Numerical Algorithms (2023) 92:377–406
https://doi.org/10.1007/s11075-022-01351-6

ORIGINAL PAPER

A Lanczos-type procedure for tensors

Stefano Cipolla[1] · **Stefano Pozza**[2] · **Michela Redivo-Zaglia**[3] · **Niel Van Buggenhout**[2]

Received: 12 May 2022 / Accepted: 7 June 2022 / Published online: 31 August 2022

Abstract
The solution of linear non-autonomous ordinary differential equation systems (also known as the time-ordered exponential) is a computationally challenging problem arising in a variety of applications. In this work, we present and study a new framework for the computation of bilinear forms involving the time-ordered exponential. Such a framework is based on an extension of the non-Hermitian Lanczos algorithm to 4-mode tensors. Detailed results concerning its theoretical properties are presented. Moreover, computational results performed on real-world problems confirm the effectiveness of our approach.

Keywords Non-Hermitian Lanczos algorithm · $\star$-Lanczos algorithm · Lanczos-type procedures for tensors · Time-ordered exponential

✉ Stefano Cipolla
scipolla@ed.ac.uk

✉ Stefano Pozza
pozza@karlin.mff.cuni.cz

✉ Michela Redivo-Zaglia
michela.redivozaglia@unipd.it

✉ Niel Van Buggenhout
buggenhout@karlin.mff.cuni.cz

[1] School of Mathematics, The University of Edinburgh, Peter Guthrie Tait Road, Edinburgh, EH9 3FD, UK

[2] Faculty of Mathematics and Physics, Charles University, Sokolovská 83, Prague, 186 75, Czech Republic

[3] Department of Mathematics "Tullio Levi-Civita", University of Padua, Via Trieste 63, Padua, 35121, Italy

1 Introduction

In this paper, we present an extension of the non-Hermitian Lanczos algorithm (see, e.g., [25]) where the inputs are a 4-mode tensor $\mathcal{A} \in \mathbb{C}^{N\times N\times M\times M}$ and vectors $\mathbf{w}, \mathbf{v} \in \mathbb{C}^N$ so that $\mathbf{w}^H\mathbf{v} \neq 0$. We aim to use the introduced algorithm to approximate the bilinear form $\mathbf{w}^H\mathbf{U}(t)\mathbf{v}$, where $\mathbf{U}(t) \in \mathbb{C}^{N\times N}$ is the so-called time-ordered exponential, i.e., the solution of the ordinary differential equation

$$\frac{d}{dt}\mathbf{U}(t) = \mathbf{A}(t)\mathbf{U}(t), \quad \mathbf{U}(a) = I_N, \quad t \in I = [a, b], \tag{1}$$

where $I_N \in \mathbb{R}^{N\times N}$ is the identity matrix and $\mathbf{A}(t) \in \mathbb{C}^{N\times N}$ is a smooth matrix-valued function defined on the real interval I. Equation (1) can emerge in a variety of applications. For example, its solution is crucial in quantum physics where the matrix $\mathbf{A}(t)$ corresponds to the Hamiltonian operator. Situations where $\mathbf{U}(t)$ has no accessible expression are frequent in literature, see, e.g., [1, 7, 38, 49]. For instance, in Nuclear Magnetic Resonance (NMR) experiments, the associated bilinear form $\mathbf{w}^H\mathbf{U}(t)\mathbf{v}$ represents the measurement of changes in an applied magnetic field caused by nuclear spins that are excited with electromagnetic waves, i.e., spectroscopy [29, 39]. Other applications are found in control theory, filter design, and model reduction problems [4, 5, 14, 37, 47]. In the mentioned applications, the matrix $\mathbf{A}(t)$ is often large-to-huge and sparse. The introduced algorithm is motivated and theoretically supported by a new expression for the bilinear form. This expression is given by combining the two symbolic methods known as Path-sum and $\star$-Lanczos algorithm [21–24]. Given the matrix-valued function $\mathbf{A}(t)$ and the vectors $\mathbf{w}, \mathbf{v}$, the two symbolic methods produce an expression for the bilinear form $\mathbf{w}^H\mathbf{U}(t)\mathbf{v}$ composed of a finite and treatable number of integrals and scalar integral equations. To our knowledge, no other symbolic method can express the bilinear form with a treatable finite number of integral subproblems. Two commonly used alternative expressions are given by the Magnus series, i.e., an infinite series of nested integrals (e.g., [40]), and by the Floquet theory, where the solution of an infinite system of coupled linear differential equations is required (e.g., [7]).

The integrals and the integral equations generated by the $\star$-Lanczos and Path-sum methods do not always have an easily accessible solution. As a consequence, a numerical approach is needed. A possible strategy for the numerical approximation of the mentioned integrals and the integral equations is outlined in [23] and it is based on the discretization of the interval I into $M-1$ equispaced subintervals. The algebraic objects resulting from the discretization strategy are the 4-mode tensor $\mathcal{A}$ (corresponding to $\mathbf{A}(t)$) and the 3-mode tensors V, W (corresponding to $\mathbf{v}$, $\mathbf{w}$).

The outputs obtained by combining the $\star$-Lanczos algorithm with the mentioned discretization strategy are mathematically equivalent to the outputs of the tensor Lanczos algorithm presented here with, as inputs, $\mathcal{A}$, $\mathbf{v}$, $\mathbf{w}$. The main goal of this paper is to show that, in fact, the tensor Lanczos algorithm can converge to the outcome of the $\star$-Lanczos method within an accuracy of the same order as the discretization strategy. Moreover, the reported numerical experiments will show that the approximation of $\mathbf{w}^H\mathbf{U}(t)\mathbf{v}$ obtained by combining the tensor Lanczos with the discretized Path-sum approach also converges to the solution within the order

of the discretization. Naturally, many numerical methods for the solution of non-autonomous ODEs can be found in literature, see, for instance, [2, 6, 7, 10, 13, 15, 30, 34–36]. For large matrices, these numerical methods are known to be highly demanding both in terms of computational cost and storage. This motivates the research of novel approaches suitable for large-scale problems. In order to be competitive with the most advanced techniques, tensor Lanczos needs to be used in combination with more accurate discretization schemes. Development of suitable, faster converging discretization schemes is an ongoing research and out of the scope of this work. At the same time, it is important to note that the algorithm here proposed is part of a wider class of tensor extensions of Krylov subspace methods that recently appeared in the literature, see, e.g., [18, 26, 33, 46].

The Lanczos-type process we introduce can also be equivalently written as a block Lanczos method since the 4-mode tensor $\mathcal{A}$ can be seen as a block matrix; information about block Krylov subspace methods can be found, e.g., in [20]. Despite this fact, we prefer to interpret such a block structure in a *tensorial* fashion. Indeed, the tensorial approach has a direct translation in terms of a discretized $\star$-Lanczos algorithm. Moreover, as we will experimentally show, this interpretation is motivated by observing that several tensors from real-world examples related with (1) are characterized by a low parametric approximation known as *Tensor Train* decomposition (TT) [41, 42]. Such a low parametric approximation allows to efficiently manipulate and store the tensors. This paves the path for further improvements of our proposal, where the TT structure is fully exploited in the Lanczos-type procedure. Examples of tensor Krylov subspace methods combined with the TT decomposition can be found in [18, 48], further motivating our *tensor-based* point of view.

More in detail, this work is organized as follows. Preliminaries and definitions of tensor operations are given in Section 2. In Section 3 we discuss how to construct the non-Hermitian Lanczos procedure for tensors and we prove several crucial properties. In Section 4 we discuss the breakdown issue which typically arises when working with non-Hermitian Lanczos approaches. Numerical experiments are presented in Section 5 where we also give several examples exposing the low-rank TT structure of the considered tensors $\mathcal{A}$. Section 6 concludes the paper and Appendix contains several proofs.

2 Preliminaries

In this work, we use a notation borrowed from Matlab®. Fixing $i_1 \in \{1, \dots, N_1\}$ and $i_2 \in \{1, \dots, N_2\}$, if $\mathcal{A} \in \mathbb{C}^{N_1 \times N_2 \times M \times M}$, then $\mathcal{A}_{i_1,i_2,:,:}$ stands for the matrix

$$\mathcal{A}_{i_1,i_2,:,:} := \left[\mathcal{A}_{i_1,i_2,j_1,j_2}\right]_{j_1,j_2=1}^{M}.$$

This notation similarly applies to 3-mode tensors, matrices, and vectors. Table 1 summarizes the notation used in the paper.

In the following, we define several tensorial operations, which can be seen as generalizations of the usual products involving matrices and vectors. We summarize them in Table 2.

Table 1 Summary of notation

Symbol	Description
$\mathbf{A}(t), \mathbf{U}(t)$	Matrix-valued functions
$\boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}}$...	4-mode tensors
A, B...	3-mode tensors
$\boldsymbol{\alpha}, \boldsymbol{\beta}$...	$M \times M$ Matrices
I_M	$M \times M$ identity matrix
$\mathbf{a}, \mathbf{b}$	Vectors
$\boldsymbol{\mathcal{A}}^{k_*}$	k-th $*$-power of $\boldsymbol{\mathcal{A}}$
$\boldsymbol{\mathcal{I}}_*$	$*$-identity

In the following definitions we consider the tensors $\boldsymbol{\mathcal{A}} \in \mathbb{C}^{N_1 \times N_2 \times M \times M}$, $\boldsymbol{\mathcal{B}} \in \mathbb{C}^{N_2 \times N_3 \times M \times M}$, $A \in \mathbb{C}^{N_2 \times M \times M}$, $B \in \mathbb{C}^{N_1 \times M \times M}$, $\boldsymbol{\alpha} \in \mathbb{C}^{M \times M}$. Moreover, the indices $i_1 \in \{1, \ldots, N_1\}$ $i_2 \in \{1, \ldots, N_2\}$ are fixed.

Definition 1 ($*$-Tensor product) The product $(\boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{B}}) \in \mathbb{C}^{N_1 \times N_3 \times M \times M}$ is defined as

$$(\boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{B}})_{i_1,i_2,:,:} := \sum_{k=1}^{N_2} \boldsymbol{\mathcal{A}}_{i_1,k,:,:} \, \boldsymbol{\mathcal{B}}_{k,i_2,:,:}.$$

Definition 2 (Tensor-Hypervector product) The product $(\boldsymbol{\mathcal{A}} * A) \in \mathbb{C}^{N_1 \times M \times M}$ is defined as

$$(\boldsymbol{\mathcal{A}} * A)_{i_1,:,:} := \sum_{k=1}^{N_2} \boldsymbol{\mathcal{A}}_{i_1,k,:,:} \, A_{k,:,:}.$$

We also need to define the action of a 3-mode tensor from the left. Every tensor with three modes that acts, will act, or is the outcome of a $*$-product from the left, will be denoted with a "D" (dual) as apex, and, in the remainder of this work, we will use $B^D_{k,:,:}$ to denote $(B^D)_{k,:,:}$. We define, $(B^D * \boldsymbol{\mathcal{A}})^D \in \mathbb{C}^{N_2 \times M \times M}$ as

$$(B^D * \boldsymbol{\mathcal{A}})^D_{i_2,:,:} := \sum_{k=1}^{N_1} B^D_{k,:,:} \, \boldsymbol{\mathcal{A}}_{k,i_2,:,:}.$$

Table 2 Each of the considered *products* involves two tensors with n and m modes and it gives as an outcome a tensor with k-modes

Symbol	Name	n, m	k	Generalizing
$*$	$*$-Tensor product	4, 4	4	Matrix-matrix product
$*$	Tensor-Hypervector product	4, 3	3	Matrix-vector product
$*$	Hypervector inner-product	3, 3	2	Inner-product
$\times$	Tensor-matrix product	4, 2	4	Matrix-scalar product
$\times$	Hypervector-matrix product	3, 2	3	Vector-scalar product
$\otimes$	Vector-to-Hypervector product	1, 2	3	Kronecker product

Note that the following 4-mode tensor is the identity for $*$-products introduced above

$$\mathbb{C}^{N_1\times N_1\times M\times M} \ni (\mathcal{I}_*)_{i_1,i_2,:,:} := \begin{cases} I_M, & \text{if } i_1 = i_2 \\ 0_M, & \text{otherwise} \end{cases}.$$

Definition 3 (Hypervector inner-product) The product $(B^D * A) \in \mathbb{C}^{M\times M}$ is defined as

$$(B^D * A)_{:,:} := \sum_{k=1}^{N_1} B^D_{k,:,:}\, A_{k,:,:}.$$

Definition 4 (Tensor-matrix product) The products $(\boldsymbol{\mathcal{A}} \times \boldsymbol{\alpha})$, $(\boldsymbol{\alpha} \times \boldsymbol{\mathcal{A}}) \in \mathbb{C}^{N_1\times N_2\times M\times M}$ are defined as

$$(\boldsymbol{\mathcal{A}} \times \boldsymbol{\alpha})_{i_1,i_2,:,:} := \boldsymbol{\mathcal{A}}_{i_1,i_2,:,:}\, \boldsymbol{\alpha} \quad \text{and} \quad (\boldsymbol{\alpha} \times \boldsymbol{\mathcal{A}})_{i_1,i_2,:,:} := \boldsymbol{\alpha}\, \boldsymbol{\mathcal{A}}_{i_1,i_2,:,:}.$$

Definition 5 (Hypervector-matrix product) The products $(A \times \boldsymbol{\alpha})$, $(\boldsymbol{\alpha} \times A) \in \mathbb{C}^{N_2\times M\times M}$ are defined as

$$(A \times \boldsymbol{\alpha})_{i_1,:,:} := A_{i_1,:,:}\boldsymbol{\alpha} \quad \text{and} \quad (\boldsymbol{\alpha} \times A)_{i_1,:,:} := \boldsymbol{\alpha} A_{i_1,:,:}.$$

Definition 6 (Vector-to-Hypervector) Given $\mathbf{a} \in \mathbb{C}^N$ we define the product $A = \mathbf{a} \otimes I_M \in \mathbb{C}^{N\times M\times M}$ as

$$A_{i_1,:,:} = \mathbf{a}_{i_1} I_M \quad i_1 \in \{1,\ldots,N\}.$$

Note that rearranging A as a block matrix, we get the usual Kronecker product. All the products are clearly distributive with respect to the usual addition. On the other hand, the associativity of some of the products is less obvious. Therefore, we state it in the following Lemma 1, postponing its proof to Appendix.

