This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Additive Tensor Decomposition Considering Structural Data Information

Shancong Mou, Andi Wang, Chuck Zhang, and Jianjun Shi

***Abstract*— Tensor data with rich structural information become increasingly important in process modeling, monitoring, and diagnosis in manufacturing medical and other applications. Here structural information is referred to the information of tensor components such as sparsity, smoothness, low-rank, and piecewise constancy. To reveal useful information from tensor data, we propose to decompose the tensor into the summation of multiple components based on their different structural information. In this article, we provide a new definition of structural information in tensor data. We then propose an additive tensor decomposition (ATD) framework to extract useful information from tensor data. This framework specifies a high dimensional optimization problem to obtain the components with distinct structural information. An alternating direction method of multipliers (ADMM) algorithm is proposed to solve it, which is highly parallelable and thus suitable for the proposed optimization problem. Two simulation examples and a real case study in medical image analysis illustrate the versatility and effectiveness of the ATD framework.**

***Note to Practitioners*—This article was motivated by a real case in medical imaging: extracting aortic valve calcification (AVC) regions from the tensor data obtained from computed tomography (CT) image series of the aortic region. The main objective is to decompose image series into multiple components corresponding to tissues, calcium deposition, and error. Similar needs are pervasive in other medical image analysis applications as well as the image-based modeling, monitoring, and diagnosis of industrial processes and systems. Existing methods fail to incorporate a detailed description of the properties of image series that reflect the physical understanding of the system in both the spatial and temporal domains. In this article, we provide a systematic description of the properties of image series and use them to develop a decomposition framework. It is applicable to various applications and can generate more accurate and interpretable results.**

***Index Terms*— Alternating direction method of multipliers (ADMM) algorithm, structural information, tensor decomposition.**

Manuscript received March 24, 2021; accepted May 17, 2021. This article was recommended for publication by Associate Editor Z. Kong and Editor X. Xie upon evaluation of the reviewers' comments. *(Corresponding author: Jianjun Shi.)*

Shancong Mou, Andi Wang, and Jianjun Shi are with the H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: shancong.mou@gatech.edu; andi.wang@gatech.edu; jianjun.shi@isye.gatech.edu).

Chuck Zhang is with the H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA, and also with the Georgia Tech Manufacturing Institute, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: chuck.zhang@gatech.edu).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TASE.2021.3096964.

Digital Object Identifier 10.1109/TASE.2021.3096964

I. Introduction

With the recent advancement of sensing technology, tensor data with rich structural information are acquired for process modeling, monitoring, and diagnosis [1]–[3]. For example, tensor data can represent multiple images obtained from computed tomography (CT) scanning to illustrate different cross sections of an organ [4]–[6]. In this example, the tensor is of order 3: the first mode represents multiple slices of images along the scanning direction, and the second and third modes define a cross-sectional image (see Fig. 1).

Useful information for an application is usually deeply buried in tensor data. To reveal this information, we usually need to decompose the tensor data into a summation of multiple tensor components with the same size, based on domain interpretations. For example, the tensor representing the data shown in Fig. 1 can be decomposed into three tensor components, representing the background of blood and aorta, aortic valve calcification (AVC) regions, and measurement errors. Generally, one of the tensor components represents quality issues of interest. In the example of the CT images, the component representing AVC regions is related to the likelihood of the paravalvular regurgitation (PVR) symptom after the transcatheter aortic valve replacement (TAVR) surgery, a common minimum invasive surgical procedure for treating aortic stenosis [6]. Based on the decomposed components, a prototype for patients' aortic root anatomies can be fabricated using multi-material 3-D printing for surgical planning to prevent PVR [5]. We refer to the decomposition of one tensor into a summation of multiple tensors as additive tensor decomposition (ATD), to differentiate it from existing low-rank tensor decomposition approaches like Tucker and CP decompositions [7].

The decomposition of the tensor data can be performed based on different structural information of the components on various modes or their combinations, driven by the physical knowledge of the process. In the CT example, we not only need to consider the spatial information in each image, such as every slice of the background tensor is smooth, the AVC regions are sparse, and measurement errors are usually small independent random values on each pixel. We also need to consider the temporal evolvement between images along the first tensor mode, such as gradual change of the component representing AVC regions across multiple slices. The structural difference between the components defines how the decomposition should be conducted. From the examples in Sections III and IV, we illustrate that the applications of

1545-5955 © 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on August 03,2021 at 02:18:39 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

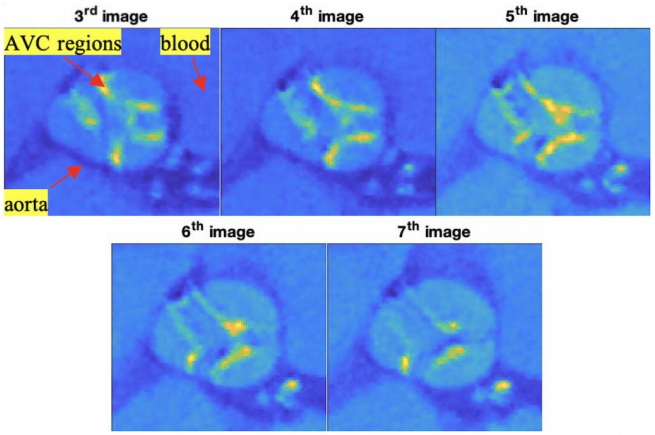


Fig. 1. Sample of sequential CT scans of the aortic root from the 3rd to the 7th image.

ATD are prevalent in engineering, including the real-time monitoring of the crack growth on a building surface [8] and the overheating areas on industrial facilities [9].

In this study, we propose a systematic framework for ATD, which is based on a set of definitions for multiple types of structural information on various modes and their combinations of a tensor. As discussed in Section III, this general framework utilizes the structural representations to specify a class of optimization problems that are easily customized for a wide range of tensor decomposition applications. In this framework, the optimization problems are defined as the summation of multiple terms and each term specifies one assumption on the structural information of one tensor component. Such assumptions on the structural information include the sparsity [10], smoothness [11], low-rank [12], and piecewise constancy which can be defined in one or multiple modes/slices of the tensor components. An alternating direction method of multipliers (ADMM) algorithm [13] is adopted for solving this problem, as it is highly parallelizable and thus suitable for high dimensional data analysis.

The rest of the article is organized as follows. The related literature is reviewed in Section II. Section III presents the definition of the structural information, the ATD framework, and the ADMM algorithm. The formulation and solution of two specific engineering examples are introduced to demonstrate the methodology. Section IV further presents the simulation studies based on these two examples. In Section V, a case study involving real data in medical imaging is presented, and Section VI concludes the article.

II. Literature Review

Traditional image analysis tools for extracting the region of interest from images can be summarized into four categories including edge-based, thresholding, region-based, and clustering-based methods [14], [15]. Edge-based methods focus on detecting the sharp discontinuity in the image, including Sobel edge detector [16], Prewitt edge detector [17], Laplacian of Gaussian edge detector [18], and Watershed method [19]. Those methods are either not robust to noise or require seed initialization [1]. Thresholding methods focus on detecting the intensity difference in the image, including the global thresholding method [20] and local thresholding method [21]. The global thresholding method performs poorly when large intensity variation exists in the image. The local thresholding method usually has the problem of inconsistent thresholding of feature region and false segmentation in the background region [1]. Region-based methods include the region growing method [22], the region merging and splitting methods [23]. Clustering-based methods includes Fuzzy C-Mean Clustering Method [24], K-means method [25], and Mean shift method [26]. Those methods suffer from low computational speed and/or undefined/local optimal solutions [14], [15].

Recently, some literature proposed decomposing a data matrix into a summation of multiple components for extracting the region of interest from images. For identifying the anomalies in images, Yan *et al.* [1] proposed a smooth-sparse decomposition (SSD) method to decompose an image into a smooth background, sparse anomalies and error and showed that their proposed method outperforms the first three categories of traditional image analysis tools mentioned above. Similar additive decomposition was demonstrated in the robust principal component analysis (RPCA) [12], which decomposed the data matrix into a low-rank component and a sparse component.

The matrix decomposition framework has demonstrated its effectiveness in process monitoring [1], [2] and moving object detection [27]. As tensors become increasingly common, there is an urgent need to extend this idea to tensor data. Compared with data matrices, a notable characteristic of tensor data is the distinct structural information of slices in certain modes. As a result, the tensor should be reshaped appropriately, so that the rich structural information can be revealed. Although some researchers extended the decomposition methods to tensor data, most of them did not incorporate this structural information into their decomposition strategies. For example, the spatio-temporal smooth sparse decomposition (ST-SSD) method [2] extended SSD [1] to handling spatio-temporal tensor data. It only assumed that the decomposed two tensor components were either smooth or sparse in both time and space, but did not utilize these properties on individual modes. The RPCA was adjusted to identify the moving objects on a static background in image streams, while it was conducted by simply reshaping the spatio-temporal tensor to a matrix [27]. To extend the matrix decomposition framework to tensor data for generating more accurate and interpretable results, it is critical to have a systematic way to define the rich assumptions on the structural information of the tensor data mathematically.

In the literature, assorted penalization methods have been used to promote the desired properties of the estimators in regularized regression. These penalties can be used in the tensor decomposition framework to enhance the smoothness in one or more slices of the tensor [11], promote the sparsity in different slices at one or more modes [10], and limit the patterns of variations in certain slices of the tensor [12]. Our proposed framework, ATD, combines these penalization methods with the unique structure of the tensor data into a

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on August 03,2021 at 02:18:39 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

high dimensional optimization problem, which can be solved efficiently with the ADMM algorithm [13].

III. Additive Tensor Decomposition Framework

In this section, we introduce the general ATD framework and the solution procedure, and further demonstrate them using two examples. Throughout the article, the set $\{1, \ldots, n\}$ is represented as $[n]$. We denote a scalar by a lowercase letter a, a vector by a boldface lowercase letter $\mathbf{a}$, and a matrix by a boldface uppercase letter $\mathbf{A}$. An order-d tensor is denoted by a calligraphic letter (calligraph A) $\in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_d}$, where I_i is the dimension of its ith mode. Element $(i_1, \ldots, i_d)$ of the tensor $\mathcal{A}$ is represented as $\mathcal{A}(i_1, \ldots, i_d)$. The $(i_{d_1}, \ldots, i_{d_k})$ slice of the tensor $\mathcal{A}$ at mode $d_1, \ldots, d_k$ is represented as $\mathcal{A}(:, \ldots, :, i_{d_1}, :, \ldots, :, i_{d_k}, :, \ldots, :)$, where $i_{d_j} \in [I_{d_j}]$ represents the elements at mode d_j, $j = 1, \ldots, k$. For ease of exposition, let $\mathcal{A}_{i_{d1}, \ldots, i_{d_k}}$ denote $\mathcal{A}(:, \ldots, :, i_{d_1}, :, \ldots, :, i_{d_k}, :, \ldots, :)$. The mode-$k$ fiber is defined by a column vector $\mathcal{A}(i_1, \ldots, :, i_{k-1}, :, i_{k+1}, \ldots, i_d)$ [7]. The mode-$(r_1, \ldots, r_L)$ matricization of $\mathcal{A}$ is a matrix $\mathcal{A}_{(I_{r_1}, \ldots, I_{r_L}) \times (I_{q_1}, \ldots, I_{q_{d-L}})} \in \mathbb{R}^{(I_{r_1}, \ldots, I_{r_L}) \times (I_{q_1}, \ldots, I_{q_{d-L}})}$, whose element at entry $(i_{r_1} + (i_{r_2} - 1) I_{r_1} + \ldots + (i_{r_L} - 1) I_{r_1}, \ldots, I_{r_{L-1}}, i_{q_1} + (i_{q_2} - 1) I_{q_1} + \ldots + (i_{q_{d-L}} - 1) I_{q_1}, \ldots, I_{q_{d-L-1}})$ is $\mathcal{A}(i_1, \ldots, i_d)$, where $\{r_1, \ldots, r_L\}$ and $\{q_1, \ldots, q_{d-L}\}$ is a partition of $\{1, \ldots, d\}$ and $i_k \in [I_k]$ for all $k \in [d]$. Specifically, $\mathcal{A}_{(j)} \in \mathbb{R}^{I_j \times I_{-j}}$ is the mode-j matricization of $\mathcal{A}$ whose rows are the slices of A at mode j, where $I_{-j} = \Pi_{i \in [d] - \{j\}} I_i$.

A. General Problem Formulation

Assume that an order-d tensor $\mathcal{M} \in \mathbb{R}^{I_1 \times \cdots \times I_d}$ is decomposed into a summation of m tensor components $\mathcal{X}_i$ with the same size, where $i \in [m]$, and that there are n_i structural assumptions on $\mathcal{X}_i$. The decomposition is achieved by minimizing the summation of regularization terms specified by all assumptions on the structural information given in Problem (1)

$$\begin{aligned} &\underset{\mathcal{X}_1, \ldots, \mathcal{X}_m}{\text{mininize}} \sum_{i=1}^{m} \sum_{j=1}^{n_i} \lambda_{i,j}\, p_{i,j}(\mathcal{X}_i) \\ &\text{s.t. } \mathcal{M} = \sum_{i=1}^{m} \mathcal{X}_i. \end{aligned} \tag{1}$$

In Problem (1), $p_{i,j}(\cdot)$ specifies the jth assumption on the structural information on $\mathcal{X}_i$, by taking a large value when $\mathcal{X}_i$ does not satisfy the specified assumptions on the structural information. The trade-off between multiple regularization terms is specified by the tuning parameters $\lambda_{i,j}$. Certain assumptions on the structural information are detailed below, which can be applied to either the entire tensor or certain slices.

