Mathematics

http://www.aimspress.com/journal/Math

AIMS Mathematics, 8(11): 25633–25653.
DOI: 10.3934/math.20231308
Received: 19 June 2023
Revised: 16 July 2023
Accepted: 19 July 2023
Published: 05 September 2023

Research article

A pre-processing procedure for the implementation of the greedy rank-one algorithm to solve high-dimensional linear systems

J. Alberto Conejero[1], **Antonio Falcó** [2,*]**and María Mora–Jiménez**[1]

[1] Instituto Universitario de Matemática Pura y Aplicada. Universitat Politècnica de València, Spain
[2] ESI International Chair@CEU-UCH, Departamento de Matemáticas, Física y Ciencias Tecnológicas, Universidad CEU Cardenal Herrera, CEU Universities, San Bartolomé 55, 46115 Alfara del Patriarca, Spain

* **Correspondence:** Email: afalco@uchceu.es.

Abstract: Algorithms that use tensor decompositions are widely used due to how well they perfor with large amounts of data. Among them, we find the algorithms that search for the solution of a linear system in separated form, where the greedy rank-one update method stands out, to be the starting point of the famous proper generalized decomposition family. When the matrices of these systems have a particular structure, called a Laplacian-like matrix which is related to the aspect of the Laplacian operator, the convergence of the previous method is faster and more accurate. The main goal of this paper is to provide a procedure that explicitly gives, for a given square matrix, its best approximation to the set of Laplacian-like matrices. Clearly, if the residue of this approximation is zero, we will be able to solve, by using the greedy rank-one update algorithm, the associated linear system at a lower computational cost. As a particular example, we prove that the discretization of a general partial differential equation of the second order without mixed derivatives can be written as a linear system with a Laplacian-type matrix. Finally, some numerical examples based on partial differential equations are given.

Keywords: tensor decompositions; rank-one tensors; high-dimensional linear systems; laplacian-like matrices; partial differential equations
Mathematics Subject Classification: 65F99, 65N22

1. Introduction

Working with large amounts of data is one of the main challenges we face today. With the rise of social networks and rapid technological advances, we must develop tools that allow us to work with so much information. At this point the use of tensor products comes into play, since their use reduces the number of and speed up the operations to be carried out. Proof of this is the recent article [1], where tensor products are used to speed up the calculation of matrix products. Other articles that exemplify the goodness of this operation include [2], where the solution of 2,3-dimensional optimal control problems with spectral fractional Laplacian-type operators is studied, and [3], where high-order problems are studied through the use of proper generalized decomposition methods.

When we try to solve a linear system of the form $A\mathbf{x} = \mathbf{b}$, in addition to the classical methods, there are methods based on tensors that can be more efficient [4], since the classical methods face the problem of the curse of dimensionality, which makes them lose effectiveness as the size of the problem increases. The tensor methods look for the solution in separated form, that is, as the tensor combination

$$\mathbf{x} = \sum_{j=1}^{\infty} \mathbf{x}_1^j \otimes \cdots \otimes \mathbf{x}_d^j,$$

where $\mathbf{x}_i^j \in \mathbb{R}^{N_i}$, d is the dimension of the problem, and $\otimes$ is the Kronecker product as reviewed in the next Section. The main family of methods that solves this problem is proper generalized decomposition family [5], and it is based on the greedy rank-one update (GROU) algorithm [6, 7]. This algorithm calculates the solution of the linear system $A\mathbf{x} = \mathbf{b}$ in separated form and, for this, in each iteration, it updates the approximation of the solution with the term resulting from minimizing the remaining residue. Furthermore, there are certain square matrices for which the GROU algorithm improves their convergence i.e., matrices of the form

$$A = \sum_{i=1}^{d} \mathrm{id}_{N_1} \otimes \cdots \otimes \mathrm{id}_{N_{i-1}} \otimes A_i \otimes \mathrm{id}_{N_{i+1}} \otimes \cdots \otimes \mathrm{id}_{N_d},$$

where id_{N_k} is the identity matrix of size $N_k \times N_k$, and $A_k \in \mathbb{R}^{N_k \times N_k}$, for $1 \leq k \leq d$. These matrices are called Laplacian-like matrices, due to their relationship with the Laplace operator written as

$$\sum_{i=1}^{d} -\frac{\partial^2}{\partial x_i^2} = \sum_{i=1}^{d} \frac{\partial^0}{\partial x_1^0} \otimes \cdots \otimes \frac{\partial^0}{\partial x_{i-1}^0} \otimes \left(-\frac{\partial^2}{\partial x_i^2}\right) \otimes \frac{\partial^0}{\partial x_{i+1}^0} \otimes \cdots \otimes \frac{\partial^0}{\partial x_d^0}.$$

It is not easy to decide when a given matrix A can be represented in that form. To do this, we can use some of the previously results obtained by the authors of [8]. In this paper, we prove that the set of Laplacian-like matrices is a linear subspace for the space of square matrices with a particular decomposition of its dimension. Moreover, we provide a greedy algorithm that provides the best Laplacian approximation L_A, for a given matrix A, as well its residue, $R_A = A - L_A$. However, an iterative algorithm it is not useful enough against a direct solution algorithm. The main goal of this paper is to provide a direct algorithm that allows one to construct the best Laplacian-like approximation by using only a particular block decomposition of the matrix A. It can be considered as a pre-processing procedure that allows one to represent a given matrix in its best Laplacian-like form, and if the residual is equal to zero, we definitively have its Laplacian-like representation form. Hence, we efficiently use the GROU algorithm to solve the high-dimensional linear system associated with the matrix A.

We remark that, by using the decomposition $A = L_A + R_A$, we can rewrite the linear system as $(L_A + R_A)\mathbf{x} = \mathbf{b}$, and when the value of the remainder is small, we can approximate the solution of the system $\mathbf{x}^*$ by using the solution of the Laplacian system $\mathbf{x}_L$. This fact is specially interesting in the case of the discretization of some partial differential equations. We also study the Laplacian decomposition of the matrix that comes from the discretization of a general second order partial differential equation of the form

$$\alpha \frac{\partial^2 \mathbf{u}}{\partial x^2} + \beta \frac{\partial^2 \mathbf{u}}{\partial y^2} + \gamma \frac{\partial \mathbf{u}}{\partial x} + \delta \frac{\partial \mathbf{u}}{\partial y} + \mu \, \mathbf{u} = \mathbf{f},$$

with homogeneous boundary conditions. Besides, to compare different numerical methods to solve partial differential equations, we consider two particular cases: the Helmholtz equation, which solves an eigenvalue problem for the Laplace operator. Furthermore, to illustrate that it is not necessary to be limited to the second order, we also consider the 4th order Swift-Hohenberg equation

$$\frac{\partial \mathbf{u}}{\partial t} = \varepsilon - \left(1 + \frac{\partial^2}{\partial x^2}\right)^2 \mathbf{u}.$$

This equation is noted for its pattern-forming behavior, and it was derived from the equations for thermal convection [9].

The paper is organized as follows. We begin by recalling some preliminary definition and results used throughout the paper in Section 2. Section 3 is devoted to the statement and the proof of the main result of this paper, which allow one to construct explicitly the best approximation of a given matrix to the linear space of Laplacian-like matrices. After that, in Section 4, we discuss how we applied this result to compute the best Laplacian approximation for the discretization of a second order partial differential equations without mixing derivatives. Finally, some numerical examples are given in Section 5.

2. Preliminary definitions and results

First at all we introduce some notations that we use throughout the paper. We denote by $\mathbb{R}^{N\times M}$, the set of $N \times M$-matrices and by A^T the transpose of a given matrix A. As usual we use

$$\langle \mathbf{x}, \mathbf{y} \rangle_2 = \langle \mathbf{x}, \mathbf{y} \rangle_{\mathbb{R}^N} = \mathbf{x}^T \, \mathbf{y} = \mathbf{y}^T \, \mathbf{x}$$

to denote the Euclidean inner product in $\mathbb{R}^N$, and its corresponding 2-norm, by $\|\mathbf{x}\|_2 = \|\mathbf{x}\|_{\mathbb{R}^N} = \langle \mathbf{x}, \mathbf{x} \rangle_2^{1/2}$.

Given a sequence $\{\mathbf{u}_j\}_{j=0}^{\infty} \subset \mathbb{R}^N$, we say that a vector $\mathbf{u} \in \mathbb{R}^N$ can be written as

$$\mathbf{u} = \sum_{j=0}^{\infty} \mathbf{u}_j$$

if and only if

$$\lim_{n\to\infty} \sum_{j=0}^{n} \mathbf{u}_j = \mathbf{u}$$

in the $\| \cdot \|_2$-topology.

The *Kronecker product* of two matrices $A \in \mathbb{R}^{N_1 \times M_1}$ and $B \in \mathbb{R}^{N_2 \times M_2}$ is defined by

$$A \otimes B = \begin{pmatrix} A_{1,1}B & A_{1,2}B & \dots & A_{1,M_1}B \\ A_{2,1}B & A_{2,2}B & \dots & A_{2,M_1}B \\ \vdots & \vdots & \ddots & \vdots \\ A_{N_1,1}B & A_{N_1,2}B & \dots & A_{N_1,M_1}B \end{pmatrix} \in \mathbb{R}^{N_1 N_2 \times M_1 M_2}.$$

We can see some of the well-known properties of the Kronecker product in [7].

