SIAM J. SCI. COMPUT.
Vol. 45, No. 4, pp. A2101–A2118

BLACK BOX APPROXIMATION IN THE TENSOR TRAIN FORMAT INITIALIZED BY ANOVA DECOMPOSITION[*]

ANDREI CHERTKOV[†], GLEB RYZHAKOV[†], AND IVAN OSELEDETS[‡]

Abstract. Surrogate models can reduce computational costs for multivariable functions with an unknown internal structure (black boxes). In a discrete formulation, surrogate modeling is equivalent to restoring a multidimensional array (tensor) from a small part of its elements. The alternating least squares (ALS) algorithm in the tensor train (TT) format is a widely used approach to effectively solve this problem in the case of nonadaptive tensor recovery from a given training set (i.e., tensor completion problem). TT-ALS allows obtaining a low-parametric representation of the tensor, which is free from the curse of dimensionality and can be used for fast computation of the values at arbitrary tensor indices or efficient implementation of algebra operations with the black box (integration, etc.). However, to obtain high accuracy in the presence of restrictions on the size of the train data, a good choice of initial approximation is essential. In this work, we construct the ANOVA representation in the TT-format and use it as an initial approximation for the TT-ALS algorithm. The performed numerical computations for a number of multidimensional model problems, including the parametric partial differential equation, demonstrate a significant advantage of our approach for the commonly used random initial approximation. For all considered model problems we obtained an increase in accuracy by at least an order of magnitude with the same number of requests to the black box. The proposed approach is very general and can be applied in a wide class of real-world surrogate modeling and machine learning problems.

Key words. ALS, ANOVA, black box approximation, low-rank representation, tensor train

MSC codes. 65D15, 41A63, 35A17

DOI. 10.1137/22M1514088

1. Introduction.

Many physical and engineering models can be represented as a real function (output of the model), which depends on a multidimensional argument (input of the model) and looks like

$$y = \mathsf{f}(\boldsymbol{x}) \in \mathbb{R}, \quad \boldsymbol{x} = [x_1, x_2, \ldots, x_d]^T \in \Omega \subset \mathbb{R}^d. \tag{1.1}$$

Such functions often have a form of a black box (BB), i.e., its internal structure and smoothness properties remain unknown. The time and/or required resources for one computation of the function f may be significant, and it is relevant to train some surrogate model g (low-parametric approximation) that can be evaluated quickly, but at the same time remains sufficiently close to the original function. Then such a simple model g can be used instead of the original BB for faster computations of its outputs from given inputs. Moreover, the statistical characteristics of the BB may be approximately recovered from this model g by fast Monte Carlo sampling or with the usage of the known specific internal structure of the function g.

[*]Submitted to the journal's Methods and Algorithms for Scientific Computing section August 5, 2022; accepted for publication (in revised form) March 23, 2023; published electronically August 11, 2023.
https://doi.org/10.1137/22M1514088
Funding: This work was supported by the Ministry of Science and Higher Education of the Russian Federation under grant 075-10-2021-068.
[†]Skolkovo Institute of Science and Technology, Moscow, 121205, Russia (a.chertkov@skoltech.ru, g.ryzhakov@skoltech.ru).
[‡]Skolkovo Institute of Science and Technology, Moscow, 121205, Russia, and AIRI, Moscow, Russia (i.oseledets@skoltech.ru).

The model g may be a decomposition by some basis functions, for example, Chebyshev polynomials [36], but in many cases, it is more natural to directly discretize the target function (1.1) on a multidimensional grid

$$\left\{x_1^{(n_1)}, x_2^{(n_2)}, \ldots, x_d^{(n_d)}\right\}, \quad n_k = 1, 2, \ldots, N_k \;\text{ for }\; k = 1, 2, \ldots, d, \tag{1.2}$$

and then represent the BB as an implicitly specified multidimensional array (tensor)[1] $\mathcal{Y} \in \mathbb{R}^{N_1 \times N_2 \times \cdots \times N_d}$ that collects all possible discrete values of the function (1.1) inside the domain Ω, i.e.,

$$\mathcal{Y}[n_1, n_2, \ldots, n_d] = \mathsf{f}\left(x_1^{(n_1)}, x_2^{(n_2)}, \ldots, x_d^{(n_d)}\right). \tag{1.3}$$

Undoubtedly, the task of explicitly constructing and storing such a tensor is too computationally expensive, and for large values of the dimension d, this is completely impossible due to the curse of dimensionality. However, the usage of the low-rank tensor approximations, namely tensor train (TT) decomposition [24], makes it possible to approximately represent the tensor in a compact low-parameter format using only a small number of explicitly computed elements. TT-decomposition is a common approach for compact approximation of multidimensional arrays and multivariable functions [10, 11, 27]. Approximation in the TT-format allows subsequent usage both for quick calculation of BB values and for constructing its various statistical characteristics. It is possible to effectively perform algebraic operations (element-by-element addition and multiplication, convolution, etc.) over tensors in the TT-format [24]. Thus, for example, it turns out to be more efficient in some cases to construct a surrogate model of a multidimensional tensor in the TT-format first, and then perform summation with it (see, e.g., [4, 30]).

TT-ALS (alternating least squares in the TT-format) [17, 16] is a tensor completion method that constructs the TT-approximation from a given set of tensor elements (e.g., random samples may be used), alternately optimizing the current approximation for each mode, with fixed values of the parameters corresponding to the rest of the modes. TT-ALS is a powerful tool for tensor approximation, and it has found a wide range of practical applications [32, 7]. To use TT-ALS, it is necessary to set an initial approximation, which can, for example, be given in the form of a random TT-tensor. However, the choice of this initial approximation plays an important role in the quality of the resulting representation in the TT-format and often the use of a random TT-tensor turns out to be unsatisfactory [20, 28, 14, 8]. So, in particular, with an unsuccessful initial approximation, too many iterations of the method may be required, or the resulting solution may converge to a local optimum that is not good enough. In this work, we propose a modified version of the ANOVA (analysis of variance) representation [33] in the TT-format as an initial approximation for the TT-ALS algorithm. The interdependence of the model inputs utilizing a split representation corresponding to the ANOVA decomposition allows us to significantly increase the stability of the TT-ALS approach for constructing a TT-tensor from restricted observations.

[1]By tensors we mean multidimensional arrays with a number of dimensions d ($d \geq 1$). A two-dimensional tensor ($d = 2$) is a matrix, and when $d = 1$ it is a vector. For scalars we use normal font, we denote vectors with bold letters, and we use uppercase calligraphic letters ($\mathcal{A}, \mathcal{B}, \mathcal{C}, \ldots$) for tensors with $d > 2$. The $(n_1, n_2, \ldots, n_d)$th entry of a d-dimensional tensor $\mathcal{A} \in \mathbb{R}^{N_1 \times N_2 \times \cdots \times N_d}$ is denoted by $\mathcal{A}[n_1, n_2, \ldots, n_d]$, where $n_k = 1, 2, \ldots, N_k$ ($k = 1, 2, \ldots, d$) and N_k is a size of the kth mode, and mode-k slice of such a tensor is denoted by $\mathcal{A}[n_1, \ldots, n_{k-1}, :, n_{k+1}, \ldots, n_d]$.

As a practically significant numerical example, we consider a parameter-dependent partial differential equation (PDE). Equations of this kind have a wide range of applications [12, 2], such as those associated with optimization or uncertainty quantification after random field discretization. These include groundwater heights in geotechnical engineering, soil parameters, wind loads, and snow loads in structural engineering, the amount of precipitation and evaporation in hydrology, etc. We consider the mean value of the PDE solution as a multivariable function of PDE parameters and construct its compact representation in the TT-format using the proposed approach. The performed numerical computations for this problem, as well as for a number of model analytical multivariable functions, demonstrate a significant advantage of our approach concerning the commonly used random initial approximation.

To summarize, our main contributions are the following:

- we construct the ANOVA expansion in the form of the TT-tensor (TT-ANOVA) and we propose to use it as an initial approximation for the TT-ALS algorithm to approximate the multidimensional BB (TT-ANOVA-ALS method);
- we implement the proposed algorithms as a publicly available Python package;[2]
- we apply our approach[3] for several multidimensional model problems, including the approximation of the parametric PDE solution, to demonstrate its robustness and performance.

2. Tensor train format.

There has been much interest lately in the development of data-sparse tensor formats for high-dimensional problems. A very promising tensor format is provided by the TT-approach [24]. It can be computed via standard SVD and QR decompositions [26] but does not suffer from the curse of dimensionality.

