Krylov-type methods for tensor computations I

Berkant Savas and Lars Eldén

Original Publication:

Berkant Savas and Lars Eldén, Krylov-type methods for tensor computations I, 2013, Linear Algebra and its Applications, (438), 891-918.
http://dx.doi.org/10.1016/j.laa.2011.12.007

Krylov-Type Methods for Tensor Computations I[☆]

Berkant Savas[a,b], Lars Eldén[a]

[a] *Department of Mathematics, Linköping University, SE-581 83 Linköping, Sweden*
[b] *Institute for Comp. Engin. and Sciences, University of Texas at Austin, Austin, TX 78712, USA*

Abstract

Several Krylov-type procedures are introduced that generalize matrix Krylov methods for tensor computations. They are denoted *minimal Krylov recursion*, *maximal Krylov recursion*, and *contracted tensor product Krylov recursion*. It is proved that, for a given tensor $\mathcal{A}$ with multilinear rank-(p, q, r), the minimal Krylov recursion extracts the correct subspaces associated to the tensor in $p+q+r$ number of tensor-vector-vector multiplications. An optimized minimal Krylov procedure is described that, for a given multilinear rank of an approximation, produces a better approximation than the standard minimal recursion. We further generalize the matrix Krylov decomposition to a *tensor Krylov decomposition*. The tensor Krylov methods are intended for the computation of low multilinear rank approximations of large and sparse tensors, but they are also useful for certain dense and structured tensors for computing their higher order singular value decompositions or obtaining starting points for the best low-rank computations of tensors. A set of numerical experiments, using real-world and synthetic data sets, illustrate some of the properties of the tensor Krylov methods.

Keywords: Tensor, Krylov-type method, tensor approximation, Tucker model, multilinear algebra, multilinear rank, sparse tensor, information science
AMS: 15A69, 65F99

1. Introduction

It has been recently shown that several applications in information sciences, such as web link analysis, social network analysis, and cross-language information retrieval, generate large data sets that are sparse tensors, see e.g. [19] and several references in the survey [20]. There are also applications in chemistry, where one wants to compress tensors given in a particular structured (canonical) format [12]. The common property of such structured tensors and sparse tensors, is that it is possible to perform tensor-vector multiplications efficiently. In addition, in both cases the tensors are often so large that storage efficiency becomes an important issue, which precludes standard methods for small and medium size tensors. In this paper we introduce new methods for efficient computations with large, sparse and structured tensors.

[☆] This work was supported by the Swedish Research Council.
Email addresses: berkant@cs.utexas.edu (Berkant Savas), lars.elden@liu.se (Lars Eldén)

Since the 1950's Krylov subspace methods have been developed so that they are now one of the main classes of algorithms for solving iteratively large and sparse matrix problems. Given a square matrix $A \in \mathbb{R}^{n\times n}$ and a starting vector $u \in \mathbb{R}^n$ the corresponding k-dimensional Krylov subspace is

$$\mathcal{K}_k(A, u) = \text{span}\{u, Au, A^2u, \ldots, A^{k-1}u\}. \tag{1}$$

In floating point arithmetic the Krylov subspace vectors in (1) become increasingly dependent and eventually useless unless they are orthonormalized. Applying Gram–Schmidt orthogonalization one obtains the Arnoldi process, which generates an orthonormal basis for the Krylov subspace $\mathcal{K}_k(A, u)$. In addition, the Arnoldi process generates the factorization

$$AU_k = U_{k+1}\widehat{H}_k, \tag{2}$$

where $U_k = [u_1 \ldots u_k]$ and $U_{k+1} = [U_k\ u_{k+1}]$ have orthonormal columns, and $\widehat{H}_k$ is a $(k + 1) \times k$ Hessenberg matrix with orthonormalization coefficients. Based on the relation in (2) one can compute an approximation of the solution of a linear system or an eigenvalue problem by projecting onto the space spanned by the columns of U_k, where k is much smaller than the dimension of A. On the subspace given by columns of U_k, the operator A is represented by the small matrix $H_k = U_k^\mathsf{T} AU_k$. This approach is particularly useful for large, and sparse problems, since it uses the matrix A in matrix-vector multiplications only.

Projection to a low-dimensional subspace is a common technique in many areas of information science. In the tensor case the problem is of computing low multilinear rank approximation or the Tucker decomposition [32] of a given tensor. Another approach involving low rank tensor approximation is the canonical decomposition [4, 13], which decomposes a tensor as a sum of rank-1 tensors. In this paper we only consider the Tucker model since the methods we propose will generate subspaces, to be used in a low multilinear approximation of a tensor. An important thread of research in this field is the computation of low multilinear rank approximations of tensors [1, 5–7, 9, 15, 17, 20, 21, 25, 28, 33]. Interested readers are referred to these references and the references therein.

The following question arises naturally:

> *Can Krylov methods be generalized to tensors, to be used for the projection to low-dimensional subspaces?*

We answer this question in the affirmative, and describe several alternative ways one can generalize Krylov subspace methods for tensors. Our method is inspired by the Golub–Kahan bidiagonalization process [11], and the Arnoldi method, see e.g. [29, p. 303]. In the bidiagonalization method two sequences of orthogonal vectors are generated. For a third order tensor, our procedures generates three sequences of orthogonal vectors. Unlike the bidiagonalization procedure, it is necessary to perform Arnoldi style orthogonalization of the generated vectors explicitly. For matrices, once an initial vector has been selected, the whole sequence is determined uniquely. For tensors, there are many ways in which the vectors can be generated. We will describe three principally different tensor Krylov methods. These are the *minimal Krylov recursion*, *maximal Krylov recursion* and *contracted tensor product Krylov recursion*. In addition we will

present an optimized version of the minimal Krylov recursion similar to [12], and we will show how to deal with tensors that are small in one mode. For a given tensor $\mathcal{A}$ with $\text{rank}(\mathcal{A}) = (p, q, r)$ the minimal Krylov recursion can extract the correct subspaces associated to $\mathcal{A}$ in $p + q + r$ tensor-vector-vector multiplications. The maximal Krylov recursion admits a tensor Krylov decomposition that generalizes the matrix Krylov decomposition. The contracted tensor product Krylov recursion is independently applied to three symmetric matrices that are obtained trough contracted tensor products. This process may be seen as a generalization of the matrix Lanczos method independently applied to the symmetric matrices $A^\mathsf{T}A$ and AA^T, which are obtained from a rectangular matrix $A \in \mathbb{R}^{m\times n}$.

Although our main motivation is to develop efficient methods for large and sparse tensors, the methods are useful for other tasks as well. In particular, they can be used for obtaining starting points for the *best* low rank tensor approximation problem, and for tensors with relatively low multilinear rank tensor Krylov methods provide a way of speeding up the computation of the *higher order singular value decomposition* (HOSVD) [6]. The latter part is done by first computing a full factorization using the minimal Krylov procedure and then computing the HOSVD of a much smaller core tensor that results from the decomposition.

The paper is organized as follows. The necessary tensor concepts are introduced in Section 2. The Arnoldi and Golub–Kahan procedures are sketched in Section 3. In Section 4 we describe different variants of Krylov methods for tensors. Section 5 contains numerical examples illustrating various aspects of the proposed methods.

As this paper is a first introduction to Krylov methods for tensors, we do not imply that it gives a comprehensive treatment of the subject. Rather our aim is to outline our discoveries so far, and point to the similarities and differences between the tensor and matrix cases.

2. Tensor Concepts

2.1. Notation and Preliminaries

Tensors will be denoted by calligraphic letters, e.g., $\mathcal{A}, \mathcal{B}$, matrices by capital roman letters, e.g. U, V, and vectors by lower case roman letters, e.g., u and v. In order not to burden the presentation with too much detail, we will occasionally not mention the dimensions of matrices and tensors, and assume that they are such that the operations are well-defined. The whole presentation will be in terms of third order tensors or, equivalently, 3-tensors. Generalization of the presented concepts to higher order tensors is obvious.

We will use the term tensor in the sense of a multidimensional array of real numbers, e.g., $\mathcal{A} \in \mathbb{R}^{l\times m\times n}$, where the vector space is equipped with some algebraic structures to be defined. The different "dimensions" of the tensor are referred to as *modes*. We will use both standard subscripts and "MATLAB-like" notation. A particular tensor element will be denoted in two equivalent ways,

$$\mathcal{A}(i, j, k) = a_{ijk}.$$

A subtensor obtained by fixing one of the indices is called a *slice*, e.g.,

$$\mathcal{A}(i, :, :).$$

Such a slice can be considered as a third order tensor, but also as a matrix. A *fibre* is a subtensor, where all indices but one are fixed, e.g.,

$$\mathcal{A}(i,:,k).$$

For a given third order tensor, there are three "associated subspaces", one for each mode. These subspaces are given by

$$\begin{aligned}
\mathbb{S}_1 &= \text{range}\{\mathcal{A}(:,j,k) \;\mid\; j=1,\cdots,m;\;\; k=1,\cdots,n\},\\
\mathbb{S}_2 &= \text{range}\{\mathcal{A}(i,:,k) \;\mid\; i=1,\cdots,l;\;\; k=1,\cdots,n\},\\
\mathbb{S}_3 &= \text{range}\{\mathcal{A}(i,j,:) \;\mid\; i=1,\cdots,l;\;\; j=1,\cdots,m\}.
\end{aligned}$$

The *multilinear rank* [8, 14, 22] of a 3-tensor is a triplet (p,q,r) such that the dimension of the subspaces $\mathbb{S}_1$, $\mathbb{S}_2$, and $\mathbb{S}_3$ are p, q, and r, respectively. A given tensor $\mathcal{A} \in \mathbb{R}^{l\times m\times n}$ with rank-(p,q,r) may be written in factored form:

$$\mathcal{A} = (X,Y,Z)\cdot\mathcal{C},$$

where $X \in \mathbb{R}^{l\times p}$, $Y \in \mathbb{R}^{m\times q}$, and $Z \in \mathbb{R}^{n\times r}$ are full (column) rank matrices and $\mathcal{C}$ is a $p\times q\times r$ core tensor. This factorization is not unique since we may change basis $\bar{X} = XA$, $\bar{Y} = YB$, and $\bar{Z} = ZC$ with any non-singular matrices A, B, and C, respectively. In the new basis the factorization becomes

$$\mathcal{A} = (X,Y,Z)\cdot\mathcal{C} = \left(\bar{X},\bar{Y},\bar{Z}\right)\cdot\bar{\mathcal{C}}, \quad \text{where} \quad \bar{\mathcal{C}} = \left(A^{-1},B^{-1},C^{-1}\right)\cdot\mathcal{C}.$$

It is obvious that the factorization of the tensor $\mathcal{A}$ is characterized by three subspaces, which are of course the same as $\mathbb{S}_1$, $\mathbb{S}_2$, and $\mathbb{S}_3$, rather than specific matrices spanning those subspaces. We will say that a matrix $U = [u_1\ \cdots\ u_p]$ spans the "correct subspace" when $\text{span}\{u_1,\cdots,u_p\} = \mathbb{S}_1$. Similarly for the other two modes.

It is customary in numerical linear algebra to write out column vectors with the elements arranged vertically, and row vectors with the elements horizontally. This becomes inconvenient when we are dealing with more than two modes. Therefore we will not make a notational distinction between mode-1, mode-2, and mode-3 vectors, and we will allow ourselves to write all vectors organized vertically. It will be clear from the context to which mode the vectors belong. However, when dealing with matrices, we will often talk of them as consisting of column vectors.

2.2. Tensor-Matrix Multiplication

We define *multilinear multiplication of a tensor by a matrix* as follows. For concreteness we first present multiplication by one matrix along the first mode and later for all three modes simultaneously. The mode-1 product of a tensor $\mathcal{A} \in \mathbb{R}^{l\times m\times n}$ by a matrix $U \in \mathbb{R}^{p\times l}$ is defined by

$$\mathbb{R}^{p\times m\times n} \ni \mathcal{B} = (U)_1\cdot\mathcal{A}, \qquad b_{ijk} = \sum_{\alpha=1}^{l} u_{i\alpha}a_{\alpha jk}. \tag{3}$$

This means that all mode-1 fibres in the 3-tensor $\mathcal{A}$ are multiplied by the matrix U. Similarly, mode-2 multiplication by a matrix $V \in \mathbb{R}^{q\times m}$ means that all mode-2 fibres are

multiplied by the matrix V. Mode-3 multiplication is analogous. With a third matrix $W \in \mathbb{R}^{r\times n}$, the tensor-matrix multiplication[1] in all modes is given by

$$\mathbb{R}^{p\times q\times r} \ni \mathcal{B} = (U, V, W) \cdot \mathcal{A}, \qquad b_{ijk} = \sum_{\alpha,\beta,\gamma=1}^{l,m,n} u_{i\alpha} v_{j\beta} w_{k\gamma} a_{\alpha\beta\gamma}, \tag{4}$$

where the mode of each multiplication[2] is understood from the order in which the matrices are given.

It is convenient to introduce a separate notation for multiplication by a transposed matrix $\bar{U} \in \mathbb{R}^{l\times p}$:

$$\mathbb{R}^{p\times m\times n} \ni \mathcal{C} = \left(\bar{U}^{\mathsf{T}}\right)_1 \cdot \mathcal{A} = \mathcal{A} \cdot \left(\bar{U}\right)_1, \qquad c_{ijk} = \sum_{\alpha=1}^{l} a_{\alpha jk} \bar{u}_{\alpha i}. \tag{5}$$

Let $u \in \mathbb{R}^l$ be a vector and $\mathcal{A} \in \mathbb{R}^{l\times m\times n}$ a tensor. Then

$$\mathbb{R}^{1\times m\times n} \ni \mathcal{B} := \left(u^{\mathsf{T}}\right)_1 \cdot \mathcal{A} = \mathcal{A} \cdot (u)_1 \equiv B \in \mathbb{R}^{m\times n}. \tag{6}$$

Thus we identify a 3-tensor with a singleton dimension with a matrix. Similarly, with $u \in \mathbb{R}^l$ and $w \in \mathbb{R}^n$, we will identify

$$\mathbb{R}^{1\times m\times 1} \ni \mathcal{C} := \mathcal{A} \cdot (u, w)_{1,3} \equiv c \in \mathbb{R}^m, \tag{7}$$

i.e., a 3-tensor with two singleton dimensions, with a vector. Formulas like (7), where we multiply a tensor by two vectors in different modes, will be denoted as a *tensor-vector-vector multiplication*. Since these multiplications are of key importance in this paper, we will state the other two versions as well:

$$\mathcal{A} \cdot (u, v)_{1,2} \in \mathbb{R}^n, \qquad \mathcal{A} \cdot (v, w)_{2,3} \in \mathbb{R}^l, \tag{8}$$

where $v \in \mathbb{R}^m$.