As we already said, we are interested solving the high-dimensional linear system $A\mathbf{x} = \mathbf{b}$ obtained from a discretization of a partial differential equation. We are interested in solving it by using a tensor-based algorithm; so, we are going to look for an approximation of the solution in separated form. To see this, we assume that the coefficient matrix A is a $(N_1 \cdots N_d) \times (N_1 \cdots N_d)$-dimensional invertible matrix for some $N_1, \cdots, N_d \in \mathbb{N}$. Next, we look for an approximation (of rank n) of $A^{-1}\mathbf{b}$ of the form

$$A^{-1}\mathbf{b} \approx \sum_{j=1}^{n} \mathbf{x}_1^j \otimes \cdots \otimes \mathbf{x}_d^j. \tag{2.1}$$

To do this, given $\mathbf{x} \in \mathbb{R}^{N_1 \cdots N_d}$, we say that $\mathbf{x} \in \mathcal{R}_1 = \mathcal{R}_1(N_1, N_2, \ldots, N_d)$ if $\mathbf{x} = \mathbf{x}_1 \otimes \mathbf{x}_2 \otimes \cdots \otimes \mathbf{x}_d$, where $\mathbf{x}_i \in \mathbb{R}^{N_i}$ for $i = 1, \ldots, d$. For $n \geq 2$, we define, inductively, that $\mathcal{R}_n = \mathcal{R}_n(N_1, N_2, \ldots, N_d) = \mathcal{R}_{n-1} + \mathcal{R}_1$, that is,

$$\mathcal{R}_n = \left\{ \mathbf{x} : \mathbf{x} = \sum_{i=1}^{k} \mathbf{x}^{(i)},\ \mathbf{x}^{(i)} \in \mathcal{R}_1 \text{ for } 1 \leq i \leq k \leq n \right\}.$$

Note that $\mathcal{R}_n \subset \mathcal{R}_{n+1}$ for all $n \geq 1$.

To perform (2.1), what we will do is minimize the difference

$$\left\| \mathbf{b} - A \left(\sum_{j=1}^{n} \mathbf{x}_d^j \otimes \cdots \otimes \mathbf{x}_d^j \right) \right\|_2,$$

that is, solve the problem

$$\arg\min_{\mathbf{u} \in \mathcal{R}_n} \|\mathbf{b} - A\mathbf{u}\|_2. \tag{2.2}$$

Here, $\| \cdot \|_2$ is the 2-norm, or the Frobenius norm, defined by

$$\|A\|_2 = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{i,j}|^2} = \sqrt{\operatorname{tr}(A^\top A)}, \quad \text{for} \quad A \in \mathbb{R}^{m \times n}.$$

Unfortunately, from Proposition 4.1(a) of [10], we have that the set $\mathcal{R}_n$ is not necessarily (or even usually) closed for each $n \geq 2$. In consequence, no best rank-n approximation exists, that is, (2.2) has no solution. However, from Proposition 4.2 of [10] it follows that $\mathcal{R}_1$ is a closed set in any norm-topology. This fact allows us to introduce the following algorithm.

The GROU algorithm is an iterative method to solve linear systems of the form $A\mathbf{x} = \mathbf{b}$ by using only rank-one updates. Thus, given $A \in \mathrm{GL}(\mathbb{R}^{N \times N})$ with $N = N_1 \cdots N_d$, and $\mathbf{b} \in \mathbb{R}^N$, we can obtain an approximation of the form

$$A^{-1}\mathbf{b} \approx \mathbf{u}_n = \sum_{j=1}^{n} \mathbf{x}_1^j \otimes \cdots \otimes \mathbf{x}_d^j$$

for some $n \geq 1$, and $\mathbf{x}_i^j \in \mathbb{R}^{N_i}$ for $i = 1, 2, \ldots, d$ and $j = 1, 2, \ldots, n$ [7]. We proceed with the following iterative procedure (see algorithm 1 below): let $\mathbf{u}_0 = \mathbf{y}_0 = 0$, and, for each $n \geq 1$, take

$$\mathbf{r}_{n-1} = \mathbf{b} - A\mathbf{u}_{n-1}, \tag{2.3}$$

$$\mathbf{u}_n = \mathbf{u}_{n-1} + \mathbf{y}_n, \quad \text{where} \quad \mathbf{y}_n \in \arg\min_{\mathbf{u}\in\mathcal{R}_1} \|\mathbf{r}_{n-1} - A\mathbf{u}\|_2. \tag{2.4}$$

Since $\mathbf{u}_n \approx A^{-1}\mathbf{b}$, we can define the $\operatorname{rank}_\otimes$ for $A^{-1}\mathbf{b}$ obtained by the GROU Algorithm as

$$\operatorname{rank}_\otimes(A^{-1}\mathbf{b}) = \begin{cases} \infty & \text{if } \{j \geq 1 : \mathbf{y}_j = 0\} = \emptyset, \\ \min\{j \geq 1 : \mathbf{y}_j = 0\} - 1 & \text{otherwise.} \end{cases}$$

The next result, presented in [7], gives the convergence of the sequence $\{\mathbf{u}_n\}_{n\geq 0}$ to the solution $A^{-1}\mathbf{b}$ of the linear system.

Theorem 2.1. *Let* $\mathbf{b} \in \mathbb{R}^{N_1\cdots N_d}$ *and* $A \in \mathbb{R}^{N_1\cdots N_d \times N_1\cdots N_d}$ *be an invertible matrix. Then, by using the iterative scheme described by (2.3) and (2.4), we obtain that the sequence* $\{\|\mathbf{r}_n\|_2\}_{n=0}^{\operatorname{rank}_\otimes(A^{-1}\mathbf{b})}$ *is strictly decreasing and*

$$A^{-1}\mathbf{b} = \lim_{n\to\infty} \mathbf{u}_n = \sum_{j=0}^{\operatorname{rank}_\otimes(A^{-1}\mathbf{b})} \mathbf{y}_j. \tag{2.5}$$

Note that the updates in the previous scheme works under the assumption that, in line 5 of algorithm 1, we have a way to obtain

$$\mathbf{y} \in \arg\min_{\mathbf{x}\in\mathcal{R}_1} \|\mathbf{r}_i - A\mathbf{x}\|_2^2. \tag{2.6}$$

To compute $\mathbf{y}$, we can use an alternating least squares (ALS) approach (see [7, 11]).

Algorithm 1 GROU algorithm

```
 1: procedure GROU($\mathbf{f}, A, \varepsilon$, tol, rank_max)
 2:     $\mathbf{r}_0 = \mathbf{f}$
 3:     $\mathbf{u} = \mathbf{0}$
 4:     for $i = 0, 1, 2, \ldots,$ rank_max do
 5:         $\mathbf{y} =$ procedure $(\min_{\mathbf{x}\in\mathcal{R}_1} \|\mathbf{r}_i - A\mathbf{x}\|_2^2)$
 6:         $\mathbf{r}_{i+1} = \mathbf{r}_i - A\mathbf{y}$
 7:         $\mathbf{u} \leftarrow \mathbf{u} + \mathbf{y}$
 8:         if $\|\mathbf{r}_{i+1}\|_2 < \varepsilon$ or $|\|\mathbf{r}_{i+1}\|_2 - \|\mathbf{r}_i\|_2| <$ tol then goto 13
 9:         end if
10:     end for
11:     return $\mathbf{u}$ and $\|\mathbf{r}_{\texttt{rank\_max}}\|_2$.
12:     break
13:     return $\mathbf{u}$ and $\|\mathbf{r}_{i+1}\|_2$
14: end procedure
```

The idea below the ALS strategy to solve (2.6) is as follows: for each $1 \leq k \leq d$, we proceed as follows. Assume that the values $\mathbf{x}_1, \ldots, \mathbf{x}_{k-1}, \mathbf{x}_{k+1}, \ldots, \mathbf{x}_d$ are given. Then, we look for the unknown $\mathbf{x}_k$, satisfying

$$\mathbf{x}_k \in \arg\min_{\mathbf{z}_k\in\mathbb{R}^{N_k\times N_k}} \|\mathbf{b} - A(\mathbf{x}_1 \otimes \cdots \otimes \mathbf{x}_{k-1} \otimes \mathbf{z}_k \otimes \mathbf{x}_{k+1} \otimes \cdots \otimes \mathbf{x}_d)\|_2,$$

where we can write

$$A(\mathbf{x}_1 \otimes \cdots \otimes \mathbf{x}_{k-1} \otimes \mathbf{z}_k \otimes \mathbf{x}_{k+1} \otimes \cdots \otimes \mathbf{x}_d) = A(\mathbf{x}_1 \otimes \cdots \otimes \mathbf{x}_{k-1} \otimes \mathrm{id}_{N_k} \otimes \mathbf{x}_{k+1} \otimes \cdots \otimes \mathbf{x}_d)\mathbf{z}_k.$$

In consequence, by using a least squares approach [11], we can obtain $\mathbf{x}_k$ by solving the following $N_k \times N_k$-dimensional linear system:

$$Z_k \mathbf{z}_k = \mathbf{b}_, \tag{2.7}$$

where

$$Z_k := (\mathbf{x}_1^T \otimes \cdots \otimes \mathbf{x}_{k-1}^T \otimes \mathrm{id}_{N_k} \otimes \mathbf{x}_{k+1}^T \otimes \cdots \otimes \mathbf{x}_d^T) A^T A(\mathbf{x}_1 \otimes \cdots \otimes \mathbf{x}_{k-1} \otimes \mathrm{id}_k \otimes \mathbf{x}_{k+1} \otimes \cdots \otimes \mathbf{x}_d)$$

and

$$\mathbf{b}_k := (\mathbf{x}_1^T \otimes \cdots \otimes \mathbf{x}_{k-1}^T \otimes \mathrm{id}_{N_k} \otimes \mathbf{x}_{k+1}^T \otimes \cdots \otimes \mathbf{x}_d^T) A^T \mathbf{b}.$$

