hat{\Lambda}_{\ell+1\uparrow}}}$.
`endif`

Theorem 2.9. *A call of* `eval`L *yields the output as specified in its definition, at the cost of* $\mathcal{O}(\#\check{\Lambda} + \#\hat{\Pi} + \#\hat{\Lambda})$ *operations.*

Remark 2.10. For finite trees $\check{\Lambda} \subset \check{\nabla}$ and $\hat{\Lambda} \subset \hat{\nabla}$, `eval`$L$ shows in particular how to apply $L|_{\check{\Lambda}\times\hat{\Lambda}}$ in $\mathcal{O}(\#\check{\Lambda} + \#\hat{\Lambda})$ operations.

Proof. After sufficiently many recursive calls of `eval`L either of the current sets $\check{\Lambda}$ or $\hat{\Pi} \cup \hat{\Lambda}$ will be empty, in which case that call returns the empty set as required.

The desired output of `eval`L can be decomposed as follows:

$$a(\check{\Psi}|_{\check{\Lambda}},\ \hat{\Phi}|_{\hat{\Pi}})\vec{d} + L|_{\check{\Lambda}\times\hat{\Lambda}}\vec{c} = \boxed{a(\check{\Psi}|_{\check{\Lambda}_\ell}, \hat{\Phi}|_{\hat{\Pi}})\vec{d} + L|_{\check{\Lambda}_\ell\times\hat{\Lambda}}\vec{c}} + \boxed{a(\check{\Psi}|_{\check{\Lambda}_{\ell+1\uparrow}}, \hat{\Phi}|_{\hat{\Pi}})\vec{d} + L|_{\check{\Lambda}_{\ell+1\uparrow}\times\hat{\Lambda}}\vec{c}}.$$

From $L|_{\check{\Lambda}_\ell\times\hat{\Lambda}}\vec{c} = 0$, we have that

$$\boxed{a(\check{\Psi}|_{\check{\Lambda}_\ell}, \hat{\Phi}|_{\hat{\Pi}})\vec{d} + L|_{\check{\Lambda}_\ell\times\hat{\Lambda}}\vec{c}} = a(\check{\Psi}|_{\check{\Lambda}_\ell}, \hat{\Phi}|_{\hat{\Pi}^{(2)}})\vec{d}^{(2)} = a((\check{M}_{\ell,1}^\top\check{\Phi}|_{\underline{\check{\Pi}}})|_{\check{\Lambda}_\ell}, \hat{M}_{\ell,0}^\top\hat{\Phi}|_{\underline{\hat{\Pi}}})\vec{d}^{(2)}$$
$$= \boxed{(\check{M}_{\ell,1}^\top a(\check{\Phi}|_{\underline{\check{\Pi}}}, \hat{\Phi}|_{\underline{\hat{\Pi}}})\underline{\vec{d}})|_{\check{\Lambda}_\ell}}.$$

By induction, the recursive call yields $\boxed{a(\check{\Psi}|_{\check{\Lambda}_{\ell+1\uparrow}},\hat{\Phi}|_{\underline{\hat{\Pi}}})\underline{\vec{d}}+L|_{\check{\Lambda}_{\ell+1\uparrow}\times\hat{\Lambda}_{\ell+1\uparrow}}\vec{c}|_{\hat{\Lambda}_{\ell+1\uparrow}}}$, the first term being equal to

$$a(\check{\Psi}|_{\check{\Lambda}_{\ell+1\uparrow}},\hat{\Phi}|_{\hat{\Pi}^{(2)}})\vec{d}^{(2)}+a(\check{\Psi}|_{\check{\Lambda}_{\ell+1\uparrow}},\hat{\Psi}|_{\hat{\Lambda}_\ell})\vec{c}|_{\hat{\Lambda}_\ell}=a(\check{\Psi}|_{\check{\Lambda}_{\ell+1\uparrow}},\hat{\Phi}|_{\hat{\Pi}})\vec{d}+L|_{\check{\Lambda}_{\ell+1\uparrow}\times\hat{\Lambda}_\ell}\vec{c}|_{\hat{\Lambda}_\ell},$$

and so in total being equal to $\boxed{a(\check{\Psi}|_{\check{\Lambda}_{\ell+1\uparrow}},\hat{\Phi}|_{\hat{\Pi}})\vec{d}+L|_{\check{\Lambda}_{\ell+1\uparrow}\times\hat{\Lambda}}\vec{c}}$. From $\check{\Lambda}=\check{\Lambda}_\ell\cup\check{\Lambda}_{\ell+1\uparrow}$, we conclude the first statement of the theorem.

The statement about the cost is proven similarly to that from Theorem 2.5. □

Remark 2.11. Since $A=U+L$, actually there is no strict need for a separate routine `eval`A. The cost of one `eval`A application is, however, lower than the sum of the costs of the corresponding applications of `eval`U and `eval`L.

3. Tensor products

With, for $n\geq 2$, $\cup\in\{\check{\cup},\hat{\cup}\}$, $\check{\boldsymbol{\Psi}}:=\otimes_{i=1}^n\check{\Psi}=\{\check{\boldsymbol{\psi}}_{\boldsymbol{\lambda}}:=\otimes_{i=1}^n\check{\psi}_{\lambda_i}:\boldsymbol{\lambda}\in\check{\boldsymbol{\nabla}}:=\prod_{i=1}^n\check{\nabla}\}$, on span $\check{\boldsymbol{\Psi}}\times$ span $\hat{\boldsymbol{\Psi}}$, we set

$$\mathbf{a}(u_1\otimes\cdots\otimes u_n,v_1\otimes\cdots\otimes v_n)=\prod_{i=1}^n a(u_i,v_i),$$

so that $\mathbf{a}(\check{\boldsymbol{\Psi}},\hat{\boldsymbol{\Psi}})=A\otimes\cdots\otimes A$.

Although for notational simplicity, we consider collections $\check{\Psi}$ and $\hat{\Psi}$, a bilinear form $a(\cdot,\cdot)$, and so a matrix A that are independent of i, our results immediately generalize to the case where they depend on i.

To define a suitable tree structure on a multi-index set, for $\mathbf{t}\in\{0,1\}^n$, and with $\mathbb{K}$ some set that contains $\check{\Lambda}\cup\hat{\Lambda}$, let $P_{\mathbf{t}}:\mathbb{K}^n\to\mathbb{K}^{|\mathbf{t}|_1}$ denote the restriction of a vector in $\mathbb{K}^n$ to the coordinates that correspond to the non-zero entries of $\mathbf{t}$. As usual, $\neg\mathbf{t}$ will denote the vector in $\mathbb{R}^n$ with, for $1\leq i\leq n$, its ith coordinate equal to $1-t_i$.

For $\boldsymbol{\nabla}=\prod_{i=1}^n\breve{\nabla}_i$, where each $\breve{\nabla}_i$ is either $\check{\nabla}$ or $\hat{\nabla}$, generally different for different i, we call $\boldsymbol{\Lambda}\subset\boldsymbol{\nabla}$ a *multi-tree* when for all $1\leq i\leq n$ and all $\boldsymbol{\mu}\in P_{\neg\mathbf{e}_i}\boldsymbol{\Lambda}$, the fiber $P_{\mathbf{e}_i}\big(P_{\neg\mathbf{e}_i}|_{\boldsymbol{\Lambda}}\big)^{-1}\{\boldsymbol{\mu}\}$ is a tree. That is, $\boldsymbol{\Lambda}$ is a multi-tree when "frozen" in any $n-1$ coordinates, at any value of these coordinates, it is a tree in the remaining coordinate in either $\check{\nabla}$ or $\hat{\nabla}$.

Consequently, for $\boldsymbol{\Lambda}$ being a multi-tree, and any $\mathbf{0}\neq\mathbf{t}\in\{0,1\}^n$, and $\boldsymbol{\mu}\in P_{\neg\mathbf{t}}\boldsymbol{\Lambda}$,

$$\boldsymbol{\Lambda}_{\mathbf{t},\boldsymbol{\mu}}:=P_{\mathbf{t}}\big(P_{\neg\mathbf{t}}|_{\boldsymbol{\Lambda}}\big)^{-1}\{\boldsymbol{\mu}\}$$

is a multi-tree. That is, $\boldsymbol{\Lambda}$, frozen in any k coordinates, where $1\leq k\leq n$, at any value of these coordinates, is a multi-tree in the remaining coordinates.

