TIME INTEGRATION OF RANK-CONSTRAINED TUCKER TENSORS[*]

CHRISTIAN LUBICH[†], BART VANDEREYCKEN[‡], AND HANNA WALACH[§]

Abstract. Dynamical low-rank approximation in the Tucker tensor format of given large time-dependent tensors and of tensor differential equations is the subject of this paper. In particular, a discrete time integration method for rank-constrained Tucker tensors is presented and analyzed. It extends the known projector-splitting integrator for dynamical low-rank approximation of matrices to Tucker tensors and is shown to inherit the same favorable properties. The integrator is based on iteratively applying the matrix projector-splitting integrator to tensor unfoldings but with inexact solution in a substep. It has the property that it reconstructs time-dependent Tucker tensors of the given rank exactly. The integrator is also shown to be robust to the presence of small singular values in the tensor unfoldings. Numerical examples with time-dependent problems from quantum physics and tensor optimization methods illustrate our theoretical results.

Key words. Tucker tensor format, tensor differential equation, dynamical low-rank approximation, projector-splitting integrator

AMS subject classifications. 15A03, 15A18, 15A69, 65L05, 65L20, 65L70

1. Introduction.

In this paper we propose and study a discrete method for approximating time-dependent tensors $A(t) \in \mathbb{R}^{n_1\times\cdots\times n_d}$ with $t_0 \le t \le T$ by tensors of a prescribed (low) multilinear rank. The tensors $A(t)$ are either given explicitly or are the unknown solution to a tensor differential equation

$$\dot{A}(t) = F(t, A(t)), \qquad A(t_0) = A^0, \tag{1.1}$$

where $\dot{A}(t) = \mathrm{d}A/\mathrm{d}t$. The approximation follows the setting of the dynamical low-rank approximation of [7], which yields a differential equation for the approximation $Y(t)$ to $A(t)$ on the manifold $\mathcal{M}$ of tensors of multilinear rank $\mathbf{r} = (r_1, \dots, r_d)$. As is known from [2], such tensors can be represented element-wise in the Tucker format (see, e.g., [8]) as follows:

$$y_{k_1,\dots,k_d}(t) = \sum_{\ell_1=1}^{r_1} \cdots \sum_{\ell_d=1}^{r_d} c_{\ell_1,\dots,\ell_d}(t)\, u^{(1)}_{k_1,\ell_1}(t) \cdots u^{(d)}_{k_d,\ell_d}(t), \tag{1.2}$$

with $k_i = 1, \dots, n_i$ for all modes $i = 1, \dots, d$. Using the multilinear product X (see, e.g., [8]), the relation (1.2) can be written more succinctly as

$$Y(t) = C(t) \bigtimes_{i=1}^{d} \mathbf{U}_i(t),$$

where $C(t) \in \mathbb{R}^{r_1\times\cdots\times r_d}$ is the time-dependent core tensor of full multilinear rank with entries $c_{\ell_1,\dots,\ell_d}(t)$, and $\mathbf{U}_i(t)$ is the mode-i time-dependent basis matrix of size $n_i \times r_i$ with entries $u^{(i)}_{k_i,\ell_i}(t)$.

[*]JANUARY 17, 2018

[†]Mathematisches Institut, Universität Tübingen, Auf der Morgenstelle 10, D–72076 Tübingen, Germany (lubich@na.uni-tuebingen.de)

[‡]Section of Mathematics, University of Geneva, Rue du Lièvre 2-4, 1211 Geneva, Switzerland (Bart.Vandereycken@unige.ch)

[§]Mathematisches Institut, Universität Tübingen, Auf der Morgenstelle 10, D–72076 Tübingen, Germany (walach@na.uni-tuebingen.de).

The differential equation for $Y(t) \in \mathcal{M}$ is obtained by projecting $F(t, Y(t))$ (or $\dot{A}(t)$ in the case of a given explicit time-dependent tensor $A(t)$) onto the tangent space $\mathcal{T}_{Y(t)}\mathcal{M}$ of $\mathcal{M}$ at the current approximation $Y(t) \in \mathcal{M}$:

$$\dot{Y}(t) = P(Y(t))F(t, Y(t)), \qquad Y(t_0) = Y^0 \in \mathcal{M}, \tag{1.3}$$

where $P(Y)\colon \mathbb{R}^{n_1\times\cdots\times n_d} \to \mathcal{T}_Y\mathcal{M}$ is the orthogonal projection, which can be given explicitly as an alternating sum of subprojections [7, 10]. The differential equation (1.3) needs to be solved numerically in virtually all applications. For example, in the context of molecular quantum dynamics, such an approach is taken in the multi-configuration time-dependent Hartree method (MCTDH) [13], where the multivariate wavefunction is approximated by a linear combination of products of univariate functions.

For the corresponding matrix problem (i.e., the particular case $d = 2$), a projector-splitting integrator with remarkable properties has been proposed in [11]. In particular, contrary to a direct integration of the arising projected differential equations, that integrator is robust to the presence of small singular values of the current approximation matrix; see [4, Thm. 2.1] (which is restated as Theorem 2.2 below). Such a situation commonly arises when the rank is chosen sufficiently large as to obtain an accurate approximation and the singular values have a decaying behaviour.

We outline the contributions and organization of the paper as follows:

The matrix projector-splitting integrator has been extended to Tucker tensors in [10] and to tensor trains (or matrix product states in the terminology of physics) in [12, 3]. In this paper we give a conceptually different derivation of an integrator for (1.3), based on the idea of an inexact solution of substeps within the matrix projector-splitting integrator applied to matricizations of (1.3). This derivation allows us to transfer the known favorable properties of the matrix integrator to the tensor case. We then show that the newly derived integrator is mathematically equivalent to the tensor projector-splitting integrator of [10], whose key properties of exactness and robustness are thus proven in the present paper. We mention that this integrator has meanwhile proved its robustness and efficiency in a first MCTDH implementation [6].

In Section 2, we briefly restate the matrix projector-splitting integrator and some of its properties. We then derive in Section 3 the Tucker tensor integrator in a recursive way from the matrix integrator. In Section 4, we show that the integrator reproduces the given matrix $A(t)$ if it is explicitly given and it is of rank $(r_1, \dots, r_d)$. This extends the exactness property of the integrator in the matrix case [11], which is fundamental for the error analysis of the matrix projector-splitting integrator in [4]. In Section 5, we extend the error analysis of [4] to the Tucker tensor case, which shows the robustness of the integrator in the presence of small singular values of its matricizations. In Section 6, we show that the integrator derived and studied here is mathematically equivalent to the projector-splitting integrator for Tucker tensors as proposed in [10]. Finally, in Section 7, we present numerical experiments that illustrate the behaviour of the integrator.

2. The matrix projector-splitting integrator.

In this section, we briefly restate the projector-splitting integrator from [11] for the matrix case, i.e., for $d = 2$. Recall that our aim is to numerically integrate the initial value problem (1.3) to obtain a low rank approximation of (1.1). To this end, we will make use of the SVD-like representation

$$\mathbf{Y}(t) = \mathbf{U}(t)\mathbf{S}(t)\,\mathbf{V}(t)^T$$

of the rank-r approximation matrix $\mathbf{Y}(t) \in \mathcal{M}$, where $\mathbf{U}(t)$ and $\mathbf{V}(t)$ are basis matrices of size $n_1 \times r$ and $n_2 \times r$ for the first and second mode, respectively. The invertible matrix $\mathbf{S}(t) \in \mathbb{R}^{r\times r}$ has the same nonzero singular values as $\mathbf{Y}(t)$, but unlike the SVD, $\mathbf{S}(t)$ is not assumed to be diagonal.

The projector-splitting integrator updates the factors $\mathbf{U}, \mathbf{S}, \mathbf{V}$, starting from the initial value $\mathbf{Y}^0 = \mathbf{U}^0\,\mathbf{S}^0\,\mathbf{V}^{0,T}$. Let $F : \mathbb{R} \times \mathbb{R}^{n_1\times n_2} \to \mathbb{R}^{n_1\times n_2}$. Then one time step from t_0 to $t_1 = t_0 + h$ proceeds as follows:

1. **K-step**: Update $\mathbf{U}^0 \to \mathbf{U}^1$, $\mathbf{S}^0 \to \widehat{\mathbf{S}}^1$.
   Integrate to $t = t_1$ the differential equation

   $$\dot{\mathbf{K}}(t) = F(t, \mathbf{K}(t)\,\mathbf{V}^{0,T})\,\mathbf{V}^0, \qquad \mathbf{K}(t_0) = \mathbf{U}^0\,\mathbf{S}^0 \tag{2.1}$$

   and perform a QR factorization $\mathbf{K}(t_1) = \mathbf{U}^1\,\mathbf{S}^1$ to orthonormalise the columns of $\mathbf{K}(t_1)$. This yields $\mathbf{U}^1$ as the final approximation of the basis matrix $\mathbf{U}(t)$ at $t = t_1$, and the temporary update $\widehat{\mathbf{S}}^1$.
2. **S-step**: Update $\widehat{\mathbf{S}}^1 \to \widetilde{\mathbf{S}}^0$.
   Integrate to $t = t_1$ the differential equation

   $$\dot{\mathbf{S}}(t) = -\,\mathbf{U}^{1,T}\,F(t, \mathbf{U}^1\,\mathbf{S}(t)\,\mathbf{V}^{0,T})\,\mathbf{V}^0, \qquad \mathbf{S}(t_0) = \widehat{\mathbf{S}}^1 \tag{2.2}$$

   This yields the temporary update $\widetilde{\mathbf{S}}^0 = \mathbf{S}(t_1)$.
3. **L-step**: Update $\mathbf{V}^0 \to \mathbf{V}^1$, $\widetilde{\mathbf{S}}^0 \to \mathbf{S}^1$.
   Integrate to $t = t_1$ the differential equation

   $$\dot{\mathbf{L}}^T(t) = \mathbf{U}^{1,T}\,F(t, \mathbf{U}^1\,\mathbf{L}(t)^T), \qquad \mathbf{L}^T(t_0) = \widetilde{\mathbf{S}}^0\,\mathbf{V}^{0,T} \tag{2.3}$$

   and perform a QR factorization $\mathbf{L}(t_1) = \mathbf{V}^1\,\mathbf{S}^{1,T}$. This yields the final approximations $\mathbf{V}^1$ and $\mathbf{S}^1$.