A tensor $\mathcal{Y} \in \mathbb{R}^{N_1 \times N_2 \times \cdots \times N_d}$ is said to be in the TT-format if its elements are represented by the following formula (see also the illustration in Figure 1):

$$\mathcal{Y}[n_1, n_2, \ldots, n_d] = \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \cdots \sum_{r_{d-1}=1}^{R_{d-1}} \mathcal{G}_1[1, n_1, r_1] \mathcal{G}_2[r_1, n_2, r_2] \cdots$$
$$\mathcal{G}_{d-1}[r_{d-2}, n_{d-1}, r_{d-1}] \mathcal{G}_d[r_{d-1}, n_d, 1], \tag{2.1}$$

where $n_k = 1, 2, \ldots, N_k$ $(k = 1, 2, \ldots, d)$ represent the multi-index, three-dimensional tensors $\mathcal{G}_k \in \mathbb{R}^{R_{k-1} \times N_k \times R_k}$ are named TT-cores, and integers $R_0, R_1, \ldots, R_d$ (with convention $R_0 = R_d = 1$) are named TT-ranks. The latter formula also can be rewritten in a more compact form,

$$\mathcal{Y}[n_1, n_2, \ldots, n_d] = G_1(n_1) G_2(n_2) \cdots G_d(n_d), \tag{2.2}$$

where $G_k(n_k) = \mathcal{G}_k[:, n_k, :]$ is an $R_{k-1} \times R_k$ matrix for each fixed n_k (since $R_0 = R_d = 1$, the result of matrix multiplications in (2.2) is a scalar).

The benefit of the TT-decomposition is the following. Storage of the TT-cores $\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_d$ requires less than or equal to $d \times \max_{1 \le k \le d} \left(N_k R_k^2 \right)$ memory cells instead of $N = N_1 N_2 \ldots N_d \sim N_0^d$ cells for the uncompressed tensor, where N_0 is an

[2]We implemented basic operations in the TT-format, as well as TT-ALS and TT-ANOVA approaches within the framework of the software product teneva, which is available from https://github.com/AndreiChertkov/teneva.

[3]The program code with numerical examples, given in this work, is publicly available in the repository https://github.com/AndreiChertkov/teneva_research_anova_and_als. For all considered model problems, including parametric PDE, we obtained an increase in accuracy by at least an order of magnitude with the same number of requests to the BB.

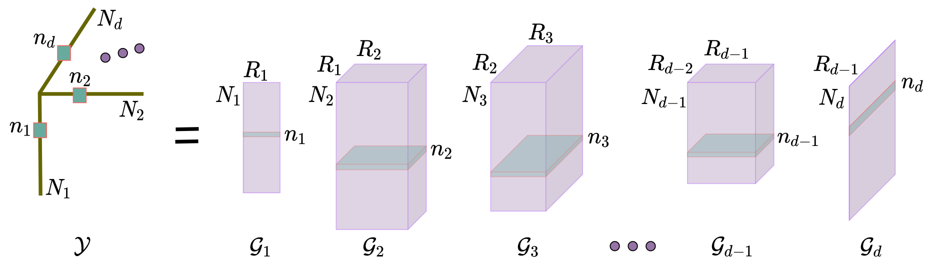

FIG. 1. *Schematic representation of the low-rank TT-decomposition. The tensor $\mathcal{Y} \in \mathbb{R}^{N_1 \times N_2 \times \cdots \times N_d}$ is represented in the low-rank TT-format as an ordered list of three-dimensional tensors $\mathcal{G}_k \in \mathbb{R}^{R_{k-1} \times N_k \times R_k}$ $(k = 1, 2, \ldots, d)$, which are named TT-cores. To compute an arbitrary element $(n_1, n_2, \ldots, n_d)$ of the tensor $\mathcal{Y}$, the product of the TT-cores according to the formula* (2.1) *or* (2.2) *should be performed.*

average size of the tensor modes, and hence the TT-decomposition is free from the curse of dimensionality if the TT-ranks are bounded. A detailed description of the TT-format and linear algebra operations in terms of this format is given in the works [26, 24].

An exact TT-representation exists for the given full tensor $\widehat{\mathcal{Y}}$, and TT-ranks of such representation are bounded by ranks of the corresponding unfolding matrices [24]. Nevertheless, in practical applications, it is more useful to construct TT-approximation with a prescribed accuracy ϵ_{TT}, and then carry out all operations (summations, products, etc.) in the TT-format, maintaining the same accuracy ϵ_{TT} of the result. For a given tensor $\widehat{\mathcal{Y}}$ in the full format and desired accuracy ϵ_{TT} in the Frobenius norm

$$||\mathcal{Y} - \widehat{\mathcal{Y}}||_F \leq \epsilon_{TT} \cdot ||\widehat{\mathcal{Y}}||_F, \tag{2.3}$$

the TT-decomposition (compression) can be performed by a stable TT-SVD algorithm, but this procedure of the tensor approximation from the full format is too costly and is even impossible for large dimensions since it requires the calculation of all elements of the original tensor. Therefore, more efficient algorithms, which allow constructing the TT-decomposition from a small part of the tensor elements (i.e., allow one to perform tensor completion), are used in multidimensional applications. We consider below the TT-ALS approach, which is one of the most popular such algorithms.

3. Black box approximation.

The proposed TT-ANOVA-ALS approach to improve the stability of the BB approximation in terms of the TT-format is schematically shown in Figure 2. We build the rough approximation for the BB by the TT-ANOVA algorithm (which is presented below in subsection 3.1), using K random samples from the BB, and then apply the well-known TT-ALS algorithm (it is briefly discussed below in subsection 3.2) on the same K samples using the result of the TT-ANOVA as an initial approximation. Note that in this model, there is no need to access the BB during iterations of the algorithm; just a random training dataset is sufficient. We show by numerical examples that this approach significantly increases the final accuracy of the approximation with virtually no increase in the computational complexity and eliminates the possibility of an accidental failure of the TT-ALS when the random initial approximation is chosen poorly.

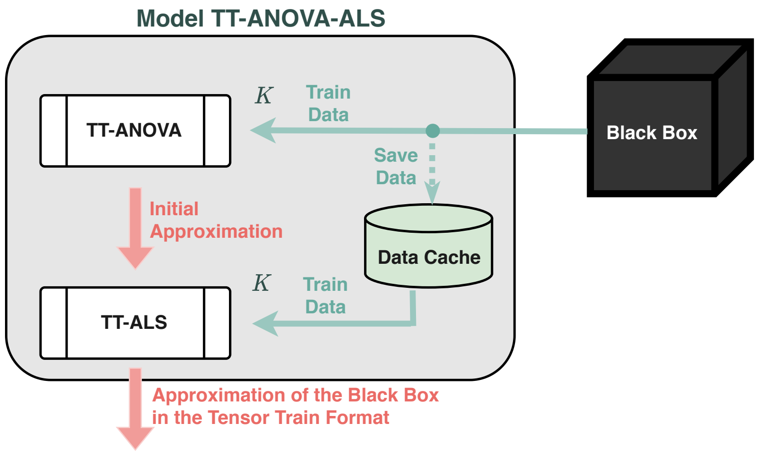

FIG. 2. *The proposed TT-ANOVA-ALS computation approach for the discretized BB approximation based on a random training dataset.*

3.1. ANOVA decomposition in the tensor train format. We assume that our model function (1.1) is defined in the hypercube $\Omega = [0,\,1]^d$ and integrable. Consider its ANOVA representation (or decomposition into summands of different dimensions) [33]

$$\text{(3.1)}\quad \mathsf{f}(x_1,\ldots,x_d) = \mathsf{f}_0 + \sum_{i=1}^{d}\mathsf{f}_i(x_i) + \sum_{1\le i<j\le d}\mathsf{f}_{ij}(x_i,\,x_j) + \cdots + \mathsf{f}_{123\ldots d}(x_1,\,\ldots,\,x_d),$$

where for all the functions $\mathsf{f}_{i_1 i_2\ldots i_s}$ $(1 \le i_1 < i_2 < \cdots < i_s \le d,\ s = 1,2,\ldots,d)$ the following equality holds:

$$\text{(3.2)}\qquad \int_\Omega \mathsf{f}_{i_1\ldots i_k\ldots i_s}(x_{i_1},\,\ldots,\,x_{i_k},\,\ldots,\,x_{i_s})\,dx_{i_k} = 0 \quad \text{for}\ \ k = 1,\,2,\,\ldots,\,s.$$

It can be shown from the definition of ANOVA (3.1) and (3.2) that this decomposition is unique and all the members are orthogonal, hence the zero and first-order terms can be expressed as follows:

$$\text{(3.3)}\qquad \mathsf{f}_0 = \int_\Omega \mathsf{f}(\boldsymbol{x})\,d\boldsymbol{x}, \quad \mathsf{f}_i(x_i) = \int_\Omega \mathsf{f}(\boldsymbol{x}) \prod_{k\ne i} dx_k - \mathsf{f}_0 \quad \text{for}\ \ i = 1,\,2,\,\ldots,\,d.$$

The analysis of variance (ANOVA) procedure is so named because it breaks down (i.e., analyzes) the variance into terms that quantify the magnitude of the overall variance in the response variable $\mathsf{f}(\boldsymbol{x})$ attributable to the parameters $\boldsymbol{x}$ and their interaction. The ANOVA representation (3.1) found a wide range of practical applications in science and technology and it is often used to estimate the sensitivity of a function to input variables or their combinations (Sobol indices) and to understand the relationship of variables or their effect on the model output [33, 5]. This information may be useful to eliminate uncertainty or simplify the model, i.e., the variables on which the output depends weakly can be fixed ("frozen"). However, in our case, the attractiveness of ANOVA lies in the possibility of its fast and efficient representation in the form of a TT-tensor using some samples from the objective function f. A more detailed analysis of this well-known decomposition can be found in [1, 6, 33].