Finally, from $\boldsymbol{\Lambda}=\cup_{\boldsymbol{\mu}\in P_{\neg\mathbf{t}}\boldsymbol{\Lambda}}\big(P_{\neg\mathbf{t}}|_{\boldsymbol{\Lambda}}\big)^{-1}\{\boldsymbol{\mu}\}$, we have $P_{\mathbf{t}}\boldsymbol{\Lambda}=\cup_{\boldsymbol{\mu}\in P_{\neg\mathbf{t}}\boldsymbol{\Lambda}}\boldsymbol{\Lambda}_{\mathbf{t},\boldsymbol{\mu}}$, which, being a union of multi-trees, is a multi-tree itself.

For index sets $\triangleleft\subset\Diamond$, let $I_{\triangleleft}^{\Diamond}$ denote the extension operator with zeros of a vector supported on $\triangleleft$ to one on $\Diamond$, and let $R_{\triangleleft}^{\Diamond}$ denotes its (formal) adjoint, being the restriction operator of a vector supported on $\Diamond$ to one on $\triangleleft$. Since the set $\Diamond$ will always be clear from the context, we will denote these operators by $I_{\triangleleft}$ and $R_{\triangleleft}$.

Theorem 3.1. *Let $\check{\boldsymbol{\Lambda}} \subset \check{\boldsymbol{\nabla}}$, $\hat{\boldsymbol{\Lambda}} \subset \hat{\boldsymbol{\nabla}}$ be finite multi-trees. Define*

$$\boldsymbol{\Sigma} = \bigcup_{\lambda \in P_{\mathbf{e}_1}\hat{\boldsymbol{\Lambda}}} \{\lambda\} \times \big\{\check{\boldsymbol{\Lambda}}_{\neg \mathbf{e}_1,\mu} : \mu \in P_{\mathbf{e}_1}\check{\boldsymbol{\Lambda}},\, |\mu| = |\lambda|+1,\, |\operatorname{supp}\check{\psi}_\mu \cap \hat{S}_\lambda| > 0\big\},$$

$$\boldsymbol{\Theta} = \bigcup_{\boldsymbol{\lambda} \in P_{\neg \mathbf{e}_1}\hat{\boldsymbol{\Lambda}}} \{\mu \in P_{\mathbf{e}_1}\check{\boldsymbol{\Lambda}} : \exists \gamma \in \hat{\boldsymbol{\Lambda}}_{\mathbf{e}_1,\boldsymbol{\lambda}} \text{ s.t. } |\gamma| = |\mu|,\, |\check{S}_\mu \cap \operatorname{supp}\hat{\psi}_\gamma| > 0\} \times \{\boldsymbol{\lambda}\}.$$

Then $\boldsymbol{\Sigma}$, $\boldsymbol{\Theta}$ are multi-trees with $\#\boldsymbol{\Sigma} \lesssim \#\check{\boldsymbol{\Lambda}}$ and $\#\boldsymbol{\Theta} \lesssim \#\hat{\boldsymbol{\Lambda}}$, and

$$\mathbf{a}(\check{\boldsymbol{\Psi}}|_{\check{\boldsymbol{\Lambda}}}, \hat{\boldsymbol{\Psi}}|_{\hat{\boldsymbol{\Lambda}}}) = \tag{3.1}$$

$$\begin{aligned} &\big[R_{\check{\boldsymbol{\Lambda}}_{\mathbf{e}_1,\boldsymbol{\mu}}} L I_{\boldsymbol{\Sigma}_{\mathbf{e}_1,\boldsymbol{\mu}}}\big]_{\boldsymbol{\mu} \in P_{\neg \mathbf{e}_1}\boldsymbol{\Sigma}} \otimes \mathrm{Id} \otimes \cdots \otimes \mathrm{Id} \\ &\quad \circ \mathrm{Id} \otimes \big[R_{\boldsymbol{\Sigma}_{\neg \mathbf{e}_1,\lambda}} (A \otimes \cdots \otimes A) I_{\hat{\boldsymbol{\Lambda}}_{\neg \mathbf{e}_1,\lambda}}\big]_{\lambda \in P_{\mathbf{e}_1}\hat{\boldsymbol{\Lambda}}} \quad + \end{aligned} \tag{3.2}$$

$$\begin{aligned} &\mathrm{Id} \otimes \big[R_{\check{\boldsymbol{\Lambda}}_{\neg \mathbf{e}_1,\lambda}} (A \otimes \cdots \otimes A) I_{\boldsymbol{\Theta}_{\neg \mathbf{e}_1,\lambda}}\big]_{\lambda \in P_{\mathbf{e}_1}\boldsymbol{\Theta}} \\ &\quad \circ \big[R_{\boldsymbol{\Theta}_{\mathbf{e}_1,\boldsymbol{\lambda}}} U I_{\hat{\boldsymbol{\Lambda}}_{\mathbf{e}_1,\boldsymbol{\lambda}}}\big]_{\boldsymbol{\lambda} \in P_{\neg \mathbf{e}_1}\hat{\boldsymbol{\Lambda}}} \otimes \mathrm{Id} \otimes \cdots \otimes \mathrm{Id}. \end{aligned} \tag{3.3}$$

The recursive procedure suggested by (3.1)–(3.3) *to evaluate the application of $\mathbf{a}(\check{\boldsymbol{\Psi}}|_{\check{\boldsymbol{\Lambda}}}, \hat{\boldsymbol{\Psi}}|_{\hat{\boldsymbol{\Lambda}}})$ by means of calls of* `evalU`, `evalL`, *and* `evalA` *requires $\mathcal{O}(\#\check{\boldsymbol{\Lambda}} + \#\hat{\boldsymbol{\Lambda}})$ operations.*

Proof. We write

$$\begin{aligned} \mathbf{a}(\check{\boldsymbol{\Psi}}|_{\check{\boldsymbol{\Lambda}}}, \hat{\boldsymbol{\Psi}}|_{\hat{\boldsymbol{\Lambda}}}) = &R_{\check{\boldsymbol{\Lambda}}}((U+L) \otimes A \otimes \cdots \otimes A) I_{\hat{\boldsymbol{\Lambda}}} = \\ &R_{\check{\boldsymbol{\Lambda}}}(L \otimes \mathrm{Id} \otimes \cdots \otimes \mathrm{Id})(\mathrm{Id} \otimes A \otimes \cdots \otimes A) I_{\hat{\boldsymbol{\Lambda}}} \quad + \end{aligned} \tag{3.4}$$

$$R_{\check{\boldsymbol{\Lambda}}}(\mathrm{Id} \otimes A \otimes \cdots \otimes A)(U \otimes \mathrm{Id} \otimes \cdots \otimes \mathrm{Id}) I_{\hat{\boldsymbol{\Lambda}}}. \tag{3.5}$$

and will show that (3.4) is equal to (3.2), and (3.5) is equal to (3.3).

With the definition of $\boldsymbol{\Sigma}$ reading as

$$\bigcup_{\lambda \in P_{\mathbf{e}_1}\hat{\boldsymbol{\Lambda}}} \{\lambda\} \times \big\{\check{\boldsymbol{\Lambda}}_{\neg \mathbf{e}_1,\mu} : \mu \in P_{\mathbf{e}_1}\check{\boldsymbol{\Lambda}},\, |\mu| \geq |\lambda|+1,\, |\operatorname{supp}\check{\psi}_\mu \cap \operatorname{supp}\hat{\psi}_\lambda| > 0\big\}, \tag{3.6}$$

the definition of L shows that

$$\begin{aligned} &R_{\check{\boldsymbol{\Lambda}}}(L \otimes \mathrm{Id} \otimes \cdots \otimes \mathrm{Id})(\mathrm{Id} \otimes A \otimes \cdots \otimes A) I_{\hat{\boldsymbol{\Lambda}}} = \\ &R_{\check{\boldsymbol{\Lambda}}}(L \otimes \mathrm{Id} \otimes \cdots \otimes \mathrm{Id}) I_{\boldsymbol{\Sigma}} \big(\mathrm{Id} \otimes [R_{\boldsymbol{\Sigma}_{\neg \mathbf{e}_1,\lambda}} (A \otimes \cdots \otimes A) I_{\hat{\boldsymbol{\Lambda}}_{\neg \mathbf{e}_1,\lambda}}]_{\lambda \in P_{\mathbf{e}_1}\hat{\boldsymbol{\Lambda}}}\big). \end{aligned} \tag{3.7}$$