Merging the computed factors results in the rank-r approximation matrix

$$\mathbf{Y}^1 = \mathbf{U}^1\,\mathbf{S}^1\,\mathbf{V}^{1,T} \tag{2.4}$$

after one time step. To continue in time, we take the factorized matrix $\mathbf{Y}^1$ as initial value for the next time step and apply this scheme again. This way we obtain a first-order splitting method for (1.3).

The matrix projector-splitting integrator has a remarkable exactness property.

THEOREM 2.1. [11, Thm. 4.1] *Let $\boldsymbol{A}(t) \in \mathbb{R}^{n_1\times n_2}$ with* rank $\boldsymbol{A}(t) \le r$ *for all t and $\boldsymbol{A}(t_0) = \boldsymbol{Y}^0$. Further, let $\boldsymbol{V}(t_1)^T\,\boldsymbol{V}(t_0)$ be invertible. Then, the splitting integrator described above (with $F(t, \boldsymbol{Y}) = \dot{\boldsymbol{A}}(t)$) reproduces the exact solution: $\boldsymbol{Y}^1 = \boldsymbol{A}(t_1)$.*

Moreover, the integrator is robust to the presence of small singular values in the solution or its approximation.

THEOREM 2.2. [4, Thm. 2.1] *Let $\boldsymbol{A}(t)$ denote the solution of (1.1) on $[t_0, T]$ in case of $d = 2$ and let $\mathcal{M}$ be the manifold of rank r matrices in $\mathbb{R}^{n_1\times n_2}$. Suppose the following assumptions are satisfied with $\|\cdot\|$ the Euclidean norm.*

(a) $F(t, \boldsymbol{Y})$ is Lipschitz continuous and bounded for all $\boldsymbol{Y}, \widetilde{\boldsymbol{Y}} \in \mathbb{R}^{n_1\times n_2}$:

$$\|F(t, \boldsymbol{Y}) - F(t, \widetilde{\boldsymbol{Y}})\| \le L\|\,\boldsymbol{Y} - \widetilde{\boldsymbol{Y}}\|, \qquad \|F(t, \boldsymbol{Y})\| \le B.$$

(b) $F(t, \boldsymbol{Y})$ can be decomposed into a tangential part and a small perturbation:

$$F(t, \boldsymbol{Y}) = M(t, \boldsymbol{Y}) + R(t, \boldsymbol{Y}),$$

$$M(t,\boldsymbol{Y}) \in \mathcal{T}_{\boldsymbol{Y}}\mathcal{M}, \quad \|R(t,\boldsymbol{Y})\| \le \varepsilon,$$

for all $\boldsymbol{Y} \in \mathcal{M}$ in a neighborhood of $\boldsymbol{A}(t)$ and for all $t \in [t_0, T]$.
(c) The initial value $\boldsymbol{A}(t_0)$ for (1.1) has rank r.
Then, the error of the splitting integrator described above after n steps with step size $h > 0$ satisfies for all $t_n = t_0 + nh \le T$

$$\| \boldsymbol{Y}_n - \boldsymbol{A}(t_n)\| \le c_1 h + c_2 \varepsilon,$$

where the constants c_1, c_2 only depend on $L, B, T - t_0$. In particular, the constants are independent of singular values of the exact or approximate solution matrix.

In general, the differential equations in the substeps (2.1)–(2.3) have to be solved numerically, e.g., by a Runge–Kutta method. In the case when $F(t,\mathbf{Y}) = \dot{\mathbf{A}}(t)$ for explicitly given matrices $\mathbf{A}(t)$, the integrator works with the increment $\mathbf{A}(t_1) - \mathbf{A}(t_0)$ and so the substeps can be solved directly. If, however, we apply a numerical integrator, then instead of $\mathbf{Y}_n$, we compute a perturbed matrix $\widetilde{\mathbf{Y}}_n$. Assuming that the arising local errors in the substeps are bounded by $h\eta$, the error bound of (2.2) after n time steps is given by

$$\|\widetilde{\mathbf{Y}}_n - \mathbf{A}(t_n)\| \le c_1 h + c_2 \varepsilon + c_3 \eta, \tag{2.5}$$

where c_3 also only depends on $L, B, T - t_0$; see Section 2.6.3 in [4].

These exactness and robustness properties will be extended to the Tucker integrator in Sections 4 and 5, respectively. But first, we present the integration scheme for tensors in the Tucker format.

3. The nested Tucker integrator.

3.1. Derivation of the integrator. To find a low-rank approximation for (1.1) in the case of Tucker tensors of general dimension d, we will now extend the matrix projector-splitting integrator to tensors. To this end, we will need to transfer tensors into a matrix setting by considering their matricizations. In particular, we denote by

$$\mathbf{Mat}_i(X) = \mathbf{X} \in \mathbb{R}^{n_i \times n_1 \cdots n_{i-1} n_{i+1} \cdots n_d}$$

the i-mode matricization of a tensor $X \in \mathbb{R}^{n_1 \times \cdots \times n_d}$. It arranges the mode-$i$ fibres of X to be the rows of the resulting matrix $\mathbf{X}$. The reversal of the i-mode matricization is called tensorization, which we denote as

$$\mathrm{Ten}_i(\mathbf{X}) = X \in \mathbb{R}^{n_1 \times \cdots \times n_d}.$$

We begin by matricizing the *tensor* ODE (1.1) in mode 1:

$$\mathbf{Mat}_1\big(\dot{A}(t)\big) = \mathbf{Mat}_1\big(F(t, A(t))\big), \qquad \mathbf{Mat}_1\big(A(t_0)\big) = \mathbf{Mat}_1\big(A^0\big). \tag{3.1}$$

This will allow us to formally apply the matrix projector-splitting integrator to this *matrix* ODE where the initial value $\mathbf{Mat}_1\big(A^0\big)$ is approximated by $\mathbf{Mat}_1\big(Y^0\big)$ with $Y^0 \in \mathcal{M}$. Since Y^0 has multilinear rank $(r_1, \dots, r_d)$, it satisfies the decomposition

$$Y^0 = C^0 \bigtimes_{i=1}^{d} \mathbf{U}_i^0$$

with $C^0 \in \mathbb{R}^{r_1\times\cdots\times r_d}$ and $\mathbf{U}_i^0 \in \mathbb{R}^{n_i\times r_i}$. Since we are dealing with the first step of the algorithm, let us denote the initial value as $Y_1^0 := Y^0$ with core tensor as $C_1^0 := C^0$. By performing the QR decomposition

$$\mathbf{Mat}_1(C_1^0)^T = \mathbf{Q}_1^0\,\mathbf{S}_1^{0,T} \in \mathbb{R}^{r_2\cdots r_d\times r_1}$$

and denoting

$$\mathbf{V}_1^{0,T} = \mathbf{Q}_1^{0,T} \bigotimes_{i=2}^{d} \mathbf{U}_i^{0,T} \in \mathbb{R}^{r_1\times n_2\cdots n_d}, \tag{3.2}$$

we obtain the necessary SVD-like representation of the initial value of (3.1) as

$$\mathbf{Mat}_1(Y_1^0) = \mathbf{U}_1^0\,\mathbf{Mat}_1(C_1^0)\bigotimes_{i=2}^{d}\mathbf{U}_i^{0,T} = \mathbf{U}_1^0\,\mathbf{S}_1^0\,\mathbf{V}_1^{0,T}.$$

Now, we are in the situation to apply the matrix projector-splitting integrator to (3.1):

1. **K-step**: Update $\mathbf{U}_1^0 \to \mathbf{U}_1^1$, $\mathbf{S}_1^0 \to \widehat{\mathbf{S}}_1^1$.
2. **S-step**: Update $\widehat{\mathbf{S}}_1^1 \to \widetilde{\mathbf{S}}_1^0$.
3. **L-step**: Update $\mathbf{V}_1^0 \to \mathbf{V}_1^1$, $\widetilde{\mathbf{S}}_1^0 \to \mathbf{S}_1^1$ by solving approximately

$$\begin{aligned} \dot{\mathbf{L}}_1^T(t) &= \mathbf{U}_1^{1,T}\,\mathbf{Mat}_1\left(F(t,\mathrm{Ten}_1(\mathbf{U}_1^1\,\mathbf{L}_1^T(t)))\right), \\ \mathbf{L}_1^T(t_0) &= \mathbf{L}_1^{0,T} = \widetilde{\mathbf{S}}_1^0\,\mathbf{V}_1^{0,T}, \end{aligned} \tag{3.3}$$

with $\mathbf{L}_1 \in \mathbb{R}^{r_1\times n_2\cdots n_d}$ and the QR factorization $\mathbf{L}_1(t_1) = \mathbf{V}_1^1\,\mathbf{S}_1^{1,T}$.

The K- and S-steps can be calculated as in the matrix case, i.e., solving (2.1) and (2.2) but applied to (3.1). However, we do not solve the matrix differential equation in the L-step directly since it is defined for a prohibitively large $\mathbf{L}_1$. More importantly, it would also not lead to an approximation for $Y(t_1)$ of multilinear rank $(r_1,\ldots,r_d)$ since the exact L-step above only reduces the rank of the first mode. Instead, we perform a low-rank approximation for (3.3) by applying the matrix projector-splitting integrator again to a reshaped version of it.