Suppose we have a training dataset $(X^{(train)}, Y^{(train)})$

$$X^{(train)} = \left\{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \dots, \boldsymbol{x}^{(M)}\right\}, \qquad \boldsymbol{x}^{(m)} \in \mathbb{R}^d,$$
$$Y^{(train)} = \left\{y^{(1)}, y^{(2)}, \dots, y^{(M)}\right\}, \qquad y^{(m)} = \mathsf{f}\left(\boldsymbol{x}^{(m)}\right) \in \mathbb{R},$$

obtained from a uniform distribution, or from any low discrepancy sequence (LHS, Sobol, etc.). Note that for the tensor approximation problem each variable x_i ($i = 1, 2, \dots, d$) takes only a discrete set of values $v^{(i)} = \{v_1^{(i)}, v_2^{(i)}, \dots, v_{N_i}^{(i)}\}$, and moreover, the number of these values N_i for different variables may be different.[4] Then, we introduce the notation

$$X_{i,j}^{(train)} = \left\{\boldsymbol{x} \in X^{(train)} \mid x_i = v_j^{(i)}\right\} \quad \text{for } j = 1, 2, \dots, N_i, \; i = 1, 2, \dots, d.$$

In other words, the set $X_{i,j}^{(train)}$ collects those and only those vectors from $X^{(train)}$ for which the ith component coincides with its jth possible value.

Taking into account the introduced notation, we can represent the zero and first-order terms from (3.3) as

$$\widehat{\mathsf{f}}_0 \approx \frac{1}{M} \sum_{m=1}^{M} \mathsf{f}(\boldsymbol{x}^{(m)}) = \text{mean}\left\{\mathsf{f}(\boldsymbol{x}) \mid \boldsymbol{x} \in X^{(train)}\right\}, \tag{3.4}$$

$$\widehat{\mathsf{f}}_i\left(v_j^{(i)}\right) \approx \frac{1}{\left|X_{i,j}^{(train)}\right|} \sum_{\boldsymbol{x} \in X_{i,j}^{(train)}} \mathsf{f}(\boldsymbol{x}) - \widehat{\mathsf{f}}_0 = \text{mean}\left\{\mathsf{f}(\boldsymbol{x}) \mid \boldsymbol{x} \in X_{i,j}^{(train)}\right\} - \widehat{f}_0, \tag{3.5}$$

where $j = 1, 2, \dots, N_i$, $i = 1, 2, \dots, d$, and $|\cdot|$ denotes the cardinality of a set.

The resulting approximations (3.4) and (3.5) allow us to construct the discretized first-order ANOVA representation for a given training dataset. In the following theorem, we consider a way to explicitly convert this representation to the TT-format.

THEOREM 3.1. *Consider the TT-decomposition $\mathcal{Y}$ for tensor $\hat{\mathcal{Y}} \in \mathbb{R}^{N_1 \times N_2 \times \cdots \times N_d}$ with the TT-cores $\mathcal{G}_1 \in \mathbb{R}^{1 \times N_1 \times 2}$, $\mathcal{G}_i \in \mathbb{R}^{2 \times N_i \times 2}$ (for $i = 2, 3, \dots, d-1$), and $\mathcal{G}_d \in \mathbb{R}^{2 \times N_d \times 1}$ such that*

$$\mathcal{G}_1[1, j, :] = \begin{pmatrix} 1 & \widehat{f}_1\left(v_j^{(1)}\right) \end{pmatrix}, \quad j = 1, 2, \dots, N_1, \tag{3.6}$$

$$\mathcal{G}_i[:, j, :] = \begin{pmatrix} 1 & \widehat{f}_i\left(v_j^{(i)}\right) \\ 0 & 1 \end{pmatrix}, \quad j = 1, 2, \dots, N_i, \quad i = 2, 3, \dots, d-1, \tag{3.7}$$

$$\mathcal{G}_d[:, j, 1] = \begin{pmatrix} \widehat{f}_d\left(v_j^{(d)}\right) + \widehat{f}_0 \\ 1 \end{pmatrix}, \quad j = 1, 2, \dots, N_d, \tag{3.8}$$

where $\widehat{f}_i$ ($i = 0, 1, \dots, d$) are the zero and first-order terms from (3.4) *and* (3.5)*. Then the TT-tensor $\mathcal{Y}$ has the same values for all possible elements as the corresponding first-order ANOVA decomposition.*

[4]For simplicity, we will further assume that the number of different values present in the dataset is the same as the size of the corresponding tensor modes $N_1, N_2, \dots, N_d$. This can be achieved, for example, by using the LHS distribution.

Proof. Let us directly calculate the matrix product of the TT-cores (3.6), (3.7), (3.8),

$$\mathcal{G}_1[1,j_1,:]\,\mathcal{G}_2[:,j_2,:] = \begin{pmatrix}1 & \widehat{\mathsf{f}}_1\left(v^{(1)}_{j_1}\right)\end{pmatrix}\begin{pmatrix}1 & \widehat{\mathsf{f}}_2\left(v^{(2)}_{j_2}\right)\\ 0 & 1\end{pmatrix} = \begin{pmatrix}1 & \widehat{\mathsf{f}}_1\left(v^{(1)}_{j_1}\right)+\widehat{\mathsf{f}}_2\left(v^{(2)}_{j_2}\right)\end{pmatrix},$$

then

$$\begin{aligned}\mathcal{G}_1[1,j_1,:]\,\mathcal{G}_2[:,j_2,:]\,\mathcal{G}_3[:,j_3,:] &= \begin{pmatrix}1 & \widehat{\mathsf{f}}_1\left(v^{(1)}_{j_1}\right)+\widehat{\mathsf{f}}_2\left(v^{(2)}_{j_2}\right)\end{pmatrix}\begin{pmatrix}1 & \widehat{\mathsf{f}}_3\left(v^{(3)}_{j_3}\right)\\ 0 & 1\end{pmatrix}\\ &= \begin{pmatrix}1 & \widehat{\mathsf{f}}_1\left(v^{(1)}_{j_1}\right)+\widehat{\mathsf{f}}_2\left(v^{(2)}_{j_2}\right)+\widehat{\mathsf{f}}_3\left(v^{(3)}_{j_3}\right)\end{pmatrix},\end{aligned}$$

and so on by induction, up to the last TT-core,

$$\begin{aligned}&\mathcal{G}_1[1,j_1,:]\,\mathcal{G}_2[:,j_2,:]\,\ldots\,\mathcal{G}_{d-1}[:,j_{d-1},:]\,\mathcal{G}_d[:,j_d,1]\\ &\quad= \begin{pmatrix}1 & \widehat{\mathsf{f}}_1\left(v^{(1)}_{j_1}\right)+\widehat{\mathsf{f}}_2\left(v^{(2)}_{j_2}\right)+\cdots+\widehat{\mathsf{f}}_{d-1}\left(v^{(d-1)}_{j_{d-1}}\right)\end{pmatrix}\begin{pmatrix}\widehat{\mathsf{f}}_d\left(v^{(d)}_{j_d}\right)+\widehat{\mathsf{f}}_0\\ 1\end{pmatrix}\\ &\quad= \widehat{\mathsf{f}}_0+\widehat{\mathsf{f}}_1\left(v^{(1)}_{j_1}\right)+\widehat{\mathsf{f}}_2\left(v^{(2)}_{j_2}\right)+\cdots+\widehat{\mathsf{f}}_d\left(v^{(d)}_{j_d}\right),\end{aligned}$$

that is, the $(j_1, j_2, \ldots, j_d)$th element of the TT-tensor (taking into account the formula (2.1)) coincides with the ANOVA representation (3.1). ∎

Thus, the given scheme for choosing the elements of the TT-cores allows us to obtain the ANOVA representation within the framework of the low-rank TT-decomposition, and we call this representation TT-ANOVA. Note that this result can be directly generalized to the case of arbitrary TT-rank if we take the corresponding additional elements of the TT-cores equal to zero.

