



Image-Based Prognostics Using Penalized Tensor Regression

Xiaolei Fang, Kamran Paynabar & Nagi Gebraeel

To cite this article: Xiaolei Fang, Kamran Paynabar & Nagi Gebraeel (2019) Image-Based Prognostics Using Penalized Tensor Regression, *Technometrics*, 61:3, 369-384, DOI: [10.1080/00401706.2018.1527727](https://doi.org/10.1080/00401706.2018.1527727)

To link to this article: <https://doi.org/10.1080/00401706.2018.1527727>



View supplementary material [↗](#)



Accepted author version posted online: 26 Nov 2018.
Published online: 08 Mar 2019.



Submit your article to this journal [↗](#)



Article views: 641



View related articles [↗](#)



View Crossmark data [↗](#)



Image-Based Prognostics Using Penalized Tensor Regression

Xiaolei Fang^a, Kamran Paynabar^b, and Nagi Gebraeel^b

^aEdward P. Fitts Department of Industrial and Systems Engineering, North Carolina State University, Raleigh, NC; ^bH. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA

ABSTRACT

This article proposes a new methodology to predict and update the residual useful lifetime of a system using a sequence of degradation images. The methodology integrates tensor linear algebra with traditional location-scale regression widely used in reliability and prognostics. To address the high dimensionality challenge, the degradation image streams are first projected to a low-dimensional tensor subspace that is able to preserve their information. Next, the projected image tensors are regressed against time-to-failure via penalized location-scale tensor regression. The coefficient tensor is then decomposed using CANDECOMP/PARAFAC (CP) and Tucker decompositions, which enables parameter estimation in a high-dimensional setting. Two optimization algorithms with a global convergence property are developed for model estimation. The effectiveness of our models is validated using two simulated datasets and infrared degradation image streams from a rotating machinery.

ARTICLE HISTORY

Received June 2017
Accepted September 2018

KEYWORDS

Image streams;
(Log)-location-scale
distribution; Penalized tensor
regression; Residual useful
lifetimes



1. Introduction

Imaging is one of the fastest growing technologies for condition monitoring and industrial asset management. Relative to most sensing techniques, industrial imaging devices are easier to use because they are generally noncontact and do not require permanent installation or fixturing. Image data also contains rich information about the object being monitored. Some examples of industrial imaging technologies include infrared images used to measure temperature distributions of equipment and components (Bagavathiappan et al. 2013), charge-coupled device (CCD) images, which capture surface quality information (e.g., cracks) of products (Neogi, Mohanta, and Dutta 2014), and others. Image data has been extensively used for process monitoring and diagnostics. For instance, infrared images have been successfully used for civil structures monitoring (Meola 2007), machinery inspection (Seo et al. 2011), fatigue damage evaluation (Pastor et al. 2008), and electronic printed circuit board (PCB) monitoring (Vellvehi et al. 2011). In steel industry, CCD cameras have been used for product surface inspection (Neogi, Mohanta, and Dutta 2014), while video cameras have been used to monitor the shape and color of furnace flames to control quality of steel tubes (Yan, Paynabar and Shi 2015). This article expands the utilization of image data by proposing an image-based prognostic modeling framework that uses degradation-based image streams to predict remaining lifetime.


Numerous prognostic methodologies have been developed in the literature. Examples of some modeling approaches include random coefficients models (Gebraeel et al. 2005; Ye and Chen 2014), models that use the Brownian motion process

(Ye, Chen, and Shen 2015; Chen et al. 2015) and gamma process (Shu, Feng, and Coit 2015; Zhang and Liao 2015), and models based on functional data analysis (Fang, Zhou, and Gebraeel 2015; Zhou et al. 2014; Fang, Paynabar, and Gebraeel 2017). These approaches are well-suited for time-series signals, but it is not clear how they can be extended to model image streams. One of the key challenges in modeling image data revolves around the analytical and computational complexities associated with characterizing high-dimensional data. High dimensionality arises from the fact that a single image stream consists of a large sequence of images (observed across the life cycle of an equipment) coupled with the large numbers of pixels embedded in each image. Additional challenges are related to the complex *spatial-temporal structures* inherent in the data streams. Pixels are spatially correlated within a single image and temporally correlated across sequential images. In recent work (Liu, Yeo, and Kalagnanam 2016), degradation image streams were modeled as a spatio-temporal process. Although spatio-temporal models have been widely used to model data with complex spatial and temporal correlation structures (Cressie and Wikle 2015), they are not necessarily accurate for long-term predictions necessary to our type of prognostic application. Most importantly, a key limitation of spatio-temporal models is that they require a pre-set failure threshold, which is usually hard to define for degradation image streams.