2.3. Inner Product, Norm, and Contractions

Given two tensors $\mathcal{A}$ and $\mathcal{B}$ with equal dimensions, we define the *inner product* to be

$$\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{\alpha,\beta,\gamma} a_{\alpha\beta\gamma} b_{\alpha\beta\gamma}. \tag{9}$$

The corresponding *tensor norm* is given by

$$\|\mathcal{A}\| = \langle \mathcal{A}, \mathcal{A} \rangle^{1/2}, \tag{10}$$

[1] To clarify the presentation, when dealing with a general third order tensor $\mathcal{A}$, we will use the convention that matrices or vectors U, U_k, u_i; V, V_k, v_i; and W, W_k, w_i, are exclusively multiplied along the first, second, and third mode of $\mathcal{A}$, respectively, and similarly with matrices X, Y, Z, and vectors x, y, z.

[2] The notation (4) was suggested by Lim [8]. An alternative notation was earlier given in [6]. Our $(X)_d \cdot \mathcal{A}$ is the same as $\mathcal{A} \times_d X$ in that system.

which is also the *Frobenius norm*. We will use this norm throughout the paper. As in the matrix case, the norm is invariant under orthogonal transformations, i.e.,

$$\|\mathcal{A}\| = \|(U, V, W) \cdot \mathcal{A}\| = \|\mathcal{A} \cdot (P, Q, S)\|$$

for any orthogonal matrices U, V, W, P, Q, and S. This is obvious from the fact that multilinear multiplication by orthogonal matrices does not change the Euclidean length of the corresponding fibres of the tensor.

For convenience we will denote the inner product of vectors[3] x and y by $x^{\mathsf{T}}y$. Let $v = \mathcal{A} \cdot (u, w)_{1,3}$; then, for a matrix $V = [v_1\ v_2\ \cdots\ v_p]$ of mode-2 vectors, we have

$$V^{\mathsf{T}}v = (V^{\mathsf{T}})_2 \cdot \left(\mathcal{A} \cdot (u, w)_{1,3}\right) = \mathcal{A} \cdot (u, V, w) \in \mathbb{R}^{1\times p\times 1}.$$

The following well-known result will be needed.

Lemma 1. *Let $\mathcal{A} \in \mathbb{R}^{l\times m\times n}$ be given along with three matrices with orthonormal columns, $U \in \mathbb{R}^{l\times p}$, $V \in \mathbb{R}^{m\times q}$, and $W \in \mathbb{R}^{n\times r}$, where $p \le l$, $q \le m$, and $r \le n$. Then the least squares problem*

$$\min_{\mathcal{S}} \|\mathcal{A} - (U, V, W) \cdot \mathcal{S}\|$$

has the unique solution

$$\mathcal{S} = \left(U^{\mathsf{T}}, V^{\mathsf{T}}, W^{\mathsf{T}}\right) \cdot \mathcal{A} = \mathcal{A} \cdot (U, V, W)\,.$$

The elements of the tensor $\mathcal{S}$ are given by

$$s_{\lambda\mu\nu} = \mathcal{A} \cdot (u_\lambda, v_\mu, w_\nu)\,, \quad 1 \le \lambda \le p, \quad 1 \le \mu \le q, \quad 1 \le \nu \le r. \tag{11}$$

Proof. The proof is a straightforward generalization of the corresponding proof for matrices. Enlarge each of the matrices so that it becomes square and orthogonal, i.e., put

$$\bar{U} = [U\ U_\perp], \quad \bar{V} = [V\ V_\perp], \quad \bar{W} = [W\ W_\perp].$$

Introducing the residual $\mathcal{R} = \mathcal{A} - (U, V, W) \cdot \mathcal{S}$ and using the invariance of the norm under orthogonal transformations, we get

$$\|\mathcal{R}\|^2 = \left\|\mathcal{R} \cdot \left(\bar{U}, \bar{V}, \bar{W}\right)\right\|^2 = \|\mathcal{A} \cdot (U, V, W) - \mathcal{S}\|^2 + C^2,$$

where $C^2 = \|\mathcal{A} \cdot (U_\perp, V_\perp, W_\perp)\|^2$ does not depend on $\mathcal{S}$. The relation (11) is obvious from the definition of the tensor-matrix multiplication. □

The inner product (9) can be considered as a special case of the *contracted product of two tensors*, cf. [18, Chapter 2], which is a tensor (outer) product followed by a contraction along specified modes. Thus, if $\mathcal{A}$ and $\mathcal{B}$ are 3-tensors, we define, using essentially the notation of [2],

[3] Note that x and y may correspond to mode-1 (columns), mode-2 (rows), or mode-3 (i.e., $1 \times 1 \times n$ dimensional tensors) vectors.

$$\mathcal{C} = \langle \mathcal{A}, \mathcal{B} \rangle_1 \ , \qquad c_{jkj'k'} = \sum_{\alpha} a_{\alpha jk} b_{\alpha j'k'} \ , \qquad \text{(4-tensor)} \, , \tag{12.a}$$

$$D = \langle \mathcal{A}, \mathcal{B} \rangle_{1,2} \ , \qquad d_{kk'} = \sum_{\alpha,\beta} a_{\alpha\beta k} b_{\alpha\beta k'} \ , \qquad \text{(2-tensor)}, \tag{12.b}$$

$$e = \langle \mathcal{A}, \mathcal{B} \rangle = \langle \mathcal{A}, \mathcal{B} \rangle_{1:3} \ , \qquad e = \sum_{\alpha,\beta,\gamma} a_{\alpha\beta\gamma} b_{\alpha\beta\gamma} \, , \qquad \text{(scalar)}. \tag{12.c}$$

It is required that the contracted dimensions are equal in the two tensors. We will refer to the first two as *partial contractions*. The subscript 1 in $\langle \mathcal{A}, \mathcal{B} \rangle_1$ and 1,2 in $\langle \mathcal{A}, \mathcal{B} \rangle_{1,2}$ indicate that the contraction is over the first mode in both arguments and in the first and second mode in both arguments, respectively. It is also convenient to introduce a notation when contraction is performed in all but one mode. For example the product in (12.b) may also be written

$$\langle \mathcal{A}, \mathcal{B} \rangle_{1,2} \equiv \langle \mathcal{A}, \mathcal{B} \rangle_{-3} \ . \tag{13}$$

The definition of contracted products is valid also when the tensors are of different order. The only assumption is that the dimension of the correspondingly contracted modes are the same in the two arguments. The dimensions of the resulting product are in the order given by the non-contracted modes of the first argument followed by the non-contracted modes of the second argument.

3. Two Krylov Methods for Matrices

In this section we will describe briefly the two matrix Krylov methods that are the starting point of our generalization to tensor Krylov methods.

3.1. The Arnoldi Procedure

The Arnoldi procedure is used to compute a low-rank approximation of a square, in general non-symmetric, matrix A. It requires a starting vector $u_1 =: U_1$ (or, alternatively, $v_1 =: V_1$), and in each step the new vector is orthogonalized against all previous vectors using the modified Gram–Schmidt process. We present the Arnoldi procedure in the style of [29, p. 303].

Algorithm 1 The Arnoldi Procedure

for $i = 1, 2, \ldots, k$ **do**

1 $h_i = U_i^{\mathsf{T}} A u_i$

2 $h_{i+1,i} u_{i+1} = A u_i - U_i h_i$

3 $U_{i+1} = [U_i \ u_{i+1}]$

4 $H_i = \begin{bmatrix} H_{i-1} & h_i \\ 0 & h_{i+1,i} \end{bmatrix}$

end for

The coefficient $h_{i+1,i}$ is used to normalize the new vector to length one. Note that the matrix H_k in the the factorization (2) is obtained by collecting the orthonormalization coefficients h_i and $h_{i+1,i}$ in an upper Hessenberg matrix.

3.2. Golub–Kahan Bidiagonalization

Let $A \in \mathbb{R}^{m\times n}$ be a matrix, and let $\beta_1 u_1, v_0 = 0$, where $\|u_1\| = 1$, be starting vectors. The Golub–Kahan bidiagonalization procedure [11] is defined by the following recursion.

Algorithm 2 Golub–Kahan bidiagonalization

for $i = 1, 2, \ldots, k$ **do**
1 $\alpha_i v_i = A^{\mathsf{T}} u_i - \beta_i v_{i-1}$
2 $\beta_{i+1} u_{i+1} = A v_i - \alpha_i u_i$
end for

The scalars α_i, β_i are chosen to normalize the generated vectors v_i and u_{i+1}. Forming the matrices $U_{k+1} = [u_1 \cdots u_{k+1}] \in \mathbb{R}^{m\times(k+1)}$ and $V_k = [v_1 \cdots v_k] \in \mathbb{R}^{n\times k}$, it is straightforward to show that

$$AV_k = U_{k+1}B_{k+1}, \qquad A^{\mathsf{T}}U_k = V_k\widehat{B}_k, \tag{14}$$

where $V_k^{\mathsf{T}}V_k = I$, $U_{k+1}^{\mathsf{T}}U_{k+1} = I$, and

$$B_{k+1} = \begin{bmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_k & \alpha_k \\ & & & \beta_{k+1} \end{bmatrix} = \begin{bmatrix} \widehat{B}_k \\ \beta_{k+1} e_k^{\mathsf{T}} \end{bmatrix} \in \mathbb{R}^{(k+1)\times k} \tag{15}$$

is bidiagonal[4].

Using tensor notation from Section 2.2, and a special case of the identification (7), we may equivalently express the Golub–Kahan bidiagonalization as in Algorithm 3.

Algorithm 3 Golub–Kahan bidiagonalization in tensor notation

for $i = 1, 2, \ldots, k$ **do**
1 $\alpha_i v_i = A \cdot (u_i)_1 - \beta_i v_{i-1}$
2 $\beta_{i+1} u_{i+1} = A \cdot (v_i)_2 - \alpha_i u_i$
end for

We observe that the u_i vectors "live" in the first mode of A, and we generate the sequence $u_2, u_3, \ldots$, by multiplication of the v_i vectors in the second mode, and vice versa.

4. Tensor Krylov Methods

4.1. A Minimal Krylov Recursion

In this subsection we will present the main process for the tensor Krylov methods. We will further prove that, for tensors with $\operatorname{rank}(\mathcal{A}) = (p, q, r)$, we can capture all three

[4]Note that the two sequences of vectors become orthogonal automatically; this is due to the fact that the bidiagonalization procedure is equivalent to the Lanczos process applied to the two symmetric matrices AA^{T} and $A^{\mathsf{T}}A$.

subspaces associated to $\mathcal{A}$ with $p+q+r$ tensor-vector-vector multiplications. Finally we will state a number of product relations that are induced by the procedure.

Let $\mathcal{A} \in \mathbb{R}^{l\times m\times n}$ be a given third order tensor. Starting from Algorithm 3, it is now straightforward to generalize the Golub–Kahan procedure. Assuming we have two starting vectors, $u_1 \in \mathbb{R}^l$ and $v_1 \in \mathbb{R}^m$, we can obtain a third mode vector by $w_1 = \mathcal{A} \cdot (u_1, v_1)_{1,2} \in \mathbb{R}^n$. We can then generate three sequences of vectors

$$u_{i+1} = \mathcal{A} \cdot (v_i, w_i)_{2,3}\,, \tag{16}$$

$$v_{i+1} = \mathcal{A} \cdot (u_i, w_i)_{1,3}\,, \tag{17}$$

$$w_{i+1} = \mathcal{A} \cdot (u_i, v_i)_{1,2}\,, \tag{18}$$

for $i = 1, \ldots, k$. We see that the first mode sequence of vectors (u_{i+1}) are generated by multiplication of second and third mode vectors (v_i) and (w_i) by the tensor $\mathcal{A}$, and similarly for the other two sequences. The newly generated vector is immediately orthogonalized against all the previous ones in its mode, using the modified Gram–Schmidt process. An obvious alternative to (17) and (18), that is consistent with the Golub–Kahan recursion, is to use the most recent vectors in computing the new one. This recursion is presented in Algorithm 4. In the algorithm description it is understood that $U_i = [u_1\, u_2\, \cdots\, u_i]$, etc. The coefficients α_u, α_v, and α_w are used to normalize the generated vectors to length one. The difference in the two alternative recursions is in the sense that the updates in (16)–(18) constitute a "Jacobi type" of iterative process, while the updates in Algorithm 4 constitute a "Gauss–Seidel" type of iterative process. It is not clear which of the two minimal Krylov recursions is to be preferred as they seem to have similar performance. For reasons that will become clear later, we will refer to any one of these two recursions as a minimal Krylov recursion.

Algorithm 4 Minimal Krylov recursion

Given: two normalized starting vectors u_1 and v_1,
$\alpha_w w_1 = \mathcal{A} \cdot (u_1, v_1)_{1,2}$
for $i = 1, 2, \ldots, k-1$ **do**
$\widehat{u} = \mathcal{A} \cdot (v_i, w_i)_{2,3}\,; \quad h_u = U_i^{\mathsf{T}} \widehat{u}$
$\alpha_u u_{i+1} = \widehat{u} - U_i h_u; \quad H_{i+1}^u = \begin{bmatrix} H_i^u & h_u \\ 0 & \alpha_u \end{bmatrix}$
$\widehat{v} = \mathcal{A} \cdot (u_{i+1}, w_i)_{1,3}\,; \quad h_v = V_i^{\mathsf{T}} \widehat{v}$
$\alpha_v v_{i+1} = \widehat{v} - V_i h_v; \quad H_{i+1}^v = \begin{bmatrix} H_i^u & h_v \\ 0 & \alpha_v \end{bmatrix}$
$\widehat{w} = \mathcal{A} \cdot (u_{i+1}, v_{i+1})_{1,2}\,; \quad h_w = W_i^{\mathsf{T}} \widehat{w}$
$\alpha_w w_{i+1} = \widehat{w} - W_i h_w; \quad H_{i+1}^w = \begin{bmatrix} H_i^w & h_w \\ 0 & \alpha_w \end{bmatrix}$
end for

The minimal Krylov recursion may break down, i.e., we obtain a new vector u_{i+1}, for instance, which is a linear combination of the vectors in U_i. This can happen in two principally different situations. The first one is when, for example, the vectors in U_i span the range space of $A^{(1)}$. If this is the case we are done generating u-vectors. The

second case is when we get a "true breakdown"[5], i.e., u_{i+1} is a linear combination of vectors in U_i, but U_i does not span the entire range space of $A^{(1)}$. This can be fixed by taking a vector $u_{i+1} \perp U_i$ with u_{i+1} in range of $A^{(1)}$. More details regarding the various breakdowns and how to deal with them numerically will be addressed in future research.

