IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 18, 2021, accepted April 16, 2021, date of publication April 22, 2021, date of current version April 30, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3074930

Fast Circulant Tensor Power Method for High-Order Principal Component Analysis

TAEHYEON KIM, (Graduate Student Member, IEEE), AND YOONSIK CHOE, (Senior Member, IEEE)
Department of Electrical and Electronic Engineering, Yonsei University, Seoul 120-749, South Korea

Corresponding author: Yoonsik Choe (yschoe@yonsei.ac.kr)

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea Government (Ministry of Science and ICT) under Grant 1711108458.

ABSTRACT To understand high-order intrinsic key patterns in high-dimensional data, tensor decomposition is a more versatile tool for data analysis than standard flat-view matrix models. Several existing tensor models aim to achieve rapid computation of high-order principal components based on the tensor power method. However, since a tensor power method does not enforce orthogonality in subsequently calculated decomposition components, it causes far more challenges on principal component analysis of high-order tensors. To address this problem, several tensor power method variant algorithms incorporating sparsity into decomposition factors have been proposed. However, because these variant algorithms require additional procedures based on data-driven hyper-parameter optimization algorithms, a trade-off between computational cost and convergence exists. In this paper, a novel tensor power method called the fast circulant tensor power method is proposed. The proposed algorithm combines tensor-train decomposition and the power method. Tensor-train decomposition is a high-order tensor decomposition method based on auxiliary unfolding matrix decomposition. Thus, the power method can be embedded into our methodology without any additional processes. Notably, a simple combination of these two methods may cause a local optima problem because the power method only guarantees convergence on each unfolding matrix in tensor-train decomposition. To solve this problem, the circulant updating method is proposed, which globally optimizes all factor vectors by reordering some steps of the factor vector updates. It is experimentally demonstrated that, compared to state-of-the-art tensor power method variant methodologies, the proposed algorithm achieves the lowest computational complexity and quantitatively good performance in various applications including large-scale color image decomposition and convolutional neural network compression.

INDEX TERMS High-order principal component analysis, tensor decomposition, tensor power method, tensor-train decomposition.

I. INTRODUCTION

The recent development of multi-sensor techniques and the emergence of deep neural networks based on large scale high-order datasets have highlighted the importance of tensor decomposition. Because tensor decomposition benefits from the power of multi-linear algebra, data analysis based on tensor decomposition extracts more general dominant components than flat-view matrix decomposition. Therefore, high-order principal component analysis based on tensor decomposition is useful and widespread in numerous applications including signal/image processing, computer vision, and many other application areas. Further details on tensor decomposition and its recent emerging applications are presented in various books [17]–[22] and tutorial papers [6], [12], [23]–[28].

The most common tensor model is the canonical polyadic decomposition (CPD), which was independently introduced under the names of canonical decomposition (CANDECOMP) [2] and parallel factor model (PARAFAC) [3]. CPD aims to represent high-order tensors as a linear combination of R rank-1 tensors, where R is the rank of CPD. Because the calculation of CPD is intrinsically multi-linear, the solution can be obtained based on a sequence of linear sub-problems similar to the alternating least squares (ALS) optimization algorithm [12], [24].

The associate editor coordinating the review of this manuscript and approving it for publication was Hossein Rahmani.

Therefore, ALS-based CPD optimizes single factor matrices, while keeping other factor matrices, fixed [3], [5]. Although ALS achieves satisfactory performance, it inevitably inherits the problem of alternating optimization techniques and is not guaranteed to converge to a stationary solution. To address these issues, Chen *et al.* developed a rectifying algorithm on ALS [5]. However, this resulted increasing the computational cost by N times per iteration, where N is the number of dimensions of the input tensor. Despite the adequate performance, the feasibility of achieving fast and efficient computation using ALS-based CPD is significantly limited [5], [6], [15], [16].

To reduce the computational cost of ALS-based CPD, numerous methodologies have been developed including line search extrapolation methods [34]–[38], compression [39], and fast damped Gauss-Newton methods [40]–[43]. Although these methods can reduce computational complexity, they are based on the product of unfolding matrices and Khatri-Rao products of all factor matrices except one. This procedure becomes computationally intensive in CPD [32]. To reduce the computation cost in ALS via tensor decomposition, *Zniyed et al.* developed trains of coupled third order CPD algorithms capable of dimensionality reduction on an input tensor before performing an ALS-based CPD procedure [31]. However, such an algorithm is based on the singular value decomposition (SVD) of unfolding matrices, which can lead to significant computational complexity when decomposing high-dimensional large-scale input tensors. Additionally, there are several works pointing out the instability problem of ALS-based CPD in practical high-order principal component analysis [46]–[48].

To reduce the computational costs of low-rank tensor approximation without ALS and SVD, a tensor power method (TPM) is introduced [4]. It can be considered as a generalization of the power method (PM) in matrix decomposition [7], [8]. Similar to the PM, the TPM approximates the Nth-order rank-1 tensor as a singular value and N factor vectors, where N is the number of dimensions of the input tensor. The multi-rank CPD problem can be solved by sequentially applying the TPM to the residuals remaining after subtracting the previously computed factors. Unlike the PM, theoretical understanding of the TPM is limited. The TPM can be viewed as a gradient descent step, corresponding to the problem of finding the optimal rank-1 tensor of an input tensor [10]. This optimization problem is non-convex. Moreover, the number of stationary points of this optimization problem is exponential in the dimensions of the input tensor. This makes the analysis of the TPM far more challenging [9]–[11]. In particular, the TPM suffers from the curse-of-dimensionality problem.

Despite the above challenges, several algorithms have been developed to enhance the robustness of TPM by incorporating sparsity into the original TPM. Allen *et al.* [1] proposed the sparse tensor power method (STPM), which directly added an L1-norm penalty to the decomposition factor vectors in the rank-1 tensor approximation problem and solved it via alternative soft threshold updating. Sun *et al.* [13] developed the tensor truncated power method (TTPM), which incorporates the amount of truncation selection into the estimation of decomposition factors to encourage sparsity of each decomposition component. Both STPM and TTPM can enhance the performance of TPM. However, these algorithms require enormous computational cost to adaptively control the amount of sparsity; this control is based on data-driven hyperparameter optimization using the Bayesian information criterion (BIC) [14]. This hyperparameter optimization is performed by solving the combinatorial optimization problem, which identifies the best combination of hyperparameters by searching all possible combinations. Therefore, the computational costs are inevitably high. These computational complexities are proportional to the number of dimensions and entries of an input tensor. The enhanced versions of the TPM experience a trade-off between efficient processing and precise approximation.

In this paper, we propose a novel tensor power method, called the fast circulant tensor power method (FCTPM). The FCTPM is based on a combination of tensor-train decomposition (TTD) [29] and the PM [7], [8]. TTD is a novel type of tensor decomposition and represents high-order tensors as consecutively connected 3rd-order core tensors [29]. By utilizing TTD, the proposed algorithm can stably decompose high-order tensors via low-rank matrix decomposition of auxiliary unfolding matrices [29]. In addition, the circulant updating method is proposed to obtain the global convergence of decomposition factors. The FCTPM approximates the most dominant rank-1 tensor of input tensor via rank-1 TTD. Each unfolding matrix is also decomposed by rank-1 matrix decomposition. This property makes each unfolding matrix ignore the connectivity between other unfolding matrices. Therefore, the circulant updating method is developed to consider the connectivity between unfolding matrices by merely changing the order of updating steps of the right singular vector after the computations for all the left singular vectors. The FCTPM utilizes PM in auxiliary unfolding matrix decomposition, which is better understood than the TPM in terms of dynamics and convergence [7], [8]. Therefore, the PM guarantees the convergence of decomposition factors and the circulant updating method enhances the connectivity between decomposition factors. In conclusion, the FCTPM aims to decompose input high order tensors via the linear combination of R rank-1 tensors, similar to the procedure followed by CPD, rank-1 TTD, and other rank-1 tensor decomposition algorithms. However, the primary purpose of the proposed algorithm is to reduce the computational costs associated with the decomposition process. As such, the proposed algorithm does not involve the use of ALS or SVD, which are usually the basic procedures used in conventional tensor decomposition algorithms and entail enormous computational costs. In this context, the TPM and its variants are our desired methods because they have the same purpose as our algorithm.

In summary, the main contributions of this study are as follows:

- This work provides a novel fast rank-1 tensor approximation based on a suitable combination of rank-1 TTD and the PM via the proposed circulant updating algorithm. Thus, the proposed algorithm can inherit the advantages of both TTD and the PM simultaneously.
- The proposed methodology is based on TTD. Thus, this method can stably decompose high-order tensors via low-rank decomposition on auxiliary unfolding matrices of the high-order input tensor. This property helps the FCTPM avoid the curse-of-dimensionality problem.
- Because the proposed algorithm utilizes the PM, a theoretically well-known matrix decomposition algorithm, the convergence of each unfolding matrix decomposition is guaranteed. The proposed circulant updating method enhances the connectivity between each unfolding matrix decomposition factor from the PM
- Unlike TPM variant algorithms, the proposed tensor model does not require the data-driven hyperparameter optimization procedures that increase the computational complexity. This property emphasizes the computational efficiency of the proposed algorithm.
- To evaluate the generalization peculiarity of the proposed algorithm, we report experimental results from various applications including numerical simulations, large-scale color image reconstruction, and convolutional neural network (CNN) compression.

The remainder of the paper organized as follows. Section II briefly introduces the notations of tensor algebraic operations and two representative tensor models, namely CPD and TTD. Section III explains existing tensor power method algorithms, namely TPM, STPM, and TTPM. In Section IV, the proposed algorithm is explained with theoretical analysis. The obtained experimental results are presented and analyzed in Section V. Finally, this work is summarized in Section VI.

II. PRELIMINARIES

This section gives simple descriptions of the notations and definitions in tensor algebra and brief explanations of two representative tensor decomposition algorithms, namely CPD and TTD. Specifically, the existing TPM and its variants are based on rank-1 CPD. The proposed tensor model, FCTPM, follows the same approach to decomposition as rank-1 TTD.

A. NOTATION AND DEFINITIONS

In this paper, the mathematical notation follows the one used in [12]. The order of a tensor is the number of dimensions, also called ways or modes. Vectors (first-order tensor) are denoted by lowercase boldface such as $\mathbf{a}$; matrices (second-order tensor) are denoted by uppercase boldface such as $\mathbf{A}$; and tensors are denoted by calligraphic uppercase boldface such as $\mathcal{A}$. The element $(i_1, \ldots, i_N)$ of an Nth-order tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ is denoted by $\mathcal{A}_{i_1,\ldots,i_N}$. The *transpose* of matrix $\mathbf{A} \in \mathbb{R}^{I_1 \times I_2}$ is denoted by $\mathbf{A}^T$, which has a size of $I_1 \times I_2$. *Mode-n unfolding* is the process that transforms the N-way tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ into the second-order tensor $\mathcal{A}_{(n)} \in \mathbb{R}^{I_n \times I_1 \ldots I_{n-1} I_{n+1} \ldots I_N}$ by reordering the elements of the input tensor $\mathcal{A}$. The *n-mode product* of a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ with a 2-mode tensor $\mathbf{B} \in \mathbb{R}^{J \times I_n}$ is represented by $\mathcal{A} \times_n \mathbf{B}$ and is of size $I_1 \times \ldots \times I_{n-1} \times J \times I_{n+1} \times \ldots \times I_N$. The *rank-1 tensor* is the Nth-order tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ which is computed from the outer product of N vectors, i.e., $\mathcal{A} = \mathbf{a}^{(1)} \circ \ldots \circ \mathbf{a}^{(N)}$ where $\circ$ is the vector outer product operation and $\mathbf{a}^{(n)}$ is an nth factor vector of size I_n. The *diagonal tensor* is the tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ if $\mathcal{A}_{i_1,\ldots,i_N}$ is not zero only if $i_1 = \ldots = i_N$. The *norm* of a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ is denoted by $||\mathcal{A}||$, which is the square root of the sum of the squares of all elements of tensor $\mathcal{A}$, i.e.,

$$||\mathcal{X}|| = \sqrt{\sum_{i_1}^{I_1} \ldots \sum_{i_N}^{I_N} \mathcal{A}^2_{i_1,\ldots,i_N}}.$$

B. CANONICAL POLYADIC DECOMPOSITION (CPD)

The CPD represents an Nth-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ as the linear combination of a finite number of rank-1 tensors [12]. It is given by

$$\mathcal{X} = \sum_{r=1}^{R} \mathcal{D}_{r,r,r} \mathbf{U}^{(1)}_{:,r} \circ \ldots \circ \mathbf{U}^{(N)}_{:,r}, \tag{1}$$

or coequally,

$$\mathcal{X} = \mathcal{D} \times_1 \mathbf{U}^{(1)} \ldots \times_N \mathbf{U}^{(N)}, \tag{2}$$

where $\mathcal{D}_{r,r,r}$ is the rth non-zero diagonal element of the Nth-order diagonal core tensor $\mathcal{D} \in \mathbb{R}^{R \times \ldots \times R}$. The $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times R}$ denotes an nth factor matrix composed of a concatenation of R factor vectors $\mathbf{u}^{(\mathbf{n})}$, which have a size of I_n. In (1), the N-dimensional tensor $\mathcal{X}$ is represented by a linear combination of R rank-1 tensors. Thus, R denotes CPD-rank. If R is fixed as 1, it is referred to as rank-1 CPD; it computes N factor vectors (singular vectors) and one singular value, where N is the number of dimensions of the input tensor.