Lemma 1 *The following states that the tensor-tensor and tensor-hypervector $*$-products are associative.*

- *Given $\boldsymbol{\mathcal{A}} \in \mathbb{C}^{N_1\times N_1\times M\times M}$, $A \in \mathbb{C}^{N_1\times M\times M}$ we have*

$$(\boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{A}}) * A = \boldsymbol{\mathcal{A}} * (\boldsymbol{\mathcal{A}} * A).$$

- *Given $B^D \in \mathbb{C}^{N_1\times M\times M}$, $A \in \mathbb{C}^{N_2\times M\times M}$, $\boldsymbol{\mathcal{A}} \in \mathbb{C}^{N_1\times N_2\times M\times M}$, then*

$$(B^D * \boldsymbol{\mathcal{A}})^D * A = B^D * (\boldsymbol{\mathcal{A}} * A).$$

- *Given $\boldsymbol{\mathcal{A}} \in \mathbb{C}^{N_1\times N_2\times M\times M}$, $\boldsymbol{\mathcal{B}} \in \mathbb{C}^{N_2\times N_3\times M\times M}$, $\boldsymbol{\mathcal{C}} \in \mathbb{C}^{N_3\times N_1\times M\times M}$, then*

$$(\boldsymbol{\mathcal{C}} * \boldsymbol{\mathcal{A}}) * \boldsymbol{\mathcal{B}} = \boldsymbol{\mathcal{C}} * (\boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{B}}).$$

Having introduced the required products and their basic properties, we are ready to derive the tensor non-Hermitian Lanczos algorithm.

3 The Lanczos-type process

Using the operations given in Table 2, we propose a sensible generalization of Krylov subspaces where, instead of the usual matrix-vector product, the tensor-hypervector product is used to generate the subspaces. Section 3.1 describes these tensor Krylov-type subspaces in detail and defines biorthogonal bases for them. A Lanczos-type algorithm which generates these biorthogonal bases is proposed in Section 3.2. In Section 3.3 two important properties of the classical Lanczos algorithm are generalized, namely, the tensor representation of the three-term recurrence relations for the biorthogonal bases and the matching moment property. The computational cost and storage requirements of the algorithm are discussed in Section 3.4.

3.1 Krylov-type tensor subspaces

Consider the tensor $\mathcal{A} \in \mathbb{C}^{N\times N\times M\times M}$. We define the *polynomials* of degree ℓ of $\mathcal{A}$ as

$$p(\mathcal{A}) := \sum_{k=0}^{\ell} \mathcal{A}^{k_*} \times \boldsymbol{\alpha}_k,$$
$$p^D(\mathcal{A}) := \sum_{k=0}^{\ell} \boldsymbol{\alpha}_k^H \times \mathcal{A}^{k_*},$$

where $\mathcal{A}^{k_*}$ stands for k $*$-multiplications of $\mathcal{A}$ by itself, and $\boldsymbol{\alpha}_k^H$ is the conjugate transpose of $\boldsymbol{\alpha}$. Given the tensors $A \in \mathbb{C}^{N\times M\times M}$, $B \in \mathbb{C}^{N\times M\times M}$ we can define the Krylov-type subspaces

$$\mathcal{K}_n(\mathcal{A}, A) := \{p(\mathcal{A}) * A \text{ s.t. } deg(p) \le n-1\},$$
$$\mathcal{K}_n^D(B^D, \mathcal{A}) := \{B^D * p^D(\mathcal{A}) \text{ s.t. } deg(p^D) \le n-1\}.$$

Every element in $\mathcal{K}_n(\mathcal{A}, A)$ is a tensor in $\mathbb{C}^{N\times M\times M}$ and can be written as

$$p(\mathcal{A}) * A = \sum_{k=0}^{n-1} (\mathcal{A}^{k_*} \times \boldsymbol{\alpha}_k) * A.$$

From now on we will assume that A is of the form $A = \mathbf{a} \otimes I_M$ for some $\mathbf{a} \in \mathbb{C}^N$. In this case, the matrices $\boldsymbol{\alpha}_k$ commute with A giving

$$p(\mathcal{A}) * A = \sum_{k=0}^{n-1} (\mathcal{A}^{k_*} * A) \times \boldsymbol{\alpha}_k.$$

An analogous result holds for $B^D * p^D(\mathcal{A})$ when $B^D = \overline{\mathbf{b}} \otimes I_M$ for any $\mathbf{b} \in \mathbb{C}^N$, $\overline{\mathbf{b}}$ is the conjugated vector.

Driven by the analogy with the matrix case, our aim is to build two "biorthonormal bases" for the Krylov-type subspaces $\mathcal{K}_n(\mathcal{A}, A)$ and $\mathcal{K}_n^D(B^D, \mathcal{A})$. The following Definition 7 allows to characterize spaces spanned by 3-mode tensors.

Definition 7 Given $V_1, \ldots, V_n \in \mathbb{C}^{N\times M\times M}$, $W_1^D, \ldots, W_n^D \in \mathbb{C}^{N\times M\times M}$, we define the subspaces

$$\langle V_1, \ldots, V_n\rangle := \left\{ V = \sum_{k=1}^{n} V_k \times \boldsymbol{\eta}_k, \text{ for } \boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_n \in \mathbb{C}^{M\times M} \right\};$$

$$\langle W_1^D, \ldots, W_n^D\rangle := \left\{ W^D = \sum_{k=1}^{n} \boldsymbol{\eta}_k \times W_k^D, \text{ for } \boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_n \in \mathbb{C}^{M\times M} \right\}.$$

We say that $V_1, \ldots, V_n$ is a *basis* for the subspace $\langle V_1, \ldots, V_n\rangle$ and $W_1^D, \ldots, W_n^D$ is a *basis* for the subspace $\langle W_1^D, \ldots, W_n^D\rangle$.

Biorthonormal bases for Krylov-type subspaces are represented by the tensors $\boldsymbol{\mathcal{V}}_n \in \mathbb{C}^{N\times n\times M\times M}$ and $\boldsymbol{\mathcal{W}}_n \in \mathbb{C}^{n\times N\times M\times M}$ satisfying

$$\boldsymbol{\mathcal{W}}_n * \boldsymbol{\mathcal{V}}_n = \boldsymbol{\mathcal{I}}_* \in \mathbb{R}^{n\times n\times M\times M}, \tag{2}$$

with the hypervectors $V_k := (\boldsymbol{\mathcal{V}}_n)_{:,k,:,:}$ and $W_k^D := (\boldsymbol{\mathcal{W}}_n)_{k,:,:,:}$, for $k = 1, \ldots, n$, forming, respectively, bases for $\mathcal{K}_n(\boldsymbol{\mathcal{A}}, A)$ and $\mathcal{K}_n^D(B^D, \boldsymbol{\mathcal{A}})$, i.e.,

$$\langle V_1, \ldots, V_n\rangle = \mathcal{K}_n(\boldsymbol{\mathcal{A}}, A), \quad \langle W_1^D, \ldots, W_n^D\rangle = \mathcal{K}_n^D(B^D, \boldsymbol{\mathcal{A}}).$$

In the following section we derive such bases by constructing the tensor non-Hermitian Lanczos Algorithm.

3.2 The tensor Lanczos process

Given the inputs $\boldsymbol{\mathcal{A}} \in \mathbb{C}^{N\times N\times M\times M}$ and $\mathbf{v}, \mathbf{w} \in \mathbb{C}^N$, Algorithm 1 constructs, when no breakdown occurs, the bases $\boldsymbol{\mathcal{W}}_n$ and $\boldsymbol{\mathcal{V}}_n$, for $\mathcal{K}_n(\boldsymbol{\mathcal{A}}, A)$ and $\mathcal{K}_n^D(B^D, \boldsymbol{\mathcal{A}})$, respectively, which satisfy the $*$-*biorthogonality conditions* (2).

Algorithm 1 non-Hermitian Lanczos for tensors.

Input: $\boldsymbol{\mathcal{A}} \in \mathbb{C}^{N\times N\times M\times M}$, $\mathbf{v}, \mathbf{w} \in \mathbb{C}^N$ such that $\mathbf{w}^H\mathbf{v} = 1$.
Output: $V_1, \ldots, V_n$, $W_1^D, \ldots, W_n^D \in \mathbb{C}^{N\times M\times M}$ spanning respectively $\mathcal{K}_n(\boldsymbol{\mathcal{A}}, V)$, $\mathcal{K}_n(W^D, \boldsymbol{\mathcal{A}})$.

Initialize: $V_0 = W_0^D = 0, \boldsymbol{\beta}_1 = 0$,
$V := \mathbf{v} \otimes I_M$, $W^D := \overline{\mathbf{w}} \otimes I_M$, $V_1 := V$, $W_1^D := W^D$ **for** $k = 1, \ldots n$ **do**
  $\boldsymbol{\alpha}_k = W_k^D * \boldsymbol{\mathcal{A}} * V_k$
  $\widehat{W}_{k+1}^D = W_k^D * \boldsymbol{\mathcal{A}} - \boldsymbol{\alpha}_k \times W_k^D - \boldsymbol{\beta}_k \times W_{k-1}^D$
  $\widehat{V}_{k+1} = \boldsymbol{\mathcal{A}} * V_k - V_k \times \boldsymbol{\alpha}_k - V_{k-1} \times \boldsymbol{\gamma}_k$
  Set a non singular matrix $\boldsymbol{\gamma}_{k+1}$
  $\boldsymbol{\beta}_{k+1} = (\boldsymbol{\gamma}_{k+1})^{-1}(\widehat{W}_{k+1}^D * \widehat{V}_{k+1})$
  if $\boldsymbol{\beta}_{k+1}$ *is singular* **then**
    `Stop`
  end
  $V_{k+1} = \widehat{V}_{k+1} \times \boldsymbol{\beta}_{k+1}^{-1}$ $W_{k+1}^D = (\boldsymbol{\gamma}_{k+1})^{-1} \times \widehat{W}_{k+1}^D$
end

Details on how the algorithm constructs these bases using three-term recurrences are described below.