Clearly,

$$\|\mathbf{b} - A(\mathbf{x}_1 \otimes \cdots \otimes \mathbf{x}_{k-1} \otimes \mathbf{z}_k \otimes \mathbf{x}_{k+1} \otimes \cdots \otimes \mathbf{x}_d)\|_2 \leq \|\mathbf{b} - A(\mathbf{x}_1 \otimes \cdots \otimes \mathbf{x}_{k-1} \otimes \mathbf{x}_k \otimes \mathbf{x}_{k+1} \otimes \cdots \otimes \mathbf{x}_d)\|_2$$

holds for all $\mathbf{z}_k \in \mathbb{R}^{N_k \times N_k}$. However, it is well known (see Section 4 in [11]) that the performance of the ALS strategy can be improved (see Algorithm 2 below) when the shape of the matrix $A^T A \in \mathbb{R}^{N \times N}$, with $N = N_1 \dots N_d$, can be written in the form

$$A^T A = \sum_{i=1}^{r} \bigotimes_{j=1}^{d} A_j^{(i)}, \tag{2.8}$$

where $\bigotimes_{j=1}^{d} A_j^{(i)} = A_1^{(i)} \otimes \cdots \otimes A_d^{(i)}$; here, $A_j^{(i)} \in \mathbb{R}^{N_j \times N_j}$ for $1 \leq j \leq d$ and $1 \leq i \leq r$. In particular, when the matrix A is given by

$$A = \sum_{i=1}^{d} \mathrm{id}_{N_1} \otimes \cdots \otimes \mathrm{id}_{N_{i-1}} \otimes A_i \otimes \mathrm{id}_{N_{i+1}} \otimes \cdots \otimes \mathrm{id}_{N_d},$$

then the matrix $A^T A$ can be easily written in the form of (2.8). These matrices were introduced in [8] as Laplacian-like matrices since they can be easily related to the classical Laplacian operator [2, 12]. The next section will be devoted to the study of this class of matrices.

3. On the best Laplacian matrix approximation

As we said in the introduction, the proper orthogonal decomposition is a popular numerical strategy in the engineering process to solve high-dimensional problems. It is based on the GROU algorithms (2.3) and (2.4), and it can be considered as a tensor-based decomposition algorithm.

There is a particular type of matrices to solve high-dimensional linear systems for which these methods work particularly well, i.e., those that satisfy the property (2.8). To this end, we introduce the following definition.

Algorithm 2 An ALS algorithm for matrices in the form of (2.8) [11, Algorithm 2]

```
1: Given $A^TA = \sum_{i=1}^r \bigotimes_{j=1}^d A_j^{(i)} \in \mathbb{R}^{N\times N}$ and $\mathbf{b} \in \mathbb{R}^N$.
2: Initialize $\mathbf{x}_i^{(0)} \in \mathbb{R}^{N_i}$ for $i = 1, 2 \ldots, d$.
3: Introduce $\varepsilon > 0$ and itermax, iter = 1.
4: while distance > $\varepsilon$ and iter < itermax do
5:     for $k = 1, 2, \ldots, d$ do
6:         $\mathbf{x}_k^{(1)} = \mathbf{x}_k^{(0)}$
7:         for $i = 1, 2, \ldots, r$ do
8:             $\alpha_k^{(i)} = \left(\prod_{j=1}^{k-1} (\mathbf{x}_j^{(0)})^T A_j^{(i)} \mathbf{x}_j^{(0)}\right)\left(\prod_{j=k+1}^{d} (\mathbf{x}_j^{(1)})^T A_j^{(i)} \mathbf{x}_j^{(1)}\right)$
9:         end for
10:        $\mathbf{x}_k^{(0)}$ solves $\left(\sum_{i=1}^r \alpha_k^{(i)} A_k^{(i)}\right)\mathbf{x}_k = (\mathbf{x}_1^{(0)} \otimes \cdots \otimes \mathbf{x}_{k-1}^{(0)} \otimes \mathrm{id}_{N_k} \otimes \mathbf{x}_k^{(0)} \otimes \cdots \otimes \mathbf{x}_d^{(0)})^T \mathbf{b}$
11:    end for
12:    iter = iter + 1.
13:    distance = $\max_{1\le i\le d} \|\mathbf{x}_i^{(0)} - \mathbf{x}_i^{(1)}\|_2$.
14: end while
```

Definition 3.1. *Given a matrix $A \in \mathbb{R}^{N\times N}$, where $N = N_1 \cdots N_d$, we say that A is a Laplacian-like matrix if there exist matrices $A_i \in \mathbb{R}^{N_i \times N_i}$ for $1 \le i \le d$ be such that*

$$A = \sum_{i=1}^d A_i \otimes \mathrm{id}_{[N_i]} \doteq \sum_{i=1}^d \mathrm{id}_{N_1} \otimes \cdots \otimes \mathrm{id}_{N_{i-1}} \otimes A_i \otimes \mathrm{id}_{N_{i+1}} \otimes \cdots \otimes \mathrm{id}_{N_d}, \tag{3.1}$$

where id_{N_j} is the identity matrix of size $N_j \times N_j$.

It is not difficult to see that the set of Laplacian-like matrices is a linear subspace $\mathbb{R}^{N\times N}$ of matrices satisfying the property (2.8). From now on, we will denote by $\mathcal{L}\left(\mathbb{R}^{N\times N}\right)$ the subspace of Laplacian-like matrices in $\mathbb{R}^{N\times N}$ for a fixed decomposition of $N = N_1 \cdots N_d$.

Now, given a matrix $A \in \mathbb{R}^{N\times N}$, our goal is to solve the following optimization problem:

$$\min_{L \in \mathcal{L}(\mathbb{R}^{N\times N})} \|A - L\|_2. \tag{3.2}$$

Clearly, if we denote by $\Pi_{\mathcal{L}(\mathbb{R}^{N\times N})}$ the orthogonal projection onto the linear subspace $\mathcal{L}\left(\mathbb{R}^{N\times N}\right)$, then $L_A := \Pi_{\mathcal{L}(\mathbb{R}^{N\times N})}(A)$ is the solution of (3.2). Observe that $\|A - L_A\|_2 = 0$ if and only if $A \in \mathcal{L}\left(\mathbb{R}^{N\times N}\right)$.

We are interested in trying to achieve a structure similar to (3.1) to study the matrices of large-dimensional problems. We search an algorithm that allows one to construct, for a given matrix A, its Laplacian-like best approximation L_A.

To do this, we will use the following theorem, which describes a particular decomposition of the space of matrices $\mathbb{R}^{N\times N}$. Observe that the linear subspace $\mathrm{span}\{\mathrm{id}_N\}$ in $\mathbb{R}^{N\times N}$ has, as the orthogonal space, the following null trace matrices:

$$\mathrm{span}\{\mathrm{id}_n\}^\perp = \{A \in \mathbb{R}^{n\times n} : \mathrm{tr}(A) = 0\},$$

with respect to the inner product $\langle A, B\rangle_{\mathbb{R}^{N\times N}} = \mathrm{tr}(A^T B)$.

Theorem 3.2. *Consider $\left(\mathbb{R}^{N\times N}, \|\cdot\|_2\right)$ as a Hilbert space where $N = N_1 \cdots N_d$. Then, there exists a decomposition*

$$\mathbb{R}^{N\times N} = \operatorname{span}\{\mathrm{id}_N\} \oplus \mathfrak{h}_N = \mathcal{L}\left(\mathbb{R}^{N\times N}\right) \oplus \mathcal{L}\left(\mathbb{R}^{N\times N}\right)^{\perp},$$

where $\mathfrak{h}_N = \operatorname{span}\{\mathrm{id}_N\}^{\perp}$ is the orthogonal complement of the linear subspace generated by the identity matrix. Moreover,

$$\mathcal{L}\left(\mathbb{R}^{N\times N}\right) = \operatorname{span}\{\mathrm{id}_N\} \oplus \Delta, \tag{3.3}$$

where $\Delta = \mathfrak{h}_N \cap \mathcal{L}(\mathbb{R}^{N\times N})$. Furthermore, $\mathcal{L}(\mathbb{R}^{N\times N})^{\perp}$ is a subspace of $\mathfrak{h}_N$ and

$$\Delta = \bigoplus_{i=1}^{d} \operatorname{span}\{\mathrm{id}_{N_1}\} \otimes \cdots \otimes \operatorname{span}\{\mathrm{id}_{N_{i-1}}\} \otimes \operatorname{span}\{\mathrm{id}_{N_i}\}^{\perp} \otimes \operatorname{span}\{\mathrm{id}_{N_{i+1}}\} \otimes \cdots \otimes \operatorname{span}\{\mathrm{id}_{N_d}\}.$$

Proof. It follows from Lemma 3.1, Theorem 3.1 and Theorem 3.2 in [8]. □

The above theorem allows us to compute the projection of matrix A onto $\mathcal{L}(\mathbb{R}^{N\times N})$ as follows. Denote by Π_i the orthogonal projection of $\mathbb{R}^{N\times N}$ onto the linear subspace

$$\operatorname{span}\{\mathrm{id}_{N_1}\} \otimes \cdots \otimes \operatorname{span}\{\mathrm{id}_{N_{i-1}}\} \otimes \operatorname{span}\{\mathrm{id}_{N_i}\}^{\perp} \otimes \operatorname{span}\{\mathrm{id}_{N_{i+1}}\} \otimes \cdots \otimes \operatorname{span}\{\mathrm{id}_{N_d}\}$$

for $1 \leq i \leq d$. Thus, $\sum_{i=1}^{k} \Pi_i$ is the orthogonal projection of $\mathbb{R}^{N\times N}$ onto the linear subspace Δ. In consequence, by using (3.3), we have

$$\frac{\operatorname{tr}(A)}{N}\mathrm{id}_N + \sum_{i=1}^{d} \Pi_i(A) = \underset{L\in\mathcal{L}\left(\mathbb{R}^{N\times N}\right)}{\arg\min} \|A - L\|_2. \tag{3.4}$$

If we further analyze (3.4), we observe that the second term on the left is of the form

$$\sum_{i=1}^{d} \Pi_i(A) = \sum_{i=1}^{d} \mathrm{id}_{N_1} \otimes \cdots \otimes \mathrm{id}_{N_{i-1}} \otimes X_i \otimes \mathrm{id}_{N_{i+1}} \otimes \cdots \otimes \mathrm{id}_{N_d},$$

and that it has only $(N_1^2 + \cdots + N_d^2 - d)$-degrees of freedom (recall that $\dim \operatorname{span}\{\mathrm{id}_{N_i}\}^{\perp} = N_i^2 - 1$). In addition, due to the tensor structure of the products, the unknowns x_l of X_k are distributed in the form of a block so that we can calculate which will be the entries of the matrix A that we can approximate. Therefore, to obtain the value of each x_l, we only need to calculate which is the value that best approximates the entries (i, j) of the original matrix that are in the same position as x_l.