Of course this is still true when the condition $|\operatorname{supp}\check{\psi}_\mu \cap \operatorname{supp}\hat{\psi}_\lambda| > 0$ in (3.6) is replaced by $|\operatorname{supp}\check{\psi}_\mu \cap \hat{S}_\lambda| > 0$ since it only makes the set larger. Secondly, it holds that $\boldsymbol{\Sigma}$ does not change when in its definition the condition $|\mu| = |\lambda|+1$ is replaced by $|\mu| \geq |\lambda|+1$. To see this, note that, by definition of a multi-tree, for any $(\mu, \gamma_2, \ldots, \gamma_n) \in \check{\boldsymbol{\Lambda}}$ with $|\mu| > |\lambda|+1$ and $|\operatorname{supp}\check{\psi}_\mu \cap \hat{S}_\lambda| > 0$, there exists a $\tilde{\mu} \in \check{\nabla}$ with $|\tilde{\mu}| = |\lambda|+1$, $|\operatorname{supp}\check{\psi}_{\tilde{\mu}} \cap \hat{S}_\lambda| > 0$, and $(\tilde{\mu}, \gamma_2, \ldots, \gamma_n) \in \check{\boldsymbol{\Lambda}}$. We conclude that (3.7) holds for the actual definition of $\boldsymbol{\Sigma}$.

Knowing (3.7), from

$$R_{\check{\boldsymbol{\Lambda}}}(L \otimes \mathrm{Id} \otimes \cdots \otimes \mathrm{Id}) I_{\boldsymbol{\Sigma}} = \big[R_{\check{\boldsymbol{\Lambda}}_{\mathbf{e}_1,\boldsymbol{\mu}}} L I_{\boldsymbol{\Sigma}_{\mathbf{e}_1,\boldsymbol{\mu}}}\big]_{\boldsymbol{\mu} \in P_{\neg \mathbf{e}_1}\boldsymbol{\Sigma}} \otimes \mathrm{Id} \otimes \cdots \otimes \mathrm{Id}$$

we conclude that (3.4) is equal to (3.2).

Since for $\lambda \in P_{\mathbf{e}_1}\hat{\boldsymbol{\Lambda}}$, $\boldsymbol{\Sigma}_{\neg \mathbf{e}_1,\lambda}$ is a union of multi-trees $\check{\boldsymbol{\Lambda}}_{\neg \mathbf{e}_1,\mu}$, it is a multi-tree. In order to conclude that $\boldsymbol{\Sigma}$ is a multi-tree, it remains to verify that for $\boldsymbol{\mu} \in P_{\neg \mathbf{e}_1}\boldsymbol{\Sigma}$, $\boldsymbol{\Sigma}_{\mathbf{e}_1,\boldsymbol{\mu}}$ is a tree in $\hat{\nabla}$. Let $\lambda \in \boldsymbol{\Sigma}_{\mathbf{e}_1,\boldsymbol{\mu}}$ with $|\lambda| > 0$. Since $P_{\mathbf{e}_1}\hat{\boldsymbol{\Lambda}}$ is a tree,

$\operatorname{supp}\hat\psi_\lambda$ is covered by the supports of $\hat\psi_\gamma$ for some $\gamma\in P_{\mathbf{e}_1}\hat{\boldsymbol\Lambda}$ with $|\gamma|=|\lambda|-1$ and $|\operatorname{supp}\hat\psi_\lambda\cap\operatorname{supp}\hat\psi_\gamma|>0$. For each of those γ, we have $\hat S_\gamma\supset\hat S_\lambda$ by (2.8), and so, recalling that the condition $|\mu|=|\lambda|+1$ in the definition of $\boldsymbol\Sigma$ can be read as $|\mu|\geq|\lambda|+1$, we conclude that $(\gamma,\boldsymbol\mu)\in\boldsymbol\Sigma$, or $\gamma\in\boldsymbol\Sigma_{\mathbf{e}_1,\boldsymbol\mu}$.

For each $\mu\in\check\nabla$, the condition $|\mu|=|\lambda|+1$ and $|\hat S_\lambda\cap\operatorname{supp}\check\psi_\mu|>0$ is satisfied for a uniformly bounded number of $\lambda\in\hat\nabla$. Consequently,

$$\#\boldsymbol\Sigma\lesssim\sum_{\mu\in P_{\mathbf{e}_1}\check{\boldsymbol\Lambda}}\#\check{\boldsymbol\Lambda}_{\neg\mathbf{e}_1,\mu}=\#\check{\boldsymbol\Lambda}.$$

Noting that in the definition of $\boldsymbol\Theta$, the condition $|\gamma|=|\mu|$ can be read as $|\gamma|\geq|\mu|$ without changing $\boldsymbol\Theta$, by definition of U we have

$$\begin{aligned}&R_{\check{\boldsymbol\Lambda}}(\mathrm{Id}\otimes A\otimes\cdots\otimes A)(U\otimes\mathrm{Id}\otimes\cdots\otimes\mathrm{Id})I_{\hat{\boldsymbol\Lambda}}=\\&R_{\check{\boldsymbol\Lambda}}(\mathrm{Id}\otimes A\otimes\cdots\otimes A)I_{\boldsymbol\Theta}\big([R_{\boldsymbol\Theta_{\mathbf{e}_1,\boldsymbol\lambda}}UI_{\hat{\boldsymbol\Lambda}_{\mathbf{e}_1,\boldsymbol\lambda}}]_{\boldsymbol\lambda\in P_{\neg\mathbf{e}_1}\hat{\boldsymbol\Lambda}}\otimes\mathrm{Id}\otimes\cdots\otimes\mathrm{Id}\big).\end{aligned}$$

From

$$R_{\check{\boldsymbol\Lambda}}(\mathrm{Id}\otimes A\otimes\cdots\otimes A)I_{\boldsymbol\Theta}=\mathrm{Id}\otimes\big[R_{\check{\boldsymbol\Lambda}_{\neg\mathbf{e}_1,\lambda}}(A\otimes\cdots\otimes A)I_{\boldsymbol\Theta_{\neg\mathbf{e}_1,\lambda}}\big]_{\lambda\in P_{\mathbf{e}_1}\boldsymbol\Theta}$$

we see that (3.5) is equal to (3.3).

For $\boldsymbol\lambda\in P_{\neg\mathbf{e}_1}\hat{\boldsymbol\Lambda}$, we have $\#\boldsymbol\Theta_{\mathbf{e}_1,\boldsymbol\lambda}\lesssim\#\hat{\boldsymbol\Lambda}_{\mathbf{e}_1,\boldsymbol\lambda}$, and so $\#\boldsymbol\Theta\lesssim\#\hat{\boldsymbol\Lambda}$.

To show that $\boldsymbol\Theta$ is a multi-tree, let $(\mu,\lambda_2,\dots,\lambda_n)\in\boldsymbol\Theta$, $2\leq i\leq n$ and $|\lambda_i|>0$. By definition of $\boldsymbol\Theta$, there exists a $(\gamma,\lambda_2,\dots,\lambda_n)\in\hat{\boldsymbol\Lambda}$ with $|\gamma|=|\mu|$ and $|\check S_\mu\cap\operatorname{supp}\hat\psi_\gamma|>0$. Since $\hat{\boldsymbol\Lambda}$ is a multi-tree, $\operatorname{supp}\hat\psi_{\lambda_i}$ is covered by the supports of $\operatorname{supp}\hat\psi_{\tilde\lambda_i}$ for some $\tilde\lambda_i$ with $|\tilde\lambda_i|=|\lambda_i|-1$ and $(\gamma,\lambda_2,\dots,\tilde\lambda_i,\dots,\lambda_n)\in\hat{\boldsymbol\Lambda}$, and so $(\mu,\lambda_2,\dots,\tilde\lambda_i,\dots,\lambda_n)\in\boldsymbol\Theta$.