- By definition, the first hypervectors of the bases are W_1^D, V_1 satisfying $W_1^D * V_1 = I_M$;
- Consider the vector $\widehat{W}_2^D \in \mathcal{K}_2^D(W^D, \boldsymbol{\mathcal{A}})$ given by

$$\widehat{W}_2^D := W_1^D * \boldsymbol{\mathcal{A}} - \boldsymbol{\alpha}_1 \times W_1^D.$$

  Imposing that $\widehat{W}_2^D$ satisfies the $*$-biorthogonal condition $\widehat{W}_2^D * V_1 = 0$, we have $\boldsymbol{\alpha}_1 = W_1^D * \boldsymbol{\mathcal{A}} * V_1$.
- Analogously, define the vector $\widehat{V}_2 \in \mathcal{K}_2(\boldsymbol{\mathcal{A}}, V)$ given by

$$\widehat{V}_2 := \boldsymbol{\mathcal{A}} * V_1 - V_1 \times \boldsymbol{\alpha}_1.$$

  Imposing the $*$-biorthogonality condition, we find the $*$-biorthogonal vectors

$$V_2 := \widehat{V}_2 \times \boldsymbol{\beta}_1^{-1} \text{ where } \boldsymbol{\beta}_1 = \widehat{W}_2^D * \widehat{V}_2 = \widehat{W}_2^D * V_1 \text{ and } W_2 = \widehat{W}_2.$$

- Clearly $\mathcal{K}_2(\boldsymbol{\mathcal{A}}, V) = \langle V_1, V_2 \rangle$ and $\mathcal{K}_2^D(W^D, \boldsymbol{\mathcal{A}}) = \langle W_1^D, W_2^D \rangle$.
- Now, assume the $*$-biorthonormal bases $V_1, \ldots, V_k$ and $W_1^D, \ldots, W_k^D$ are available. Consider the hypervector

$$\widehat{W}_{k+1}^D := W_k^D * \boldsymbol{\mathcal{A}} - \sum_{i=1}^{k} \boldsymbol{\eta}_i \times W_i^D.$$

  The matrices $\boldsymbol{\eta}_i$ are determined by the conditions $\widehat{W}_{k+1}^D * V_i = 0$, for $i = 1, \ldots, k$, which give

$$\boldsymbol{\eta}_i = W_k^D * \boldsymbol{\mathcal{A}} * V_i, \quad \text{for } i = 1, \ldots, k.$$

  In particular, since $\boldsymbol{\mathcal{A}} * V_i \in \mathcal{K}_{i+1}(\boldsymbol{\mathcal{A}}, V)$, we get $\boldsymbol{\eta}_i = 0$ for $i = 1, \ldots, k-2$. An analogous argument is valid for $\widehat{V}_{k+1}$. This leads to the following three-term recurrences

$$W_{k+1}^D = W_k^D * \boldsymbol{\mathcal{A}} - \boldsymbol{\alpha}_k \times W_k^D - \boldsymbol{\beta}_k \times W_{k-1}^D, \tag{3a}$$

$$V_{k+1} \times \boldsymbol{\beta}_{k+1} = \boldsymbol{\mathcal{A}} * V_k - V_k \times \boldsymbol{\alpha}_k - V_{k-1}, \tag{3b}$$

  with coefficients

$$\boldsymbol{\alpha}_k = W_k^D * \boldsymbol{\mathcal{A}} * V_k, \quad \boldsymbol{\beta}_{k+1} = W_{k+1}^D * \boldsymbol{\mathcal{A}} * V_k. \tag{4}$$

- To prove that $\langle V_1, \ldots, V_n \rangle = \mathcal{K}_n(\boldsymbol{\mathcal{A}}, V)$ and $\langle W_1^D, \ldots, W_n^D \rangle = \mathcal{K}_n(W^D, \boldsymbol{\mathcal{A}})$, it is enough to use induction and the fact that $V_k \in \mathcal{K}_k(\boldsymbol{\mathcal{A}}, V)$ and $W_k^D \in \mathcal{K}_k^D(W^D, \boldsymbol{\mathcal{A}})$ for all $k = 1, \ldots, n$.

Let us finally observe that, should $\boldsymbol{\beta}_{k+1}$ not be invertible, we would get a *breakdown* in the algorithm.

Different rescaling strategies are possible by setting an invertible coefficient $\boldsymbol{\gamma}_{k+1}$ and noticing that

$$(\boldsymbol{\gamma}_{k+1})^{-1} \times W_{k+1}^D * V_{k+1} \times \boldsymbol{\gamma}_{k+1} = I_M.$$

This last observation completes the construction of Algorithm 1.

3.3 Main properties of the tensor Lanczos algorithm

It is important to note that the coefficients in the three-term recurrences (3a–3b) can be represented by a sparse 4-mode tensor. To this aim, let us consider $\mathcal{T}_n \in \mathbb{C}^{n\times n\times M\times M}$ as the tensor defined as

$$(\mathcal{T}_n)_{i_1,i_2,:,:} := \begin{cases} \boldsymbol{\alpha}_{i_1}, & \text{if } i_1 = i_2 \quad \text{and } 1 \le i_1 \le n \\ \boldsymbol{\gamma}_{i_1}, & \text{if } i_2 = i_1 + 1 \quad \text{and } 1 \le i_1 \le n-1 \\ \boldsymbol{\beta}_{i_1}, & \text{if } i_2 = i_1 - 1 \quad \text{and } 2 \le i_1 \le n \\ \mathbf{0}, & \text{otherwise} \end{cases}. \tag{5}$$

where $\boldsymbol{\alpha}_{i_1}, \boldsymbol{\beta}_{i_1}, \boldsymbol{\gamma}_{i_1}$ are the matrices in Algorithm 1. The tensor $\mathcal{T}_n$ is a generalization of the so-called *(complex) Jacobi matrix* associated with the non-Hermitian Lanczos algorithm; see, e.g., [45] and references therein. By using $\mathcal{T}_n$, Theorem 1 provides a compact representation of the three-term recurrences constructing the biorthogonal bases.

Theorem 1 *The three-term recurrences* (3a–3b) *can be written in the compact form*

$$\mathcal{A} * \mathcal{V}_n = \mathcal{V}_n * \mathcal{T}_n + \widetilde{\mathcal{V}}_n \tag{6a}$$

$$\mathcal{W}_n * \mathcal{A} = \mathcal{T}_n * \mathcal{W}_n + \widetilde{\mathcal{W}}_n \tag{6b}$$

where $\widetilde{\mathcal{V}}_n \in \mathbb{C}^{N\times n\times M\times M}$ *is*

$$(\widetilde{\mathcal{V}}_n)_{:,k,:,:} := \begin{cases} V_{n+1} \times \boldsymbol{\beta}_{n+1}, & \text{if } k = n \\ \mathbf{0}, & \text{otherwise} \end{cases},$$

and $\widetilde{\mathcal{W}}_n \in \mathbb{C}^{n\times N\times M\times M}$ *is*

$$(\widetilde{\mathcal{W}}_n)_{k,:,:,:} := \begin{cases} \boldsymbol{\gamma}_{n+1} \times W_{n+1}^D, & \text{if } k = n \\ \mathbf{0}, & \text{otherwise} \end{cases}.$$

Proof By direct inspection. We have, for all $i_1 \in \{1,\dots,N\}$, $i_2 \in \{1,\dots,n-1\}$

$$(\mathcal{A} * \mathcal{V}_n)_{i_1,i_2,:,:} = \sum_{k=1}^{N} \mathcal{A}_{i_1,k,:,:}(\mathcal{V}_n)_{k,i_2,:,:} = \sum_{k=1}^{N} \mathcal{A}_{i_1,k,:,:}(V_{i_2})_{k,:,:} = (\mathcal{A} * V_{i_2})_{i_1,:,:} \tag{7}$$

and

$$\begin{aligned} (\mathcal{V}_n * \mathcal{T}_n + \widetilde{\mathcal{V}}_n)_{i_1,i_2,:,:} &= \sum_{k=1}^{n} (\mathcal{V}_n)_{i_1,k,:,:}(\mathcal{T}_n)_{k,i_2,:,:} \\ &= (V_{i_2})_{i_1}\boldsymbol{\alpha}_{i_2} + (V_{i_2+1})_{i_1}\boldsymbol{\beta}_{i_2+1} + (V_{i_2-1})_{i_1} \\ &= (V_{i_2} \times \boldsymbol{\alpha}_{i_2} + V_{i_2+1} \times \boldsymbol{\beta}_{i_2+1} + V_{i_2-1})_{i_1}. \end{aligned} \tag{8}$$

The equality between (7) and (8) follows using (3b) and proves (6a). The remaining part of the theorem can be proved analogously. □

If we $*$-multiply (6a) by $\mathcal{W}_n$ from the left we obtain the expression

$$\mathcal{T}_n = \mathcal{W}_n * \mathcal{A} * \mathcal{V}_n.$$

The tensor $\mathcal{T}_n$ satisfies a generalization of the *matching moment property* which is stated in Theorem 2.

Theorem 2 (Matching Moment Property) *Let $\mathcal{A}, V, W$ and $\mathcal{T}_n$ be as described above, then*

$$W^D * (\mathcal{A}^{k_*}) * V = E_1^D * (\mathcal{T}_n)^{k_*} * E_1, \quad \text{for} \quad k = 0, \dots, 2n-1,$$

where $E_1 = \mathbf{e}_1 \otimes I_M$ and $\mathbf{e}_1$ is the first vector of the Euclidean base of $\mathbb{C}^{N\times N}$.

The proof of Theorem 2 can be found in Appendix.

3.4 Numerical properties

The tensor $\mathcal{A}$ is obtained by discretizing $\mathbf{A}(t)$ and stores in $\mathcal{A}_{k,l,:,:}$ the coefficients representing the (k,l)-th element of $\mathbf{A}(t)$. Different methods of discretization are possible. In this paper, following [23], we discretize the interval I obtaining the mesh

$$\tau_i = h(i-1) + a, \quad i = 1, \dots, M, \quad h = \frac{b-a}{M-1}. \tag{9}$$

For this mesh the discretization of $\mathbf{A}(t) = \left[\mathbf{A}_{k,\ell}(t)\right]_{k,\ell}^N$ is the tensor

$$\mathcal{A}_{k,\ell,:,:} := \boldsymbol{\nu}_{k,\ell}, \quad k,\ell = 1, \dots, N, \tag{10}$$

where the matrices $\boldsymbol{\nu}_{k,\ell} \in \mathbb{C}^{M\times M}$ are lower triangular matrices defined as

$$\left(\boldsymbol{\nu}_{k,\ell}\right)_{i,j} = \begin{cases} \mathbf{A}_{k,\ell}(\tau_i)h, & i \geq j \\ 0 & i < j \end{cases}.$$

This discretization scheme has an accuracy of order $\mathcal{O}(h) = \mathcal{O}(1/M)$ and, indeed, in Section 5, we show that when this discretization scheme is used, the approximation of the bilinear form of interest also has an accuracy of $\mathcal{O}(1/M)$.

The computational cost of Algorithm 1 depends on the chosen number of discretization points M and on the number of iterations n. In this algorithm the dominant cost is the multiplication of a 4-mode tensor with a 3-mode tensor, i.e., $\mathcal{A} * V_k$ and $W_k^D * \mathcal{A}$. The worst case complexity of one such product is $\mathcal{O}(M^3N^2)$, for a total cost of $\mathcal{O}(2nM^3N^2)$. However, since $\mathbf{A}(t)$ is sparse in all practical applications, the computational cost can be much lower. For example, if there are $N_{\text{nz}} < N$ nonzeros in each column of $\mathbf{A}(t)$, then the cost reduces to $\mathcal{O}(2nM^3NN_{\text{nz}})$. It is important to note, moreover, that the term M^3 arises from the matrix-matrix multiplication between V_k and the blocks in $\mathcal{A}$. Since these blocks arise from a discretization strategy, it is likely that they will exhibit a particular structure that can be exploited for efficient computations. E.g., in the discretization used in this work, these blocks are lower triangular matrices for which the matrix-matrix multiplication has a cost of $\frac{M^3}{2}$.

Finally, the storage cost of Algorithm 1 is three basisvectors V_i, three basisvectors W_i^D and $3n-1$ nonzero elements of $\mathcal{T}_n$, for a total of $\mathcal{O}(6M^2N + 3M^2n)$. Only three basisvectors must be kept in memory thanks to the underlying three-term recurrence relation.

Let us conclude this section observing that, as highlighted from the previous discussion, both, the computational cost and storage requirement depend strongly on the number M of discretization points used. For the discretization scheme described above, we expect that a large number of discretization points is required since its accuracy is $\mathcal{O}(1/M)$. This justifies the search for more accurate discretization schemes, for example Legendre polynomial approximation. However, other discretization schemes will not be discussed here since they are subject of future research and since the discretization scheme introduced above suffices to illustrate the potential of Algorithm 1.

4 Breakdowns

If the matrix $\boldsymbol{\beta}_{k+1}$ is singular, then line 11 in Algorithm 1 cannot be performed and the algorithm breaks down. This breakdown issue is inherited from the (usual) non-Hermitian Lanczos algorithm; see, e.g., [19, 27, 28, 43, 52]. There are two different kinds of breakdowns. The first one, the so-called *lucky breakdown*, occurs when one of the Krylov-type subspaces $\mathcal{K}_k(\boldsymbol{\mathcal{A}}, A)$ or $\mathcal{K}_k^D(B^D, \boldsymbol{\mathcal{A}})$ becomes invariant under $\star$-multiplication with $\mathcal{A}$ from the left or right, respectively. Suppose that $\mathcal{K}_k(\boldsymbol{\mathcal{A}}, A)$ is an invariant subspace, this will result, in exact arithmetic, in $\widehat{V}_{k+1} = \mathbf{0}$ in Line 5 of Algorithm 1. In finite precision $\widehat{V}_{k+1} \in \mathbb{C}^{N\times M\times M}$ will never be exactly zero. Therefore, the Frobenius norm

$$\|\widehat{V}_{k+1}\|_F := \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{\ell=1}^{M}\left|\left(\widehat{V}_{k+1}\right)_{i,j,\ell}\right|^2$$

is used to define the following criterion to detect a lucky breakdown:

$$\frac{\|\widehat{V}_{k+1}\|_F}{\|V_k\|_F} < \epsilon,$$

with $\epsilon << 1$, a user-defined threshold close to machine precision. The same applies to the case $\widehat{W}_{k+1}^D = \mathbf{0}$. The second kind of breakdown occurs when both $\widehat{V}_{k+1} \neq \mathbf{0}$ and $\widehat{W}_{k+1}^D \neq \mathbf{0}$, but $\boldsymbol{\beta}_{k+1} \in \mathbb{C}^{M\times M}$ is still singular, then the algorithm breaks down. This case is known as a *serious breakdown*. In numerical computation, the condition number of $\boldsymbol{\beta}_{k+1}$ is monitored to decide if Line 11 can be computed sufficiently accurate. A user-defined threshold $\epsilon_s >> 1$ specifies an upper bound on the allowed condition number of $\boldsymbol{\beta}_{k+1}$. That is, if the ratio of its largest and smallest singular value is larger than ϵ_s, i.e., $\sigma_{\max}(\boldsymbol{\beta}_{k+1})/\sigma_{\min}(\boldsymbol{\beta}_{k+1}) > \epsilon_s$, then the algorithm breaks down. Note that the choice of $\boldsymbol{\gamma}_{k+1}$ will influence the condition number of $\boldsymbol{\beta}_{k+1}$.