1) Assumption of the Structural Information on Smoothness: We say that the entire tensor $\mathcal{X}$ or the slices at certain modes are smooth if the elements in this tensor or each of its slices has similar values, whenever their indices are close to each other [1], [2]. We use mode-$(d_1, \ldots, d_k)$ smoothness of a tensor to describe the similarity among different slices at mode $d_1, \ldots, d_k$, which can be represented as $p(\mathcal{X}) = \sum_{s=1}^{k} \|\mathbf{D}_{d_s}^{l} \mathcal{X}_{(d_s)}\|_F^2$, where $\mathbf{D}_{d_s}^{l} \in \mathbb{R}^{I_{d_s} \times I_{d_s}}$, is the finite difference matrix of order l on mode s [1], $s \in [k]$. The notation $\|\mathbf{A}\|_F$ represents the Frobenius norm of $\mathbf{A}$. For example, when $\mathcal{X}$ is an order-3 tensor whose first mode represents time and every $\mathcal{X}(i_1, :, :)$ represents an image, the mode-$(2, 3)$ smoothness essentially indicates that each image is smooth [see Fig. 2(a)]. The mode-(2) smoothness, on the other hand, indicates that all vectors $\mathcal{X}(i_1, :, i_3)$ compose a smooth curve for every element (i_1, i_3), where $i_1 \in [I_1]$ and $i_3 \in [I_3]$. It means that the slices of $\mathcal{X}$ change smoothly in the second mode [see Fig. 2(b)].

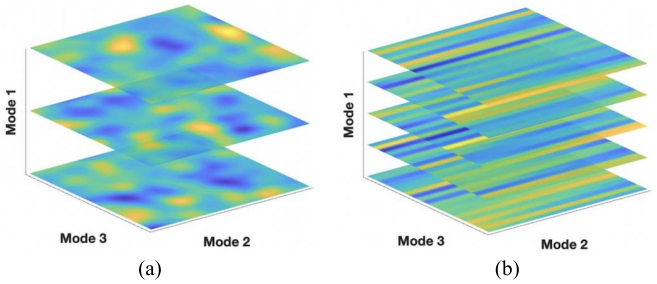


Fig. 2. Illustration of the definition of mode-$(d_1, \ldots, d_k)$ smoothness. (a) Mode-$(2, 3)$ smoothness. (b) Mode-(2) smoothness.

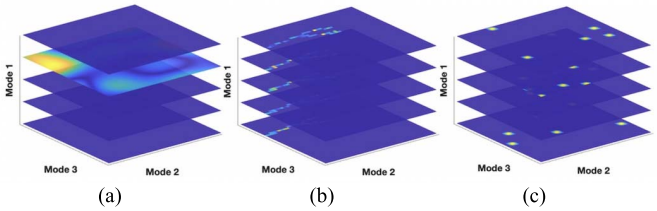


Fig. 3. Illustration of the definition of mode-$(d_1, \ldots, d_k)$ sparsity. (a) Mode-(1) sparsity. (b) Mode-$(2, 3)$ sparsity. (c) Mode-$(1, 2, 3)$ sparsity.

2) Assumption of the Structural Information on Sparsity: Some tensor components represent the sparse anomaly and isolated features. To reflect this information, we may specify the sparsity among slices $1, \ldots, k$ as $p(\mathcal{X}) = \sum_{i_1=1}^{I_1} \ldots \sum_{i_k=1}^{I_k} \|\text{vec}(\mathcal{X}(1, \ldots, k, :, \ldots, :))\|_2$, using the group Lasso penalty. It can be extended to arbitrary mode-$(d_1, \ldots, d_k)$-sparsity naturally. For example, the mode-(1) sparsity indicates that only some slices at mode 1 contain nonzero values [see Fig. 3(a)]. The mode-$(2, 3)$ sparsity indicates that each slice at mode 1 is sparse [see Fig. 3(b)]. Note that mode-$(1, \ldots, d)$ sparsity is equivalent to the sparsity of the whole tensor since $p(\mathcal{X}) = \sum_{i_1=1}^{I_1} \ldots \sum_{i_d=1}^{I_d} \|\mathcal{X}(i_1, \ldots, i_d)\|_2 = \|\text{vec}(\mathcal{X})\|_1$, as indicated in Fig. 3(c).

3) Assumption of the Structural Information on Variation Patterns: In many applications, certain modes of the tensor data have a limited number of variation patterns. With appropriate reshaping operations, associated slices of the tensor can be transformed to low-rank matrices. We define the mode-$(l_1, \ldots, l_q)$ low rank of slices at mode $d_1, \ldots, d_k$ for this structural information, where $(l_1, \ldots, l_q) \cap (d_1, \ldots, d_k) = \emptyset$. Without loss of generality, we say that a tensor component $\mathcal{X}$ is mode-$(1, \ldots, q)$ low rank for its slices at mode $q+1, \ldots, q+k$ if the mode-$(1, \ldots, q)$ matricization of each $\mathcal{X}$'s slice at mode $(q+1, \ldots, q+k)$, i.e., each $\mathcal{X}(:, \ldots, :, i_{q+1}, \ldots, i_{q+k}, :, \ldots, :)$,

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on August 03,2021 at 02:18:39 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

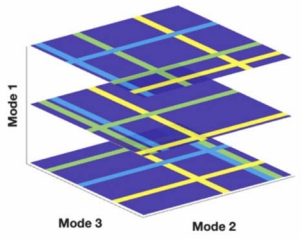
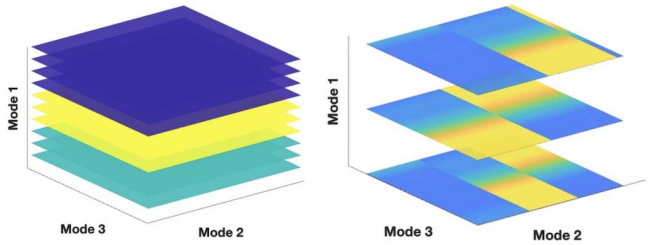


Fig. 4. Mode-(3) low rank of slices at mode 1.



(a) (b)

Fig. 5. Illustration of the definition of mode-$(1,\ldots,q)$ piecewise constancy. (a) Mode-(1) piecewise constancy of $\mathcal{X}$. (b) Mode-(2) piecewise constancy of slices at mode 1.

is low rank. Therefore, this information can be represented as

$$p(\mathcal{X}) = \sum_{i_q=1}^{I_q} \cdots \sum_{i_{q+k}=1}^{I_{q+k}} \left\| \left(\mathcal{X}(:,\ldots,:,i_{q+1},\ldots,i_{q+k},:,\ldots,:)\right)_{(I_1\ldots I_q)\times(I_{q+k+1}\ldots I_n)} \right\|_*$$

where $\|\cdot\|_*$ is the nuclear norm. We say that a tensor is mode-$(1,\ldots,q)$ low rank if its mode-$(1,\ldots,q)$ matricization is low rank. For order-3 tensor $\mathcal{X}$'s whose first mode is time and the rest represent images, we give two examples of this structural information. First, consider that each image in $\mathcal{X}$ represents a textured background with repetitive vertical and horizontal patterns [28]. Then each slice $\mathcal{X}(i_1,:,:)$ at mode 1 is of mode-(3) low rank (see Fig. 4), and this information can be represented as $p(\mathcal{X}) = \sum_{i_1=1}^{I_1} \|(\mathcal{X}(i_1,:,:))\|_*$, where $\|\cdot\|_*$ is the nuclear norm. Second, consider that all images in the tensor $\mathcal{X}$ are similar, representing a static background. Then $\mathcal{X}$ is of mode-(1) low rank and this information is represented by the regularization $p(\mathcal{X}) = \|\mathcal{X}_{(1)}\|_*$.

4) Assumption of the Structural Information on Piecewise Constancy: In certain applications, some slices of the tensor are piecewise constant. Without loss of generality, we formally define mode-$(1,\ldots,q)$ piecewise constancy of slices at mode $q+1,\ldots,q+k$ as follows.

Definition: $\mathcal{X}$ is mode -$(1,\ldots,q)$ piecewise constancy of slices at mode $q+1,\ldots,q+k$, if for any $i_{q+1} \in [I_{q+1}],\ldots,i_{q+k} \in [I_{q+k}]$, the set of grid points $I_{q+k+1} \times \cdots \times I_d$ can be partitioned into multiple continuous regions $R_1(i_{q+1},\ldots,i_{q+k}),\ldots,R_g(i_{q+1},\ldots,i_{q+k})$, such that any two-order q sub-tensors

$$\mathcal{X}(:,\ldots,:,i_{q+1},:,\ldots,:,i_{q+k},i_{q+k+1},\ldots,i_d) = \mathcal{X}(:,\ldots,:,i_{q+1},:,\ldots,:,i_{q+k},i'_{q+k+1},\ldots,i'_d)$$

where $(i_{q+k+1},\ldots,i_d)$ and $(i'_{q+k+1},\ldots,i'_d)$ are in the one of the same regions.

In multiple change-point detections, most adjacent slices along temporal mode d_s in slices at mode $(d_1,\ldots,d_k)$ shall be the same if instantaneous change seldom happens, where $s \notin [k]$. It is defined as mode-(d_s) piecewise constancy of slices at mode $d_1,\ldots,d_k$. This characteristic of the tensor can be regularized using a fused lasso penalty [29]

$$p(\mathcal{X}) = \sum_{i_{d1}=1}^{I_{d_1}} \cdots \sum_{i_{d_k}=1}^{I_{d_k}} \sum_{i_{d_s}=1}^{I_{d_s}-1} \left\| \mathrm{vec}\left(\mathcal{X}(:,\ldots,:,i_{d_1},:,\ldots,:,i_{d_k},:,\ldots,i_{d_s}+1,\ldots,:) - \mathcal{X}(:,\ldots,:,i_{d_1},:,\ldots,:,i_{d_k},:,\ldots,i_{d_s},\ldots,:)\right)\right\|_2.$$

For example, when $\mathcal{X}$ is an order-3 tensor whose first mode represents time and every $\mathcal{X}(i_1,:,:)$ represents an image, mode-(1) piecewise constancy of $\mathcal{X}$ indicates that instantaneous change among images seldom happens at temporal mode [see Fig. 5(a)]. For another example, mode-(2) piecewise constancy of slices at mode 1 means that it is piecewise constant inside each image along mode 2 [see Fig. 5(b)].

With the regularization terms promoting the above-mentioned properties, formulation (1) is versatile and can be tailored for many applications. In order for the readers to better understand our framework and adopt it in their own applications, we give two specific examples below.

5) Example 1: Monitoring Crack Growth on the Surface of Engineering Structures: Engineering structures like concrete beams are often subject to fatigue stress which leads to cracks. Image-based crack detection becomes popular due to its high efficiency and objective assessment of deterioration. However, the irregular size of cracks and irregularly illumination of the surfaces are the main challenges in this inspection method [30]. In this example, we show the capability of the ATD method in monitoring the crack growth under irregular illumination. Consider the following situation, we take images of a concrete wall at a fixed orientation every day to monitor the growth of a crack on it. All images collected in I_1 days can be represented in a tensor $\mathcal{M} \in \mathbb{R}^{I_1\times I_2\times I_3}$, where $I_2 \times I_3$ is the size of each image. Based on the general formulation in Problem (1), the objective function and constraints in this specific application can be determined. Specifically, we decompose $\mathcal{M}$ into a summation of two components, the background of the wall $\mathcal{X}_1$ and the crack $\mathcal{X}_2$. The backgrounds of the images are all smooth, but they are subject to the variation caused by the unstable brightness conditions. To describe the smoothness of each image, the regularization is represented as mode-(2, 3) smoothness $p_{1,1}(\mathcal{X}_1) = \|\mathbf{D}_2^1\mathcal{X}_{1(2)}\|_F^2$ and $p_{1,2}(\mathcal{X}_1) = \|\mathbf{D}_3^1\mathcal{X}_{1(3)}\|_F^2$, where $\mathbf{D}_i^{1\prime}s \in \mathbb{R}^{I_i\times I_i}$ are first-order

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on August 03,2021 at 02:18:39 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

difference matrix [1] for $i = 1, \ldots, 3$

$$\mathbf{D}_i^1 = \begin{bmatrix} 1 & -1 & & & \mathbf{0} \\ 1 & -1 & & & \\ & 1 & -1 & & \\ & & \ddots & \ddots & \\ & & 1 & -1 & \\ \mathbf{0} & & & 1 & -1 \end{bmatrix}.$$

The crack is a consecutive line that grows slowly across frames. Pixels on this line is subject to the variation caused by the unstable brightness conditions, and the pixel intensity on the same line can be different. Therefore, we only assume the temporal smoothness of the measurements of each individual pixel, and we represent this assumption as mode-(1) smoothness $\|\mathbf{D}_1^2 \mathcal{X}_{2(1)}\|_F^2$, where $\mathbf{D}_1^2 \in \mathbb{R}^{I_1 \times I_1}$ is the second-order difference matrix with modified Neumann boundary condition [31]

$$\mathbf{D}_1^2 = \begin{bmatrix} -1 & 1 & & & \mathbf{O} \\ 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ \mathbf{O} & & & 1 & -1 \end{bmatrix}.$$

Furthermore, the crack is limited to a local region on the wall [8], and thus we assume that it has mode-(1, 2, 3) sparsity property, represented as $\|\text{vec}(\mathcal{X}_2)\|_1$. In total, the optimization problem specified by the ATD framework is given as

$$\begin{aligned} \underset{\mathcal{X}_1, \mathcal{X}_2}{\text{minimize}} \;& \lambda_{1,1} \left\| \mathbf{D}_2^1 \mathcal{X}_{1(2)} \right\|_F^2 + \lambda_{1,2} \left\| \mathbf{D}_3^1 \mathcal{X}_{1(3)} \right\|_F^2 \\ & + \lambda_{2,1} \left\| \mathbf{D}_1^2 \mathcal{X}_{2(1)} \right\|_F^2 + \lambda_{2,2} \|\text{vec}(\mathcal{X}_2)\|_1 \\ \text{s.t. } & \mathcal{M} = \mathcal{X}_1 + \mathcal{X}_2 \end{aligned} \tag{2}$$

where $\lambda_{1,1}$, $\lambda_{1,2}$, $\lambda_{2,1}$, and $\lambda_{2,2}$ are tuning parameters.