Remark 3.2. The proved theorem allows us to construct a first-order ANOVA representation in the TT-format. As will be shown below in numerical experiments, it is a good initial approximation for a tensor completion problem, even in the case when the objective function is very badly represented in terms of the first-order ANOVA. At the same time, the use of a higher-order term turns out to be unjustified, since this significantly complicates the process of constructing the corresponding TT-tensor.

3.2. ALS algorithm for tensor completion. The idea of the TT-ALS tensor completion algorithm [17, 16] is to successively refine each of the TT-cores by solving an overdetermined system of linear equations with the right-hand side equal to the given tensor values from the train dataset $(X^{(train)}, Y^{(train)})$ using the least squares method. TT-ALS starts from some initial approximation, which is usually a random TT-tensor, and successively refines the TT-cores in passes from left to right (i.e., from the first TT-core to the last one) and back. One complete pass from left to right and back is called a sweep, and the total number of sweeps is a hyperparameter of the algorithm.

Suppose at the current step we need to update the values of the ith TT-core. Consider a point $\boldsymbol{x}$ from the train dataset in which the value is known and is equal to $y = \mathsf{f}(\boldsymbol{x})$. In the discrete setting, this point corresponds to some multi-index $\boldsymbol{j} = (j_1, j_2, \ldots, j_d)$ of the tensor. We denote by $\boldsymbol{g}_l$ and $\boldsymbol{g}_r$ the two vectors that are

Algorithm 3.1. TT-ALS algorithm.

```
Require: input multi-indices $J$ and related outputs $Y$ of the BB; initial
    approximation $\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_d$ ($\mathcal{G}_i \in \mathbb{R}^{R_{i-1} \times N_i \times R_i}$ for $i = 1, \ldots, d$)
Ensure: TT-cores of the tensor that approximate the given BB
 1: while stooping criterion is not met do
 2:   for $i$ from 1 to $d$ do
 3:     for $n_k$ from 1 to $N_k$ do
 4:       $l \leftarrow 1$
 5:       Allocate $A \in \mathbb{R}^{K \times (R_{i-1} R_i)}$, $\boldsymbol{b} \in \mathbb{R}^K$, where $K$ is the number of samples $J$
          for which the $i$th component equal to $n_k$
 6:       for $\boldsymbol{j}$ from $J$ such that its $i$th component is equal to $n_k$ do
 7:         $A[l, :] \leftarrow \boldsymbol{a}$, where vector $\boldsymbol{a}$ is defined in (3.10) and is based on the
            current TT-cores and the components of the vector $\boldsymbol{j}$
 8:         $\boldsymbol{b}[l] \leftarrow$ the value from $Y$ corresponds to input $\boldsymbol{j}$
 9:         $l \leftarrow l + 1$
10:       end for
11:       Solve $A\boldsymbol{g} = \boldsymbol{b}$ for $\boldsymbol{g}$ using least squares method
12:       Update $\mathcal{G}_i[:, n_k, :] \leftarrow \text{reshape}(\boldsymbol{g}, (R_{i-1}, R_i))$
13:     end for
14:   end for
15: end while
16: return $\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_d$
```

obtained after the multiplication of the TT-cores to the left and right of the ith TT-core correspondingly and by G the slice of the ith TT-core:

$$\begin{aligned} \boldsymbol{g}_l^T &= \mathcal{G}_1[1, j_1, :]\, \mathcal{G}_2[:, j_2, :] \cdots \mathcal{G}_{i-1}[:, j_{i-1}, :], & \boldsymbol{g}_l &\in \mathbb{R}^{R_{i-1}}, \\ \boldsymbol{g}_r &= \mathcal{G}_{i+1}[:, j_{i+1}, :]\, \mathcal{G}_{i+2}[:, j_{i+2}, :] \cdots \mathcal{G}_d[:, j_d, 1], & \boldsymbol{g}_r &\in \mathbb{R}^{R_i}, \\ G &= \mathcal{G}_i[:, j_i, :], \quad G \in \mathbb{R}^{R_{i-1} \times R_i}, \end{aligned}$$

where R_{i-1} and R_i are the corresponding TT-ranks. Given the explicit form of the TT-decomposition (2.1), we need to achieve, at least approximately, the equality

$$\boldsymbol{g}_l^T G \boldsymbol{g}_r = y \tag{3.9}$$

by selecting the elements of the matrix G. Note that we can rewrite the left-hand side of the equality (3.9) in a form of the scalar product,

$$\begin{aligned} &\sum_{r=1}^{R_i \cdot R_{i+1}} \boldsymbol{a}[r] \cdot (\text{vec } G)[r] = y, \\ &\boldsymbol{a}[r] = \boldsymbol{g}_l[\, \lfloor (r-1)/R_i \rfloor + 1\,] \cdot \boldsymbol{g}_r[\,(r-1) \mod R_i + 1\,], \end{aligned} \tag{3.10}$$

where vec $G \in \mathbb{R}^{R_{i-1} \cdot R_i}$ denotes vectorization of the matrix G, i.e.,

$$(\text{vec } G)[\,r_2 + (r_1 - 1) \cdot R_i\,] = G[\,r_1, r_2\,], \quad r_1 = 1, 2, \ldots, R_{i-1}, \; r_2 = 1, 2, \ldots, R_i,$$

operation "$\cdot \mod \cdot$" denotes taking the remainder of a division, and $\lfloor \cdot \rfloor$ denotes rounding down to the nearest integer.

Now we select all points from the training dataset with the value of the ith element of the multi-index equal to the selected value of j_i. Let there be K such points and $K \geq R_{i-1}R_i$. Thus we have K equations of the form (3.10) for the unknown G. We solve this overdetermined system using the method of least squares. We do so consistently for all values of $j_i = 1, 2, \ldots, N_i$ obtaining all slices of the TT-core $\mathcal{G}_i$. And then we repeat this procedure sequentially for all TT-cores $\mathcal{G}_i$ for $i = 1, 2, \ldots, d$, making several such passes from left to right and back. We summarize the described procedure in Algorithm 3.1. A stopping condition in the algorithm may be the reaching of a certain number of iterations (sweeps) or a threshold for changing TT-cores components.[5]

4. Numerical experiments.

To demonstrate the effectiveness of the proposed TT-ANOVA-ALS approach, we consider 12 model seven-dimensional analytical functions (benchmarks), which are presented in Table 1. Note that the list of functions includes the Piston function, which corresponds to the practical problem of modeling the time that a piston takes to complete one cycle within a cylinder; the description of related parameters and their ranges are shown in Table 2.

We also consider the more complicated problem of approximating the mean solution of the parameter-dependent PDE [3, 35, 20],

$$-\mathrm{div}\big(\mathsf{k}(\boldsymbol{x},\boldsymbol{p})\nabla \mathsf{u}(\boldsymbol{x},\boldsymbol{p})\big) = \mathsf{f}(\boldsymbol{x}), \quad \boldsymbol{x} = [x_1, x_2]^T \in \Omega = [0,1]^2, \quad \mathsf{u}\big|_{\partial\Omega} = \mathsf{u}_D,$$

where

$$\mathsf{f}(\boldsymbol{x}) \equiv 1, \quad \mathsf{u}_D \equiv 0, \quad \mathsf{k}(\boldsymbol{x},\boldsymbol{p}) = \begin{cases} p_\mu & \text{if } \boldsymbol{x} \in S_\mu,\ \mu = 1, 2, \ldots, m^2, \\ 1 & \text{otherwise,} \end{cases}$$

$\boldsymbol{p} = [p_1, p_2, \ldots, p_{m^2}]$, and $\{S_\mu\}_{\mu=1}^{m^2}$ is a set of m^2 disks of radius $\rho = \frac{1}{4m+2}$ (note that these disks form an $m \times m$ grid) with the centers located in the points (x_1^μ, x_2^μ),

$$\begin{aligned} x_1^\mu &= i \cdot q + (2i-1) \cdot \rho, \quad x_2^\mu = j \cdot q + (2j-1) \cdot \rho, \\ q &= \frac{1 - 2m\rho}{m+1}, \quad \mu = (i-1)m + j, \quad i, j = 1, 2, \ldots, m. \end{aligned}$$

As a scalar value of interest we consider the average temperature over domain Ω,

$$\mathsf{PDE\text{-}VOI}\,(\boldsymbol{p}) = \int_\Omega \mathsf{u}(\boldsymbol{x},\boldsymbol{p})\, d\boldsymbol{x}, \quad \boldsymbol{p} \in [0.01, 1]^{m^2}.$$

We select $m = 3$, hence we have nine independent parameters, and in discrete representation, the PDE-VOI is the nine-dimensional implicit tensor.