The ALS method is a representative CPD computation algorithm. The cost function of rank-R CPD can be expressed as follows:

$$\min_{\mathcal{D},\mathbf{U}^{(1)},\ldots,\mathbf{U}^{(N)}} ||\mathcal{X} - \mathcal{D} \times_1 \mathbf{U}^{(1)} \ldots \times_N \mathbf{U}^{(N)}||^2. \tag{3}$$

The ALS approach optimizes the target factor matrix while keeping the remaining factor matrices fixed. For example, when ALS solves $\mathbf{U}^{(1)}$, the other factor matrices $\mathbf{U}^{(2)}, \ldots, \mathbf{U}^{(N)}$ are fixed. This procedure in ALS performed multiple times until the prespecified convergence criterion is satisfied such as the maximum number of iterations, minimum reconstruction error. Although the ALS methodology is simple to understand and implement, it can take numerous iterations to converge [5], [6], [15], [16]. Therefore, the TPM-based algorithms are developed to efficiently compute the rank-1 tensor. The TPM and its variant algorithms are presented and explained in Section III.

C. TENSOR-TRAIN DECOMPOSITION (TTD)

TTD is a numerically reliable tensor decomposition approach to tackle the curse-of-dimensionality problem. The TTD represents an Nth-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ as N consecutive connected 3rd-order core tensors $\mathcal{G}$ [29]. TTD of an N-dimensional tensor $\mathcal{X}$ can be represented as follows:

$$\mathcal{X} = \sum_{r_1=1}^{R_1} \ldots \sum_{r_{N-1}=1}^{R_{N-1}} \mathcal{G}^{(1)}_{r_0,:,r_1} \circ \mathcal{G}^{(2)}_{r_1,:,r_2} \circ \ldots \circ \mathcal{G}^{(N)}_{r_{N-1},:,r_N}, \quad (4)$$

or correspondingly,

$$\mathcal{X} = \mathcal{G}^{(1)} \times_1 \mathcal{G}^{(2)} \ldots \times_1 \mathcal{G}^{(N)}, \quad (5)$$

where $\mathcal{G}^{(n)} \in \mathbb{R}^{R_{n-1} \times I_n \times R_n}$ is the nth core tensor and $\mathcal{G}^{(n)}_{r_{n-1},:,r_n} \in \mathbb{R}^{I_n}$ denotes a fiber vector of the nth core tensor.

In TTD, the entries of tensor $\mathcal{X}_{i_1,i_2,\ldots,i_N}$ are computed by the product of slice matrices of each core tensor, so the boundary condition $R_0 = R_D = 1$ must be imposed.

$$\mathcal{X}_{i_1,i_2,\ldots,i_N} = \mathcal{G}^{(1)}_{:,i_1,:} \mathcal{G}^{(2)}_{:,i_2,:} \ldots \mathcal{G}^{(N)}_{:,i_N,:}, \quad (6)$$

where $\mathcal{G}^{(n)}_{:,i_n,:} \in \mathbb{R}^{R_{n-1} \times R_n}$ is the i_nth slice matrix of the nth core tensor. Therefore the ranks of tensor $\mathcal{X}$ in TTD are defined as $[1, R_1, \ldots, R_{N-1}, 1]$, which is called TT-rank. Similar to the rank-1 CPD, if all elements in the rank of the TTD are set to 1, it is called rank-1 TTD. In other words, all the 3rd-order core tensors in rank-1 TTD can be considered as factor vectors in rank-1 CPD.

The core tensor $\mathcal{G}^{(n)}$ computation in TTD is based on the low-rank approximation of auxiliary matrices of the N-dimensional input tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$, which is as follow:

$$\begin{aligned}
\mathbf{U}^{(1)}\mathbf{V}^{(1)T} &= svd(\mathcal{X}_{(1)}, R_1), \\
\mathcal{G}^{(1)} &= reshape(\mathbf{U}^{(1)}, [1, I_1, R_1]), \\
\mathbf{X}^{(2)} &= reshape(\mathbf{V}^{(1)T}, [R_1\, I_2, I_3 \ldots I_N]) \\
\mathbf{U}^{(2)}\mathbf{V}^{(2)T} &= svd(\mathbf{X}^{(2)}, R_2), \\
\mathcal{G}^{(2)} &= reshape(\mathbf{U}^{(2)}, [R_1, I_2, R_2]), \\
&\vdots \\
\mathcal{G}^{(N)} &= reshape(\mathbf{V}^{(N-1)T}, [R_{N-1}, I_N, 1]),
\end{aligned} \quad (7)$$

where $svd(\mathbf{A}, R_n)$ denotes the rank-R_n singular value decomposition (SVD) on the matrix $\mathbf{A}$; $\mathbf{U}^{(n)}$ is a left singular matrix, $\mathbf{V}^{(n)}$ is a right singular matrix, and R_n is the rank of SVD. $reshape(\mathbf{A}, [I_1, I_2, I_3])$ denotes reshaping of the matrix $\mathbf{A}$ to a size of $I_1 \times I_2 \times I_3$.

Equation (7) shows that each core tensor in TTD is calculated via low-rank matrix decomposition. This characteristic of TTD makes it stable and lets it avoid the curse-of-dimensionality problem. Therefore, the proposed tensor model adopts TTD to inherit these advantages. The details of TTD in the proposed algorithm are described in Section IV.

Algorithm 1 Basic Pseudo-Code for Tensor Power Method (TPM)

Require: A Nth-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$, number of iterations K, and rank R.
Output: The Nth-order diagonal tensor $\mathcal{D} \in \mathbb{R}^{R \times R \times \ldots \times R}$, and N factor matrices $\mathbf{U}^{(1)} \in \mathbb{R}^{I_1 \times R}$, $\mathbf{U}^{(2)} \in \mathbb{R}^{I_2 \times R}$, $\ldots$, $\mathbf{U}^{(N)} \in \mathbb{R}^{I_N \times R}$.
while $r = 1$ **to** R **do**
  Initialize N random unit vectors $\mathbf{u}^{(1)} \in \mathbb{R}^{I_1}$, $\mathbf{u}^{(2)} \in \mathbb{R}^{I_2}$, $\ldots$, $\mathbf{u}^{(N)} \in \mathbb{R}^{I_N}$.
  while $k = 1$ **to** K **do**
    while $n = 1$ **to** N **do**
      $\mathbf{u}^{(n)} \leftarrow \mathbf{u}^{(n)}$ in Eq. (10)
    end while
  end while
  $\mathcal{D}_{r,r,r} \leftarrow \mathcal{X} \times_1 \mathbf{u}^{(1)T} \times_2 \mathbf{u}^{(2)T} \ldots \times_N \mathbf{u}^{(N)T}$.
  $\mathbf{U}^{(1)}_{:,r} \leftarrow \mathbf{u}^{(1)}$, $\mathbf{U}^{(2)}_{:,r} \leftarrow \mathbf{u}^{(2)}$, $\ldots$, $\mathbf{U}^{(N)}_{:,r} \leftarrow \mathbf{u}^{(N)}$.
  $\mathcal{X} \leftarrow \mathcal{X} - \mathcal{D}_{r,r,r}\mathbf{u}^{(1)} \circ \mathbf{u}^{(2)} \circ \ldots \circ \mathbf{u}^{(N)}$.
end while

III. RELATED WORKS

This section provides explanations of the TPM, STPM, and TTPM, which are developed to approximate the best rank-1 tensor of an input tensor. Subsequently, the problem is defined. Both STPM and TTPM aim to enforce sparsity in subsequently computed components in TPM. All these related methods are summarized in Algorithms 1, 2, and 3, respectively. For clarity, these algorithms are described for rank-R CPD on a Nth-order tensor.

A. TENSOR POWER METHOD (TPM)

The rank-1 CPD optimization problem on the Nth-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ can be defined as follows:

$$\begin{aligned}
&\min_{d, \mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(N)}} ||\mathcal{X} - d\mathbf{u}^{(1)} \circ \ldots \circ \mathbf{u}^{(N)}||^2 \\
&\text{subject to } d > 0 \text{ and } \mathbf{u}^{(n)T}\mathbf{u}^{(n)} = 1 \text{ for } n = 1, 2, \ldots, N,
\end{aligned} \quad (8)$$

where $\mathbf{u}^{(n)} \in \mathbb{R}^{I_n}$ is the nth factor vector. Because the optimal solution of (8) must satisfy $d = \mathcal{X} \times_1 \mathbf{u}^{(1)T} \ldots \times_N \mathbf{u}^{(N)T}$ [12], the above rank-1 CPD optimization problem can be rewritten as follows:

$$\begin{aligned}
&\max_{\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(N)}} ||\mathcal{X} \times_1 \mathbf{u}^{(1)T} \ldots \times_N \mathbf{u}^{(N)T}||^2 \\
&\text{subject to } \mathbf{u}^{(n)T}\mathbf{u}^{(n)} = 1 \text{ for } n = 1, 2, \ldots, N.
\end{aligned} \quad (9)$$

In the TPM, the solution of factor vector $\mathbf{u}^{(n)}$ in (8) is simply calculated by (10), as shown at the bottom of the page, where $||.||_2$ is the L2-norm operator.

$$\mathbf{u}^{(n)} = \frac{\mathcal{X} \times_1 \mathbf{u}^{(1)T} \ldots \times_{n-1} \mathbf{u}^{(n-1)T} \times_{n+1} \mathbf{u}^{(n+1)T} \ldots \times_N \mathbf{u}^{(N)T}}{||\mathcal{X} \times_1 \mathbf{u}^{(1)T} \ldots \times_{n-1} \mathbf{u}^{(n-1)T} \times_{n+1} \mathbf{u}^{(n+1)T} \ldots \times_N \mathbf{u}^{(N)T}||_2}, \quad (10)$$

Algorithm 2 Basic Pseudo-Code for Sparse Tensor Power Method (STPM)

Require: A Nth-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$, number of iterations K, rank R, and N sets of bandwidth parameters $\Lambda^{(1)}, \Lambda^{(2)}, \ldots, \Lambda^{(N)}$.
Output: The Nth-order diagonal tensor $\mathcal{D} \in \mathbb{R}^{R \times R \times \ldots \times R}$, and N factor matrices $\mathbf{U}^{(1)} \in \mathbb{R}^{I_1 \times R}, \mathbf{U}^{(2)} \in \mathbb{R}^{I_2 \times R}, \ldots, \mathbf{U}^{(N)} \in \mathbb{R}^{I_N \times R}$.
while $r = 1$ **to** R **do**
  Initialize N random unit vectors $\mathbf{u}^{(1)} \in \mathbb{R}^{I_1}$, $\mathbf{u}^{(2)} \in \mathbb{R}^{I_2}, \ldots, \mathbf{u}^{(N)} \in \mathbb{R}^{I_N}$.
  while $k = 1$ **to** K **do**
    while $n = 1$ **to** N **do**
      $\mathbf{u}^{(n)} \leftarrow \mathbf{u}^{(n)}$ in Eq. (10)
      Compute $\lambda^{(n)}$ in Eq. (13).
      $\mathbf{u}^{(n)} \leftarrow \mathit{Soft}(\mathbf{u}^{(n)}, \lambda^{(n)})$.
      $\mathbf{u}^{(n)} \leftarrow \begin{cases} \mathbf{u}^{(n)}/||\mathbf{u}^{(n)}||_2 & ||\mathbf{u}^{(n)}||_2 > 0 \\ 0 & \text{otherwise.} \end{cases}$
    end while
  end while
  $\mathcal{D}_{r,r,r} \leftarrow \mathcal{X} \times_1 \mathbf{u}^{(1)^T} \times_2 \mathbf{u}^{(2)^T} \ldots \times_N \mathbf{u}^{(N)^T}$.
  $\mathbf{U}^{(1)}_{:,r} \leftarrow \mathbf{u}^{(1)}, \mathbf{U}^{(2)}_{:,r} \leftarrow \mathbf{u}^{(2)}, \ldots, \mathbf{U}^{(N)}_{:,r} \leftarrow \mathbf{u}^{(N)}$.
  $\mathcal{X} \leftarrow \mathcal{X} - \mathcal{D}_{r,r,r}\mathbf{u}^{(1)} \circ \mathbf{u}^{(2)} \circ \ldots \circ \mathbf{u}^{(N)}$.
end while