4.1.1. Tensors with Cubical Ranks

Assume that the $l \times m \times n$ tensor has a cubical low rank, i.e., $\operatorname{rank}(\mathcal{A}) = (r, r, r)$ with $r \leq \min(l, m, n)$. Then there exist a tensor $\mathcal{C} \in \mathbb{R}^{r \times r \times r}$, and full column rank matrices X, Y, and Z such that $\mathcal{A} = (X, Y, Z) \cdot \mathcal{C}$.

We will now prove that, when the starting vectors u_1, v_1, and w_1 are in the range of the respective subspaces, the minimal Krylov procedure generates matrices U, V, W, such that $\operatorname{span}(U) = \operatorname{span}(X)$, $\operatorname{span}(V) = \operatorname{span}(Y)$ and $\operatorname{span}(W) = \operatorname{span}(Z)$ after r steps. Of course, for the low multilinear rank approximation problem of tensors it is the subspaces that are important, not their actual representation. The specific basis for, e.g., $\operatorname{span}(X)$, is ambiguous.

Theorem 2. *Let $\mathcal{A} = (X, Y, Z) \cdot \mathcal{C} \in \mathbb{R}^{l \times m \times n}$ with $\operatorname{rank}(\mathcal{A}) = (r, r, r)$. Assume we have starting vectors in the associated range spaces, i.e., $u_1 \in \operatorname{span}(X)$, $v_1 \in \operatorname{span}(Y)$, and $w_1 \in \operatorname{span}(Z)$. Assume also that the process does not break down[6] within r iterations. Then, the minimal Krylov procedure in Algorithm 4 generates matrices U_r, V_r, W_r with*

$$\operatorname{span}(U_r) = \operatorname{span}(X), \quad \operatorname{span}(V_r) = \operatorname{span}(Y), \quad \operatorname{span}(W_r) = \operatorname{span}(Z).$$

Proof. First observe that the recursion generates vectors in the span of X, Y, and Z, respectively:

$$\begin{aligned}
\mathcal{A} \cdot (v, w)_{2,3} &= \mathcal{C} \cdot \left(X^{\mathsf{T}}, Y^{\mathsf{T}} v, Z^{\mathsf{T}} w\right) = \mathcal{C} \cdot \left(X^{\mathsf{T}}, \bar{v}, \bar{w}\right) = X c_1, \\
\mathcal{A} \cdot (u, w)_{1,3} &= \mathcal{C} \cdot \left(X^{\mathsf{T}} u, Y^{\mathsf{T}}, Z^{\mathsf{T}} w\right) = \mathcal{C} \cdot \left(\bar{u}, Y^{\mathsf{T}}, \bar{w}\right) = Y c_2, \\
\mathcal{A} \cdot (u, v)_{1,2} &= \mathcal{C} \cdot \left(X^{\mathsf{T}} u, Y^{\mathsf{T}} v, Z^{\mathsf{T}}\right) = \mathcal{C} \cdot \left(\bar{u}, \bar{v}, Z^{\mathsf{T}}\right) = Z c_3,
\end{aligned}$$

where in the first equation $\bar{v} = Y^{\mathsf{T}} v$, $\bar{w} = Z^{\mathsf{T}} w$ and $c_1 = \mathcal{C} \cdot (\bar{v}, \bar{w})_{2,3}$. The other two equations are analogous. Consider the first mode vector u. Clearly it is a linear combination of the column vectors in X. Since we orthonormalize every newly generated u-vector against all the previous vectors, and since we assume that the process does not break down, it follows that, for $k \leq r$, $\dim(\operatorname{span}([u_1 \cdots u_k])) = k$ will increase by one with every new u-vector. Given that $u_1 \in \operatorname{span}(X)$ then, for $k = r$, we have that $\operatorname{span}([u_1 \cdots u_r]) = \operatorname{span}(X)$. The proof is analogous for the second and third modes. □

Remark (1). It is straightforward to show that when the starting vectors are not in the associated range spaces we would only need to do one more iteration, i.e., in total $r + 1$ iterations, to obtain matrices U_{r+1}, V_{r+1}, and W_{r+1}, that would contain the column spaces of X, Y, and Z, respectively.

[5] In the matrix case a breakdown occurs in the Krylov recursion for instance if the matrix and the starting vector have the structure

$$A = \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix}, \qquad v = \begin{bmatrix} v_1 \\ 0 \end{bmatrix}.$$

An analogous situation can occur with tensors.

[6] The newly generated vector is not a linear combination of previously generated vectors.

Remark (2). It is easy to obtain starting vectors $u_1 \in \text{span}(X)$, $v_1 \in \text{span}(Y)$, and $w_1 \in \text{span}(Z)$: choose any single nonzero mode-k vector or the mean of the mode-k vectors.

Remark (3). The situation is not much different when the starting vectors are not in the range spaces of X, Y, and Z. First observe that we may write

$$\mathcal{A} \cdot (v, w) = A^{(1)}(v \otimes w) = X^{\mathsf{T}} C^{(1)}(Y \otimes Z)(v \otimes w),$$

where $A^{(1)} \in \mathbb{R}^{l \times mn}$ denotes the matricization or unfolding of $\mathcal{A}$ along the first mode, similarly for $C^{(1)}$. Further, since $\text{rank}(\mathcal{A}) = (r, r, r)$, it easy to show that the rank of the matrix $A^{(1)}$ is r. This means that at most r linearly independent first mode vectors can be generated by multiplying $\mathcal{A}$ with vectors along the second and third mode. Taking into account that the starting vector $u \notin \text{span}(X)$ we realize that $\text{span}(X) \in \text{span}(U_{r+1})$, i.e., in $r+1$ iterations (assuming the process does not bread down) we will obtain a matrix that contains the desired subspaces. In a subsequent step it is easy to extract a matrix $\tilde{U}$ with r columns spanning the correct subspace.

4.1.2. Tensors with General Low Multilinear Rank

Next we discuss the case when the tensor $\mathcal{A} \in \mathbb{R}^{l \times m \times n}$ has $\text{rank}(\mathcal{A}) = (p, q, r)$ with $p < l$, $q < m$, and $r < n$. Without loss of generality we can assume $p \leq q \leq r$. Then $\mathcal{A} = (X, Y, Z) \cdot \mathcal{C}$ where $\mathcal{C}$ is a $p \times q \times r$ tensor and X, Y, and Z are full column rank matrices with conformal dimensions. The discussion assumes exact arithmetic and that no breakdown occurs unless we have a set of vectors that span the full range of the different modes.

From the proof of Theorem 2 we see that the vectors generated are in the span of X, Y, and Z, respectively. Therefore, after having performed p steps, we will not be able to generate any new vectors, that would increase the dimension of the first mode subspace beyond p. This can be detected from the fact that the result of the orthogonalization is zero. We can now continue generating vectors in the second and third modes, using any of the first mode vectors, or a (possibly random) linear combination of them[7]. This can be repeated until we have generated q vectors in the second and third modes. The final $r - q$ mode-3 vectors can then be generated using combinations of mode-1 and mode-2 vectors that have not been used before, or, alternatively, random linear combinations of previously generated mode-1 and mode-2 vectors. We refer to the procedure described in this paragraph as the *modified minimal Krylov recursion.*

The use of random linear combinations may be motivated from the desire to extract "new information" or produce vectors that "enlarge" the currently available subspaces in new directions. For example, assume that we only need to generate mode-3 vectors using tensor-vector-vector multiplications of the form $\mathcal{A} \cdot (u, v)_{1,2}$ for some u and v. Then, using the same u (or using the same linear combination of all mode-1 vectors) will restrict the possible subspaces that can be extracted just by varying the v vector. Taking new random linear combinations of all mode-1 vectors and all mode-2 vectors, when generating a new mode-3 vector w, increases the probability that w is not a linear combination of previously generated mode-3 vectors. Obviously, if the third mode rank

[7] Also the optimization approach of Section 4.3 can be used.

of the tensor is r, then we can only generate r linearly independent mode-3 vectors in total, regardless of how we choose the mode-1 and mode-2 vectors.

Theorem 3. *Let $\mathcal{A} \in \mathbb{R}^{l\times m\times n}$ be a tensor of* $\operatorname{rank}(\mathcal{A}) = (p, q, r)$ *with $p \le q \le r$. We can then write $\mathcal{A} = (X, Y, Z) \cdot \mathcal{C}$, where $\mathcal{C}$ is a $p \times q \times r$ tensor and X, Y, and Z are full column rank matrices with conforming dimensions. Assume that the starting vectors satisfy $u_1 \in \operatorname{span}(X)$, $v_1 \in \operatorname{span}(Y)$, and $w_1 \in \operatorname{span}(Z)$. Assume also that the process does not break down except when we obtain a set of vectors spanning the full range spaces of the different modes. Then in exact arithmetic, and in a total of r steps, the modified minimal Krylov recursion produces matrices U_p, V_q, and W_r, which span the same subspaces as X, Y, and Z, respectively. Counting the number of tensor-vector-vector multiplications yields $p + q + r$.*

The actual numerical implementation of the procedure in floating point arithmetic is, of course, more complicated. For instance, the ranks will never be exact, so one must devise a criterion for determining the numerical ranks, which will depend on the choice of tolerances. This will be the topic of our future research.

4.1.3. Relations from the Minimal Krylov Recursion

In general it is not possible to write the minimal Krylov recursion directly as a tensor Krylov decomposition, analogous to (14). However, having generated three orthonormal matrices U_k, V_k, and W_k, we can easily compute a low-rank tensor approximation of $\mathcal{A}$ using Lemma 1,

$$\mathcal{A} \approx (U_k, V_k, W_k) \cdot \mathcal{H}, \qquad \mathcal{H} = \left(U_k^{\mathsf{T}}, V_k^{\mathsf{T}}, W_k^{\mathsf{T}}\right) \cdot \mathcal{A} \in \mathbb{R}^{k\times k\times k}. \tag{19}$$

Obviously, $\mathcal{H}_{\lambda\mu\nu} = \mathcal{A} \cdot (u_\lambda, v_\mu, w_\nu)$. Comparing with Algorithm 4 we will show that $\mathcal{H}$ contains elements identical to the elements of the Hessenberg matrices[8] H_k^u, H_k^v, and H_k^w. For example, the entries of the i'th column of H_k^u are identical to $\mathcal{H}(:, i, i)$ for $i = 1, \cdots, k-1$. As a result certain product relations hold between the matrices U_k, V_k, W_k (that are generated with the minimal tensor Krylov procedure), the original tensor A, and $\mathcal{H}$. The details are given in the following proposition.

Proposition 4. *Assume that U_k, V_k, and W_k have been generated by the minimal Krylov recursion in Algorithm 4 and that $\mathcal{H} = \mathcal{A} \cdot (U_k, V_k, W_k)$. Then, for $1 \le i \le k-1$,*

$$(\mathcal{A} \cdot (V_k, W_k)_{2,3})(:, i, i) = ((U_k)_1 \cdot \mathcal{H})(:, i, i) = U_k H_k^u(:, i), \tag{20}$$

$$(\mathcal{A} \cdot (U_k, W_k)_{1,3})(i+1, :, i) = ((V_k)_2 \cdot \mathcal{H})(i+1, :, i) = V_k H_k^v(:, i), \tag{21}$$

$$(\mathcal{A} \cdot (U_k, V_k)_{1,2})(i+1, i+1, :) = ((W_k)_3 \cdot \mathcal{H})(i+1, i+1, :) = W_k H_k^w(:, i). \tag{22}$$

Proof. Let $1 \le i \le k-1$ and consider the fiber

$$\mathcal{H}(:, i, i) = [h_{1ii} \;\; h_{2ii} \;\; \cdots \;\; h_{kii}]^{\mathsf{T}}.$$

[8] Recall from Algorithm 4 that H_k^u contains orthogonalization and normalization coefficients for the first mode vectors. Similarly, H_k^v, and H_k^w contain orthogonalization and normalization coefficients for the second and third mode vectors, respectively.

From the minimal recursion we have that

$$\mathcal{A} \cdot (v_i, w_i)_{2,3} = \sum_{\lambda=1}^{i+1} h_{\lambda ii} u_\lambda = U_{i+1} H^u_{i+1}(:, i).$$

Then for $i + 2 \leq s \leq k$,

$$h_{sii} = \mathcal{A} \cdot (u_s, v_i, w_i) = \left(u_s^{\mathsf{T}}\right)_1 \cdot (\mathcal{A} \cdot (v_i, w_i)_{2,3}) = 0.$$

Thus $h_{i+2,ii} = \ldots = h_{kii} = 0$. We now observe that the left hand side of (20) is exactly $\mathcal{A} \cdot (v_i, w_i)_{2,3}$, which can be written as $U_k H^u_k(:, i)$, since the normalization coefficients $H^u_k(:, i) = \mathcal{H}(:, i, i)$ and only first $i + 1$ of them are non-zero. The rest of the proof is analogous. □

If the sequence of vectors is generated according to Equations (16)–(18), then a similar (and simpler) proposition will hold. For example we would have

$$(\mathcal{A} \cdot (U_k, W_k)_{1,3})(i, :, i) = (V_k)_2 \cdot \mathcal{H}(i, :, i) = V_k H^v_k(:, i), \quad i = 1, \ldots, k.$$

4.2. A Maximal Krylov Recursion

Note that when a new vector u_{i+1} is generated in the minimal Krylov procedure, then we use the most recently computed v_i and w_i. In fact, we might choose any combination of previously computed v_j and w_k with $1 \leq j, k \leq i$, that have not been used before to generate a u-vector. Let $j \leq i$ and $k \leq i$, and consider the computation of a new u-vector, which we may write as

$$h_u = U_i^{\mathsf{T}} (\mathcal{A} \cdot (v_j, w_k)_{2,3}),$$
$$h_{*jk} u_{i+1} = \mathcal{A} \cdot (v_j, w_k)_{2,3} - U_i h_u.$$

Thus, if we are prepared to use all previously computed v- and w-vectors, then we have a much richer combinatorial structure, which we illustrate in the following diagram. Assume that u_1 and v_1 are given. In the first steps of the maximal Krylov procedure the following vectors can be generated by combining previous vectors.