6) Example 2: Monitoring the Spots of Overheating on a Heated Surface: High temperature is one of the root causes for equipment degradation. Infrared thermography (IRT) can provide the thermal image of the entire measured equipment. Therefore, it has been successfully utilized in the condition monitoring of numerous industries including nuclear, aerospace, and article industries [32]. In this example, we show the capability of the ATD framework of monitoring the spot of overheating on a heated surface. Consider the case where a thermal camera is installed above a surface of a fluidized catalytic cracking regenerator, to monitor the spots of overheating [9]. Spots of overheating are observed as few small regions on the monitored surface in which the temperature is much higher than other sections of the surface. In general, some hotspots' locations may change in different time frames and others may appear at the same location throughout the monitoring period. Apart from the hotspots, the background temperature of the surface is smooth and vary slowly over time, driven by the heated material inside the container and the environmental condition that affects the cooling.

With the general formulation in Problem (1), the objective function and constraints in this specific application can be determined as follows. First, we form a tensor $\mathcal{M} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ from I_1 collected thermal images of sizes $I_2 \times I_3$. Our objective is to decompose it into three tensors $\mathcal{X}_1$, $\mathcal{X}_2$, and $\mathcal{X}_3$ that represent the varying background, the static hotspots, and the moving hotspots, respectively. Among them, all images in X_1 are smooth, and this property is regularized by mode-(2, 3) smoothness $p_{1,1}(\mathcal{X}_1) = \|\mathbf{D}_2^1 \mathcal{X}_{1(2)}\|_F^2$ and $p_{1,2}(\mathcal{X}_1) = \|\mathbf{D}_3^1 \mathcal{X}_{1(3)}\|_F^2$. Also, variations of the images of the background are only driven by the heating and cooling effects, and thus all background images reside in a low-rank subspace. Therefore, $\mathcal{X}_{1(1)}$ should have the low-rank property and thus regularization $p_{1,3}(\mathcal{X}_1) = \|\mathcal{X}_{1(1)}\|_*$ is employed. Most elements in $\mathcal{X}_2$ are zero, which is regularized by $p_{2,2}(\mathcal{X}_2) = \|\text{vec}(\mathcal{X}_2)\|_1$. Also, the images in $\mathcal{X}_2$ are similar, which means that they should also reside in a low-dimensional subspace, and thus $\mathcal{X}_2$ is also regularized by mode-(1) low rank penalty $p_{2,1}(\mathcal{X}_2) = \|\mathcal{X}_{2(1)}\|_*$. Finally, most elements in $\mathcal{X}_3$ are zero, so the regularization on $\mathcal{X}_3$ is given by $p_{3,1}(\mathcal{X}_3) = \|\text{vec}(\mathcal{X}_3)\|_1$. Put everything together, we formulate Problem (3) using the ATD framework

$$\begin{aligned} \underset{\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_3}{\text{minimize}} \;& \lambda_{1,1} \left\| \mathbf{D}_2^1 \mathcal{X}_{1(2)} \right\|_F^2 + \lambda_{1,2} \left\| \mathbf{D}_3^1 \mathcal{X}_{1(3)} \right\|_F^2 + \lambda_{1,3} \left\| \mathcal{X}_{1(1)} \right\|_* \\ & + \lambda_{2,1} \left\| \mathcal{X}_{2(1)} \right\|_* + \lambda_{2,2} \|\text{vec}(\mathcal{X}_2)\|_1 + \lambda_{3,1} \|\text{vec}(\mathcal{X}_3)\|_1 \\ \text{s.t. } & \mathcal{M} = \mathcal{X}_1 + \mathcal{X}_2 + \mathcal{X}_3 \end{aligned} \tag{3}$$

where $\lambda_{1,1}$, $\lambda_{1,2}$, $\lambda_{1,3}$, $\lambda_{2,1}$, $\lambda_{2,2}$, and $\lambda_{3,1}$ are tuning parameters.

We will follow up with the solution procedures, simulated images, and simulation results for the Problems (2) and (3) in the latter part of the article.

B. Problem Solution

Notice that Problem (1) is convex and bounded from below by zero. Therefore, an optimal solution to this problem exists. To deal with the high dimensionality of the decision variables, we adopt an ADMM algorithm to solve this problem [13]. In Problem (1), there are n_i additive terms associated with the same variable $\mathcal{X}_i$. We introduce n_i new ancillary tensors $\mathcal{X}_i^{(1)}, \ldots, \mathcal{X}_i^{(n_i)}$ as copies of $\mathcal{X}_i$, and further let $\tilde{\mathcal{X}}_i = (\mathcal{X}_i^{(1)}, \ldots, \mathcal{X}_i^{(n_i)})$ and $\tilde{\mathcal{X}} = (\tilde{\mathcal{X}}_1, \ldots, \tilde{\mathcal{X}}_m)$. Then Problem (1) can be solved using the ADMM algorithm listed in Algorithm 1.

In Algorithm 1, the parameter η defines the step size. The proximal operator is defined as $\text{prox}_h(\mathcal{X}) = \text{argmin}_{\mathcal{Y}}(h(\mathcal{Y}) + (1/2)\|\text{vec}(\mathcal{X} - \mathcal{Y})\|_2^2)$. The derivation of the ADMM algorithm is similar to that of the ADMM allocation problem in Parikh and Boyd [13]. For completeness, we list it in Appendix I. Note that in step (3), the size of input $\mathbf{x}$ for each element $(i_1, \ldots, i_d)$ is $\sum_{i=1}^m n_i$, which is the total number of assumptions on the structural information of all components. This is generally a small number and is irrelevant to the total number of elements in the tensor. Therefore, the size of $\mathbf{A}$ is not too big. Also, the matrix $\mathbf{A}$ is the same for any element $(i_1, \ldots, i_d)$. Therefore, the matrix $\mathbf{A}^\top (\mathbf{A}\mathbf{A}^\top)^{-1}$ only needs to be calculated once. Finally, the operation for each element $(i_1, \ldots, i_d)$ can be evaluated in parallel. These three features significantly boost the computational speed.

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on August 03,2021 at 02:18:39 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

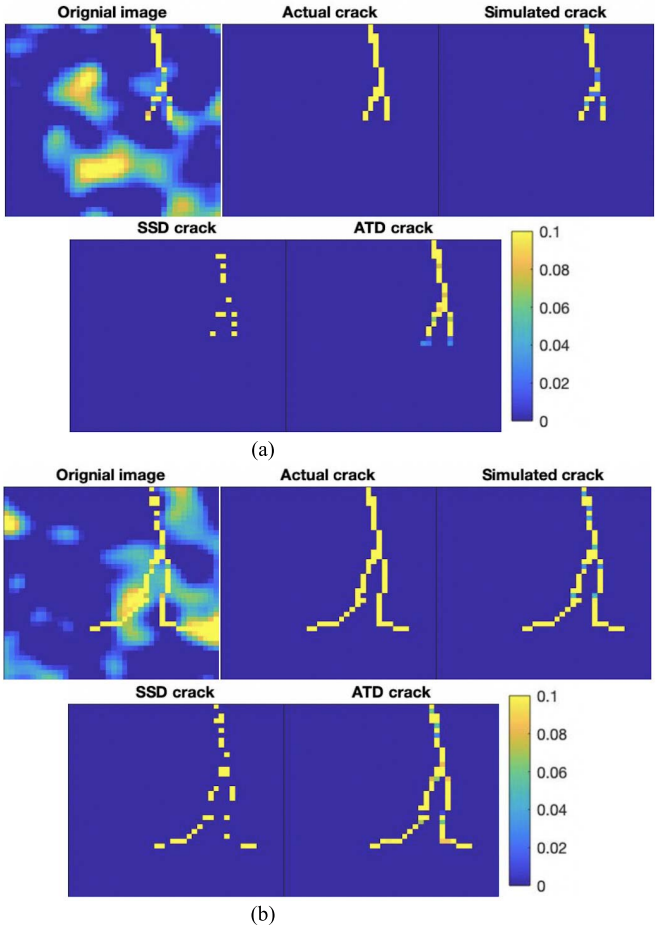


Fig. 6. Illustration of decomposed images using the ATD and the SSD methods. (a) 20th image. (b) 30th image.

Now, let us revisit those two examples in the previous section and give the solution procedure for these specific problems.

1) The Solution to Example 1: To solve Problem (2), we define $\tilde{\mathcal{X}} = (\mathcal{X}_1^{(1)}, \mathcal{X}_1^{(2)}, \mathcal{X}_2^{(1)}, \mathcal{X}_2^{(2)})$.

The proximal operators of $p_{1,1}$, $p_{1,2}$, $p_{2,1}$, $p_{2,2}$, and g are evaluated using the following procedure.

If $(\mathbf{x}) = \|\mathbf{A}\mathbf{x}\|_2^2$, then the proximal operator of $\eta p(\cdot)$ is $\text{prox}_{\eta p(\cdot)}(\mathbf{x}) = (2\eta\mathbf{A}^\top\mathbf{A} + \mathbf{I})^{-1}\mathbf{x}$. Therefore, the proximal operators associated with the smoothness penalization $p_{1,1}$, $p_{1,2}$, and $p_{2,1}$ can be, respectively, expressed as

$$[\text{prox}_{\eta\lambda_{i,j}p_{i,j}}(\mathcal{X})]_{(s)} = \left(2\lambda_{i,j}\eta\mathbf{D}_s^{l\top}\mathbf{D}_s^l + \mathbf{I}\right)^{-1}\mathcal{X}_{(s)}. \quad (4)$$

The proximal operator of $p_{2,2}$ can be evaluated using

$$\text{prox}_{\eta\lambda_{2,2}p_{2,2}}(\mathcal{X}) = \left(\mathcal{X} - \lambda_{2,2}\eta I\right)_+ \quad (5)$$

where $I \in \mathbb{R}^{I_1\times I_2\times I_3}$ is a tensor with all elements being 1's.

In step (3) of Algorithm 1, the calculation of $(\mathcal{Z}_1^{(1)}, \mathcal{Z}_1^{(2)}, \mathcal{Z}_2^{(1)}, \mathcal{Z}_2^{(2)}) = \text{prox}_{\eta g}[\mathcal{X}_1^{(1)}, \mathcal{X}_1^{(2)}, \mathcal{X}_2^{(1)}, \mathcal{X}_2^{(2)}]$ is performed for each set of corresponding elements of $\mathcal{Z}_1^{(1)}, \mathcal{Z}_1^{(2)}, \mathcal{Z}_2^{(1)}, \mathcal{Z}_2^{(2)}$.

Algorithm 1 ADMM Algorithm for Problem (1)

Initialize $\tilde{\mathcal{Z}}$ and $\tilde{\mathcal{U}}$ as the same data structure as $\tilde{\mathcal{X}}$, with all their elements being 0.

Do:

(1) Save $(\tilde{\mathcal{Z}}_{\text{prev}}, \tilde{\mathcal{U}}_{\text{prev}}) \leftarrow (\tilde{\mathcal{Z}}, \tilde{\mathcal{U}})$.