Now let $(\mu,\boldsymbol\lambda)\in\boldsymbol\Theta$ with $|\mu|>0$. Since $P_{\mathbf{e}_1}\check{\boldsymbol\Lambda}$ is a tree, $\operatorname{supp}\check\psi_\mu$ is covered by the supports of $\check\psi_{\tilde\mu}$ for some $\tilde\mu\in P_{\mathbf{e}_1}\check{\boldsymbol\Lambda}$ with $|\tilde\mu|=|\mu|-1$ and $|\operatorname{supp}\check\psi_\mu\cap\operatorname{supp}\check\psi_{\tilde\mu}|>0$. For each of those $\tilde\mu$, we have $\check S_{\tilde\mu}\supset\check S_\mu$ by (2.8), and so, recalling that the condition $|\gamma|=|\mu|$ in the definition of $\boldsymbol\Theta$ can be read as $|\gamma|\geq|\mu|$, we conclude that $(\tilde\mu,\boldsymbol\lambda)\in\boldsymbol\Theta$, or $\boldsymbol\Theta$ is a multi-tree.

In view of Theorem 2.7 and 2.9 concerning the cost of the application of U and L, the cost of the application of $\mathbf{a}(\check{\boldsymbol\Psi}|_{\check{\boldsymbol\Lambda}},\hat{\boldsymbol\Psi}|_{\hat{\boldsymbol\Lambda}})$ by means of (3.2) and (3.3) is bounded by some absolute multiple of

$$\begin{aligned}&\sum_{\boldsymbol\mu\in P_{\neg\mathbf{e}_1}\boldsymbol\Sigma}\#\check{\boldsymbol\Lambda}_{\mathbf{e}_1,\boldsymbol\mu}+\#\boldsymbol\Sigma_{\mathbf{e}_1,\boldsymbol\mu}+\sum_{\lambda\in P_{\mathbf{e}_1}\hat{\boldsymbol\Lambda}}\#\boldsymbol\Sigma_{\neg\mathbf{e}_1,\lambda}+\#\hat{\boldsymbol\Lambda}_{\neg\mathbf{e}_1,\lambda}+\\&\sum_{\lambda\in P_{\mathbf{e}_1}\boldsymbol\Theta}\#\check{\boldsymbol\Lambda}_{\neg\mathbf{e}_1,\lambda}+\#\boldsymbol\Theta_{\neg\mathbf{e}_1,\lambda}+\sum_{\boldsymbol\lambda\in P_{\neg\mathbf{e}_1}\hat{\boldsymbol\Lambda}}\#\boldsymbol\Theta_{\mathbf{e}_1,\boldsymbol\lambda}+\#\hat{\boldsymbol\Lambda}_{\mathbf{e}_1,\boldsymbol\lambda},\end{aligned}\tag{3.8}$$

where for the second and third sum we used induction, which is justified by the correctness of the statement for $n=2$ by an application of Theorem 2.5. The expression (3.8) is bounded by

$$\#\check{\boldsymbol\Lambda}+\#\boldsymbol\Sigma+\#\boldsymbol\Sigma+\#\hat{\boldsymbol\Lambda}+\#\check{\boldsymbol\Lambda}+\#\boldsymbol\Theta+\#\boldsymbol\Theta+\#\hat{\boldsymbol\Lambda}\lesssim\#\check{\boldsymbol\Lambda}+\#\hat{\boldsymbol\Lambda}.\quad\square$$

Remark 3.2. Since it is actually not needed that for $\boldsymbol\lambda\in P_{\neg\mathbf{e}_1}\boldsymbol\Theta$, $\boldsymbol\Theta_{\mathbf{e}_1,\boldsymbol\lambda}$ is a tree, $\check S_\mu$ in the definition of $\boldsymbol\Theta$ can be replaced by $\operatorname{supp}\check\psi_\mu$.

Remark 3.3. With proper data structures, the sets $\boldsymbol{\Sigma}$, $\boldsymbol{\Theta}$ from Theorem 3.1 can be *constructed* in $\mathcal{O}(\#\check{\boldsymbol{\Lambda}} + \#\hat{\boldsymbol{\Lambda}})$ operations. For $\boldsymbol{\lambda} \in P_{\neg \mathbf{e}_1}\hat{\boldsymbol{\Lambda}}$, the set $\boldsymbol{\Theta}_{\mathbf{e}_1,\boldsymbol{\lambda}}$ is most efficiently generated along with the "downward sweep" of the `evalU` routine.

4. Numerics

Our aim in this last section is twofold. Using the example of a two-dimensional elliptic PDE problem, we investigate the performance of the evaluation scheme from Theorem 3.1 and, moreover, highlight the usefulness of the concept of multi-trees within adaptive tensor product wavelet Galerkin schemes. Within this adaptive scheme to be explained below, the evaluation scheme will be used for both the solution of a finite Galerkin system ($\check{\boldsymbol{\Lambda}} = \hat{\boldsymbol{\Lambda}}$) as well as for the residual computation ($\check{\boldsymbol{\Lambda}} \supset \hat{\boldsymbol{\Lambda}}$). The set up of the problem will be such that multi-trees, which allow for local refinement, are actually required to realize the best possible approximation rate.

4.1. **A two-dimensional model problem.** We consider a second order elliptic PDE problem with non-constant, but separable coefficients on $\square := (0,1)^2$ which reads as follows: For $f \in H^{-1}(\square)$, find $u \in H^1_0(\square)$ such that

$$\bar{\boldsymbol{a}}(u,v) := \sum_{i=1,2} \boldsymbol{a}_i(u,v) := \sum_{i=1,2} \int_{\square} \boldsymbol{p}_i\, \partial_i u\, \partial_i v = f(v), \quad \forall v \in H^1_0(\square). \tag{4.1}$$

For our convenience, we choose $\boldsymbol{p}_i(x_1,x_2) \equiv \boldsymbol{p}(x_1,x_2) := p(x_1)\,p(x_2)$ for $p(x_i) := (x_i - \frac{1}{2})^2 + 1$ ($i \in \{1,2\}$), and $f \equiv 20$. For $\Psi := \check{\Psi} = \hat{\Psi}$, we use biorthogonal B-spline wavelets as constructed in [9, Ch.2] with primal and dual orders $d = \tilde{d} = 3$, and homogeneous Dirichlet boundary conditions of order 1 at primal- and dual side.

In order to apply an adaptive wavelet scheme, we first have to reformulate (4.1) as an *equivalent* $\ell_2(\boldsymbol{\nabla})$-problem (cf. [5]). To this end, observe that the $|\cdot|_{H^1(\square)}$-normalized tensor basis $\mathbf{D}\boldsymbol{\Psi}$, where $\mathbf{D}$ is a bi-infinite diagonal matrix with entries $\mathbf{D}_{\boldsymbol{\lambda}} := \mathbf{D}_{\boldsymbol{\lambda}\boldsymbol{\lambda}} := |\boldsymbol{\psi}_{\boldsymbol{\lambda}}|^{-1}_{H^1(\square)}$, is a Riesz basis for $H^1_0(\square)$, i.e., $\|\mathbf{v}\|_{\ell_2(\boldsymbol{\nabla})} \eqsim \|v\|_{H^1(\square)}$ for all $v = \mathbf{v}^\top \mathbf{D}\boldsymbol{\Psi} \in H^1_0(\square)$. By expanding the solution u of (4.1) in $\mathbf{D}\boldsymbol{\Psi}$, i.e., $u = \mathbf{u}^\top \mathbf{D}\boldsymbol{\Psi}$, (4.1) can equivalently be stated as the discrete $\ell_2(\boldsymbol{\nabla})$ problem of finding $\mathbf{u} \in \ell_2(\boldsymbol{\nabla})$ such that

$$\mathbf{A}\mathbf{u} = \mathbf{f}. \tag{4.2}$$

Here, $\mathbf{A} : \ell_2(\boldsymbol{\nabla}) \to \ell_2(\boldsymbol{\nabla})$ given by $\mathbf{A} := [\mathbf{D}_{\boldsymbol{\lambda}}\, \bar{\boldsymbol{a}}(\boldsymbol{\psi}_{\boldsymbol{\lambda}}, \boldsymbol{\psi}_{\boldsymbol{\mu}})\mathbf{D}_{\boldsymbol{\mu}}]_{\boldsymbol{\lambda},\boldsymbol{\mu}\in\boldsymbol{\nabla}}$ is a symmetric, boundedly invertible operator and $\mathbf{f} := [\mathbf{D}_{\boldsymbol{\lambda}} f(\boldsymbol{\psi}_{\boldsymbol{\lambda}})]_{\boldsymbol{\lambda}\in\boldsymbol{\nabla}}$. Moreover, with

$$X := \left[\int_0^1 p\, \psi'_\lambda\, \psi'_\mu\right]_{\lambda,\mu\in\nabla}, \quad M := \left[\int_0^1 p\, \psi_\lambda\, \psi_\mu\right]_{\lambda,\mu\in\nabla}, \tag{4.3}$$

being univariate stiffness and mass matrices, we infer that

$$\mathbf{A} = \mathbf{D}(\boldsymbol{a}_1(\boldsymbol{\Psi},\boldsymbol{\Psi}) + \boldsymbol{a}_2(\boldsymbol{\Psi},\boldsymbol{\Psi}))\mathbf{D} = \mathbf{D}(X \otimes M + M \otimes X)\mathbf{D}. \tag{4.4}$$