Defining $Y_2(t) = \mathrm{Ten}_1(\mathbf{L}_1^T(t)) \in \mathbb{R}^{r_1\times n_2\times\cdots\times n_d}$, we first retensorize (3.3) as

$$\dot{Y}_2(t) = F(t, Y_2(t)\times_1\mathbf{U}_1^1)\times_1\mathbf{U}_1^{1,T}, \qquad Y_2(t_0) = \mathrm{Ten}_1(\mathbf{L}_1^{0,T}). \tag{3.4}$$

Observe that $Y_2(t)$ is usually of significantly smaller size than $Y(t)$ since typically $r_1 \ll n_1$. Next, we unfold (3.4) in the second mode. For simplicity of notation, we denote this 2-mode unfolding of Y_2 by $\mathbf{Y}_{[2]} := \mathbf{Mat}_2(Y_2) \in \mathbb{R}^{n_2\times r_1 n_3\cdots n_d}$. This gives the *matrix* differential equation

$$\dot{\mathbf{Y}}_{[2]}(t) = \mathbf{Mat}_2\Big(F(t,\mathrm{Ten}_2(\mathbf{Y}_{[2]}(t))\times_1\mathbf{U}_1^1)\times_1\mathbf{U}_1^{1,T}\Big)$$

and, using (3.3) and (3.2), the initial value

$$\mathbf{Y}_{[2]}(t_0) = \mathbf{Mat}_2\left(\mathrm{Ten}_1(\mathbf{L}_1^{0,T})\right) = \mathbf{Mat}_2\Big(\mathrm{Ten}_1(\widetilde{\mathbf{S}}_1^0\,\mathbf{Q}_1^{0,T}\bigotimes_{i=2}^{d}\mathbf{U}_i^{0,T})\Big).$$

Defining $C_2^0 = \mathrm{Ten}_1\big(\widetilde{\mathbf{S}}_1^0\,\mathbf{Q}_1^{0,T}\big) \in \mathbb{R}^{r_1\times\cdots\times r_d}$ and $\mathbf{C}_{[2]}^0 = \mathbf{Mat}_2(C_2^0)$, we also have

$$\mathbf{Y}_{[2]}(t_0) = \mathbf{Mat}_2\big(C_2^0\bigtimes_{i=2}^{d}\mathbf{U}_i^0\big) = \mathbf{U}_2^0\,\mathbf{C}_{[2]}^0\Big(\mathbf{I}_{r_1}\otimes\bigotimes_{i=3}^{d}\mathbf{U}_i^{0,T}\Big).$$

As before we have to determine the SVD-like representation of $\mathbf{Y}_{[2]}(t_0)$. To this end, compute the QR factorization $\mathbf{C}_{[2]}^{0,T} = \mathbf{Q}_2^0\,\mathbf{S}_2^{0,T}$. We then obtain the desired result as $\mathbf{Y}_{[2]}(t_0) = \mathbf{U}_2^0\,\mathbf{S}_2^0\,\mathbf{V}_2^{0,T}$ with $\mathbf{V}_2^{0,T} = \mathbf{Q}_2^{0,T}\Big(\mathbf{I}_{r_1} \otimes \bigotimes_{i=3}^d \mathbf{U}_i^{0,T}\Big) \in \mathbb{R}^{r_2 \times r_1 n_3 \cdots n_d}$.

Now that we have set up the matrix problem again, we can apply the matrix projector-splitting integrator to $\dot{\mathbf{Y}}_{[2]}(t)$.

1. **K-step**: Update $\mathbf{U}_2^0 \to \mathbf{U}_2^1$, $\mathbf{S}_2^0 \to \widehat{\mathbf{S}}_2^1$
2. **S-step**: Update $\widehat{\mathbf{S}}_2^1 \to \widetilde{\mathbf{S}}_2^0$
3. **L-step**: Update $\mathbf{V}_2^0 \to \mathbf{V}_2^1$, $\widetilde{\mathbf{S}}_2^0 \to \mathbf{S}_2^1$ by solving approximately

$$\begin{aligned}\dot{\mathbf{L}}_2^T(t) &= \mathbf{U}_2^{1,T}\,\mathbf{Mat}_2\Big(F(t, \mathrm{Ten}_2(\mathbf{U}_2^1\,\mathbf{L}_2^T(t)) \times_1 \mathbf{U}_1^1) \times_1 \mathbf{U}_1^{1,T}\Big),\\ \mathbf{L}_2^T(t_0) &= \mathbf{L}_2^{0,T} = \widetilde{\mathbf{S}}_2^0\,\mathbf{V}_2^{0,T},\end{aligned}$$

with $\mathbf{L}_2 \in \mathbb{R}^{r_2 \times r_1 n_3 \cdots n_d}$ and the QR factorization $\mathbf{L}_2(t_1) = \mathbf{V}_2^1\,\mathbf{S}_2^{1,T}$.

We continue recursively with solving the L-step approximately in each iteration step of the integrator. Generalising the pattern for modes 1 and 2 from above to general i, this requires us to find $Y_i(t) \in \mathbb{R}^{r_1 \times \cdots \times r_{i-1} \times n_i \times \cdots \times n_d}$ that satisfies the ODE

$$\dot{Y}_i(t) = F\Big(t, Y_i(t) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^1\Big) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^{1,T}, \qquad Y_i(t_0) = \mathrm{Ten}_{i-1}\big(\mathbf{L}_{i-1}^{0,T}\big). \tag{3.5}$$

The K- and S-steps for mode $i-1$ calculate, in particular, the matrices $\widetilde{\mathbf{S}}_{i-1}^0$ and $\mathbf{Q}_{i-1}^0$. This implies that the initial value in the above ODE is available as

$$\mathbf{L}_{i-1}^{0,T} = \widetilde{\mathbf{S}}_{i-1}^0\,\mathbf{Q}_{i-1}^{0,T}\Big(\bigotimes_{k=1}^{i-2} \mathbf{I}_{r_k} \otimes \bigotimes_{k=i}^{d} \mathbf{U}_k^{0,T}\Big) \in \mathbb{R}^{r_{i-1} \times r_1 \cdots r_{i-2} n_i \cdots n_d}.$$

To obtain a suitable matrix version of (3.5), we unfold it in mode i and define $\mathbf{Y}_{[i]} = \mathbf{Mat}_i(Y_i) \in \mathbb{R}^{n_i \times r_1 \cdots r_{i-1} n_{i+1} \cdots n_d}$. This gives

$$\begin{aligned}\dot{\mathbf{Y}}_{[i]}(t) &= \mathbf{Mat}_i\Big(F\Big(t, \mathrm{Ten}_i\big(\mathbf{Y}_{[i]}(t)\big) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^1\Big) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^{1,T}\Big),\\ \mathbf{Y}_{[i]}^0 &= \mathbf{Mat}_i\Big(C_i^0 \bigtimes_{k=1}^{i-1} \mathbf{I}_{r_k} \bigtimes_{k=i}^{d} \mathbf{U}_k^0\Big) = \mathbf{U}_i^0\,\mathbf{S}_i^0\,\mathbf{V}_i^{0,T}\end{aligned} \tag{3.6}$$

with $C_i^0 = \mathrm{Ten}_{i-1}\big(\widetilde{\mathbf{S}}_{i-1}^0\,\mathbf{Q}_{i-1}^{0,T}\big) \in \mathbb{R}^{r_1 \times \cdots \times r_d}$ and the QR decomposition

$$\mathbf{Mat}_i(C_i^0)^T = \mathbf{Q}_i^0\,\mathbf{S}_i^{0,T} \in \mathbb{R}^{r_1 \cdots r_{i-1} r_{i+1} \cdots r_d \times r_i}.$$

In addition, we have also used

$$\mathbf{V}_i^{0,T} = \mathbf{Q}_i^{0,T}\Big(\bigotimes_{k=1}^{i-1} \mathbf{I}_{r_k} \otimes \bigotimes_{k=i+1}^{d} \mathbf{U}_k^{0,T}\Big) \in \mathbb{R}^{r_i \times r_1 \cdots r_{i-1} n_{i+1} \cdots n_d}. \tag{3.7}$$

In this way, we can indeed apply the K- and S-steps of the matrix projector-splitting to (3.6). The L-step is recursively solving

$$\begin{aligned}\dot{\mathbf{L}}_i^T(t) &= \mathbf{U}_i^{1,T}\,\mathbf{Mat}_i\Big(F\Big(t, \mathrm{Ten}_i\big(\mathbf{U}_i^1\,\mathbf{L}_i^T(t)\big) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^1\Big) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^{1,T}\Big),\\ \mathbf{L}_i^T(t_0) &= \mathbf{L}_i^{0,T} = \widetilde{\mathbf{S}}_i^0\,\mathbf{V}_i^{0,T}\,.\end{aligned} \tag{3.8}$$

with the scheme we just explained. The recursion ends at $i = d$ since then the L-step,

$$\begin{aligned} \dot{\mathbf{L}}_d^T &= \mathbf{U}_d^{1,T}\,\mathbf{Mat}_d\Big(F\Big(t, \mathrm{Ten}_d\big(\mathbf{U}_d^1\,\mathbf{L}_d^T\big)\bigtimes_{i=1}^{d-1}\mathbf{U}_i^1\Big)\bigtimes_{i=1}^{d-1}\mathbf{U}_i^{1,T}\Big), \\ \mathbf{L}_d^T(t_0) &= \mathbf{L}_d^{0,T} = \widetilde{\mathbf{S}}_d^0\,\mathbf{V}_d^{0,T}, \end{aligned} \tag{3.9}$$

can then be solved explicitly for $\mathbf{L}_d^T(t_1) \in \mathbb{R}^{r_d \times r_1 \cdots r_{d-1}}$. Observe that this means that $\mathbf{L}_d(t)$ actually corresponds to the update of the core tensor $C(t)$ itself. Hence, with such an explicit L step we have calculated the final update $\mathrm{Ten}_d(\mathbf{L}_d(t_1)) = C^1$.

