Tensor Train Approximation of Multivariate Functions

1st Petr Tichavský
Inst. of Information Theory and Automation
Czech Academy of Sciences
Prague 8, Czech Republic
tichavsk@utia.cas.cz

2nd Ondřej Straka
Department of Cybernetics
University of West Bohemia
Pilsen, Czech Republic
straka30@kky.zcu.cz

***Abstract*—The tensor train is a popular model for approximating high-dimensional rectangular data structures that cannot fit in any computer memory due to their size. The tensor train can approximate complex functions with many variables in the continuous domain. The traditional method for obtaining the tensor train model is based on a skeleton decomposition, which is better known for matrices. The skeleton (cross) decomposition has the property that the tensor approximation is accurate on certain tensor fibers but may be poor on other fibers. In this paper, we propose a technique for fitting a tensor train to an arbitrary number of tensor fibers, allowing flexible modeling of multivariate functions that contain noise. Two examples are studied: a noisy Rosenbrock function and a noisy quadratic function, both of order 20.**

I. Introduction

Multidimensional data appear in many applications, e.g., chemistry, astronomy, psychology, and other fields. Such data is obtained when multivariate functions are sampled in a rectangular grid. The number of tensor elements is an exponential function of the number of tensor dimensions (tensor order). Some authors speak of a *curse of dimensionality*.

In practice, getting by with only a fraction of the tensor elements may be possible. The most common methods for handling multidimensional data are known as Parallel factor analysis (PARAFAC) [1], or Canonical decomposition (CANDECOMP) [2], or CP [3], [4]. Another popular method for coping with high-dimensional data is the tensor train (TT) [5], [6]. In quantum chemistry, this is known as the "matrix product state", and it is an outcome of the algorithm called "density matrix renormalization group" (DMRG) [14].

Both models, the canonical polyadic decomposition (CPD) and the tensor train (TT), have in common the fact that they are multilinear. When some model parameters are fixed, the tensor is linear with respect to the remaining parameters. When the fitting criterion is quadratic, the optimization is performed in closed form with respect to the linear part. The alternating least squares (ALS) method consists of a cyclic optimization with respect to a part of the parameters. The ALS method is the most classical algorithm for the CPD. It has also been applied to TT [17], [18]. In the latter paper, the algorithm is used for tensor completion (estimating missing tensor elements). Our paper solves a similar problem where the observed tensor elements are arranged in tensor fibers.

This work was supported by the Czech Science Foundation through the project No. 22-11101S.

In the case of TT decomposition, it is usually much more convenient to use singular value decomposition (SVD), provided that all tensor elements are available and their number is not too large. When the number of tensor elements is large, it has been suggested to use the skeleton decomposition instead, an extension of the skeleton matrix decomposition [16]. This algorithm, also known under the name TT-cross, exists in several variants [7], [9], [12], all of them use the *maximum volume algorithm* [15]. The method is useful when the data has no noise and obeys the low-rank model.

The novelty of this work is twofold. First, to the best of our knowledge, it is the first paper to apply the ALS algorithm to tensors that would never fit in any computer memory while using the observed tensor elements stored in bundles of fibers. Second, it is shown that some interesting or important multivariate functions can be represented as functional tensor trains with low bond dimensions.

The paper is organized as follows. In Section 2, we introduce the TT and TT-cross decompositions. In Section 3, we present an ALS technique for the TT decomposition that allows the model to fit many more fibers of the tensor than the TT-cross decomposition. A similar technique exists for CPD [19], [20]. Here, the TT-cross decomposition is used to initialize ALS. Section 4 presents a numerical experiment using the Rosenbrock function and a quadratic function, both with 20 variables. Section 5 concludes the paper.

II. TT decomposition

Assume that we wish to model a function $V(\boldsymbol{\xi})$ of N variables, $\boldsymbol{\xi} = (x_1, \ldots, x_N)$ sampled at the rectangular grid $[g_1^1, \ldots, g_{I_1}^1] \times \ldots \times [g_1^N, \ldots, g_{I_N}^N]$. The sampled function constitutes an order$-N$ tensor $\mathcal{T}$ of dimension $I_1 \times I_2 \times \ldots \times I_N$ with elements

$$T(i_1, \ldots, i_N) = V(g_{i_1}^1, \ldots, g_{i_N}^N)$$

for $i_n = 1, \ldots, I_n$, $n = 1, \ldots, N$.