1:	$\{u_1\} \times \{v_1\}$	$\longrightarrow$	w_1
2:	$\{v_1\} \times \{w_1\}$	$\longrightarrow$	u_2
3:	$\{u_1, u_2\} \times \{w_1\}$	$\longrightarrow$	$\{v_2, v_3\}$
4:	$\{u_1, u_2\} \times \{v_1, v_2, v_3\}$	$\longrightarrow$	$\{(w_1), w_2, w_3, w_4, w_5, w_6\}$
5:	$\{v_1, v_2, v_3\} \times \{w_1, w_2, \ldots, w_6\}$	$\longrightarrow$	$\{(u_2), u_3, \ldots, u_{19}\}$
6:	$\{u_1, u_2, \ldots, u_{19}\} \times \{w_1, w_2, \ldots, w_6\}$	$\longrightarrow$	$\{(v_2), (v_3), v_4, \ldots, v_{115}\}$

Vectors computed at a previous step are within parentheses. Of course, we can only generate new orthogonal vectors as long as the total number of vectors is smaller than the dimension of that mode. Further, if, at a certain stage in the procedure, we have generated α and β vectors in two modes, then we can generate altogether $\gamma = \alpha\beta$ vectors

in the third mode (where we do not count the starting vector in that mode, if there was one).

We will now describe the first three steps in some detail. Let u_1 and v_1 be length one starting vectors in the first and second mode, respectively. We will store the normalization and orthogonalization coefficients in a tensor $\mathcal{H}$, whose dimensions will increase during the process in a similar fashion as the dimensions of the Hessenberg matrix increase during the Arnoldi process. The entries of $\mathcal{H}$ will be denoted with $h_{ijk} = \mathcal{H}(i,j,k)$. Also when subscripts are given, they will indicate the dimensions of the tensor, e.g., $\mathcal{H}_{211}$ will be a $2 \times 1 \times 1$ tensor.

Step (1). In the first step we generate an new third mode vector by computing

$$\mathcal{A} \cdot (u_1, v_1)_{1,2} = h_{111} w_1 = (w_1)_3 \cdot \mathcal{H}_{111}, \tag{23}$$

where $h_{111} = \mathcal{H}_{111}$ is a normalization constant.

Step (2). Here we compute a new first mode vector;

$$\widehat{u}_2 = \mathcal{A} \cdot (v_1, w_1)_{2,3}\,.$$

The orthogonalization coefficient satisfies

$$u_1^{\mathsf{T}} \widehat{u}_2 = u_1^{\mathsf{T}} (\mathcal{A} \cdot (v_1, w_1)_{2,3}) = \mathcal{A} \cdot (u_1, v_1, w_1) = w_1^{\mathsf{T}} (\mathcal{A} \cdot (u_1, v_1)_{1,2}) = h_{111}, \tag{24}$$

from (23). After orthogonalization and normalization,

$$h_{211} u_2 = \widehat{u}_2 - h_{111} u_1, \tag{25}$$

and rearranging the terms in (25), we have the following relation

$$\mathcal{A} \cdot (v_1, w_1)_{2,3} = ([u_1,\, u_2])_1 \cdot \mathcal{H}_{211}, \qquad \mathcal{H}_{211} = \begin{bmatrix} h_{111} \\ h_{211} \end{bmatrix}.$$

Step (3). In the third step we obtain two second mode vectors. To get v_2 we compute

$$\widehat{v}_2 = \mathcal{A} \cdot (u_1, w_1)_{1,3}\,, \qquad h_{121} v_2 = \widehat{v}_2 - h_{111} v_1;$$

the orthogonalization coefficient becomes h_{111} using an argument analogous to that in (24).

Combining u_2 with w_1 will yield v_3 as follows; first we compute

$$\widehat{v}_3 = \mathcal{A} \cdot (u_2, w_1)_{1,3}\,,$$

and orthogonalize

$$v_1^{\mathsf{T}} \widehat{v}_3 = \mathcal{A} \cdot (u_2, v_1, w_1) = u_2^{\mathsf{T}} (\mathcal{A} \cdot (v_1, w_1)_{2,3}) = u_2^{\mathsf{T}} \widehat{u}_2 = h_{211}.$$

We see from (25) that h_{211} is already computed. The second orthogonalization becomes

$$v_2^{\mathsf{T}} \widehat{v}_3 = \mathcal{A} \cdot (u_2, v_2, w_1) =: h_{221}.$$

Then

$$h_{231}v_3 = \widehat{v}_3 - h_{211}v_1 - h_{221}v_2.$$

After a completed third step we have a new relation

$$\mathbb{R}^{2\times m\times 1} \ni \mathcal{A} \cdot (U_2, w_1)_{1,3} = (V_3)_2 \cdot \mathcal{H}_{231}, \quad \mathcal{H}_{231} = \begin{bmatrix} h_{111} & h_{121} & 0 \\ h_{211} & h_{221} & h_{231} \end{bmatrix},$$

where $U_2 = [u_1\, u_2]$ and $V_3 = [v_1\, v_2\, v_3]$. Note that the orthogonalization coefficients are given by

$$h_{\lambda\mu\nu} = \mathcal{A} \cdot (u_\lambda, v_\mu, w_\nu)\,.$$

This maximal procedure is presented in Algorithm 5. The algorithm has three main loops, and it is maximal in the sense that in each such loop we generate as many new vectors as can be done, before proceeding to the next main loop. Consider the u-loop (the other loops are analogous). The vector h_α is a mode-1 vector[9] and contains orthogonalization coefficients with respect to u-vectors computed at previous steps. These coefficients are values of the tensor $\mathcal{H}$. The vector h_i on the other hand contains orthogonalization coefficients with respect to u-vectors that are computed within the current step. Its dimension is equal to the current number of vectors in U. The coefficients h_i together with the normalization constant $h_{\alpha+1,\bar{\beta},\bar{\gamma}}$ of the newly generated vector $u_{\alpha+i}$ are appended at the appropriate positions of the tensor $\mathcal{H}$. Specifically the coefficients for the u-vector obtained using $v_{\bar{\beta}}$ and $w_{\bar{\gamma}}$ are stored as first mode fiber, i.e. $\mathcal{H}(:, \bar{\beta}, \bar{\gamma}) = [h_\alpha^\mathsf{T}\ \ h_i^\mathsf{T}\ \ h_{\alpha+i,\bar{\beta},\bar{\gamma}}]^\mathsf{T}$. Since the number of vectors in U are increasing for every new u-vector the dimension of $[h_\alpha^\mathsf{T}\ \ h_i^\mathsf{T}\ \ h_{\alpha+i,\bar{\beta},\bar{\gamma}}]^\mathsf{T}$ and thus the dimension of $\mathcal{H}$ along the first mode increases by one as well. The other mode-1 fibers of $\mathcal{H}$ are filled out with a zero at the bottom. Continuing with the v-loop, the dimension of the coefficient tensor $\mathcal{H}$ increases in the second mode.

It is clear that $\mathcal{H}$ has a zero-nonzero structure that resembles that of a Hessenberg matrix. If we break the recursion after any complete outer **for all**-statement, we can form a relation denote as *tensor Krylov decomposition*, which generalizes the matrix Krylov decomposition. First we formally state these definitions.

Definition 4.1. (Matrix Krylov decomposition [29, p. 309]) Let $A \in \mathbb{R}^{n\times n}$ be given and $u_1, \cdots, u_{k+1}$ be a set of linearly independent vectors. Then, the relation

$$AU_k = U_{k+1}B_{k+1} = U_k\widehat{B}_k + u_{k+1}b_{k+1}^T, \qquad B_{k+1} = \begin{bmatrix} \widehat{B}_k \\ b_{k+1}^T \end{bmatrix}$$

where B_{k+1} is a $(k+1)\times k$ matrix, is defined to be a Krylov decomposition or order k.

From the definition it is easy to show (by induction) that $u_1, \cdots, u_k$ constitute a basis of the Krylov subspace (1). For a rectangular matrix one may consider the relation (14) as a Krylov decomposition. We will generalize that to tensors.

Definition 4.2 (Tensor Krylov decomposition)**.** Let $\mathcal{A} \in \mathbb{R}^{l\times m\times n}$ be given. Let also $u_1, \cdots, u_\alpha$; $v_1, \cdots, v_\beta$; and $w_1, \cdots, w_\gamma$ be three sets of linearly independent vectors. A relation of the form

$$\mathcal{A} \cdot (V_\beta, W_\gamma)_{2,3} = (U_\alpha)_1 \cdot \mathcal{B}_{\alpha\beta\gamma},$$

[9] We here refer to the identification (7).

Algorithm 5 The Maximal Krylov recursion

u_1 and v_1 are given starting vectors of length one
$h_{111}w_1 = \mathcal{A} \cdot (u_1, v_1)_{1,2}$
$\alpha = \beta = \gamma = 1$, $U_\alpha = u_1$, $V_\beta = v_1$ and $W_\gamma = w_1$
while $\alpha \leq \alpha_{\max}$ and $\beta \leq \beta_{\max}$ and $\gamma \leq \gamma_{\max}$ **do**
  %——————— *u-loop* ———————%
  $U_\alpha = [u_1 \ \dots \ u_\alpha]$, $U = [\,]$, $V_\beta = [v_1 \ \dots \ v_\beta]$, $W_\gamma = [w_1 \ \dots \ w_\gamma]$, $i = 1$
  for all $(\bar{\beta}, \bar{\gamma})$ such that $\bar{\beta} \leq \beta$ and $\bar{\gamma} \leq \gamma$ **do**
    if the pair $(\bar{\beta}, \bar{\gamma})$ has not been used before **then**
      $h_\alpha = \mathcal{H}(1{:}\alpha, \bar{\beta}, \bar{\gamma})$
      $h_i = \mathcal{A} \cdot \big(U, v_{\bar{\beta}}, w_{\bar{\gamma}}\big)$
      $h_{\alpha+i,\bar{\beta}\bar{\gamma}} u_{\alpha+i} = \mathcal{A} \cdot (v_{\bar{\mu}}, w_{\bar{\lambda}})_{2,3} - U_\alpha h_\alpha - U h_i$
      $\mathcal{H}(\alpha + 1{:}\alpha + i, \bar{\beta}, \bar{\gamma}) = [h_i^\mathsf{T} \ \ h_{\alpha+i,\bar{\beta}\bar{\gamma}}]^\mathsf{T}$
      $U = [U \ u_{\alpha+i}]$, $i = i + 1$
    end if
  end for
  $U_{\beta\gamma+1} = [U_\alpha \ U]$, $\alpha = \beta\gamma + 1$
  %——————— *v-loop* ———————%
  $U_\alpha = [u_1 \ \dots \ u_\alpha]$, $V_\beta = [v_1 \ \dots \ v_\beta]$, $V = [\,]$, $W_\gamma = [w_1 \ \dots \ w_\gamma]$, $j = 1$
  for all $(\bar{\alpha}, \bar{\gamma})$ such that $\bar{\alpha} \leq \alpha$ and $\bar{\gamma} \leq \gamma$ **do**
    if the pair $(\bar{\alpha}, \bar{\gamma})$ has not been used before **then**
      $h_\beta = \mathcal{H}(\bar{\alpha}, 1{:}\beta, \bar{\gamma})$
      $h_j = \mathcal{A} \cdot (u_{\bar{\alpha}}, V, w_{\bar{\gamma}})$
      $h_{\bar{\alpha},\beta+j,\bar{\gamma}} v_{\beta+j} = \mathcal{A} \cdot (u_{\bar{\alpha}}, w_{\bar{\gamma}})_{1,3} - V_\beta h_\beta - V h_j$
      $\mathcal{H}(\bar{\alpha}, \beta + 1{:}\beta + j, \bar{\gamma}) = [h_j^\mathsf{T} \ \ h_{\bar{\alpha},\beta+j,\bar{\gamma}}]^\mathsf{T}$
      $V = [V \ v_{\beta+j}]$, $j = j + 1$
    end if
  end for
  $V_{\alpha\gamma+1} = [V_\beta \ V]$, $\beta = \alpha\gamma + 1$
  %——————— *w-loop* ———————%
  $U_\alpha = [u_1 \ \dots \ u_\alpha]$, $V_\beta = [v_1 \ \dots \ v_\beta]$, $W_\gamma = [w_1 \ \dots \ w_\gamma]$, $W = [\,]$, $k = 1$
  for all $(\bar{\alpha}, \bar{\beta})$ such that $\bar{\alpha} \leq \alpha$ and $\bar{\beta} \leq \beta$ **do**
    if the pair $(\bar{\alpha}, \bar{\beta})$ has not been used before **then**
      $h_\gamma = \mathcal{H}(\bar{\alpha}, \bar{\beta}, 1{:}\gamma)$
      $h_k = \mathcal{A} \cdot \big(u_{\bar{\alpha}}, v_{\bar{\beta}}, W\big)$
      $h_{\bar{\alpha}\bar{\beta},\gamma+k} w_{\gamma+k} = \mathcal{A} \cdot \big(u_{\bar{\alpha}}, v_{\bar{\beta}}\big)_{1,2} - W_\gamma h_\gamma - W h_k$
      $\mathcal{H}(\bar{\alpha}, \bar{\beta}, \gamma + 1{:}\gamma + k) = [h_k^\mathsf{T} \ \ h_{\bar{\alpha}\bar{\beta},\gamma+k}]^\mathsf{T}$
      $W = [W \ w_{\gamma+k}]$, $k = k + 1$
    end if
  end for
  $W_{\alpha\beta} = [W_\gamma \ W]$, $\gamma = \alpha\beta$
end while

is defined to be a tensor Krylov decomposition of order (α, β, γ). Analogous definitions can be made with multiplications in the other modes.

We will now show that the maximal Krylov recursion naturally admits a tensor Krylov decomposition.