The scheme from above operates on tensors $Y_i(t)$ that consecutively get smaller for $i = 1, 2, \ldots, d$. However, we can also interpret it as computing an approximation Y^1 for the Tucker tensor $Y(t_1) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ in (1.3). In particular, we have

$$Y^1 = \mathrm{Ten}_1(\mathbf{U}_1^1\,\mathbf{L}_1^T(t_1)) = \mathrm{Ten}_1\left(\mathbf{U}_1^1\,\mathbf{Mat}_1(Y_2(t_1))\right) = Y_2(t_1) \times_1 \mathbf{U}_1^1$$

with $\mathbf{L}_1(t_1)$ the approximate solution of (3.3) obtained using $Y_2(t_1)$ in (3.4). In turn, $Y_2(t_1)$ is solved similarly using $Y_3(t)$:

$$Y_2(t_1) = \mathrm{Ten}_2(\mathbf{U}_2^1\,\mathbf{L}_2^T(t_1)) = Y_3(t_1) \times_2 \mathbf{U}_2^1\,.$$

Hence, continuing recursively for all modes, we obtain

$$\begin{aligned} Y^1 &= Y_2(t_1) \times_1 \mathbf{U}_1^1 = Y_3(t_1) \times_2 \mathbf{U}_2^1 \times_1 \mathbf{U}_1^1 \\ &= \cdots = Y_{i+1}^1 \bigtimes_{k=1}^{i} \mathbf{U}_k^1 = C^1 \bigtimes_{k=1}^{d} \mathbf{U}_k^1\,. \end{aligned}$$

3.2. Practical algorithm. As explained above, the nested Tucker integrator follows the scheme of recursively applying the matrix projector-splitting integrator with solving the first two steps, but performing a low-rank approximation for the third substep in each mode. The implementation of this integration scheme is straightforward and results in Alg. 3.1. It simply updates the basis matrices $\mathbf{U}_i$ in the K-step and the auxiliary matrix $\mathbf{S}_i$ in the S-step for each mode. Quite remarkably, in the approximate L-step it suffices to only update the core tensor C^0. This also reduces the size of the matrix differential equation that has to be solved for the next mode. For computational efficiency, we have written the operations using multilinear products. For example, line 4 is equivalent to (3.7).

The differential equations for $\mathbf{K}, \mathbf{S}, \mathbf{L}$ that need to be solved during the integration scheme, can be solved approximately, e.g., by a Runge–Kutta method. In the case, where $F(t, Y)$ is solution-independent and solely given by a tensor $A(t) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, those differential equations can be solved directly.

The integration scheme in Alg. 3.1 consists of recursively applying the matrix projector-splitting integrator. Since we do not solve the full matrix scheme, but rather the first two steps in order to update $\mathbf{U}_i$ and $\mathbf{S}_i$ for all modes $i = 1, \ldots, d$, it is a nested matrix projector-splitting integrator for Tucker tensors, or in short, the nested Tucker integrator.

Algorithm 3.1 One time step of the nested Tucker integrator

```
Data: Tucker tensor $Y^0 = C^0 \bigtimes_{i=1}^d \mathbf{U}_i^0$, $F(t,Y)$, $t_0$, $t_1$
Result: Tucker tensor $Y^1 = C^1 \bigtimes_{i=1}^d \mathbf{U}_i^1$
begin
    for $i = 1$ to $d$ do
        compute QR factorization $\mathbf{Mat}_i(C^0)^T = \mathbf{Q}_i^0\, \mathbf{S}_i^{0,T}$
        set $\mathbf{V}_i^{0,T} = \mathbf{Mat}_i\Big(\mathrm{Ten}_i(\mathbf{Q}_i^{0,T}) \bigtimes_{l=i+1}^{d} \mathbf{U}_l^{0,T}\Big)$
        set $\mathbf{K}_i^0 = \mathbf{U}_i^0\, \mathbf{S}_i^0$
        set $\mathbf{Y}_{[i]}^{+}(t) = \mathbf{K}_i(t)\, \mathbf{V}_i^{0,T}$
        solve $\dot{\mathbf{K}}_i(t) = \mathbf{Mat}_i\Big(F\big(t, \mathrm{Ten}_i(\mathbf{Y}_{[i]}^{+}) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^1\big) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^{1,T}\Big)\, \mathbf{V}_i^0$,
            with initial value $\mathbf{K}_i(t_0) = \mathbf{K}_i^0$ and return $\mathbf{K}_i^1 = \mathbf{K}_i(t_1)$
        compute QR factorization $\mathbf{K}_i^1 = \mathbf{U}_i^1\, \widehat{\mathbf{S}}_i^1$
        set $\mathbf{Y}_{[i]}^{-}(t) = \mathbf{U}_i^1\, \mathbf{S}_i(t)\, \mathbf{V}_i^{0,T}$
        solve $\dot{\mathbf{S}}_i(t) = -\,\mathbf{U}_i^{1,T}\, \mathbf{Mat}_i\Big(F\big(t, \mathrm{Ten}_i(\mathbf{Y}_{[i]}^{-}) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^1\big) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^{1,T}\Big)\, \mathbf{V}_i^0$,
            with initial value $\mathbf{S}_i(t_0) = \widehat{\mathbf{S}}_i^1$ and return $\widetilde{\mathbf{S}}_i^0 = \mathbf{S}_i(t_1)$
        set $C^0 = \mathrm{Ten}_i(\widetilde{\mathbf{S}}_i^0\, \mathbf{Q}_i^{0,T})$
    set $\mathbf{L}^{0,T} = \mathbf{Mat}_d(C^0)$
    solve $\dot{\mathbf{L}}^T(t) = \mathbf{U}_d^{1,T}\, \mathbf{Mat}_d\Big(F\big(t, \mathrm{Ten}_d(\mathbf{U}_d^1\, \mathbf{L}(t)^T) \bigtimes_{k=1}^{d-1} \mathbf{U}_k^1\big) \bigtimes_{k=1}^{d-1} \mathbf{U}_k^{1,T}\Big)$,
        with initial value $\mathbf{L}^T(t_0) = \mathbf{L}^{0,T}$ and return $\mathbf{L}^{1,T} = \mathbf{L}^T(t_1)$
    set $C^1 = \mathrm{Ten}_d(\mathbf{L}^{1,T})$
    set $Y^1 = C^1 \bigtimes_{i=1}^{d} \mathbf{U}_i^1$
```

4. An exactness property of the integrator. Let $\mathcal{M} \subset \mathbb{R}^{n_1 \times \cdots \times n_d}$ be the manifold of tensors with multilinear rank $(r_1, \dots, r_d)$. Suppose that $A(t) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ is given explicitly, hence, we formally have $F(t,Y) = \dot{A}(t)$ in (1.1) and (1.3). In addition, we assume that $A(t) \in \mathcal{M}$ for $t_0 \le t \le T$. Our aim in this section is to prove that Alg. 3.1, the nested Tucker integrator, is in that case exact. In other words, Alg. 3.1 solves the initial value problem (1.3) exactly even though it is a discrete time stepping method. As mentioned in Theorem 2.1, the projector-splitting integrator for matrices already has this property but it does not hold for more standard integrators on $\mathcal{M}$, like the projected Runge–Kutta methods in [5].

Since $A(t) \in \mathcal{M}$ for all t, we can write its i-mode matricization as

$$\mathbf{Mat}_i(A(t)) = \mathbf{U}_i(t)\mathbf{S}_i(t)\, \mathbf{W}_i(t)^T, \tag{4.1}$$

where $\mathbf{U}_i(t) \in \mathbb{R}^{n_i \times r_i}$ and $\mathbf{W}_i(t) \in \mathbb{R}^{n_1 \cdots n_{i-1} \cdot n_{i+1} \cdots n_d \times r_i}$ have orthonormal columns and $\mathbf{S}_i(t) \in \mathbb{R}^{r_i \times r_i}$ for all $i = 1, \dots, d$. With this SVD-like representation we can state and prove the following exactness result.

THEOREM 4.1. *Let $A(t)$ be of multilinear rank $(r_1,\ldots,r_d)$ for all $t\in(t_0,t_1)$ and let $Y(t_0)=A(t_0)$. Further, let $\boldsymbol{W}_i(t_1)^T\,\boldsymbol{W}_i(t_0)$ be invertible for all $i=2,\ldots,d$. Then, Algorithm 3.1 for $F(t,Y)=\dot{A}(t)$ reproduces the exact solution: $Y^1=A(t_1)$.*

Proof. Recall that the nested Tucker integrator in Alg. 3.1 is designed to approximately solve the initial value subproblems (see (3.6))

$$\dot{\mathbf{Y}}_{[i]}(t)=\mathbf{Mat}_i\Big(\dot{A}(t)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,T}\Big),\qquad \mathbf{Y}_{[i]}(t_0)=\mathbf{Y}_{[i]}^0=\mathbf{U}_i^0\,\mathbf{S}_i^0\,\mathbf{V}_i^{0,T},\tag{4.2}$$

where $\mathrm{Ten}_i(\mathbf{V}_i^{0,T})=\mathrm{Ten}_i(\mathbf{Q}_i^{0,T})\bigtimes_{l=i+1}^{d}\mathbf{U}_l^0\in\mathbb{R}^{r_1\times\cdots\times r_i\times n_{i+1}\times\cdots\times n_d}$ for each mode $i=1,\ldots,d$. In addition, the tensorized result $Y_i^1=\mathrm{Ten}_i(\mathbf{Y}_{[i]}(t_1))$ after one time step is in the low-rank manifold $\mathcal{M}_i:=\{Y_i\in\mathbb{R}^{r_1\times\cdots\times r_{i-1}\times n_i\times\cdots\times n_d}:\operatorname{rank}\mathbf{Mat}_i(Y_i)=r_i\}$.