Algorithm 3 Basic Pseudo-Code for Tensor Truncated Power Method (TTPM)

Require: A Nth-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$, number of iterations K, rank R, and set of cardinality values $S^{(1)}, S^{(2)}, \ldots, S^{(N)}$.
Output: The Nth-order diagonal tensor $\mathcal{D} \in \mathbb{R}^{R \times R \times \ldots \times R}$, and N factor matrices $\mathbf{U}^{(1)} \in \mathbb{R}^{I_1 \times R}, \mathbf{U}^{(2)} \in \mathbb{R}^{I_2 \times R}, \ldots, \mathbf{U}^{(N)} \in \mathbb{R}^{I_N \times R}$.
while $r = 1$ **to** R **do**
  Initialize N random unit vectors $\mathbf{u}^{(1)} \in \mathbb{R}^{I_1}$, $\mathbf{u}^{(2)} \in \mathbb{R}^{I_2}, \ldots, \mathbf{u}^{(N)} \in \mathbb{R}^{I_N}$.
  while $k = 1$ **to** K **do**
    while $n = 1$ **to** N **do**
      Compute $s^{(1)}, s^{(2)}, \ldots, s^{(N)}$ in Eq. (16).
      $\mathbf{u}^{(n)} \leftarrow \mathbf{u}^{(n)}$ in Eq. (10)
      $\mathbf{u}^{(n)} \leftarrow \mathit{Truncate}(\mathbf{u}^{(n)}, \mathrm{supp}(\mathbf{u}^{(n)}, s^{(n)}))$.
      $\mathbf{u}^{(n)} \leftarrow \dfrac{\mathbf{u}^{(n)}}{||\mathbf{u}^{(n)}||_2}$.
    end while
  end while
  $\mathcal{D}_{r,r,r} \leftarrow \mathcal{X} \times_1 \mathbf{u}^{(1)^T} \times_2 \mathbf{u}^{(2)^T} \ldots \times_N \mathbf{u}^{(N)^T}$.
  $\mathbf{U}^{(1)}_{:,r} \leftarrow \mathbf{u}^{(1)}, \mathbf{U}^{(2)}_{:,r} \leftarrow \mathbf{u}^{(2)}, \ldots, \mathbf{U}^{(N)}_{:,r} \leftarrow \mathbf{u}^{(N)}$.
  $\mathcal{X} \leftarrow \mathcal{X} - \mathcal{D}_{r,r,r}\mathbf{u}^{(1)} \circ \mathbf{u}^{(2)} \circ \ldots \circ \mathbf{u}^{(N)}$.
end while

Hence, the TPM iteratively updates randomly initialized factor vectors $\mathbf{u}^{(n)}$ based on the above equation until it converges to the solution. In a rank-R CPD optimization problem, the TPM can be applied to the residual tensor left after removing a previously calculated rank-1 tensor. However, this method only converges to a local optimum of (9) [12] and does not enforce orthogonality in subsequently computed components [1]. The overall process of TPM-based Nth-order tensor decomposition is summarized in Algorithm 1.

B. SPARSE TENSOR POWER METHOD (STPM)

To encourage sparsity in the TPM, the STPM reformulates (9) by directly adding L1-norm penalties to each of the factor vectors $\mathbf{u}^{(n)}$ [1]. The reformulated objective function in the STPM is as follows:

$$\max_{\mathbf{u}^{(1)},\ldots,\mathbf{u}^{(N)}} \mathcal{X} \times_1 \mathbf{u}^{(1)^T} \ldots \times_N \mathbf{u}^{(N)^T} - \sum_{n=1}^{N} \lambda^{(n)} ||\mathbf{u}^{(n)}||_1$$
$$\text{subject to } \mathbf{u}^{(n)^T}\mathbf{u}^{(n)} \leq 1 \text{ for } n = 1, 2, \ldots, N, \quad (11)$$

where $||.||_1$ is the L1-norm operator and $\lambda^{(n)}$ is the non-negative bandwidth parameter, which controls the amount of sparsity in the factor vectors $\mathbf{u}^{(n)}$. Because the STPM has relaxed the equality constraints in (9) into inequality constraints in (11), the analytical solution of factor vector $\mathbf{u}^{(n)}$ in STPM can be obtained by a soft-threshold operation given by

$$\mathbf{u}^{(n)}_i = \mathrm{sign}(\mathbf{u}^{(n)}_i)\max(|\mathbf{u}^{(n)}_i| - \lambda^{(n)}, 0) \text{ for } i = 1, \ldots, I_n. \quad (12)$$

Hereinafter, the soft-threshold operation on $\mathbf{u}^{(n)}$ with bandwidth parameter $\lambda^{(n)}$ is denoted as $\mathit{Soft}(\mathbf{u}^{(n)}, \lambda^{(n)})$.

To select the bandwidth parameters, the STPM, which is a data-driven algorithm, utilizes the Bayesian information criterion (BIC) [14] in every optimization iteration:

$$\lambda^{(n)} = \arg\min_{\lambda^{(n)} \in \Lambda^{(n)}} \mathit{BIC}_{\mathrm{STPM}}(\Lambda^{(n)}), \quad (13)$$

where $\lambda^{(n)}$ is one of the elements of the pre-specified set of bandwidth parameters $\Lambda^{(n)}$ [1], and $\mathit{BIC}_{\mathrm{STPM}}(\Lambda^{(n)})$ is

$$\log\left(\frac{||\mathcal{X} - d\mathbf{u}^{(1)} \circ \ldots \circ \mathbf{u}^{(N)}||^2}{I_1 \ldots I_N}\right) + \frac{\log(I_1 \ldots I_N)}{I_1 \ldots I_N}||\mathbf{u}^{(\mathbf{n})}||_0, \quad (14)$$

where $||\mathbf{u}^{(n)}||_0$ is the L0-norm of $\mathbf{u}^{(n)}$, which returns the number of non-zero elements of the factor vector $\mathbf{u}^{(n)}$, and d denotes the singular value of the approximated rank-1 tensor. The procedure for using STPM with Nrd-order tensor decomposition is described in Algorithm 2.

C. TENSOR TRUNCATED POWER METHOD (TTPM)

The TTPM encourages sparsity in each decomposition component by embedding the truncation procedure into the TPM [13]. Specifically, after updating the factor vectors $\mathbf{u}^{(n)}$ using TPM, the TTPM truncates its elements to preserve the entries of $s^{(n)}$ largest magnitudes. In this paper, we denote the truncation step in TTPM according to the notation in [13]:

$$\mathit{Truncate}(\mathbf{u}^{(n)}, \mathrm{supp}(\mathbf{u}^{(n)}, s^{(n)})), \quad (15)$$

where $\text{supp}(\mathbf{u}^{(n)}, s^{(n)})$ denotes the set of indices of $\mathbf{u}^{(n)}$ corresponding to its largest $s^{(n)}$ absolute values, and $s^{(n)}$ is the cardinality parameter that controls the sparsity in TTPM. Therefore, the TTPM imposes a desirable sparsity for the decomposition components [13]. The detailed TTPM procedure is shown in Algorithm 3.

Similar to the STPM, the BIC-type criterion is utilized in TTPM to estimate cardinality parameters. Thus, the best combination of cardinality parameters is obtained by solving the following optimization problem:

$$s^{(1)}, .., s^{(N)} = \underset{s^{(1)} \in S^{(1)}, .., s^{(N)} \in S^{(N)}}{\arg\min} BIC_{\text{TTPM}}(S^{(1)}, \ldots, S^{(N)}), \tag{16}$$

where $S^{(n)}$ is a set of cardinality parameters which include the $s^{(n)}$ and $BIC_{\text{TTPM}}(S^{(1)}, \ldots, S^{(N)})$ defined as follows:

$$\log\left(\frac{||\boldsymbol{\mathcal{X}} - d\mathbf{u}^{(1)} \circ \ldots \circ \mathbf{u}^{(N)}||^2}{I_1 \ldots I_N}\right) + \frac{\log(I_1 \ldots I_N)}{I_1 \ldots I_N} \sum_{n=1}^{N} ||\mathbf{u}^{(\mathbf{n})}||_0, \tag{17}$$

where d denotes the singular value of approximated rank-1 tensor, and $\mathbf{u}^{(n)}$ denotes the factor vector.

D. PROBLEM DEFINITION

The TPM can be considered as a gradient descent step with infinite step size, corresponding to the problem of obtaining the best rank-1 approximation of the input tensor, which is non-convex optimization problem [9], [10]. Unlike the PM, where the isolated stationary points are at most the number of dimensions, the number of stationary points for the TPM is exponential in the input dimension [11]. Moreover, the TPM does not enforce orthogonality subsequently in calculated decomposition factors [1], [4], [13]. These problems, defined as the curse-of-dimensionality problem, complicate the analysis of TPM far more challenging.

Despite the above challenges, both STPM and TTPM are developed to enhance the performance of the TPM by incorporating sparsity in the decomposition factors. Because the primary purpose of embedding to enforce orthogonality in TPM, the methods exhibit enhanced performance based on a well-defined theoretical analysis [1], [13]. However, both methods control the amount of sparsity via the combinatorial optimization process, which based on data-driven BIC method [14]. Because the combinatorial optimization problem is based on the calculation of all possible combinations of variables, the computational cost is significantly higher than TPM. Furthermore, the data-driven BIC generates more complex TPM variants in large-scale high dimensional tensor decomposition. Therefore, both of them do not meet the set requirements for the TPM, such as ensuring efficiency for rank-1 tensor estimation.

IV. PROPOSED METHOD

In this section, a novel tensor power method called the FCTPM is proposed. The FCTPM is based on a proper combination of TTD and the PM via the proposed circulant updating method. The proposed tensor model inherits stability from TTD and low computational cost from the PM. The proposed algorithm is summarized in Algorithm 4 and Fig. 1.

Algorithm 4 Basic Pseudo-Code for Fast Circulant Tensor Power Method (FCTPM)

Require: A Nth-order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$, number of iterations K, rank R.