In our next result, we will see how to carry out this procedure. To do this, we make the following observation. Given a matrix $A = (a_{i,j}) \in \mathbb{R}^{KL\times KL}$ for some integers $K, L > 1$, we can write A as a matrix block:

$$A = \begin{pmatrix} A_{1,1}^{(K,L)} & A_{1,2}^{(K,L)} & \cdots & A_{1,L}^{(K,L)} \\ A_{2,1}^{(K,L)} & A_{2,2}^{(K,L)} & \cdots & A_{2,L}^{(K,L)} \\ \vdots & \vdots & \ddots & \vdots \\ A_{L,1}^{(K,L)} & A_{L,2}^{(K,L)} & \cdots & A_{L,L}^{(K,L)} \end{pmatrix}, \tag{3.5}$$

where the block $A_{i,j}^{(K,L)} \in \mathbb{R}^{K\times K}$ for $1 \le i, j \le L$ is given by

$$A_{i,j}^{(K,L)} = \begin{pmatrix} a_{(i-1)K+1,(j-1)K+1} & \cdots & a_{(i-1)K+1,jK} \\ \vdots & \ddots & \vdots \\ a_{iK,(j-1)K+1} & \cdots & a_{iK,jK} \end{pmatrix}.$$

Moreover,

$$\|A\|^2_{\mathbb{R}^{KL\times KL}} = \sum_{i=1}^{KL}\sum_{j=1}^{KL} a_{i,j}^2 = \sum_{r=1}^{L}\sum_{s=1}^{L} \|A_{r,s}^{(K,L)}\|^2_{\mathbb{R}^{K\times K}}.$$

Observe that K and L can be easily interchanged. To simplify the notation, from now on given $N = N_1N_2\cdots N_d$, we denote it by $N_{[k]} = N_1\cdots N_{k-1}N_{k+1}\cdots N_d$ for each $1 \le k \le d$.

Theorem 3.3. *Let $A \in \mathbb{R}^{N\times N}$, with $N = N_1\cdots N_d$. For each fixed $1 \le k \le d$, consider the linear function $P_k : \mathbb{R}^{N_k\times N_k} \longrightarrow \mathbb{R}^{N\times N}$ given by*

$$P_k(X_k) := \mathrm{id}_{N_1} \otimes \cdots \otimes \mathrm{id}_{N_{k-1}} \otimes X_k \otimes \mathrm{id}_{N_{k+1}} \otimes \cdots \otimes \mathrm{id}_{N_d}.$$

Then, the solution of the minimization problem

$$\min_{X_k\in\mathbb{R}^{N_k\times N_k}} \|A - P_k(X_k)\|_2 \tag{3.6}$$

is given by

$$(X_k)_{i,j} = \begin{cases} \dfrac{1}{N_{[1]}}\displaystyle\sum_{n=1}^{N_{[1]}} a_{(i-1)N_{[1]}+n,(j-1)N_{[1]}+n} & \text{if} \quad k=1, \\ \dfrac{1}{N_{[k]}}\displaystyle\sum_{m=1}^{N_{k+1}\cdots N_d}\left(\sum_{n=1}^{N_1\cdots N_{k-1}} A_{n,n}^{(N_k\cdots N_d, N_1\cdots N_{k-1})}\right)_{(i-1)N_{k+1}\cdots N_d+m,(j-1)N_{k+1}\cdots N_d+m} & \text{if} \quad 1<k<d, \\ \dfrac{1}{N_{[d]}}\left(\displaystyle\sum_{n=1}^{N_{[d]}} A_{n,n}^{(N_d,N_{[d]})}\right)_{i,j} & \text{if} \quad k=d. \end{cases}$$

Proof. First, let us observe that $\mathrm{id}_{N_1} \otimes \cdots \otimes \mathrm{id}_{N_k} = \mathrm{id}_{N_1\cdots N_k}$; so, we can find three different situations in the calculation of the projections:

(1). $P_1(A) = X_1 \otimes \mathrm{id}_{N_{[1]}}$; in this case,

$$P_1(X_1) = \begin{pmatrix} (X_1)_{1,1}\mathrm{id}_{N_{[1]}} & (X_1)_{1,2}\mathrm{id}_{N_{[1]}} & \ldots & (X_1)_{1,N_1}\mathrm{id}_{N_{[1]}} \\ (X_1)_{2,1}\mathrm{id}_{N_{[1]}} & (X_1)_{2,2}\mathrm{id}_{N_{[1]}} & \ldots & (X_1)_{2,N_1}\mathrm{id}_{N_{[1]}} \\ \vdots & \vdots & \ddots & \vdots \\ (X_1)_{N_1,1}\mathrm{id}_{N_{[1]}} & (X_1)_{N_1,2}\mathrm{id}_{N_{[1]}} & \ldots & (X_1)_{N_1,N_1}\mathrm{id}_{N_{[1]}} \end{pmatrix} \in \mathbb{R}^{N_{[1]}N_1\times N_{[1]}N_1}.$$

(2). $P_d(X_d) = \mathrm{id}_{N_{[d]}} \otimes X_d$; in this case,

$$P_d(X_d) = \begin{pmatrix} X_d & O_d & \cdots & O_d \\ O_d & X_d & \cdots & O_d \\ \vdots & \vdots & \ddots & \vdots \\ O_d & O_d & \cdots & X_d \end{pmatrix} \in \mathbb{R}^{N_d N_{[d]} \times N_d N_{[d]}},$$

where O_d denotes the zero matrix in $\mathbb{R}^{N_d \times N_d}$.

(3). $P_i(X_i) = \mathrm{id}_{N_1 \cdots N_{i-1}} \otimes X_i \otimes \mathrm{id}_{N_{i+1} \cdots N_d}$ for $i = 2, \ldots, d-1$; in this case, for a fixed $2 \le i \le d-1$, we write $N_\ell = N_1 \cdots N_{i-1}$ and $N_r = N_{i+1} \cdots N_d$. Thus,

$$\begin{aligned} P_i(X_i) &= \mathrm{id}_{N_\ell} \otimes X_i \otimes \mathrm{id}_{N_r} \\ &= \mathrm{id}_{N_\ell} \otimes \begin{pmatrix} (X_i)_{1,1}\mathrm{id}_{N_r} & (X_i)_{1,2}\mathrm{id}_{N_r} & \ldots & (X_i)_{1,N_1}\mathrm{id}_{N_r} \\ (X_i)_{2,1}\mathrm{id}_{N_r} & (X_i)_{2,2}\mathrm{id}_{N_r} & \ldots & (X_i)_{2,N_1}\mathrm{id}_{N_r} \\ \vdots & \vdots & \ddots & \vdots \\ (X_i)_{N_1,1}\mathrm{id}_{N_r} & (X_i)_{N_1,2}\mathrm{id}_{N_r} & \ldots & (X_i)_{N_1,N_1}\mathrm{id}_{N_r} \end{pmatrix} \\ &= \begin{pmatrix} X_i \otimes \mathrm{id}_{N_r} & O_i \otimes \mathrm{id}_{N_r} & \cdots & O_i \otimes \mathrm{id}_{N_r} \\ O_i \otimes \mathrm{id}_{N_r} & X_i \otimes \mathrm{id}_{N_r} & \cdots & O_i \otimes \mathrm{id}_{N_r} \\ \vdots & \vdots & \ddots & \vdots \\ O_i \otimes \mathrm{id}_{N_r} & O_i \otimes \mathrm{id}_{N_r} & \cdots & X_i \otimes \mathrm{id}_{N_r} \end{pmatrix} \in \mathbb{R}^{(N_i N_r)N_\ell \times (N_i N_r)N_\ell}. \end{aligned}$$

In either case, a difference of the form

$$\min_{X_k \in \mathbb{R}^{N_k \times N_k}} \|A - P_k(A)\|_2$$

must be minimized. To this end, we will consider each case A as a block matrix $A \in \mathbb{R}^{KL \times KL}$ in the form of (3.5).