This article proposes a tensor-based regression framework that uses degradation image streams to predict remaining useful life (RUL), and provide advance warning of impending failures of industrial assets. Specifically, we build a (log)-location-scale (LLS) tensor regression model in which the

CONTACT Xiaolei Fang  xfang8@ncsu.edu  Edward P. Fitts Department of Industrial and Systems Engineering, North Carolina State University, Raleigh, NC 27695.

Color versions of one or more of the figures in the article can be found online at www.tandfonline.com/r/TECH.

 Supplementary materials for this article are available online. Please go to www.tandfonline.com/r/TECH.

© 2019 American Statistical Association and the American Society for Quality

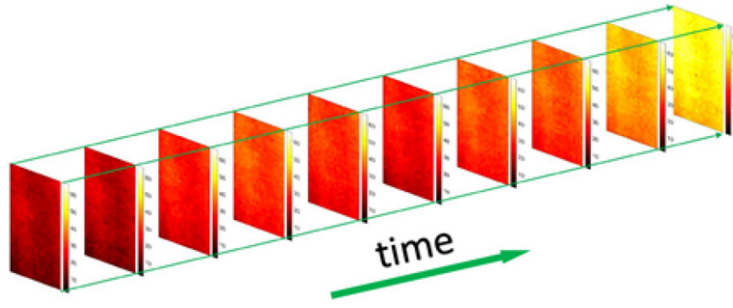


Figure 1. An illustration of a degradation image stream (three-order tensor).

time-to-failure (TTF) is treated as the response and degradation image streams as covariates. LLS regression has been widely used in reliability and survival analysis (Doray 1994) because it provides a flexible framework capable of modeling a variety of TTF distributions such as *(log)normal*, *(log)logistic*, *smallest extreme value (SEV)*, *Weibull*, etc. To model the *spatio-temporal structure* of degradation image streams, the regression model treats each image stream as a *tensor*. A tensor is defined as a *multi-dimensional array*—a one-order tensor is a vector, a two-order tensor is a matrix, and objects of order three or higher are called high-order tensors. More details about tensor theory and applications can be found in a survey paper by Kolda and Bader (2009). As illustrated in Figure 1, a degradation image stream constitutes a three-order tensor in which the first two dimensions capture the spatial structure of a single image, whereas the third dimension is used to model the temporal structure of the image stream. One of the key benefits of modeling a degradation image stream as a tensor is that tensors maintain the *spatio-temporal structure* within and between images which allows for a relatively accurate RUL prediction model. In this article, *degradation image stream(s)* and *degradation tensor(s)* are used exchangeably hereafter.

The high dimensionality of degradation image streams poses significant computational challenges, especially ones related to parameter estimation. For example, a tensor-regression model for a degradation image stream consisting of 50 images each with 20×20 pixels generates a three-order tensor-coefficient consisting of 20,000 elements that need to be estimated. In an effort to improve model computations, we develop two estimation methods that integrate dimensionality reduction and tensor decomposition. Dimensionality reduction is used as the first step for both estimation methods as it helps reduce the number of parameters. Degradation tensors are projected to a low-dimensional tensor subspace that preserves most of their information. This is achieved using a multilinear dimension reduction technique, such as multilinear principal component analysis (MPCA) (Lu, Plataniotis, and Venetsanopoulos 2008). We utilize the fact that essential information embedded in high-dimensional tensors can be captured in a low-dimensional tensor subspace. Next, the tensor-coefficients corresponding to the projected degradation tensors are decomposed using two popular tensor decomposition approaches namely, CANDECOMP/PARAFAC (CP) (Carroll and Chang 1970) and Tucker (1966). The CP approach decomposes a high-dimensional coefficient tensor as a product of several low-rank basis matrices. In contrast, the Tucker approach expresses the tensor-coefficient

as a product of a low-dimensional core tensor and several factor matrices. Therefore, instead of estimating the tensor-coefficient, we only estimate its corresponding core tensors and factor/basis matrices, which significantly reduces the computational complexity and the required sample size. Block relaxation algorithms are also developed for model estimation with guaranteed global convergence to a stationary point.

