HIGH-ORDER TENSOR COMPLETION FOR DATA RECOVERY VIA SPARSE TENSOR-TRAIN OPTIMIZATION

Longhao Yuan$^{1,2}$, Qibin Zhao$^{2,3}$ and Jianting Cao$^1$

Graduate School of Engineering, Saitama Institute of Technology, Japan
Tensor Learning Unit, RIKEN Center for Advanced Intelligence Project (AIP), Japan
School of Automation, Guangdong University of Technology, China

ABSTRACT

In this paper, we aim at the problem of tensor data completion. Tensor-train decomposition is adopted because of its powerful representation performance and tensor order linear scalability. We propose an algorithm named STTO (Sparse Tensor-train Optimization) which considers incomplete data as sparse tensor and uses first-order optimization method to find the factors of tensor-train decomposition. Our algorithm is shown to perform well in simulation experiments at both low-order cases and high-order cases. We also employ a tensorization method to transform data to a higher-order to enhance the performance of our algorithm. The image recovery experiment results in various cases manifest that our method outperforms other completion algorithms. Especially when the missing rate is very high, e.g. 90% to 99%, our method can achieve much better performance than other state-of-the-art methods.

Index Terms— incomplete data, tensor-train decomposition, high-order tensorization, optimization

1. INTRODUCTION

Tensors are multi-dimensional arrays with high-order generation of vectors and matrices [11]. Most of the real world data like color images, videos, multichannel encephalography (EEG) signals, etc. are more than two dimensions. Tensor data representation can keep the original format of data, which is good for retaining high dimensional structure and adjacent relation information of data. Due to the flexibility and highly compressive of tensor decomposition, in recent decades, many tensor methodologies have been proposed in various fields such as image and video completion [2, 3], brain computer interface [4], signal processing [5, 6], etc. The main concept of solving tensor completion problem is that we use the observed entries of incomplete data to find the tensor decomposition factors which contain the latent features of the data, then we use the strong feature representation ability of tensor decomposition to reconstruct the missing entries. The most studied and popular decomposition format in recent years are CANDECOMP/PARAFAC (CP) decomposition [7, 8] and Tucker decomposition [9]. CP decomposition and Tucker decomposition have been applied to many data completion methods. CP weighted optimization (CP-WOPT) in paper [2] builds objective function by the difference of weighted approximated tensor and observed tensor, then it uses optimization method to find the optimal CP factor matrices from the observed data. Bayesian CP factorization [3] conducts Bayesian probabilistic model to automatically determine the rank of CP tensor and find the best CP factor matrices at the same time. The method in [10] recovers low-n-rank tensor data with its convex relaxation by alternating direction method of multipliers (ADM). Paper [11] uses low-n-rank Tucker tensor completion method and the experiment shows better results than other nuclear norm minimization methods.

However, due to the nature limitation of CP and Tucker, though they can reach a relatively high accuracy in low-order tensors, when it comes to high-order tensors and high missing rate of data, the performance of these two decomposition methods will decrease rapidly. Tensor-train (TT) [12] which is free from ‘the curse of dimensionality’ is a better format to process high dimension data. The works in our paper are concluded as below: (a) We propose an algorithm named STTO (Sparse Tensor-train Optimization) which considers incomplete data as sparse tensor and optimize all the factors of tensor-train decomposition in sparse method, the computational cost is much improved. The tensor decomposition factors are used to recover the missing entries. (b) We conduct simulation experiments to compare our algorithm with other state-of-the-art algorithms in four different dimensions. (c) We provide a data dimension ascending strategy for image data which can increase the performance of our algorithm. Also it is a good strategy to process image data with non-random missing like whole row missing and block missing. (d) We carry out several real world data experiments to compare our algorithm with other state-of-the-art algorithms. The experiment results in simulation data and image data show that our method...
outperforms other state-of-the-art approaches.

2. NOTATIONS AND TENSOR-TRAIN DECOMPOSITION

2.1. Notations

Notations in [11] is employed in this paper. Scalars are denoted by normal lowercase letters, e.g., \( x \), and vectors are denoted by boldface lowercase letters, e.g., \( \mathbf{x} \). Matrices are denoted by boldface capital letters, e.g., \( \mathbf{X} \). Tensors of order \( N \geq 3 \) are denoted by Euler script letters, e.g., \( \mathcal{X} \). \( \mathcal{X}^{(n)} \) denotes the \( n \)-th matrix of a matrix sequence, and the representations of vector and tensor sequence is denoted by the same way. When the tensor \( \mathcal{X} \) is in the space of \( \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N} \), the \((i_1, i_2, \ldots, i_N)\)-th element of \( \mathcal{X} \) is denoted by \( x_{i_1i_2\cdots i_N} \) or \( \mathcal{X}(i_1, i_2, \ldots, i_N) \).