(2) Update $\tilde{\mathcal{X}}$:

$$\tilde{\mathcal{X}} = \left(\text{prox}_{\eta\lambda_{1,1}p_{11}}\left(\mathcal{Z}_{1,\text{prev}}^{(1)} - \mathcal{U}_{1,\text{prev}}^{(1)}\right), \ldots, \text{prox}_{\eta\lambda_{m,n_m}p_{mn_m}}\left(\mathcal{Z}_{m,\text{prev}}^{(n_m)} - \mathcal{U}_{m,\text{prev}}^{(n_m)}\right)\right)$$

(3) Update $\tilde{\mathcal{Z}}$

For all $(i_1, \ldots, i_d)$:

$$\left(\mathcal{Z}_1^{(1)}(i_1, \ldots, i_d), \ldots, \mathcal{Z}_m^{(n_i)}(i_1, \ldots, i_d)\right) \leftarrow \mathbf{x} - \mathbf{A}^\top\left(\mathbf{A}\mathbf{A}^\top\right)^{-1}(\mathbf{A}\mathbf{x} - \mathbf{b}),$$

where

$$\mathbf{x} = \left(\mathcal{X}_1^{(1)}(i_1, \ldots, i_d) + \mathcal{U}_1^{(1)}(i_1, \ldots, i_d), \ldots, \mathcal{X}_m^{(n_i)}(i_1, \ldots, i_d) + \mathcal{U}_m^{(n_i)}(i_1, \ldots, i_d)\right)^T,$$

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & & \mathbf{0} \\ & \mathbf{A}_2 & \\ & \ddots & \\ \mathbf{0} & & \mathbf{A}_m \\ \mathbf{a}_1 & \mathbf{a}_2 \ \ldots & \mathbf{a}_m \end{bmatrix}, \mathbf{b} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ \mathcal{M}(i_1, \ldots, i_d) \end{bmatrix}.$$

$\mathbf{a}_i \in \mathbb{R}^{1\times n_i}$, a_i=[1, 0, ..., 0] and $\mathbf{A}_i \in \mathbb{R}^{n_i\times n_i}$,

$$\mathbf{A}_i = \begin{bmatrix} 1 & -1 & 0 & \ldots & 0 \\ 1 & 0 & -1 & & 0 \\ 1 & 0 & 0 & & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & \ldots & -1 \end{bmatrix} \in \mathbb{R}^{(n_i-1)\times n_i}.$$

(4) Update $\tilde{\mathcal{U}} = \tilde{\mathcal{U}}_{\text{prev}} + \tilde{\mathcal{X}} - \tilde{\mathcal{Z}}$.

Until: $\left\|\tilde{\mathcal{U}} - \tilde{\mathcal{U}}_{\text{prev}}\right\| < \epsilon$, $\left\|\tilde{\mathcal{Z}} - \mathcal{Z}_{\text{prev}}\right\| < \epsilon$.

Specifically, the (i_1, i_2, i_3) element is updated via

$$\left[\mathcal{Z}_1^{(1)}(i_1, i_2, i_3), \mathcal{Z}_1^{(2)}(i_1, i_2, i_3), \mathcal{Z}_2^{(1)}(i_1, i_2, i_3), \mathcal{Z}_2^{(2)}(i_1, i_2, i_3)\right]^\top = \mathbf{x} - \mathbf{A}^\top\left(\mathbf{A}\mathbf{A}^\top\right)^{-1}(\mathbf{A}\mathbf{x} - \mathbf{b}) \quad (6)$$

where

$$\mathbf{x} = \left(\mathcal{X}_1^{(1)}(i_1, i_2, i_3), \ldots, \mathcal{X}_2^{(2)}(i_1, i_2, i_3)\right)^\top$$

$$\mathbf{b} = \begin{bmatrix} 0 \\ 0 \\ \mathcal{M}(i_1, i_2, i_3) \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 1 & 0 & 1 & 0 \end{bmatrix}.$$

Here matrix $\mathbf{I}$ represents the identity matrices of appropriate dimension.

The optimization Problem (2) can therefore be solved using the ADMM algorithm listed in Algorithm 2.

The convergence of the ADMM algorithm for Example 1 is guaranteed in Parikh and Boyd [13]. Next, we will analyze the computational complexity of each step of the iteration. The convergence process is illustrated in Fig. 7 in Section IV-A.

The computational complexity of each step is analyzed as follows.

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on August 03,2021 at 02:18:39 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Algorithm 2 ADMM Algorithm for Example *1*

Initialize $\tilde{\mathcal{Z}} = \left(\mathcal{Z}_1^{(1)}, \mathcal{Z}_1^{(2)}, \mathcal{Z}_2^{(1)}, \mathcal{Z}_2^{(2)}\right)$ and $\tilde{\mathcal{U}} = \left(\mathcal{U}_1^{(1)}, \mathcal{U}_1^{(2)}, \mathcal{U}_2^{(1)}, \mathcal{U}_2^{(2)}\right)$ with the same data structure as $\tilde{\mathcal{X}} = \left(\mathcal{X}_1^{(1)}, \mathcal{X}_1^{(2)}, \mathcal{X}_2^{(1)}, \mathcal{X}_2^{(2)}\right)$. Set all their elements to 0.

Do:

(1) Save $(\tilde{\mathcal{Z}}_{\text{prev}}, \tilde{\mathcal{U}}_{\text{prev}}) \leftarrow (\tilde{\mathcal{Z}}, \tilde{\mathcal{U}})$.

(2) Update all mode-2 fibers in $\mathcal{X}_1^{(1)}$, all mode-3 fibers in $\mathcal{X}_1^{(2)}$ and all mode-1 fibers in $\mathcal{X}_2^{(1)}$ in the For loops of (2a-2c) and assign each element of $\mathcal{X}_2^{(2)}$ in (2d) in parallel.

(2a) **For** all $i \in [I_1], k \in [I_3]$:
$\mathcal{X}_1^{(1)}(i,:,k) = \left(2\lambda_{1,1}\eta \mathbf{D}_2^{1\top}\mathbf{D}_2^1 + \mathbf{I}_1^{(1)}\right)^{-1}$
$\left(\mathcal{Z}_{1,\text{prev}}^{(1)}(i,:,k) - \mathcal{U}_{1,\text{prev}}^{(1)}(i,:,k)\right)$, where $\mathbf{D}_2^1 \in \mathbb{R}^{I_2\times I_2}$ is the first-order difference matrix and $\mathbf{I}_1^{(1)} \in \mathbb{R}^{I_2\times I_2}$ is the identity matrix;

(2b) **For** all $i \in [I_1], j \in [I_2]$:
$\mathcal{X}_1^{(2)}(i,j,:) = \left(2\lambda_{1,2}\eta \mathbf{D}_3^{1\top}\mathbf{D}_3^1 + \mathbf{I}_1^{(2)}\right)^{-1}$
$\left(\mathcal{Z}_{1,\text{prev}}^{(2)}(i,j,:) - \mathcal{U}_{1,\text{prev}}^{(2)}(i,j,:)\right)$, where $\mathbf{D}_3^1 \in \mathbb{R}^{I_3\times I_3}$ is the first-order difference matrix and $\mathbf{I}_1^{(2)} \in \mathbb{R}^{I_3\times I_3}$ is the identity matrix;

(2c) **For** all $j \in [I_2], k \in [I_3]$:
$\mathcal{X}_2^{(1)}(:,j,k) = \left(2\lambda_{2,1}\eta \mathbf{D}_1^{2\top}\mathbf{D}_1^2 + \mathbf{I}_2^{(1)}\right)^{-1}$
$\left(\mathcal{Z}_{2,\text{prev}}^{(1)}(:,j,k) - \mathcal{U}_{2,\text{prev}}^{(1)}(:,j,k)\right)$, where $\mathbf{D}_1^2 \in \mathbb{R}^{I_1\times I_1}$ is the second-order difference matrix and $\mathbf{I}_2^{(1)} \in \mathbb{R}^{I_1\times I_1}$ is the identity matrix;

(2d) $\mathcal{X}_2^{(2)} = \left(\mathcal{Z}_{2,\text{prev}}^{(2)} - \mathcal{U}_{2,\text{prev}}^{(2)} - \lambda_{2,2}\eta\mathbf{I}\right)_+$.

(2e) $\tilde{\mathcal{X}} \leftarrow (\mathcal{X}_1^{(1)}, \mathcal{X}_1^{(2)}, \mathcal{X}_2^{(1)}, \mathcal{X}_2^{(2)})$.

(3) **For** all (i_1, i_2, i_3):
$\left(\mathcal{Z}_1^{(1)}(i_1,i_2,i_3), \ldots, \mathcal{Z}_2^{(2)}(i_1,i_2,i_3)\right)$
$\leftarrow \mathbf{x} - \mathbf{A}^\top\left(\mathbf{A}\mathbf{A}^\top\right)^{-1}(\mathbf{A}\mathbf{x} - \mathbf{b})$,
where
$\mathbf{x} = \left(\mathcal{X}_1^{(1)}(i_1,i_2,i_3) + \mathcal{U}_1^{(1)}(i_1,i_2,i_3), \ldots,\right.$
$\left.\mathcal{X}_2^{(2)}(i_1,i_2,i_3) + \mathcal{U}_2^{(2)}(i_1,i_2,i_3)\right)^T$
and
$\mathbf{A} = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$, $\mathbf{b} = \begin{bmatrix} 0 \\ 0 \\ \mathcal{M}(i_1,i_2,i_3) \end{bmatrix}$.

(4) Update $\tilde{\mathcal{U}} = \tilde{\mathcal{U}}_{\text{prev}} + \tilde{\mathcal{X}} - \tilde{\mathcal{Z}}$.

Until: $\left\|\tilde{\mathcal{U}} - \tilde{\mathcal{U}}_{\text{prev}}\right\| < \epsilon$, $\left\|\tilde{\mathcal{Z}} - \mathcal{Z}_{\text{prev}}\right\| < \epsilon$.

All steps in (2a)–(2d) can be performed in parallel. In steps (2a)–(2c), each vector in the form $\mathcal{X}_1^{(j)}(i,:,k)$ is updated individually, $j = 1,2,3, i \in [I]$ and $k \in [K]$. Each update can be calculated with Proposition 2 in Wang and Shi [33], involving a computational complexity of $O(I_j \log I_j)$. In total, (2a)–(2c) involve $I_1 I_2$ parallelizable operations of complexity $O(I_3 \log I_3)$, $I_1 I_3$ parallelizable operations of complexity $O(I_2 \log I_2)$, and $I_2 I_3$ parallelizable operations of complexity $O(I_1 \log I_1)$. Step (2d) performs elementwise addition and shrinkage, involving $I_1 I_2 I_3$ parallelizable operations of complexity $O(1)$.

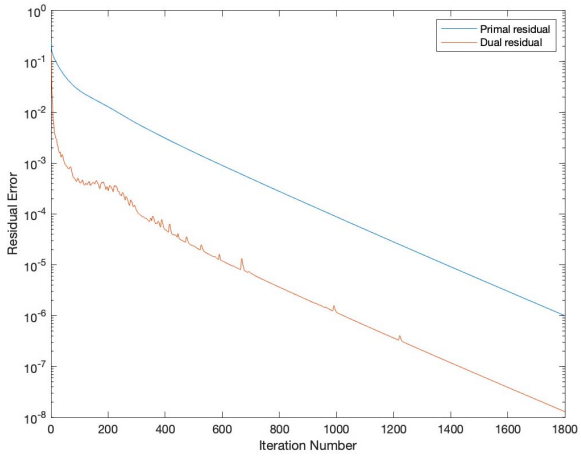


Fig. 7. Plot of the residual error versus iteration number.

Step (3) involves $I_1 I_2 I_3$ parallelizable operations of $O(1)$, updating a four-dimensional vector through the same linear operation.

Step (4) conducts elementwise addition operations, which involves $I_1 I_2 I_3$ parallelizable operations of $O(1)$.

From the above discussion, we can see that in each iteration, the computational complexity is $O(I_1 \log I_1 + I_2 \log I_2 + I_3 \log I_3)$ if the algorithm is fully parallelized. In another extreme, the time complexity is of order $O(I_1 I_2 I_3(\log I_1 + \log I_2 + \log I_3))$ if the algorithm is not parallelized. Note that the above analysis does not take the communication cost involved in the parallel computation into consideration. When implementing the algorithm, the level of parallelization should be determined based on the tradeoff between the communication cost and computational cost.

2) The Solution to Example 2: First, we define the copies $\tilde{\mathcal{X}} = (\mathcal{X}_1^{(1)}, \mathcal{X}_1^{(2)}, \mathcal{X}_1^{(3)}, \mathcal{X}_2^{(1)}, \mathcal{X}_2^{(2)}, \mathcal{X}_3^{(1)})$.

The proximal operators of $p_{1,1}$ and $p_{1,2}$ are evaluated in the same way as that $p_{1,1}$ and $p_{1,2}$ in Example 1

$$\left[\text{prox}_{\eta\lambda_{i,j}p_{i,j}}(\mathcal{X})\right]_{(s)} = \left(2\lambda_{i,j}\eta \mathbf{D}_s^{l\top}\mathbf{D}_s^l + \mathbf{I}\right)^{-1}\mathcal{X}_{(s)}. \tag{7}$$

The proximal operators of $p_{1,3}$ and $p_{2,1}$ are in the form of the nuclear norms, whose proximal operator is given by

$$\text{prox}_{\eta\|\cdot\|_*}(\mathbf{A}) = \sum_i (\sigma_i - \eta)_+ \mathbf{u}_i \mathbf{v}_i^\top \tag{8}$$

for which $\mathbf{A} = \sum_i \sigma_i \mathbf{u}_i \mathbf{v}_i^\top$ is the singular value decomposition of $\mathbf{A}$ [13].