Remark 4.1. We remark that constant coefficients, i.e., $\boldsymbol{p}_i \equiv c_i > 0$, allow for simplifications of the evaluation scheme from Theorem 3.1 that we want to exclude for demonstration purposes (see also Remark 1.1). For example, the usage of *prewavelets* (cf. [16, 11]) leads to M being a sparse diagonal block matrix with its strict lower block thus being zero. The same holds true for L_2-orthonormal multiwavelets (cf. [7, 8]) where M is even a diagonal matrix. The multiwavelet construction analyzed in [6] would even yield sparse M and X. In the latter case,

the exact multiplication of $\mathbf{A}$ with any finitely supported vector can be performed in linear complexity.

Remark 4.2. The scaling functions $\phi_\lambda \in \Phi$ associated to Ψ are B-splines. So, with the univariate coefficients p being polynomials, $\int_0^1 p\phi_\mu\phi_\lambda$ and $\int_0^1 p\phi'_\mu\phi'_\lambda$ can be evaluated *exactly* in $\mathcal{O}(1)$ operations.

4.2. **Adaptive tensor product wavelet Galerkin method (`AWGM`).** The idea of the `AWGM` for solving (4.2), as presented in e.g. [5, 12], can be outlined in short as follows: For each $\mathbf{\Lambda}$ from a sequence of finite, nested index sets $\mathbf{\Lambda}_0 \subset \mathbf{\Lambda}_1 \subset \cdots \subset \boldsymbol{\nabla}$, the finite dimensional Galerkin system

$$\mathbf{A}_{\mathbf{\Lambda}}\,\mathbf{u}_{\mathbf{\Lambda}} = \mathbf{f}_{\mathbf{\Lambda}}, \tag{4.5}$$

where $\mathbf{A}_{\mathbf{\Lambda}} := R_{\mathbf{\Lambda}}\mathbf{A}I_{\mathbf{\Lambda}}$, $\mathbf{f}_\Lambda := R_{\mathbf{\Lambda}}\mathbf{f}$, is solved within a fixed, sufficiently small relative tolerance, yielding an approximate solution $\mathbf{w}_{\mathbf{\Lambda}}$. In order to ensure convergence of the scheme, for some constant $\mu > 0$, the index sets $\mathbf{\Lambda}_k$ have to satisfy

$$\|R_{\mathbf{\Lambda}_{k+1}}\mathbf{r}_k\|_{\ell_2(\mathbf{\Lambda}_{k+1})} \geq \mu\|\mathbf{r}_k\|_{\ell_2(\boldsymbol{\nabla})}, \tag{4.6}$$

where $\mathbf{r}_k$ is a (finitely supported) approximation to the residual $\mathbf{f}-\mathbf{A}I_{\mathbf{\Lambda}_k}\mathbf{w}_{\mathbf{\Lambda}_k}$ within a fixed, sufficiently small relative tolerance.

If, additionally, μ is small enough (only depending on $\kappa(\mathbf{A})$), and the index set $\mathbf{\Lambda}_{k+1}$ is selected to have, up to some constant multiple, minimal cardinality among all index sets that satisfy (4.6), then the `AWGM` is proven to converge with the optimal rate: If, for whatever $s > 0$, $\mathbf{u}$ belongs to the *non-linear approximation class*,

$$\{\mathbf{v} \in \ell_2(\boldsymbol{\nabla}) : |\mathbf{v}|_{\mathcal{A}^s} := \sup_{N\in\mathbb{N}_0} N^s\|\mathbf{v}-\mathbf{v}_N\|_{\ell_2(\boldsymbol{\nabla})} < \infty\},$$

where $\mathbf{v}_N$ is a *best N-term approximation* to $\mathbf{v}$ (i.e., it minimizes the distance to $\mathbf{v}$ among all its approximations with support length N), then the computed approximate Galerkin solutions satisfy

$$\|\mathbf{u} - I_{\mathbf{\Lambda}_k}\mathbf{w}_{\mathbf{\Lambda}_k}\|_{\ell_2(\boldsymbol{\nabla})} \lesssim (\#\mathbf{\Lambda}_{k+1})^{-s}. \tag{4.7}$$

The above statements concerning convergence, in particular with the optimal rate, remain valid when inside the `AWGM` only $\mathbf{\Lambda}$ from a subset of $\mathcal{P}(\boldsymbol{\nabla})$ are allowed, assuming this subset is closed under taking unions, as e.g. the collection of all multi-trees. Of course, in this case also the definition of the approximation class should be adapted by considering only best N-term approximations supported on sets from the same subset of $\mathcal{P}(\boldsymbol{\nabla})$.

We will speak about *unconstrained* or *multi-tree approximation* when $\mathbf{\Lambda}$ is allowed to be any subset of $\boldsymbol{\nabla}$, or when it has to be a multi-tree, respectively.

Remark 4.3. For the problem (4.1) at hand, with f being some general, sufficiently smooth function, it follows from [10, Thm 4.3(b), Thm. 5.2, discussion in §5.4] that the best multi-tree approximation rate for $\mathbf{u}$ is $s = d - 1 = 2$. Clearly, this rate is the best that can be expected for approximation in $H^1(\square)$ by piecewise polynomials of order d, showing that there is no penalty because of the restriction to multi-tree approximation. In the same paper it was shown, in any case for the Poisson problem, that for general, sufficiently smooth f, the best possible rate using the non-adaptive, (optimized) sparse-grid approximation, i.e., approximation from the span of tensor product wavelets with indices in the multi-tree

$$\mathbf{\Lambda}_{\ell,\gamma} := \{\boldsymbol{\lambda} \in \boldsymbol{\nabla} : \||\boldsymbol{\lambda}|\|_1 - \gamma\||\boldsymbol{\lambda}|\|_\infty \leq (1-\gamma)\ell\}, \quad \ell \in \mathbb{N},$$

with $\gamma > 0$ sufficiently small, is $\frac{1}{2} + \frac{1}{n} = 1$. Only for f that additionally vanishes at all four corners of the domain, the latter rate would be 2. Not surprisingly, our numerical results will indicate that for also our problem (4.1) and $f \equiv 20$, the optimized sparse-grid rate is 1.

Optimized sparse grids, i.e., $\mathbf{\Lambda}_{\ell,\gamma}$ for $\gamma > 0$, were introduced in [13]. The rate of best approximation in $H^t(\square)$ for $0 < t < d$ from the span of tensor product wavelets with indices in $\mathbf{\Lambda}_{\ell,\gamma}$ for γ sufficiently small, and for a sufficiently smooth u is $d - t$, so without loss of any log-factors.

The computationally most expensive step in the AWGM is the computation of the finitely supported approximation $\mathbf{r}_k$ to $\mathbf{f} - \mathbf{A}I_{\mathbf{\Lambda}_k}\mathbf{w}_{\mathbf{\Lambda}_k}$ within a fixed, sufficiently small relative tolerance $\eta < 1$. It can be achieved by performing the following "inner" loop: Starting with $\varepsilon \eqsim \|\mathbf{f} - \mathbf{A}I_{\mathbf{\Lambda}_{k-1}}\mathbf{w}_{\mathbf{\Lambda}_{k-1}}\|_{\ell_2(\boldsymbol{\nabla})}$, approximate both $\mathbf{f}$ and $\mathbf{A}I_{\mathbf{\Lambda}_k}\mathbf{w}_{\mathbf{\Lambda}_k}$ within some absolute tolerance $\varepsilon/2$; check whether ε is less than or equal to $\frac{\eta}{1+\eta}$ times the norm of the resulting computed approximate residual $\mathbf{r}_k$, and if not repeat with ε replaced by $\varepsilon/2$.