For the numerical solution of the PDE, we use the popular software product FEniCS [22] with efficient implementation of the finite element method. In Figure 3 we present the PDE-VOI and computation time for different numbers of vertices of the spatial two-dimensional mesh. As can be estimated from the plots, for sufficient stability and accuracy of the solution, at least $5 \cdot 10^4$ vertices are required (and we will use this

[5]It is necessary that the training samples contain all possible values of the indices, i.e., $j_i = 1, 2, \ldots, N_i$ for all $i = 1, \ldots, d$. Moreover, we need to have at least $R_{i-1}R_i$ of them for the least squares matrix to be a full-rank matrix. This can be achieved, for example, by using the LHS distribution with a sample of sufficient size. However, a more critical constraint is associated with possible overfitting, i.e., the number of samples should not be less than the number of parameters in the TT-decomposition, which equals to $d \cdot \langle NR^2 \rangle$.

Table 1
Benchmark functions for performance analysis of the proposed algorithm.

Function	Ref.	Bounds	Analytical formula
Ackley	[19]	$[-32.768,\ 32.768]$	$\mathsf{f}(\boldsymbol{x}) = -Ae^{-B\sqrt{\frac{1}{d}\sum_{i=1}^d x_i^2}} - e^{\frac{1}{d}\sum_{i=1}^d \cos{(Cx_i)}} + A + e^1$, where $A=20$, $B=0.2$ and $C=2\pi$
Alpine	[19]	$[-10,\ 10]$	$\mathsf{f}(\boldsymbol{x}) = \sum_{i=1}^d \lvert x_i \sin x_i + 0.1 x_i \rvert$
Dixon	[19]	$[-10,\ 10]$	$\mathsf{f}(\boldsymbol{x}) = (x_1-1)^2 + \sum_{i=2}^d i \cdot \left(2x_i^2 - x_{i-1}\right)^2$
Exponential	[19]	$[-1,\ 1]$	$\mathsf{f}(\boldsymbol{x}) = -e^{-\frac{1}{2}\sum_{i=1}^d x_i^2}$
Griewank	[19]	$[-600,\ 600]$	$\mathsf{f}(\boldsymbol{x}) = \sum_{i=1}^d \frac{x_i^2}{4000} - \prod_{i=1}^d \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$
Michalewicz	[37]	$[0,\ \pi]$	$\mathsf{f}(\boldsymbol{x}) = -\sum_{i=1}^d \sin{(x_i)} \sin^{2m}\left(\frac{ix_i^2}{\pi}\right)$
Piston	[39]	See Table 2	$\mathsf{f}(M,\ S,\ V_0,\ k,\ P_0,\ T_a,\ T_0) = 2\pi\sqrt{\frac{M}{k+S^2\frac{P_0 V_0}{T_0}\frac{T_a}{V^2}}}$,where $V=\frac{S}{2k}\left(\sqrt{A^2+4k\frac{P_0V_0}{T_0}T_a} - A\right)$ and $A = P_0 S + 19.62M - \frac{kV_0}{S}$
Qing	[19]	$[0,\ 500]$	$\mathsf{f}(\boldsymbol{x}) = \sum_{i=1}^d \left(x_i^2 - i\right)^2$
Rastrigin	[13]	$[-5.12,\ 5.12]$	$\mathsf{f}(\boldsymbol{x}) = A \cdot d + \sum_{i=1}^d \left(x_i^2 - A \cdot \cos{(2\pi \cdot x_i)}\right)$,where $A=10$
Rosenbrock	[19]	$[-2.048,\ 2.048]$	$\mathsf{f}(\boldsymbol{x}) = \sum_{i=1}^{d-1} \left(100 \cdot (x_{i+1} - x_i^2)^2 + (1-x_i)^2\right)$
Schaffer	[19]	$[-100,\ 100]$	$\mathsf{f}(\boldsymbol{x}) = \sum_{i=1}^{d-1} \left(0.5 + \frac{\sin^2\left(\sqrt{x_i^2+x_{i+1}^2}\right) - 0.5}{\left(1+0.001(x_i^2+x_{i+1}^2)\right)^2}\right)$
Schwefel	[13]	$[-500,\ 500]$	$\mathsf{f}(\boldsymbol{x}) = 418.9829 \cdot d - \sum_{i=1}^d x_i \cdot \sin{\left(\sqrt{\lvert x_i \rvert}\right)}$

TABLE 2
Description of Piston *function parameters. For each parameter, its description, range of values, and physical units are provided.*

Variable	Name	Range	Units
M	Piston weight	$[30, 60]$	kg
S	Piston surface area	$[0.005, 0.020]$	m^2
V_0	Initial gas volume	$[0.002, 0.010]$	m^3
k	Spring coefficient	$[1000, 5000]$	N/m
P_0	Atmospheric pressure	$[90000, 110000]$	N/m^2
T_a	Ambient temperature	$[290, 296]$	K
T_0	Filling gas temperature	$[340, 360]$	K

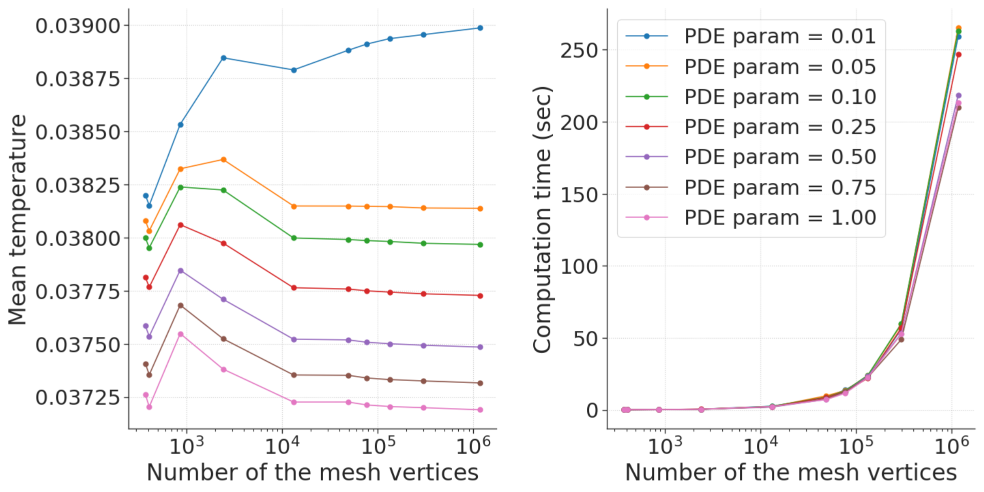

FIG. 3. *The* PDE-VOI *(on the left plot) and computation time (on the right plot) for different numbers of mesh vertices and different values of the first PDE parameter (p_1), while the values of the other parameters ($p_2, \ldots, p_9$) are fixed on the value* 0.5.

number in our computations), while the time of one calculation is more than 5 seconds. Thus, for this problem, it seems justified to search for a low-parameter approximation, which makes it possible to carry out calculations orders of magnitude faster.

For all benchmarks, including PDE-VOI, we consider the tensor that arises when the corresponding function is discretized on a uniform grid with 10 nodes in each dimension. We fixed the TT-rank value at $r = 5$ and the number of ALS sweeps at $s = 50$ (the dependence of the accuracy on the values of these parameters will be discussed later). The calculations with a random initial approximation were repeated 10 times and the averaged result was used. To train the model, we used $M = 10^4$ tensor elements, which are randomly generated from the LHS distribution. To check the accuracy of the obtained approximation $\mathcal{Y}$, we used a set of another $M^{(test)} = 10^4$ random samples (the function ind_to_poi transforms the tensor multi-index into a related spatial point)

$$\begin{aligned}
I^{(test)} &= \left\{\boldsymbol{i}^{(1)}, \boldsymbol{i}^{(2)}, \ldots, \boldsymbol{i}^{(M^{(test)})}\right\}, & \boldsymbol{i}^{(m)} &\in \mathbb{N}_+^d,\\
X^{(test)} &= \left\{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(M^{(test)})}\right\}, & \boldsymbol{x}^{(m)} &= \mathsf{ind\_to\_poi}(\boldsymbol{i}^{(m)}) \in \mathbb{R}^d,\\
Y^{(test)} &= \left\{y^{(1)}, y^{(2)}, \ldots, y^{(M^{(test)})}\right\}, & y^{(m)} &= \mathsf{f}(\boldsymbol{x}^{(m)}) \in \mathbb{R},
\end{aligned}$$

and calculated the error on it,

$$e^{(test)} = \sqrt{\frac{\sum_{m=1}^{M^{(test)}} \left(\mathcal{Y}[\boldsymbol{i}^{(m)}] - y^{(m)}\right)^2}{\sum_{m=1}^{M^{(test)}} \left(y^{(m)}\right)^2}}. \tag{4.1}$$

The results of numerical calculations are presented in Table 3. We report the error on the train and test data for each benchmark and the following approximation methods: the TT-ANOVA algorithm, i.e., the first-order ANOVA in the TT-format (column "ANOVA"); the TT-ALS algorithm with random initial approximation[6] (column "ALS"); and the TT-ANOVA-ALS algorithm, i.e., the TT-ALS algorithm, which uses the result of the TT-ANOVA as an initial approximation (column "ANOVA-ALS"). As can be seen from the results, in the latter case (TT-ANOVA-ALS), the accuracy improves by at least an order of magnitude concerning the TT-ALS method.[7]

To test the robustness of the algorithm, we repeated the same calculations on the noisy training data, i.e., we replaced each train value y by

TABLE 3
Relative error on the train and test data for all benchmarks. The reported values for the TT-ALS method (column "ALS") are averaged over 10 independent runs.