Theorem 5. *Let a tensor $\mathcal{A} \in \mathbb{R}^{l\times m\times n}$ and two starting vectors u_1 and v_1 be given. Assume that we have generated matrices with orthonormal columns using the maximal Krylov procedure of Algorithm 5, and a tensor $\mathcal{H}$ of orthonormalization coefficients. Assume that after a complete u-loop the matrices U_α, V_β, and W_γ, and the tensor $\mathcal{H}_{\alpha\beta\gamma} \in \mathbb{R}^{\alpha\times\beta\times\gamma}$, have been generated, where $\alpha \le l$, $\beta \le m$, and $\gamma \le n$. Then, we have the tensor Krylov decomposition*

$$\mathcal{A} \cdot (V_\beta, W_\gamma)_{2,3} = (U_\alpha)_1 \cdot \mathcal{H}_{\alpha\beta\gamma}. \tag{26}$$

Further, assume that after the following complete v-loop we have orthonormal matrices U_α, $V_{\bar\beta}$, W_γ, and the tensor $\mathcal{H}_{\alpha\bar\beta\gamma} \in \mathbb{R}^{\alpha\times\bar\beta\times\gamma}$ where $\bar\beta = \alpha\gamma + 1 > \beta$. Then, we have a new tensor Krylov decomposition

$$\mathcal{A} \cdot (U_\alpha, W_\gamma)_{1,3} = \left(V_{\bar\beta}\right)_2 \cdot \mathcal{H}_{\alpha\bar\beta\gamma}. \tag{27}$$

Similarly, after the following complete w-loop, we will have orthonormal matrices U_α, $V_{\bar\beta}$, $W_{\bar\gamma}$, and the tensor $\mathcal{H}_{\alpha\bar\beta\bar\gamma} \in \mathbb{R}^{\alpha\times\bar\beta\times\bar\gamma}$ where $\bar\gamma = \alpha\bar\beta > \gamma$. Then, again, we have have a tensor Krylov decomposition

$$\mathcal{A} \cdot \left(U_\alpha, V_{\bar\beta}\right)_{1,2} = (W_{\bar\gamma})_3 \cdot \mathcal{H}_{\alpha\bar\beta\bar\gamma}. \tag{28}$$

It also holds that $\mathcal{H}_{\alpha\beta\gamma} = \mathcal{H}_{\alpha\bar\beta\gamma}(1\!:\!\alpha, 1\!:\!\beta, 1\!:\!\gamma)$ and $\mathcal{H}_{\alpha\bar\beta\gamma} = \mathcal{H}_{\alpha\bar\beta\bar\gamma}(1\!:\!\alpha, 1\!:\!\bar\beta, 1\!:\!\gamma)$, i.e., all orthonormalization coefficients from the u-, v- and w-loops are stored in a single and common tensor $\mathcal{H}$, that grows in its dimensions as new vectors are generated.

Proof. We prove that (26) holds; the other two equations are analogous. Using the definition of matrix-tensor multiplication we see that $\mathcal{A}\cdot(V_\beta, W_\gamma)_{2,3}$ is a tensor in $\mathbb{R}^{l\times\beta\times\gamma}$, where the first mode fiber at position (j,k) with $j \le \beta$ and $k \le \gamma$ is given by $\widehat{u}_\lambda = \mathcal{A} \cdot (v_j, w_k)_{2,3}$ with $\lambda = (j-1)\gamma + k + 1$.

On the right hand side the corresponding first mode fiber $\mathcal{H}(:, j, k)$ is equal to

$$\begin{bmatrix} h_{1jk} \\ h_{2jk} \\ \vdots \\ h_{\lambda-1jk} \\ h_{\lambda jk} \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathcal{A} \cdot (u_1, v_j, w_k) \\ \mathcal{A} \cdot (u_2, v_j, w_k) \\ \vdots \\ \mathcal{A} \cdot (u_{\lambda-1}, v_j, w_k) \\ h_{\lambda jk} \\ \mathbf{0} \end{bmatrix}.$$

Thus we have

$$\widehat{u}_\lambda = \mathcal{A} \cdot (v_j, w_k)_{2,3} = \sum_{i=1}^{\lambda} h_{ijk} u_i,$$

which is the equation for computing u_λ in the algorithm. □

Let U_i and V_j be two matrices with orthonormal columns that have been generated by any tensor Krylov method (i.e., not necessarily a maximal one) with tensor $\mathcal{A}$. Assume

that we then generate a sequence of $k = ij$ vectors $(w_1, w_2, \ldots, w_k)$ as in the w-loop of the maximal method. We say that W_k is *maximal with respect to U_i and V_j*. From the proof of Theorem 5 we see that we then have a tensor Krylov decomposition of the type (28).

Corollary 6. *Assume that the column vectors in U_i, V_j, W_k have been generated by a tensor-Krylov procedure without breakdown, such that W_k is maximal with respect to U_i and V_j. Then*

$$\mathcal{A} \cdot (U_i, V_j)_{1,2} = (W_k)_3 \cdot \mathcal{H}_{ijk}, \qquad \mathcal{H}_{ijk} = \mathcal{A} \cdot (U_i, V_j, W_k)\,. \tag{29}$$

4.3. Optimized Minimal Krylov Recursion

In some applications it may be a disadvantage that the maximal Krylov method generates so many vectors in each mode. In addition, when applied as described in Section 4.2 it generates different numbers of vectors in the different modes. Therefore it is natural to ask whether one can modify the minimal Krylov recursion so that it uses "optimal" vectors in two modes for the generation of a vector in the third mode. Such procedures have recently been suggested in [12]. We will describe this approach in terms of the recursion of a vector in the third mode. The corresponding computations in first and second modes are analogous.

Assume that we have computed i vectors in the first two modes, for instance, and that we are about to compute w_i. Further, assume that we will use linear combinations of the vectors from the first and second modes, i.e., we compute

$$\widehat{w} = \mathcal{A} \cdot (U_i\theta, V_i\eta)_{1,2}\,,$$

where $\theta, \eta \in \mathbb{R}^i$ are yet to be specified. We want the new vector to enlarge the third mode subspace as much as possible. This is the same as requiring that w_i be as large (in norm) as possible under the constraint that it is orthogonal to the previous mode-3 vectors. Thus we want to solve

$$\begin{aligned} \max_{\theta,\eta} \|\widehat{w}\|, \quad \text{where} \quad & \widehat{w} = \mathcal{A} \cdot (U_i\theta, V_i\eta)_{1,2}\,, \\ & \widehat{w} \perp W_{i-1}, \quad \|\theta\| = \|\eta\| = 1, \quad \theta, \eta \in \mathbb{R}^i. \end{aligned} \tag{30}$$

The solution of this problem is obtained by computing the best rank-$(1,1,1)$ approximation $(\theta, \eta, \omega) \cdot \mathcal{S}$ of the tensor

$$\mathcal{C}_w = \mathcal{A} \cdot \left(U_i, V_i, I - W_{i-1}W_{i-1}^{\mathsf{T}}\right). \tag{31}$$

A suboptimal solution can be obtained from the HOSVD of $\mathcal{C}_w$.

Recall the assumption that $\mathcal{A} \in \mathbb{R}^{l\times m\times n}$ is large and sparse. Clearly the optimization approach has the drawback that the tensor $\mathcal{C}_w$ is generally a dense tensor of dimension $i \times i \times n$, and the computation of the best rank-$(1,1,1)$ approximation or the HOSVD of that tensor can be quite time-consuming. Of course, in an application, where it is essential to have a good approximation of the tensor with as small dimensions of the subspaces as possible, it may be worth the extra computation needed for the optimization. However, we can avoid handling large, dense tensors by modifying the optimized recursion, so that

an approximation of the solution of the maximization problem (30) is computed using t steps of the minimal Krylov recursion on the tensor $\mathcal{C}_w$, for small t.

Assume that we have computed a rank-(t,t,t) approximation of $\mathcal{C}_w$,

$$\mathcal{C}_w \approx (\Theta, H, \Omega) \cdot \mathcal{S}_w,$$

for some small value of t, using the minimal Krylov method. By computing the best rank-$(1,1,1)$ approximation (or truncated HOSVD) of the small tensor $\mathcal{S}_w \in \mathbb{R}^{t\times t\times t}$, we obtain an approximation of the solution of (30). It remains to demonstrate that we can apply the minimal Krylov recursion to $\mathcal{C}_w$ without forming that tensor explicitly. Consider the computation of a vector ω in the third mode, given the vectors θ and η:

$$\begin{aligned}\widehat{\omega} = \mathcal{C}_w \cdot (\theta,\eta)_{1,2} &= \big(\mathcal{A}\cdot\big(U_i, V_i, I - W_{i-1}W_{i-1}^{\mathsf{T}}\big)\big)\cdot(\theta,\eta)_{1,2} \\ &= \Big(\mathcal{A}\cdot(U_i\theta, V_i\eta)_{1,2}\Big)\cdot\big(I - W_iW_i^{\mathsf{T}}\big)_3 = (I - W_iW_i^{\mathsf{T}})\tilde{\omega}.\end{aligned} \tag{32}$$

Note that the last matrix-vector multiplication is equivalent to the Gram–Schmidt orthogonalization in the minimal Krylov algorithm. Thus, we have only one sparse tensor-vector-vector operation, and a few matrix-vector multiplications, and similarly for the computation of $\widehat{\theta}$ and $\widehat{\eta}$.

It is crucial for the performance of this outer-inner Krylov procedure that a good enough approximation of the solution of (30) is obtained for small t, e.g., $t = 2$ or $t = 3$. We will see in our numerical examples that it gives almost as good results as the implementation of the full optimization procedure.

4.4. Mode with small dimension

In information science applications it often happens that one of the tensor modes has much smaller dimension than the others. For concreteness, assume that the first mode is small, i.e., $l \ll \min(m,n)$. Then in the Krylov variants described so far, after l steps the algorithm has produced a full basis in that mode, and no more need to be generated. Then the question arises which u-vector to choose, when new basis vectors are generated in the other two modes. One alternative it to use the vectors $u_1,\ldots,u_l$ in a cyclical way, another is to take a random linear combination. One may also apply the optimization idea in that mode, i.e., in the computation of w_i perform the maximization

$$\max_{\theta} \|\widehat{w}\|, \quad \text{where } \widehat{w} = \mathcal{A}\cdot(U_l\theta, v_i)_{1,2}, \quad \widehat{w} \perp W_{i-1}, \quad \|\theta\| = 1, \quad \theta \in \mathbb{R}^l.$$

The problem can be solved by computing a best rank-1 approximation of

$$C_w = \mathcal{A}\cdot\big(U_i, v_i, I - W_{i-1}W_{i-1}^{\mathsf{T}}\big) \in \mathbb{R}^{i\times 1\times n},$$

which is an $i \times n$ matrix[10] after suppressing the singleton dimension. As before, this is generally a dense matrix with one large mode. A rank one approximation can again be computed, without forming the dense matrix explicitly, using a Krylov method (here the Arnoldi method).

[10]Note that the tensor A is multiplied by a vector in the second mode, which results in a singleton dimension in that mode.

4.5. Krylov Subspaces for Contracted Tensor Products

Recall from Section 3.2 that the Golub–Kahan bidiagonalization procedure generated matrices U_k and V_k, which are orthonormal basis vectors for the Krylov subspaces of AA^T and $A^\mathsf{T}A$, respectively. In tensor notation those products may be written as

$$\langle A, A\rangle_{-1} = AA^\mathsf{T}, \qquad \langle A, A\rangle_{-2} = A^\mathsf{T}A.$$

For a third order tensor $\mathcal{A} \in \mathbb{R}^{l\times m\times n}$, and starting vectors $u \in \mathbb{R}^l$, $v \in \mathbb{R}^m$, and $w \in \mathbb{R}^n$, we may consider the matrix Krylov subspaces

$$\begin{aligned}
&\mathcal{K}_p(\langle \mathcal{A}, \mathcal{A}\rangle_{-1}, u), && [\langle \mathcal{A}, \mathcal{A}\rangle_{-1}]_{ij} = \sum_{\alpha,\beta} a_{i\alpha\beta}a_{j\alpha\beta},\\
&\mathcal{K}_q(\langle \mathcal{A}, \mathcal{A}\rangle_{-2}, v), && [\langle \mathcal{A}, \mathcal{A}\rangle_{-2}]_{ij} = \sum_{\alpha,\beta} a_{\alpha i\beta}a_{\alpha j\beta},\\
&\mathcal{K}_r(\langle \mathcal{A}, \mathcal{A}\rangle_{-3}, w), && [\langle \mathcal{A}, \mathcal{A}\rangle_{-3}]_{ij} = \sum_{\alpha,\beta} a_{\alpha\beta i}a_{\alpha\beta j}.
\end{aligned}$$

It is easy to verify that $\langle \mathcal{A}, \mathcal{A}\rangle_{-1} \in \mathbb{R}^{l\times l}$, $\langle \mathcal{A}, \mathcal{A}\rangle_{-2} \in \mathbb{R}^{m\times m}$, and $\langle \mathcal{A}, \mathcal{A}\rangle_{-3} \in \mathbb{R}^{n\times n}$. In this case we reduce a third order tensor to three different (symmetric) matrices, for which we compute the usual matrix subspaces through the Lanczos recurrence. This can be done without explicitly computing the products $\langle \mathcal{A}, \mathcal{A}\rangle_{-i}$, thus taking advantage of sparsity. To illustrate this consider the matrix times vector operation $(\langle \mathcal{A}, \mathcal{A}\rangle_{-1})u$, which can be written

$$[A_1\ \ldots\ A_n][A_1\ \ldots\ A_n]^\mathsf{T}u = \sum_{i=1}^{n} A_iA_i^\mathsf{T}u, \tag{33}$$

where $A_i = \mathcal{A}(:,:,i)$ is the i'th frontal slice of $\mathcal{A}$.

Observe that the approach in this section is not directly related to the methods described in previous sections. Here we compute Krylov subspaces of symmetric matrices. The tensor relation comes from the fact that the three symmetric matrices are obtained through contracted tensor products. The result of the Lanczos process separately on these three contracted tensor products is three sets of orthonormal basis vectors for each of the modes of the tensor, collected in U_p, V_q, and W_r, say. A low-rank approximation of the tensor can then be obtained using Lemma 1.

It is straightforward to show that if $\mathcal{A} = (X, Y, Z)\cdot\mathcal{C}$ with $\operatorname{rank}(\mathcal{A}) = (p, q, r)$, then the contracted tensor products

$$\langle \mathcal{A}, \mathcal{A}\rangle_{-1} = A^{(1)}(A^{(1)})^\mathsf{T} = XC^{(1)}(Y\otimes Z)^\mathsf{T}(Y\otimes Z)(C^{(1)})^\mathsf{T}X^\mathsf{T}, \tag{34}$$

$$\langle \mathcal{A}, \mathcal{A}\rangle_{-2} = A^{(2)}(A^{(2)})^\mathsf{T} = YC^{(2)}(X\otimes Z)^\mathsf{T}(X\otimes Z)(C^{(2)})^\mathsf{T}Y^\mathsf{T}, \tag{35}$$

$$\langle \mathcal{A}, \mathcal{A}\rangle_{-3} = A^{(3)}(A^{(3)})^\mathsf{T} = ZC^{(3)}(X\otimes Y)^\mathsf{T}(X\otimes Y)(C^{(3)})^\mathsf{T}Z^\mathsf{T}, \tag{36}$$

are matrices with ranks p, q, and r, respectively. Recall that $A^{(i)}$ denotes the matricization of $\mathcal{A}$ along the i'th mode. Then it is clear that the separate Lanczos recurrences will generate matrices U, V, and W, that span the same subspaces as X, Y, and Z in p, q, and r iterations, respectively.