Output: The Nth-order diagonal tensor $\boldsymbol{\mathcal{D}} \in \mathbb{R}^{R \times R \times \ldots \times R}$, and N factor matrices $\mathbf{U}^{(1)} \in \mathbb{R}^{I_1 \times R}$, $\mathbf{U}^{(2)} \in \mathbb{R}^{I_2 \times R}$, $\ldots$, $\mathbf{U}^{(N)} \in \mathbb{R}^{I_N \times R}$.

while $r = 1$ **to** R **do**
  Initialize $N-1$ random unit vectors $\mathbf{g}^{(1)} \in \mathbb{R}^{I_1}$, $\mathbf{g}^{(2)} \in \mathbb{R}^{I_2}$, $\ldots$, $\mathbf{g}^{(N-1)} \in \mathbb{R}^{I_{N-1}}$.
  $\mathbf{X}^{(1)} \leftarrow \boldsymbol{\mathcal{X}}_{(1)}$
  while $k = 1$ **to** K **do**
    while $n = 1$ **to** $N-2$ **do**
      $\mathbf{v}^{(n)} \leftarrow \dfrac{\mathbf{X}^{(n)^T}\mathbf{g}^{(n)}}{||\mathbf{X}^{(n)^T}\mathbf{g}^{(n)}||_2}$.
      $\mathbf{X}^{(n+1)} \leftarrow reshape(\mathbf{v}^{(n)}, [I_{n+1}, \prod_{j=n+2}^{N} I_j])$.
    end while
    $\mathbf{g}^{(N)} \leftarrow \dfrac{\mathbf{X}^{(N-1)^T}\mathbf{g}^{(N-1)}}{||\mathbf{X}^{(N-1)^T}\mathbf{g}^{(N-1)}||_2}$.
    $\mathbf{v}^{(N-1)} \leftarrow reshape(\mathbf{g}^{(N)}, [I_N])$
    while $n = N-1$ **to** 2 **do**
      $\mathbf{g}^{(n)} \leftarrow \dfrac{\mathbf{X}^{(n)}\mathbf{v}^{(n)}}{||\mathbf{X}^{(n)}\mathbf{v}^{(n)}||_2}$.
      $\mathbf{v}^{(n-1)} \leftarrow reshape(\mathbf{g}^{(n)}\mathbf{v}^{(n)^T}, [\prod_{j=n}^{N} I_j])$
    end while
    $\mathbf{g}^{(1)} \leftarrow \dfrac{\mathbf{X}^{(1)}\mathbf{v}^{(1)}}{||\mathbf{X}^{(1)}\mathbf{v}^{(1)}||_2}$.
  end while
  $\boldsymbol{\mathcal{D}}_{r,r,r} \leftarrow \boldsymbol{\mathcal{X}} \times_1 \mathbf{g}^{(1)^T} \times_2 \mathbf{g}^{(2)^T} \ldots \times_N \mathbf{g}^{(N)^T}$.
  $\mathbf{U}^{(1)}_{:,r} \leftarrow \mathbf{u}^{(1)}$, $\mathbf{U}^{(2)}_{:,r} \leftarrow \mathbf{u}^{(2)}$, $\ldots$, $\mathbf{U}^{(N)}_{:,r} \leftarrow \mathbf{u}^{(N)}$.
  $\boldsymbol{\mathcal{X}} \leftarrow \boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{D}}_{r,r,r}\mathbf{g}^{(1)} \circ \mathbf{g}^{(2)} \circ \ldots \circ \mathbf{g}^{(N)}$.
end while

A. FAST CIRCULANT TENSOR POWER METHOD (FCTPM)

The FCTPM consists of two major components: rank-1 TTD for approximating rank-1 tensors through low-rank decomposition of auxiliary unfolding matrices, and the PM for determining right and left singular vectors from randomly initialized vectors.

1) DEFINITION OF RANK-1 TTD

In FCTPM, the rank-1 tensor approximation of a high-dimensional input tensor is performed using rank-1 TTD. To comprehend rank-1 TTD, let us assume an N-dimensional input tensor $\boldsymbol{\mathcal{X}}$ of size $I_1 \times I_2 \times \ldots \times I_N$. The TTD represents the input tensor $\boldsymbol{\mathcal{X}}$ as N sequentially connected core tensors $\boldsymbol{\mathcal{G}}^{(n)} \in \mathbb{R}^{R_{n-1} \times I_n \times R_n}$, as defined in (4). As previously stated, rank-1 TTD utilizes TT-rank, which only comprises 1; in that

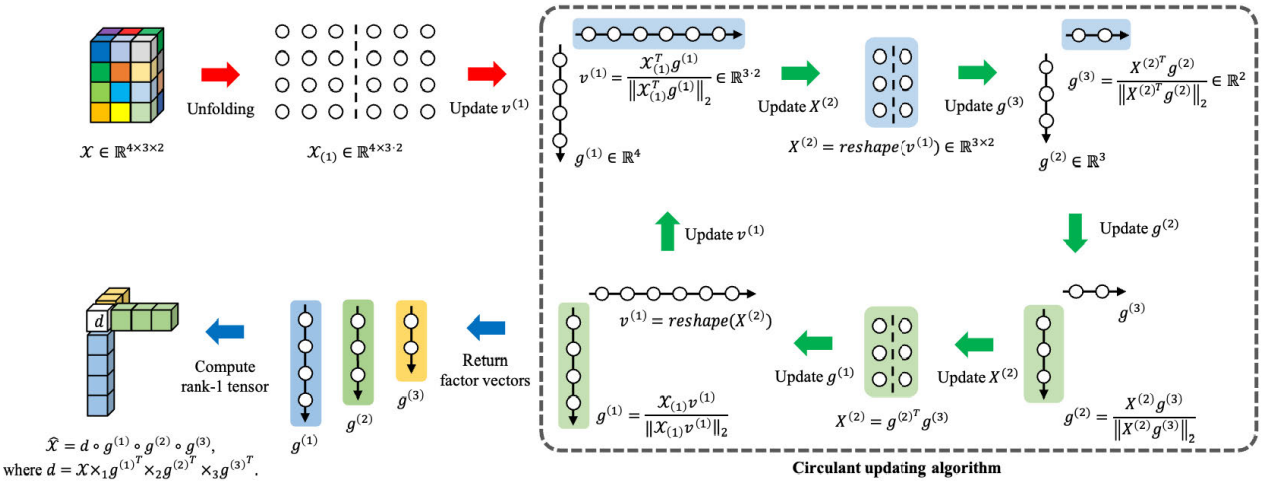


FIGURE 1. The overall procedures of the proposed algorithm in a 3rd-order tensor decomposition example.

case, (4) can be rewritten as follows:

$$\mathcal{X} = \mathcal{G}^{(1)}_{1,:,1} \circ \mathcal{G}^{(2)}_{1,:,1} \circ \ldots \circ \mathcal{G}^{(N)}_{1,:,1}, \tag{18}$$

where $\mathcal{G}^{(n)}$ denotes the core tensor of rank-1 TTD of size $1 \times I_n \times 1$. Because all the elements of TT-rank are 1, the summation operation in (4) can be discarded in the aforementioned equation. Furthermore, if the dimensions of the core tensor $\mathcal{G}^{(n)}$ in the above equation is squeezed, the core tensors can be considered as the vector data $\mathbf{g}^{(n)}$. Therefore, (18) can be rewritten as follows:

$$\mathcal{X} = \mathbf{g}^{(1)} \circ \mathbf{g}^{(2)} \circ \ldots \circ \mathbf{g}^{(N)}, \tag{19}$$

where $\mathbf{g}^{(n)} \in \mathbb{R}^{I_n}$ is the 1st-order tensor of the core tensor $\mathcal{G}^{(n)}$ in (18). The above equation proves that the definition of rank-1 TTD is exactly the same as that of rank-1 CPD. In other words, the core tensors of rank-1 TTD are the same as the factor vectors of rank-1 CPD.

2) COMPUTATION OF RANK-1 TTD IN FCTPM

By utilizing rank-1 TTD in (19), the proposed objective function is defined as follows:

$$\min_{\mathbf{g}^{(1)},\ldots,\mathbf{g}^{(N)}} ||\mathcal{X} - d\mathbf{g}^{(1)} \circ \ldots \circ \mathbf{g}^{(N)}||^2$$
$$\text{subject to } d > 0 \text{ and } \mathbf{g}^{(n)^T}\mathbf{g}^{(n)} = 1 \text{ for } n = 1, 2, \ldots, N, \tag{20}$$

where $\mathcal{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ denotes the input tensor, d denotes the most dominant singular value of the approximated rank-1 tensor, d denotes the approximated singular value, and $\mathbf{g}^{(n)} \in \mathbb{R}^{I_n}$ denotes the factor vector from the core tensor of rank-1 TTD.

As stated previously in Section II, the TTD decomposes the high-order tensor via the low-rank decomposition of the auxiliary unfolding matrices of the input tensor. Therefore, the proposed FCTPM, which follows the decomposition approach of TTD, solves the objective function, as shown in (20). Hence, the computation of the first component vector $\mathbf{g}^{(1)}$ in (20) is as follows:

$$\min_{\mathbf{g}^{(1)},\mathbf{v}} ||\mathcal{X}_{(1)} - d^{(1)}\mathbf{g}^{(1)}\mathbf{v}^{(1)^T}||_F^2$$
$$\text{subject to } d^{(1)} > 0, \mathbf{g}^{(1)^T}\mathbf{g}^{(1)} = 1, \text{ and } \mathbf{v}^{(\mathbf{1})^T}\mathbf{v}^{(\mathbf{1})} = 1, \tag{21}$$

where $||\mathbf{X}||_F$ denotes the Frobenius norm of matrix $\mathbf{X}$, $\mathcal{X}_{(1)}$ denotes the mode-1 unfolding matrix of size $I_1 \times I_2 \ldots I_N$, $d^{(1)}$ represents the largest singular value of $\mathcal{X}_{(1)}$, $\mathbf{g}^{(1)}$ represents the first factor vector of size I_1, $\mathbf{v}^{(1)} \in \mathbb{R}^{I_2 \ldots I_N}$ is the left singular vector of $\mathcal{X}_{(1)}$, and $d^{(1)}$ denotes the calculated singular value of $\mathcal{X}_{(1)}$.

As shown in the above optimization problem, the first-factor vector $\mathbf{g}^{(1)}$ is obtained by rank-1 matrix decomposition of the mode-1 unfolding $\mathcal{X}_{(1)}$. This characteristic implies that the proposed tensor model can be incorporated with the theoretically well-understood matrix decomposition algorithm for the overall computation of factor vectors.

After solving the above matrix decomposition problem, $\mathbf{v}^{(1)}$ is reshaped into a matrix $\mathbf{X}^{(2)}$ of size $I_2 \times \prod_{j=3}^{N} I_j$, for the calculation of the second factor vector $\mathbf{g}^{(2)}$:

$$\mathbf{X}^{(2)} = reshape\left(\mathbf{v}^{(1)}, [I_2, \prod_{j=3}^{N} I_j]\right). \tag{22}$$

After reshaping the right singular vector $\mathbf{v}^{(1)}$ in (21), the rank-1 matrix decomposition on $\mathbf{X}^{(2)}$ is performed as follows:

$$\min_{\mathbf{g}^{(2)},\mathbf{v}^{(2)}} ||\mathbf{X}^{(2)} - d^{(2)}\mathbf{g}^{(2)}\mathbf{v}^{(2)^T}||_F^2$$
$$\text{subject to } d^{(2)} > 0, \mathbf{g}^{(2)^T}\mathbf{g}^{(2)} = 1, \text{ and } \mathbf{v}^{(\mathbf{2})^T}\mathbf{v}^{(\mathbf{2})} = 1, \tag{23}$$

where $d^{(2)}$ is the most dominant singular value of $\mathbf{X}_{(2)}$, and $\mathbf{g}^{(2)}$ is the second factor vector of the rank-1 TTD.

After optimizing (23), $\mathbf{v}^{(2)}$ is reshaped, and the third-factor vector $\mathbf{g}^{(3)}$ is computed. This sequential process is repeatedly

performed until the right singular vector of the matrix decomposition procedure is the last factor vector $\mathbf{g}^{(N)} \in \mathbb{R}^{I_N}$, which is represented as follows:

$$\min_{\mathbf{g}^{(N-1)},\mathbf{g}^{(N)}} ||\mathbf{X}^{(N-1)} - d^{(N-1)}\mathbf{g}^{(N-1)}\mathbf{g}^{(N)^T}||_F^2$$
$$\text{subject to } d > 0,\ \mathbf{g}^{(N-1)^T}\mathbf{g}^{(N-1)} = 1, \text{ and } \mathbf{g}^{(N)^T}\mathbf{g}^{(N)} = 1, \tag{24}$$

where $\mathbf{X}^{(N-1)}$ is expressed as $reshape(\mathbf{v}^{(N-2)}, [I_{N-1}, I_N])$, $d^{(N-1)}$ is the dominant singular value of the reshaped matrix $\mathbf{X}^{(N-1)}$, and $\mathbf{g}^{(N-1)}$ and $\mathbf{g}^{(N)}$ are the $N-1$th and Nth factor vectors, respectively.

From (21) to (24), the N core vectors $\mathbf{g}^{(n)}$ of rank-1 TTD can be computed; they are considered as the factor vectors in the approximated rank-1 tensor. The singular value of the rank-1 tensor can be computed as follows:

$$d = \prod_{i=1}^{N-1} d^{(i)}, \tag{25}$$

where $d^{(i)}$ is a computed singular value of the ith auxiliary unfolding matrix decomposition.