The proximal operators of functions $p_{2,2}$ and $p_{3,1}$ are in the form of ℓ_1-norm. Their proximal operators are given by

$$\text{prox}_{\eta\|\cdot\|_1}(\mathcal{X}) = (\mathcal{X} - \eta)_+. \tag{9}$$

The step (3) of Algorithm 1 becomes

$$\left[\mathcal{Z}_1^{(1)}(i_1,i_2,i_3), \mathcal{Z}_1^{(2)}(i_1,i_2,i_3), \mathcal{Z}_1^{(3)}(i_1,i_2,i_3)\right.$$
$$\left.\mathcal{Z}_2^{(1)}(i_1,i_2,i_3), \mathcal{Z}_2^{(2)}(i_1,i_2,i_3), \mathcal{Z}_3^{(1)}(i_1,i_2,i_3)\right]^\top$$
$$= \mathbf{x} - \mathbf{A}^\top\left(\mathbf{A}\mathbf{A}^\top\right)^{-1}(\mathbf{A}\mathbf{x} - \mathbf{b}) \tag{10}$$

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on August 03,2021 at 02:18:39 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

where $\mathbf{x} = (\mathcal{X}_1^{(1)}(i_1,i_2,i_3), \ldots, \mathcal{X}_3^{(1)}(i_1,i_2,i_3))^\mathsf{T}$

$$\mathbf{A} = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \text{ and}$$

$$\mathbf{b} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \mathcal{M}(i_1,i_2,i_3) \end{bmatrix}.$$

Problem (3) can be solved using the ADMM algorithm listed in Algorithm 3.

The convergence of ADMM algorithm for Example 2 is also guaranteed and the computational complexity of each step can be analyzed similarly as the ADMM algorithm in Example 1. The only difference is that the singular value decomposition involved in steps (2c) and (2d) has time complexity $O(I_1 I_2 I_3)$, assuming $I_1 < I_2 I_3$. The convergence rate is illustrated with the simulation study in Fig. 9 in Section IV-B.

C. Discussion

It can be seen from those two examples that the ATD framework successfully integrates multiple structural information required for every tensor component. It is both a unification and an extension of the existing decomposition methods, such as SSD, ST-SSD, and RPCA. In particular, when we ignore the temporal smoothness property of the crack component in Example 1, the formulation will be reduced to the SSD method. If we invoke the temporal smoothness property to the background, we will arrive at the ST-SSD method. However, neither SSD nor ST-SSD fully describes the setup of Example 1, because neither of them incorporates the information that the crack is smooth in time but sparse in space and that the background is smooth in space but may not be smooth in time. In Example 2, if we do not distinguish the static anomaly and static background, the problem formulation reduces to RPCA for foreground detection. However, RPCA cannot fully describe the setup of Example 2 because it does not incorporate the sparsity information of static hotspots or smoothness information of the background. It can also be seen from those two examples that the ATD framework is fundamentally different from the Tucker decomposition: the ATD decomposes the input data tensor into the summation of multiple tensor components based on different assumptions on their structural information. However, Tucker decomposition decomposes a low-rank data tensor into the multiplication of a set of matrices with one small core tensor.

We can also see from solution algorithms of those two examples that the adopted ADMM algorithm is highly parallelable. For example, in steps 2(a)–(c) of Algorithm 2, the update of each tensor component can be done fiber by fiber. In step (3), the update of each element can be performed in parallel. This operation can be distributed on several processors which is highly suitable for tensor with high dimensions.

In the following subsections, we first discuss the selection of tuning parameters and then discuss the options we have if the decomposition yields nonunique solution.

Algorithm 3 ADMM Algorithm for Example *2*

Initialize $\tilde{\mathcal{Z}} = \left(\mathcal{Z}_1^{(1)}, \mathcal{Z}_1^{(2)}, \mathcal{Z}_1^{(3)}, \mathcal{Z}_2^{(1)}, \mathcal{Z}_2^{(2)}, \mathcal{Z}_3^{(1)}\right)$ and $\tilde{\mathcal{U}} = \left(\mathcal{U}_1^{(1)}, \mathcal{U}_1^{(2)}, \mathcal{U}_1^{(3)}, \mathcal{U}_2^{(1)}, \mathcal{U}_2^{(2)}, \mathcal{U}_3^{(1)}\right)$ as the same data structure as $\tilde{\mathcal{X}} = \left(\mathcal{X}_1^{(1)}, \mathcal{X}_1^{(2)}, \mathcal{X}_1^{(3)}, \mathcal{X}_2^{(1)}, \mathcal{X}_2^{(2)}, \mathcal{X}_3^{(1)}\right)$, with all their elements being 0.

Do:

(1) Save $(\tilde{\mathcal{Z}}_{\text{prev}}, \tilde{\mathcal{U}}_{\text{prev}}) \leftarrow (\tilde{\mathcal{Z}}, \tilde{\mathcal{U}})$.

(2) Update all mode-2 fibers in $\mathcal{X}_1^{(1)}$, all mode-3 fibers in $\mathcal{X}_1^{(2)}$, all elements of $\mathcal{X}_2^{(2)}$, all elements of $\mathcal{X}_3^{(1)}$ in parallel with $\mathcal{X}_{1(1)}^{(3)}$ and $\mathcal{X}_{2(1)}^{(1)}$ in (2a)-(2e) below.

(2a) **For** all $i \in [I_1] k \in [I_3]$:
$\mathcal{X}_1^{(1)}(i,:,k) = \left(2\lambda_{1,1}\eta \mathbf{D}_2^{1\top}\mathbf{D}_2^1 + \mathbf{I}_1^{(1)}\right)^{-1} \left(\mathcal{Z}_{1,\text{prev}}^{(1)}(i,:,k) - \mathcal{U}_{1,\text{prev}}^{(1)}(i,:,k)\right)_{(2)}$, where $\mathbf{D}_2^1 \in \mathrm{R}^{(I_2-1)\times I_2}$ is the first-order difference matrix and $\mathbf{I}_1^{(1)} \in \mathrm{R}^{I_2\times I_2}$ is the identity matrix;

(2b) **For** all $i \in [I_1]$, $j \in [I_2]$:
$\mathcal{X}_1^{(2)}(i,j,:) = \left(2\lambda_{1,2}\eta \mathbf{D}_3^{1\top}\mathbf{D}_3^1 + \mathbf{I}_1^{(2)}\right)^{-1} \left(\mathcal{Z}_{1,\text{prev}}^{(2)}(i,j,:) - \mathcal{U}_{1,\text{prev}}^{(2)}(i,j,:)\right)_{(3)}$, where $\mathbf{D}_3^1 \in \mathbb{R}^{(I_3-1)\times I_3}$ is the first-order difference matrix and $\mathbf{I}_1^{(2)} \in \mathbb{R}^{I_3\times I_3}$ is the identity matrix;

(2c) $\mathcal{X}_{1(1)}^{(3)} = \sum_i \left(\sigma_i - \lambda_{1,3}\eta\right)_+ \mathbf{u}_i \mathbf{v}_i^\top$, where $\sum_i \sigma_i \mathbf{u}_i \mathbf{v}_i^\top$ is the singular value decomposition of $\left(\mathcal{Z}_{1,\text{prev}}^{(3)} - \mathcal{U}_{1,\text{prev}}^{(3)}\right)_{(1)}$;

(2d) $\mathcal{X}_{2(1)}^{(1)} = \sum_i \left(\sigma_i - \lambda_{2,1}\eta\right)_+ \mathbf{u}_i \mathbf{v}_i^\top$, where $\sum_i \sigma_i \mathbf{u}_i \mathbf{v}_i^\top$ is the singular value decomposition of $\left(\mathcal{Z}_{2,\text{prev}}^{(1)} - \mathcal{U}_{2,\text{prev}}^{(1)}\right)_{(1)}$;

(2e) $\mathcal{X}_2^{(2)} = \left(\mathcal{Z}_{2,\text{prev}}^{(2)} - \mathcal{U}_{2,\text{prev}}^{(2)} - \lambda_{2,2}\eta\right)_+$ and
$\mathcal{X}_3^{(1)} = \left(\mathcal{Z}_{3,\text{prev}}^{(1)} - \mathcal{U}_{3,\text{prev}}^{(1)} - \lambda_{3,1}\eta\right)_+$.

(3) **For** all (i_1,i_2,i_3):
$\left(\mathcal{Z}_1^{(1)}(i_1,i_2,i_3), \ldots, \mathcal{Z}_3^{(1)}(i_1,i_2,i_3)\right) \leftarrow \mathbf{x} - \mathbf{A}^\top\left(\mathbf{A}\mathbf{A}^\mathsf{T}\right)^{-1}(\mathbf{A}\mathbf{x} - \mathbf{b})$,
where
$\mathbf{x} = \left(\mathcal{X}_1^{(1)}(i_1,i_2,i_3) + \mathcal{U}_1^{(1)}(i_1,i_2,i_3), \ldots, \mathcal{X}_3^{(1)}(i_1,i_2,i_3)(i_1,i_2,i_3) + \mathcal{U}_3^{(1)}\right)^T$,
$\mathbf{A} = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}$, and $\mathbf{b} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \mathcal{M}(i_1,i_2,i_3) \end{bmatrix}$.

(4) Update $\tilde{\mathcal{U}} = \tilde{\mathcal{U}}_{\text{prev}} + \tilde{\mathcal{X}} - \tilde{\mathcal{Z}}$.

Until: $\left\|\tilde{\mathcal{U}} - \tilde{\mathcal{U}}_{\text{prev}}\right\| < \epsilon$, $\left\|\tilde{\mathcal{Z}} - \mathcal{Z}_{\text{prev}}\right\| < \epsilon$.

D. Selection of Tuning Parameters

In the general formulation (1), the tuning parameters need to be specified. A large value of the tuning parameter $\lambda_{i,j}$ tends

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on August 03,2021 at 02:18:39 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

to enhance the corresponding penalty terms $p_{i,j}(\cdot)$. Below we propose a formal procedure of selecting the tuning parameters.

1) Case I: we have training samples. If some training samples contain one or more existing tensor data $\mathcal{X}_{\{1\}}, \ldots, \mathcal{X}_{\{n\}}$ and their real additive components of interest $\mathcal{X}_{\{1\}i}, \ldots, \mathcal{X}_{\{n\}i}, i \in [m]$ are available, the optimal tuning parameters can be determined by minimizing the error [29]

$$\hat{\lambda} = \text{argmin}_{\boldsymbol{\lambda}} \left\{ \sum_{i=1}^{m} \sum_{j=1}^{n} L_i\big(\mathcal{X}_{\{j\}i} - \hat{\mathcal{X}}_{\{j\}i}(\lambda)\big) \right\}.$$

Here $\hat{\mathcal{X}}_{\{j\}i}(\lambda)$ is the estimation of the ith tensor component by solving the problem involving the jth training tensor $\mathcal{X}_{\{j\}}$ and using the tuning parameters $\boldsymbol{\lambda}$. $L_i(\cdot)$ specifies the loss involving the difference between the true component $\mathcal{X}_{\{1\}}$ and the estimated component $\hat{\mathcal{X}}_{\{1\}}(\lambda)$. The loss function $L_i(\cdot)$ can be selected depending on specific scenarios: if we only care about the nonzero part in some components, as in the application of anomaly detections, we should use 0–1 loss function $L_i(\mathcal{X}) = \|\text{vec}(\mathcal{X})\|_0$; If we care about the values of the extract features, we may prefer to use the Frobenius loss: $L_i(\mathcal{X}) = \|\mathcal{X}\|_F$. If the decomposition result of the ith component is not of interest, the loss function of $L_i(\cdot)$ can be set to 0.

2) Case II: we do not have training samples. In this case, $\boldsymbol{\lambda}$ should be selected empirically based on the structural assumption that we want to incorporate. We propose a formal procedure that assists the selection of $\boldsymbol{\lambda}$.

1) Identify the sensitivity of every tuning parameter $\lambda_{i,j}$ by adjusting each tuning parameter and observe the outcome of the decomposition. From the observation, we identify the tuning parameters that affect the outcome most.
2) Adjust the tuning parameters one at a time. Start with the most sensitive parameter, and then adjust the rest of them to achieve the desired property of decomposition.

In Example 1, the decomposed background image generated by larger values of $\lambda_{1,1}$ and $\lambda_{1,2}$ tends to be smoother than that generated by small values of $\lambda_{1,1}$ and $\lambda_{1,2}$. In other words, large values of $\lambda_{1,1}$ and $\lambda_{1,2}$ lead to smooth background image. The changes among extracted crack images generated by a large value of $\lambda_{2,1}$ tend to be smoother, and the extracted crack images generated by a large value of $\lambda_{2,2}$ tend to be sparser. Therefore, we select the tuning parameters in this application using the following procedure: we first select $\lambda_{1,1}$, $\lambda_{1,2}$, and $\lambda_{2,2}$ because the selected tensor components are very sensitive to the value $(\lambda_{2,2}/\lambda_{1,1} + \lambda_{1,2})$. Then, the tuning parameters for temporal smoothness penalty $\lambda_{2,1}$ is selected accordingly to preserve the low-intensity pixels on the crack and to shrink intensity of other pixels that belongs to the background, because the sparsity penalty tends to shrink the value of low intensity pixels in the tensor component $\mathcal{X}_2$ to zero, regardless of the low-intensity pixels are on the crack or not.

In Example 2, a large $\lambda_{1,3}$ and $\lambda_{2,1}$ help to enhance the low rank property of the mode-(1) matricization of the background tensor and the static hotspot tensor; the extracted moving hotspots generated by a large value of $\lambda_{3,1}$ tend to be sparser, and the background generated by large values of $\lambda_{1,1}$ and $\lambda_{1,2}$ tends to be smoother; the extracted static hotspots generated by a large value of $\lambda_{2,2}$ tend to be sparser. We select the tuning parameters in Example 2 as follows. First, we select $\lambda_{2,1}$, $\lambda_{1,3}$, and $\lambda_{3,1}$ to separate the moving hotspots and static components, including the background and static hotspots. Then, we select $\lambda_{1,1}$, $\lambda_{1,2}$, and $\lambda_{2,2}$ that control the smoothness of the image in the background tensor and the sparsity of the static hotspot tensor, to separate the static hotspot from the smooth background.