In the first part of the proof, we show that the initial value for (4.2) can be written in terms of $A(t_0)$:

$$\mathbf{Y}_{[i]}(t_0)=\mathbf{Mat}_i\Big(A(t_0)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,T}\Big).\tag{4.3}$$

This ensures that $\mathbf{Y}_{[i]}^0$ has rank r_i. With this initial value, the exact solution of (4.2) has rank r_i as well, since $A(t)$ is assumed to have multilinear rank $(r_1,\ldots,r_d)$:

$$\begin{aligned}\mathbf{Y}_{[i]}(t)&=\mathbf{Y}_{[i]}(t_0)+\int_{t_0}^{t}\dot{\mathbf{Y}}_{[i]}(s)ds\\&=\mathbf{Mat}_i\Big(A(t_0)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,T}\Big)+\mathbf{Mat}_i\Big(\big(A(t)-A(t_0)\big)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,T}\Big)\\&=\mathbf{Mat}_i\Big(A(t)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,T}\Big)\\&=\mathbf{U}_i(t)\mathbf{S}_i(t)\,\mathbf{V}_i(t)^T,\end{aligned}$$

where we use the decomposition (4.1) and set

$$\mathbf{V}_i(t)=(\mathbf{U}_1^1\otimes\cdots\otimes\mathbf{U}_{i-1}^1\otimes\mathbf{I}_i\otimes\cdots\otimes\mathbf{I}_d)\;\mathbf{W}_i(t)\tag{4.4}$$

To show (4.3), we use an induction argument. With the abbreviation $\Delta A=A(t_1)-A(t_0)$ we have

$$\begin{aligned}\mathbf{U}_{i-1}^1\,\mathbf{L}_{i-1}^{0,T}&=\mathbf{U}_{i-1}^1\,\widehat{\mathbf{S}}_{i-1}^1\,\mathbf{V}_{i-1}^{0,T}-\mathbf{U}_{i-1}^1\,\mathbf{U}_{i-1}^{1,T}\,\mathbf{Mat}_{i-1}\Big(\Delta A\bigtimes_{k=1}^{i-2}\mathbf{U}_k^{1,T}\Big)\,\mathbf{V}_{i-1}^0\,\mathbf{V}_{i-1}^{0,T}\\&=\mathbf{U}_{i-1}^0\,\mathbf{S}_{i-1}^0\,\mathbf{V}_{i-1}^{0,T}+\mathbf{Mat}_{i-1}\Big(\Delta A\bigtimes_{k=1}^{i-2}\mathbf{U}_k^{1,T}\Big)\,\mathbf{V}_{i-1}^0\,\mathbf{V}_{i-1}^{0,T}\\&\qquad-\mathbf{U}_{i-1}^1\,\mathbf{U}_{i-1}^{1,T}\,\mathbf{Mat}_{i-1}\Big(\Delta A\bigtimes_{k=1}^{i-2}\mathbf{U}_k^{1,T}\Big)\,\mathbf{V}_{i-1}^0\,\mathbf{V}_{i-1}^{0,T}\\&=\mathbf{Mat}_{i-1}\Big(A(t_0)\bigtimes_{k=1}^{i-2}\mathbf{U}_k^{1,T}\Big)\\&\qquad+\big(\mathbf{I}-\mathbf{U}_{i-1}^1\,\mathbf{U}_{i-1}^{1,T}\big)\Big(\mathbf{Mat}_{i-1}\Big(\Delta A\bigtimes_{k=1}^{i-2}\mathbf{U}_k^{1,T}\Big)\,\mathbf{V}_{i-1}^0\,\mathbf{V}_{i-1}^{0,T}\Big),\end{aligned}$$

where the last equality holds by the induction hypothesis. It follows that

$$\mathbf{L}_{i-1}^{0,T}=\mathbf{U}_{i-1}^{1,T}\,\mathbf{Mat}_{i-1}\Big(A(t_0)\bigtimes_{k=1}^{i-2}\mathbf{U}_k^{1,T}\Big)=\mathbf{Mat}_{i-1}\big(A(t_0)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,T}\big).$$

Retensorizing and taking the i-mode unfolding yields

$$\mathbf{Y}_{[i]}(t_0) = \mathbf{Mat}_i(\mathrm{Ten}_{i-1}(\mathbf{L}_{i-1}^{0,T})),$$

which becomes (4.3) with the above formula for $\mathbf{L}_{i-1}^{0,T}$.

To show the exactness of Alg. 3.1, we first consider the d-mode unfolded subproblem. Here, the last substep of the nested Tucker integrator is the same as applying the matrix projector-splitting integrator to (4.2) with initial value (4.3) for $i = d$. Since the updated basis matrices $\mathbf{U}_k^1$ for $k = 1, \dots i-1$ are not time-dependent from the i-th integration step onwards, we observe by means of (4.4), that

$$\begin{aligned}\mathbf{V}_i(t_1)^T\,\mathbf{V}_i(t_0) &= \mathbf{W}_i^T(t_0)\,(\mathbf{U}_1^{1,T}\,\mathbf{U}_1^1 \otimes \cdots \otimes \mathbf{U}_{i-1}^{1,T}\,\mathbf{U}_{i-1}^1 \otimes \mathbf{I}_i \otimes \cdots \otimes \mathbf{I}_d)\;\mathbf{W}_i(t_0)\\ &= \mathbf{W}_i(t_1)^T\,\mathbf{W}_i(t_0),\end{aligned}$$

for all $i = 1, \dots, d$. Additionally, by assumption $\mathbf{W}_i(t_1)^T\,\mathbf{W}_i(t_0)$ is non-singular and so we conclude by Theorem 2.1 that the integrator is exact for the d-mode setting after one time step from t_0 to t_1:

$$\mathbf{Y}_{[d]}^1 = \mathbf{Mat}_d\Big(A(t_1)\bigtimes_{k=1}^{d-1}\mathbf{U}_k^{1,T}\Big).$$

We now show by induction for $i = d, \dots, 1$, that

$$\mathbf{Y}_{[i]}^1 = \mathbf{Mat}_i\Big(A(t_1)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,T}\Big). \tag{4.5}$$

Suppose this has been shown for $\mathbf{Y}_{[d]}^1, \dots, \mathbf{Y}_{[i+1]}^1$. The substep of Alg. 3.1 in the i-mode unfolding solves exactly the differential equations

$$\begin{aligned}\dot{\mathbf{K}}_i(t) &= \mathbf{Mat}_i\Big(\dot{A}(t)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,T}\Big)\,\mathbf{V}_i^0, &\qquad \mathbf{K}_i(t_0) &= \mathbf{Y}_{[i]}^0\,\mathbf{V}_i^0\\ \dot{\mathbf{S}}_i(t) &= \mathbf{U}_i^{1,T}\,\mathbf{Mat}_i\Big(\dot{A}(t)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,T}\Big)\,\mathbf{V}_i^0, &\qquad \mathbf{S}_i(t_0) &= \mathbf{U}_i^{1,T}\,\mathbf{Y}_{[i]}^0\,\mathbf{V}_i^0,\end{aligned}$$

and approximately the differential equation

$$\dot{\mathbf{L}}_i^T(t) = \mathbf{U}_i^{1,T}\,\mathbf{Mat}_i\Big(\dot{A}(t)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,T}\Big), \qquad \mathbf{L}_i^T(t_0) = \mathbf{U}_i^{1,T}\,\mathbf{Y}_{[i]}^0\,.$$

Since $\mathbf{Y}_{[i+1]}^1$ is the exact solution for the $(i+1)$-mode setting, we conclude by induction hypothesis

$$\begin{aligned}\mathbf{L}_i^{1,T} &= \mathbf{Mat}_i\big(\mathrm{Ten}_{i+1}(\mathbf{Y}_{[i+1]}^1)\big) = \mathbf{Mat}_i\Big(A(t_1)\bigtimes_{k=1}^{i}\mathbf{U}_k^{1,T}\Big)\\ &= \mathbf{U}_i^{1,T}\,\mathbf{Mat}_i\Big(A(t_1)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,T}\Big) = \mathbf{L}_i^T(t_1).\end{aligned}$$

Hence also the differential equation in the third substep of the i-mode unfolded subproblem is solved exactly. By the exactness result for the matrix projector-splitting integrator, Alg. 3.1 solves (4.2) with initial value (4.3) exactly, so that (4.5) is satisfied. Hence, (4.5) holds also for $i = 1$, which yields $Y^1 = A(t_1)$. ∎

5. Error bounds for the nested Tucker integrator.

We now show that, just like in Thm. 2.2 for the matrix case, the nested Tucker integrator is robust to small singular values. Since this integrator is based on recursively applying the matrix projector-splitting integrator, the plan is to analyse these recursive steps from the matrix perspective so that we can apply Thm. 2.2. To this end, we first need to generalise the assumptions of Thm. 2.2.

Let $A(t)$ be the solution of (1.1) on $[t_0, T]$. We denote again by $\mathcal{M}$ the manifold of tensors of multilinear rank $(r_1, \dots, r_d)$. Let

$$\mathcal{M}_i = \{Y \in \mathbb{R}^{n_1 \times \cdots \times n_d} \colon \operatorname{rank}(\mathbf{Mat}_i(Y)) = r_i\},$$

so that $\mathcal{M} = \mathcal{M}_1 \cap \cdots \cap \mathcal{M}_d$. We assume that for each $i = 1, \dots, d$, the i-mode unfolding of (1.1) satisfies the following conditions.

- $F(t, Y)$ is Lipschitz continuous and bounded for all $Y, \widetilde{Y} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$:

$$\|F(t,Y) - F(t,\widetilde{Y})\| \le L\|Y - \widetilde{Y}\|, \qquad \|F(t,Y)\| \le B. \tag{5.1}$$

- $F(t, Y)$ can be decomposed into a tangential part and a small perturbation:

$$\begin{aligned} &F(t,Y) = M_i(t,Y) + R_i(t,Y), \\ &M_i(t,Y) \in \mathcal{T}_Y \mathcal{M}_i, \quad \|R_i(t,Y)\| \le \varepsilon, \end{aligned} \tag{5.2}$$

  for all $Y \in \mathcal{M}_i$ in a neighborhood of $A(t)$ and for all $t \in [t_0, T]$.
- The initial value $A(t_0)$ for (1.1) has multilinear rank $(r_1, \dots, r_d)$.