In the usual non-Hermitian Lanczos algorithm, a serious breakdown can be treated by using a so-called *look-ahead* strategy; see, e.g., [8, 9, 19, 27, 28, 43, 51]. Connection between serious breakdowns, (formal) orthogonal polynomials, and matching moment property can be found in [16, 44]. If needed, an analogous look-ahead strategy may be implemented for the tensor Lanczos algorithm. At the moment, an easier strategy to deal with serious breakdowns is to reformulate the problem so to change the input vectors $\mathbf{v}$, $\mathbf{w}$. For instance, when $\mathbf{w} = \boldsymbol{e}_i$ and $\mathbf{v} = \boldsymbol{e}_i$, a serious breakdown is

likely to happen due to the sparsity of $\mathcal{A}$. However, we can rewrite the approximation of the time-ordered exponential $\mathbf{U}(t)$ as

$$\boldsymbol{e}_i^H \mathbf{U}(t)\boldsymbol{e}_j = (\boldsymbol{e}+\boldsymbol{e}_i)^H \mathbf{U}(t)\boldsymbol{e}_j - \boldsymbol{e}^H \mathbf{U}(t)\boldsymbol{e}_j,$$

with $\boldsymbol{e} = (1,\ldots,1)^H$. Then one can approximate $(\boldsymbol{e}+\boldsymbol{e}_i)^H \mathbf{U}(t)\,\boldsymbol{e}_j$ and $\boldsymbol{e}^H \mathbf{U}(t)\,\boldsymbol{e}_j$ separately, which are less likely going to have a breakdown thanks to the fact that $\boldsymbol{e}$ is a full vector; see, e.g., [25, Section 7.3] and [23].

5 Numerical examples

Let us consider the following smooth matrix-valued function defined on a real interval $I = [a, b]$:

$$\mathbf{A}(t) : I \subset \mathbb{C} \to \mathbb{C}^{N\times N}.$$

As anticipated in the Introduction, the time-ordered exponential of $\mathbf{A}(t)$ is the unique matrix-valued function $\mathbf{U}(t) \in \mathbb{C}^{N\times N}$ defined on I that is the solution of the system of linear ordinary differential equations

$$\frac{d}{dt}\mathbf{U}(t) = \mathbf{A}(t)\mathbf{U}(t), \quad \mathbf{U}(a) = I_N, \quad t \in I,$$

see [17]. In this section, we aim to approximate the bilinear form $\mathbf{w}^H\mathbf{U}(t)\mathbf{v}$ by using the tensor non-Hermitian Lanczos algorithm. If the matrix $\mathbf{A}$ is so that $\mathbf{A}(\tau_1)\mathbf{A}(\tau_2) - \mathbf{A}(\tau_2)\mathbf{A}(\tau_1) = 0$ for all $\tau_1, \tau_2 \in I$, then $\mathbf{U}(t) = \exp\left(\int_s^t \mathbf{A}(\tau)\,\mathrm{d}\tau\right)$. Unfortunately, $\mathbf{U}(t)$ – and the related bilinear forms – cannot be expressed by an analogous simple form in the general case. Indeed, even for small matrices, $\mathbf{U}(t)$ may be given by complicated special functions [32, 53].

A new approach for the approximation of a time-ordered exponential bilinear form was introduced in [22–24] and it is based on $\star$-Lanczos, which is a symbolic algorithm. This method is able to approximate the bilinear form

$$\mathbf{w}^H\mathbf{U}(t)\mathbf{v}, \quad t \in I$$

for the given vectors $\mathbf{w}$, $\mathbf{v}$, with $\mathbf{w}^H, \mathbf{v} \neq 0$. The matrices $\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_n, \boldsymbol{\beta}_2, \ldots, \boldsymbol{\beta}_n$, and $\boldsymbol{\gamma}_2, \ldots, \boldsymbol{\gamma}_n$, which compose the 4-mode tensor $\mathcal{T}_n$ in (5), are obtained by running n iterations of Algorithm 1 with, as inputs, the 4-mode tensor $\mathcal{A}$ in (10) and the vectors $\mathbf{v}$, $\mathbf{w}$.

Sampling the true solution $\mathbf{w}^H\mathbf{U}(t)\mathbf{v}$ on the discretization nodes τ_i gives the vector $\hat{\mathbf{s}}$ defined as

$$\hat{\mathbf{s}} := \left[\mathbf{w}^H\mathbf{U}(\tau_1)\mathbf{v}\ \ \mathbf{w}^H\mathbf{U}(\tau_2)\mathbf{v}\ \ \ldots\ \ \mathbf{w}^H\mathbf{U}(\tau_M)\mathbf{v}\right]^\top.$$

Exploiting the results described in [23], the sampled solution vector $\hat{\mathbf{s}}$ can be approximated by

$$\mathbf{s}_n := \frac{1}{h}\left(\boldsymbol{\theta} \times (R_*(\mathcal{T}_n))_{1,1,:,:}\right)\mathbf{e}_1 \approx \hat{\mathbf{s}}, \tag{11}$$

where R_* is the $*$-resolvent , i.e., the tensor

$$R_*(\mathcal{T}_n) := \mathcal{I}_* + \sum_{k=1}^{\infty} (\mathcal{T}_n)^{k_*},$$

and

$$\boldsymbol{\theta} := h \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 1 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 1 & 1 & \dots & 1 & 0 \\ 1 & 1 & \dots & 1 & 1 \end{bmatrix} \in \mathbb{C}^{M \times M}.$$

Overall, the accuracy of the approximation in (11) can not be better than $\mathcal{O}(h)$. This is due to the fact that, as explained in [23], the discretization (10) is based on a rectangular quadrature rule. Finally, using the Path-sum method [21] we get the following explicit expression for the $*$-resolvent in terms of a continued fraction

$$R_*(\boldsymbol{\mathcal{T}_n})_{1,1,:,:} = \left(\widetilde{\boldsymbol{\alpha}}_1 - \boldsymbol{\beta}_2 \left(\widetilde{\boldsymbol{\alpha}}_2 - \boldsymbol{\beta}_3 \left(\cdots \boldsymbol{\beta}_{n-1} \widetilde{\boldsymbol{\alpha}}_n^{-1} \boldsymbol{\gamma}_{n-1} \cdots\right)^{-1} \boldsymbol{\gamma}_3\right)^{-1} \boldsymbol{\gamma}_2\right)^{-1}, \tag{12}$$

with $\widetilde{\boldsymbol{\alpha}}_i = I_M - \boldsymbol{\alpha}_i$. (12) is computed from the most inner term moving outward, where the inversion operation is performed using the `backslash` operator in Matlab®. Note that the $*$-resolvent and all inverses appearing in (12) are expected to exist for h small enough, since their continuous counterparts exist under certain regularity conditions on $\mathbf{A}(t)$; see [22, 24].

The rest of the section is structured as follows. Section 5.1 describes the measures that will be used to quantify the errors of the final solution and of the computed biorthonormal bases for Krylov subspaces. In Section 5.2 two examples are discussed for which an analytical solution is available. This allows us to compare the approximation to an exact solution and to show that it converges with the expected rate of convergence. Small-scale examples from NMR spectroscopy are discussed in Section 5.3. Finally, in Section 5.4, we analyze the approximability of the previously considered tensors by the Tensor Train representation.

5.1 Error measures

In this section we define a series of error measures which quantify the quality of the generated biorthogonal bases and the accuracy of the approximation (11). These measures use the Frobenius norm, which, for a 4-modes tensor, is defined as

$$\|\boldsymbol{\mathcal{A}}\|_F := \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{M} \sum_{l=1}^{M} \left|(\boldsymbol{\mathcal{A}})_{i,j,k,l}\right|^2.$$

The main goal is to analyze the rate of convergence as the number of discretization points M is increased. To stress the dependence on M of computed quantities we use the superscript "(M)".

A generalization of the usual error measures for Krylov subspace methods are used. As a measure for the biorthonormality of the bases $\boldsymbol{\mathcal{V}}_n \in \mathbb{C}^{N \times n \times M \times M}$ and $\boldsymbol{\mathcal{W}}_n \in \mathbb{C}^{n \times N \times M \times M}$ generated by n steps of the algorithm, we use

$$\mathrm{err_o} := \frac{\|\boldsymbol{\mathcal{W}}_n^{(M)} * \boldsymbol{\mathcal{V}}_n^{(M)} - \boldsymbol{\mathcal{I}}_*\|_F}{\max(\|\boldsymbol{\mathcal{V}}_n^{(M)}\|_F, \|\boldsymbol{\mathcal{W}}_n^{(M)}\|_F)}.$$

For a robust algorithm, it is paramount that the term $\max(\|\mathcal{V}_n^{(M)}\|_F, \|\mathcal{W}_n^{(M)}\|_F)$ remains small as n increases. This can be obtained by employing an appropriate strategy to rescale the basisvectors in $\mathcal{V}_n^{(M)}$ and $\mathcal{W}_n^{(M)}$, i.e., by γ_{k+1} in Algorithm 1. In this section we will choose $\gamma_{k+1} = I_M$ for all k, i.e., no rescaling. An effective rescaling strategy can improve on the numerical results reported below, but developing such a strategy is subject to future research.

To measure the quality of the recurrences (1), we use

$$\mathrm{err_V} := \frac{\|\mathcal{A} * \mathcal{V}_n^{(M)} - \mathcal{V}_n^{(M)} * \mathcal{T}_n^{(M)} - \widetilde{\mathcal{V}}_n^{(M)}\|_F}{\max(\|\mathcal{A} * \mathcal{V}_n^{(M)}\|_F, \|\mathcal{V}_n^{(M)} * \mathcal{T}_n^{(M)} + \widetilde{\mathcal{V}}_n^{(M)}\|_F)},$$
$$\mathrm{err_W} := \frac{\|\mathcal{W}_n^{(M)} * \mathcal{A} - \mathcal{T}_n^{(M)} * \mathcal{W}_n^{(M)} - \widetilde{\mathcal{W}}_n^{(M)}\|_F}{\max(\|\mathcal{W}_n^{(M)} * \mathcal{A}\|_F, \|\mathcal{T}_n^{(M)} * \mathcal{W}_n^{(M)} + \widetilde{\mathcal{W}}_n^{(M)}\|_F)}.$$

As a measure for the Matching Moment Property, see Theorem 2, we use

$$\mathrm{err_M}(k) := \frac{\|W^D * (\mathcal{A}^{k_*}) * V - E_1^D * (\mathcal{T}_n^{(M)})^{k_*} * E_1\|_F}{\max(\|W^D * (\mathcal{A}^{k_*}) * V\|_F, \|E_1^D * (\mathcal{T}_n^{(M)})^{k_*} * E_1\|_F)},$$

which should be close to zero for $k = 0, \ldots, 2n-1$.

Finally, to quantify the quality of the solution, we consider as error measure for (11) the quantity

$$\mathrm{err_{sol}} := \frac{\|\hat{\mathbf{s}} - \mathbf{s}_n^{(M)}\|_2}{\|\hat{\mathbf{s}}\|_2},$$

where, if no analytic expression is available, an approximation of $\mathbf{s}$ is obtained by using `ode45` in Matlab®. In the formula above, $\|\cdot\|_2$ stands for the usual Euclidean norm. The rate at which $\mathrm{err_{sol}}$ decreases as M increases is expected to be $\mathcal{O}(h) = \mathcal{O}(1/M)$, i.e., the accuracy of the discretization used here.