A TT is a tensor model with a moderate number of parameters arranged in so-called wagons or carriages. The wagons are order-3 tensors except for the first and last wagons, which are order-2, i.e. matrices. The tensor train is given as

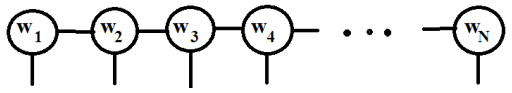

Fig. 1. Tensor train $\mathcal{T} = \{\{\mathbf{w}\}\}$ with wagons $\mathbf{w}_1, \ldots, \mathbf{w}_N$.

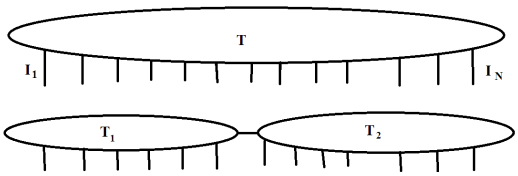

Fig. 2. Strangulation: An approximation of an order-N tensor by a contraction of two tensors smaller in size.

a contraction of these wagons; see Fig. 1. If the wagons are denoted $\mathbf{w}_1, \ldots, \mathbf{w}_N$, then their contraction will be written as $\mathcal{T} = \{\{\mathbf{w}\}\}$. Elements of the tensor train are computed as

$$T(i_1, \ldots, i_N) = \sum_{b_1=1}^{R_1} \sum_{b_2=1}^{R_2} \ldots \sum_{b_{N-1}=1}^{R_{N-1}} \mathbf{w}_1(i_1, b_1)\mathbf{w}_2(b_1, i_2, b_2) \ldots \mathbf{w}_N(b_{N-1}, i_N) \ .$$

The integers $R_1, \ldots, R_{N-1}$ are called bond dimensions. A continuous extension of the tensor train is the functional decomposition [10],

$$V(\boldsymbol{\xi}) = \mathbf{W}_1(x_1)\mathbf{W}_2(x_2) \ldots \mathbf{W}_N(x_N) \tag{1}$$

where $\boldsymbol{\xi} = [x_1, \ldots, x_n]$ and $\mathbf{W}_n(x_n)$, $n = 1, \ldots N$ are univariate matrix functions of scalars x_n for $n = 1, \ldots, N$. The bond dimensions refer to sizes of the matrices; $\mathbf{W}_n(\cdot)$ has the size $R_{n-1} \times R_n$ for $n = 1, \ldots, N$. Here, we set $R_0 = R_N = 1$.

For future descriptions, we need to consider the notion of tensor fiber. Let $\mathbf{v} = [v_1, \ldots, v_N]$ be a vector of indices, $v_n \in [1, \ldots, I_n]$. Then, the fiber with indices $\mathbf{v}$ along the dimension k is an $I_k \times 1$ vector

$$\mathbf{f}(\mathcal{T}, \mathbf{v}, k) = T(v_1, \ldots, v_{k-1}, :, v_{k+1}, \ldots, v_N)$$

The colon ":" means that the index at the $k-$th position is free, and all remaining indexes are fixed, selected from $\mathbf{v}$.

A. TT-SVD decomposition

The TT decomposition [5], [6] can be obtained through a series of *strangulations*, see Fig. 2. Here, the tensor $\mathcal{T}$ is approximated by a contraction of two smaller tensors, say $\mathcal{T}_1$ and $\mathcal{T}_2$. The strangulation can be accomplished by singular value decomposition (SVD) of a matrix obtained by reshaping the tensor.

The strangulation can be applied $N-1$ times to achieve the tensor train. Since SVD is a computationally cheap operation, the TT decomposition is easy to accomplish unless the number of the tensor elements is too large. In the latter case, a TT-cross algorithm is an option [7]–[9].

B. Skeleton Matrix Decomposition

The skeleton matrix decomposition is a special case of the TT-cross decomposition for order-2 tensors/matrices. Let an $N \times M$ matrix $\mathbf{M}$ have rank R, $R \leq N$, $R \leq M$. Let $\mathcal{I}$ be a set of R indices, $\mathcal{I} \subset \{1, \ldots, N\}$ and let $\mathcal{J}$ be a set of R indices, $\mathcal{J} \subset \{1, \ldots, M\}$. Let $\mathbf{M}(\mathcal{I}, :)$ denote a submatrix of $\mathbf{M}$ composed of the rows of the matrices with indices in $\mathcal{I}$. Similarly, $\mathbf{M}(:, \mathcal{J})$ denotes a submatrix of $\mathbf{M}$ composed of the columns of the matrices with indices in $\mathcal{J}$. Finally, $\mathbf{M}(\mathcal{I}, \mathcal{J})$ is the $R \times R$ submatrix with row indices in $\mathcal{I}$ and column indices in $\mathcal{J}$.