		ANOVA	ALS	ANOVA-ALS
Ackley	TRAIN	1.3E-02	5.1E-01	2.4E-02
	TEST	1.1E-02	7.4E-01	2.5E-02
Alpine	TRAIN	2.1E-02	8.0E-01	4.7E-10
	TEST	2.1E-02	1.8E+00	5.1E-10
Dixon	TRAIN	4.7E-02	6.5E-01	2.1E-04
	TEST	4.7E-02	2.9E+00	2.7E-04
Exponential	TRAIN	1.3E-01	6.6E-01	5.0E-10
	TEST	1.3E-01	2.1E+00	5.2E-10
Griewank	TRAIN	2.0E-02	7.1E-01	1.6E-04
	TEST	2.0E-02	2.0E+00	1.6E-04
Michalewicz	TRAIN	3.9E-02	6.6E-01	6.3E-10
	TEST	4.0E-02	5.9E+00	6.7E-10
Piston	TRAIN	9.3E-02	7.1E-01	3.6E-03
	TEST	9.4E-02	1.6E+00	4.7E-03
Qing	TRAIN	3.3E-02	7.4E-01	5.5E-08
	TEST	3.3E-02	3.4E+00	8.1E-08
Rastrigin	TRAIN	7.9E-03	8.4E-01	5.0E-10
	TEST	8.1E-03	1.3E+00	5.4E-10
Rosenbrock	TRAIN	2.0E-01	7.2E-01	2.2E-03
	TEST	2.0E-01	4.3E+00	3.6E-03
Schaffer	TRAIN	3.9E-02	6.8E-01	7.9E-03
	TEST	4.0E-02	9.9E-01	8.2E-03
Schwefel	TRAIN	1.3E-02	6.6E-01	8.7E-12
	TEST	1.3E-02	1.2E+00	1.1E-11
PDE-VOI	TRAIN	3.3E-03	8.0E-01	7.5E-05
	TEST	3.4E-03	1.4E+00	9.9E-05

[6]We generate all TT-cores $\mathcal{G}_k$ $(k=1,2,\ldots,d)$ from a random normal distribution.

[7]We note an unusual result for the Ackley function in Table 3, for which the initial approximation (TT-ANOVA) is more accurate than the final TT-ANOVA-ALS approximation. A similar situation is also observed for the Dixon and Exponential functions in Table 4, where the training dataset is noisy. This is due to our choice of a fixed TT-rank and training dataset size. A detailed analysis of the convergence and stability of the TT-ALS method can be found, for example, in the works [29, 18, 38].

TABLE 4
Relative error on the train and test data for all selected benchmarks. Multiplicative noise (4.2) *is applied to the training data. The reported values for TT-ALS method (column "ALS") are averaged over* 10 *independent runs.*

		ANOVA	ALS	ANOVA-ALS
Ackley	TRAIN	1.6E-02	6.8E-01	1.0E-02
	TEST	1.1E-02	9.8E-01	2.7E-03
Alpine	TRAIN	2.3E-02	8.1E-01	1.0E-02
	TEST	2.1E-02	1.9E+00	1.5E-03
Dixon	TRAIN	4.8E-02	7.3E-01	1.6E-02
	TEST	4.7E-02	3.3E+00	1.1E-01
Exponential	TRAIN	1.3E-01	7.4E-01	8.8E-01
	TEST	1.3E-01	2.4E+00	1.2E+00
Griewank	TRAIN	2.2E-02	7.1E-01	9.8E-03
	TEST	2.0E-02	1.8E+00	1.6E-03
Michalewicz	TRAIN	4.0E-02	6.6E-01	9.8E-03
	TEST	3.9E-02	6.7E+00	2.4E-03
Piston	TRAIN	9.4E-02	7.1E-01	1.0E-02
	TEST	9.4E-02	1.6E+00	6.8E-03
Qing	TRAIN	3.4E-02	7.4E-01	9.7E-03
	TEST	3.3E-02	3.4E+00	9.0E-03
Rastrigin	TRAIN	1.3E-02	6.7E-01	9.8E-03
	TEST	8.1E-03	1.1E+00	1.5E-03
Rosenbrock	TRAIN	2.0E-01	7.2E-01	8.6E-03
	TEST	2.0E-01	4.2E+00	9.9E-03
Schaffer	TRAIN	4.0E-02	8.5E-01	1.3E-02
	TEST	4.0E-02	1.2E+00	8.4E-03
Schwefel	TRAIN	1.7E-02	5.8E-01	1.1E-02
	TEST	1.3E-02	1.1E+00	7.6E-03

$$\hat{y} = \left(1 + 10^{-2} z\right) \cdot y, \tag{4.2}$$

where random variable z have standard normal distribution, i.e., $z \sim \mathcal{N}(0, 1)$. The corresponding results are reported in Table 4. The errors were computed, as above, according to the formula (4.1). As follows, even in the presence of noise, a high approximation accuracy is maintained when using the TT-ANOVA-ALS approach. Thus, the proposed TT-ANOVA-ALS approach significantly improves the accuracy of the TT-ALS method with virtually no increase in the computational complexity.[8] Also note that this approach is deterministic (on a fixed training set), which means that it is not prone to random failures of the TT-ALS when the initial approximation is chosen poorly.

The above results correspond to a fixed value of the TT-rank r and the number of iterations (sweeps) s. To check the correctness of the proposed method for other values of these parameters, we carried out a series of additional computations. For the parametric PDE (i.e., the PDE-VOI function), for the Piston function (as the most complex benchmark) and for the Dixon function (as a benchmark, which may be fairly well approximated by the basic TT-ALS method), we performed numerical calculations for $r = 2, 4, 6, 8, 10$ and $s = 1, 2, 5, 10, 50, 100$. The approximation errors on the test dataset (according to (4.1)) are presented in Figures 4, 5, and 6. We averaged the results for the TT-ALS method over 100 independent runs and present the mean

[8]Calculations were conducted on a regular laptop. For our model problem, the time for building the TT-ANOVA approximation turns out to be less than 10 milliseconds, and the average time for building the TT-ALS approximation is about 6 seconds.

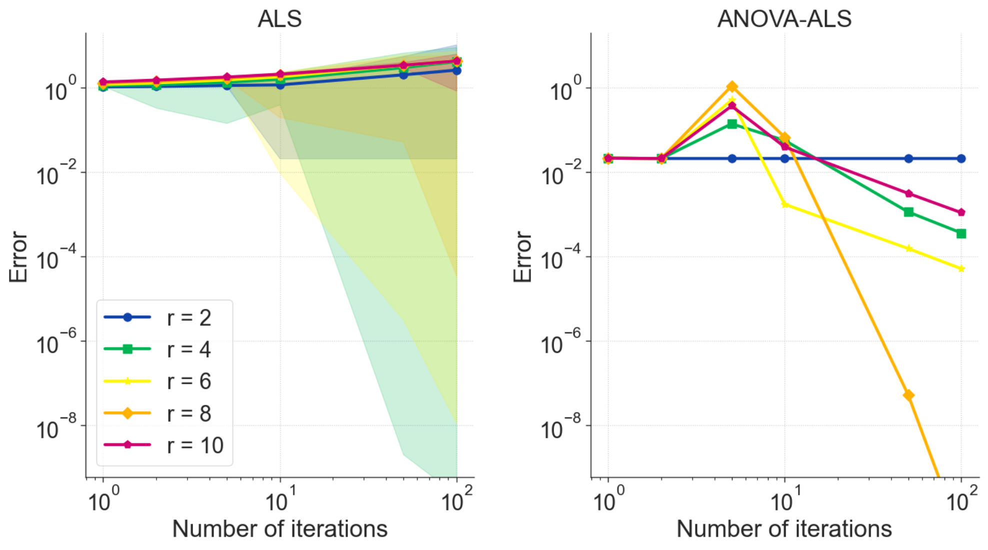

FIG. 4. *The approximation error of the* Dixon *function by the* TT-ALS *method (on the left plot) and by the* TT-ANOVA-ALS *method (on the right plot).*