Case 1: For $P_1(X_1)$, we take $K = N_{[1]}$ and $L = N_1$; hence,

$$A - P_1(X_1) = \begin{pmatrix} A^{(K,L)}_{1,1} - (X_1)_{1,1}\mathrm{id}_{N_{[1]}} & A^{(K,L)}_{1,2} - (X_1)_{1,2}\mathrm{id}_{N_{[1]}} & \ldots & A^{(K,L)}_{1,N_1} - (X_1)_{1,N_1}\mathrm{id}_{N_{[1]}} \\ A^{(K,L)}_{2,1} - (X_1)_{2,1}\mathrm{id}_{N_{[1]}} & A^{(K,L)}_{2,2} - (X_1)_{2,2}\mathrm{id}_{N_{[1]}} & \ldots & A^{(K,L)}_{2,N_1} - (X_1)_{2,N_1}\mathrm{id}_{N_{[1]}} \\ \vdots & \vdots & \ddots & \vdots \\ A^{(K,L)}_{N_1,1} - (X_1)_{N_1,1}\mathrm{id}_{N_{[1]}} & A^{(K,L)}_{N_1,2} - (X_1)_{N_1,2}\mathrm{id}_{N_{[1]}} & \ldots & A^{(K,L)}_{N_1,N_1} - (X_1)_{N_1,N_1}\mathrm{id}_{N_{[1]}} \end{pmatrix}.$$

In this situation, we have

$$\|A - P_1(X_1)\|^2_{\mathbb{R}^{N \times N}} = \sum_{i=1}^{N_1} \sum_{j=1}^{N_1} \|A^{(K,L)}_{i,j} - (X_1)_{i,j}\mathrm{id}_{N_{[1]}}\|^2_{\mathbb{R}^{N_{[1]} \times N_{[1]}}};$$

hence, we wish for each $1 \le i$ and $j \le N_1$ to yield

$$(X_1)_{i,j} = x \in \arg\min_{x \in \mathbb{R}} \|A^{(K,L)}_{i,j} - x\,\mathrm{id}_{N_{[1]}}\|^2_{\mathbb{R}^{N_{[1]} \times N_{[1]}}} = \arg\min_{x \in \mathbb{R}} \sum_{n=1}^{N_{[1]}} (a_{(i-1)N_{[1]}+n,(j-1)N_{[1]}+n} - x)^2.$$

Thus, it is not difficult to see that

$$(X_1)_{i,j} = \frac{1}{N_{[1]}} \sum_{n=1}^{N_{[1]}} a_{(i-1)N_{[1]}+n,(j-1)N_{[1]}+n}$$

for $1 \leq i, j \leq N_1$.
Case 2: For $P_d(X_d)$, we take $K = N_d$ and $L = N_{[d]}$; hence,

$$A - P_d(X_d) = \begin{pmatrix} A_{1,1}^{(K,L)} - X_d & A_{1,2}^{(K,L)} - O_d & \cdots & A_{1,N_{[d]}}^{(K,L)} - O_d \\ A_{2,1}^{(K,L)} - O_d & A_{2,2}^{(K,L)} - X_d & \cdots & A_{2,N_{[d]}}^{(K,L)} - O_d \\ \vdots & \vdots & \ddots & \vdots \\ A_{N_{[d]},1}^{(K,L)} - O_d & A_{N_{[d]},2}^{(K,L)} - O_d & \cdots & A_{N_{[d]},N_{[d]}}^{(K,L)} - X_d \end{pmatrix}.$$

Now, we have

$$\|A - P_d(X_d)\|^2_{\mathbb{R}^{N\times N}} = \sum_{i=1}^{N_{[d]}} \|A_{i,i}^{(K,L)} - X_d\|^2_{\mathbb{R}^{N_d\times N_d}} + \sum_{i=1,j=1,i\neq j}^{N_[d]} \|A_{i,i}^{(K,L)}\|^2_{\mathbb{R}^{N_d\times N_d}}.$$

Thus, $X_d \in \mathbb{R}^{N_d\times N_d}$ minimizes $\|A - P_d(X_d)\|^2_{\mathbb{R}^{N\times N}}$ if and only if

$$X_d \in \arg\min_{X\in\mathbb{R}^{N_d\times N_d}} \sum_{i=1}^{N_{[d]}} \|A_{i,i}^{(K,L)} - X\|^2_{\mathbb{R}^{N_d\times N_d}}.$$

In consequence,

$$X_d = \frac{1}{N_{[d]}} \sum_{i=1}^{N_{[d]}} A_{i,i}^{(K,L)}.$$

Case 3: For $P_i(X_i)$, we take $K = N_iN_r$ and $L = N_\ell$; hence,

$$A - P_i(X_i) = \begin{pmatrix} A_{1,1}^{(K,L)} - X_i \otimes \mathrm{id}_{N_r} & A_{1,2}^{(K,L)} - O_i \otimes \mathrm{id}_{N_r} & \cdots & A_{1,N_\ell}^{(K,L)} - O_i \otimes \mathrm{id}_{N_r} \\ A_{2,1}^{(K,L)} - O_i \otimes \mathrm{id}_{N_r} & A_{2,2}^{(K,L)} - X_i \otimes \mathrm{id}_{N_r} & \cdots & A_{1,N_\ell}^{(K,L)} - O_i \otimes \mathrm{id}_{N_r} \\ \vdots & \vdots & \ddots & \vdots \\ A_{N_\ell,1}^{(K,L)} - O_i \otimes \mathrm{id}_{N_r} & A_{N_\ell,2}^{(K,L)} - O_i \otimes \mathrm{id}_{N_r} & \cdots & A_{N_\ell,N_\ell}^{(K,L)} - X_i \otimes \mathrm{id}_{N_r} \end{pmatrix}.$$

In this case,

$$\|A - P_i(X_i)\|^2_{\mathbb{R}^{N\times N}} = \sum_{n=1}^{N_\ell} \|A_{n,n}^{(K,L)} - X_i \otimes \mathrm{id}_{N_r}\|^2_{\mathbb{R}^{N_iN_r\times N_iN_r}} + \sum_{n=1,j=1,n\neq j}^{N_\ell} \|A_{n,j}^{(K,L)}\|^2_{\mathbb{R}^{N_iN_r\times N_iN_r}},$$

so we need to solve the following problem:

$$\min_{X\in\mathbb{R}^{N_i\times N_i}} \sum_{n=1}^{N_\ell} \|A_{n,n}^{(K,L)} - X \otimes \mathrm{id}_{N_r}\|^2_{\mathbb{R}^{N_iN_r\times N_iN_r}}. \tag{3.7}$$

Since $X \otimes \mathrm{id}_{N_r} \in \mathbb{R}^{N_i \times N_i} \otimes \mathrm{span}\{\mathrm{id}_{N_r}\}$, we can write (3.7) as

$$\min_{Z \in \mathbb{R}^{N_i \times N_i} \otimes \mathrm{span}\{\mathrm{id}_{N_r}\}} \sum_{n=1}^{N_\ell} \|A_{n,n}^{(K,L)} - Z\|^2_{\mathbb{R}^{N_i N_r \times N_i N_r}}. \tag{3.8}$$

Observe that

$$A^* = (a^*_{u,v}) = \frac{1}{N_\ell} \sum_{n=1}^{N_\ell} A_{n,n}^{(K,L)} = \arg \min_{U \in \mathbb{R}^{N_i N_r \times N_i N_r}} \sum_{n=1}^{N_\ell} \|A_{n,n}^{(K,L)} - U\|^2_{\mathbb{R}^{N_i N_r \times N_i N_r}}.$$

To simplify the notation, we write $\mathcal{U} := \mathbb{R}^{N_i \times N_i} \otimes \mathrm{span}\{\mathrm{id}_{N_r}\}$. Then, we have the following orthogonal decomposition, $\mathbb{R}^{N_i N_r \times N_i N_r} = \mathcal{U} \oplus \mathcal{U}^\perp$. Denote by $\Pi_{\mathcal{U}}$ the orthogonal projection onto the linear subspace $\mathcal{U}$. Then, for each $Z \in \mathcal{U}$, we have

$$\begin{aligned} \|A_{n,n}^{(K,L)} - Z\|^2 &= \|(\mathrm{id} - \Pi_{\mathcal{U}})(A_{n,n}^{(K,L)}) + \Pi_{\mathcal{U}}(A_{n,n}^{(K,L)}) - Z\|^2 \\ &= \|(\mathrm{id} - \Pi_{\mathcal{U}})(A_{n,n}^{(K,L)})\|^2 + \|\Pi_{\mathcal{U}}(A_{n,n}^{(K,L)}) - Z\|^2, \end{aligned}$$

because $(\mathrm{id} - \Pi_{\mathcal{U}})(A_{n,n}^{(K,L)}) \in \mathcal{U}^\perp$ and $\Pi_{\mathcal{U}}(A_{n,n}^{(K,L)}) - Z \in \mathcal{U}$. In consequence, the process of solving (3.8) is equivalent that for solving the following optimization problem:

$$\min_{Z \in \mathcal{U}} \sum_{n=1}^{N_\ell} \|\Pi_{\mathcal{U}}(A_{n,n}^{(K,L)}) - Z\|^2_{\mathbb{R}^{N_i N_r \times N_i N_r}}. \tag{3.9}$$

Thus,

$$Z^* = \frac{1}{N_\ell} \sum_{n=1}^{N_\ell} \Pi_{\mathcal{U}}(A_{n,n}^{(K,L)}) = \arg\min_{Z \in \mathcal{U}} \sum_{n=1}^{N_\ell} \|\Pi_{\mathcal{U}}(A_{n,n}^{(K,L)}) - Z\|^2_{\mathbb{R}^{N_i N_r \times N_i N_r}},$$

that is, $Z^* = \Pi_{\mathcal{U}}(A^*)$; hence,

$$Z^* = \arg\min_{Z \in \mathcal{U}} \|A^* - Z\|^2 = X_i \otimes \mathrm{id}_{N_r} = \arg \min_{X \in \mathbb{R}^{N_i \times N_i}} \|A^* - X \otimes \mathrm{id}_{N_r}\|^2_{\mathbb{R}^{N_i N_r \times N_i N_r}}.$$

Proceeding in a similar way as in Case 1, we obtain

$$(X_i)_{u,v} = \frac{1}{N_r} \sum_{m=1}^{N_r} a^*_{(u-1)N_r+m,(v-1)N_r+m} = \frac{1}{N_r} \frac{1}{N_l} \sum_{m=1}^{N_r} \left(\sum_{n=1}^{N_l} A_{n,n}^{(K,L)} \right)_{(u-1)N_r+m,(v-1)N_r+m}$$

for $1 \leq u, v \leq N_i$. This concludes the proof of the theorem. □

To conclude, we obtain the following useful corollary.