The second condition (5.2) is formulated in terms of $\mathcal{M}_i$ that are essentially fixed matrix manifolds. Since we are solving (1.3) on a fixed rank Tucker manifold $\mathcal{M}$, it seems more natural to impose that $F(t, Y)$ is close to the tangent space of $\mathcal{M}$, that is,

$$\|F(t,Y) - P(Y)F(t,Y)\| \le \varepsilon. \tag{5.3}$$

However, since $\mathcal{M} = \mathcal{M}_1 \cap \cdots \cap \mathcal{M}_d$, by definition of a tangent space we get $T_Y\mathcal{M} \subseteq T_Y\mathcal{M}_1 \cap \cdots \cap T_Y\mathcal{M}_d$ for $Y \in \mathcal{M}$. Hence, $P(Y)F(t,Y) \in T_Y\mathcal{M}_i$ for all $i = 1, \dots, d$ and so (5.3) actually implies (5.2) for all $Y \in \mathcal{M}$.

Theorem 5.1. *Under the above assumptions, the error of the nested Tucker integrator after n steps with step size $h > 0$ satisfies for all $t_n = t_0 + nh \le T$:*

$$\|Y_n - A(t_n)\| \le c_1 h + c_2 \varepsilon,$$

where the constants c_1, c_2 only depend on $L, B, T - t_0$ and the dimension d. In particular, the constants are independent of singular values of matricizations of the exact or approximate solution tensor.

Proof. Recall from (3.5) and (3.8) for the derivation of the nested Tucker integrator, that Alg. 3.1 solves approximately the following subproblem for each mode i on $\mathbb{R}^{r_1 \times \cdots \times r_{i-1} \times n_i \times \cdots \times n_d}$:

$$\dot{Y}_i(t) = F\Big(t, Y_i(t) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^1\Big) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^{1,T}, \qquad Y_i(t_0) = \mathrm{Ten}_{i-1}(\mathbf{L}_{i-1}^{0,T})$$

with $\mathbf{L}_{i-1}^{0,T} = \widetilde{\mathbf{S}}_{i-1}^0 \, \mathbf{V}_{i-1}^{0,T}$. Introducing

$$Z_i(t) = Y_i(t) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^1 \ \in \mathcal{M}_1 \cap \cdots \cap \mathcal{M}_{i-1} \subset \mathbb{R}^{n_1 \times \cdots \times n_d},$$

we obtain the equivalent initial value problem on $\mathbb{R}^{n_1\times\cdots\times n_d}$

$$\dot{Z}_i(t) = F\big(t, Z_i(t)\big) \bigtimes_{k=1}^{i-1} (\mathbf{U}_k^1\, \mathbf{U}_k^{1,T}), \qquad Z_i(t_0) = \mathrm{Ten}_{i-1}(\mathbf{L}_{i-1}^{0,T}) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^1 \,. \tag{5.4}$$

We note that since $\mathbf{L}_{i-1}^{0,T}$ has full rank, we have $Z_i(t_0) \in \mathcal{M}_i$. Alg. 3.1 now applies the matrix projector-splitting integrator with inexact integration in the third substep to the i-mode unfolded differential equation (5.4). This results in the approximation $Z_i^1 \in \mathcal{M}_i$ to $Z_i(t_1)$.

We show by induction for $i = d, \dots, 1$ the local error bound

$$\|Z_i^1 - Z_i(t_1)\| = \mathcal{O}(h(\varepsilon + h)), \tag{5.5}$$

where the constants symbolized by the $\mathcal{O}$ notation depend only on L, B and d. For $i = d$, the approximation is obtained by the matrix projector-splitting algorithm with exact solution of all three substeps. We verify that the conditions (a–c) of Thm. 2.2 applied to (5.4) are satisfied: Assumption (a) is trivially satisfied by (5.1) since $\mathbf{U}_k^1\, \mathbf{U}_k^{1,T}$ is an orthogonal projector. Using (5.2), the d-mode unfolding of the right hand side of (5.4) can be decomposed, for $Y \in \mathcal{M}_d$, as

$$\begin{aligned}\mathbf{Mat}_d\Big(F(t,Y) \bigtimes_{k=1}^{d-1} (\mathbf{U}_k^1\, \mathbf{U}_k^{1,T})\Big) &= \mathbf{Mat}_d\Big(M_d(t,Y) \bigtimes_{k=1}^{d-1} (\mathbf{U}_k^1\, \mathbf{U}_k^{1,T})\Big) \\ &\quad + \mathbf{Mat}_d\Big(R_d(t,Y) \bigtimes_{k=1}^{d-1} (\mathbf{U}_k^1\, \mathbf{U}_k^{1,T})\Big),\end{aligned}$$

where $M_d(t,Y) \in \mathcal{T}_Y\mathcal{M}_d$ and $\|R_d(t,Y)\| \le \varepsilon$.

We note that if $M_d(t,Y) \in \mathcal{T}_Y\mathcal{M}_d$ and $Y = Y \bigtimes_{k=1}^{d-1} (\mathbf{U}_k^1\, \mathbf{U}_k^{1,T})$ (as is the case for $Y = Z_i(t)$ in (5.4)), then we also have $M_d(t,Y) \bigtimes_{k=1}^{d-1} (\mathbf{U}_k^1\, \mathbf{U}_k^{1,T}) \in \mathcal{T}_Y\mathcal{M}_d$. This holds true because if we consider the singular value decomposition of $\mathbf{Mat}_d(Y) = \mathbf{U}\,\mathbf{S}\,\mathbf{V}^T$, then $\mathbf{V}^T = \mathbf{V}^T \bigotimes_{k=1}^{d-1} \mathbf{U}_k^1\, \mathbf{U}_k^{1,T}$. Now, $M \in \mathcal{T}_Y\mathcal{M}_d$ means that $\mathbf{Mat}_d(M) = \delta\mathbf{U}\,\mathbf{S}\,\mathbf{V}^T + \mathbf{U}\,\delta\mathbf{S}\,\mathbf{V}^T + \mathbf{U}\,\mathbf{S}\,\delta\mathbf{V}^T$ for some suitable $\delta\mathbf{U}, \delta\mathbf{S}, \delta\mathbf{V}$. But then, since $\mathbf{V}^T = \mathbf{V}^T \bigotimes_{k=1}^{d-1} \mathbf{U}_k^1\, \mathbf{U}_k^{1,T}$, this implies that $\mathbf{Mat}_d\big(M \bigtimes_{k=1}^{d-1} (\mathbf{U}_k^1\, \mathbf{U}_k^{1,T})\big)$ is of the same form with a modifed $\delta\mathbf{V}$, and hence $M \bigtimes_{k=1}^{d-1} (\mathbf{U}_k^1\, \mathbf{U}_k^{1,T}) \in \mathcal{T}_Y\mathcal{M}_d$.

By definition, $\mathbf{Mat}_d(\mathcal{M}_d) = \{\mathbf{Mat}_d(Y) : Y \in \mathcal{M}_d\}$ is the manifold of matrices of rank r_d of dimension $(n_d \times n_1 \cdots n_{d-1})$. Moreover, for $Y \in \mathcal{M}_d$ and $\mathbf{Y} = \mathbf{Mat}_d(Y)$, we have $\mathcal{T}_{\mathbf{Y}}\,\mathbf{Mat}_d(\mathcal{M}_d) = \mathbf{Mat}_d(\mathcal{T}_Y\mathcal{M}_d)$. We conclude that

$$\mathbf{Mat}_d\Big(M_d(t,Y) \bigtimes_{k=1}^{d-1} (\mathbf{U}_k^1\, \mathbf{U}_k^{1,T})\Big) \in \mathcal{T}_{\mathbf{Y}}\,\mathbf{Mat}_d(\mathcal{M}_d)$$

and the corresponding term with R_d is still bounded by ε thanks to (5.2). Hence, assumption (b) is verified. Since assumption (c) was shown above, we are now in the situation to apply Thm. 2.2, which yields (5.5) for $i = d$.

We proceed similarly for $i = d-1$ down to 1. In these cases, we apply the matrix projector-splitting algorithm to the ith unfolding with an inexact solution of the third substep. The error of this inexact solution is given by (5.5) for $i+1$. In the same way as before, the conditions of Thm. 2.2 are verified for the rank r_i matrix manifold $\mathbf{Mat}_i(\mathcal{M}_i)$. With the induction hypothesis that (5.5) holds for $i+1, \dots, d$, we conclude from the error bound (2.5) (for the situation of inexact solutions in the substeps) that (5.5) also holds for i.

For $i = 1$, this gives the stated error bound. $\square$

6. Equivalence with the tensor projector-splitting integrator of [10].

The derivation of the nested Tucker integrator presented in Section 3 is based on the idea of solving the differential equations for $\mathbf{K}_i$ and $\mathbf{S}_i$ directly, but computing a low-rank approximation of the evolution equation for $\mathbf{L}_i$, where $i = 1, \dots, d-1$.

This is a conceptually different derivation compared to the description of the time integrator in [10]. Its full algorithm is restated in Alg. 6.1.