2.2. Tensor-train Decomposition

The biggest advantage of tensor-train decomposition is that no matter how high the dimension of a tensor data is, it decomposes the tensor into a sequence of three-way tensors. So the amount of model parameters will not grow exponentially by the increase of data dimension. Tensor-train decomposition is to decompose a tensor into a sequence of three-way core tensors. In particular, the TT decomposition of a tensor \( \mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N} \) can be expressed as follow:

\[
\mathcal{X} = \ll \mathcal{G}^{(1)}, \mathcal{G}^{(2)}, \ldots, \mathcal{G}^{(N)} \rr,
\]

where \( \mathcal{G}^{(1)}, \mathcal{G}^{(2)}, \ldots, \mathcal{G}^{(N)} \) is a sequence of three-way core tensors with size of \( r_0 \times I_1 \times r_1, r_1 \times I_2 \times r_2, \ldots, r_{N-1} \times I_N \times r_N \). \( r_0 = r_N = 1 \), so \( \mathcal{G}^{(1)} \) and \( \mathcal{G}^{(N)} \) can also be considered as two matrices. The sequence \( \{r_0, r_1, r_2, \ldots, r_{N-1}, r_N\} \) is named TT-rank which limits the size of every core tensor. Furthermore, Each element of tensor \( \mathcal{X} \) can be represented by core tensors as follow:

\[
x_{i_1i_2\cdots i_N} = \mathcal{G}^{(1)}_{i_1} \times \mathcal{G}^{(2)}_{i_2} \times \cdots \times \mathcal{G}^{(N)}_{i_N},
\]

where \( \mathcal{G}^{(n)}_{i_n} \) is the \( i_n \)-th slice of the \( n \)-th core tensor with the size of \( r_{n-1} \times r_{n} \). \( i_n = 1, 2, \ldots, I_n \). \( \mathcal{G}^{(1)}_{i_1} \) and \( \mathcal{G}^{(N)}_{i_N} \) are the matrices extracted from first and last core tensor, so they can be considered as two vectors.

3. SPARSE TENSOR-TRAIN OPTIMIZATION

3.1. Our Previous Work

In our previous work [13], we proposed an algorithm called TT-WOPT (Tensor-train Weighted OPTimization) which achieves high performance in data completion task. However, when calculating gradients of every core tensor w.r.t. objective function, TT-WOPT considers all the missing entries of data as zero, so it computes the whole tensor scale for every iteration. If the scale of data is huge and missing rate is very high, TT-WOPT will waste much computational storage and be less effective as it computes the whole scale tensor which most of the data is zero and only a small percentage of data entries is useful.

3.2. STTO

In order to solve the problems TT-WOPT has, STTO (Sparse Tensor-train Optimization) which only uses observed entries to compute gradient of every core tensor is proposed. Consider the number of all the observed entries is \( M \), and define the index of the \( m \)-th observed entry as \( \{i^m_1, i^m_2, \ldots, i^m_N\} \), so we have:

\[
x_m = \mathcal{X}(i^m_1, i^m_2, \ldots, i^m_N).
\]

Define \( y_m \) is the \( m \)-th observed entries calculated by equation (2) w.r.t. indices \( \{i^m_1, i^m_2, \ldots, i^m_N\}, m = 1, \ldots, M \), so \( y_m \) can be written as:

\[
y_m = \mathcal{G}^{(1)}_{i^m_1} \times \mathcal{G}^{(2)}_{i^m_2} \times \cdots \times \mathcal{G}^{(N)}_{i^m_N},
\]

For one observed entry \( x_m \), the objective function is:

\[
f(G^{(1)}_{i^m_1}, G^{(2)}_{i^m_2}, \ldots, G^{(N)}_{i^m_N}) = \frac{1}{2} \| (y_m - x_m) \|^2. \tag{5}
\]