The remainder of the article is organized as follows. Section 2 provides an overview of the basic notations and definitions in multilinear algebra. Section 3 presents the degradation and prognostic modeling framework. Sections 3.1 and 3.2 discuss the estimation algorithm based on CP decomposition and Tucker decomposition, respectively. In Section 4, we discuss the RUL prediction and realtime updating. The effectiveness of our model is validated using simulated data in Sections 5 and 6 along with real degradation image streams from a rotating machinery in Section 7. Finally, Section 8 is devoted to concluding remarks.

2. Preliminaries

This section presents some basic notations, definitions, and operators in multilinear algebra and tensor analysis that are used throughout the article. Vectors are denoted by lowercase boldface letters, for example, \mathbf{b} , matrices are denoted by upper-case boldface letters, for example, \mathbf{B} , and tensors are denoted by calligraphic letters, for example, \mathcal{B} . The *order* of a tensor is the number of dimensions, also known as *modes* or *ways*. For example, the order of vectors and matrices are 1 and 2, respectively. A D -order tensor is denoted by $\mathcal{B} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$, where I_d for $d = 1, \dots, D$ represents the dimension of the mode- d of \mathcal{B} . The (i_1, i_2, \dots, i_D) th component of \mathcal{B} is denoted by b_{i_1, i_2, \dots, i_D} , where the index $i_d = 1, \dots, I_d$ for $d = 1, \dots, D$. A *fiber* of \mathcal{B} is a vector obtained by fixing all indices of \mathcal{B} but one. A matrix column is a mode-1 fiber and a matrix row is a mode-2 fiber. Third-order tensors have column, row, and tube fibers (see, e.g., Figure 2). A *vectorization* of \mathcal{B} , denoted by $\text{vec}(\mathcal{B})$, is obtained by stacking all mode-1 fibers of \mathcal{B} . The *mode- d matricization* of \mathcal{B} , denoted by $\mathbf{B}_{(d)}$, is a matrix whose columns are mode- d fibers of \mathcal{B} in the lexicographical order. The *mode- d product* of a tensor $\mathcal{B} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ with a matrix $\mathbf{A} \in \mathbb{R}^{J \times I_d}$, denoted by $(\mathcal{B} \times_d \mathbf{A})$, is a tensor whose component is $(\mathcal{B} \times_d \mathbf{A})_{i_1, \dots, i_{d-1}, j, i_{d+1}, \dots, i_D} = \sum_{i_d=1}^{I_d} b_{i_1, i_2, \dots, i_D} a_{j, i_d}$. The *inner product* of two tensors $\mathcal{B} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$, $\mathcal{S} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ is denoted by $\langle \mathcal{B}, \mathcal{S} \rangle = \sum_{i_1, \dots, i_D} b_{i_1, \dots, i_D} s_{i_1, \dots, i_D}$. A rank-one tensor $\mathcal{B} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ can be represented by outer products of vectors, that

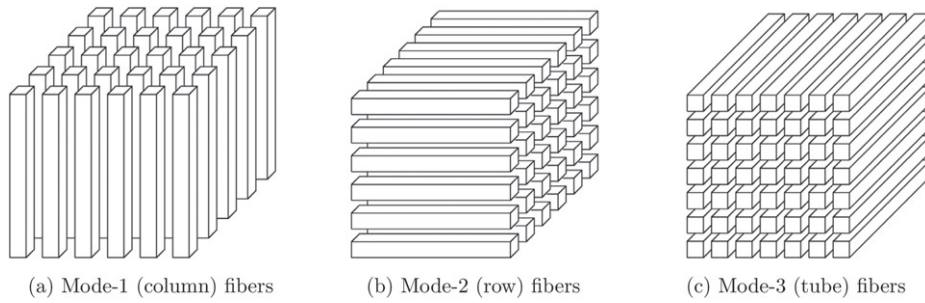


Figure 2. Fibers of a third-order tensor (Kolda and Bader 2009).

is, $\mathcal{B} = \mathbf{b}_1 \circ \mathbf{b}_2 \circ \dots \circ \mathbf{b}_D$, where \mathbf{b}_d is an I_d -dimension vector and “ \circ ” is the *outer product* operator. The *Kronecker product* of two matrices $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{B} \in \mathbb{R}^{p \times q}$, denoted by $\mathbf{A} \otimes \mathbf{B}$ is an $mp \times nq$ block matrix defined by

$$\mathbf{M} = \begin{pmatrix} a_{11}\mathbf{B} & \dots & a_{1n}\mathbf{B} \\ \vdots & \ddots & \vdots \\ a_{m1}\mathbf{B} & \dots & a_{mn}\mathbf{B} \end{pmatrix}.$$