Algorithm 6.1 One time step of the Tucker integrator of [10]

```
Data: Tucker tensor $Y^0 = C^0 \bigtimes_{i=1}^d \mathbf{U}_i^0$, $F(t,Y)$, $t_0$, $t_1$
Result: Tucker tensor $Y^1 = C^1 \bigtimes_{i=1}^d \mathbf{U}_i^1$
1  begin
2      for $i = 1$ to $d$ do
3          compute QR factorization $\mathbf{Mat}_i(C^0)^T = \mathbf{Q}_i^0 \, \mathbf{S}_i^{0,T}$
4          set $\mathbf{V}_i^{0,T} = \mathbf{Mat}_i\Big(\mathrm{Ten}_i(\mathbf{Q}_i^{0,T}) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^{1,T} \bigtimes_{l=i+1}^{d} \mathbf{U}_l^{0,T}\Big)$
5          set $\mathbf{K}_i^0 = \mathbf{U}_i^0 \, \mathbf{S}_i^0$
6          set $\mathbf{Y}_{[i]}^+(t) = \mathbf{K}_i(t) \, \mathbf{V}_i^{0,T}$
7          solve $\dot{\mathbf{K}}_i(t) = \mathbf{Mat}_i\big(F(t, \mathrm{Ten}_i(\mathbf{Y}_{[i]}^+))\big) \, \mathbf{V}_i^0$,
             with initial value $\mathbf{K}_i(t_0) = \mathbf{K}_i^0$ and return $\mathbf{K}_i^1 = \mathbf{K}_i(t_1)$
8          compute QR factorization $\mathbf{K}_i^1 = \mathbf{U}_i^1 \, \widehat{\mathbf{S}}_i^1$
9          set $\mathbf{S}_i^0 = \widehat{\mathbf{S}}_i^1$
10         set $\mathbf{Y}_{[i]}^-(t) = \mathbf{U}_i^1 \, \mathbf{S}_i(t) \, \mathbf{V}_i^{0,T}$
11         solve $\dot{\mathbf{S}}_i(t) = -\,\mathbf{U}_i^{1,T} \, \mathbf{Mat}_i\big(F(t, \mathrm{Ten}_i(\mathbf{Y}_{[i]}^-))\big) \, \mathbf{V}_i^0$,
             with initial value $\mathbf{S}_i(t_0) = \mathbf{S}_i^0$ and return $\mathbf{S}_i^1 = \mathbf{S}_i(t_1)$
12         set $C^0 = \mathrm{Ten}_i(\mathbf{S}_i^1 \, \mathbf{Q}_i^{0,T})$
13     solve $\dot{C}(t) = F\big(t, C(t) \bigtimes_{i=1}^{d} \mathbf{U}_i^1\big) \bigtimes_{i=1}^{d} \mathbf{U}_i^{1,T}$,
         with initial value $C(t_0) = C^0$ and return $C^1 = C(t_1)$
14     set $Y^1 = C^1 \bigtimes_{i=1}^{d} \mathbf{U}_i^1$
```

Comparing Alg. 6.1 with Alg. 3.1 for the nested Tucker integrator, we see that the two algorithms solve different matrix differential equations for $\mathbf{K}_i(t)$ and $\mathbf{S}_i(t)$. The reason is the positioning of the updated basis matrices $\mathbf{U}_i^1$, i.e., in Alg. 6.1, the corange $\mathbf{V}_i^{0,T}$ of the current unfolded approximation tensor takes those basis matrices after performing one time step, whereas in Alg. 3.1, they are provided on the right hand side of the differential equation for $Y_i(t)$. Therefore, we observe that the time integrator described in [10] does not reduce the dimension in each mode, contrary to the nested Tucker integrator, which diminishes the dimension of the current mode due to the inexact solution in the third substep.

The derivation of the two integrators also draws a distinction regarding the known favorable properties of the matrix integrator to the Tucker tensor case. As we have seen in Sections 4 and 5, the exactness result and the robustness with respect to singular values can be transfered to the nested Tucker integrator, whereas it is unclear how to show those results directly for the integrator proposed in [10].

Nevertheless, those two properties also hold for the Tucker integrator of Alg. 6.1, since we will see in the following that both integration methods are equivalent.

THEOREM 6.1. *The nested Tucker integrator presented in Section 3 (Alg. 3.1) and the Tucker integrator described in [10] (Alg. 6.1) applied on the tensor differential equation (1.1) are equivalent in the sense that they yield the same low-rank approximation Y^1 after one time step.*

Proof. It is sufficient to show equivalence of the matrix differential equations that appear in Alg. 3.1 and 6.1. In order to distinguish between the factors computed by those two methods, we will denote those for Alg. 6.1 using $\overline{\cdot}$, e.g., $\overline{\mathbf{V}}_i$, $\overline{\mathbf{K}}_i(t)$, and $\overline{\mathbf{S}}_i(t)$. The notation for Alg. 3.1 is left unchanged.

We start with Alg. 6.1. Writing line 4 as

$$\overline{\mathbf{V}}_i^{0,T} = \mathbf{Q}_i^{0,T}\Big(\bigotimes_{k=1}^{i-1} \mathbf{U}_k^{1,T} \otimes \bigotimes_{k=i+1}^{d} \mathbf{U}_k^{0,T}\Big),$$

the equation of motion of $\mathbf{K}_i$ becomes

$$\begin{aligned}\dot{\overline{\mathbf{K}}}_i(t) &= \mathbf{Mat}_i\Big(F\big(t, \mathrm{Ten}_i(\overline{\mathbf{K}}_i(t)\overline{\mathbf{V}}_i^{0,T})\big)\Big)\overline{\mathbf{V}}_i^0\\ &= \mathbf{Mat}_i\Big(F\Big(t, \mathrm{Ten}_i\Big(\overline{\mathbf{K}}_i(t)\,\mathbf{Q}_i^{0,T}\Big(\bigotimes_{k=1}^{i-1} \mathbf{U}_k^{1,T} \otimes \bigotimes_{k=i+1}^{d} \mathbf{U}_k^{0,T}\Big)\Big)\Big)\Big)\overline{\mathbf{V}}_i^0.\end{aligned}$$

For Alg. 3.1, on the other hand, that equation reads

$$\dot{\mathbf{K}}_i(t) = \mathbf{Mat}_i\Big(F\Big(t, \mathrm{Ten}_i(\mathbf{K}_i(t)\,\mathbf{V}_i^{0,T})\bigtimes_{k=1}^{i-1}\mathbf{U}_k^1\Big)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,T}\Big)\,\mathbf{V}_i^0\,.$$

We first expand the argument of F in this ODE. Writing line 4 in Alg. 3.1 as

$$\mathbf{V}_i^{0,T} = \mathbf{Q}_i^{0,T}\Big(\bigotimes_{k=1}^{i-1}\mathbf{I}_{r_k} \otimes \bigotimes_{k=i+1}^{d}\mathbf{U}_k^{0,T}\Big)$$

and substituting, we obtain

$$\begin{aligned}\mathrm{Ten}_i(\mathbf{K}_i(t)\,\mathbf{V}_i^{0,T})\bigtimes_{k=1}^{i-1}\mathbf{U}_k^1 &= \mathrm{Ten}_i(\mathbf{K}_i(t)\,\mathbf{Q}_i^{0,T})\bigtimes_{k=i+1}^{d}\mathbf{U}_k^0\bigtimes_{k=1}^{i-1}\mathbf{U}_k^1\\ &= \mathrm{Ten}_i\Big(\mathbf{K}_i(t)\,\mathbf{Q}_i^{0,T}\Big(\bigotimes_{k=i+1}^{d}\mathbf{U}_k^{0,T}\otimes\bigotimes_{k=1}^{i-1}\mathbf{U}_k^{1,T}\Big)\Big).\end{aligned}$$

Hence, we see that F has the same arguments in both algorithms for the K-step. Omitting it, we continue with

$$\begin{aligned}\dot{\mathbf{K}}_i(t) &= \mathbf{Mat}_i\Big(F(t,\cdot)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,T}\Big)\,\mathbf{V}_i^0\\ &= \mathbf{Mat}_i\Big(F(t,\cdot)\Big)\Big(\bigotimes_{k=1}^{i-1}\mathbf{U}_k^{1,T}\otimes\bigotimes_{k=i+1}^{d}\mathbf{I}_{r_k}\Big)\Big(\bigotimes_{k=1}^{i-1}\mathbf{I}_{r_k}\otimes\bigotimes_{k=i+1}^{d}\mathbf{U}_k^0\Big)\,\mathbf{Q}_i^0\\ &= \mathbf{Mat}_i\Big(F(t,\cdot)\Big)\Big(\bigotimes_{k=1}^{i-1}\mathbf{U}_k^{1,T}\otimes\bigotimes_{k=i+1}^{d}\mathbf{U}_k^0\Big)\,\mathbf{Q}_i^0\,.\end{aligned}$$

Comparing with $\overline{\mathbf{V}}_i^0$ above, we see that the differential equations for $\mathbf{K}_i(t)$ and $\overline{\mathbf{K}}_i(t)$ are indeed equivalent.

Hence, applying the same numerical method to both of them would give the same result $\mathbf{K}_i^1 = \overline{\mathbf{K}}_i^1$.

The equivalence of the evolution equations for $\dot{\mathbf{S}}_i(t)$ and $\dot{\overline{\mathbf{S}}}_i(t)$ can be shown in the same way as above. This again gives the same numerical solutions after one time step, i.e., $\mathbf{S}_i^1 = \overline{\mathbf{S}}_i^1$.

Finally, for the core tensor, we compare the evolution equation for $C(t)$ in Alg. 6.1,

$$\dot{C}(t) = F\big(t, C(t) \bigtimes_{i=1}^{d} \mathbf{U}_i^1\big) \bigtimes_{i=1}^{d} \mathbf{U}_i^{1,T}, \qquad C(t_0) = C^0,$$

with that of $\mathbf{L}(t)$ from Alg. 3.1. Retensorizing the latter in the dth mode yields

$$\begin{aligned}
\mathrm{Ten}_d(\dot{\mathbf{L}}^T(t)) &= \mathrm{Ten}_d\Big(\mathbf{U}_d^{1,T}\,\mathbf{Mat}_d\Big(F\Big(t, \mathrm{Ten}_d(\mathbf{U}_d^1 L^T(t)) \bigtimes_{i=1}^{d-1} \mathbf{U}_i^1\Big) \bigtimes_{i=1}^{d-1} \mathbf{U}_i^{1,T}\Big)\Big) \\
&= \mathrm{Ten}_d\Big(\mathbf{U}_d^{1,T}\,\mathbf{Mat}_d\Big(F\Big(t, \mathrm{Ten}_d(\mathbf{L}^T(t)) \bigtimes_{i=1}^{d} \mathbf{U}_i^1\Big) \bigtimes_{i=1}^{d-1} \mathbf{U}_i^{1,T}\Big)\Big) \\
&= F\big(t, \mathrm{Ten}_d(\mathbf{L}^T(t)) \bigtimes_{i=1}^{d} \mathbf{U}_i^1\big) \bigtimes_{i=1}^{d} \mathbf{U}_i^{1,T}.
\end{aligned}$$

Identifying now $C(t)$ as $\mathrm{Ten}_d(\mathbf{L}^T(t))$, we see that differential equations are the same. Since the same holds true for the initial values,

$$\mathrm{Ten}_d(\mathbf{L}_d^{0,T}) = \mathrm{Ten}_d(\mathbf{Mat}_d(C^0)) = C^0,$$

both algorithms deliver the same same low-rank approximation Y^1. ∎

7. Numerical experiments.

We present two numerical examples to illustrate our theoretical results of the proposed Tucker integrator. We consider examples that are tensor variants of the examples in [4, Section 4] for the matrix case, in particular, a discrete nonlinear Schrödinger equation and an example of approximately adding tensors in the Tucker format.