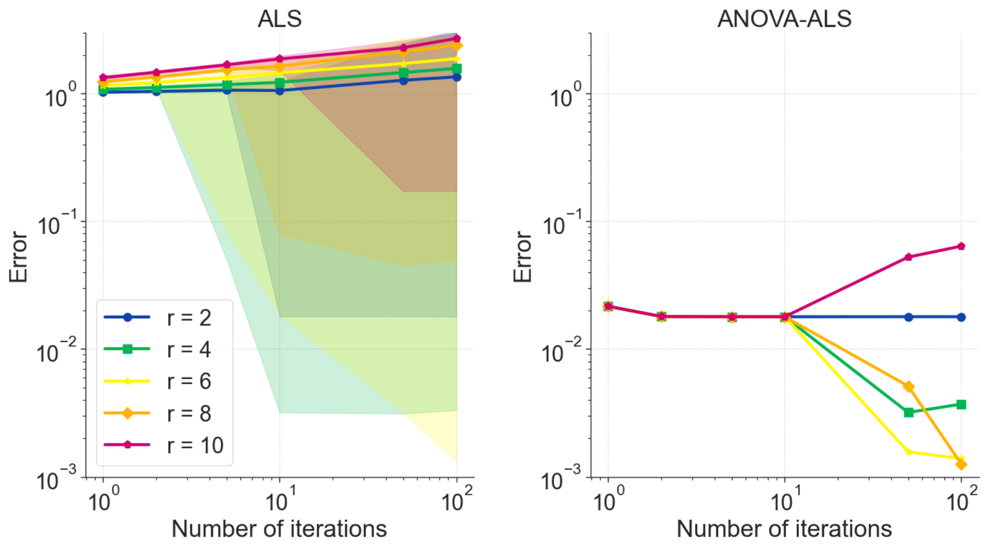

FIG. 5. *The approximation error of the* Piston *function by the* TT-ALS *method (on the left plot) and by the* TT-ANOVA-ALS *method (on the right plot).*

value of the error (solid lines on the plots) as well as the full range of error values (filled region). As can be seen, in all cases the result obtained using the TT-ANOVA-ALS method is much better than the average result when using the basic TT-ALS method with a random initial approximation. Also, in most cases (especially for the PDE-VOI), our method gives higher accuracy than the best result of 100 runs of the basic TT-ALS method (note that in this case our method turns out to be two orders of magnitude faster than the TT-ALS method). In Figure 7, for the same functions,

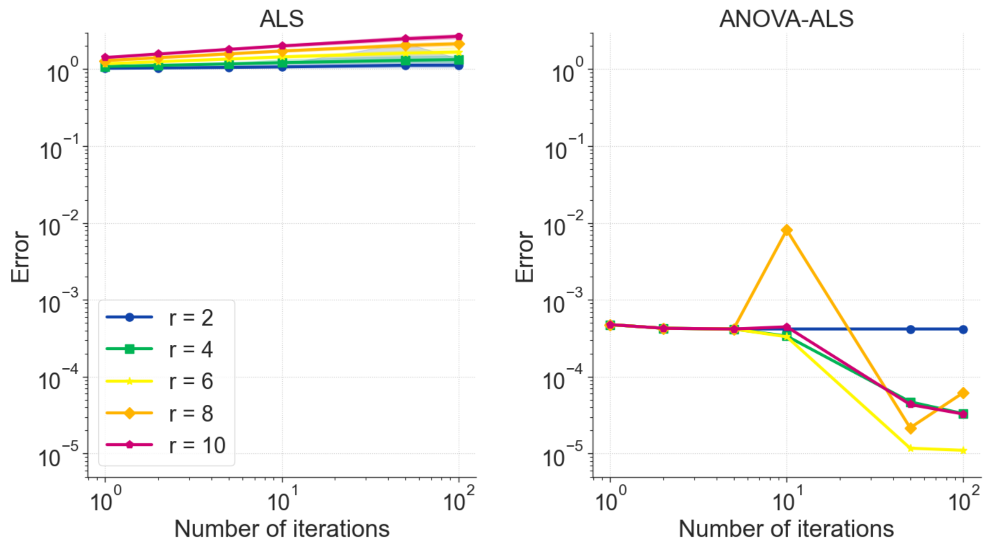

Fig. 6. *The approximation error of the* PDE-VOI *function by the* TT-ALS *method (on the left plot) and by the* TT-ANOVA-ALS *method (on the right plot).*

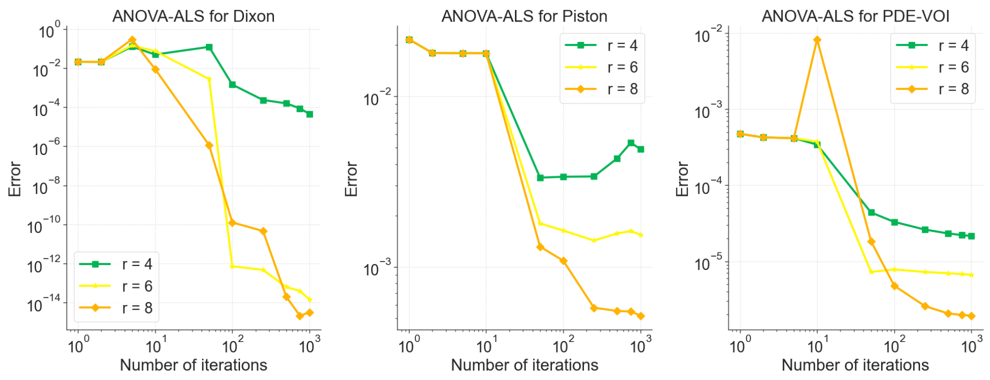

Fig. 7. *Dependence of the approximation error of the* Dixon*,* Piston*, and* PDE-VOI *functions on the number of iterations for different values of the TT-rank.*

we present the TT-ANOVA-ALS result for a higher number of iterations (up to 1000). As can be seen, for each TT-rank value (4, 6, and 8), the accuracy generally increases with the number of iterations. At the same time, the nature of the curve remains a bit irregular due to the dependence of the accuracy on the size of the train dataset and the actual TT-rank of the function being approximated.

5. Related work.

Today, TT-decomposition is a popular approach for compact approximation of multidimensional arrays and multivariable functions [10, 27, 31, 32], including applications in the field of data analysis, machine learning, solution of PDEs, etc. Various practical methods based on the TT-decomposition have been proposed to solve multidimensional [28, 8], parametric [14, 15], and multiscale [25, 9] PDEs, to optimize multivariable functions [34, 23], etc. We note that in the works [28, 8, 14, 15], special attention is paid to the choice of the initial approximation for constructing the TT-decomposition for the PDE solution, and as such an approximation, the solution obtained from the previous time step is used. At the same time, in the works [25, 9, 34, 23], a random initial approximation is used.

Until now only a few works have been directly devoted to the selection of the initial approximation in the TT-format. We note the recent work [20], where the authors apply Gaussian process regression to construct a preliminary approximation of the function, using a training dataset. After that, the initial approximation is generated by the TT-cross method, which is applied to the regression model. This approach is very promising, but it requires significant additional numerical computations to run TT-cross and it also has all the limitations of regression methods when applied to multidimensional problems. We also note that the authors conducted numerical experiments for only one benchmark, which does not allow us to evaluate the general robustness of the approach.

An interesting method was considered in [21] for constructing an initial approximation in the framework of the proposed new completion method. The authors perform a special interpolation based on the known tensor values, and then build an initial approximation using the classical TT-SVD algorithm. However, this approach was developed only for the problem of image and video processing, and it cannot be transferred to essentially multidimensional problems due to the need to build a full tensor for the TT-SVD algorithm. Also, we note that the ANOVA decomposition was already used in combination with the TT-approach; see, for example, [40, 5]. But in these works, effective methods for construction of the Sobol indices through the calculation of multidimensional integrals by TT-decomposition were considered, and this is a different problem than the one presented in our work.

6. Conclusions.

In this work, we proposed the TT-ANOVA representation as an initial approximation for the TT-ALS approach, which makes it possible to effectively approximate a functionally given black box. We have implemented the corresponding method TT-ANOVA-ALS in software and demonstrated its effectiveness for the list of model multidimensional problems, including the approximation of the solution to the parametric PDE. For all considered model problems we obtained an increase in accuracy by at least an order of magnitude with the same number of training samples from the black box and with virtually no increase in the computational complexity. The proposed approach is deterministic (on a fixed training set) and it is not prone to random failures of the TT-ALS when the initial approximation is chosen poorly. The TT-ANOVA-ALS method can be applied to a wide class of practically significant problems, including surrogate modeling and various machine learning applications.