5.2 Proof of concept

As a proof of concept, we test our proposal on two problems which originally appeared in [23]. In both experiments a discretization with M points is used and we run n iterations of Algorithm 1 with $\gamma_{k+1} \equiv I_M$. This produces the tensor $\mathcal{T}_n^{(M)}$ defined in (5) with coefficients $\boldsymbol{\alpha_1}^{(M)}, \ldots, \boldsymbol{\alpha_n}^{(M)}$ and $\boldsymbol{\beta_2}^{(M)}, \ldots, \boldsymbol{\beta_n}^{(M)}$, depending on M. For the two experiments considered here the result of the $\star$-Lanczos algorithm [23] is known. The coefficients resulting from the latter algorithm are bivariate functions $\alpha_1(t,s), \ldots, \alpha_n(t,s)$ and $\beta_2(t,s), \ldots, \beta_n(t,s)$, because $\star$-Lanczos is a symbolic algorithm. The tensor Lanczos algorithm is a discretization of the $\star$-Lanczos algorithm, which means that $\boldsymbol{\alpha_i}^{(M)}$ and $\boldsymbol{\beta_i}^{(M)}$ can be seen as discretizations of the functions $\alpha_i(t,s)$ and $\beta_i(t,s)$, respectively.
Consider the evaluation of these functions on the mesh τ_i:

$$\hat{\boldsymbol{\alpha}}_i := \left[\alpha(\tau_i, \tau_j)\right]_{i,j=1}^{M}, \qquad \hat{\boldsymbol{\beta}}_i := \left[\beta(\tau_i, \tau_j)\right]_{i,j=1}^{M},$$

then, we can define the errors $\frac{\|\hat{\boldsymbol{\alpha}}_i - \boldsymbol{\alpha}_i^{(M)}\|_2}{\|\hat{\boldsymbol{\alpha}}_i\|_2}$, $i = 1, \ldots, n$, and $\frac{\|\hat{\boldsymbol{\beta}}_i - \boldsymbol{\beta}_i^{(M)}\|_2}{\|\hat{\boldsymbol{\beta}}_i\|_2}$, $i = 2, \ldots, n$. These errors will be used as a measure for the accuracy of the computed tensor $\mathcal{T}_n^{(M)}$. The number of iterations n is chosen equal to the problem size N, which allows us to compare all the available functions $\alpha_1(t,s), \ldots, \alpha_N(t,s)$ and $\beta_2(t,s), \ldots, \beta_N(t,s)$ with the elements in $\mathcal{T}_N^{(M)}$ and track the convergence rate with M of the latter.

5.2.1 Time-independent matrix

Consider a constant matrix and starting vectors

$$\mathbf{A}(t) = \begin{bmatrix} -1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & -1 \end{bmatrix}, \qquad \mathbf{v}, \mathbf{w} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix},$$

and the interval $I = [0, 1]$. The inputs of Algorithm 1 are the starting hypervectors $\mathbf{v} \otimes I_M$, $\mathbf{w} \otimes I_M$ and the tensor $\boldsymbol{\mathcal{A}}$ whose components $\boldsymbol{\mathcal{A}}_{i_1,i_2,:,:}$ are defined as

$$\boldsymbol{\mathcal{A}}_{i_1,i_2,:,:} = \begin{cases} \boldsymbol{\theta}, & \text{if } i_1 = i_2 = 1 \text{ or } i_1 = i_2 = 3 \\ \mathbf{0}, & \text{if } i_1 = i_2 = 2 \\ -\boldsymbol{\theta}, & \text{otherwise} \end{cases},$$

where $\mathbf{0} \in \mathbb{R}^{M \times M}$ is the null matrix. The tensor $\boldsymbol{\mathcal{A}}$ is obtained by sampling the matrix-valued function $\mathbf{A}(t)$ on the M point mesh (9) and following the definition in (10). The output for $n = N = 3$ iterations is $\mathcal{T}_N^{(M)}$. Table 3 reports the Krylov error measures, which behave as expected for the recurrence measures. The loss of biorthogonality observed for increasing values of M is presumably due to the fact that no rescaling is used in the algorithm.

On the other hand, this loss of biorthogonality does not compromise the moment matching capabilities of $\mathcal{T}_N^{(M)}$, as it becomes evident from Table 4 where we report $\mathrm{err_M}(k)$ for $k \leq 5 = 2n - 1$.

Moreover, as the values reported in Table 5 confirm, we can observe that the elements $\boldsymbol{\beta}_i$ converge at the expected rate of $\mathcal{O}(1/M)$. The elements $\boldsymbol{\alpha}_i$ are very accurate for $M = 10$ whereas for larger M, the accuracy of $\boldsymbol{\alpha}_i$ decreases: this decrease is, presumably, the result of error propagation in the numerical algorithm. This error is still smaller than the expected order of $\mathcal{O}(1/M)$.

Table 3 Krylov error measures for time-independent matrix

M	10	100	1000
$\mathrm{err_o}$	2.212e-15	4.127e-13	5.389e-11
$\mathrm{err_V}$	2.234e-16	6.356e-15	4.327e-14
$\mathrm{err_W}$	0	0	0

Table 4 Measure for moment matching $\mathrm{err}_{\mathrm{M}}(k)$ for time-independent matrix. Entries for $k = 0, 1, 2$ are omitted since they are equal to zero

$k \backslash M$	10	100	1000
3	6.078e-17	9.464e-17	9.580e-16
4	6.611e-17	2.806e-16	1.199e-15
5	9.840e-17	2.558e-16	2.844e-15

Moreover, in this particular case, the analytical solution to the ODE is known; see [23]:

$$\hat{\mathbf{s}} = \left[(\exp(A\tau_1))_{11} \ (\exp(A\tau_2))_{11} \ \dots \ (\exp(A\tau_M))_{11}\right]^\top,$$

with $(\exp(At))_{11} = -\frac{1}{2}\sinh(2t) + \frac{1}{2}\cosh(2t) + \frac{1}{2}\cosh(\sqrt{2}t)$. Hence, it is possible to compare this exact solution with (11). Note that for $n = 3$ the $*$-resolvent is given by the continued fraction:

$$R_*(\mathcal{T}_3)_{1,1,:,:} = \left(I_M - \boldsymbol{\alpha}_1 - \left(I_M - \boldsymbol{\alpha}_2 - (I_M - \boldsymbol{\alpha}_3)^{-1}\boldsymbol{\beta}_3\right)^{-1}\boldsymbol{\beta}_2\right)^{-1}.$$

Table 6 shows the error measure $\mathrm{err}_{\mathrm{sol}}$ for increasing M, which convergences at the expected rate $\mathcal{O}(1/M)$.

5.2.2 Time-dependent matrix

Consider the time-dependent matrix

$$\tilde{\mathbf{A}}(t) = \begin{bmatrix} \cos(t) & 0 & 1 & 2 & 1 \\ 0 & \cos(t) - t & 1 - 3t & t & 0 \\ 0 & t & 2t + \cos(t) & 0 & 0 \\ 0 & 1 & 2t + 1 & t + \cos(t) & t \\ t & -t - 1 & -6t - 1 & 1 - 2t & \cos(t) - 2t \end{bmatrix},$$

the starting vectors $\mathbf{v} = \mathbf{w} = \left[1\ 0\ 0\ 0\ 0\right]^\top$, and the interval $I = [10^{-4}, 1]$. As it becomes apparent from the results reported in Table 7, for this particular experiment, we obtain that the Krylov error measures and the recurrence measures are small whereas the biorthogonality measure is large.

Table 5 Error on the elements of the tridiagonal tensor for time-independent matrix

M	10		100		1000	
i	$\boldsymbol{\alpha}_i$	$\boldsymbol{\beta}_i$	$\boldsymbol{\alpha}_i$	$\boldsymbol{\beta}_i$	$\boldsymbol{\alpha}_i$	$\boldsymbol{\beta}_i$
1	0	/	0	/	0	/
2	7.639e-16	2.365e-01	3.867e-13	2.250e-02	1.555e-10	2.240e-03
3	2.875e-15	2.365e-01	1.283e-11	2.250e-02	5.118e-08	2.240e-03

Table 6 Error of approximation to the quantity of interest $\mathbf{w}^H \mathbf{U}(t)\mathbf{v}$ for time-independent matrix

M	10	100	1000
err_{sol}	8.230e-02	7.019e-03	6.918e-04

The loss of orthogonality is an inherent feature of Lanczos-like algorithms, and it does not necessarily compromise algorithm's capability to produce an approximation to the bilinear form. Indeed, Table 8 confirms that the matching moment property is not affected by the loss of $*$-biorthogonality.

On the other hand, the results presented in Table 9, where we report the error measures for the coefficients computed by Algorithm 1, show that the loss of $*$-biorthogonality of the computed bases has a limited impact for the convergence of the algorithm to the solution. Indeed, in this case, for all $\boldsymbol{\beta}_i$ the expected convergence rate is observed whereas, for $\boldsymbol{\alpha_i}$, only $i = 1, 2, 3$ show the expected decrease in the error measure.

Table 10 shows that the approximation to the quantity of interest converges at the rate $\mathcal{O}(1/M)$. Hence, the loss of biorthogonality and the inaccurate coefficients of the tridiagonal tensor did not compromise this approximation.

5.3 NMR experiments

Nuclear magnetic resonance (NMR) spectroscopy studies the structure and dynamics of molecules by looking at nuclear spins [31, 39]. Computer simulations of NMR experiments are important because they can improve the design and analysis of laboratory experiments [50]. In this section, three small, realistic examples arising from NMR spectroscopy [3] are discussed. The ODE that governs the dynamics of nuclear spins during NMR spectroscopy is the Schrödinger equation

$$\frac{d}{dt}\phi(t) = -\imath 2\pi H(t)\phi(t), \quad t \in \left[0, \tau_{\text{exp}}\right]$$

where $H(t)$ is the so-called Hamiltonian, $\phi(t)$ the wave function, τ_{exp} the duration of the experiment and $\imath = \sqrt{-1}$. The size of the Hamiltonian is related to the number of nuclear spins present in the system, for l nuclear spins $H(t)$ is of the size $2^l \times 2^l$.

Table 7 Krylov error measures for time-dependent matrix

M	10	100	1000
err_{o}	1.069e-01	7.866e-01	8.645e-01
err_{V}	5.262e-16	1.557e-14	8.780e-16
err_{W}	7.062e-18	1.077e-17	1.041e-17

Table 8 Measure for moment matching $\text{err}_{\text{M}}(k)$ for time-dependent matrix

$k \backslash M$	10	100	1000
3	5.439e-17	1.375e-16	4.226e-16
4	1.357e-16	1.923e-16	4.514e-16
5	1.231e-16	2.370e-16	5.053e-15
6	1.443e-16	3.069e-16	7.870e-15
7	1.479e-16	3.553e-16	2.704e-14
8	1.654e-16	3.271e-16	5.282e-14
9	1.530e-16	3.775e-16	3.022e-13

Entries for $k = 0, 1, 2$ are omitted since they are equal to zero

Hence, $H(t)$ grows exponentially with the number of spins, but it is a sparse matrix, making it an ideal candidate for a Lanczos-like algorithm.

The experiments discussed in this section use $M = 500$ discretization points, because of memory constraints. It is important to note that such memory constraints may be overcome using the Tensor Train approximation presented in Section 5.4. The number of iterations of the tensor Lanczos algorithm is chosen to obtain the maximal attainable accuracy, which is determined by the discretization scheme with $M = 500$. Since the discretization used here has an accuracy of $\mathcal{O}(1/M)$, the smallest number of iterations n such that err_{sol} is of order 10^{-3} suffices. Choosing larger n will not decrease err_{sol} further for a fixed M and will increase the computational cost.

5.3.1 Experiment 1: Weak coupling

Consider four nuclear spins with heteronuclear dipolar couplings. In this framework, the Hamiltonian for a magic angle spinning (MAS) experiment [29] is the diagonal matrix

$$H(t) = \text{diag}\left[\{f_k(t)\}_{k=1}^{16}\right], \quad f_k(t) = \alpha_k + \beta_k \cos(2\pi\nu t) + \gamma_k \cos(4\pi\nu t),$$

Table 9 Error on the elements of the tridiagonal tensor for time-dependent matrix

M	10		100		1000	
i	$\boldsymbol{\alpha}_i$	$\boldsymbol{\beta}_i$	$\boldsymbol{\alpha}_i$	$\boldsymbol{\beta}_i$	$\boldsymbol{\alpha}_i$	$\boldsymbol{\beta}_i$
1	0	/	0	/	0	/
2	6.986e-13	2.560e-01	2.139e-10	2.388e-02	4.751e-09	2.373e-03
3	2.034e-02	4.057e-01	2.098e-03	3.493e-02	6.507e-03	3.445e-03
4	6.473e-02	2.858e-01	6.679e-03	2.842e-02	9.859e-01	2.845e-03
5	1.452e-01	1.754e-01	2.551e+01	1.695e-02	8.365e+05	1.840e-03

Table 10 Error of approximation to the quantity of interest $\mathbf{w}^H \mathbf{U}(t)\mathbf{v}$ for time-dependent matrix

M	10	100	1000
err_{sol}	2.360e-01	2.257e-02	2.404e-03

with $\alpha_k, \beta_k, \gamma_k \in \mathbb{R}$ and $\nu = 10^4$. The diagonal matrix $A(t) = -\imath 2\pi H(t)$ commutes with itself at all times and thus the solution $U(t)$ can be computed

$$U(t) = \text{diag}\left[\left\{\exp\left(-\imath\alpha_k t - \imath\frac{\beta_k}{2\pi\nu}\sin(2\pi\nu t) - \imath\frac{\gamma_k}{4\pi\nu}\sin(4\pi\nu t)\right)\right\}_{k=1}^{16}\right].$$

The starting vectors are chosen to excite and measure the lowest oscillatory components in $U(t)$: $\mathbf{w} = \mathbf{v} = \begin{bmatrix}0 & 1 & 1 & 0 & 1 & 1 & \dots & 0 & 1 & 1\end{bmatrix}^\top$. A typical experiment would run for a time of the order 10^2 seconds. Since the problem is (highly) oscillatory and the current discretization requires many points to accurately compute a solution, we choose to restrict the experiment time to $\tau_{\text{exp}} = 5 \times 10^{-5}$. This is a valid approach since the total time interval of 10^{-2} can be split into subintervals of length 5×10^{-5} and the solutions on the subintervals can be combined to obtain the solution on the whole interval.