If the cost of approximating $\mathbf{f} - \mathbf{A}I_{\mathbf{\Lambda}_k}\mathbf{w}_{\mathbf{\Lambda}_k}$ within an absolute tolerance ε, and so in particular the support length of $\mathbf{r}_k$, is

$$\mathcal{O}(\varepsilon^{-1/s} + \#\mathbf{\Lambda}_k), \tag{4.8}$$

then the AWGM is of *optimal computational complexity* in the sense that

$$\sup_k (\#\mathrm{ops}_{k+1})^s \|\mathbf{u} - I_{\mathbf{\Lambda}_k}\mathbf{w}_{\mathbf{\Lambda}_k}\|_{\ell_2(\boldsymbol{\nabla})} < \infty,$$

where $\#\mathrm{ops}_k$ is the number of arithmetic operations used to compute $\mathbf{w}_{\mathbf{\Lambda}_k}$. Indeed, by the choice of the initial value of ε and (4.7), the cost of any iteration in the above loop is $\mathcal{O}(\varepsilon^{-1/s})$ for the current value of ε. For the final ε, it holds that $\varepsilon \eqsim \|\mathbf{u} - I_{\mathbf{\Lambda}_k}\mathbf{w}_{\mathbf{\Lambda}_k}\|_{\ell_2(\boldsymbol{\nabla})}$, and so $\#\mathrm{ops}_{k+1} \lesssim \sum_{\nu=1}^k \|\mathbf{u} - I_{\mathbf{\Lambda}_\nu}\mathbf{w}_{\mathbf{\Lambda}_\nu}\|_{\ell_2(\boldsymbol{\nabla})}^{-1/s} \lesssim \|\mathbf{u} - I_{\mathbf{\Lambda}_k}\mathbf{w}_{\mathbf{\Lambda}_k}\|_{\ell_2(\boldsymbol{\nabla})}^{-1/s}$.

In the unconstrained approximation setting, in [5] an approximate matrix-vector multiplication routine APPLY was developed for approximating $\mathbf{A}I_{\mathbf{\Lambda}_k}\mathbf{w}_{\mathbf{\Lambda}_k}$. It consists of an approximation scheme for the columns of $\mathbf{A}$ with accuracies that are increasing as function of the modulus of the corresponding entry in the input vector. Assuming a sufficient near-sparsity of $\mathbf{A}$ and that of $\mathbf{w}_{\mathbf{\Lambda}_k}$, the latter in the sense that $\sup_k |I_{\mathbf{\Lambda}_k}\mathbf{w}_{\mathbf{\Lambda}_k}|_{\mathcal{A}^s} < \infty$, the cost of the APPLY-routine with tolerance $\varepsilon/2$ satisfies (4.8). The sufficient near-sparsity of $\mathbf{A}$ follows from the smoothness of the coefficients $\boldsymbol{p}_i$ of the differential operator, together with the smoothness and the vanishing moments of the wavelets, whereas $\sup_k |I_{\mathbf{\Lambda}_k}\mathbf{w}_{\mathbf{\Lambda}_k}|_{\mathcal{A}^s} < \infty$ is a consequence of $\mathbf{u} \in \mathcal{A}^s$ and the convergence of $(I_{\mathbf{\Lambda}_k}\mathbf{w}_{\mathbf{\Lambda}_k})_k$ towards $\mathbf{u}$ with rate s.

Usually, approximating $\mathbf{f}$ does not pose any problem. From the aforementioned near-sparsity of $\mathbf{A}$, it follows that if $\mathbf{u} \in \mathcal{A}^s$, then $\mathbf{f} = \mathbf{A}\mathbf{u} \in \mathcal{A}^s$.

The application of APPLY turns out to be quantitatively expensive, and therefore we searched for alternatives. In the accompanying paper [15], we develop an alternative scheme that applies with multi-tree approximation, and with piecewise polynomial univariate wavelets that are contained in $H^2(0,1)$. Instead of approximating $\mathbf{f}$ and $\mathbf{A}I_{\mathbf{\Lambda}_k}\mathbf{w}_{\mathbf{\Lambda}_k}$ separately, with $\mathcal{A}(v)(w) := \bar{\mathbf{a}}(v,w)$, the idea is to find a finite representation of both f and $\mathcal{A}((I_{\mathbf{\Lambda}_k}\mathbf{w}_{\mathbf{\Lambda}_k})^\top\mathbf{D}\mathbf{\Psi})$ in some *common* auxiliary tensor product basis, facilitating the approximation of $\mathbf{f} - \mathbf{A}I_{\mathbf{\Lambda}_k}\mathbf{w}_{\mathbf{\Lambda}_k} = \big[\big(f - \mathcal{A}((I_{\mathbf{\Lambda}_k}\mathbf{w}_{\mathbf{\Lambda}_k})^\top\mathbf{D}\mathbf{\Psi})\big)(\psi_{\boldsymbol{\lambda}})\big]_{\boldsymbol{\lambda}\in\boldsymbol{\nabla}}$.

For general $f \in H^{-1}(\square)$, the existence of such a finite representation can actually not be expected. But, as will be shown in [15], it can be approximated in $H^{-1}(\square)$ within tolerance ε from the span of $\mathcal{O}(\varepsilon^{-1/s})$ of these auxiliary basis functions. From this, it will follow that there exists a multi-tree $\tilde{\mathbf{\Lambda}}_k \supset \mathbf{\Lambda}_k$ with $\#\tilde{\mathbf{\Lambda}}_k \lesssim \#\mathbf{\Lambda}_k + \varepsilon^{-1/s}$, and

$$\|(I - I_{\tilde{\mathbf{\Lambda}}_k} R_{\tilde{\mathbf{\Lambda}}_k})(\mathbf{f} - \mathbf{A} I_{\mathbf{\Lambda}_k} \mathbf{w}_{\mathbf{\Lambda}_k})\|_{\ell_2(\boldsymbol{\nabla})} \leq \eta/2 \|I_{\tilde{\mathbf{\Lambda}}_k} R_{\tilde{\mathbf{\Lambda}}_k} (\mathbf{f} - \mathbf{A} I_{\mathbf{\Lambda}_k} \mathbf{w}_{\mathbf{\Lambda}_k})\|_{\ell_2(\boldsymbol{\nabla})} + \mathcal{O}(\varepsilon).$$

Theorem 3.1 shows that $R_{\tilde{\mathbf{\Lambda}}_k} \mathbf{A} I_{\mathbf{\Lambda}_k} \mathbf{w}_{\mathbf{\Lambda}_k}$ can be computed in $\mathcal{O}(\#\tilde{\mathbf{\Lambda}}_k + \#\mathbf{\Lambda}_k)$ operations. So under the assumption that $R_{\tilde{\mathbf{\Lambda}}_k} \mathbf{f}$ can be evaluated in $\mathcal{O}(\#\tilde{\mathbf{\Lambda}}_k)$ operations, we have that the cost of the evaluation of the approximate residual $\mathbf{r}_k := I_{\tilde{\mathbf{\Lambda}}_k} R_{\tilde{\mathbf{\Lambda}}_k} (\mathbf{f} - \mathbf{A} I_{\mathbf{\Lambda}_k} \mathbf{w}_{\mathbf{\Lambda}_k})$ satisfies (4.8). Upon replacing ε by ε/C for a suitable constant $C > 0$, its error is bounded by $\varepsilon/2 + \eta/2 \|\mathbf{r}_k\|_{\ell_2(\boldsymbol{\nabla})}$.

From the fact that the previous approximate residual $\tilde{\mathbf{r}}_k$, so with ε reading as 2ε, apparently satisfied $\varepsilon + \eta/2 \|\tilde{\mathbf{r}}_k\|_{\ell_2(\boldsymbol{\nabla})} > \frac{\eta}{1+\eta} \|\tilde{\mathbf{r}}_k\|_{\ell_2(\boldsymbol{\nabla})}$, an elementary analysis shows that $\varepsilon/2 + \eta/2 \|\mathbf{r}_k\|_{\ell_2(\boldsymbol{\nabla})} \eqsim \varepsilon$. We conclude that this approximate residual evaluation satisfies the condition for optimal computational complexity.

Other than with the `APPLY` routine, the approximate matrix-vector product $R_{\tilde{\mathbf{\Lambda}}_k} \mathbf{A} I_{\mathbf{\Lambda}_k} \mathbf{w}_{\mathbf{\Lambda}_k}$ depends linearly on $\mathbf{w}_{\mathbf{\Lambda}_k} \in \ell_2(\mathbf{\Lambda}_k)$, and we expect it to be quantitatively much more efficient.