Assume the submatrix $\mathbf{M}(\mathcal{I}, \mathcal{J})$ is invertible. Then, it holds [7]

$$\mathbf{M} = \mathbf{M}(:, \mathcal{J})[\mathbf{M}(\mathcal{I}, \mathcal{J})]^{-1}\mathbf{M}(\mathcal{I}, :) \ . \tag{2}$$

In other words, we can write $\mathbf{M}$ as a product of two "wagons",

$$\mathbf{M} = \mathbf{w}_1\mathbf{w}_2 \tag{3}$$

where $\mathbf{w}_1 = \mathbf{M}(:, \mathcal{J})$, $\mathbf{w}_2 = [\mathbf{M}(\mathcal{I}, \mathcal{J})]^{-1}\mathbf{M}(\mathcal{I}, :)$. The bond dimension is the rank R. Note that the sets of the indices $\mathcal{I}$ and $\mathcal{J}$ are not selected at random but through an *maxvol* algorithm that aims to maximize the absolute value of the determinant of $\mathbf{M}(\mathcal{I}, \mathcal{J})$ [15].

C. TT-cross decomposition

The TT-cross decomposition exists in several variants [7]–[9]. In all of them, as in the skeleton matrix decomposition, the tensor has to be evaluated along some bundles of fibers. Assume that for each $n = 1, \ldots, N$, we are given a set of the fibers along the dimension n stored as columns of a matrix $\mathbf{F}_n$ of the size $I_n \times B_n$, and indices (position) of the fibers in matrix $\mathbf{V}_n$ of the size $N \times B_n$. If $\mathbf{v}_k$ is the $k-$th column of $\mathbf{V}_n$, then the $k-$th column of $\mathbf{F}_n$ is $\mathbf{f}(\mathcal{T}, \mathbf{v}_k, n)$ for $k = 1, \ldots, B_n$. The number B_n (the number of the columns) depends on the bond dimensions. The variants of the algorithm differ in the way how the fibers in $(\mathbf{F}_n, \mathbf{V}_n)$ are selected. Here, we describe the main idea only.

Assume that the tensor obeys the TT model with bond dimensions $(1, R_1, \ldots, R_{N-1}, 1)$. If it is reshaped into the matrix $\mathbf{M}_1$ of the size $I_1 \times (I_2 \ldots I_N)$, the matrix would have rank R_1. The bundle of the fibers $\mathbf{F}_1$ is a submatrix of $\mathbf{M}_1$, i.e. $\mathbf{F}_1 = \mathbf{M}_1(:, \mathcal{J}_1)$ where $\mathcal{J}_1$ depends on the indices in $\mathbf{V}_1$. If the fibers in $\mathbf{F}_1$ were selected well and their number is $B_1 = R_1$, then $\mathbf{F}_1$ has full rank, R_1. Applying the *maxvol* algorithm, we can find the index set $\mathcal{I}_1$ so that absolute value of the determinant of $\mathbf{F}_1(\mathcal{I}_1, :)$ is maximized. The first wagon in the train is then $\mathbf{w}_1 = (\mathbf{F}_1(\mathcal{I}_1, :))^{-1}\mathbf{F}_1$.

Next, we consider a matrix $\mathbf{M}_2$ obtained by reshaping the sub tensor $\mathcal{T}(\mathcal{I}_1, :, \ldots, :)$ to the size $(R_1 I_2) \times (I_3 \ldots I_N)$. According to the TT model, $\mathbf{M}_2$ has rank R_2. We can select R_2 its columns (e.g., randomly, and $\mathbf{V}_2$ will be constructed accordingly) to get a submatrix $\widetilde{\mathbf{M}_2}$ of the size $(R_1 I_2) \times R_2$. Then, $\widetilde{\mathbf{M}_2}$ can be obtained by reshaping $\mathbf{F}_2$ or vice versa. Note that $\mathbf{V}_2$ and $\mathbf{F}_2$ contain $B_2 = R_1 R_2$ columns. Now, we apply the *maxvol* algorithm to $\widetilde{\mathbf{M}_2}$ to find the set $\mathcal{I}_2$ of R_2

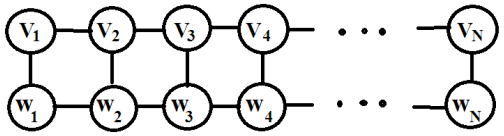

Fig. 3. Scalar product of two tensor trains $\{\{\mathbf{v}\}\}$ and $\{\{\mathbf{w}\}\}$

indices such that $\widetilde{\mathbf{M}_2}(\mathcal{I}_2,:)$ has a large volume. The wagon $\mathbf{w}_2$ is obtained by reshaping the matrix $(\widetilde{\mathbf{M}_2}(\mathcal{I}_2,:))^{-1}\widetilde{\mathbf{M}_2}$ in the required format (R_1, I_2, R_2). Similarly, the other wagons are obtained. If the tensor $\mathcal{T}$ obeys the TT model exactly, we receive another exact (or identical) TT representation of the tensor.