So by equation (4) and (5), for \( n = 1, 2, \ldots, N \), the partial derivatives of every used slice \( G^{(n)}_{i^m_n} \) of this entry is calculated as:

\[
\frac{\partial f}{\partial G^{(n)}_{i^m_n}} = (y_m - x_m)(G^{>n}_{i^m_n} \times (G^{<n}_{i^m_n})^T), \tag{6}
\]

where

\[
G^{>n}_{i^m_n} = G^{(n+1)}_{i^m_{n+1}} \times G^{(n+2)}_{i^m_{n+2}} \times \cdots \times G^{(N)}_{i^m_N}, \tag{7}
\]

\[
G^{<n}_{i^m_n} = G^{(1)}_{i^m_1} \times G^{(2)}_{i^m_2} \times \cdots \times G^{(n-1)}_{i^m_{n-1}}. \tag{8}
\]

When all the observed entries are considered, we define \( v^{(n)} \) as \( \{G^{(n)}_{i^m_1}, G^{(n)}_{i^m_2}, \ldots, G^{(n)}_{i^m_N}\}^T \) which contains the slices extracted from the same core tensor \( \mathcal{G}^{(n)} \) according to the indices from the same dimension, then \( v \) can be represented by:

\[
y = v^{(1)} \ast v^{(2)} \ast \cdots \ast v^{(N)}. \tag{9}
\]

All the observed entries \( x_m \) are restored in vector \( x \) of size \( M \times 1 \). Then the optimization objective function of all missing entries can be formulated by:

\[
f(v^{(1)}, v^{(2)}, \ldots, v^{(N)}) = \frac{1}{2} \| (y - x) \|^2. \tag{10}
\]

The vector gradient of the \( n \)-th core is:

\[
\frac{\partial f}{\partial v^{(n)}} = (y - x)(v^{>n} \ast (v^{<n})^T). \tag{11}
\]
The total gradient of every slice $G^{(n)}_m$ of every core tensor is the accumulation of the slice gradients in equation (6) with the same index, that is:

$$\frac{\partial f}{\partial G_j^{(n)}} = \sum_{m=1}^{M} (y_m - x_m)(G^{>n}_m \times (G^{<n}_{m})^T), \quad (12)$$

$j = 1, 2, \cdots, I_n$. After all the gradients of every slice of core tensors are obtained, any first-order optimization method can be applied to the STTO algorithm. In this paper, we apply a state-of-the-art algorithm named Adaptive Moment Estimation (Adam) from paper [14] as our gradient descent method. And the optimization stopping conditions are the objective function value error or the maximum iteration. The whole process of STTO is summarized in Algorithm1.

\textbf{Algorithm 1} Sparse Tensor-train Optimization (STTO)

\textbf{Input}: an $N$-way incomplete sparse tensor $X$.

\textbf{Parameters}: error $\varepsilon$, maximum iteration $K_{iter}$, TT - ranks $r$.

\textbf{Initialization}: core tensors $G^{(1)}, G^{(2)}, \ldots, G^{(N)}$ of tensor $X$.

1. Let $x$ be the length-$M$ vector contains only observed entries of $X$ w.r.t. indices $\{i_1^n, i_2^n, \ldots, i_M^n\}$, $m = 1, 2, \ldots, M$.

\textbf{while}

2. Compute $y = \psi^{(1)} * \psi^{(2)} * \cdots * \psi^{(N)}$.

3. Compute objective function value: $f = \frac{1}{2} \|y\|^2 - y^T x + \frac{1}{2} \|x\|^2$.

4. Compute all $\frac{\partial f}{\partial G_j^{(n)}} = \sum_{m=1}^{M} (y_m - x_m)(G^{>n}_m \times (G^{<n}_{m})^T)$, $j = 1, 2, \ldots, I_n$.

5. Use gradient descent method to update $G^{(1)}, G^{(2)}, \ldots, G^{(N)}$.

\textbf{Return} updated core tensors $G^{(1)}, G^{(2)}, \ldots, G^{(N)}$.

\textbf{if} $f < \varepsilon$ or iteration reaches $K_{iter}$, \textbf{break}

\textbf{end while}

\section{Experiments}

In this section, our proposed STTO is compared with two state-of-the-art algorithms: CP weighted optimization (CP-WOPT) [2] and Fully Bayesian CP (FBCP) [3]. Simulation experiments, color image data experiments are conducted to validate the effectiveness of our algorithm. In addition, we provide a data dimension ascending method or called tensorization method to transform data to a higher dimension. This method can enhance the structure relation information of data and improve the performance of our algorithm. For evaluation indices, we use RSE (Relative Square Error) for simulation data and image data, and PSNR (Peak Signal-to-noise Ratio) is used to measure the quality of reconstructed image data.