Remark. Computing p (or q or r) dominant eigenvectors of the symmetric positive semidefinite matrices $\langle \mathcal{A}, \mathcal{A} \rangle_{-1}, \langle \mathcal{A}, \mathcal{A} \rangle_{-2}$, $\langle \mathcal{A}, \mathcal{A} \rangle_{-3}$, respectively, is equivalent to computing the truncated HOSVD of $\mathcal{A}$. We will show the calculations for the first mode. Using the HOSVD $\mathcal{A} = (U, V, W) \cdot \mathcal{S}$, where now U, V, and W are orthogonal matrices and the core $\mathcal{S}$ is all-orthogonal [6], we have

$$\langle \mathcal{A}, \mathcal{A} \rangle_{-1} = U S^{(1)} (V \otimes W)^{\mathsf{T}} (V \otimes W) (S^{(1)})^{\mathsf{T}} U^{\mathsf{T}} = U \bar{S} U^{\mathsf{T}},$$

where $\bar{S} = S^{(1)}(S^{(1)})^{\mathsf{T}} = \operatorname{diag}(\sigma_1^2, \sigma_2^2, \ldots, \sigma_l^2)$ with $\sigma_i^2 \geq \sigma_{i+1}^2$ and σ_i are first mode multilinear singular values of $\mathcal{A}$.

4.6. Complexity

In this subsection we will discuss the amount of computations associated to the different methods. For simplicity we let that $p = q = r = k$. Assuming that the tensor is large and sparse, or otherwise structured, it is likely that, for small values of k (compared to l, m, and n), the dominating work in computing a rank-(k, k, k) approximation is due to tensor-vector-vector multiplications.

Minimal Krylov recursion. Considering Equation (19), it is clear that computing the $k \times k \times k$ core tensor $\mathcal{H}$ is necessary to have a low rank approximation of $\mathcal{A}$. From the proof of Proposition 4 we see that $\mathcal{H}$ has a certain Hessenberg structure along and close to its "two-mode diagonals". However, away from the "diagonals" there will be no systematic structure. We can estimate that the total number of tensor-vector-vector multiplications for computing the $k \times k \times k$ tensor $\mathcal{H}$ is k^2. The computation of $\mathcal{H}$ can be split as

$$\mathcal{H} = \mathcal{A} \cdot (U_k, V_k, W_k) = \mathcal{A}_{uv} \cdot (W_k)_3, \quad \text{where} \quad \mathcal{A}_{uv} = \mathcal{A} \cdot (U_k, V_k)_{1,2}.$$

There are k^2 tensor-vector-vector multiplications for computing the $k \times k \times n$ tensor $\mathcal{A}_{uv}$. The complexity of the following computation $\mathcal{A}_{uv} \cdot (W_k)_3$ is $O(k^3 n)$, i.e. about k^3 vector-vector inner products.

Several of the elements of the core tensor are available from the generation of the Krylov vectors. Naturally they should be saved to avoid unnecessary work. Therefore we need not include the $3k$ tensor-vector-vector multiplications from the recursion in the complexity.

Maximal Krylov Recursion. There are several different possibilities[11] to use the maximal Krylov recursion in order to compute a rank-(k, k, k) approximation of a given tensor. For example, we could apply the method without any modifications until all subspaces have dimensions larger than k. In a subsequent step the subspaces would need to be reduced to the desired sizes. In view of the combinatorial complexity of the method the number of tensor-vector-vector multiplications can be much larger than in the minimal Krylov recursion. Alternatively, we could modify the recursion so that we do not generate more than k vectors in any mode. The latter variant has about the same complexity as the minimal Krylov recursion.

[11] We will only give a vague description of two principally different approaches as complete presentations could be quite detailed.

Optimized Krylov Recursion. The optimized recursion can be implemented in different ways. In Section 4.3 we described a variant based on "inner Krylov steps". Assuming that we perform t inner Krylov steps, finding the (almost) optimal $\widehat{w}$ in equation (32) requires $3t$ tensor-vector-vector multiplications. Since the optimization is done in k outer Krylov steps in three modes we perform $9kt$ such multiplications. The total complexity becomes $k^2 + 9kt$. In [12] another variant is described where the optimization is done on the core tensor.

Mode with small dimension. Assume that the tensor has one small mode and that a random or fixed combination of vectors is chosen in this mode when new vectors are generated in the other modes. Then the complexity becomes $k^2 + 2k$.

Krylov Subspaces for Contracted Tensor Products. In each step of the Krylov method a vector is multiplied by a contracted tensor product. This can be implemented using (33). If we assume that each such operation has the same complexity as two tensor-vector-vector multiplications, then the complexity becomes k^2+6k, where the second order term is for computing the core tensor. Our assumption above on two tensor-vector-vector multiplication actually overestimates the exact computation since computing $[A_1 \cdots A_n]^\mathsf{T} u$ is equivalent to a tensor-vector multiplication yielding a matrix. In tensor-vector-vector multiplication there is an additional product which is formally between a matrix and the second vector.

The complexities for four of the methods are summarized in Table 1.

Method	Complexity
Minimal Krylov	k^2
Optimized minimal Krylov	$k^2 + 9kt$
Mode with small dimension	$k^2 + 2k$
Contracted tensor products	$k^2 + 6k$

Table 1: Computational complexity in terms of tensor-vector-vector multiplications for the computation of a rank-(k, k, k) approximation with four different tensor Krylov methods. In the optimized Krylov recursion t inner Krylov steps are made.

5. Numerical Examples

The purpose of the examples in this section is to make a preliminary investigation of the usefulness of the concepts proposed. We will generate matrices U, V,and W using the various Krylov procedures. In some examples we will compute the truncated HOSVD for comparison. Given a tensor $\mathcal{A}$ and matrices U, V, and W the approximating tensor $\tilde{\mathcal{A}}$ has the form

$$\tilde{\mathcal{A}} = (U, V, W) \cdot \mathcal{C}, \quad \text{where} \quad \mathcal{C} = \mathcal{A} \cdot (U, V, W)\,. \tag{37}$$

Of course, for large problems computing $\tilde{\mathcal{A}}$ explicitly (by multiplying together the matrices and the core) will not be feasible, since that tensor will be dense. However, it is

$\dim(\mathcal{A})\backslash\text{rank}(\mathcal{A})$	$(10,10,10)$		$(10,15,20)$		$(20,30,40)$	
Method	(1)	(2)	(1)	(2)	(1)	(2)
$50 \times 70 \times 60$	0.087	0.165	0.113	0.162	0.226	0.169
$100 \times 100 \times 100$	0.38	2.70	0.44	2.72	0.91	2.71
$150 \times 180 \times 130$	1.32	11.27	1.44	11.07	3.01	11.01

Table 2: Timing in seconds for computing the HOSVD of low rank tensors using: (1) the modified minimal Krylov method and HOSVD of the smaller core $\mathcal{H}$; and (2) truncated HOSVD approximation of $\mathcal{A}$.

easy to show that the approximation error is

$$\|\mathcal{A} - \tilde{\mathcal{A}}\| = \left(\|\mathcal{A}\|^2 - \|\mathcal{C}\|^2\right)^{1/2}.$$

For many applications a low rank approximation is only an intermediate or auxiliary result, see e.g., [27]. It sometimes holds that the better the approximation (in norm), the better it will perform in the particular application. But quite often, especially in information science applications, good performance is obtained using an approximation with quite high error, see e.g. [16]. Our experiments will focus on how good approximations are obtained by the proposed methods. How these low rank approximations are used will depend on the application as well as on the particular data set.

For the timing experiments in Sections 5.1 and 5.2 we used a MacBook laptop with 2.4GHz processor and 4GB of main memory. For the experiments on the Netflix data in Section 5.3 we used a 64 bit Linux machine with 2.2GHz processor and 32GB of main memory running Ubuntu. The calculations were performed using Matlab and the TensorToolbox, which supports computation with sparse tensors [2, 3].

5.1. Minimal Krylov Procedures

We first made a set of experiments to confirm that the result in Theorem 3 holds for a numerical implementation, using synthetic data generated with a specified low rank.

Computing the HOSVD of a tensor $\mathcal{A}$ with "exact" low multilinear rank can be done by direct application of the SVD on the different matricizations $A^{(i)}$ for $i = 1, 2, 3$. An alternative is to first compute U_p, V_q, and W_r using the modified minimal Krylov procedure. Then we have the decomposition $\mathcal{A} = (U_p, V_q, W_r) \cdot \mathcal{H}$. To obtain the HOSVD of $\mathcal{A}$ we compute the HOSVD of the much smaller[12] tensor $\mathcal{H} = \left(\bar{U}, \bar{V}, \bar{W}\right) \cdot \mathcal{C}$. It follows that the matrices containing the multilinear singular vectors for $\mathcal{A}$ are given by $U_p\bar{U}$, $V_q\bar{V}$, and $W_r\bar{W}$. We conducted a few experiments to compare timings for the two approaches. Tensors with three different dimensions were generated and for each case we used three different low ranks. The tensor dimensions, their ranks and the computational times for the respective cases are presented in Table 2. We see that for the larger problems the computational time for the HOSVD is 2–8 times longer than for the modified minimal Krylov procedure with HOSVD on the core tensor. Of course, timings of MATLAB codes

[12] $\mathcal{A}$ is a $l \times m \times n$ tensor and $\mathcal{H}$ is a $p \times q \times r$, and usually the multilinear ranks p, q, and r are much smaller than the dimensions l, m, and n, respectively.

are unreliable in general, since the efficiency of execution depends on how much of the algorithm is user-coded and how much is implemented in MATLAB low-level functions (e.g., LAPACK-based). It should be noted that the tensors in this experiment are dense, and much of the HOSVD computations are done in low-level functions. Therefore, we believe that the timings are rather realistic.

It is not uncommon that tensors originating from signal processing applications have approximate low multilinear ranks, e.g., we may have $\mathcal{A} = \mathcal{A}_{\text{signal}} + \mathcal{A}_{\text{noise}}$, where $\mathcal{A}_{\text{signal}}$ is a signal tensor with exact low multilinear rank and $\mathcal{A}_{\text{noise}}$ is a noise tensor. We conjecture that the situation will be similar when the tensor has approximate low multilinear rank, i.e., that the minimal Krylov recursion will extract the signal part of the tensor in the same number of iterations as in the noiseless situation. Then a HOSVD on a small tensor, with a following change of basis, will give an approximation of the HOSVD of $\mathcal{A}_{\text{signal}}$ that is accurate to the level of noise. Indeed, in a numerical example, we created a $50 \times 50 \times 50$ signal tensor $\mathcal{A}_{\text{signal}}$ with rank-$(5, 5, 5)$ and noise tensors $\mathcal{A}_{\text{noise},\rho}$ with different levels of noise ρ. Then we applied 5 iterations (since the rank of $\mathcal{A}_{\text{signal}}$ in all modes is 5) of the minimal Krylov recursion on $\mathcal{A}_\rho = \mathcal{A}_{\text{signal}} + \mathcal{A}_{\text{noise},\rho}$. Denoting the approximation by $\tilde{\mathcal{A}}_\rho$ obtained by the algorithm, we computed $q_\rho = \|\mathcal{A}_\rho - \tilde{\mathcal{A}}_\rho\|/\|\mathcal{A}_{\text{noise},\rho}\|$, i.e., the ratio of the error in the approximation to the noise. In a series of 30 experimental runs we varied the noise to signal ratio $\|\mathcal{A}_{\text{noise},\rho}\|/\|\mathcal{A}_{\text{signal}}\|$ in the range $(10^{-3}, 10^{-10})$. The mean, variance, and worst case of the ratios q_ρ was 1.27, 0.002, and 1.40, respectively.

Tensors (and matrices) from information science applications seldom have a low rank structure (i.e., they are not of the type signal+noise mentioned above). Still the use of low rank approximations often enhances the performance of algorithms in data mining and pattern recognition, see e.g. [10]. Synthetic examples of this type can be constructed using random data. Here we let $\mathcal{A} \in \mathbb{R}^{50\times 60\times 40}$ be a random tensor, and computed a rank-$(10, 10, 10)$ approximation using the minimal Krylov recursion and a different approximation using the truncated HOSVD. Let the optimal cores computed using Lemma 1 be denoted $\mathcal{H}_{\text{min}}$ and $\mathcal{H}_{\text{hosvd}}$, respectively. We made this calculation for 100 different random tensors and report $(\|\mathcal{H}_{\text{min}}\| - \|\mathcal{H}_{\text{hosvd}}\|)/\|\mathcal{H}_{\text{hosvd}}\|$ for each case. Figure 1 illustrates the outcome. Clearly, if the relative difference is larger than 0, then the Krylov method gives a better approximation. In about 80% of the runs the minimal Krylov method generated better approximations than the truncated HOSVD, but the difference was quite small.

In the following experiment we compared the performance of different variants of the optimized minimal Krylov recursion applied to sparse tensors. We generated tensors based on Facebook graphs for different US universities [31]. The Caltech graph is represented by a 597×597 sparse matrix. For each individual there is housing information. Using this we generated a tensor of dimension $597 \times 597 \times 64$, with 25,646 nonzeros. The purpose was to see how good approximations the different methods gave as a function of the subspace dimension. We compared the minimal Krylov recursion to the following optimized variants:

Opt-HOSVD. The minimal Krylov recursion with optimization based on HOSVD of the core tensor (31). This variant is very costly and is included only as a benchmark.