D. Computing error of the TT model

It is not trivial to evaluate the accuracy of the TT approximation unless we can compare the full tensor and the TT model at every grid point. We can calculate the error only approximately.

One option is to select (e.g., randomly) a large number of the entries where the two tensors (the original $\mathcal{T}$ and its approximation $\widehat{\mathcal{T}}$) are compared; say $\{\mathbf{v}_1, \ldots, \mathbf{v}_P\}$, where $\mathbf{v}_p$, $p = 1, \ldots, P$ are vectors of the tensor indices. The approximation error would be computed as the standard deviation of errors between the selected entries of the tensors,

$$\varepsilon_P = \sqrt{\frac{1}{P}\sum_{p=1}^{P}[T(\mathbf{v}_p) - \widehat{T}(\mathbf{v}_p)]^2} \ . \tag{4}$$

A more sophisticated approximation of the estimation error can be obtained by TT modeling of the error tensor $\widehat{\mathcal{T}} - \mathcal{T}$. Let $\{\{\widetilde{\mathbf{w}}_i\}\}$=TTD$(\widehat{\mathcal{T}} - \mathcal{T})$ be its TT approximation. The error

$$\varepsilon_{\text{TTD}} = \frac{1}{\sqrt{n_e}}\|\text{TTD}(\widehat{\mathcal{T}} - \mathcal{T})\|_F \ . \tag{5}$$

where n_e is the number of the element in $\mathcal{T}$. To compute ε_{TTD}, we use the same procedure applied to obtain $\widehat{\mathcal{T}} = \{\{\mathbf{w}_i\}\}$, possibly with lower bond dimensions, because the error can be estimated with lower accuracy. We leave the question of the bond dimension of the error tensor to future experimental analysis; we only note that we obtained satisfactory results with quite low bond dimensions. Under normal condition and large P, both errors should be similar, $\varepsilon_P \approx \varepsilon_{\text{TTD}}$. Note that the Frobenius norm of a tensor in TT format can be computed with a low complexity without handling all the tensor elements, see Fig. 3.

III. ALS FOR TT DECOMPOSITION

In this section, the computation of TT decomposition by ALS is proposed. We start with the case $N = 2$, i.e, decomposition of a matrix $\mathbf{M}$. Assume that we are given some initial rank-R model of the matrix, i.e., matrices $\mathbf{A}_0, \mathbf{B}_0$, obtained, e.g., by the skeleton algorithm, and next assume that we get two submatrices $\mathbf{F}_1 = \mathbf{M}(:, \mathcal{J})$ and $\mathbf{F}_2^T = \mathbf{M}(\mathcal{I}, :)$ of $\mathbf{M}$ on which the model $\widehat{\mathbf{M}} = \mathbf{A}\mathbf{B}^T$ will be fitted. The sets $\mathcal{I}$, $\mathcal{J}$ should contain the indices used in the skeleton algorithm and some additional ones. We wish to minimize the criterion penalizing a discrepancy between submatrices $\mathbf{F}_1$, $\mathbf{F}_2$ and their approximations

$$\begin{aligned}\mathcal{E}(\mathbf{A},\mathbf{B}) &= \|\mathbf{F}_1 - \widehat{\mathbf{F}}_1(\mathbf{A},\mathbf{B})\|_F^2 + \|\mathbf{F}_2 - \widehat{\mathbf{F}}_2(\mathbf{A},\mathbf{B})\|_F^2 \\ &\quad -\|\mathbf{F}_{12} - \widehat{\mathbf{F}}_{12}(\mathbf{A},\mathbf{B})\|_F^2 \ .\end{aligned} \tag{6}$$

where $\mathbf{F}_{12} = \mathbf{M}(\mathcal{I}, \mathcal{J})$, $\widehat{\mathbf{F}}_1(\mathbf{A},\mathbf{B}) = \widehat{\mathbf{M}}(:, \mathcal{J}) = \mathbf{A}\mathbf{B}(\mathcal{J},:)^T$, and $\widehat{\mathbf{F}}_2(\mathbf{A},\mathbf{B}) = \widehat{\mathbf{M}}(\mathcal{I},:)^T = \mathbf{B}\mathbf{A}(\mathcal{I},:)$. Note that the third term in (6) is already included in both the first two terms, and therefore, it is subtracted so that the equal weight applies to all available tensor elements.