3) POWER METHOD (PM) IN FCTPM

In FCTPM, the component vector $\mathbf{g}^{(n)}$ is computed via the rank-1 matrix decomposition on auxiliary unfolding matrices, because the proposed tensor model utilizes the rank-1 TTD. In other words, the proposed method can utilize matrix decomposition methodologies, enabling the FCTPM to avoid the curse-of-dimensionality problem. During the factor vector calculation of the FCTPM, the algorithm requires only the most dominant singular value and its associated right and left singular vectors. Therefore, the proposed algorithm operates in combination with the PM which is a simple and powerful matrix decomposition algorithm.

In the 1st factor vector calculation problem in (21), the PM initializes $\mathbf{g}^{(1)}$ as a random unit vector, i.e., $||\mathbf{g}^{(1)}||_2 = 1$. Subsequently, the PM iteratively performs vector-matrix operations until the pre-specific termination condition is satisfied, as follows:

$$\mathbf{v}^{(1)} = \frac{\boldsymbol{\mathcal{X}}_{(1)}^T\mathbf{g}^{(1)}}{||\boldsymbol{\mathcal{X}}_{(1)}^T\mathbf{g}^{(1)}||_2} \in \mathbb{R}^{\prod_{j=2}^N I_j},$$
$$\mathbf{g}^{(1)} = \frac{\boldsymbol{\mathcal{X}}_{(1)}\mathbf{v}^{(1)}}{||\boldsymbol{\mathcal{X}}_{(1)}\mathbf{v}^{(1)}||_2} \in \mathbb{R}^{I_1}, \tag{26}$$

where $\boldsymbol{\mathcal{X}}_{(1)} \in \mathbb{R}^{I_1 \times I_2 \ldots I_N}$ is the mode-1 unfolding matrix of $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$, and $\mathbf{g}^{(1)}$ and $\mathbf{v}^{(1)}$ are the left and right singular vectors of $\boldsymbol{\mathcal{X}}_{(1)}$, respectively. The singular value $d^{(1)}$, which is associated with the left and right singular vectors in the above equation, are as follows:

$$d^{(1)} = \mathbf{g}^{(1)^T}\boldsymbol{\mathcal{X}}_{(1)}\mathbf{v}^{(1)}. \tag{27}$$

For the convergence analysis of the PM in the proposed algorithm, we assume that there exist orthonormal bases $\mathbf{U} \in \mathbb{R}^{I_1 \times R}$ and $\mathbf{V} \in \mathbb{R}^{I_2 \ldots I_N \times R}$ which satisfy the following condition:

$$\boldsymbol{\mathcal{X}}_{(1)} = \sum_{j=1}^{R} \mathbf{d}_j \mathbf{U}_{:,j}\mathbf{V}_{:,j}^T, \tag{28}$$

where $\mathbf{d} \in \mathbb{R}^R$ is a vector in which each element indicates the singular value corresponding to the singular vectors. According to the equations above, after k iterations of (26), $\mathbf{g}^{(1)}$ and $\mathbf{v}^{(1)}$ are defined as follows:

$$\mathbf{g}^{(1)} = \delta_{\mathbf{g}^{(1)}} \sum_{j=1}^{R} \mathbf{d}_j^{2k}\mathbf{U}_{:,j}\mathbf{U}_{:,j}^T\mathbf{g}^{(1)},$$
$$\mathbf{v}^{(1)} = \delta_{\mathbf{v}^{(1)}} \sum_{j=1}^{R} \mathbf{d}_j^{2k+1}\mathbf{V}_{:,j}\mathbf{U}_{:,j}^T\mathbf{g}^{(1)}, \tag{29}$$

where $\delta_{\mathbf{g}^{(1)}}$ and $\delta_{\mathbf{v}^{(1)}}$ are the corresponding normalization factors. Therefore, after k iterations of the PM, $||\mathbf{v}^{(1)}||_2^2/||\mathbf{g}^{(1)}||_2^2$ satisfies:

$$\frac{||\mathbf{v}^{(1)}||_2^2}{||\mathbf{g}^{(1)}||_2^2} = 1$$
$$= \mathbf{d}_1^2 \left(\frac{\delta_{\mathbf{v}^{(1)}}^2}{\delta_{\mathbf{g}^{(1)}}^2}\right)\left(\frac{C + \sum_{j=\mu+1}^{R}(\frac{\mathbf{d}_j}{\mathbf{d}_1})^{4k+2}||\mathbf{d}_j\mathbf{V}_{:,j}\mathbf{U}_{:,j}^T\mathbf{g}^{(1)}||_2^{-2}}{C + \sum_{j=\mu+1}^{R}(\frac{\mathbf{d}_j}{\mathbf{d}_1})^{4k}||\mathbf{d}_j\mathbf{U}_{:,j}\mathbf{U}_{:,j}^T\mathbf{g}^{(1)}||_2^{-2}}\right), \tag{30}$$

where μ is the multiplicity of the singular value $\mathbf{d}_1$ and $C = \sum_{j=1}^{\mu}(\mathbf{U}_{:,j}^T\mathbf{g}^{(1)})^2$. Thus, $\delta_{\mathbf{v}^{(1)}}/\delta_{\mathbf{g}^{(1)}}$ converges to the largest singular value $\mathbf{d}_1$. From this convergence property, $||\boldsymbol{\mathcal{X}}_{(1)}^T\mathbf{g}^{(1)} - \mathbf{d}_1\mathbf{v}^{(1)}||_2$ becomes zero as $\boldsymbol{\mathcal{X}}_{(1)}^T\mathbf{g}^{(1)} = (\delta_{\mathbf{v}^{(1)}}/\delta_{\mathbf{g}^{(1)}})\mathbf{v}^{(1)}$.

4) CIRCULANT UPDATING ALGORITHM IN FCTPM

From the explanation of rank-1 TTD in the previous subsection, the nth factor vector $\mathbf{g}^{(n)}$ depends on the previously computed right singular vector $\mathbf{v}^{(n-1)}$, which is obtained from the $n-1$th factor vector $\mathbf{g}^{(n-1)}$ computation. Moreover, the matrix decomposition for each factor vector is independently performed. These properties can significantly contribute to the local optima problem, causing as critical performance loss in the FCTPM. Therefore, the circulant updating algorithm is proposed to prevent this problem.

To integrate each unfolding matrix decomposition procedure, the circulant updating algorithm simply changes the order of the left singular vector computations. In the existing PM, the left singular vector calculation is performed immediately after the right singular vector computations, defined as (26). According to the decomposition approach of rank-1 TTD, the right singular vector $\mathbf{v}^{(n)}$ is reshaped and utilized for the computation of the next factor vectors $\mathbf{g}^{(n+1)}$ and $\mathbf{v}^{(n+1)}$ after the computation by the PM. This property implies that the $\mathbf{v}^{(n)}$ can be reconstructed by $\mathbf{g}^{(n+1)}$ and $\mathbf{v}^{(n+1)}$, which is as follows:

$$\mathbf{v}^{(n)} \approx \hat{\mathbf{v}}^{(n)} = reshape(d^{(n+1)}\mathbf{g}^{(n+1)}\mathbf{v}^{(n+1)^T}, [I_{n+1} \ldots I_N]), \tag{31}$$

TABLE 1. Comparison of computational complexities.

Method	Computational complexity
FCTPM (ours)	$O\left(RK(2I^N + 4\sum_{j=2}^{N-1} I^j)\right)$
TPM [4]	$O\left(RK(N\sum_{j=1}^{N} I^j)\right)$
STPM [1]	$O\left(RK(2\Lambda N I^N + \Lambda N\sum_{j=1}^{N} I^j)\right)$
TTPM [13]	$O\left(RK(2S^N I^N + \sum_{j=1}^{N} I^j)\right)$
Rank-1 TTD [29]	$O\left(R\sum_{j=3}^{N+1}(I^j + I^3 + I^2)\right)$
ALS-based CPD [5]	$O\left(K(2^N I^N + NR^2)\right)$

R: # of ranks, N: # of dimensions, K: # of iterations, I^n: size of nth axis, Λ: # of bandwidth parameters in STPM, S: # of cardinally values in TTPM.

where $\hat{\mathbf{v}}^{(n)}$ is the reconstructed $\mathbf{v}^{(n)}$, and $d^{(n+1)}$ denotes the singular value associated with both $\mathbf{g}^{(n+1)}$ and $\mathbf{v}^{(n+1)}$. Therefore, the nth left singular vector $\mathbf{g}^{(n)}$ is updated in the FCTPM by the reconstructed right singular vector $\hat{\mathbf{v}}^{(n)}$ as follows:

$$\mathbf{g}^{(n)} = \frac{\mathbf{X}^{(n)}\hat{\mathbf{v}}^{(n)}}{||\mathbf{X}^{(n)}\hat{\mathbf{v}}^{(n)}||_2}. \tag{32}$$

By utilizing the circulant updating algorithm, the FCTPM can stably approximate a rank-1 tensor by globally optimizing each factor vector.

The overall procedure of the proposed algorithm is presented in Algorithm 4 and Fig. 1. Especially, Fig. 1. shows the proposed algorithm with a 3rd-order tensor decomposition example to help ease of understanding. The above Algorithm 4 and Fig. 1. show that the FCTPM is a fast rank-1 tensor approximation based on a suitable combination of rank-1 TTD and the PM via the novel circulant updating algorithm. Thus, the proposed algorithm can inherit the advantages of both TTD and the PM simultaneously.

B. COMPUTATIONAL COMPLEXITY ANALYSIS

To compare the computational complexity of the FCTPM with that of the other methods, we assume the decomposition of the N-dimensional tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I\times\ldots\times I}$ to R rank-1 tensors over K iterations. The analysis only takes into account the dominant term using Big O notation, $O(.)$. The results of this comparison are summarized in Table.1. From the results, it is observed that the proposed algorithm has the lowest computational cost compared to the other algorithms. In TPM, the major computational cost originates from the vector-by-unfolding matrix multiplication operation. On the contrary, since the FCTPM is based on rank-1 TTD, the computational complexity is reduced because the vector-by-unfolding matrix multiplication is converted to vector-by-auxiliary unfolding matrix multiplication. In addition, the TTD is a well-known tensor decomposition algorithm that enhances the representation compactness and numerical estimation performance [31]. The proposed algorithm can inherit the aforementioned advantages and demonstrate a lower computational complexity. The STPM and TTPM are developed for enhancing the performance of TPM via a combinatorial optimization-based sparsity hyperparameter combination selection. It is confirmed that the additional procedures induce huge computation costs. Although the enhanced versions of TPM can enforce orthogonality in the subsequently computed decomposition factors, the associated computational costs are huge, and therefore, cannot meet the primary condition of the tensor power method, which is to decompose high-order tensors with an efficient computation. Obviously, the FCTPM has a lower computational cost than the STPM and TTPM.

In the computational complexity analysis, we also included the computational costs of rank-1 TTD and ALS-based CPD. The rank-1 TTD means that the CPD computation is performed using the equivalence between CPD and TTD; this method resembles the proposed algorithm in terms of the decomposition method used. However, since the TTD is originally based on the SVD of unfolding auxiliary matrices, the computational cost is larger than that of the proposed algorithm. The ALS-based CPD is a representative computation of CPD, which addresses the decomposition problem by converting the nonconvex optimization problem into convex subproblems [12]. Each convex subproblem is then optimized by a matrix-Khatri-Rao matrix, where the Khatri-Rao matrix has a size of $R\times I^{N-1}$. Therefore, this optimization singularity leads to huge computational costs in ALS-based CPD. Due to the effective combination between rank-1 TTD and PM via a novel circulant updating algorithm, the FCTPM achieves the lowest computational costs compared to conventional methods.

V. EXPERIMENTS

This section describes both numerical simulations and image reconstruction experiments that were performed to verify the computational complexity and rank-1 tensor approximation performance of the proposed algorithm. The FCTPM and the algorithms used for comparisons, including the TPM, STPM, and TTPM, were actualized using TensorLy [44], Python-based tensor algebra framework. All the experiments were performed on an Intel core i7 3.0 GHz processor and 32 GB memory. The STPM and TTPM experiments required the bandwidth parameter sets $\Lambda^{(n)}$ in (13) and the cardinality sets $S^{(n)}$ in (16), respectively. These hyperparameter sets in STPM and TTPM were composed of $[10^{-2}, 10^{-1.8}, 10^{-1.6}, \ldots, 10^{0}] \in \mathbb{R}^{11}$.