Opt-Krylov. The minimal Krylov recursion that utilized three inner Krylov steps to obtain approximations to the optimized linear combinations. This is an implemen-

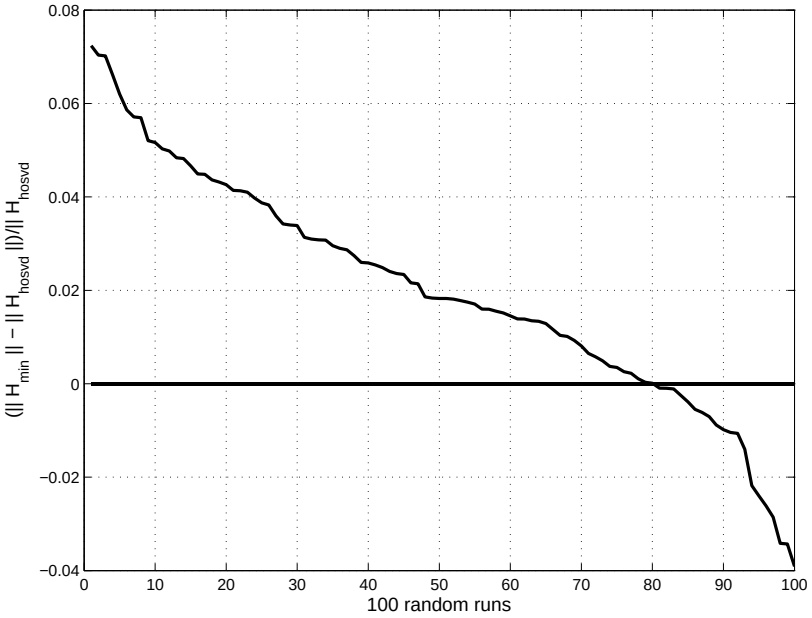

Figure 1: Difference between $\|\mathcal{A}_{\min}\|$, approximation obtained with the minimal Krylov method, and $\|\mathcal{A}_{\text{hosvd}}\|$, approximation obtained by the truncated HOSVD of a $50 \times 60 \times 40$ tensor $\mathcal{A}$. The rank of the approximations were $(10, 10, 10)$.

tation of the discussion from the second part of Section 4.3.

Opt-Alg8. Algorithm 8 in [12][13].

Truncated HOSVD. This was included as a benchmark comparison.

minK-back. In this method we used the minimal Krylov method but performed 10 extra steps. Then we formed the core $\mathcal{H}$ and computed a truncated HOSVD approximation of $\mathcal{H}$. As a last step we truncated the Krylov subspaces accordingly.

In all Krylov-based methods we used four initial minimal Krylov steps before we started using the optimizations.

Another sparse tensor was created using the Facebook data from Princeton. Here the tensor was constructed using a student/faculty flag as third mode, giving a 6,593 $\times$ 6,593 $\times$ 29 tensor with 585,754 non-zeros.

The results are illustrated in Figure 2. We see that for the Caltech tensor the "backward-looking" variant (minK-back) gives good approximations for small dimensions as long as there is a significant improvement in each step of the minimal Krylov recursion. After some ten steps all optimized variants give approximations that are rather close to that of the HOSVD.

[13]The algorithm involves the approximation of the dominant singular vectors of a matrix computed from the core tensor. In [12] the power method was used for this computation. We used a linear combination of the first three singular vectors of the matrix, weighted by the singular values.

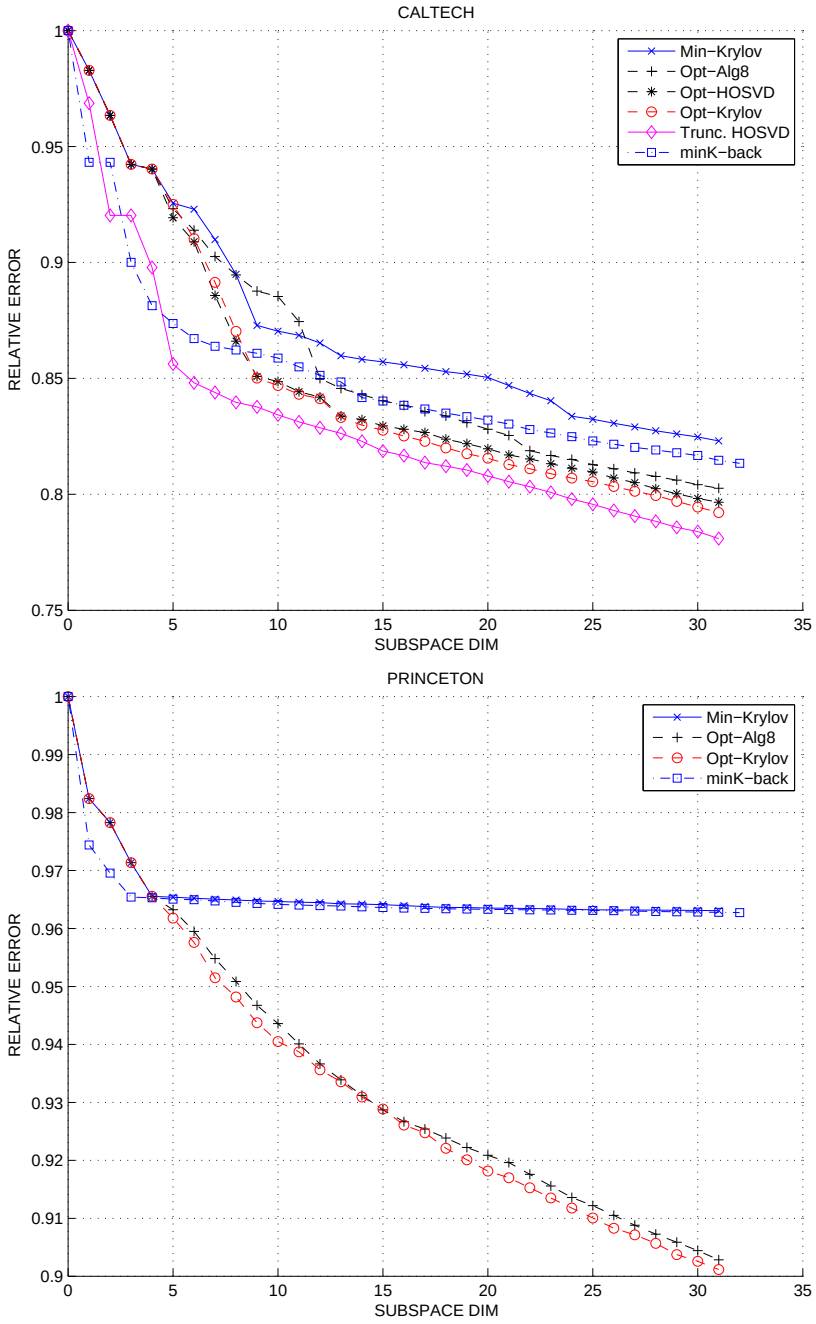

Figure 2: Errors in the low rank approximations of the sparse Caltech (top) and Princeton (bottom) tensors.

For the Princeton tensor we only ran the minimal Krylov recursion and two of the optimizations. Here the optimized versions continued to give significant improvements as the dimension is increased, in spite of the poor performance of the minimal Krylov procedure itself.

5.2. Test on Handwritten Digits

Tensor methods for the classification of handwritten digits are described in [26, 27]. We have performed tests using the handwritten digits from the US postal service database. Digits from the database were formed into a tensor $\mathcal{D}$ of dimensions $400 \times 1{,}194 \times 10$. The first mode of the tensor represents pixels[14], the second mode represents the variation within the different classes and the third mode represents the different classes. Our goal was to find low dimensional subspaces U_p and V_q in the first and second mode, respectively. The approximation of the original tensor can be written as

$$\mathbb{R}^{400\times 1,194\times 10} \ni \mathcal{D} \approx (U_p, V_q)_{1,2} \cdot \mathcal{F} \equiv \tilde{\mathcal{D}}. \tag{38}$$

An important difference compared to the previous sections is that here we wanted to find only two of three matrices. The class (third) mode of the tensor was not reduced to lower rank, i.e., we were computing a rank-$(p, q, 10)$ approximation of $\mathcal{D}$. We computed low rank approximations for this tensor using five different methods: (1) truncated HOSVD; (2) modified minimal Krylov recursion; (3) contracted tensor product Krylov recursion; (4) maximal Krylov recursion; and (5) optimized minimal Krylov recursion. Figure 3 shows the obtained results for low rank approximations with $(p, q) = \{(5, 10),\ (10, 20),\ (15, 30),\ \cdots,\ (50, 100)\}$. The reduction of dimensionality in two modes required special treatment for several of the methods. We will describe each case separately. In each case the rank-$(p, q, 10)$ approximation $\tilde{\mathcal{D}}$ is given by

$$\tilde{\mathcal{D}} = \left(U_p U_p^{\mathsf{T}}, V_q V_q^{\mathsf{T}}\right)_{1,2} \cdot \mathcal{D}, \tag{39}$$

where the matrices U_p and V_q were obtained using different methods.

Truncated HOSVD. We computed the HOSVD $\mathcal{D} = (U, V, W) \cdot \mathcal{C}$ and truncated the matrices $U_p = U(:, 1{:}p)$ and $V_q = V(:, 1{:}q)$.

Modified minimal Krylov recursion. The minimal Krylov recursion was modified in several respects. We ran Algorithm 4 for 10 iterations and obtained U_{10}, V_{10}, and W_{10}. Next we ran $p - 10$ iterations and generated only u- and v-vectors. For every new u_{k+1} we used $\bar{v}$ and $\bar{w}$ as random liner combination of vectors in V_k and W_{10}, respectively. In the last $q - p$ iterations we only generated new v-vectors using again random linear combinations of vectors in U_p and W_{10}.

Contracted tensor product Krylov recursion. For this method we applied p and q steps of the Lanczos method with random starting vectors on the matrices $\langle \mathcal{A}, \mathcal{A} \rangle_{-1}$ and $\langle \mathcal{A}, \mathcal{A} \rangle_{-2}$, respectively.

[14] Each digit is smoothed and reshaped to a vector. The smoothing process removes sharp edges from the images and enhances the performance of the classifier, see [26] for details.

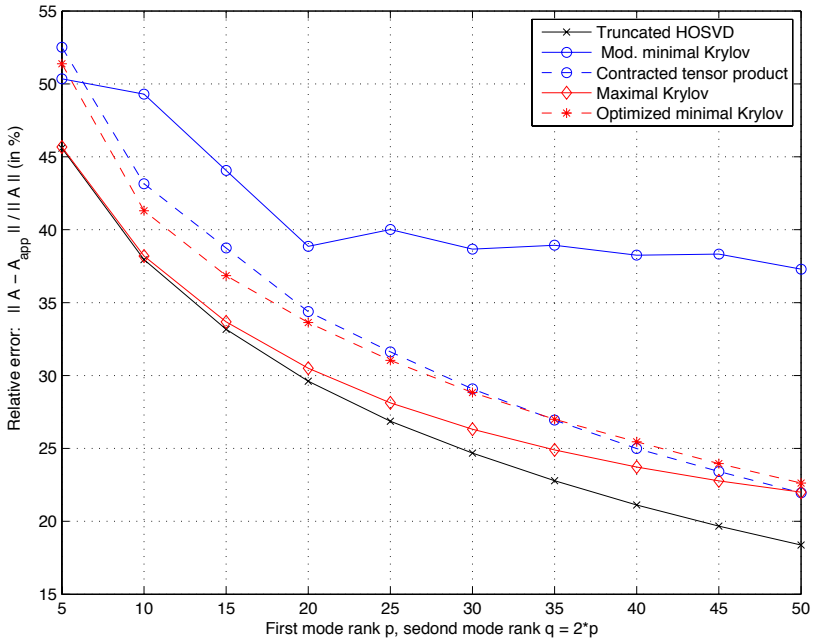

Figure 3: The relative error of the low rank approximations obtained using five different methods. The x-axis indicates the ranks $(p, q, 10) = (p, 2p, 10)$ in the approximation.

Maximal Krylov recursion. We used the maximal Krylov recursion with starting vectors u_1 and v_1 to generate $W_1 \to U_2 \to V_3 \to W_6 \to U_{19} \to V_{115}$. Clearly, we now need to make modification to the general procedure. W_{10} was obtained by truncating the third mode orthogonal matrix in the HOSVD of the product $\mathcal{D} \cdot (U_{19}, V_{115})_{1,2}$. As a last step, we computed U_{100} as the 100-dimensional dominant subspace obtained from the first mode orthogonal matrix in the HOSVD of $\mathcal{D} \cdot (V_{114}, W_{10})_{2,3}$. The U_p and V_q matrices, used for the approximation in (39), were constructed by forming the product $\bar{\mathcal{C}} = \mathcal{D} \cdot (U_{100}, V_{115})_{1,2}$, computing the HOSVD of $\bar{\mathcal{C}} = (\tilde{U}, \tilde{V}, \tilde{W}) \cdot \tilde{\mathcal{C}}$, and finally by computing $U_p = U_{100}\tilde{U}(:, 1{:}p)$ and $V_q = U_{115}\tilde{V}(:, 1{:}q)$.

Optimized minimal Krylov recursion. For this minimal Krylov recursion we used the optimization approach, for every new vector, from the first part of Section 4.3. The coefficients for the linear combinations, e.g., θ and η in (30), were obtained by best rank-$(1, 1, 1)$ approximation of factors as $\mathcal{C}_w$ in equation (31).

The experiments with the handwritten digits are illustrated in Figure 3. We made the following observations: (1) The truncated HOSVD give the best approximation for all cases; (2) The minimal Krylov approach did not perform well. We observed several breakdowns for this methods as the ranks in the approximation increased. Every time the process broke down we used a randomly generated vector that was orthonormalized against all previously generated vectors; (3) The Lanczos method on the contracted tensor products performed very similar as the optimized minimal Krylov method; (4) The performance of the maximal Krylov method was initially as good as the truncated

HOSVD but its performance degraded eventually.

Classifications results using these subspaces in the algorithmic setting presented in [27] are similar for all methods, indicating that all of the methods capture subspaces that can used for classification purposes.

5.3. Tests on the Netflix Data

A few years ago, the Netflix company announced a competition[15] to improve their algorithm for movie recommendations. Netflix made available movie ratings from 480,189 users/costumers on 17,770 movies during a time period of 2,243 days. In total there were over 100 million ratings. We will not address the Netflix problem, but we will use the data to test some of the Krylov methods we are proposing. For our experiments we formed the tensor $\mathcal{A}$ that is $480{,}189 \times 17{,}770 \times 2{,}243$ and contains all the movie ratings. Entries in the tensor for which we do not have any rating were considered as zeros. We used the minimal Krylov recursion and the Lanczos process on the products $\langle \mathcal{A}, \mathcal{A} \rangle_{-1}$, $\langle \mathcal{A}, \mathcal{A} \rangle_{-2}$, and $\langle \mathcal{A}, \mathcal{A} \rangle_{-3}$ to obtain low rank approximations of $\mathcal{A}$.