The *Khatri-Rao* product of two matrices $\mathbf{A} \in \mathbb{R}^{m \times r}$, $\mathbf{B} \in \mathbb{R}^{p \times r}$, denoted by $\mathbf{A} \odot \mathbf{B}$, is a $mp \times r$ matrix defined by $[\mathbf{a}_1 \otimes \mathbf{b}_1 \quad \mathbf{a}_2 \otimes \mathbf{b}_2 \quad \dots \quad \mathbf{a}_r \otimes \mathbf{b}_r]$, where $\mathbf{a}_i \in \mathbb{R}^{m \times 1}$, and $\mathbf{b}_i \in \mathbb{R}^{p \times 1}$ for $i = 1, \dots, r$.

3. Prognostic Modeling Using Degradation Tensors

This article considers engineering systems with degradation process that can be represented by tensors, for example, degradation image streams or profiles. The underlying premise of our prognostic modeling framework rests on using LLS regression to model TTF as a function of degradation tensors. One of the main challenges in fitting such regression models is the high-dimensionality of data which makes coefficients estimation intractable. To address this issue, we use the fact that the essential information of high-dimensional data is often embedded in a low-dimensional subspace. Specifically, we project degradation and coefficient tensors onto a low-dimensional tensor subspace that preserves their inherent information.

To further reduce the number of estimated parameters, coefficient tensors are decomposed using two widely used tensor decomposition techniques, CP and Tucker. The CP decomposition expresses a high-dimensional coefficient tensor as a product of several smaller sized basis matrices (Carroll and Chang 1970). Tucker decomposition, however, expresses a high-dimensional coefficient tensor as a product of a low-dimensional core tensor and several factor matrices (Tucker 1966). Thus, instead of estimating the coefficient tensor, we only need to estimate its corresponding core tensors and factor/basis matrices, which significantly helps reduce the computational complexity and the required sample for estimation. The parameters of the reduced LLS regression model are estimated using the maximum likelihood (ML) approach. To obtain the ML estimates, we propose optimization algorithms for CP-based and Tucker-based methods. The optimization algorithms are based on the block relaxation method (De Leeuw 1994; Lange 2010), which alternately updates one block of the parameters while keeping other parameters fixed. Finally, the estimated LLS regression is used to predict and update the RUL of a

functioning system. In the following, the details of the proposed methodology is presented.

Our framework is applicable in settings that have a historical dataset of degradation image streams (i.e., degradation tensor) for a sample of units with corresponding TTFs. Let n denote the number of units that make up the historical (training) sample. Let $\mathcal{S}_i \in \mathbb{R}^{q_1 \times q_2 \times q_3}$, for $i = 1, \dots, n$, denote the degradation tensor and \tilde{y}_i represent the TTF. The following LLS regression model expresses the TTF as a function of a degradation tensor:

$$y_i = \alpha + \langle \mathcal{B}, \mathcal{S}_i \rangle + \sigma \epsilon_i, \quad (1)$$

where $y_i = \tilde{y}_i$ for a location-scale model and $y_i = \ln(\tilde{y}_i)$ for a log-location-scale model, the scalar α is the intercept of the regression model, and $\mathcal{B} \in \mathbb{R}^{q_1 \times q_2 \times q_3}$ is the tensor of regression coefficients. $\alpha + \langle \mathcal{B}, \mathcal{S}_i \rangle$ is known as the location parameter and σ is the scale parameter. Similar to common LLS regression models (Doray 1994), we assume that only the location parameter is a function of the covariates, that is, the degradation tensor. The term ϵ_i is the random noise term with a standard location-scale density $f(\epsilon)$. For example, $f(\epsilon) = \exp(\epsilon - \exp(\epsilon))$ for SEV distribution, $f(\epsilon) = \exp(\epsilon)/(1 + \exp(\epsilon))^2$ for logistic distribution, and $f(\epsilon) = 1/\sqrt{2\pi} \exp(-\epsilon^2/2)$ for normal distribution. Consequently, y_i has a density in the form of $\frac{1}{\sigma} f\left(\frac{y_i - \alpha - \langle \mathcal{B}, \mathcal{S}_i \rangle}{\sigma}\right)$.