The ALS method consists of a cyclic optimization of the above criterion with respect to $\mathbf{A}$ and $\mathbf{B}$. Details are summarized in Algorithm 1.

Algorithm 1 (ALS for fibers'-based matrix decomposition)
Input: Index sets $\mathcal{I}$, $\mathcal{J}$, fiber matrices $\mathbf{F}_1 = \mathbf{M}(:, \mathcal{J})$,
$\mathbf{F}_2 = \mathbf{M}(\mathcal{I},:)^T$, initial factors $\mathbf{A} = \mathbf{A}_0$, $\mathbf{B} = \mathbf{B}_0$
Repeat until convergence
$\mathbf{A}(\mathcal{I},:) = \mathbf{F}_2^T(\mathbf{B}^T)^\dagger$
$\mathbf{A}(\overline{\mathcal{I}},:) = \mathbf{F}_1(\overline{\mathcal{I}},:)(\mathbf{B}(\mathcal{J},:)^T)^\dagger$
$\mathbf{B}(\mathcal{J},:) = \mathbf{F}_1^T(\mathbf{A}^T)^\dagger$
$\mathbf{B}(\overline{\mathcal{J}},:) = \mathbf{F}_2(\overline{\mathcal{J}},:)(\mathbf{A}(\mathcal{I},:)^T)^\dagger$
End
$\overline{\mathcal{I}} = \{1, \ldots, M\} - \mathcal{I}$, $\overline{\mathcal{J}} = \{1, \ldots, N\} - \mathcal{J}$,
† denotes matrix pseudoinverse.

The ALS for higher-order *tensor* decomposition in the tensor fibers is similar. The input is the set of the fiber matrices $\mathbf{F}_n$ with indices in $\mathbf{V}_n$, $n = 1, \ldots, N$, and an initial tensor train $\{\{\mathbf{w}_n\}\}$. For simplicity, we do not care about the intersection of the fibers (which is an analogy to $\mathbf{F}_{12}$ and may be empty) and minimize the criterion

$$\mathcal{E}(\{\mathbf{w}\}) = \sum_{n=1}^{N}\|\mathbf{F}_n - \widehat{\mathbf{F}}_n(\{\mathbf{w}\})\|_F^2$$

where $\widehat{\mathbf{F}}_n(\{\mathbf{w}\})$ is the matrix of fibers for $\widehat{\mathcal{T}} = \{\{\mathbf{w}\}\}$ (with indices in $\mathbf{V}_n$), and $\|\cdot\|_F$ is the Frobenius norm.

All fiber matrices are separately linear functions of each wagon in the train. Say

$$\text{vec}(\widehat{\mathbf{F}}_m) = \mathbf{L}_{mn}\text{vec}(\mathbf{w}_n)$$

for $m, n = 1, \ldots, N$, where $\mathbf{L}_{mn}$ are suitable matrices that depend on all other wagons $\mathbf{w}_k$, $k \neq n$, and "vec" the vectorization operator. The least-square estimate of the $n-$th wagon is then

$$\text{vec}(\hat{\mathbf{w}}_n) = \left[\sum_{m=1}^{N}\mathbf{L}_{mn}^T\mathbf{L}_{mn}\right]^{-1}\left[\sum_{m=1}^{N}\mathbf{L}_{mn}^T\text{vec}(\mathbf{F}_m)\right] \ .$$

Details of the algorithm (i.e., the matrices $\mathbf{L}_{mn}$) are presented in Appendix A.

The algorithm has the advantage that the number of fibers used for estimation can be increased adaptively. The quality of the current model (the approximation error) can be tested on

a bunch of randomly selected fibers. If the error is not small enough, it is advised to add the testing fibers to the set of fibers used for estimation and perform additional iterations of the ALS algorithm to distribute the error among all the fibers. Note that the ALS algorithm can still be modified using the ideas in [18] to enhance its performance.

IV. Examples

A. Noisy Rosenbrock function

In this example, we consider a Rosenbrock function [23]. It is a non-convex function used as a performance test problem for optimization algorithms. A multidimensional version of the function is defined as

$$f_N(x_1,\dots,x_N) = \sum_{i=1}^{N-1} b(x_{i+1}-x_i^2)^2 + (a-x_i)^2 \ . \quad (7)$$

Typically, $a=1$, $b=100$.