In general, when changing the value of each tuning parameter, the practitioner may observe how the characteristics of the decomposition result change. In this procedure, the practitioner can adjust each tuning parameter to find a combination that leads to a desirable decomposition result.

E. Problem of Nonunique Solution

It should be pointed out that the solution to the optimization problem sometimes may not be unique. This is because structural assumptions in the framework are not rich enough to determine each component. For example, two tensor components with the same structural information of sparsity are not distinguishable under the ATD framework. To generate a unique solution, we propose the following two solutions. One solution is to incorporate prior knowledge that distinguishes the tensor components that cannot be uniquely identified. For instance, if we know that the nonzero elements in two sparse tensor components form different shapes, such as square and circle, we can express these components using two different sets of functional bases to ensure a unique solution. If no prior knowledge is available, we propose to add another regularization term $\epsilon \sum_{i=1}^{d} \|\mathcal{X}_i\|_F^2$ to the objective function, where ϵ is a small number. With this term, the objective function becomes strongly convex and the unique solution exists.

IV. Simulation Studies for Performance Evaluation

In this section, we will present our simulation studies on those two examples to illustrate the ATD framework and demonstrate its effectiveness.

A. Simulation Study for Example 1

The data for monitoring the crack growing process include $I_1 = 30$ consecutive measurement images of size 40×40. These images form a tensor $\mathcal{M} \in \mathbb{R}^{30\times 40\times 40}$. We simulate $\mathcal{M}$ by summing up two tensors $\mathcal{X}_1$ and $\mathcal{X}_2$ that represent the background and the crack, respectively. Each $\mathcal{X}_1(i,:,:)$, $i \in [I_1]$ representing the background is generated by a 2-D smooth Gaussian process with mean 0 and covariance function $K(\mathbf{x}, \mathbf{x}') = \exp(-0.05\|\mathbf{x} - \mathbf{x}'\|^2)$, where $\mathbf{x}$ and $\mathbf{x}'$ are the locations of corresponding pixels. Most values in the image $\mathcal{X}_2(i,:,:)$ are zeros, and the number of nonzero values in $\mathcal{X}_2(i,:,:)$ gradually grows when the index i increases from 1 to 30, representing the crack growing on the wall. These values are generated from independent identically distributed (i.i.d.) $N(0.1, 0.1)$ random variables to represent the

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on August 03,2021 at 02:18:39 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE I
FPR, TPR VALUES FOR SSD AND ATD METHOD

	FPR	TPR
SSD (auto)	0.0006	0.4481
SSD (best)	0.0230	0.8898
ATD	0.0203	1

random lighting and shadowing conditions. The first images of Fig. 6(a) and (b) illustrate the 20th and the 30th images in $\mathcal{M}$, the second images in Fig. 6(a) and (b) illustrate the corresponding images for actual crack which is a continuous line and the third images in Fig. 6(a) and (b) illustrate the images of the simulated crack under irregularly illumination.

We then decompose $\mathcal{M}$ into two components $\hat{\mathcal{X}}_1$ and $\hat{\mathcal{X}}_2$ by solving Problem (2). In the ADMM algorithm, the step size is $\eta = 0.01$, and the tuning parameters are $\lambda_{1,1} = \lambda_{1,2} = 1$, $\lambda_{2,1} = 10$, and $\lambda_{2,2} = 0.08$. The algorithm is implemented in MATLAB on a personal laptop with Intel Core i5 3.1-GHz CPU and 8-GB memory. The stopping criterion is that the l_2 norms of both the primal residual and dual residual are below 10^{-6}. The convergence process is illustrated in Fig. 7. In Fig. 7, the x-axis is the iteration number and y-axis is the value of residual error in logarithmic scale. We can see that both the algorithm achieves linear convergence rate and both the primal residual and dual residuals drop below 10^{-6} after 1800 iterations in 4.58 s. The fifth images of Fig. 6(a) and (b) illustrate the estimated crack of the 20th and the 30th images using the ATD-based method. It is shown that the ATD-based method captures the growth of the whole crack accurately. The video illustrating the result of the decomposition is provided in the supplementary material of this article.

For comparison, we also applied the SSD method to each image in $\mathcal{M}$. The crack images obtained from the SSD method are shown in the fourth images in Fig. 6(a) and (b), in which the tuning parameters are determined by the SSD method automatically. It can be seen that the SSD method only captures a subset of points on the crack because the smoothness of the crack in the temporal mode is ignored. In order to evaluate the result quantitatively, we calculated and compared false positive rate (FPR), defined as the proportion of noncrack pixels predicted as crack pixels, and true positive rate (TPR), defined as the proportion of crack pixels predicted as crack pixels. The FPR and TPR values are reported in Table I.

Notice that, although we can increase the TPR value of the SSD method by selecting the tuning parameters manually, the FPR value also increases. As we set similar FPR values for the ATD and the SSD methods, we find that the best TPR value of the SSD method is 0.89, also listed in Table I. As the TPR of ATD is 1, we can see that the ATD method performs better than the SSD method. The ℓ_1 regularization in the SSD encourages sparsity of the anomaly, but it pushes all pixels with low intensity to zero, including the ones on the crack. By contrast, the ATD-based method also promotes the temporal smoothness of $\mathcal{X}_2$, which preserves the pixels with low intensity on the crack, whereas penalizes other pixels to zeros.

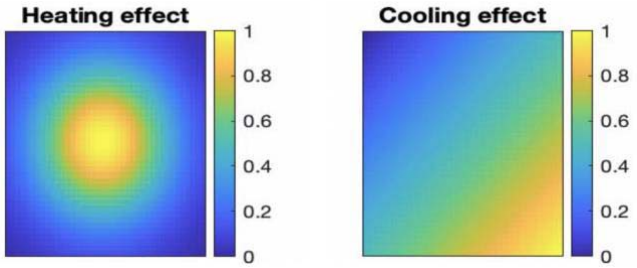


Fig. 8. Illustration of cooling and heating effect temperature field.

B. Simulation Study for Example 2

The tensor $\mathcal{M}$ in Example 2 is generated to simulate the consecutive measurements taken from a thermal camera in a heated surface monitoring process. It also contains 30 images of size 40×40, and it is generated by summing up three tensors $\mathcal{X}_1$, $\mathcal{X}_2$, and $\mathcal{X}_3$ of the same size that represents the true background, the static hotspot, and the moving hotspot, respectively. Among them, each mode 1 slice of the tensor $\mathcal{X}_1$ is generated from

$$\mathcal{X}(i, :, :) = U_i \mathbf{T}_0 + (1 - U_i)\mathbf{T}_1$$

where $\mathbf{T}_0$ is a 40×40 matrix representing the heating effect of the heating process, $\mathbf{T}_1$ is a matrix of the same size representing the cooling effect, and U_i is a uniform distributed random variable in range $[0, 1]$, i.e., $U_i \sim U[0, 1]$, representing a random combination of the two effects.

The images representing the matrices $\mathbf{T}_0$ and $\mathbf{T}_1$ are shown in Fig. 8. To simulate the heating effect with a point heat source at the center of the image, we generate $\mathbf{T}_0(i, j)$ using the value of $f_0(i, j)$, where f_0 is the probability density function of $N\big((20, 20)^\top, 10\mathbf{I}\big)$, where $\mathbf{I}$ is a 2×2 identity matrix. Then, we transform all values of $\mathbf{T}_0$ linearly such that the maximum and minimum value of $\mathbf{T}_0$ are 1 and 0, respectively. With this setup, the maximum value within $\mathbf{T}_0$ is 1, located at the center of the image; when the pixel moves farther way from the center, the value of $\mathbf{T}_0(i, j)$ gradually drops to 0. To simulate the cooling effect, we generate $\mathbf{T}_1(i, j)$ using a linear function $f_1(i, j) = c_1(i + j) + c_2$, where c_1 and c_2 are adjusted so that the maximum and minimum values within the matrix $\mathbf{T}_1$ are 0 and 1, respectively. It represents that the coolant flows from the upper-left corner to the bottom-right corner of the image.

All images within the tensor $\mathcal{X}_2$ are the same. The only nonzero values in these images are located in a fixed 2×2 rectangular region at the lower-left part of the images, with intensity value 1. The nonzero values in each image of the tensor $\mathcal{X}_3$ are also located in a 2×2 rectangle with intensity value 1. However, this rectangle locates at the upper-left part of the image. When the image index i increases, the rectangle in $\mathcal{X}_3(i, :, :)$ moves from the left side to the right side across the images.

We decompose the tensor $\mathcal{M}$ into components $\hat{\mathcal{X}}_1$, $\hat{\mathcal{X}}_2$, and $\hat{\mathcal{X}}_3$ by solving Problem (3). The step size $\eta = 0.01$. Tuning parameters are $\lambda_{1,1} = \lambda_{1,2} = 30$, $\lambda_{1,3} = \lambda_{2,1} = 1$, $\lambda_{2,2} = 1.9$,

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on August 03,2021 at 02:18:39 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

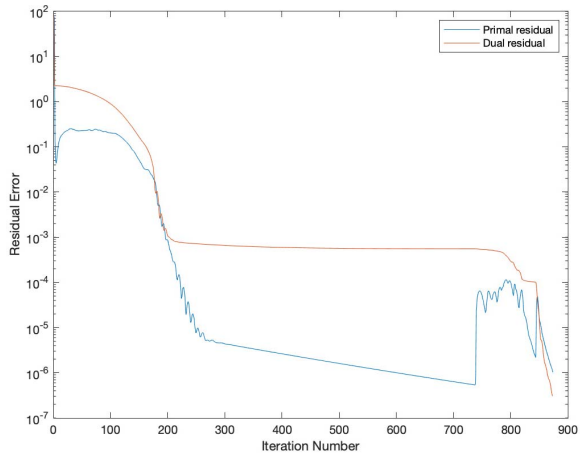


Fig. 9. Plot of the residual error versus iteration number.

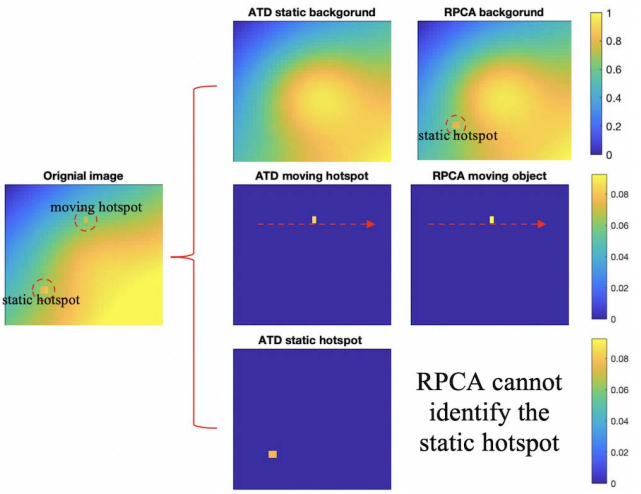


Fig. 10. Illustration of the 20th decomposed images using the ATD and the RPCA methods.

and $\lambda_{3,1} = 2$. We adopt the same stopping criterion as Example 1 for the ADMM algorithm. Fig. 9 illustrates the convergence process. The x-axis and y-axis represent the iteration number and the value of residual error in logarithmic scale, respectively. In this simulation, although the ADMM algorithm is not very efficient when the accuracy is high, we can still achieve desired accuracy in acceptable iterations. The primal and dual residuals become smaller than the threshold with 900 iterations in 106.138 s.

In the first and the second columns of Fig. 10, we illustrate the 20th simulated images, with the decomposed background, the static hotspot, and the moving hotspot using the ATD method. The decomposed background and the moving object from the RPCA method are illustrated in the third column. The video illustrating the result of the decomposition is provided in the supplementary material of this article. It can be seen that the proposed ATD method successfully separates the static background, the static hotspot, and the moving hotspot. Compared with the result of ATD, RPCA can only identify the moving object, but it fails to separate the true background and the static hotspot. This is because RPCA does not consider the spatial smoothness of the background.

From the two simulation studies, we verify that the ATD framework provides greater capability of decomposing the tensor of interest into multiple components than the existing methods, by considering richer assumptions on the structural information of the components in individual slices.

V. Case Study With the 3-D Medical Image Analysis

In this section, we revisit the case study in the Introduction, medical image analysis for 3-D printed phantoms [6] to further demonstrate the advantage of the ATD method.

TAVR is an alternative option for aortic stenosis patients with high surgical risk to perform aortic valve replacement [6]. AVC has been proposed as an important determinant for PVR of TAVR. To avoid PVR, the physicians use multi-material 3-D printing to fabricate a prototype for patients' aortic root anatomies, upon which they plan the surgery. The parts representing the heart tissues and the AVC regions in the prototype are manufactured with materials of different mechanical properties for accurate simulation. Characterizing the shape of the AVC regions based on the patients' CT scanning images is critical for fabricating a geometrically accurate prototype, and this task is conventionally accomplished by a board-certified cardiologist [6]. In this case study, we formulate a problem using the ATD framework, aiming at extracting the AVC regions automatically.