A. NUMERICAL SIMULATION EXPERIMENT

The primary purpose of the FCTPM is to approximate several dominant rank-1 tensors from the input tensor. In other words, the proposed algorithm aims to identify the most dominant singular values and its associated factor vectors. For numerical simulation experiments, input tensor $\hat{\boldsymbol{\mathcal{X}}}$ were constructed as follows:

$$\hat{\boldsymbol{\mathcal{X}}} = \boldsymbol{\mathcal{X}} + \alpha\boldsymbol{\mathcal{N}}, \tag{33}$$

where $\boldsymbol{\mathcal{X}}$ is a normalized low-rank tensor, $\boldsymbol{\mathcal{N}}$ is a randomly generated tensor from the ''continuous uniform'' distribution,

TABLE 2. Experimental results on numerical simulation experiments.

Case	1		2		3	
Tensor Size	$30 \times 30 \times 30 \times 30 \times 30$		$50 \times 50 \times 50 \times 50 \times 50$		$15 \times 15 \times 15 \times 15 \times 15 \times 15 \times 15$	
Rank R	15		20		8	
Iteration K	10		15		15	
Metrics	Recon.Err	Proc.Time (sec)	Recon.Err	Proc.Time (sec)	Recon.Err	Proc.Time (sec)
FCTPM (ours)	**0.0738**	**5.619**	**0.0274**	**81.271**	**0.0291**	**32.889**
TPM [4]	0.0796	21.613	0.0278	277.07	0.9301	99.982
STPM [1]	0.0814	154.174	0.0294	1958.187	0.4536	678.878
TTPM [13]	**0.0738**	10558.736	0.0279	13577.632	0.0315	61133.709

and α is the coefficient of $\mathcal{N}$. Naturally, the sizes of $\mathcal{X}$ and $\mathcal{N}$ are the same for element-wise addition. The three cases of numerical simulation can be defined as follows:

- Case 1: Decomposition of the 5th-order tensor $\hat{\mathcal{X}} \in \mathbb{R}^{30\times\ldots\times30}$ with rank $R = 15$, number of iterations $K = 10$, and $\alpha = 10^{-3}$.
- Case 2: Decomposition of the 5th-order tensor $\hat{\mathcal{X}} \in \mathbb{R}^{50\times\ldots\times50}$ with rank $R = 20$, number of iterations $K = 10$ and $\alpha = 10^{-5}$.
- Case 3: Decomposition of the 7th-order tensor $\hat{\mathcal{X}} \in \mathbb{R}^{15\times\ldots\times15}$ with rank $R = 8$, number of iterations $K = 15$ and $\alpha = 10^{-5}$.

Two metrics were used in the simulation experiments, namely the processing time and reconstruction error. The processing time was used to measure the computational complexity of each algorithm. The reconstruction error can be expressed as follows:

$$\text{Reconstruction error} := \frac{||\mathcal{X} - \sum_{r=1}^{R} \mathcal{D}_{r,\ldots,r} \mathbf{U}^{(1)}_{:,r} \circ \ldots \circ \mathbf{U}^{(N)}_{:,r}||_2}{||\mathcal{X}||_2} \tag{34}$$

where $\mathcal{X}$ is a noise free low-rank tensor in (33) and the $\sum_{r=1}^{R} \mathcal{D}_{r,\ldots,r} \mathbf{U}^{(1)}_{:,r} \circ \ldots \circ \mathbf{U}^{(N)}_{:,r}$ in the numerator is the tensor reconstructed from the noised tensor $\hat{\mathcal{X}}$ in (33).

Throughout these experiments, it was confirmed that the reconstruction performance on various high dimensional tensors and the processing time vary according to the size of tensor. Furthermore, the robustness of the model was evaluated by the utilizing noise tensor $\mathcal{N}$. The results are summarized in Table. 2. These three simulations were independently repeated 50 times. Hence, the entry in Table. 2 is the average of the 50 results.

Overall, the FCTPM exhibited the shortest processing time as compared to the other considered algorithms in the numerical simulation results. Therefore, the FCTPM is the most efficient algorithm amongst all the other methodologies. Furthermore, it was observed that the TPM required the longest processing time for computing the vector via unfolding matrix multiplication when comparing it to the proposed algorithm. Unlike the TPM, the FCTPM converted this multiplication operation to auxiliary unfolding matrix multiplication because it is based on rank-1 TTD; this singularity of the proposed algorithm is the major reason behind its observed computational efficiency.

In simulation case 3, the reconstruction error of TPM is much larger than in the other simulation cases. As mentioned in section III, this problem arises because TPM does not enforce orthogonality in the subsequently computed decomposition factors. Hence, the STPM and TTPM were developed to solve this problem by embedding sparsity into the decomposition components. The selection of the sparsity control hyperparameters was optimized using combinatorial optimization, which requires searching through all possible combinations of hyperparameters. As a result, although STPM and TTPM enhance the reconstruction performance of the TPM, they require a high processing time. Even in simulation cases 1 and 2, the STPM recorded a higher reconstruction error. In simulation case 3, a moderate improvement in the reconstruction error was observed. On the contrary, the TTPM can effectively enhance the approximation performance of TPM. However, this method recorded the highest processing time. In other words, the conventional enhanced version of TPM cannot meet the primary purpose of the tensor power method: the efficient approximation of rank-1 tensors.

From the experimental results, it can be concluded that the FCTPM incurs the lowest computational costs while ensuring a good reconstruction performance in high dimensional tensor decomposition when compared to TPM, STPM, and TTPM.

B. IMAGE RECONSTRUCTION EXPERIMENT

To further understand the performance of the FCTPM, image reconstruction experiments were performed. A colored image is a type of representative tensor data, and its size can be expressed as width × height × color. In our experiment, the number of color dimensions was kept at 3 since only RGB colored images were used. Two image of different sizes were utilized described as follows:

- The ''Lenna'' image, with a size of $512 \times 512 \times 3$. This image is shown at the top of the first column of Fig. 2.
- The ''Frymire'' image, with a size of $1024 \times 1024 \times 3$. The image is shown at the third row of the first column of Fig. 2.

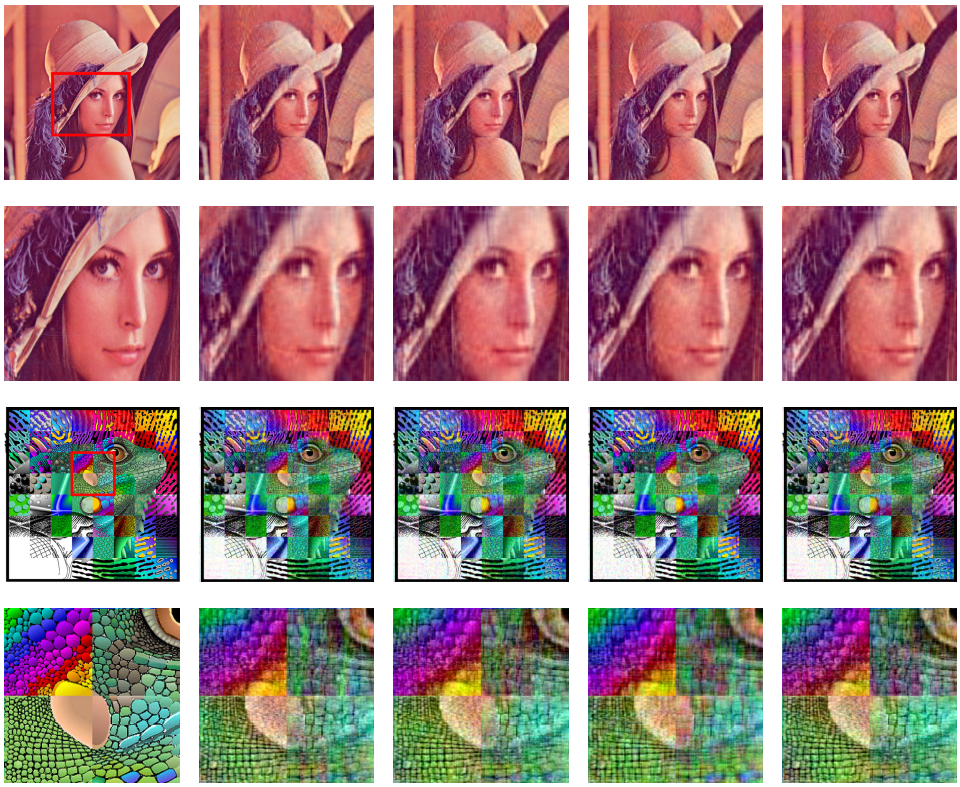


FIGURE 2. Experimental results of clean image reconstruction. First row: experimental results on noise free "Lenna" image of size 512 × 512 × 3. Second row: experimental results of noise free "Frymire" image of size 1024 × 1024 × 3. (a) Input images. (b) Images reconstructed via FCTPM. (c) Images reconstructed via TPM. (d) Images reconstructed via STPM. (e) Images reconstructed via TTPM.

To evaluate the reconstruction performance in a practical situation, original images were reconstructed from corrupted ones. For this purpose, noised image were corrupted using Gaussian noise with a mean of 0 and standard deviation of 0.3. The noised images are depicted at the first column of first and third rows in Fig. 3.

Similar to the numerical simulation experiments, the computational complexity can be determined based on the processing time, and the reconstruction accuracy can be measured based on the representative image quality metrics, peak signal-to-noise ratio (PSNR) and structural similarity index map (SSIM) [45]. These metrics can be defined as follows:

$$\text{PSNR} := 10 \log_{10} \left(\frac{1}{\text{MSE}} \right), \tag{35}$$

where, MSE denotes the mean square error and is defined as:

$$\text{MSE} := \sum_{w=1}^{\text{width}} \sum_{h=1}^{\text{height}} \sum_{c=1}^{\text{color}} \left(\mathcal{X}_{w,h,c} - \mathcal{X}^*_{w,h,c} \right), \tag{36}$$

where $\mathcal{X}^*$ denotes the reconstructed image and $\mathcal{X}$ means the input image. Thus, the PSNR represents a measure of the peak error. Unlike PSNR, the SSIM evaluates the differences between the target image and reconstructed image as perceived by a human visual system rather than in terms of numerical errors. Both these evaluation metrics were actualized by a MATLAB image processing toolbox.

The experiment results help us visually compare the robustness of the proposed algorithm with competing algorithms. All the image reconstruction experimental results are summarized in Table. 3 and Figs. 2 and 3. Each of these four experiments was repeated 50 times. As a result, the entry in Table. 3 shows the average value of the 50 values obtained from these repetitions.