In [22], it was shown that for $N=3$, the tensor can be decomposed as a TT with a bond dimension $(1,3,3,1)$. Applying the skeleton TT algorithm shows that for higher N, the bond dimension is $(1,3,\dots,3,1)$, at least for all dimensions we tried (up to 20). The tensor was sampled on the grid $(-2:0.1:2)^{20}$ so that the tensor has 41^{20} elements, more than any existing computer can handle. We can apply the TT-cross algorithm to obtain an exact TT representation of the tensor. The fitting error is a machine zero.

A noisy version of the tensor will be obtained by rounding the tensor elements to the closest integers. Each tensor element's expected mean square error (assuming a uniform distribution of the errors) is then 1/12, and the standard deviation is $\sqrt{3}/6 \approx 0.222$. The TT-cross decomposition of the noisy tensor has the problem that it provides either a poor low-rank approximation based on a few tensor fibers only or a high-rank (high bond dimension) approximation with moderate but not small error because of over-fitting. To be specific, we compute the fitting error as

$$\phi = \frac{1}{\sqrt{n_T}} \|\widehat{\mathcal{T}} - \mathcal{T}\|_F$$

where n_T is the number of elements of $\mathcal{T}$.

First, we applied the algorithm *greedy2_cross* of [13] with a tolerance such that the bond dimension was 3. The resultant fitting error ϕ was 29.14. The bond dimension grew to 21 with a lower tolerance, and the fitting error became 1.0910. It was impossible to reduce the error ϕ further by lowering the tolerance and increasing the bond dimension. Second, we applied 20 sweeps of our ALSTT algorithm with bond dimension 3 and a growing number of random tensor fibers. The result is shown in Fig. 4. We can see a decrease in the error. With 500 randomly chosen fibers along each dimension, the error decreased to 0.05. The advantage of the proposed algorithm is that it allows for the achievement of much lower fitting errors than TT-core in noisy scenarios.

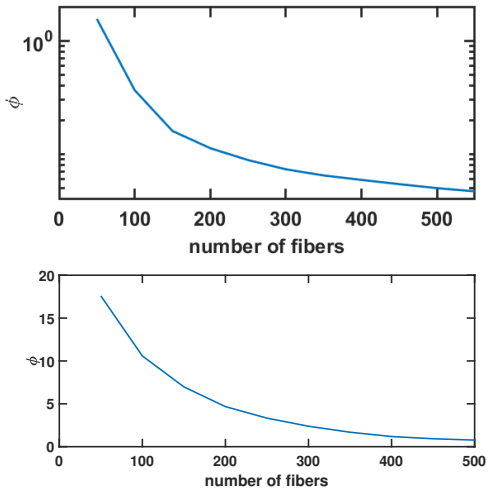

Fig. 4. Fitting error as a function of number of added fibers. Upper diagram: Noisy Rosenbrock function. Lower diagram: Noisy quadratic function.

B. Noisy Quadratic Function

Let $\mathbf{Q}$ be a general positive definite matrix of the size $N \times N$. Consider the quadratic function

$$V(\boldsymbol{\xi}) = \boldsymbol{\xi}^T \mathbf{Q} \boldsymbol{\xi} \quad (8)$$

It can be shown that this function can be represented as a functional tensor train (1) with maximum bond dimension $N/2+2$. For example, for $N=20$, the bond dimensions can be $(1,3,4,5,6,7,8,9,10,11,12,11,10,9,8,7,6,5,4,3,1)$, see Appendix B. We can choose any grids, e.g., $(-2:0.1:2)^{20}$ again, and a tensor train with zero fitting error with this bond dimension exists. The matrix $\mathbf{Q}$ was chosen such as $\mathbf{Q} = \mathbf{X}\mathbf{X}^T$ where $\mathbf{X}$ is a random 20×20 matrix with elements generated as independent with the standard normal distribution, $\mathcal{N}(0,1)$. Again, a noisy tensor will be obtained by rounding the tensor elements to the closest integers. The task is to reconstruct the original (noiseless) tensor by tensor train modeling. The results are similar to the noisy Rosenbrock function. The fitting error of the TT-cross was 5.22; ALS with added 50 fibers had an even worse performance of 17.59. However, the error decreased to a value of 0.7692 when 500 random fibers were added. The difference with the Rosenbrock function is that the quadratic function is more complex. It has more parameters and is, therefore, more difficult to fit.

V. Conclusions

The proposed algorithm allows the modeling of high-order multivariate functions in the presence of noise better than the traditional TT-cross algorithm. We plan to use it to solve the Bellman equation in dynamic programming. In this application, the multivariate Bellman function varies in an iterative

scheme in each step. We wish to handle it in the TT format and thus break the curse of dimensionality.