Algorithm 1 is run for $n = 3$ iterations and the corresponding Krylov error measures are shown in Table 11. A first observation concerns the fact that going from $M = 5$ to $M = 50$ a large decrease of the biorthogonality measure is observed. This is due to the fact that when discretizing with fewer discretization points, e.g., $M = 5$, the original matrix in the ODE $-\imath 2\pi H(t)$ is translated into a simpler (and inaccurate) discretized input, for which the tensor Lanczos iteration converges fast. The discretization $M = 50$ represents the original input better, as is suggested by the stagnation of err_{o} going from $M = 50$ to $M = 500$.

The error err_{sol} is computed using the analytical solution $\mathbf{w}^H \mathbf{U}(t)\mathbf{v}$ evaluated in the discretization points and decays at the expected rate $\mathcal{O}(1/M)$. Figure 1 shows $\hat{\mathbf{s}}$ and the approximation $\mathbf{s}_n^{(M)}$ as a function of time; such approximation clearly converges for increasing M (x-axis reports time).

Table 11 Error measures for Experiment 1

M	5	50	500
err_{o}	1.435e-03	1.422e-08	1.298e-08
err_{V}	2.947e-16	8.804e-14	7.063e-11
err_{W}	4.015e-17	2.553e-17	2.772e-17
err_{sol}	1.810e-01	2.024e-02	2.076e-03

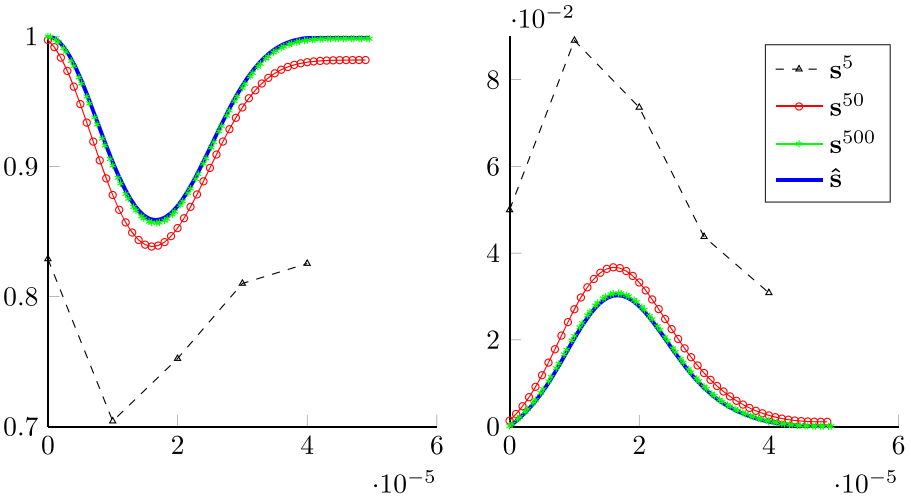

Fig. 1 Quantity of interest $\hat{\mathbf{s}}$ and approximation $\mathbf{s}_n^{(M)}$ for Experiment 1. Real part on the left and imaginary part on the right. x-axis reports time

5.3.2 Experiment 2: Strong coupling

MAS with four nuclear spins with homonuclear dipolar couplings leads to the Hamiltonian

$$H(t) = \operatorname{diag}\left[\{\alpha_k\}_{k=1}^{16}\right] + B\cos(2\pi\nu t) + C\cos(4\pi\nu t),$$

where $\alpha_k \in \mathbb{R}$ is a scalar, and $B, C \in \mathbb{R}^{16\times 16}$ are matrices with a sparsity structure as shown in Fig. 2.

A typical experiment time is 10^{-2} seconds and $\nu = 10^4$. The simulated experiment time is $\tau_{\text{exp}} = 5 \times 10^{-6}$, the size of the Krylov subspace is $k = 4$ and $\mathbf{v} = \mathbf{w} = \left[0\ 1\ 1\ 0\ 1\ 1\ \ldots\ 0\ 1\ 1\right]^\top$. The corresponding error measures are shown in Table 12,

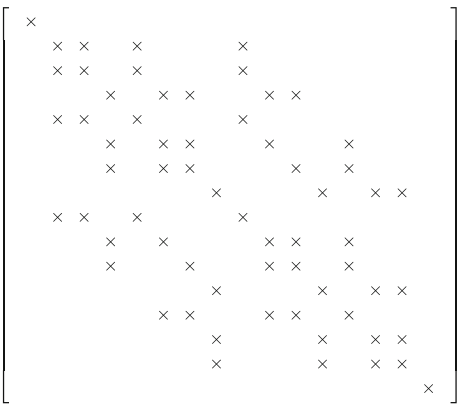

Fig. 2 Sparsity structure of B and C in Experiment 2, $\times$ denotes a nonzero element

Table 12 Error measures for Experiment 2

M	5	50	500
err_o	3.596e-03	7.096e-05	7.028e-06
err_V	1.798e-16	4.573e-15	1.492e-13
err_W	3.783e-17	3.454e-17	3.508e-17
err_sol	3.877e-01	5.018e-02	4.281e-03

which show a similar behavior as for Experiment 1. The measure err_sol is computed by comparing $\hat{\mathbf{s}}$ to the solution obtained by `ode45`.

5.3.3 Experiment 3: Uncoupled spins under a pulse wave

The Hamiltonian for four uncoupled spins under a pulse wave is

$$\begin{aligned} H(t) &= \text{diag}\left[\{\alpha_k\}_{k=1}^{16}\right] + B(0.5 + \cos(4t) + \sin(10t) - 0.4\sin(16t)) \\ &\quad + C(\sin(4t) + \cos(8t) + 2\sin(12t)), \end{aligned}$$

with $\alpha_k \in \mathbb{R}$ and $B \in \mathbb{R}^{16\times 16}$, $C \in \mathbb{C}^{16\times 16}$ have a structure as shown in Fig. 3.

A practical experiment time ranges from 10^{-6} to 10^{-3} seconds, here $\tau_{\text{exp}} = 10^{-3}$ is used. The starting vectors are $\mathbf{v} = \mathbf{w} = \begin{bmatrix}1 & \dots & 1\end{bmatrix}^\top$ and $n = 4$ iterations of the tensor Lanczos algorithm are run. The Krylov error measures shown in Table 13 behave similarly to the measures observed for Experiments 1 and 2. The measure

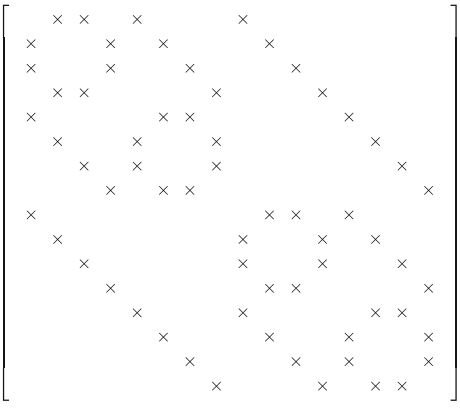

Fig. 3 Structure of B and C in Experiment 3, $\times$ denotes a nonzero element

Table 13 Error measures for Experiment 3

M	5	50	500
err_o	2.994e-03	2.730e-05	2.708e-07
err_V	2.520e-16	4.122e-15	1.237e-13
err_W	1.203e-17	1.814e-17	1.008e-17
err_sol	3.476e-01	4.447e-02	5.145e-03

err_sol is obtained via `ode45` and shows a convergence rate a bit slower than $\mathcal{O}(1/M)$. The slower convergence rate can, in part, be explained by the fact that a comparison is made with the `ode45` solution. Additional errors are incurred when comparing $\mathbf{s}$ with $\hat{\mathbf{s}}$, because the former is only available in the points τ_i and the latter is available only in points which are determined by `ode45`.

5.4 Tensor Train approximations

As briefly mentioned in the Introduction, despite the fact that the block matrix and the tensor formulation of the problem (1) are mathematically equivalent, the tensor formulation introduced and analyzed in this work, allows the exploitation of particular low parametric representations. The aim of this section is indeed to show that for all the examples previously presented, the resulting tensors can be accurately and conveniently approximated using a low parametric representation called Tensor Train (TT) format [41, 42].

Table 14 TT ranks and compression factor for Experiment 1

M	$nnz(\boldsymbol{\mathcal{A}})$	TT Ranks	$\sum_{k=1}^{4} r_{k-1} \times n_k \times r_k$	C.F.
		$\nu = 1e+4,\ tol = 1e-5$		
500	2004000	1 16 2 498 1	747768	0.37314
1000	8008000	1 16 2 900 1	2700768	0.33726
1500	18012000	1 16 2 1476 1	6642768	0.3688
		$\nu = 1e+4,\ tol = 1e-10$		
500	2004000	1 16 2 500 1	750768	0.37463
1000	8008000	1 16 2 998 1	2994768	0.37397
1500	18012000	1 16 2 1500 1	6750768	0.37479
		$\nu = 1e+1,\ tol = 1e-10$		
500	2004000	1 16 2 500 1	750768	0.37463
1000	8008000	1 16 2 1000 1	3000768	0.37472
1500	18012000	1 16 2 1500 1	6750768	0.37479

Table 15 TT ranks and compression factor for Experiment 2

M	$nnz(\mathcal{A})$	TT Ranks	$\sum_{k=1}^{4} r_{k-1} \times n_k \times r_k$	C.F.
		$\nu = 1e+4,\ tol = 1e-5$		
500	8016000	1 15 1 498 1	498480	0.062186
1000	29627200	1 15 1 900 1	1800480	0.060771
1500	72048000	1 15 1 1476 1	4428480	0.061466
		$\nu = 1e+4,\ tol = 1e-10$		
500	8016000	1 16 2 500 1	750768	0.093659
1000	29627200	1 16 2 997 1	2991768	0.10098
1500	72048000	1 16 2 1500 1	6750768	0.093698
		$\nu = 1e+1,\ tol = 1e-10$		
500	8016000	1 16 2 500 1	750768	0.093659
1000	32032000	1 16 2 1000 1	3000768	0.09368
1500	72048000	1 16 2 1500 1	6750768	0.093698

As a matter of fact, multilinear algebra, tensor analysis, and the theory of tensor approximations play increasingly important roles in nowadays computational mathematics and numerical analysis, thereby attracting tremendous interest in recent years [12]. In this panorama, Tensor Train (TT) approximations are a powerful technique for dealing with *the curse of dimensionality*, i.e., the particularly unpleasant feature where the number of unknowns and the computational complexity grow exponentially when the dimension of the problem increases.

Before presenting the computational results, we briefly survey the main features of the TT representation, addressing the interested reader to the surveys [11, 12]. We consider the Tensor Train (TT) format [41] for the tensors of interest in this work. Specifically, a 4-mode $\mathcal{A} \in \mathbb{C}^{N_1 \times N_2 \times M \times M}$ tensor is expressed in TT format when

$$\mathcal{A}_{i_1,i_2,i_3,i_4} = G_1(i_1)G_2(i_2)G_3(i_3)G_4(i_4)$$

Table 16 TT ranks and compression factor for Experiment 3

M	$nnz(\mathcal{A})$	TT Ranks	$\sum_{k=1}^{4} r_{k-1} \times n_k \times r_k$	C.F.
		$\nu = 1e+4,\ tol = 1e-5$		
500	10020000	1 12 2 499 1	749076	0.074758
1000	40040000	1 12 2 996 1	2988576	0.07464
1500	90060000	1 12 2 1493 1	6719076	0.074607
		$\nu = 1e+4,\ tol = 1e-10$		
500	10020000	1 16 2 500 1	750768	0.074927
1000	40040000	1 16 2 1000 1	3000768	0.074944
1500	90060000	1 16 2 1500 1	6750768	0.074959

where $G_k(i_k)$ is a matrix of dimension $r_{k-1} \times r_k$ and $r_0 = r_4 = 1$. The numbers r_k are called *TT-ranks*, and $G_k(i_k)$ are the *cores* of the TT decomposition. If $r_k \leq r$, $n_k \leq n$, then storing the TT representation requires memorizing $\leq 4nr^2$ numbers. If r is small, then the memory requirement is much smaller than storing the full tensor, i.e., storing n^4 numbers.

It is important to note that the TT representation allows to approximate various tensor operations efficiently, see, e.g., [41, Sec. 4]. In this paper, we do not propose a low parametric TT version of Algorithm 1. To be efficient, such a TT version would need a TT representation of the tensor products used in Algorithm 1. This paper aims to show that Algorithm 1 works; further enhancements are postponed to future investigations.