Corollary 3.4. *Let* $A \in \mathbb{R}^{N \times N}$, *with* $N = N_1 \cdots N_d$. *For each fixed* $1 \leq k \leq d$, *consider the linear function* $P_k : \mathbb{R}^{N_k \times N_k} \longrightarrow \mathbb{R}^{N \times N}$ *given by*

$$P_k(X_k) := \mathrm{id}_{N_1} \otimes \cdots \otimes \mathrm{id}_{N_{k-1}} \otimes X_k \otimes \mathrm{id}_{N_{k+1}} \otimes \cdots \otimes \mathrm{id}_{N_d}.$$

For each $1 \leq k \leq d$, *let* $X_k \in \mathbb{R}^{N_k \times N_k}$ *be the solution of the optimization problem* (3.6). *Then,*

$$L_A = \frac{\mathrm{tr}(A)}{N} \mathrm{id}_N + \sum_{k=1}^{d} P_k \left(X_k - \frac{\mathrm{tr}(X_k)}{N_k} \mathrm{id}_{N_k} \right) = \arg\min_{L \in \mathcal{L}(\mathbb{R}^{N \times N})} \|A - L\|_2. \tag{3.10}$$

Proof. Observe that, for $1 \leq k \leq d$, the matrix X_k satisfies

$$P_k(X_k) = \arg\min_{Z \in \mathfrak{h}^{(k)}} \|A - Z\|_2,$$

where

$$\mathfrak{h}^{(k)} := \operatorname{span}\{\mathrm{id}_{N_1}\} \otimes \cdots \otimes \operatorname{span}\{\mathrm{id}_{N_{k-1}}\} \otimes \mathbb{R}^{N_k \times N_k} \otimes \operatorname{span}\{\mathrm{id}_{N_{k+1}}\} \otimes \cdots \otimes \operatorname{span}\{\mathrm{id}_{N_d}\}$$

is a linear subspace of $\mathbb{R}^{N \times N}$ that is linearly isomorphic to $\mathbb{R}^{N_k \times N_K}$. Since $\mathbb{R}^{N_k \times N_k} = \operatorname{span}\{\mathrm{id}_{N_k}\} \oplus \operatorname{span}\{\mathrm{id}_{N_k}\}^{\perp}$, then

$$X_k = \frac{\operatorname{tr}(X_k)}{N_k}\mathrm{id}_{N_k} + \left(X_k - \frac{\operatorname{tr}(X_k)}{N_k}\mathrm{id}_{N_k}\right);$$

hence

$$\begin{aligned} P_k(X_k) &= P_k\left(\frac{\operatorname{tr}(X_k)}{N_k}\mathrm{id}_{N_k}\right) + P_k\left(X_k - \frac{\operatorname{tr}(X_k)}{N_k}\mathrm{id}_{N_k}\right) \\ &= \frac{\operatorname{tr}(X_k)}{N_k}\mathrm{id}_{N} + P_k\left(X_k - \frac{\operatorname{tr}(X_k)}{N_k}\mathrm{id}_{N_k}\right). \end{aligned}$$

We can conclude that $\Pi_k(A) = P_k\left(X_k - \frac{\operatorname{tr}(X_k)}{N_k}\mathrm{id}_{N_k}\right)$; recall that Π_k is the orthogonal projection of $\mathbb{R}^{N \times N}$ onto the linear subspace

$$\operatorname{span}\{\mathrm{id}_{N_1}\} \otimes \cdots \otimes \operatorname{span}\{\mathrm{id}_{N_{k-1}}\} \otimes \operatorname{span}\{\mathrm{id}_{N_k}\}^{\perp} \otimes \operatorname{span}\{\mathrm{id}_{N_{k+1}}\} \otimes \cdots \otimes \operatorname{span}\{\mathrm{id}_{N_d}\}.$$

From (3.4), the corollary is proved. □

4. The best Laplacian approximation for the discretization of second-order partial differential equations without mixing derivatives

In this section, we consider the general equation of a generic second-order partial differential equation without mixing derivatives with homogeneous boundary conditions. More precisely, let

$$\alpha \mathbf{u}_{xx} + \beta \mathbf{u}_{yy} + \gamma \mathbf{u}_x + \delta \mathbf{u}_y + \mu \mathbf{u} = \mathbf{f} \text{ for } (x, y) \in (0, 1) \times (0, 1), \tag{4.1}$$

$$\mathbf{u}(x, 0) = \mathbf{u}(x, 1) = \mathbf{u}(0, y) = \mathbf{u}(1, y) = 0 \text{ for all } 0 \leq x \leq 1 \text{ and } 0 \leq y \leq 1. \tag{4.2}$$

We discretize (4.1) with the help of the following derivative approximations:

$$\mathbf{u}_x(x, y) \approx \frac{\mathbf{u}(x_{i+1}, y_j) - \mathbf{u}(x_{i-1}, y_j)}{2h}, \quad \mathbf{u}_y(x, y) \approx \frac{\mathbf{u}(x_i, y_{j+1}) - \mathbf{u}(x_i, y_{j-1})}{2k},$$

and

$$\begin{aligned} \mathbf{u}_{xx}(x, y) &\approx \frac{\mathbf{u}(x_{i+1}, y_j) - 2\mathbf{u}(x_i, y_j) + \mathbf{u}(x_{i-1}, y_j)}{h^2}, \\ \mathbf{u}_{yy}(x, y) &\approx \frac{\mathbf{u}(x_i, y_{j+1}) - 2\mathbf{u}(x_i, y_j) + \mathbf{u}(x_i, y_{j-1})}{k^2} \end{aligned}$$

for $i = 1, \ldots, N$ and $j = 1, \ldots, M$. From (4.2), we have that $\mathbf{u}(x, y_0) = \mathbf{u}(x, y_{M+1}) = \mathbf{u}(x_0, y) = \mathbf{u}(x_{N+1}, y) = 0$ for all $0 \leq x \leq 1$ and $0 \leq y \leq 1$.

Next, in order to obtain a linear system, we put $\mathbf{u}_\ell := \mathbf{u}(x_i, y_j)$ and $\mathbf{f}_\ell := \mathbf{f}(x_i, y_j)$, where $\ell := (i-1)M + j$ for $1 \le i \le N$ and $1 \le j \le M$. In this way, the represented mesh is traversed as shown in Figure 1, and the elements $U = (\mathbf{u}_\ell)_{\ell=1}^{MN}$ and $F = \{\mathbf{f}_\ell\}_{\ell=1}^{MN}$ are column vectors. It allows us to represent (4.1) and (4.2) as the linear system $AF = U$, where A is the $MN \times MN$-block matrix

$$A = \begin{pmatrix} T & D_1 & & & \\ D_2 & T & D_1 & & \\ & \ddots & \ddots & \ddots & \\ & & D_2 & T & D_1 \\ & & & D_2 & T \end{pmatrix} \tag{4.3}$$

for $T \in \mathbb{R}^{M\times M}$, given by

$$T = \begin{pmatrix} 2\mu h^2k^2 - 4\alpha k^2 - 4\beta h^2 & 2\beta h^2 + \delta h^2 k & 0 & \dots & 0 \\ 2\beta h^2 - \delta h^2 k & 2\mu h^2k^2 - 4\alpha k^2 - 4\beta h^2 & 2\beta h^2 + \delta h^2 k & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & 2\beta h^2 - \delta h^2 k & 2\mu h^2k^2 - 4\alpha k^2 - 4\beta h^2 \end{pmatrix},$$

and $D_1, D_2 \in \mathbb{R}^{M\times M}$ are the diagonal matrices:

$$D_1 = (2\alpha k^2 + \gamma h k^2)\mathrm{id}_M, \quad D_2 = (2\alpha k^2 - \gamma h k^2)\mathrm{id}_M.$$

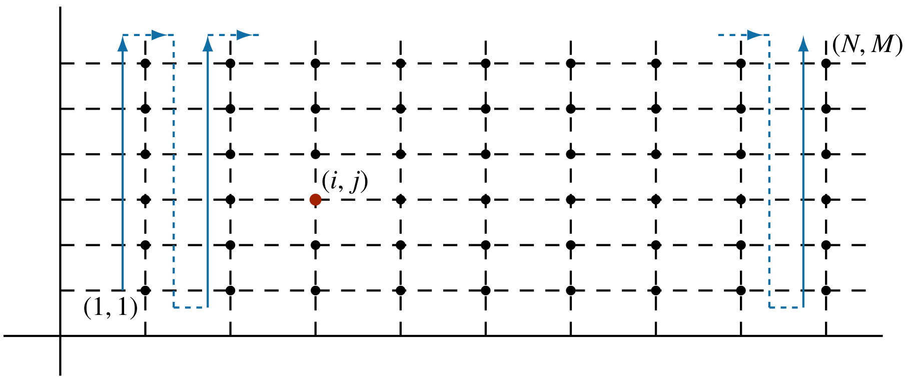

Figure 1. Proceeding from $(1,1)$ to $(1,M)$; $(2,1),\dots,(2,M)$; and, ending at $(N,1),\dots,(N,M)$.