APPENDIX A

In this Appendix, we show how one fiber of the tensor $\widehat{\mathcal{T}} = \{\{\mathbf{w}\}\}$ along dimension m depends on one wagon of the train, say $\mathbf{w}_n$, which may be subject of optimization, $n, m = 1, \ldots, N$. Let the fiber $\mathbf{f}$ be given as $\mathbf{f} = \mathbf{f}(\mathcal{T}, \mathbf{v}, m)$, i.e., its indices are in vector $\mathbf{v}$.

Assume for simplicity that $n = m$. Let $\mathbf{X}_L$ be the left part of the train with the wagons $\mathbf{w}_1, \ldots, \mathbf{w}_{n-1}$ with the middle entry replaced by the corresponding element of $\mathbf{v}$. Then, $\mathbf{X}_L$ is the product of one vector and $n-2$ matrices,

$$\mathbf{X}_L = \mathbf{w}_1(1, v_1, :)\mathbf{w}_2(:, v_2, :) \ldots \mathbf{w}_{n-1}(:, v_{n-1}, :) \ .$$

so that vector of the length R_{n-1}. Similarly the right part, $\mathbf{X}_R$ is the vector

$$\mathbf{X}_R = \mathbf{w}_{n+1}(:, v_{n+1}, :)\mathbf{w}_{n+2}(:, v_{n+2}, :) \ldots \mathbf{w}_N(:, v_N, 1) \ .$$

With this notation, it holds

$$\mathbf{f} = \mathbf{w}_n \times_1 \mathbf{X}_L \times_3 \mathbf{X}_R$$

where $\times_k$ denotes the tensor-matrix multiplication along the dimension k, $k = 1, 2, 3$. The last equation can be rewritten as

$$\mathbf{f} = (\mathbf{X}_R \otimes \mathbf{I}_{I_n} \otimes \mathbf{X}_L)\mathrm{vec}\, \mathbf{w}_n$$

where $\otimes$ denotes the Kronecker product and $\mathbf{I}_{I_n}$ is the identity matrix of the size $I_n \times I_n$. The construction of the matrices $\mathbf{L}_{mn}$ is an extension of these ideas.

APPENDIX B

Let $\mathbf{Q}$ be a symmetric matrix of the size $N \times N$, $N > 1$. Consider the quadratic function of N variables,

$$V_N(\boldsymbol{\xi}) = \boldsymbol{\xi}^T \mathbf{Q} \boldsymbol{\xi}$$

where $\boldsymbol{\xi} = (x_1, \ldots, x_N)$. We present $V_N(\boldsymbol{\xi})$ in the form of the functional tensor train decomposition (1) for $N = 2, 3, 4$ for now only. The extension to higher N is straightforward.

1) For $N = 2$, the bond dimensions are $(1, 3, 1)$

$$V_2(\boldsymbol{\xi}) = [x_1^2,\, x_1,\, 1][Q_{11},\, 2Q_{12}x_2,\, Q_{22}x_2^2]^T$$

2) For $N = 3$, the bond dimensions are $(1, 3, 3, 1)$

$$V_3(\boldsymbol{\xi}) = [x_1^2, x_1, 1] \begin{bmatrix} 0 & 0 & Q_{11} \\ 0 & 2Q_{13} & 2Q_{12}x_2 \\ Q_{33} & 2Q_{23}x_2 & Q_{22}x_2^2 \end{bmatrix} \begin{bmatrix} x_3^2 \\ x_3 \\ 1 \end{bmatrix}$$

3) For $N = 4$, the bond dimensions are $(1, 3, 4, 3, 1)$

$$V_4(\boldsymbol{\xi}) \quad = \quad [x_1^2, x_1, 1] \begin{bmatrix} 0 & 0 & 0 & Q_{11} \\ 0 & 0 & 1 & 2Q_{12}x_2 \\ 1 & x_2 & 0 & Q_{22}x_2^2 \end{bmatrix} \cdot \begin{bmatrix} Q_{44} & 2Q_{34}x_3 & Q_{33}x_3^2 \\ 0 & 2Q_{24} & 2Q_{23}x_3 \\ 0 & 2Q_{14} & 2Q_{13}x_3 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_4^2 \\ x_4 \\ 1 \end{bmatrix}$$

REFERENCES

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

[2] J.D. Carroll and J.J. Chang, "Analysis of individual differences in multidimensional scaling via an n-way generalization of Eckart-Young decomposition," *Psychometrika*, vol. 35, no. 3, pp. 283–319, 1970.

[3] N.D. Sidiropoulos, L. De Lathauwer, X. Fu, K Huang, E.E Papalexakis, and C. Faloutsos, "Tensor decomposition for signal processing and machine learning", *IEEE Transactions on Signal Processing*, vol. 65, no. 13, pp. 3551–3582, 2017.