In Tables 14, 15, and 16 we present the TT ranks for all the tensors considered in Section 5.3. In particular, the tables present the details of the TT approximations obtained using the `TT-toolbox` [41] when the required accuracy for the approximation is set to 10^{-5} and 10^{-10}. As becomes evident from the presented results, all the considered tensors are amenable of a low parametric representation provided by the TT format and, indeed, for all the presented results the Compression Factor (C.F.), which is defined as $(\sum_{k=1}^{4} r_{k-1} \times n_k \times r_k)/nnz(\mathcal{A})$, with $nnz(\mathcal{A})$ the number of nonzero elements of $\mathcal{A}$, lies in the interval $(10^{-3}, 0.5)$. It is important to observe that when increasing the accuracy from 10^{-5} to 10^{-10} the C.F. does not significantly change, suggesting the interesting fact that for the considered tensors, the TT format is closer to an *exact representation* rather than an *approximation*. Finally, it is important to note that the ranks of the TT approximations are robust across the choices of the parameter ν (cfr. the TT ranks in Table 14 and in Table 15) and to note also that, for some of the considered problems, the number of parameters needed for the *approximation* can be two orders of magnitude smaller than $nnz(\mathcal{A})$; see Tables 15 and 16.

6 Conclusions

In this work we introduced a non-Hermitian Lanczos algorithm for tensors and we provided the corresponding theoretical analysis. In particular, after introducing all the necessary theoretical background, we are able to interpret such Lanczos-type process in terms of tensor polynomials and to prove the related matching moment property. A series of numerical experiments performed on real-world problems confirm the effectiveness of our approach. Using a linearly converging approximation for the inputs, the algorithm produces a linearly converging approximation of the bilinear form $\mathbf{w}^H \mathbf{U}(t)\mathbf{v}$, where $\mathbf{U}(t)$ is the solution of the ODE (1). More accurate approximation schemes for the inputs are currently being developed by some of the paper's authors, possibly leading to faster convergence. Moreover, in all the considered examples, the related tensors show a low parametric structure in terms of Tensor Train representation. This important feature paves the path for future efficiency improvements of our proposal where this representation is fully exploited in the Lanczos-type procedure.

Appendix

A.1 The $*$-product associativeness

In the following, we prove the three statements of Lemma 1. First, we have

$$((\mathcal{A} * \mathcal{A}) * A)_{i_1,:,:} = \sum_{k=1}^{N_1} (\mathcal{A} * \mathcal{A})_{i_1,k,:,:} A_{k,:,:} = \sum_{k=1}^{N_1} (\sum_{j=1}^{N_1} \mathcal{A}_{i_1,j,:,:} \mathcal{A}_{j,k,:,:}) A_{k,:,:} =$$
$$\sum_{j=1}^{N_1} \mathcal{A}_{i_1,j,:,:} (\sum_{k=1}^{N_1} \mathcal{A}_{j,k,:,:} A_{k,:,:}) = \sum_{j=1}^{N_1} \mathcal{A}_{i_1,j,:,:} (\mathcal{A} * A)_{j,:,:} = (\mathcal{A} * (\mathcal{A} * A))_{i_1,:,:}.$$

The second statement is proved by direct inspection as follows:

$$((B^D * \mathcal{A})^D * A)_{:,:} = \sum_{k=1}^{N_2} (B^D * \mathcal{A})^D_{k,:,:} A_{k,:,:} = \sum_{k=1}^{N_2} (\sum_{j=1}^{N_1} B^D_{j,:,:} \mathcal{A}_{j,k,:,:}) A_{k,:,:} =$$
$$\sum_{j=1}^{N_1} B^D_{j,:,:} (\sum_{k=1}^{N_2} \mathcal{A}_{j,k,:,:} A_{k,:,:}) = \sum_{j=1}^{N_1} B^D_{j,:,:} (\mathcal{A} * A)_{j,:,:} = B^D * (\mathcal{A} * A).$$

Finally, we have the following equality

$$((\mathcal{C} * \mathcal{A}) * \mathcal{B})_{i_1,i_2,:,:} = \sum_{k=1}^{N_2} (\mathcal{C} * \mathcal{A})_{i_1,k,:,:} \mathcal{B}_{k,i_2,:,:} = \sum_{k=1}^{N_2} (\sum_{j=1}^{N_1} \mathcal{C}_{i_1,j,:,:} \mathcal{A}_{j,k,:,:}) \mathcal{B}_{k,i_2,:,:} =$$
$$\sum_{j=1}^{N_1} \mathcal{C}_{i_1,j,:,:} (\sum_{k=1}^{N_2} \mathcal{A}_{j,k,:,:} \mathcal{B}_{k,i_2,:,:}) = \sum_{j=1}^{N_1} \mathcal{C}_{i_1,j,:,:} (\mathcal{A} * \mathcal{B})_{j,i_2,:,:} = (\mathcal{C} * (\mathcal{A} * \mathcal{B}))_{i_1,i_2,:,:}.$$

This concludes the proof of Lemma 1.

A.2 Matching moment property

We prove Theorem 2 by giving a polynomial interpretation of the Lanczos process for tensors. The proof follows the same principles as the proof given in [23] for a different but analogous case. For simplicity, we consider the case in which the matrices $\boldsymbol{\gamma}_k$ are set equal to the matrix identity. The proof can be easily extended to the general case. Let us define the following set of polynomials

$$\mathcal{P}_* := \left\{ p(\lambda) = \sum_{k=0}^{\ell} \lambda^{k_*} \times \boldsymbol{\eta}_k \right\},$$

with $\boldsymbol{\eta}_k \in \mathbb{C}^{M \times M}$. We say that the map $[\cdot, \cdot] : \mathcal{P}_* \times \mathcal{P}_* \to \mathbb{C}^{M \times M}$ is a *sesquilinear block form* if and only if, given $p, q, r, s \in \mathcal{P}_*$ and $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{C}^{M \times M}$, it satisfies

$$[q \times \boldsymbol{\alpha}, p \times \boldsymbol{\beta}] = \boldsymbol{\alpha}^H \times [q, p] \times \boldsymbol{\beta},$$
$$[q + s, p + r] = [q, p] + [s, p] + [q, r] + [s, r].$$

In addition, from now on we assume that for every sesquilinear block form $[\cdot, \cdot]$ it holds that

$$[\lambda * q, p] = [q, \lambda * p]. \tag{13}$$

Then, $[\cdot,\cdot]$ is determined by its *moments* defined as

$$\boldsymbol{\mu}_k := [\lambda^{k_*}, 1] = [1, \lambda^{k_*}], \quad k = 0, 1, \ldots .$$

We say that the sequences $p_0, p_1, \ldots$ and $q_0, q_1, \ldots$ from $\mathcal{P}_*$ are sequences of *biorthonormal polynomials* with respect to $[\cdot,\cdot]$ if and only if

$$[q_i, p_j] = \delta_{ij} I_M, \tag{14}$$

with δ_{ij} the Kronecker delta (hereafter the subindex k in p_k and q_k will stand for the degree of the polynomial). From now on, we also assume $m_0 = I_M$, getting $p_0 = q_0 = I_M$. Let q_1 be the polynomial from $\mathcal{P}_*$

$$q_1(\lambda) = \lambda * q_0(\lambda) - q_0(\lambda) \times \boldsymbol{\alpha}_0^H,$$

the orthogonality conditions (14) imply

$$\boldsymbol{\alpha}_0 = [\lambda * q_0, p_0].$$

Analogously, we get

$$p_1(\lambda) \times \boldsymbol{\beta}_1 = \lambda * p_0(\lambda) - p_0(\lambda) \times \boldsymbol{\alpha}_0,$$

with

$$\boldsymbol{\alpha}_0 = [q_0, \lambda * p_0], \quad \boldsymbol{\beta}_1 = [q_1, \lambda * p_0].$$

Repeating such a biorthogonalization process, we obtain the three-term recurrences for $k = 0, 1, \ldots$

$$q_{k+1}(\lambda) = \lambda * q_k(\lambda) - q_k(\lambda) \times \boldsymbol{\alpha}_k^H - q_{k-1}(\lambda) \times \boldsymbol{\beta}_k^H \tag{15a}$$

$$p_{k+1}(\lambda) \times \boldsymbol{\beta}_{k+1} = \lambda * p_k(\lambda) - p_k(\lambda) \times \boldsymbol{\alpha}_k - p_{k-1}(\lambda), \tag{15b}$$

with $p_{-1} = q_{-1} = 0$ and

$$\boldsymbol{\alpha}_k = [q_k, \lambda * p_k], \quad \boldsymbol{\beta}_{k+1} = [q_{k+1}, \lambda * p_k]. \tag{16}$$

We remark that the recurrences are obtained using property (13). The previous derivation also constructively proves that the biorthonormal polynomials $p_0, \ldots, p_n$ and $q_0, \ldots, q_n$ exist if $\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_n$ are invertible matrices.

Let $\boldsymbol{\mathcal{A}}$, V, W be as in Theorem 2 and let us define the sesquilinear block form

$$[q, p]_{\boldsymbol{\mathcal{A}}} = W^D * q^D(\boldsymbol{\mathcal{A}}) * p(\boldsymbol{\mathcal{A}}) * V.$$

Assume that there exist polynomials $p_0, \ldots, p_n$ and $q_0, \ldots, q_n$ from $\mathcal{P}_*$ which are biorthonormal with respect to $[\cdot,\cdot]_{\boldsymbol{\mathcal{A}}}$. Defining the vectors

$$V_k = p_{k-1}(\boldsymbol{\mathcal{A}}) * V, \qquad W_k^D = W^D * q_{k-1}^D(\boldsymbol{\mathcal{A}}),$$

and using the recurrences (1) we get the recurrences (3.2) of the non-Hermitian Lanczos for tensors. Moreover, the coefficients in (16) are the coefficients in (4).

Let $\boldsymbol{\mathcal{T}}_n$ be as in Theorem 2. We can define the sesquilinear block form $[q, p]_n : \mathcal{P}_* \times \mathcal{P}_* \to \mathbb{C}^{M \times M}$ as

$$[q, p]_n = E_1^D * q^D(\boldsymbol{\mathcal{T}}_n) * p(\boldsymbol{\mathcal{T}}_n) * E_1.$$

Note that here the vector $\mathbf{e}_1$ in the definition of $E_1 = \mathbf{e}_1 \otimes I_M$ has length $n \leq N$. The following Lemmas will show that

$$\boldsymbol{\mu}_k = [\lambda^{*k}, 1]_{\boldsymbol{\mathcal{A}}} = [\lambda^{*k}, 1]_n, \quad k = 0, \ldots, 2n-1,$$

concluding the proof of Theorem 2.

Lemma 2 *Let $p_0, \dots, p_n \in \mathcal{P}_*$ and $q_0, \dots, q_n \in \mathcal{P}_*$ be biorthonormal polynomials with respect to $[\cdot,\cdot]_{\mathcal{A}}$. Assume that $\boldsymbol{\beta}_1, \dots, \boldsymbol{\beta}_n$ in (1) are invertible matrices. Then the polynomials are also biorthonormal with respect to $[\cdot,\cdot]_n$ as defined above.*

Proof Let us define the tensors $E_i := \mathbf{e}_i \otimes I_M$ for $i = 1, \dots, n$, with $I_n = [\mathtt{e}_1, \dots, \mathtt{e}_n]$. We will first prove by induction that for $i = 0, \dots, n-1$

$$E_{i+1} = p_i(\boldsymbol{\mathcal{T}}_n) * E_1, \quad E^D_{i+1} = E^D_1 * q^D_i(\boldsymbol{\mathcal{T}}_n). \tag{17}$$

For $i = 0$, the relations (17) are trivial. Assume now that (17) hold for $i = 1, \dots, k$, by (1) we get

$$\begin{aligned} E^D_1 * q^D_{k+1}(\boldsymbol{\mathcal{T}}_n) &= E^D_{k+1} * \boldsymbol{\mathcal{T}}_n - \boldsymbol{\alpha}_k \times E^D_{k+1} - \boldsymbol{\beta}_k \times E^D_k, \\ p_{k+1}(\boldsymbol{\mathcal{T}}_n) \times \boldsymbol{\beta}_{k+1} &= \boldsymbol{\mathcal{T}}_n * E_{k+1} - E_{k+1} \times \boldsymbol{\alpha}_k - E_k. \end{aligned}$$

Since $\boldsymbol{\beta}_{k+1}$ is invertible, direct computations prove that (17) holds $i = k+1$.