In the simple case that f has a representation with finite, "small" support in the auxiliary tensor product basis, as with our forcing function $f \equiv 20$, the definition of the multi-tree $\tilde{\mathbf{\Lambda}}_k$ reads as

$$\tilde{\mathbf{\Lambda}}_k := \big\{ \boldsymbol{\lambda} \in \boldsymbol{\nabla} : \exists \boldsymbol{\mu} \in \mathbf{\Lambda}_k \text{ s.t. } \forall i,\ |\lambda_i| \leq |\mu_i| + \ell \wedge \operatorname{dist}(\operatorname{supp} \psi_{\mu_i}, \operatorname{supp} \psi_{\lambda_i}) \leq C 2^{-|\lambda_i|} \big\}, \tag{4.9}$$

with the constant C from Lemma 2.1.

4.3. **Numerical results.** We use the `AWGM` to solve (4.1) numerically, where we focus on the computational cost of the application of $R_{\mathbf{\Lambda}} \mathbf{A} I_{\mathbf{\Lambda}}$, for the Galerkin solves, and that of $R_{\tilde{\mathbf{\Lambda}}} \mathbf{A} I_{\mathbf{\Lambda}}$, and on the structure of the adaptively computed multi-trees $\mathbf{\Lambda}$. For the implementation we used the C++ library `LAWA` ([19]).

Within our numerical experiments, it has turned out that $\ell = 1$ in the definition of $\tilde{\mathbf{\Lambda}} = \tilde{\mathbf{\Lambda}}_k$ from (4.9) for the approximation of the residual is sufficient. We set the bulk chasing parameter μ from (4.6) to $\mu = 0.4$ and solve arising Galerkin system (4.5) in each iteration approximately within the relative tolerance $\|I_{\mathbf{\Lambda}_k}(\boldsymbol{A}_{\mathbf{\Lambda}_k} \mathbf{w}_{\mathbf{\Lambda}_k} - \mathbf{f}_{\mathbf{\Lambda}_k})\|_{\ell_2(\boldsymbol{\nabla})} \leq \omega \|\mathbf{r}_{k-1}\|_{\ell_2(\boldsymbol{\nabla})}$ with $\omega = 0.2$.

In Figure 1 (a), we show the computed numerical solution for (4.1) obtained by the adaptive scheme. For comparison, we also solved (4.1) with an optimized sparse grid with $\gamma = 0.2$. The convergence of both the optimized sparse grid method and the `AWGM` can be found in Figure 1 where we show the $\ell_2(\boldsymbol{\nabla})$-norm of the residual

$$\|\mathbf{r}_k\|_{\ell_2(\boldsymbol{\nabla})} \eqsim \|\mathbf{f} - \mathbf{A} I_{\mathbf{\Lambda}_k} \mathbf{w}_{\mathbf{\Lambda}_k}\|_{\ell_2(\boldsymbol{\nabla})} \eqsim \|u - (I_{\mathbf{\Lambda}_k} \mathbf{w}_{\mathbf{\Lambda}_k})^\top \mathbf{D} \mathbf{\Psi}\|_{H^1(\square)}.$$

Observe that the `AWGM` realizes the optimal rate $d - 1 = 2$, whereas the non-adaptive optimized sparse grid method converges at the reduced rate $\frac{1}{2} + \frac{1}{n} = 1$.

In Figure 2, we show the computation times for the adaptively created multi-trees that occurred during the adaptive solution of (4.1). We consider exemplarily $X \otimes M$. Observe that the computation times for the application of $R_{\mathbf{\Lambda}}(X \otimes M) I_{\mathbf{\Lambda}}$ as well as for the application of $R_{\tilde{\mathbf{\Lambda}}}(X \otimes M) I_{\mathbf{\Lambda}}$ scale linearly with the corresponding

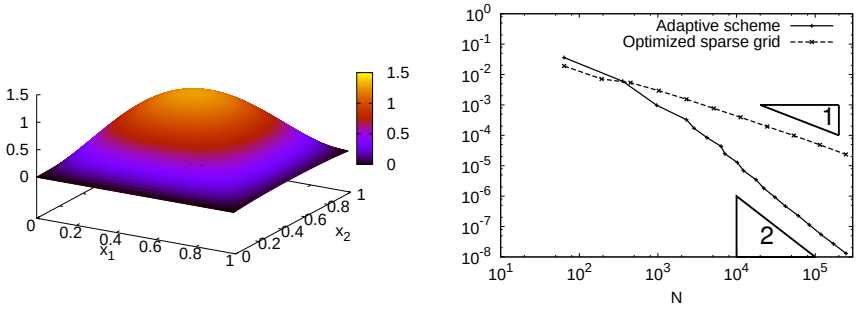
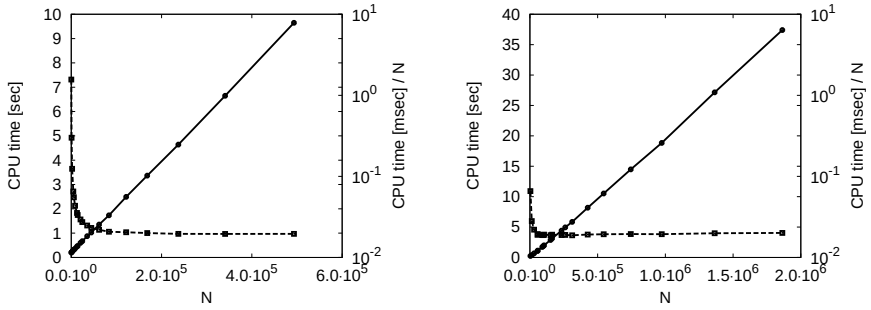


FIGURE 1. Numerical solution for (4.1) (left), and convergence of $\|\mathbf{r}_k\|_{\ell_2(\boldsymbol{\nabla})}$ for the adaptive scheme as well as for the optimized sparse grid method (right).

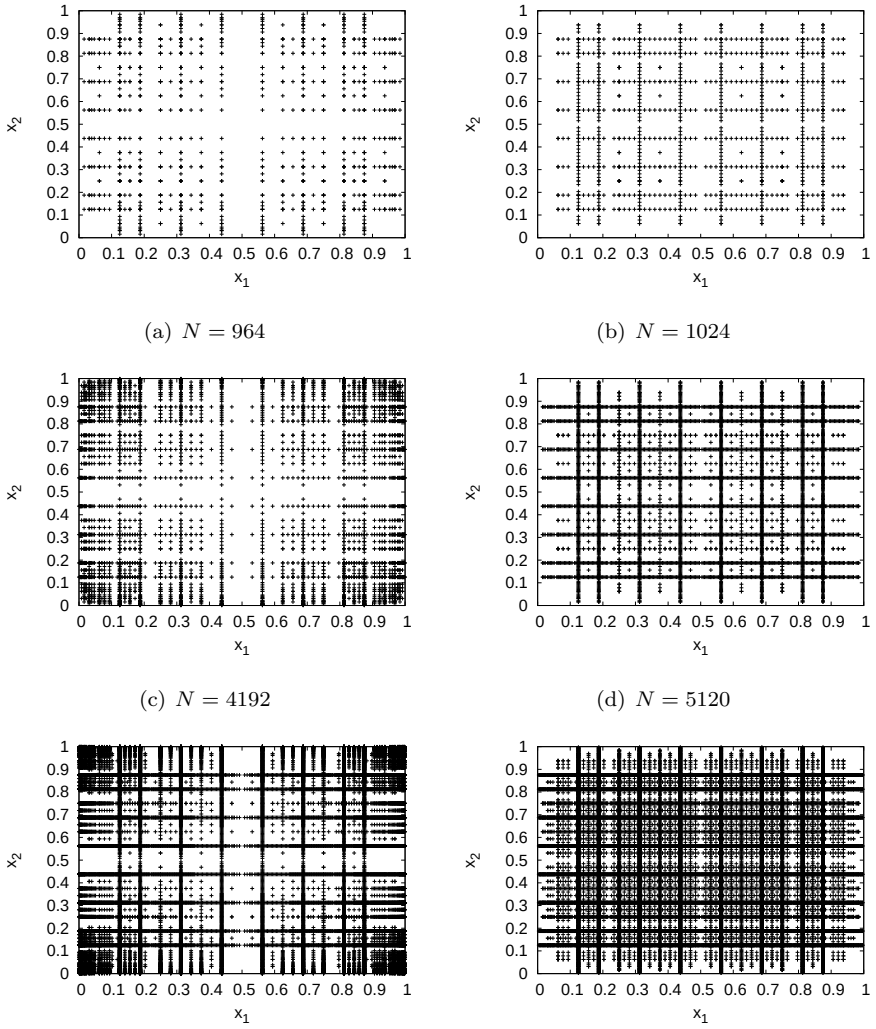
number of degrees of freedom. We also show the ratio "time [in milliseconds] per degree of freedom" (in a log-scale) which is asymptotically constant.