[4] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Review*, vol. 51, no. 3, pp. 455–500, Sep. 2009.

[5] I.V. Oseledets and E.E. Tyrtyshnikov, "Breaking the curse of dimensionality, or how to use SVD in many dimensions," *SIAM Journal on Scientific Computing*, vol. 31, no. 5, pp. 3744–3759, 2009.

[6] I. V. Oseledets, "Tensor–train decomposition", *SIAM J. Sci. Comput.*, vol. 33, no. 5, pp. 2295–2317, Jan. 2011.

[7] I. V. Oseledets and E. E. Tyrtyshnikov, "TT-cross approximation for multidimensional arrays", *Linear Algebra Appl.*, vol. 432, no. 1, pp. 70–88, 2010. doi: 10.1016/j.laa.2009.07.02

[8] E. Tyrtyshnikov, "Incomplete cross approximation in the mosaic-skeleton method". *Computing*, vol. 64, no.4, pp. 367–380, 2000.

[9] D.V. Savostyanov, "Quasioptimality of maximum-volume cross interpolation of tensors", *Linear Algebra and its Applications*, Vol. 458, pp. 217-244, 2014.

[10] A. Gorodetsky, S. Karaman, Y.M. Marzouk, "A continuous analogue of the tensor-train decomposition", *Computer Methods in Applied Mechanics and Engineering*, vol. 347, no.4, pp. 59-84, 2018. DOI: 10.1016/j.cma.2018.12.015

[11] A. Gorodetsky, S. Karaman, Y.M. Marzouk, "High-dimensional stochastic optimal control using continuous tensor decompositions", arXiv:1611.04706v2 [cs.RO] 11 Jan 2018.

[12] K. Sozykin, A. Chertkov, R. Schutski, A.-H. Phan, A. Cichocki, and I. Oseledets, "TTOpt: A Maximum volume quantized tensor train-based optimization and its application to reinforcement learning", Proc. Thirty-sixth Conference on Neural Information Processing Systems (NeurIPS 2022) https://arxiv.org/abs/2205.00293.

[13] I. Oseledets, "MATLAB Toolbox for working with high-dimensional tensors in the Tensor-Train (TT)-format" https://github.com/oseledets/TT-Toolbox.

[14] S. R. White, "Density matrix formulation for quantum renormalization groups.", *Physical Review Letters* 69 (1992), pp. 28–63.

[15] S.A. Goreinov, E.E. Tyrtyshnikov, "The maximal-volume concept in approximation by low-rank matrices", *Contemporary Math.* vol. 208, pp. 47–51, 2001.

[16] S. A. Goreinov, E. E. Tyrtyshnikov, N. L. Zamarashkin, "A theory of pseudo-skeleton approximations", *Linear Algebra Appl.*, vol. 261, pp. 1–21, (1997).

[17] S. Holtz, T. Rohwedder, and R. Schneider, "The alternating linear scheme for tensor optimization in the tensor train format", *SIAM J. Sci. Comput.*, vol. 34, no. 2, pp. A683–A713, 2012. doi: 10.1137/100818893.

[18] L. Grasedyck, M. Kluge, and S. Krämer, "Variants of alternating least squares tensor completion in the tensor train format", *SIAM Journal on Scientific Computing*, vol. 37, no. 5, pp. A2424-A2450 (2015)10.1137/130942401

[19] M. Sörensen and L.De Lathauwer, "Fiber sampling approach to canonical polyadic decomposition and application to tensor completion", *SIAM Journal on Matrix Analysis and Applications*, vol. 40, no. 3, pp 888–917, 2019.

[20] N. Vervliet, O. Debals, L. Sorber, L. De Lathauwer, "Breaking the curse of dimensionality using decompositions of incomplete tensors: Tensor-based scientific computing in big data analysis". *IEEE Signal Processing Magazine*, vol. 31, no. 5, pp. 71–79, 2014.

[21] A. Sobral, E. Zahzah, "Matrix and tensor completion algorithms for background model initialization: A comparative evaluation", *Pattern Recognition Letters*, vol. 96, pp. 22–33, 2017.

[22] P. Tichavský, and A.H. Phan, "Tensor chain decomposition and function interpolation", *IEEE Statistical Signal Processing Workshop*, Hanoi, Vietnam, pp. 557–561, 2023.

[23] H.H. Rosenbrock, "An automatic method for finding the greatest or least value of a function". *The Computer Journal.*, Vol. 3, no. 3, pp. 175–184, 1960.