\subsection{Simulations}

The three tensor completion algorithms use various tensor decomposition methods so it is not proper to use synthetic data generated from one decomposition method as the simulation data. Therefore, we consider to use values produced from a highly oscillating function: $f(x) = \sin \frac{x}{2} \cos (x^2)$ [15] as simulation data, which is expected to be well approximated by all the tensor completion algorithms. The four different data structures are $26 \times 26 \times 26$ (3D), $7 \times 7 \times 7 \times 7 \times 7$ (5D), $5 \times 5 \times 5 \times 5 \times 5 \times 5 \times 5 \times 5 \times 5 \times 5$ (7D), $3 \times 3 \times 3 \times 3 \times 3 \times 3 \times 3 \times 3 \times 3 \times 3 \times 3 \times 3 \times 3 \times 3 \times 3 \times 3 \times 3 \times 3$ (9D). The TT-ranks and CP-ranks of the four simulation data are set to make the number of model parameters of the three algorithms as close as possible respectively.

From Figure 1.1, we can see, our method performs best among the three algorithms almost in every situation. Especially when the dimension of data is increase, our algorithm can maintain the completion accuracy while the performance of the other two algorithms falls quickly.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig.png}
\caption{RSE comparison results of three algorithms under four data dimension. Missing rate is tested from 0% to 90%}
\end{figure}

\subsection{Image Data Completion}

\subsubsection{Data dimension ascending method}

From the simulation results we can see STTO can perform well in high-order cases, so we provide the below method to transform color image data to higher-order to enhance the performance of our algorithm. By applying this method, the structure relation information of data can be enhanced. The original size of every image data is $256 \times 256 \times 3$. First it is reshaped to a seventeen-way tensor of size $2 \times 2 \times \cdots \times 2 \times 3$ and permute the tensor according to the order of $\{1 \ 9 \ 2 \ 10 \ 3 \ 11 \ 4 \ 12 \ 5 \ 13 \ 6 \ 14 \ 7 \ 15 \ 8 \ 16 \ 17\}$. Then we reshape the tensor to a nine-way tensor of size $4 \times 4 \times \cdots \times 4 \times 3$. This nine-way tensor is a better structure of the image data. The first-order of the transformed tensor...
contains the data of a $2 \times 2$ pixel block of the image and the following orders of the tensor describe the expanding pixel blocks of the image. This tensorization method is applied to STTO in all of the following image experiments.

4.2.2. Random missing

We first use one picture to see the best performance of all the algorithms in random missing cases with different missing rate. Because all the proposed completion methods are good at low missing rate cases, we only compare the three algorithms in high missing rate situations. TT-ranks are set to 16 and CP-ranks are set to 25 according to experience. The visualized experiment results in Figure 2 show that our STTO algorithm outperforms other algorithms distinctly. Particularly, when the missing rate reaches 98% and 99%, our algorithm can recover the image well while other algorithms fail totally.

4.2.3. Irregular missing

In this experiment, images with whole row missing and block missing are transformed to nine-way tensor by our dimension ascending method and sent to STTO algorithm. The other two algorithms use original three way tensor form because they perform better in low-order data form. The visualized results of Figure 3 show that STTO algorithm with data dimension ascending can recover images with whole row missing and block missing well. And the other two algorithms that use original data structure cannot recover the missing area properly. In addition, values of RSE and PSNR from Table 1 show the advantage of our algorithm over the other two algorithms.

![Fig. 2. Visualizing results of image inpainting performance of three different algorithms under five different missing rates of 85%, 90%, 95%, 98% and 99%.](image)

![Fig. 3. Visualizing results of image inpainting performance of three different algorithms under two special missing conditions for three images.](image)

| Table 1. Comparison of the inpainting performance (RSE and PSNR) of three algorithms under two special missing conditions for three images. |
|---|---|---|---|---|
| image | row missing | block missing |
| | | lena | peppers | sailboat | lena | peppers | sailboat |
| STTO | RSE | 0.1138 | 12.60 | 19.93 | 0.1323 | 19.34 | 0.1949 |
| PSNR | 24.00 | 20.80 | 19.93 | 22.69 | 19.34 | 19.08 |
| CP-WOPT | RSE | 0.1503 | 10.85 | 0.38 | 0.1946 | 10.85 | 0.2252 |
| PSNR | 10.86 | 10.34 | 10.18 | 20.61 | 18.27 | 19.00 |
| FBCP | RSE | 0.5503 | 0.5594 | 0.5586 | 0.3856 | 0.4543 | 0.4167 |
| PSNR | 10.46 | 10.58 | 10.18 | 11.97 | 10.99 | 11.36 |