REFERENCES

[1] G. Archer, A. Saltelli, and I. Sobol, *Sensitivity measures, ANOVA-like techniques and the use of bootstrap*, J. Stat. Comput. Simul., 58 (1997), pp. 99–120.

[2] I. Babuska, R. Tempone, and G. Zouraris, *Galerkin finite element approximations of stochastic elliptic partial differential equations*, SIAM J. Numer. Anal., 42 (2004), pp. 800–825.

[3] J. Ballani and L. Grasedyck, *Hierarchical tensor approximation of output quantities of parameter-dependent PDEs*, SIAM/ASA J. Uncertain. Quantif., 3 (2015), pp. 852–872.

[4] R. Ballester-Ripoll, *Tensor approximation of cooperative games and their semivalues*, Internat. J. Approx. Reason., 142 (2022), pp. 94–108.

[5] R. Ballester-Ripoll, E. Paredes, and R. Pajarola, *Sobol tensor trains for global sensitivity analysis*, Reliab. Eng. Syst. Safe., 183 (2019), pp. 311–322.

[6] R. N. Cardinal and M. R. Aitken, *ANOVA for the Behavioral Sciences Researcher*, Psychology Press, London, 2013.

[7] H. Chen, L. Deng, Z. Qu, L. Liang, T. Yan, Y. Xie, and G. Li, *Tensor train decomposition for solving large-scale linear equations*, Neurocomputing, 464 (2021), pp. 203–217.

[8] A. Chertkov and I. Oseledets, *Solution of the Fokker-Planck equation by cross approximation method in the tensor train format*, Front. Artificial Intelligence, 4 (2021), 99.

[9] A. Chertkov, I. Oseledets, and M. Rakhuba, *Robust Discretization in Quantized Tensor Train Format for Elliptic Problems in Two Dimensions*, preprint, https://arxiv.org/abs/1612.01166, 2016.

[10] A. Cichocki, N. Lee, I. Oseledets, A.-H. Phan, Q. Zhao, and D. Mandic, *Tensor networks for dimensionality reduction and large-scale optimization: Part 1. Low-rank tensor decompositions*, Found. Trends Mach. Learn., 9 (2016), pp. 249–429.

[11] A. Cichocki, A. Phan, Q. Zhao, N. Lee, I. Oseledets, M. Sugiyama, and D. Mandic, *Tensor networks for dimensionality reduction and large-scale optimization: Part 2. Applications and future perspectives*, Found. Trends Mach. Learn., 9 (2017), pp. 431–673.

[12] A. Cliffe, M. Giles, R. Scheichl, and A. Teckentrup, *Multilevel Monte Carlo methods and applications to elliptic PDEs with random coefficients*, Comput. Vis. Sci., 14 (2011), pp. 3–15.

[13] J. Dieterich and B. Hartke, *Empirical review of standard benchmark functions using evolutionary global optimization*, Appl. Math., 3 (2012), pp. 1552–1564.

[14] S. Dolgov and R. Scheichl, *A hybrid alternating least squares–TT-cross algorithm for parametric PDEs*, SIAM/ASA J. Uncertain. Quantif., 7 (2019), pp. 260–291.

[15] K. Glau, D. Kressner, and F. Statti, *Low-rank tensor approximation for Chebyshev interpolation in parametric option pricing*, SIAM J. Financial Math., 11 (2020), pp. 897–927.

[16] L. Grasedyck, M. Kluge, and S. Krämer, *Alternating Directions Fitting (ADF) of Hierarchical Low Rank Tensors*, preprint, DFG-Schwerpunktprogramm 1324, 2013.

[17] S. Holtz, T. Rohwedder, and R. Schneider, *The alternating linear scheme for tensor optimization in the tensor train format*, SIAM J. Sci. Comput., 34 (2012), pp. A683–A713.

[18] S. Hu and K. Ye, *Linear convergence of an alternating polar decomposition method for low rank orthogonal tensor approximations*, Math. Program., 199 (2023), pp. 1305–1364.

[19] M. Jamil and X.-S. Yang, *A literature survey of benchmark functions for global optimization problems*, J. Math. Model. Numer. Optim., 4 (2013), pp. 150–194.

[20] Y. Kapushev, I. Oseledets, and E. Burnaev, *Tensor completion via Gaussian process–based initialization*, SIAM J. Sci. Comput., 42 (2020), pp. A3812–A3824.

[21] C.-Y. Ko, K. Batselier, L. Daniel, W. Yu, and N. Wong, *Fast and accurate tensor completion with total variation regularized tensor trains*, IEEE Trans. Image Process., 29 (2020), pp. 6918–6931.

[22] A. Logg, K. Mardal, and G. Wells, *Automated Solution of Differential Equations by the Finite Element Method: The FEniCS Book*, Lect. Notes Comput. Sci. Eng. 84, Springer, New York, 2012.

[23] A. Nikitin, A. Chertkov, R. Ballester-Ripoll, I. Oseledets, and E. Frolov, *Are Quantum Computers Practical Yet? A Case for Feature Selection in Recommender Systems using Tensor Networks*, preprint, https://arxiv.org/abs/2205.04490, 2022.

[24] I. V. Oseledets, *Tensor-train decomposition*, SIAM J. Sci. Comput., 33 (2011), pp. 2295–2317.

[25] I. Oseledets, M. Rakhuba, and A. Chertkov, *Black-box solver for multiscale modelling using the QTT format*, in Proceedings of ECCOMAS, 2016.

[26] I. V. Oseledets and E. E. Tyrtyshnikov, *Breaking the curse of dimensionality, or how to use SVD in many dimensions*, SIAM J. Sci. Comput., 31 (2009), pp. 3744–3759.

[27] A. Phan, A. Cichocki, A. Uschmajew, P. Tichavskỳ, G. Luta, and D. Mandic, *Tensor networks for latent variable analysis: Novel algorithms for tensor train approximation*, IEEE Trans. Neural Netw. Learn. Syst., 31 (2020), pp. 4622–4636.

[28] L. Richter, L. Sallandt, and N. Nüsken, *Solving high-dimensional parabolic PDEs using the tensor train format*, in Proceedings of the International Conference on Machine Learning, PMLR, 2021, pp. 8998–9009.

[29] T. Rohwedder and A. Uschmajew, *On local convergence of alternating schemes for optimization of convex problems in the tensor train format*, SIAM J. Numer. Anal., 51 (2013), pp. 1134–1162.

[30] G. Ryzhakov and I. Oseledets, *Constructive TT-representation of the tensors given as index interaction functions with applications*, in Proceedings of Eleventh International Conference on Learning Representations (ICLR-2023), 2023.

[31] F. Sedighin and A. Cichocki, *Image completion in embedded space using multistage tensor ring decomposition*, Front. Artificial Intelligence, 4 (2021).

[32] F. Sedighin, A. Cichocki, and A.-H. Phan, *Adaptive rank selection for tensor ring decomposition*, IEEE J. Sel. Top. Signal Process., 15 (2021), pp. 454–463.

[33] I. Sobol, *Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates*, Math. Comput. Simul., 55 (2001), pp. 271–280.

[34] 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*, in Proceedings of Thirty-sixth Conference on Neural Information Processing Systems (NeurIPS-2022), 2022.

[35] C. Tobler, *Low-Rank Tensor Methods for Linear Systems and Eigenvalue Problems*, Ph.D. thesis, ETH Zurich, 2012.

[36] L. N. Trefethen, *Approximation Theory and Approximation Practice*, Extended Edition, SIAM, Philadelphia, 2019.

[37] C. Vanaret, J.-B. Gotteland, N. Durand, and J.-M. Alliot, *Certified Global Minima for a Benchmark of Difficult Optimization Problems*, preprint, https://arxiv.org/abs/2003.09867, 2020.

[38] Y. Yang, *On global convergence of alternating least squares for tensor approximation*, Comput. Optim. Appl., 84 (2023), pp. 509–529.

[39] V. Zankin, G. Ryzhakov, and I. Oseledets, *Gradient Descent-Based D-Optimal Design for the Least-Squares Polynomial Approximation*, preprint, https://arxiv.org/abs/1806.06631, 2018.

[40] Z. Zhang, X. Yang, I. Oseledets, G. Karniadakis, and L. Daniel, *Enabling high-dimensional hierarchical uncertainty quantification by ANOVA and tensor-train decomposition*, IEEE Trans. Comput. Aid. Des. Integr. Circuits Syst., 34 (2014), pp. 63–76.