7.1. A discrete nonlinear Schrödinger equation.

We model a dilute Bose–Einstein condensate, trapped in a periodic potential [14], on a regular lattice of width γ. The dynamics of its phase diagram is governed by the discrete nonlinear Schrödinger equation

$$\begin{aligned}
i\dot{A}(t) &= -\frac{1}{2}L[A(t)] + \varepsilon|A(t)|^2 \odot A(t) \\
A_{jkl}(t_0) &= \exp\big(-1/\gamma^2((j-j_1)^2 - (k-k_1)^2 - (l-l_1)^2)\big) \\
&\quad + \exp\big(-1/\gamma^2((j-j_2)^2 - (k-k_2)^2 - (l-l_2)^2)\big),
\end{aligned} \tag{7.1}$$

where $A(t) \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ with $n_i = 100$ for all $i \in \{1,2,3\}$. The bounded linear operator $L\colon \mathbb{R}^{n_1 \times n_2 \times n_3} \to \mathbb{R}^{n_1 \times n_2 \times n_3}$ describes the interaction between the grid points centred at (j,k,l) for all $j,k,l = 1,\dots,100$. It is defined component-wise as

$$\begin{aligned}
L[A](j,k,l) &= A(j-1,k,l) + A(j+1,k,l) + A(j,k-1,l) + A(j,k+1,l) \\
&\quad + A(j,k,l-1) + A(j,k,l+1),
\end{aligned}$$

where terms with indices outside the range from 1 to 100 are interpreted as 0. Compared to the more standard seven-point stencil for the discrete Laplace operator in three dimensions, the operator L does not take the centred grid point into account. The entries of the tensor $|A|^2$ are the squares of the absolute values of the corresponding entries of A, and the $\odot$ in $|A|^2 \odot A$ desnotes the entrywise (Hadamard) product. The parameter ε determines the degree of nonlinearity.

We consider two excitations of the system, which are located at grid-points $(j_1, k_1, l_1) = (75, 25, 1)$ and $(j_2, k_2, l_2) = (25, 75, 100)$. We take $\gamma = 10$.

To compute a low-rank approximation $Y(t) \in \mathcal{M}$ with multilinear rank $r = (10, 10, 10)$ to the solution of the nonlinear differential equation (7.1), we apply our nested Tucker integrator (Alg. 3.1) to (7.1). The differential equations appearing in the substeps of each mode are solved by the 4th order Runge–Kutta method with time step size $h = 10^{-3}$. This approximate solution is compared to a full rank reference solution, which is also computed by a 4th order Runge–Kutta method, but with $h = 0.5 \cdot 10^{-3}$. In Table 7.1 we show the error behavior for different parameters ε and time step sizes h:

$\varepsilon \setminus h$	1	10^{-1}	10^{-2}	10^{-3}
1	4.59e-1	4.01e-2	3.88e-2	3.88e-2
10^{-1}	9.39e-2	9.68e-4	1.61e-4	1.47e-4
10^{-2}	9.27e-3	3.20e-5	2.19e-6	1.30e-6
10^{-3}	5.36e-4	3.18e-6	8.93e-8	3.54e-8
10^{-4}	5.12e-5	2.73e-7	3.23e-9	1.91e-9

TABLE 7.1
Error in Frobenius norm at $t = 1$ of the rank-(10,10,10) Tucker integrator applied to (7.1).

For each time step size h, we see the error decaying with ε. This observation is due to the fact that the linear term $L[A(t)]$ in (7.1) maps onto the tangent-space $\mathcal{T}_Y\mathcal{M}$ of the manifold $\mathcal{M}$ of multilinear rank. The nonlinear term is of full rank, but it is controlled by the factor ε. This makes the dependence of the error behaviour on ε explicit. We also see in the first row, that the error stagnates from time step size $h = 10^{-2}$ on and this shows the dominance of the perturbation factor ε. We would observe the same behaviour for smaller ε, but for smaller time step sizes.

Finally, in the last row, where the influence of ε is small, we observe convergence of the error in terms of the time step size h.

7.2. Approximate addition of tensors. Let $A \in \mathbb{C}^{n_1 \times \cdots \times n_d}$ be a tensor of multilinear rank $r = (r_1, \ldots, r_d)$ and let $B \in \mathbb{C}^{n_1 \times \cdots \times n_d}$. We consider the addition of the two given tensors, which results in

$$C = A + B, \tag{7.2}$$

where C typically is not of low rank. We aim to find an approximation tensor of multilinear rank r. Such a computation is for example required in optimization problems on low-rank manifolds, under the name of retractions, and need to be computed in each iterative step [1]. There, the increment is typically a tangential tensor $B \in \mathcal{T}_A\mathcal{M}$, which after adding directly as in (7.2) yields a tensor C of multilinear rank 2r. Afterwards, the result is projected onto $\mathcal{M}$ by an SVD-based rank-r approximation in order to obtain an approximation tensor $D \in \mathcal{M}$. With this procedure, we first leave the low-rank manifold and then project back onto $\mathcal{M}$.

Instead, we propose to apply one time step of the nested Tucker integrator starting with $t_0 = 0$ and with time step size $h = 1$ in order to solve

$$\dot{Y}(t) = P(Y)B, \qquad Y(t_0) = A.$$

This gives an approximate solution $Y^1 \in \mathcal{M}$ for the result of the direct addition (7.2). Contrary to the standard approach, we never leave the low-rank manifold when applying the nested Tucker integrator.

For our numerical example, we initialise A as a random Tucker tensor of size $100 \times 100 \times 100$ and multi-linear rank $r = (10, 10, 10)$. The increment B is constructed to be a random tensor in the tangent space $\mathcal{T}_A\mathcal{M}$. We compare the full rank addition (7.2) with the low-rank approximation $Y^1 \in \mathcal{M}$ obtained by the nested Tucker integrator. We also compare those results with the retracted rank-$2r$ result, for which we perform a best rank-r approximation. The figure below illustrates those comparisons:

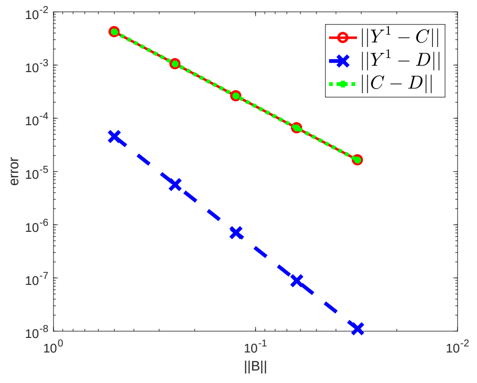

FIG. 7.1. *Errors for tensor addition for tangential increments B of decreasing norm.*

We observe that the errors decrease with decreasing norm of the increment tensor B. We also see that the difference between the errors of the splitting integrator and the projected direct addition is marginal — or rather we do not see it because the error curves of both approaches are not distinguishable in the figure.

Acknowledgements. This work was supported by a grant from DFG through the GRK 1838. We thank Balázs Kovács for helpful discussions about numerical examples.

REFERENCES

[1] P.-A. Absil and I. V. Oseledets. Low-rank retractions: a survey and new results. *Comput. Optim. Appl.*, 62, 2014.
[2] L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications*, 21:1253–1278, 2000.
[3] J. Haegeman, C. Lubich, I. Oseledets, B. Vandereycken, and F. Verstraete. Unifying time evolution and optimization with matrix product states. *Physical Review B*, 94, 2016.
[4] E. Kieri, C. Lubich, and H. Walach. Discretized dynamical low-rank approximation in the presence of small singular values. *SIAM J. Numer. Anal.*, 54:1020–1038, 2016.
[5] E. Kieri and B. Vandereycken. Projection methods for dynamical low-rank approximation of high-dimensional problems. Tech. report (submitted), 2017.

[6] B. Kloss, I. Burghardt, and C. Lubich. Implementation of a novel projector-splitting integrator for the multi-configurational time-dependent hartree approach. *The Journal of Chemical Physics*, 146, 2017.

[7] O. Koch and C. Lubich. Dynamical tensor approximation. *SIAM J. Matrix Anal. Appl.*, 31:2360–2375, 2010.

[8] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM review*, 51:455–500, 2009.

[9] D. Kressner, R. Kumar, F. Nobile, and C. Tobler. Low-rank tensor approximation for high-order correlation functions of Gaussian random fields. *SIAM/ASA J. Uncertain. Quantif.*, 3(1):393–416, 2015.

[10] C. Lubich. Time integration in the multiconfiguration time-dependent hartree method of molecular quantum dynamics. *Applied Mathematics Research eXpress*, 2015:311–328, 2015.

[11] C. Lubich and I. V. Oseledets. A projector-splitting integrator for dynamical low-rank approximation. *BIT*, 54:171–188, 2014.

[12] C. Lubich, I. V. Oseledets, and B. Vandereycken. Time integration of tensor trains. *SIAM Journal on Numerical Analysis*, 53:917–941, 2015.

[13] H.-D. Meyer, F. Gatti, and G. A. Worth. *Multidimensional quantum dynamics.* John Wiley & Sons, 2009.

[14] A. Trombettoni and A. Smerzi. Discrete solitons and breathers with dilute Bose–Einstein condensates. *Phys. Rev. Lett.*, 86:2353–2356, 2001.