(a) $R_{\boldsymbol{\Lambda}}(X \otimes M)I_{\boldsymbol{\Lambda}}$, $N := 2 \cdot \#\boldsymbol{\Lambda}$ (b) $R_{\tilde{\boldsymbol{\Lambda}}}(X \otimes M)I_{\boldsymbol{\Lambda}}$, $N := \#\tilde{\boldsymbol{\Lambda}} + \#\boldsymbol{\Lambda}$

FIGURE 2. Computation times in seconds (solid line) and computation time in milliseconds per degree of freedom (dashed line) for the indicated matrices.

Finally, we show some of the computed multi-trees computed by the AWGM, and compare them to the ones from the optimized sparse grid scheme in Figure 3. We observe that the adaptive scheme automatically resolves the singularities at the corners, where the optimized sparse grid scheme due to its construction spends too few degrees of freedom. In Figure 4, we show instances of the one-dimensional trees for both methods. We fix the (one-dimensional) index μ that corresponds to a scaling function index at the left boundary and consider the (one-dimensional) tree $\boldsymbol{\Lambda}_{\neg \mathbf{e}_1,\mu}$. Here, each rectangle represents one wavelet index $\lambda \in \boldsymbol{\Lambda}_{\neg \mathbf{e}_1,\mu}$ and indicates the level of λ as well as the position of the support of ψ_λ *relative* to the supports of the other wavelets on the same level. We observe the adaptive refinement in the corners where the solution u is singular. In contrast, only few wavelets are required in the interior of the domain where the solution is smooth.



(a) $N = 964$ (b) $N = 1024$

(c) $N = 4192$ (d) $N = 5120$

(e) $N = 22195$ (f) $N = 24576$

FIGURE 3. Visualization of the multi-trees arising from the adaptive scheme (left) and from the optimized sparse grid scheme with $\gamma = 0.2$ (right) for the indicated numbers of degrees of freedom. A cross (+) refers to the barycenter of the support of $\psi_{\lambda_1} \otimes \psi_{\lambda_2}$.

REFERENCES

[1] S. Achatz. *Adaptive finite Dünngitter-Elementen höherer Ordnung für elliptische partiele Differentialgleigungen mit variablen Koefficienten*. PhD thesis (in German), Technische Universität München, 2003.

[2] H.-J. Bungartz and M. Griebel. Sparse grids. *Acta Numer.*, 13:147–269, 2004.

[3] H.-J. Bungartz. A multigrid algorithm for higher order finite elements on sparse grids. *Electron. Trans. Numer. Anal.*, 6(Dec.):63–77 (electronic), 1997. Special issue on multilevel methods (Copper Mountain, CO, 1997).

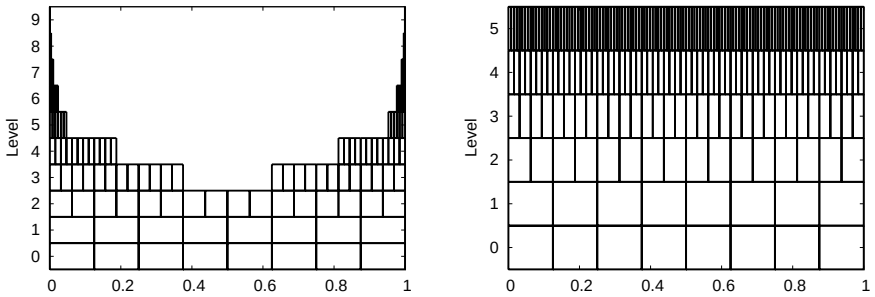


FIGURE 4. One-dimensional trees $\mathbf{\Lambda}_{\neg \mathbf{e}_1,\mu}$ from the adaptive scheme with $\#\mathbf{\Lambda} = 4192$ (left) and from the optimized sparse grid for $\ell = 5$ and $\gamma = 0.2$, $\#\mathbf{\Lambda}_{\ell,\gamma} = 5120$, (right) for μ corresponding to a scaling function index at the left boundary.

[4] R. Balder and Ch. Zenger. The solution of multidimensional real Helmholtz equations on sparse grids. *SIAM J. Sci. Comput.*, 17(3):631–646, 1996.

[5] A. Cohen, W. Dahmen, and R. DeVore. Adaptive wavelet methods for elliptic operator equations – Convergence rates. *Math. Comp*, 70:27–75, 2001.

[6] N.G. Chegini and R.P. Stevenson. The adaptive tensor product wavelet scheme: Sparse matrices and the application to singularly perturbed problems. *IMA J. Numer. Anal.*, 32(1):75–104, 2011.

[7] G.C. Donovan, J.S. Geronimo, and D.P. Hardin. Intertwining multiresolution analyses and the construction of piecewise-polynomial wavelets. *SIAM J. Math. Anal.*, 27(6):1791–1815, 1996.

[8] T.J. Dijkema, Ch. Schwab, and R.P. Stevenson. An adaptive wavelet method for solving high-dimensional elliptic PDEs. *Constr. Approx.*, 30(3):423–455, 2009.

[9] T.J. Dijkema. *Adaptive tensor product wavelet methods for solving PDEs*. PhD thesis, Utrecht University, 2009.

[10] M. Dauge and R.P. Stevenson. Sparse tensor product wavelet approximation of singular functions. *SIAM J. Math. Anal.*, 42(5):2203–2228, 2010.

[11] C. Feuersänger. Dünngitterverfahren für hochdimensionale elliptischer partielle Differentialgleigungen. Master's thesis (in German), Institut für Numerische Simulation, Universität Bonn, 2005.

[12] T. Gantumur, H. Harbrecht, and R.P. Stevenson. An optimal adaptive wavelet method without coarsening of the iterands. *Math. Comp.*, 76:615–629, 2007.

[13] M. Griebel and S. Knapek. Optimized tensor-product approximation spaces. *Constr. Approx.*, 16(4):525–540, 2000.

[14] H Harbrecht, R. Schneider, and Ch. Schwab. Multilevel frames for sparse tensor product spaces. *Numer. Math.*, 110(2):199–220, 2008.

[15] S. Kestler and R.P. Stevenson. An efficient approximate residual evaluation in the adaptive tensor product wavelet method. Technical report, August 2012. Submitted.

[16] A. Niedermeier and S. Zimmer. Implementational aspects of prewavelet sparse grid methods. In *Eleventh International Conference on Domain Decomposition Methods (London, 1998)*, pages 314–321. 1999.

[17] D. Pflüger. *Spatially Adaptive Sparse Grids for High-Dimensional Problems*. PhD thesis, TUM, München, 2010.

[18] Ch. Schwab and R.P. Stevenson. Adaptive wavelet algorithms for elliptic PDEs on product domains. *Math. Comp.*, 77:71–92, 2008.

[19] A. Stippler. LAWA - Library for Adaptive Wavelet Applications. *http://lawa.sourceforge.net*, 2009. (last accessed November 8, 2012).

[20] J. Shen and H. Yu. Efficient spectral sparse grid methods and applications to high-dimensional elliptic problems. *SIAM J. Sci. Comput.*, 32(6):3228–3250, 2010.
[21] A. Zeiser. Fast matrix-vector multiplication in the sparse-grid Galerkin method. *J. Sci. Comput.*, 47(3):328–346, 2011.

Sebastian Kestler, Institute for Numerical Mathematics, University of Ulm, Helmholtzstrasse 20, D-89069 Ulm, Germany
E-mail address: sebastian.kestler@uni-ulm.de

Rob Stevenson, Korteweg-de Vries Institute for Mathematics, University of Amsterdam, P.O. Box 94248, 1090 GE Amsterdam, The Netherlands
E-mail address: r.p.stevenson@uva.nl