Yan *et al.* [1] showed that their proposed method outperformed the first three categories of traditional image analysis tools mentioned in Section II. As the ATD method extends the SSD method in Yan *et al.* [1] to tensors with richer structures, we expect that our method shall maintain the benefit of SSD over other existing methods. Therefore, we will only compare our method with the SSD method and the K-means method in our article.

From the pre-procedural contrast-enhanced CT scan of some patients' aortic region, we select 12 images of size 101×101 that are collected sequentially. These images form a tensor $\mathcal{M}$ of size $12 \times 101 \times 101$. Each image shall be divided into three regions: a contrast-enhanced blood pool with moderate intensity, the soft tissues with low intensity, and the AVC regions with high intensity. We focus on separating AVC regions from the blood pool and the soft tissues.

Specifically, the tensor $\mathcal{M}$ is decomposed into three components: the background of the blood pool and the soft tissues $\mathcal{X}_1 \in \mathbb{R}^{12\times101\times101}$, the AVC regions $\mathcal{X}_2 \in \mathbb{R}^{12\times101\times102}$, and the measurement error $\mathcal{X}_3 \in \mathbb{R}^{12\times101\times102}$. This is a high-dimensional optimization problem, as $3 \times 12 \times 101 \times 101$ unknown variables need to be estimated. Anatomic structure of the heart indicates that the background images are smooth, but they are different from each other because the outer profiles of the aorta are not the same. Therefore, the smoothness regularization is applied to every image of $\mathcal{X}_1$. The regions of AVC are small and attached to the inner side of the aorta and they do not change much between consecutive cross sections. Therefore, the smoothness

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on August 03,2021 at 02:18:39 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

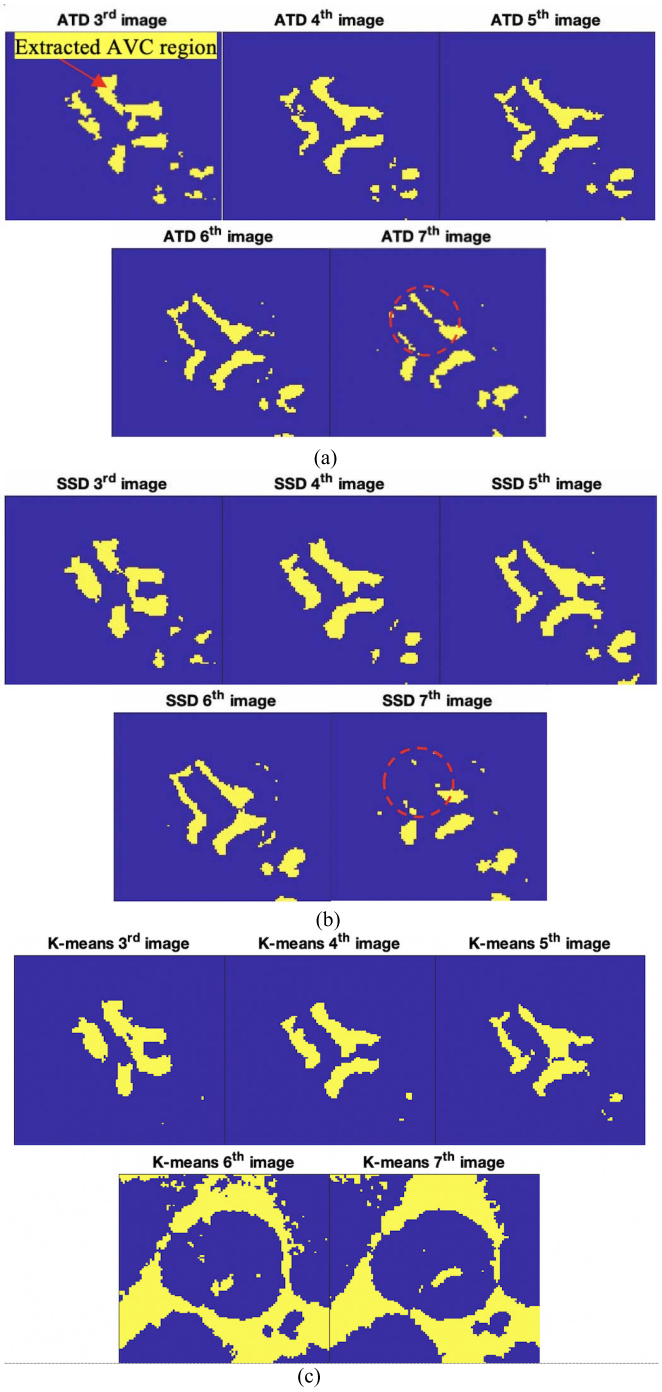


Fig. 11. Extracted AVC regions using the ATD, the SSD, and K-means methods. (a) ATD method decomposed AVC regions in the 4th–7th images. (b) SSD method decomposed AVC regions in the 4th–7th images. (c) K-means method decomposed AVC regions in the 4th–7th images.

regularization in time and the sparsity regularization in space is applied to $\mathcal{X}_2$. The measurement error generally takes small values, and thus squared-ℓ_2 regularization is applied to $\mathcal{X}_3$. The problem formulation is given as follows:

$$\begin{aligned}\underset{\mathcal{X}_1,\mathcal{X}_2,\mathcal{X}_3}{\text{minimize}}\ & \lambda_{1,1}\left(\left\|\mathbf{D}_1\mathcal{X}_{1(2)}\right\|_F^2+\left\|\mathbf{D}_1\mathcal{X}_{1(3)}\right\|_F^2\right)\\ &+\lambda_{2,1}\left\|\mathbf{D}_1\mathcal{X}_{2(1)}\right\|_F^2+\lambda_{2,2}\|\text{vec}(\mathcal{X}_2)\|_1+\lambda_{3,1}\|\mathcal{X}_3\|_2^2\\ \text{s.t. }& \mathcal{M}=\mathcal{X}_1+\mathcal{X}_2+\mathcal{X}_3. \end{aligned}\tag{11}$$

With tuning parameters $\lambda_{1,1}=10$, $\lambda_{2,1}=0.7$, $\lambda_{2,2}=0.16$, and $\lambda_{3,1}=1$, we solved this problem using the ADMM algorithm. Fig. 11(a) illustrates the 3rd–7th images in the solution of $\mathcal{X}_2$, reflecting the AVC regions in five consecutive cross sections. In fabricating the prototypes of patients' aortic root anatomies, we care about the shape of the AVC region. Therefore, Fig. 11 illustrates nonzero values in the images in $\mathcal{X}_2$. The video illustrating the result of the decomposition is provided in the supplementary material.

The result shows that the smooth change of AVC regions across different images is captured by the proposed ATD method, which reflects the anatomic reality. For comparison, we also conduct SSD and K-means methods to extract the AVC regions on individual images. Since the ground truth of exact AVC regions is not available, we compare the results of decomposition by showing the images of the estimated AVC regions. As shown in Fig. 11, K-means performs the worst among those three methods. For the 3rd–6th images, the ATD method achieves similar performance in extracting the AVC regions when the intensity of the anomaly region is high with respect to the background and noise. In the circled region of the 7th image, when the intensity of the AVC regions is low, the SSD method fails to fully extract the anomaly region. When we use a small tuning parameter for sparse penalty in the SSD method, much noise will be introduced into the extracted AVC regions. However, the ATD method can still achieve satisfactory performance in this case, because ATD takes the similarity of the AVC regions' locations between adjacent images into consideration, which helps to preserve the pixels with low intensity in the AVC regions, while setting other pixels to zeros. This case study further illustrates the versatility of the ATD framework and demonstrates its ability in solving real-world problems. It is worth to mention that the values of the extracted AVC regions using the ATD method is smaller than those using the SSD method, due to the extra smoothness penalty added in the formulation of the ATD method. However, it does not affect the boundary of the model, determined by the nonzero values in the decomposed images.

VI. Conclusion

Tensor data become increasingly common in engineering applications. In this article, we propose an ATD framework to extract the quality-related information from tensor data based on the assumptions on the structural information of the tensor components.

The ATD framework achieves the tensor decomposition through integrating multiple types of regularizations on the tensor components, corresponding to assorted structural information such as smoothness, sparsity, and low rank on different modes. In this framework, the corresponding assumption on the structural information is systematically defined for tensor data for the first time. As a unification and extension of the existing decomposition methods like SSD, ST-SSD, and RPCA, ATD provides a general framework to solve a class of ATD problem.

Computation is a major challenge of solving ATD problems: the number of decision variables is the number of the

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on August 03,2021 at 02:18:39 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

tensor components multiplies the number of elements in each tensor, which can be huge. To solve this large-scale problem efficiently, we adopt a highly parallelizable ADMM method.

Throughout the article, we use two examples to demonstrate the versatility and effectiveness of the ATD framework. The case study in medical image analysis demonstrates that the ATD framework can accurately identify the AVC regions by capturing its smooth change between consecutive images. The authors believe that the ATD framework can be applied in an even wider range of applications for tensor data analysis.

APPENDIX A
DERIVATION OF ALGORITHM *1*

This appendix discusses the derivation of Algorithm 1 from a general ADMM algorithm.

After introducing n_i new ancillary tensors $\mathcal{X}_i^{(1)}, \ldots, \mathcal{X}_i^{(n_i)}$ as copies of $\mathcal{X}_i$, and further let $\tilde{\mathcal{X}}_i = (\mathcal{X}_i^{(1)}, \ldots, \mathcal{X}_i^{(n_i)})$ and $\tilde{\mathcal{X}} = (\tilde{\mathcal{X}}_1, \ldots, \tilde{\mathcal{X}}_m)$, formulation (1) is transformed into

$$\text{minimize } f(\tilde{\mathcal{X}}) + g(\tilde{\mathcal{X}}) \tag{12}$$

where $f(\tilde{\mathcal{X}}) = \sum_{i=1}^{m} \sum_{j=1}^{n_i} \lambda_{i,j} p_{i,j}(\mathcal{X}_i^{(j)})$, and $g(\tilde{\mathcal{X}}) = I_C(\tilde{\mathcal{X}})$. Here $I_\Omega(x)$ refers to an indicator function that takes value 0 when $x \in \Omega$, and takes value $+\infty$ if $x \notin \Omega$. The set $C = \{\mathcal{X}_i^{(1)} = \cdots = \mathcal{X}_i^{(n_i)}, i \in [m]\} \cap \{\mathcal{M} = \sum_{i=1}^{m} \mathcal{X}_i^{(1)}\}$.

The form (12) can be solved using the general ADMM algorithm listed in Algorithm 4.

Algorithm 4 General ADMM Algorithm

Initialize $\tilde{\mathcal{Z}}$ and $\tilde{\mathcal{U}}$ as the same data structure as $\tilde{\mathcal{X}}$, with all their elements being 0.

Do:

(1) Save $(\tilde{\mathcal{Z}}_{\text{prev}}, \tilde{\mathcal{U}}_{\text{prev}}) \leftarrow (\tilde{\mathcal{Z}}, \tilde{\mathcal{U}})$.
(2) Update $\tilde{\mathcal{X}} = \text{prox}_{\eta f}(\tilde{\mathcal{Z}}_{\text{prev}} - \tilde{\mathcal{U}}_{\text{prev}})$.
(3) Update $\tilde{\mathcal{Z}} = \text{prox}_{\eta g}(\tilde{\mathcal{X}} + \tilde{\mathcal{U}}_{\text{prev}})$.
(4) Update $\tilde{\mathcal{U}} = \tilde{\mathcal{U}}_{\text{prev}} + \tilde{\mathcal{X}} - \tilde{\mathcal{Z}}$.

Until: $\|\tilde{\mathcal{U}} - \tilde{\mathcal{U}}_{\text{prev}}\| < \epsilon$, $\|\tilde{\mathcal{Z}} - \mathcal{Z}_{\text{prev}}\| < \epsilon$.

Two essential steps of Algorithm 4 are evaluating the proximal operators for ηf and ηg in steps (2) and (3). As f is the summation of multiple terms involving nonoverlapping tensors $\mathcal{X}_i^{(j)}$'s, its proximal operator can be expressed using the proximal operators of individual $p_{i,j}(\cdot)$'s, as indicated by the separable property of the proximal operator [13]

$$\text{prox}_h(\mathbf{x}_1, \mathbf{x}_2) = \big(\text{prox}_{h_1}(\mathbf{x}_1), \text{prox}_{h_2}(\mathbf{x}_2)\big), \quad \text{if } h(\mathbf{x}_1, \mathbf{x}_2) = h_1(\mathbf{x}_1) + h_2(\mathbf{x}_2).$$

Using the separable property again, the proximal operators of each $p_{i,j}(\mathcal{X}_i^{(j)})$ can be expressed via the proximal operators of quadratic functions, norms, and other simple functions, whose closed-forms are available [13]. Therefore, step (2) can be written as

$$\tilde{\mathcal{X}} = \Big(\text{prox}_{\eta\lambda_{1,1}p_{11}}\Big(\mathcal{Z}_{1,\text{prev}}^{(1)} - \mathcal{U}_{1,\text{prev}}^{(1)}\Big), \ldots, \times \text{prox}_{\eta\lambda_{m,n_m}p_{mn_m}}\Big(\mathcal{Z}_{m,\text{prev}}^{(n_m)} - \mathcal{U}_{m,\text{prev}}^{(n_m)}\Big)\Big).$$

As for $g(\tilde{\mathcal{X}})$, the separable property of the proximal operator I_C can be invoked again, by noting that $I_C(\tilde{\mathcal{X}}) = \sum_{i_1,\ldots,i_d} I_{C_{i_1,\ldots,i_d}}(\tilde{\mathcal{X}}_1(i_1, \ldots, i_d), \ldots, \tilde{\mathcal{X}}_m(i_1, \ldots, i_d))$, where

$$C_{i_1,\ldots,i_d} = \Big\{\mathcal{X}_i^{(1)}(i_1, \ldots, i_d) = \cdots = \mathcal{X}_i^{(n_i)}(i_1, \ldots, i_d), \forall i \in [m]\Big\} \cap \left\{\mathcal{M}(i_1, \ldots, i_d) = \sum_{i=1}^{m} \mathcal{X}_i^{(1)}(i_1, \ldots, i_d)\right\}$$

and $\tilde{\mathcal{X}}_i(i_1, \ldots, i_d) = (\mathcal{X}_i^{(1)}(i_1, \ldots, i_d), \ldots, \mathcal{X}_i^{(n_i)}(i_1, \ldots, i_d))$.