In the image reconstruction experiments on the noise free "Lenna" image, the FCTPM and TTPM showed similar performances in terms of the image quality metrics. However, with regard to the processing time, the FCTPM was the fastest and requiring a time of less than one second to compute 50 dominant rank-1 tensors of the "Lenna" image, which has a size of 512 × 512 × 3. This shows the FCTPM is the fastest rank-1 tensor approximation algorithm and has better reconstruction performance than the other algorithms. In the clean "Frymire" image experiment, the FCTPM demonstrated the lowest processing time and showed the highest reconstruction performance. Furthermore, it was observed that the FCTPM achieved a good image reconstruction performance regardless of the image size. Fig. 2 shows the images reconstructed by the FCTPM and the other algorithms. It was difficult to visually identify the differences between the FCTPM and the

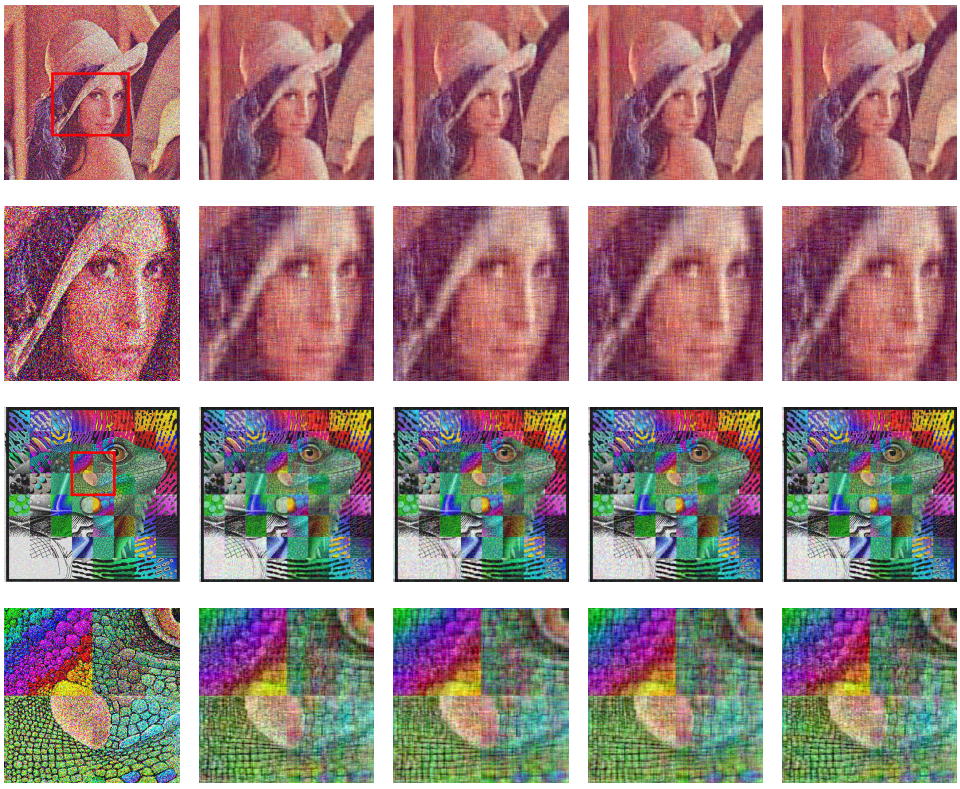


FIGURE 3. Experimental results of noised image reconstruction. First row: experimental results on Gaussian noised "Lenna" image of size 512 × 512 × 3. Second row: experimental results on Gaussian noised "Frymire" image of size 1024 × 1024 × 3. (a) Input images. (b) Images reconstructed via FCTPM. (c) Images reconstructed via TPM. (d) Images reconstructed via STPM. (e) Images reconstructed via TTPM.

TABLE 3. Experimental results on image reconstruction experiments.

Image		Lenna			Frymire		
Size		$512 \times 512 \times 3$			$1024 \times 1024 \times 3$		
Rank R		50			100		
Iteration K		15			15		
Metric		PSNR (dB)	SSIM	Proc.Time (sec)	PSNR (dB)	SSIM	Proc.Time (sec)
FCTPM (ours)	Noise free	26.964	0.955	**0.629**	**15.883**	**0.641**	**5.919**
	Gaussian Noise	**19.281**	**0.759**		**14.296**	**0.506**	
TPM [4]	Noise free	26.764	**0.956**	2.725	15.831	0.633	45.691
	Gaussian Noise	19.212	0.753		14.216	0.505	
STPM [1]	Noise free	26.956	**0.956**	43.846	15.694	0.629	520.738
	Gaussian Noise	19.205	0.754		14.192	0.503	
TTPM [13]	Noise free	26.926	0.955	3809.471	15.878	0.64	22832.067
	Gaussian Noise	19.29	0.758		14.289	**0.506**	

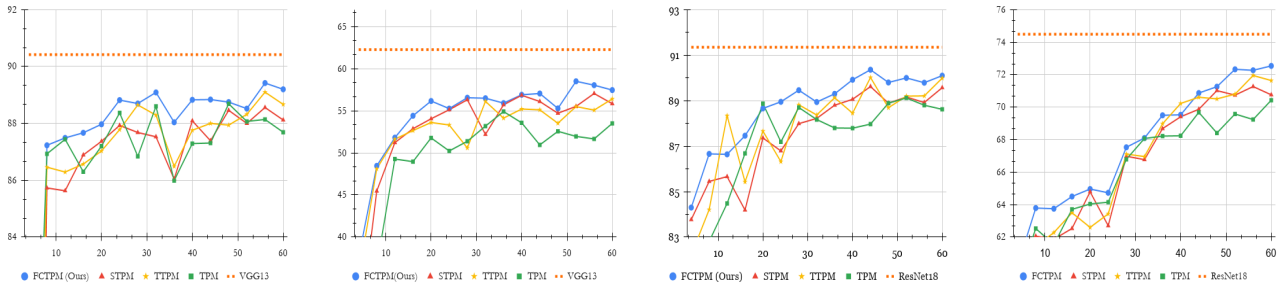
other algorithms in the noise free "Lenna" image experiments. However, in the clean "Frymire" image experiments, the FCTPM could effectively reproduce details such as texture and pattern.

From the results of the image reconstruction experiments summarized in Table. 3 and Figs. 2 and 3, it can be observed that the FCTPM has a low reconstruction performance loss compared to the other algorithms. In other words, the FCTPM has the highest robustness amongst all the other algorithms considered for comparison.

This proves that the FCTPM has the lowest computational cost while having a desirable reconstruction performance in noised images as compared to the other algorithms that were considered for comparison.

TABLE 4. Experimental results on CNN compression experiments.

Dataset	CIFAR10			CIFAR100		
Rank R	VBMF [51]			VBMF [51]		
Iteration K	15			20		
Metric	Accuracy (%)	# of parameters	FLOPS	Accuracy (%)	# of parameters	FLOPS
VGG13 [54]	90.42	9.42×10^6	2.29×10^8	62.26	9.42×10^6	2.29×10^8
FCTPM (ours)	**89.42 (-1.0)**			**58.5 (-3.76)**		
TPM [4]	88.69 (-1.73)	1.46×10^6	5.47×10^7	54.91 (-7.35)	2.07×10^6	1.32×10^8
STPM [1]	88.56 (-1.86)	(×6.45)	(×4.19)	56.41 (-5.85)	(×5.39)	(×4.2)
TTPM [13]	89.1 (-1.32)			57.05 (-5.21)		
ResNet18 [55]	91.35	1.12×10^7	5.57×10^8	74.49	1.12×10^7	5.57×10^8
FCTPM (ours)	**90.36 (-0.99)**			**72.54 (-2.04)**		
TPM [4]	89.12 (-1.23)	1.85×10^6	4.50×10^7	70.43 (-4.06)	1.88×10^6	1.15×10^8
STPM [1]	89.62 (-1.73)	(×5.12)	(×5.09)	71.27 (-3.22)	(×5.97)	(×4.83)
TTPM [13]	90.02 (-1.35)			71.95 (-2.54)		



(a) (b) (c) (d)

FIGURE 4. Accuracy of compressed CNN in fine-tuning, The x- and y-axes of all sub-figures are fine-tuning epochs and accuracy, respectively. (a) VGG13 on CIFAR10, (b) VGG13 on CIFAR100, (c) ResNet18 on CIFAR10, and (d) ResNet18 on CIFAR100.

C. CNN COMPRESSION EXPERIMENT

The FCTPM-based CNN compression experiment was performed to evaluate the proposed method's performance on a practical high-order principal component analysis. The motivation behind tensor decomposition for CNN compression is to search for a compact convolutional weight tensor, which is close to the original by removing redundant components [49], [50].

In [46], TPM has been used as a compression tool to decompose a weight tensor of convolutional layer $\mathcal{W} \in \mathbb{R}^{D\times D\times S\times T}$ into into three components, where $D \times D$ means the spatial window size, S and T are the number of channels of the input and output, respectively.

The original operation of convolutional layer in CNN can be represented by a linear mapping that inputs $\mathcal{X} \in \mathbb{R}^{H\times W\times S}$ into $\mathcal{Y} \in \mathbb{R}^{H'\times W'\times T}$ as follows:

$$\begin{aligned} \mathcal{Y}_{h',w',t} &= \sum_{i=1}^{D}\sum_{j=1}^{D}\sum_{s=1}^{S} \mathcal{W}_{i,j,s,t}\mathcal{X}_{h,w,s}, \\ h &= (h'-1)\triangle + i - P, \\ w &= (w'-1)\triangle + j - P, \end{aligned} \tag{37}$$

where, $\triangle$ is the stride and P is the size of the zero-padding. Therefore, the storage complexity and computational complexity of the original operation of above equation are equivalent to D^2ST and $D^2STW'H'$, respectively.

From the above equation, the rank-R CPD-based convolutional weight tensor $\mathcal{W}$ decomposition is as follows:

$$\mathcal{Y}_{h',w',t} = \sum_{r=1}^{R} \mathcal{U}^{(2)}_{1,1,r,t} \sum_{i=1}^{D}\sum_{j=1}^{D} \mathcal{D}_{i,j,1,r} \sum_{s=1}^{S} \mathcal{U}^{(1)}_{1,1,s,r}\mathcal{X}_{h,w,s}, \tag{38}$$

where $\mathcal{D}, \mathcal{U}^{(1)}, \mathcal{U}^{(2)}$ are the three factors of size $D\times D\times 1\times R$, $1\times 1\times S\times R$, and $1\times 1\times T\times R$, respectively. Because the spatial window size $D \times D$ is already small, *e.g.*, 3×3, they do not need to be decomposed.

In conclusion, the TPM-based convolutional layer compression changes the original convolutional layer to three sequentially connected layers which are a 1×1 convolutional dimension reduction layer; $\mathcal{U}^{(1)}$, a $D \times D$ depth-wise convolutional layer; $\mathcal{D}$, and a 1×1 convolutional dimension expansion layer; $\mathcal{U}^{(2)}$. The storage complexity and computational complexity of TPM-based compressed convolutional

layer are $SR + D^2R + TR$ and $SRWH + D^2RW'H' + TRW'H'$, respectively.

We obtained the experimental results by replacing the TPM in [46] with FCTPM, STPM, and TTPM. To fairly compare the proposed algorithm and the other competing algorithms, we select the rank R through the variational Bayesian matrix factorization (VBMF) [51], widely used in CNN compression methodologies [48], [49], [52], and fine-tune the compressed CNNs from each algorithm with an identical learning-rate schedule. The experiments are conducted on two representative image classification datasets, CIFAR100 [53] and CIFAR10 [53] and two representative CNN architectures, VGG13 [54] and ResNet18 [55].

Table 4 shows that the proposed algorithm achieved a lower accuracy degradation (the highest accuracy) than the other algorithms, include TPM, STPM, and TTPM. Note that the floating-point operations per second (FLOPS) is the computational cost. Because the rank selection algorithm determines the storage and computational complexities in tensor decomposition-based CNN compression, the compressed neural networks inevitably have the same compression ratio for the number of parameters and computational cost. Therefore, the experimental results show that the proposed algorithm has better high-order principal component analysis performance than the competing algorithms. The proposed algorithm produces more significant components than the TPM and its variants in every convolutional layer, regardless of both the scale of datasets and CNN architecture. Fig. 4 illustrates classification accuracy curves in every fine-tuning epochs. In almost all overall epochs, the proposed algorithm reached a higher accuracy than the other algorithms. This peculiarity implies that the FCTPM-based CNN compression has a faster convergence speed than the others, resulting from the stable decomposition property of the proposed algorithm.

VI. CONCLUSION

In this paper, a novel and efficient TPM called FCTPM is proposed, involving a combination of rank-1 TTD and the PM via the circulant updating method. The FCTPM inherits the advantages of both rank-1 TTD and PM. Therefore, the computational cost can decrease considerably because the FCTPM approximates the rank-1 tensor via auxiliary unfolding matrix decomposition. This property facilitates the use of the PM, which is a powerful representative matrix decomposition algorithm. Moreover, the proposed circulant updating algorithm is a major component in the FCTPM; it helps the factor vectors to globally converge on the solution. To show the superior performance of FCTPM, we conducted several experiments, including numerical simulations, image reconstruction, and CNN compression. In the numerical simulation and image reconstruction, it can be seen that the FCTPM produces better high-order principal component analysis performance than the other methods with the lowest processing times. The 7th-order tensor decomposition in the numerical simulation and the corrupted image reconstruction experiment shows that the proposed algorithm is a better tensor analysis tool than conventional algorithms regardless of the dimensions of the input tensor and noise corruption. In CNN compression experiments, the FCTPM produces a lower accuracy loss than TPM and its variants. From this experiment, it is verified that our method can be used in practical tensor analysis applications. In conclusion, we developed a rank-1 tensor approximation algorithm to understand high-order intrinsic key patterns in high-dimensional data with computational costs lower than those of the conventional TPM and its variants.