In this case, $\mathrm{tr}(A) = NM(2\mu h^2k^2 - 4\alpha k^2 - 4\beta h^2)$; so, instead of looking for L_A, as in (3.10), we will look for $L_{\hat{A}}$, where

$$\hat{A} = \left(A - \frac{\mathrm{tr}(A)}{NM}\mathrm{id}_{NM}\right)$$

has a null trace. Proceeding according to Theorem 3.3 for sizes $N_1 = N$ and $N_2 = M$, we obtain the following decomposition:

$$X_1 = \begin{pmatrix} 0 & 2\alpha k^2 + \gamma hk^2 & 0 & \dots & 0 \\ 2\alpha k^2 - \gamma hk^2 & 0 & 2\alpha k^2 + \gamma hk^2 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 2\alpha k^2 - \gamma hk^2 & 0 & 2\alpha k^2 + \gamma hk^2 \\ 0 & \dots & 0 & 2\alpha k^2 - \gamma hk^2 & 0 \end{pmatrix} \in \mathbb{R}^{N\times N}$$

and

$$X_2 = \begin{pmatrix} 0 & 2\beta h^2 + \delta h^2 k & 0 & \dots & 0 \\ 2\beta h^2 - \delta h^2 k & 0 & 2\beta h^2 + \delta h^2 k & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 2\beta h^2 - \delta h^2 k & 0 & 2\beta h^2 + \delta h^2 k \\ 0 & \dots & 0 & 2\beta h^2 - \delta h^2 k & 0 \end{pmatrix} \in \mathbb{R}^{M\times M}.$$

We remark that $\mathrm{tr}(X_1) = \mathrm{tr}(X_2) = 0$. Moreover, the residual of the approximation $L_{\hat{A}}$ of $\hat{A}$ is $\|\hat{A} - L_{\hat{A}}\| = 0$. In consequence, we can write the original matrix A as

$$A = \frac{\mathrm{tr}(A)}{NM}\mathrm{id}_{NM} + X_1 \otimes \mathrm{id}_M + \mathrm{id}_N \otimes X_2.$$

Recall that the first term is

$$\frac{\mathrm{tr}(A)}{NM}\mathrm{id}_{NM} = \left(2\mu h^2k^2 - 4\alpha k^2 - 4\beta h^2\right)\cdot \mathrm{id}_{NM} = \left(2\mu h^2k^2 - 4\alpha k^2 - 4\beta h^2\right)\cdot \mathrm{id}_N \otimes \mathrm{id}_M;$$

hence, A can be written as

$$A = Z_1 \otimes \mathrm{id}_M + \mathrm{id}_N \otimes Z_2,$$

where

$$Z_1 = \begin{pmatrix} \mu h^2k^2 - 2\alpha k^2 - 2\beta h^2 & 2\alpha k^2 + \gamma hk^2 & 0 & \dots & 0 \\ 2\alpha k^2 - \gamma hk^2 & \mu h^2k^2 - 2\alpha k^2 - 2\beta h^2 & 2\alpha k^2 + \gamma hk^2 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 2\alpha k^2 - \gamma hk^2 & \mu h^2k^2 - 2\alpha k^2 - 2\beta h^2 & 2\alpha k^2 + \gamma hk^2 \\ 0 & \dots & 0 & 2\alpha k^2 - \gamma hk^2 & \mu h^2k^2 - 2\alpha k^2 - 2\beta h^2 \end{pmatrix}$$

is an $N \times N$-matrix and

$$Z_2 = \begin{pmatrix} \mu h^2k^2 - 2\alpha k^2 - 2\beta h^2 & 2\beta h^2 + \delta h^2 k & 0 & \dots & 0 \\ 2\beta h^2 - \delta h^2 k & \mu h^2k^2 - 2\alpha k^2 - 2\beta h^2 & 2\beta h^2 + \delta h^2 k & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 2\beta h^2 - \delta h^2 k & \mu h^2k^2 - 2\alpha k^2 - 2\beta h^2 & 2\beta h^2 + \delta h^2 k \\ 0 & \dots & 0 & 2\beta h^2 - \delta h^2 k & \mu h^2k^2 - 2\alpha k^2 - 2\beta h^2 \end{pmatrix}$$

is a $M \times M$-matrix.

Now, we can use this representation of A to implement the GROU Algorithm 1, together with the ALS strategy given by Algorithm 2, to solve the following linear system:

$$AU = (Z_1 \otimes \mathrm{id}_M + \mathrm{id}_N \otimes Z_2)U = F.$$

This study can be extended to high-dimensional equations, as occurs in [8] with the three-dimensional Poisson equation.

5. Numerical examples

Next, we are going to consider some particular equations to analyze their numerical behavior. In all cases, the characteristics of the computer used are as follows: 11th Gen Intel$^{(R)}$ Core$^{(TM)}$ i7-11370H @ 3.30GHz, RAM 16 GB, 64-bit operating system; and, Matlab version R2021b [13].

5.1. The Helmholtz equation

Let us consider the particular case of the second-order partial differential equation with $\alpha = \beta = 1$, $\mu = c^2$ and $\mathbf{f} = 0$, that is,

$$\mathbf{u}_{xx} + \mathbf{u}_{yy} + c^2\mathbf{u} = 0.$$

This is the 2D-Helmholtz equation. To obtain the linear system associated with the discrete problem, we need some boundary conditions; for example,

$$\begin{cases} \mathbf{u}(x,0) = \sin(\omega x) + \cos(\omega x) & \text{for} \quad 0 \le x \le, \\ \mathbf{u}(0,y) = \sin(\omega y) + \cos(\omega y) & \text{for} \quad 0 \le y \le T, \end{cases}$$

and

$$\begin{cases} \mathbf{u}(x,T) = \sin(\omega(x+T)) + \cos(\omega(x+T)) & \text{for} \quad 0 \le x \le, \\ \mathbf{u}(L,y) = \sin(\omega(y+L)) + \cos(\omega(y+L)) & \text{for} \quad 0 \le y \le T. \end{cases}$$

This initial value problem has a closed solution for $\omega = \dfrac{c}{\sqrt{2}}$,

$$\mathbf{u}(x,y) = \sin(\omega(x+y)) + \cos(\omega(x+y)).$$

From the above operations, and by taking $h = k$ for simplicity, we can write the matrix of the discrete linear system associated with the equation of Helmholtz as

$$A = \begin{pmatrix} 2c^2h^4 - 8h^2 & 2h^2 & 0 & \dots & 0 \\ 2h^2 & 2c^2h^4 - 8h^2 & 2h^2 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & 2h^2 & 2c^2h^4 - 8h^2 \end{pmatrix} \otimes \mathrm{id}_M + \mathrm{id}_N \otimes \begin{pmatrix} 0 & 2h^2 & 0 & \dots & 0 \\ 2h^2 & 0 & 2h^2 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & 2h^2 & 0 \end{pmatrix}$$

or, equivalently,

$$A = \begin{pmatrix} c^2h^4 - 4h^2 & 2h^2 & 0 & \dots & 0 \\ 2h^2 & c^2h^4 - 4h^2 & 2h^2 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & 2h^2 & c^2h^4 - 4h^2 \end{pmatrix} \otimes \mathrm{id}_M$$
$$+ \mathrm{id}_N \otimes \begin{pmatrix} c^2h^4 - 4h^2 & 2h^2 & 0 & \dots & 0 \\ 2h^2 & c^2h^4 - 4h^2 & 2h^2 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & 2h^2 & c^2h^4 - 4h^2 \end{pmatrix}.$$

If we solve this linear system $A\mathbf{u}_l = \hat{\mathbf{f}}_l$ for the case $c = \sqrt{2}$, $L = T =$, and with $N = M$, we obtain the temporary results shown in Figure 2. To carry out this experiment, we have used the following parameter values: for the GROU Algorithm 1: $\texttt{tol} = 2.2204e-16$; $\varepsilon = 2.2204e-16$; $\texttt{rank\_max} = 10$; ($\texttt{iter-max} = 5$ and $\varepsilon = 2.2204e-16$ were used to perform Algorithm 2); and, the number of nodes in $(0,1)^2$ (that is, the number of rows or columns of the matrix A) was increased from 10^2 to 200^2.

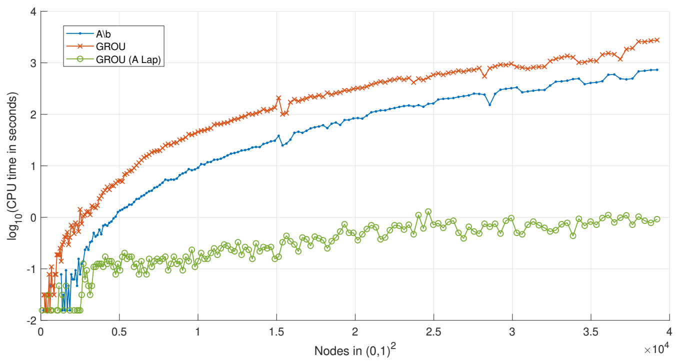

Figure 2. CPU time, in seconds, employed to solve the discrete Helmholtz initial value problem by using the Matlab command $A\backslash b$, the GROU Algorithm 1 and the GROU Algorithm 1 with A written as L_A, as obtained from Corollary 3.3.

To measure the goodness of the approximations obtained, we calculated the normalized errors, that is, the value of the difference, in absolute value, of the results obtained and the real solution, between the length of the solution, i.e.,

$$\varepsilon = \frac{|exact\ solution - approximate\ solution|}{N^2}$$

for the different approximations obtained. The values of these errors were of the order of 10^{-4} and can be seen in Figure 3.