5. CONCLUSIONS

In this paper, we first elaborate the basis of tensor and tensor-train decomposition. Then STTO algorithm which is a gradient-based first-order optimization method is proposed. It uses sparse tensor data to find the factors of tensor-train decomposition and recovers the missing entries. From the simulation experiments, we can see our algorithm outperforms the other state-of-the-art methods in both low-order cases and high-order cases, especially when the order of data is high. And image completion experiments prove that STTO algorithm with our dimension ascending method can achieve a high performance with very high missing rate. The prominent results on irregular missing cases also show advantages of our method.

The good performance of our method also demonstrates that tensor-train decomposition with high order tensorizations can achieve high compressive and representation abilities. Furthermore, it should be noted that the accuracy of tensor-train decomposition is sensitive to the selection of TT-ranks. Hence, we will study on how to optimize tensor decomposition factors and TT-ranks simultaneously in our future work.
6. REFERENCES

[1] Tamara G Kolda and Brett W Bader, “Tensor decompositions and applications,” SIAM Review, vol. 51, no. 3, pp. 455–500, 2009.

[2] Evrim Acar, Daniel M Dunlavy, Tamara G Kolda, and Morten Morup, “Scalable tensor factorizations for incomplete data,” Chemometrics and Intelligent Laboratory Systems, vol. 106, no. 1, pp. 41–56, 2011.

[3] Qibin Zhao, Liqing Zhang, and Andrzej Cichocki, “Bayesian cp factorization of incomplete tensors with automatic rank determination,” IEEE transactions on pattern analysis and machine intelligence, vol. 37, no. 9, pp. 1751–1763, 2015.

[4] J Mocks, “Topographic components model for event-related potentials and some biophysical considerations,” IEEE transactions on biomedical engineering, vol. 35, no. 6, pp. 482–484, 1988.

[5] Lieven De Lathauwer and Josaphine Castaing, “Blind identification of underdetermined mixtures by simultaneous matrix diagonalization,” IEEE Transactions on Signal Processing, vol. 56, no. 3, pp. 1096–1105, 2008.

[6] Damien Muti and Salah Bourennane, “Multidimensional filtering based on a tensor approach,” Signal Processing, vol. 85, no. 12, pp. 2338–2353, 2005.

[7] Richard A Harshman, “Foundations of the parafac procedure: Models and conditions for an” explanatory” multi-modal factor analysis,” 1970.

[8] Mikael Sorensen, Lieven De Lathauwer, Pierre Comon, Sylvie Icart, and L Deneire, “Canonical polyadic decomposition with orthogonality constraints,” SIAM Journal on Matrix Analysis and Applications, 2012.

[9] Ledyard R Tucker, “Some mathematical notes on three-mode factor analysis,” Psychometrika, vol. 31, no. 3, pp. 279–311, 1966.

[10] Silvia Gandy, Benjamin Recht, and Isao Yamada, “Tensor completion and low-n-rank tensor recovery via convex optimization,” Inverse Problems, vol. 27, no. 2, pp. 025010, 2011.

[11] Marko Filipović and Ante Jukić, “Tucker factorization with missing data with application to low-n-rank tensor completion,” Multidimensional systems and signal processing, vol. 26, no. 3, pp. 677–692, 2015.

[12] Ivan V Oseledets, “Tensor-train decomposition,” SIAM Journal on Scientific Computing, vol. 33, no. 5, pp. 2295–2317, 2011.

[13] Longhao Yuan and Qibin Zhao, “Completion of high order tensor data with missing entries via tensor-train decomposition,” arXiv preprint arXiv:1709.02641, 2017.

[14] Sebastian Ruder, “An overview of gradient descent optimization algorithms,” arXiv preprint arXiv:1609.04747, 2016.

[15] Boris N Khoromskij, “Tensor numerical methods for multidimensional pdes: theoretical analysis and initial applications,” ESAIM: Proceedings and Surveys, vol. 48, pp. 1–28, 2015.