Each set $C_{i_1,\ldots,i_d}$ is an affine subset within $\mathbb{R}^{\sum_{i=1}^{m} n_i}$ defined by a system of linear equations

$$C_{i_1,\ldots,i_d} = \Big\{\mathbf{x} \in \mathbb{R}^{\sum_{i=1}^{m} n_i} | \mathbf{A}\mathbf{x} = \mathbf{b}\Big\}$$

where $\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_m \end{bmatrix}$, $\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & & & \mathbf{0} \\ & \mathbf{A}_2 & & \\ & & \ddots & \\ \mathbf{0} & & & \mathbf{A}_m \\ \mathbf{a}_1 & \mathbf{a}_2 & \ldots & \mathbf{a}_m \end{bmatrix}$ and $\mathbf{b} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ \mathcal{M}(i_1, \ldots, i_d) \end{bmatrix}$. Here $\mathbf{x}_i \in \mathbb{R}^{n_i \times 1}$, $\mathbf{a}_i = [1, 0, \ldots, 0]^{\mathrm{T}} \in \mathbb{R}^{n_i \times 1}$, and $\mathbf{A}_i = \begin{bmatrix} 1 & -1 & 0 & \ldots & 0 \\ 1 & 0 & -1 & & 0 \\ 1 & 0 & 0 & & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & \ldots & -1 \end{bmatrix} \in \mathbb{R}^{(n_i-1)\times n_i}$. The proximal operator of $I_{C_{i_1,\ldots,i_d}}$ is thus a projection onto $C_{i_1,\ldots,i_d}$ that can be evaluated using Proposition 1 (cited from [13]).

Proposition 1: Let $\Omega = \{\mathbf{x}|\mathbf{A}\mathbf{x} = \mathbf{b}\}$. The proximal operator of $\eta I_\Omega(\mathbf{x})$ is given by: $\text{prox}_{\eta I_\Omega}(\mathbf{x}) = \text{proj}_\Omega(\mathbf{x}) = \mathbf{x} - \mathbf{A}^\top(\mathbf{A}\mathbf{A}^\top)^{-1}(\mathbf{A}\mathbf{x} - \mathbf{b})$, where $\text{proj}_S(\mathbf{x}) = \text{argmin}_{\mathbf{y}\in\mathbf{S}}\|\mathbf{x} - \mathbf{y}\|_2$.

Therefore, step (3) can be written as

$$\textbf{For all } (i_1, \ldots, i_d): \Big(\mathcal{Z}_1^{(1)}(i_1, \ldots, i_d), \ldots, \mathcal{Z}_m^{(n_i)}(i_1, \ldots, i_d)\Big) \leftarrow \mathbf{x} - \mathbf{A}^\top\big(\mathbf{A}\mathbf{A}^\top\big)^{-1}(\mathbf{A}\mathbf{x} - \mathbf{b})$$

where

$$\mathbf{x} = \Big(\mathcal{X}_1^{(1)}(i_1, \ldots, i_d) + \mathcal{U}_1^{(1)}(i_1, \ldots, i_d), \ldots, \mathcal{X}_m^{(n_i)}(i_1, \ldots, i_d) + \mathcal{U}_m^{(n_i)}(i_1, \ldots, i_d)\Big)^T$$

where $\mathbf{A}$ and $\mathbf{b}$ are defined the same as the ones in (11).

From the above derivation, after updating steps (2) and (3) in Algorithm 4, we arrive at Algorithm 1.

REFERENCES

[1] H. Yan, K. Paynabar, and J. Shi, "Anomaly detection in images with smooth background via smooth-sparse decomposition," *Technometrics*, vol. 59, no. 1, pp. 102–114, 2017.
[2] H. Yan, K. Paynabar, and J. Shi, "Real-time monitoring of high-dimensional functional data streams via spatio-temporal smooth sparse decomposition," *Technometrics*, vol. 60, no. 2, pp. 181–197, 2018.
[3] H. Yan, K. Paynabar, and J. Shi, "Image-based process monitoring using low-rank tensor decomposition," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 1, pp. 216–227, Jan. 2015.

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on August 03,2021 at 02:18:39 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

[4] O. K. Khalique *et al.*, "Quantity and location of aortic valve complex calcification predicts severity and location of paravalvular regurgitation and frequency of post-dilation after balloon-expandable transcatheter aortic valve replacement," *JACC, Cardiovascular Interventions*, vol. 7, no. 8, pp. 885–894, Aug. 2014.

[5] J. Chen, K. Wang, C. Zhang, and B. Wang, "An efficient statistical approach to design 3D-printed metamaterials for mimicking mechanical properties of soft biological tissues," *Additive Manuf.*, vol. 24, pp. 341–352, Dec. 2018.

[6] Z. Qian *et al.*, "Quantitative prediction of paravalvular leak in transcatheter aortic valve replacement based on tissue-mimicking 3D printing," *JACC, Cardiovascular Imag.*, vol. 10, no. 7, pp. 719–731, Jul. 2017.

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

[8] N.-D. Hoang, "Detection of surface crack in building structures using image processing technique with an improved Otsu method for image thresholding," *Adv. Civil Eng.*, vol. 2018, pp. 1–10, Apr. 2018.

[9] P. Duda, Ł. Felkowski, and P. Cyklis, "Identification of overheating of an industrial fluidized catalytic cracking regenerator," *Appl. Thermal Eng.*, vol. 129, pp. 1466–1477, Jan. 2018.

[10] R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight, "Sparsity and smoothness via the fused lasso," *J. Roy. Stat. Soc., B, Stat. Methodol.*, vol. 67, no. 1, pp. 91–108, 2005.

[11] J. O. Ramsay, "Monotone regression splines in action," *Stat. Sci.*, vol. 3, no. 4, pp. 425–441, Nov. 1988.

[12] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *J. ACM*, vol. 58, no. 1, pp. 1–37, 2009.

[13] N. Parikh and S. Boyd, "Proximal algorithms," *Found. Trends Optim.*, vol. 1, no. 3, pp. 127–239, Jan. 2014.

[14] L. K. Lee, S. C. Liew, and W. J. Thong, "A review of image segmentation methodologies in medical image," in *Advanced Computer and Communication Engineering Technology*. Cham, Switzerland: Springer, 2015, pp. 1069–1080.

[15] D. J. Withey and Z. J. Koles, "A review of medical image segmentation: Methods and available software," *Int. J. Bioelectromagnetism*, vol. 10, no. 3, pp. 125–148, 2008.

[16] I. Sobel and G. Feldman, "A 3×3 isotropic gradient operator for image processing," Talk Stanford Artif. Project, Tech. Rep., 1968, pp. 271–272.

[17] J. M. Prewitt, "Object enhancement and extraction," *Picture Process. Psychopictorics*, vol. 10, no. 1, pp. 15–19, 1970.

[18] D. Marr and E. Hildreth, "Theory of edge detection," *Proc. Roy. Soc. London B, Biol. Sci.*, vol. 207, pp. 187–217, Feb. 1980.

[19] S. Beucher and F. Meyer, "The morphological approach to segmentation: The watershed transformation," *Opt. Eng.*, vol. 34, pp. 433–481, Jan. 1993.

[20] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-9, no. 1, pp. 62–66, Jan. 1979.

[21] D. Bradley and G. Roth, "Adaptive thresholding using the integral image," *J. Graph. Tools*, vol. 12, no. 2, pp. 13–21, 2007.

[22] R. Adams and L. Bischof, "Seeded region growing," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 6, pp. 641–647, Jun. 1994.

[23] I. N. Manousakas, P. E. Undrill, G. G. Cameron, and T. W. Redpath, "Split-and-merge segmentation of magnetic resonance medical images: Performance evaluation and extension to three dimensions," *Comput. Biomed. Res.*, vol. 31, no. 6, pp. 393–412, Dec. 1998.

[24] J. C. Bezdek, *Pattern Recognition With Fuzzy Objective Function Algorithms*. New York, NY, USA: Springer, 2013.

[25] M. R. Kaus, J. von Berg, J. Weese, W. Niessen, and V. Pekar, "Automated segmentation of the left ventricle in cardiac MRI," *Med. Imag. Anal.*, vol. 8, no. 3, pp. 245–254, Sep. 2004.

[26] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 603–619, May 2002.

[27] T. Bouwmans and E. H. Zahzah, "Robust PCA via principal component pursuit: A review for a comparative evaluation in video surveillance," *Comput. Vis. Image Understand.*, vol. 122, pp. 22–34, May 2014.

[28] Z. Zhang, A. Ganesh, X. Liang, and Y. Ma, "TILT: Transform invariant low-rank textures," *Int. J. Comput. Vis.*, vol. 99, no. 1, pp. 1–24, 2012.

[29] J. Friedman, T. Hastie, and R. Tibshirani, *The Elements of Statistical Learning* (Springer Series in Statistics), no. 10. New York, NY, USA: Springer, 2001.

[30] A. Mohan and S. Poobal, "Crack detection using image processing: A critical review and analysis," *Alexandria Eng. J.*, vol. 57, no. 2, pp. 787–798, 2018.

[31] F. O'sullivan, "Discretized Laplacian smoothing by Fourier methods," *J. Amer. Stat. Assoc.*, vol. 86, no. 415, pp. 634–642, Sep. 1991.

[32] S. Bagavathiappan, B. Lahiri, T. Saravanan, J. Philip, and T. Jayakumar, "Infrared thermography for condition monitoring—A review," *Infr. Phys. Technol.*, vol. 60, pp. 35–55, Sep. 2013.

[33] A. Wang and J. Shi, "Holistic modeling and analysis of multistage manufacturing processes with sparse effective inputs and mixed profile outputs," *IISE Trans.*, vol. 53, no. 5, pp. 582–596, 2021.

Shancong Mou received the B.S. degree in energy and power engineering from Xi'an Jiaotong University, Xi'an, China, in 2017. He is currently pursuing the Ph.D. degree with the School of Industrial and Systems Engineering, Georgia Tech, Atlanta, GA, USA.

His research interests include data analytics for monitoring, control, and diagnostics of complex engineering systems.

Mr. Mou is also a member of the Institute of Industrial and Systems Engineers (IISE) and the Institute for Operations Research and the Management Sciences (INFORMS).

Andi Wang received the B.S. degree in statistics from Peking University, Beijing, China, in 2012, the Ph.D. degree from The Hong Kong University of Science and Technology, Hong Kong, in 2016, and the Ph.D. degree from the School of Industrial and Systems Engineering, Georgia Tech, Atlanta, GA, USA.

His research interests include advanced statistical modeling, large-scale optimization, and machine learning for system performance improvement.

Dr. Wang is also a member of the Institute of Industrial and Systems Engineers (IISE) and the Institute for Operations Research and the Management Sciences (INFORMS).

Chuck Zhang received the B.S. and M.S. degrees in mechanical engineering from Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 1984 and 1987, respectively, the M.S. degree in industrial engineering from the State University of New York, New York, NY, USA in 1990, and the Ph.D. degree in industrial engineering from The University of Iowa, Iowa City, IA, USA, in 1993.

He is currently a Harold E. Smalley Professor with the School of Industrial and Systems Engineering, Georgia Tech, Atlanta, GA, USA. His research interests include additive manufacturing, advanced composite and nanocomposite manufacturing, and bio-manufacturing.

Dr. Zhang is also a fellow of the Institute of Industrial and Systems Engineers (IISE).

Jianjun Shi received the B.S. and M.S. degrees in automation from the Beijing Institute of Technology, Beijing, China, in 1984 and 1987, respectively, and the Ph.D. degree in mechanical engineering from the University of Michigan, Ann Arbor, MI, USA, in 1992.

He is currently the Carolyn J. Stewart Chair and a Professor with the School of Industrial and Systems Engineering, Georgia Tech, Atlanta, GA, USA.

Dr. Shi is also a fellow of the American Society of Mechanical Engineers (ASME), the Institute of Industrial and Systems Engineers (IISE), the Institute for Operations Research and the Management Sciences (INFORMS), the International Statistical Institute (ISI), and the Society of Manufacturing Engineers (SME), an Academician of the International Academy for Quality (IAQ), and a member of the National Academy of Engineers (NAE).

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on August 03,2021 at 02:18:39 UTC from IEEE Xplore. Restrictions apply.