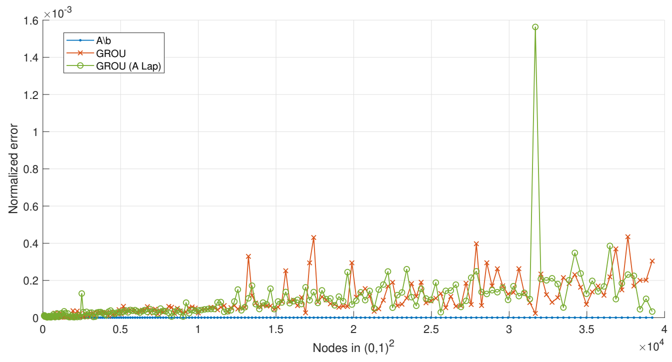

Figure 3. Normalized error between the solution of the discrete Helmholtz initial value problem and the solutions obtained by using the Matlab command $A\backslash b$, the GROU Algorithm 1 and the GROU Algorithm 1 with A written as L_A, as obtained from Corollary 3.3.

5.2. The Swift-Hohenberg equation

Now, let us consider the partial differential equation of order four:

$$\frac{\partial u}{\partial t} = \varepsilon - \left(1 + \frac{\partial^2}{\partial x^2}\right)^2 u, \tag{5.1}$$

with the boundary conditions

$$\begin{cases} u(x,0) = \sin(kx), \\ u(x,T) = \sin(kx)e^T \end{cases} \quad \text{for} \quad 0 \leq x \leq L, \tag{5.2}$$

and

$$u(0,t) = u(L,t) = 0 \quad \text{for} \quad 0 \leq t \leq T. \tag{5.3}$$

For $k = \sqrt{1 + \sqrt{\varepsilon - 1}}\ L = 2\pi/k$, the initial value problem (5.1)–(5.3) has the following as a solution:

$$u(x,t) = \sin(kx)e^t.$$

If we discretize the problem described by (5.1)–(5.3) as in the previous example with the same step size for both variables, h, we obtain a linear system of the form $A\mathbf{u}_l = \hat{\mathbf{f}}_l$, where A, in Laplacian-like form, is the matrix

$$A = \begin{pmatrix} 12 - 8h^2 + (2-2\varepsilon)h^4 & 4h^2 - 8 & 2 & 0 & \dots & 0 \\ 4h^2 - 8 & 12 - 8h^2 + (2-2\varepsilon)h^4 & 4h^2 - 8 & 2 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & 2 & 4h^2 - 8 & 12 - 8h^2 + (2-2\varepsilon)h^4 \end{pmatrix} \otimes \mathrm{id}_M$$
$$+ \mathrm{id}_N \otimes \begin{pmatrix} 0 & h^3 & 0 & \dots & 0 \\ -h^3 & 0 & h^3 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & -h^3 & 0 \end{pmatrix},$$

and $l = (i-1)M + j$ is the order established for the indices, with $1 \leq i \leq N$ and $1 \leq j \leq M$.

To perform a numerical experiment, we set $\varepsilon = 2$, $L = T = 2\pi$ and the same number of points for the two variables. At this point, we can solve the linear system associated with the Swift-Hohenberg discrete problem by using our tools: the Matlab command $A\backslash b$, the GROU Algorithm 1, and the GROU Algorithm 1, together with the ALS Algorithm 2 with A written in Laplacian-like form. In this case, we used the following parameter values in the algorithms: `tol` $= 2.2204e - 16$; $\varepsilon = 2.2204e - 16$; `rank_max` $= 10$ for the GROU Algorithm 1, with `iter-max` $= 5$ for the ALS step; and, the number of nodes in $(0, 2\pi)^2$ was increased from 10^2 to 200^2. Figure 4 shows the results obtained.

Again, we calculated the normalized errors to estimate the goodness of the approximations, the results of which are shown in Figure 5.

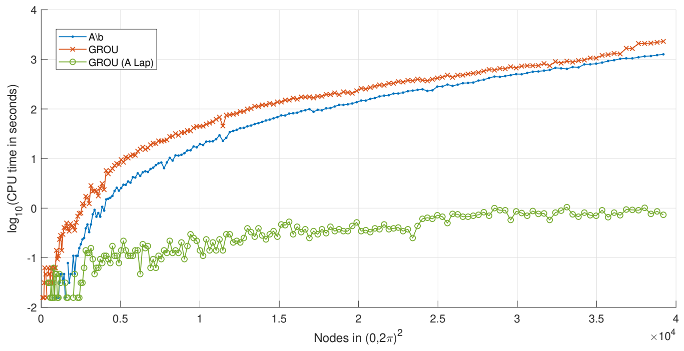

Figure 4. CPU time, in seconds, employed to solve the discrete Swift-Hohenberg initial value problem by using the Matlab command $A\backslash b$, the GROU Algorithm 1 and the GROU Algorithm 1 with A written in Laplacian form.

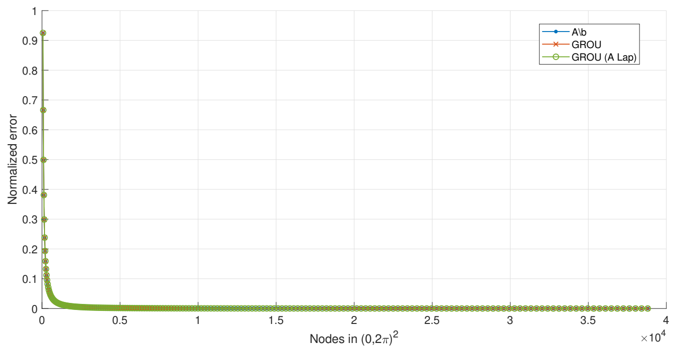

Figure 5. Normalized error between the solution of the discrete Swift-Hohenberg initial value problem and the solutions obtained by using the Matlab command $A\backslash b$, the GROU Algorithm 1 and the GROU Algorithm 1 with A written in Laplacian form.

6. Conclusions

In this work, we have studied the Laplacian decomposition algorithm, which, given any square matrix, calculates its best Laplacian approximation. Furthermore, in Theorem 3.3, we have shown how it is implemented optimally.

For us, the greatest interest in this algorithm lies in the computational improvement of combining it with the GROU Algorithm 1 to solve linear systems through the discretization of a partial derivative equation. Said improvement can be seen in the different numerical examples shown, where we have compared this procedure with the standard resolution of Matlab by means of the instruction $A\backslash b$.

This proposal is a new way of dealing with certain large-scale problems, where classical methods prove to be more inefficient.

Use of AI tools declaration

The authors declare that they have not used artificial intelligence tools in the creation of this article.

Acknowledgments

J. A. Conejero acknowledges funding from grant PID2021-124618NB-C21, funded by MCIN/AEI/ 10.13039/501100011033, and by "ERDF: A way of making Europe", by the "European Union"; M. Mora-Jiménez was supported by the Generalitat Valenciana and the European Social Fund under grant number ACIF/2020/269; A. Falcó was supported by the MICIN grant number RTI2018-093521-B-C32 and Universidad CEU Cardenal Herrera under grant number INDI22/15.

Conflict of interest

The authors declare that they no have conflicts of interest.

References

1. A. Fawzi, M. Balog, A. Huang, T. Hubert, B. Romera-Paredes, M. Barekatain, et al., Discovering faster matrix multiplication algorithms with reinforcement learning, *Nature*, **610** (2022), 47–53. https://doi.org/10.1038/s41586-022-05172-4
2. G. Heidel, V. Khoromskaia, B. Khoromskij, V. Schulz, Tensor product method for fast solution of optimal control problems with fractional multidimensional Laplacian in constraints, *J. Comput. Phys.*, **424** (2021), 109865. https://doi.org/10.1016/j.jcp.2020.109865
3. C. Quesada, G. Xu, D. González, I. Alfaro, A. Leygue, M. Visonneau, et al., Un método de descomposición propia generalizada para operadores diferenciales de alto orden, *Rev. Int. Metod. Numer.*, **31** (2015), 188–197. https://doi.org/10.1016/j.rimni.2014.09.001
4. A. Nouy, Low-rank methods for high-dimensional approximation and model order reduction, *Soc. Ind. Appl. Math.*, 2017, 171–226.
5. A. Falcó, A. Nouy, Proper generalized decomposition for nonlinear convex problems in tensor Banach spaces, *Numer. Math.*, **121** (2012), 503–530. https://doi.org/10.1007/s00211-011-0437-5
6. I. Georgieva, C. Hofreither, Greedy low-rank approximation in Tucker format of solutions of tensor linear systems, *J. Comput. Appl. Math.*, **358** (2019), 206–220. https://doi.org/10.1016/j.cam.2019.03.002
7. A. Ammar, F. Chinesta, A. Falcó, On the convergence of a Greedy Rank-One Update algorithm for a class of linear systems, *Arch. Comput. Methods Eng.*, **17** (2010), 473–486. https://doi.org/10.1007/s11831-010-9048-z
8. J. A. Conejero, A. Falcó, M. Mora-Jiménez, Structure and approximation properties of laplacian-like matrices, *Results Math.*, **78** (2023), 184. https://doi.org/10.1007/s00025-023-01960-0
9. J. Swift, P. C. Hohenberg, Hydrodynamic fluctuations at the convective instability, *Phys. Rev. A*, **15** (1977), 319–328.
10. V. de Silva, L. H. Lim, Tensor rank and the ill-posedness of the best low-rank approximation problem, *SIAM J. Matrix Anal. Appl.*, **30** (2008), 1084–1127. https://doi.org/10.1137/06066518X

11. A. Falcó, L. Hilario, N. Montés, M. Mora, Numerical strategies for the galerkin–proper generalized decomposition method, *Math. Comput. Model.*, **57** (2013), 1694–1702. https://doi.org/10.1016/j.mcm.2011.11.012

12. W. Hackbusch, B. Khoromskij, S. Sauter, E. Tyrtyshnikov, Use of tensor formats in elliptic eigenvalue problems, *Numer. Lin. Algebra Appl.*, **19** (2012), 133–151. https://doi.org/10.1002/nla.793

13. MATLAB, version R2021b, The MathWorks Inc., Natick, Massachusetts, 2021.