As a consequence we have

$$[q_i, p_j]_n = E^D_{i+1} * E_{j+1} = \delta_{ij} I_M,$$

which concludes the proof. □

Lemma 3 *Let $p_0, \dots, p_{n-1} \in \mathcal{P}_*$ and $q_0, \dots, q_{n-1} \in \mathcal{P}_*$ be biorthonormal polynomials with respect to a sesquilinear block form $[\cdot,\cdot]_{\mathcal{A}}$ and to a sesquilinear block form $[\cdot,\cdot]_{\mathcal{B}}$. If $[1,1]_{\mathcal{A}} = [1,1]_{\mathcal{B}} = I_M$, then $[\lambda^{k_*}, 1]_{\mathcal{A}} = [\lambda^{k_*}, 1]_{\mathcal{B}}$ for $k = 0, \dots, 2n-1$.*

Proof The proof is by induction. Let $\boldsymbol{\mu}_k = [\lambda^{k_*}, 1]_{\mathcal{A}}$ and $\widehat{\boldsymbol{\mu}}_j = [\lambda^{k_*}, 1]_{\mathcal{B}}$ for $k = 0, 1, \dots, 2n-1$. The coefficient formula (16) gives

$$[q_0, \lambda * p_0]_{\mathcal{A}} = \boldsymbol{\alpha}_0 = [q_0, \lambda * p_0]_{\mathcal{B}}.$$

Hence $\boldsymbol{\mu}_1 = \boldsymbol{\alpha}_0 = \widehat{\boldsymbol{\mu}}_1$. Considering the induction assumptions $\boldsymbol{\mu}_k = \widehat{\boldsymbol{\mu}}_k$ for $k = 0, \dots, 2j-3$, we prove that $\boldsymbol{\mu}_{2j-2} = \widehat{\boldsymbol{\mu}}_{2j-2}$ and $\boldsymbol{\mu}_{2j-1} = \widehat{\boldsymbol{\mu}}_{2j-1}$, for $j = 2, \dots, n$. By the formula in (16) we get

$$[q_{j-1}, \lambda * p_{j-2}]_{\mathcal{A}} = \boldsymbol{\beta}_{j-1} = [q_{j-1}, \lambda * p_{j-2}]_{\mathcal{B}},$$

which we can rewrite as

$$\sum_{i=0}^{j-1} \sum_{k=0}^{j-2} \boldsymbol{\eta}_i^H \times \boldsymbol{\mu}_{i+k+1} \times \widehat{\boldsymbol{\eta}}_k = \sum_{i=0}^{j-1} \sum_{k=0}^{j-2} \boldsymbol{\eta}_i^H \times \widehat{\boldsymbol{\mu}}_{i+k+1} \times \widehat{\boldsymbol{\eta}}_k,$$

where $\boldsymbol{\eta}_i, \widehat{\boldsymbol{\eta}}_k \in \mathbb{C}^{M \times M}$ are the coefficients respectively of q_{j-1} and p_{j-2}. By the induction assumption we obtain

$$\boldsymbol{\eta}^H_{j-1} \times \boldsymbol{\mu}_{2j-2} \times \widehat{\boldsymbol{\eta}}_{j-2} = \boldsymbol{\eta}^H_{j-1} \times \widehat{\boldsymbol{\mu}}_{2j-2} \times \widehat{\boldsymbol{\eta}}_{j-2}.$$

The leading coefficients of the polynomials q_{2j-2} and p_{2j-2} are respectively $\boldsymbol{\eta}_{j-1} = 1$ and $\widehat{\boldsymbol{\eta}}_{j-2} = (\boldsymbol{\beta}_{j-2} \times \cdots \times \boldsymbol{\beta}_1)^{-1}$. Hence $\boldsymbol{\mu}_{2j-2} = \widehat{\boldsymbol{\mu}}_{2j-2}$. We conclude the proof repeating the same argument with the coefficient $\boldsymbol{\alpha}_{j-1}$ (16). □

Acknowledgements The authors want to thank Enikö Baligács and Christian Bonhomme (Laboratoire de chimie de la matière condensée de Paris, Sorbonne University) for providing the data from real-world applications used in Section 5. This work was supported by Charles University Research programs No. PRIMUS/21/SCI/009 and UNCE/SCI/023, and by the Magica project ANR-20-CE29-0007 funded by the French National Research Agency. The author M.R.-Z. is a member of the GNCS-INdAM group.

Data availability Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.

Declarations

Conflict of interest The authors declare no competing interests.

References

1. Autler, S.H., Townes, C.H.: Stark effect in rapidly varying fields. Phys. Rev. **100**, 703–722 (1955)
2. Bader, P., Iserles, A., Kropielnicka, K., Singh, P.: Efficient methods for linear Schrödinger equation in the semiclassical regime with time-dependent potential. Proc R Soc A Math Phys Eng Sci **472**, 20150733 (2016)
3. Baligács, E., Bonhomme, C.: https://github.com/BaligacsEni/TOMEexamples.git. Accessed 1st July 2022 (2022)
4. Benner, P., Cohen, A., Ohlberger, M., Willcox, K.: Model Reduction and Approximation: Theory and Algorithms. Computational Science and Engineering. SIAM, Philadelphia (2017)
5. Blanes, S.: High order structure preserving explicit methods for solving linear-quadratic optimal control problems. Numer. Algorithms **69**(2), 271–290 (2015)
6. Blanes, S., Casas, F.: A concise introduction to geometric numerical integration. CRC Press, Bocan Raton (2017)
7. Blanes, S., Casas, F., Oteo, J., Ros, J.: The Magnus expansion and some of its applications. Phys. Rep. **470**(5), 151–238 (2009)
8. Brezinski, C., Redivo-Zaglia, M., Sadok, H.: Avoiding breakdown and near-breakdown in Lanczos type algorithms. Numer. Algorithms **1**(3), 261–284 (1991)
9. Brezinski, C., Redivo Zaglia, M., Sadok, H.: A breakdown-free Lanczos type algorithm for solving linear systems. Numer. Math. **63**(1), 29–38 (1992)
10. Budd, C., Iserles, A., Nørsett, S.: On the solution of linear differential equations in Lie groups. Philos. Trans. R. Soc. London. Series A: Mathematical Phys. Eng. Sci. **357**(1754), 983–1019 (1999)
11. Cichocki, A., Lee, N., Oseledets, I., Phan, A.-H., Zhao, Q., Mandic, D.: Low-rank tensor networks for dimensionality reduction and large-scale optimization problems: Perspectives and challenges part 1. arXiv:1609.00893 (2016)
12. Cichocki, A., Mandic, D., Caiafa, C., Phan, A., Zhou, G., Zhao, Q., De Lathauwer, L.: Tensor decompositions for signal processing applications. IEEE Signal Processing Mag. (2013)
13. Cohen, D., Jahnke, T., Lorenz, K., Lubich, C.: Numerical integrators for highly oscillatory Hamiltonian systems: A review. In: Mielke, A. (ed.) Analysis, Modeling and Simulation of Multiscale Problems, pp. 553–576. Springer, Berlin (2006)
14. Corless, M., Frazho, A.: Linear Systems and Control: An Operator Perspective. Pure and Applied Mathematics. Marcel Dekker, New York (2003)

15. Degani, I., Schiff, J.: RCMS: Right correction Magnus series approach for oscillatory ODEs. J. Comput. Appl. Math. **193**(2), 413–436 (2006)
16. Draux, A.: Formal orthogonal polynomials revisited. Applic. Numer. Algorithms **11**(1), 143–158 (1996)
17. Dyson, F.J.: Divergence of perturbation theory in quantum electrodynamics. Phys. Rev. **85**(4), 631–632 (1952)
18. Feng, J., Yang, L.T., Zhang, R., Qiang, W., Chen, J.: Privacy preserving high-order Bi-Lanczos in cloud-fog computing for industrial applications. IEEE Transactions on Industrial Informatics, 1–1 (2020)
19. Freund, R.W., Gutknecht, M.H., Nachtigal, N.M.: An implementation of the look-ahead Lanczos algorithm for non-Hermitian matrices. SIAM J. Sci Comput. **14**(1), 137–158 (1993)
20. Frommer, A., Lund, K., Szyld, D.B.: Block Krylov subspace methods for functions of matrices. Electron. Trans. Numer Anal. **47**, 100–126 (2017)
21. Giscard, P.-L., Lui, K., Thwaite, S.J., Jaksch, D.: An exact formulation of the time-ordered exponential using path-sums. J. Math. Phys. **56**(5), 053503 (2015)
22. Giscard, P.-L., Pozza, S.: Lanczos-like algorithm for the time-ordered exponential: The $*$-inverse problem. Appl. Math. **65**(6), 807–827 (2020)
23. Giscard, P.-L., Pozza, S.: A Lanczos-like method for non-autonomous linear ordinary differential equations. arXiv:1909.03437 (2021)
24. Giscard, P.-L., Pozza, S.: Tridiagonalization of systems of coupled linear differential equations with variable coefficients by a Lanczos-like method. Linear Algebra Appl. **624**, 153–173 (2021)
25. Golub, G.H., Meurant, G.: Matrices, Moments and Quadrature with Applications. Princeton Ser. Appl. Math. Princeton University Press, Princeton (2010)
26. Guide, M.E., Ichi, A.E., Jbilou, K., Sadaka, R.: On tensor GMRES and Golub-Kahan methods via the T-product for color image processing. Electron. J. Linear Algebra **37**, 524–543 (2021)
27. Gutknecht, M.H.: A completed theory of the unsymmetric Lanczos process and related algorithms. I. SIAM J. Matrix Anal. Appl. **13**(2), 594–639 (1992)
28. Gutknecht, M.H.: A completed theory of the unsymmetric Lanczos process and related algorithms. II. SIAM J. Matrix Anal. Appl. **15**(1), 15–58 (1994)
29. Hafner, S., Spiess, H.-W.: Advanced solid-state NMR spectroscopy of strongly dipolar coupled spins under fast magic angle spinning. Concepts Magnetic Resonance **10**(2), 99–128 (1998)
30. Hochbruck, M., Lubich, C.: Exponential integrators for quantum-classical molecular dynamics. BIT Numer. Math. **39**(4), 620–645 (1999)
31. Hore, P.J.: NMR principles. In: Lindon, J.C. (ed.) Encyclopedia of Spectroscopy and Spectrometry, 2nd edn, pp. 1833–1840. Academic Press, Oxford (1999)
32. Hortaçsu, M.: Heun functions some of their applications in physics. Adv High Energy Phys **2018**, 8621573 (2018)
33. Huang, B., Xie, Y., Ma, C.: Krylov subspace methods to solve a class of tensor equations via the Einstein product. Numer. Linear Algebra Appl. **26**, 4 (2019)
34. Iserles, A.: On the global error of discretization methods for highly-oscillatory ordinary differential equations. BIT Numer. Math. **42**(3), 561–599 (2002)
35. Iserles, A.: On the method of Neumann series for highly oscillatory equations. BIT Numer. Math. **44**(3), 473–488 (2004)
36. Iserles, A., Munthe-Kaas, H.Z., Nørsett, S.P., Zanna, A.: Lie-group methods. Acta Numer. **9**, 215–365 (2000)
37. Kwakernaak, H., Sivan, R.: Linear Optimal Control Systems, vol. 1. Wiley-interscience, New York (1972)
38. Lauder, M., Knight, P., Greenland, P.: Pulse-shape effects in intense-field laser excitation of atoms. Opt Acta **33**(10), 1231–1252 (1986)
39. Levitt, M.H. Spin Dynamics: Basics of Nuclear Magnetic Resonance, 2nd edn. Wiley, Chichester (2008)
40. Magnus, W.: On the exponential solution of differential equations for a linear operator. Comm. Pure Appl. Math. **7**(4), 649–673 (1954)
41. Oseledets, I.: Tensor-train decomposition. SIAM J. Sci. Comput. **33**(5), 2295–2317 (2011)
42. Oseledets, I., Tyrtyshnikov, E.: TT-cross approximation for multidimensional arrays. Linear Algebra Appl. **432**(1), 70–88 (2010)
43. Parlett, B.N., Taylor, D.R., Liu, Z.A.: A look-ahead Lanczos algorithm for unsymmetric matrices. Math Comp. **44**(169), 105–124 (1985)

44. Pozza, S., Pranić, M.: The Gauss quadrature for general linear functionals, Lanczos algorithm, and minimal partial realization. Numer. Algorithms **88**, 647–678 (2021)
45. Pozza, S., Pranić, M.S., Strakoš, Z.: Gauss quadrature for quasi-definite linear functionals. IMA J. Numer Anal. **37**(3), 1468–1495 (2017)
46. Reichel, L., Ugwu, U.O.: Tensor Arnoldi-Tikhonov and GMRES-type methods for ill-posed problems with a t-product structure. Journal of Scientific Computing, 90(1) dec (2021)
47. Reid, W.T.: Riccati matrix differential equations and non-oscillation criteria for associated linear differential systems. Pacific J. Math. **13**(2), 665–685 (1963)
48. Ruymbeek, K., Meerbergen, K., Michiels, W.: Tensor-Krylov method for computing eigenvalues of parameter-dependent matrices. J. Comput. Appl Math. **408**, 113869 (2022)
49. Shirley, J.H.: Solution of the Schrödinger equation with a Hamiltonian periodic in time. Phys. Rev. **138**, B979–B987 (1965)
50. Smith, S.A., Palke, W.E., Gerig, J.T.: The Hamiltonians of NMR. Part I. Concepts in Magnetic Resonance **4**(2), 107–144 (1992)
51. Taylor, D.R.: Analysis of the Look Ahead Lanczos Algorithm. PhD thesis. University of, California, Berkeley (1982)
52. Wilkinson, J.H.: The Algebraic Eigenvalue Problem. Monographs on Numerical Analysis. The Clarendon Press Oxford University Press, New York (1988)
53. Xie, Q., Hai, W.: Analytical results for a monochromatically driven two-level system. Phys. Rev. A **82**, 032117 (2010)