In Figure 4 (left plot) we present the norm of the approximation, i.e., $\| (U_k, V_k, W_k) \cdot \mathcal{C}_{\min} \| = \| \mathcal{C}_{\min} \|$, where $\mathcal{C}_{\min} = \mathcal{A} \cdot (U_k, V_k, W_k)$. We have the same rank in the approximation in each mode, i.e., $p = q = r = k$, and the ranks range from $k = 5, 10, 15, \ldots, 100$. The plot contains four curves. Three runs with different random initial vectors in all three modes and a fourth run that is initialized with the column means of $A^{(1)}$, $A^{(2)}$, and $A^{(3)}$, respectively. Observe that for this size of tensors it is practically impossible to form the approximation $\mathcal{A}_{\min} = \left(U_k U_k^{\mathsf{T}}, V_k V_k^{\mathsf{T}}, W_k W_k^{\mathsf{T}} \right) \cdot \mathcal{A}$ since the approximation will be dense. But the quantity $\| \mathcal{C}_{\min} \|$ is computable and indicates the quality of the approximation. Larger $\| \mathcal{C}_{\min} \|$ means better approximation. In fact, for orthonormal matrices U_k, V_k, and W_k, it holds that $\| \mathcal{A}_{\min} \| = \| \mathcal{C}_{\min} \| \leq \| \mathcal{A} \|$.

Figure 4 (right plot) contains similar plots, but now the matrices U_k, V_k, and W_k are obtained using the Lanczos process on the symmetric matrices $\langle \mathcal{A}, \mathcal{A} \rangle_{-1}$, $\langle \mathcal{A}, \mathcal{A} \rangle_{-2}$, and $\langle \mathcal{A}, \mathcal{A} \rangle_{-3}$, respectively. We never formed the first two products, but use the computational formula from equation (33) for obtaining first mode vectors and a similar one for obtaining second mode vectors. We did form $\langle \mathcal{A}, \mathcal{A} \rangle_{-3}$ explicitly since it is a relatively small ($2{,}243 \times 2{,}243$) matrix. We ran the Lanczos process with ranks $k = 5, 10, 15, \ldots, 100$ using random starting vectors in all three modes. Three tests were made and we used the Lanczos vectors in U_k, V_k, and W_k. In addition we computed the top 100 eigenvectors for each one of the contracted products. Observe that the scale for the y-axis in the two plots are different and that the minimal Krylov method (left plot) considerably outperforms the contracted tensor product method (right plot). This difference may partially be explained by the fact that the contracted tensor product method has the theoretical optimal solution given by the truncated HOSVD. Thus the Lanczos method produces iterates that converge to a suboptimal solution. For the minimal Krylov method we already have seen in section 5.1 that it often performs better than the truncated HOSVD.

We remark that this Netflix tensor is special in the sense that every third mode fibre, i.e., $\mathcal{A}(i, j, :)$, contains only one nonzero entry. It follows that the product $\langle \mathcal{A}, \mathcal{A} \rangle_{-3}$ is

[15] The competition has obtained huge attention from many researcher and non-researchers. The improvement of 10 % that was necessary to claim the prize for the contest was achieved by join efforts of a few of the top teams [24].

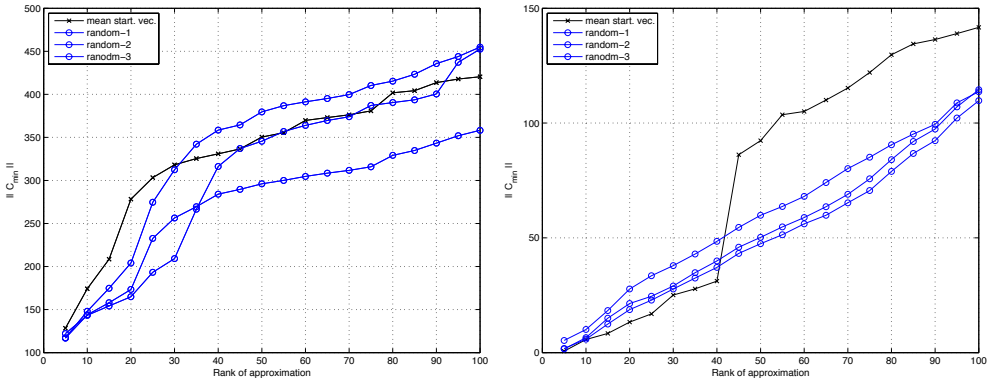

Figure 4: We plot $\|\mathcal{C}_{\min}\|$ as a function of the rank $(p,q,r) = (p,p,p)$ in the approximation with $p = 5, 10, 15, \ldots, 100$. **Left plot:** U_p, V_p, and W_p are obtained using the minimal Krylov recursion. Four runs are presented: one using starting vectors u_1, v_1, and w_1 as the means of the first, second and third mode fibers of $\mathcal{A}$, respectively, and three runs with different set of random initial vectors. **Right plot:** The subspaces for this case were obtained from separate Lanczos recurrences of contracted tensor products. The starting vectors were chosen as in the left plot. Note the difference in the y-axis, showing that the minimal Krylov recursion considerably outperforms the method using contracted tensor products.

a diagonal matrix. Our emphasis for these tests was to show that the proposed tensor Krylov methods can be employed on very large and sparse tensors.

In this experiment it turned out to be more efficient to store the sparse Netflix tensor "slice-wise", where each slice was stored separately as a sparse matrix, than using the sparse tensor format from the TensorToolbox.

6. Conclusions and Future Work

In this paper we propose several ways to generalize matrix Krylov methods to tensors, having applications with sparse tensor in mind. In particular we introduce three different methods for tensor approximations. These are the *minimal Krylov recursion*, the *maximal Krylov recursion*, and the *contracted tensor product recursion*. We prove that, given a tensor of the form $\mathcal{A} = (X, Y, Z) \cdot \mathcal{C}$ with $\operatorname{rank}(\mathcal{A}) = \operatorname{rank}(\mathcal{C}) = (p,q,r)$, a modified version of the minimal Krylov recursion extracts the associated subspaces of $\mathcal{A}$ in $\max(p,q,r)$ iterations, or equivalently in $p+q+r$ tensor-vector-vector multiplications. We also investigate a variant of the optimized minimal Krylov recursion [12], which gives better approximation than the minimal recursion, and which can be implemented using only sparse tensor-vector-vector operations. We also show that the maximal Krylov approach generalizes the matrix Krylov decomposition to a corresponding tensor Krylov decomposition.

The experiments clearly indicate that the Krylov methods are useful for multilinear rank approximations of large and sparse tensors. In [12] it is also shown that they are efficient for further compression of dense tensors, that are given in canonical format. The tensor Krylov methods can also be used to speed up HOSVD computations.

As the research on tensor Krylov methods is still in a very early stage, there are numerous questions that need to be answered, and which will be the subject of our

continued research. We have hinted to some in the text; here we list a few others.

1. Details with respect to detecting true break down, in floating point arithmetic, and distinguishing those from the case when a complete subspaces is obtained need to be worked out.

2. A difficulty with Krylov methods for very large problems is that the basis vectors generated are in most cases dense. Also, when the required subspace dimension is comparatively large, the cost for (re)orthogonalization will be high. For matrices the subspaces can be improved using the implicitly restarted Arnoldi (Krylov–Schur) approach [23, 30]. Preliminary tests indicate that similar procedures for tensors may be efficient. The properties of such methods and their implementation will be studied.

3. The efficiency of the different variants of Krylov methods in terms of the number of tensor-vector-vector operations, and taking into account the convergence rate will be investigated.

[1] Andersson, C., Bro, R., 1998. Improving the speed of multi-way algorithms: - part i. tucker3. Chemometrics and Intelligent Laboratory Systems 42, 93–103.

[2] Bader, B. W., Kolda, T. G., 2006. Algorithm 862: MATLAB tensor classes for fast algorithm prototyping. ACM Trans. Math. Softw. 32 (4), 635–653.

[3] Bader, B. W., Kolda, T. G., 2007. Efficient MATLAB computations with sparse and factored tensors. SIAM Journal on Scientific Computing 30 (1), 205–231.
URL `http://link.aip.org/link/?SCE/30/205/1`

[4] Carroll, J. D., Chang, J. J., 1970. Analysis of individual differences in multidimensional scaling via an n-way generalization of Eckart-Young decomposition. Psychometrika 35, Psychometrika.

[5] Comon, P., 2002. Tensor decompositions. In: McWhirter, J. G., Proudler, I. K. (Eds.), Mathematics in Signal Processing V. Clarendon Press, Oxford, UK, pp. 1–24.

[6] De Lathauwer, L., De Moor, B., Vandewalle, J., 2000. A multilinear singular value decomposition. SIAM Journal on Matrix Analysis and Applications 21 (4), 1253–1278.
URL `http://link.aip.org/link/?SML/21/1253/1`

[7] De Lathauwer, L., Moor, B. D., Vandewalle, J., 2000. On the best rank-1 and rank-$(R_1, R_2, \ldots, R_N)$ approximation of higher-order tensors. SIAM Journal on Matrix Analysis and Applications 21 (4), 1324–1342.
URL `http://link.aip.org/link/?SML/21/1324/1`

[8] de Silva, V., Lim, L.-H., 2008. Tensor rank and the ill-posedness of the best low-rank approximation problem. SIAM Journal on Matrix Analysis and Applications 30 (3), 1084–1127.
URL `http://link.aip.org/link/?SML/30/1084/1`

[9] Eldén, L., Savas, B., 2009. A Newton–Grassmann method for computing the best multilinear rank-(r_1, r_2, r_3) approximation of a tensor. SIAM Journal on Matrix Analysis and Applications 31, 248–271.

[10] Eldén, L., 2007. Matrix Methods in Data Mining and Pattern Recognition. SIAM.

[11] Golub, G. H., Kahan, W., 1965. Calculating the singular values and pseudo-inverse of a matrix. SIAM J. Numer. Anal. Ser. B 2, 205–224.

[12] Goreinov, S., Oseledets, I., Savostyanov, D., April 2010. Wedderburn rank reduction and Krylov subspace method for tensor approximation. Part 1: Tucker case. Tech. Rep. Preprint 2010-01, Inst. Numer. Math., Russian Academy of Sciences.

[13] Harshman, R. A., 1970. Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multi-modal factor analysis. UCLA Working Papers in Phonetics 16, 1–84.

[14] Hitchcock, F. L., 1927. Multiple invariants and generalized rank of a p-way matrix or tensor. J. Math. Phys. Camb. 7, 39–70.

[15] Ishteva, M., De Lathauwer, L., Absil, P.-A., Van Huffel, S., 2009. Best low multilinear rank approximation of higher-order tensors, based on the riemannian trust-region scheme. Tech. Rep. 09-142, ESAT-SISTA, K.U.Leuven (Leuven, Belgium).

[16] Jessup, E. R., Martin, J. H., 2001. Taking a new look at the latent semantic analysis approach to information retrieval. In: Berry, M. W. (Ed.), Computational Information Retrieval. SIAM, Philadelphia, PA, pp. 121–144.
[17] Khoromskij, B. N., Khoromskaia, V., 2009. Multigrid accelerated tensor approximation of function related multidimensional arrays. SIAM Journal on Scientific Computing 31 (4), 3002–3026.
URL http://link.aip.org/link/?SCE/31/3002/1
[18] Kobayashi, S., Nomizu, K., 1963. Foundations of Differential Geometry. Interscience Publisher.
[19] Kolda, T. G., Bader, . W., Kenny, J. P., November 2005. Higher-order web link analysis using multilinear algebra. In: ICDM 2005: Proceedings of the 5th IEEE International Conference on Data Mining. pp. 242–249.
[20] Kolda, T. G., Bader, B. W., 2009. Tensor decompositions and applications. SIAM Review 51 (3), 455–500.
URL http://link.aip.org/link/?SIR/51/455/1
[21] Kroonenbeg, P. M., 1980. Principal component analysis of three-mode data by means of alternating least squares algorithms. Psychometrika 45, 69–97.
[22] Kruskal, J. B., 1989. Rank, decomposition, and uniqueness for 3-way and n-way arrays. North-Holland Publishing Co., Amsterdam, The Netherlands, The Netherlands, pp. 7–18.
URL http://portal.acm.org/citation.cfm?id=120565.120567
[23] Lehoucq, R., Sorensen, D., Yang, C., 1998. Arpack Users' Guide: Solution of Large Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods. SIAM, Philadelphia.
[24] Lohr, S., 2009. A $1 million research bargain for netflix, and maybe a model for others. The New York TimesSeptember, 21.
URL http://www.nytimes.com/2009/09/22/technology/internet/22netflix.html?_r=1
[25] Oseledets, I. V., Savostianov, D. V., Tyrtyshnikov, E. E., 2008. Tucker dimensionality reduction of three-dimensional arrays in linear time. SIAM Journal on Matrix Analysis and Applications 30 (3), 939–956.
URL http://link.aip.org/link/?SML/30/939/1
[26] Savas, B., 2003. Analyses and tests of handwritten digit recognition algorithms. Master's thesis, Linköping University, http://www.mai.liu.se/~besav/.
[27] Savas, B., Eldén, L., 2007. Handwritten digit classification using higher order singular value decomposition. Pattern Recognition 40, 993–1003.
[28] Savas, B., Lim, L.-H., 2010. Quasi-Newton methods on Grassmannians and multilinear approximations of tensors. SIAM Journal on Scientific Computing 32 (6), 3352–3393.
URL http://link.aip.org/link/?SCE/32/3352/1
[29] Stewart, G. W., 2001. Matrix Algorithms II: Eigensystems. SIAM, Philadelphia.
[30] Stewart, G. W., 2002. A Krylov–Schur algorithm for large eigenproblems. SIAM Journal on Matrix Analysis and Applications 23 (3), 601–614.
URL http://link.aip.org/link/?SML/23/601/1
[31] Traud, A. L., Kelsic, E. D., Mucha, P. J., Porter, M. A., 2008. Community structure in online collegiate social networks. Tech. rep., arXiv:physics.soc-ph/0809.0690.
[32] Tucker, L. R., 1964. The extension of factor analysis to three-dimensional matrices. Contributions to Mathematical Psychology, 109–127.
[33] Zhang, T., Golub, G. H., 2001. Rank-one approximation to high order tensors. SIAM Journal on Matrix Analysis and Applications 23 (2), 534–550.
URL http://link.aip.org/link/?SML/23/534/1