REFERENCES

[1] G. Allen, "Sparse higher-order principal components analysis," in *Artificial Intelligence and Statistics*. La Palma, Spain: PMLR, 2012. [Online]. Available: http://proceedings.mlr.press/v22/allen12.html

[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] R. A. Harshman, "Foundations of the PARAFAC procedure: Models and conditions for an 'explanatory' multimodal factor analysis," UCLA Work. Papers Phonetics, Univ. Microfilms, Ann Arbor, MI, USA, Tech. Rep. 16 and 10,085, 1970, pp. 1–84. [Online]. Available: https://www.psychology.uwo.ca/faculty/harshman/wpppfac0.pdf

[4] G. Golub and C. F. Van Loan, *Matrix Computations (Johns Hopkins Studies in Mathematical Sciences)*. Baltimore, MD, USA: Johns Hopkins Univ. Press, 1996.

[5] B. Chen, S. He, Z. Li, and S. Zhang, "Maximum block improvement and polynomial optimization," *SIAM J. Optim.*, vol. 22, no. 1, pp. 87–107, Jan. 2012.

[6] A. Cichocki, D. Mandic, L. De Lathauwer, G. Zhou, Q. Zhao, C. Caiafa, and H. A. Phan, "Tensor decompositions for signal processing applications: From two-way to multiway component analysis," *IEEE Signal Process. Mag.*, vol. 32, no. 2, pp. 145–163, Mar. 2015.

[7] R. V. Mises and H. Pollaczek-Geiringe, "Praktische Verfahren der Gleichungsauflösung," *ZAMM-J. Appl. Math. Mech./Zeitschrift für Angew. Mathematik und Mech.*, vol. 9, no. 1, pp. 58–77, 1929.

[8] A. H. Bentbib and A. Kanber, "Block power method for SVD decomposition," *Analele Universitatii 'Ovidius' Constanta - Seria Matematica*, vol. 23, no. 2, pp. 45–58, Jun. 2015.

[9] A. Anandkumar, R. Ge, and M. Janzamin, "Analyzing tensor power method dynamics in overcomplete regime," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 752–791, 2017.

[10] A. Anandkumar, R. Ge, D. Hsu, S. M. Kakade, and M. Telgarsky, "Tensor decompositions for learning latent variable models," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 2773–2832, 2014.

[11] D. Cartwright and B. Sturmfels, "The number of eigenvalues of a tensor," *Linear Algebra its Appl.*, vol. 438, no. 2, pp. 942–952, Jan. 2013.

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

[13] W. W. Sun, J. Lu, H. Liu, and G. Cheng, "Provable sparse tensor decomposition," *J. Roy. Stat. Soc., B (Stat. Methodol.)*, vol. 79, no. 3, pp. 899–916, Jun. 2017.

[14] G. Schwarz, "Estimating the dimension of a model," *Ann. Statist.*, vol. 6, no. 2, pp. 461–464, Mar. 1978.

[15] M. J. Mohlenkamp, "Musings on multilinear fitting," *Linear Algebra Appl.*, vol. 438, no. 2, pp. 834–852, Jan. 2013.

[16] M. Razaviyayn, M. Hong, and Z.-Q. Luo, "A unified convergence analysis of block successive minimization methods for nonsmooth optimization," *SIAM J. Optim.*, vol. 23, no. 2, pp. 1126–1153, Jan. 2013.

[17] A. Smilde, R. Bro, and P. Geladi, *Multi-Way Analysis: Applications in the Chemical Sciences*. Hoboken, NJ, USA: Wiley, 2005.

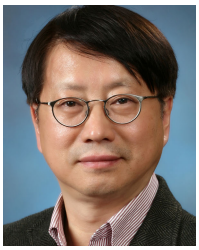
[18] P. M. Kroonenberg, *Applied Multiway Data Analysis*, vol. 702. Hoboken, NJ, USA: Wiley, 2008.

[19] P. Comon and Christian Jutten, eds. *Handbook of Blind Source Separation: Independent Component Analysis and Applications*. New York, NY, USA: Academic, 2010.

[20] A. Cichocki, *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-Way Data Analysis and Blind Source Separation*. Hoboken, NJ, USA: Wiley, 2009.

[21] J. M. Landsberg, "Tensors: Geometry and applications," *Represent. Theory*, vol. 381, no. 402, p. 3, 2012.
[22] W. Hackbusch, *Tensor Spaces and Numerical Tensor Calculus*, vol. 42. Berlin, Germany: Springer, 2012.
[23] E. Acar and B. Yener, "Unsupervised multiway data analysis: A literature survey," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 1, pp. 6–20, Jan. 2009.
[24] P. Comon, X. Luciani, and A. L. F. de Almeida, "Tensor decompositions, alternating least squares and other tales," *J. Chemometrics*, vol. 23, nos. 7–8, pp. 393–405, Jul. 2009.
[25] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, "A survey of multilinear subspace learning for tensor data," *Pattern Recognit.*, vol. 44, no. 7, pp. 1540–1551, Jul. 2011.
[26] M. Mørup, "Applications of tensor (multiway array) factorizations and decompositions in data mining," *WIREs Data Mining Knowl. Discovery*, vol. 1, no. 1, pp. 24–40, Jan. 2011.
[27] B. N. Khoromskij, "Tensors-structured numerical methods in scientific computing: Survey on recent advances," *Chemometric Intell. Lab. Syst.*, vol. 110, no. 1, pp. 1–19, Jan. 2012.
[28] 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 Trans. Signal Process.*, vol. 65, no. 13, pp. 3551–3582, Jul. 2017.
[29] I. V. Oseledets, "Tensor-train decomposition," *SIAM J. Sci. Comput.*, vol. 33, no. 5, pp. 2295–2317, Jan. 2011.
[30] K. Batselier, H. Liu, and N. Wong, "A constructive algorithm for decomposing a tensor into a finite sum of orthonormal Rank-1 terms," *SIAM J. Matrix Anal. Appl.*, vol. 36, no. 3, pp. 1315–1337, Jan. 2015.
[31] Y. Zniyed, R. Boyer, A. L. F. de Almeida, and G. Favier, "High-order tensor estimation via trains of coupled third-order CP and tucker decompositions," *Linear Algebra Appl.*, vol. 588, pp. 304–337, Mar. 2020.
[32] A.-H. Phan, P. Tichavsky, and A. Cichocki, "Fast alternating LS algorithms for high order CANDECOMP/PARAFAC tensor factorizations," *IEEE Trans. Signal Process.*, vol. 61, no. 19, pp. 4834–4846, Oct. 2013.
[33] A.-H. Phan, P. Tichavsky, and A. Cichocki, "CANDECOMP/PARAFAC decomposition of high-order tensors through tensor reshaping," *IEEE Trans. Signal Process.*, vol. 61, no. 19, pp. 4847–4860, Oct. 2013.
[34] R. A. Harshman, "Foundations of the PARAFAC procedure: Models and conditions for an 'explanatory' multimodal factor analysis," in *Proc. UCLA Work. Papers Phonetics*, vol. 16, 1970, pp. 1–84.
[35] C. A. Andersson and R. Bro, "The N-way toolbox for MATLAB," *Chemometrics Intell. Lab. Syst.*, vol. 52, no. 1, pp. 1–4, 2000.
[36] M. Rajih, P. Comon, and R. A. Harshman, "Enhanced line search: A novel method to accelerate PARAFAC," *SIAM J. Matrix Anal. Appl.*, vol. 30, no. 3, pp. 1128–1147, Jan. 2008.
[37] G. Tomasi, "Practical and computational aspects in chemometric data analysis," Ph.D. dissertation, Dept. Food Sci., Roy. Vet. Agricult. Univ., Frederiksberg, Denmark, 2006.
[38] Y. Chen, D. Han, and L. Qi, "New ALS methods with extrapolating search directions and optimal step size for complex-valued tensor decompositions," *IEEE Trans. Signal Process.*, vol. 59, no. 12, pp. 5888–5898, Dec. 2011.
[39] H. A. L. Kiers, "A three-step algorithm for CANDECOMP/PARAFAC analysis of large data sets with multicollinearity," *J. Chemometrics*, vol. 12, no. 3, pp. 155–171, May 1998.
[40] P. Paatero, "A weighted non-negative least squares algorithm for three-way 'PARAFAC' factor analysis," *Chemometrics Intell. Lab. Syst.*, vol. 38, no. 2, pp. 223–242, 1997.
[41] P. Tichavsky and Z. Koldovsky, "Simultaneous search for all modes in multilinear models," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Mar. 2010, pp. 4114–4117.
[42] A.-H. Phan, P. Tichavský, and A. Cichocki, "Low complexity damped gauss-Newton algorithms for CANDECOMP/PARAFAC," *SIAM J. Matrix Anal. Appl.*, vol. 34, no. 1, pp. 126–147, Jan. 2013.
[43] A. H. Phan, P. Tichavsky, and A. Cichocki, "Fast damped gauss-Newton algorithm for sparse and nonnegative tensor factorization," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2011, pp. 1988–1991.
[44] J. Kossaifi, Y. Panagakis, A. Anandkumar, and M. Pantic, "Tensorly: Tensor learning in python," *J. Mach. Learn. Res.* vol. 20, no. 1, pp. 925–930, 2019.
[45] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.
[46] M. Astrid and S.-I. Lee, "CP-decomposition with tensor power method for convolutional neural networks compression," in *Proc. IEEE Int. Conf. Big Data Smart Comput. (BigComp)*, Feb. 2017, pp. 115–118.
[47] V. Lebedev, Y. Ganin, M. Rakhuba, I. Oseledets, and V. Lempitsky, "Speeding-up convolutional neural networks using fine-tuned CP-decomposition," 2014, *arXiv:1412.6553*. [Online]. Available: http://arxiv.org/abs/1412.6553
[48] Y.-D. Kim, E. Park, S. Yoo, T. Choi, L. Yang, and D. Shin, "Compression of deep convolutional neural networks for fast and low power mobile applications," 2015, *arXiv:1511.06530*. [Online]. Available: http://arxiv.org/abs/1511.06530
[49] J. Cheng, P.-S. Wang, G. Li, Q.-H. Hu, and H.-Q. Lu, "Recent advances in efficient computation of deep convolutional neural networks," *Frontiers Inf. Technol. Electron. Eng.*, vol. 19, no. 1, pp. 64–77, Jan. 2018.
[50] T. Kim, J. Lee, and Y. Choe, "Bayesian optimization-based global optimal rank selection for compression of convolutional neural networks," *IEEE Access*, vol. 8, pp. 17605–17618, 2020.
[51] S. Nakajima, M. Sugiyama, S. D. Babacan, and R. Tomioka, "Global analytic solution of fully-observed variational Bayesian matrix factorization," *J. Mach. Learn. Res.* vol. 14, pp. 1–37, Jan. 2013.
[52] M. Astrid, S.-I. Lee, and B.-S. Seo, "Rank selection of CP-decomposed convolutional layers with variational Bayesian matrix factorization," in *Proc. 20th Int. Conf. Adv. Commun. Technol. (ICACT)*, Feb. 2018.
[53] A. Krizhevsky, "Learning multiple layers of features from tiny images," M.S. thesis, Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, 2009.
[54] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: http://arxiv.org/abs/1409.1556
[55] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

TAEHYEON KIM (Graduate Student Member, IEEE) received the B.S. degree in electronics from Kangwon National University, Chuncheon, South Korea, in 2017. He is currently pursuing the Ph.D. degree in electrical and electronic engineering with Yonsei University, Seoul, South Korea. His research interests include computer vision, neural network compression, tensor decomposition, and Bayesian optimization.



YOONSIK CHOE (Senior Member, IEEE) received the B.S. degree in electrical engineering from Yonsei University, Seoul, South Korea, in 1979, the M.S.E.E. degree in systems engineering from Case Western Reserve University, Cleveland, OH, USA, in 1984, the M.S. degree in electrical engineering from Pennsylvania State University, State College, PA, USA, in 1987, and the Ph.D. degree in electrical engineering from Purdue University, West Lafayette, IN, USA, in 1990. From 1990 to 1993, he was a Principal Research Staff with the Industrial Electronics Research Center, Hyundai Electronics Company Ltd. Since 1993, he has been with the Department of Electrical and Electronic Engineering, Yonsei University, where he is currently a Professor. His research interests include video coding, video communication, statistical signal processing, and digital image processing.