The number of parameters in Model (1) is given by $2 + \prod_{d=1}^3 q_d$. Recall that q_d represents the dimension of the d -mode of \mathcal{B} . If we consider a simple example of an image stream constituting 100 images of size 40×50 , that is, \mathcal{S}_i is a three-order tensor in $\mathbb{R}^{40 \times 50 \times 100}$, the number of parameters to be estimated will be quite large: $200,002 = 2 + 40 \times 50 \times 100$. To reduce the number of parameters, as mentioned earlier, we project the degradation tensors and the coefficient tensor onto a low-dimensional tensor subspace that captures the relevant information of the degradation tensors. The following proposition shows that by using multilinear dimension reduction techniques, we can significantly reduce the dimensions of the coefficient tensor without significant loss of information.

Proposition 1. Suppose $\{\mathcal{S}_i\}_{i=1}^n$ can be expanded by $\mathcal{S}_i = \tilde{\mathcal{S}}_i \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2 \times_3 \mathbf{U}_3$, where $\tilde{\mathcal{S}}_i \in \mathbb{R}^{p_1 \times p_2 \times p_3}$ is a low-dimensional tensor and matrices $\mathbf{U}_d \in \mathbb{R}^{p_d \times q_d}$, $\mathbf{U}_d^\top \mathbf{U}_d = \mathbf{I}_{p_d}$, $p_d < q_d$, $d = 1, 2, 3$. If the coefficient tensor, \mathcal{B} , is projected onto the tensor subspace spanned by $\{\mathbf{U}_1, \mathbf{U}_2, \mathbf{U}_3\}$, that is, $\tilde{\mathcal{B}} = \mathcal{B} \times_1 \mathbf{U}_1^\top \times_2 \mathbf{U}_2^\top \times_3 \mathbf{U}_3^\top$, where $\tilde{\mathcal{B}}$ is the projected coefficient tensor, then $\langle \mathcal{B}, \mathcal{S}_i \rangle = \langle \tilde{\mathcal{B}}, \tilde{\mathcal{S}}_i \rangle$.

The proof of [Proposition 1](#) is given in online Appendix A (supplementary materials). [Proposition 1](#) implies that the original high-dimensional tensors, (i.e., \mathcal{B} and \mathcal{S}) and their corresponding low-rank projections (i.e., $\tilde{\mathcal{B}}$ and $\tilde{\mathcal{S}}_i$) result in similar estimates of the location parameter. Using [Proposition 1](#), we can reexpress Equation (1) as follows:

$$y_i = \alpha + \langle \tilde{\mathcal{B}}, \tilde{\mathcal{S}}_i \rangle + \sigma \epsilon_i. \quad (2)$$

The low-dimensional tensor space defined by factor matrices $\mathbf{U}_d \in \mathbb{R}^{p_d \times q_d}$ can be obtained by applying multilinear dimension reduction techniques, such as multilinear principal component analysis (MPCA) (Lu, Plataniotis, and Venetsanopoulos 2008), on the training degradation tensor, $\{\mathcal{S}_i\}_{i=1}^n$. The objective of MPCA is to construct a low-dimensional tensor subspace and project a set of high-dimensional tensors onto the subspace such that the variation among the projected low-dimensional tensors is maximized. To be specific, let $\{\mathcal{S}_i\}_{i=1}^n$ denote the set of high-dimensional tensors and $\{\mathbf{U}_d \in \mathbb{R}^{p_d \times q_d}, \mathbf{U}_d^\top \mathbf{U}_d = \mathbf{I}_{q_d}, p_d < q_d\}_{d=1}^3$ be a set of orthogonal factor matrices that span the low-dimensional tensor subspace. Then, the set of projected low-dimensional tensors can be denoted by $\{\mathcal{S}_i \times_1 \mathbf{U}_1^\top \times_2 \mathbf{U}_2^\top \times_3 \mathbf{U}_3^\top\}_{i=1}^n$ and the variation among the projected tensors is $\|(\mathcal{S}_i - \bar{\mathcal{S}}) \times_1 \mathbf{U}_1^\top \times_2 \mathbf{U}_2^\top \times_3 \mathbf{U}_3^\top\|_F^2$. Here, $\bar{\mathcal{S}} = \frac{1}{n} \sum_{i=1}^n \mathcal{S}_i$ is the mean tensor and $\|\cdot\|_F^2$ is the Frobenius norm. As a result, MPCA can be mathematically formalized into the following optimization problem:

$$\begin{aligned} & \{\hat{\mathbf{U}}_1, \hat{\mathbf{U}}_2, \hat{\mathbf{U}}_3\} \\ &= \arg \max_{\mathbf{U}_1, \mathbf{U}_2, \mathbf{U}_3} \sum_{i=1}^n \left\| (\mathcal{S}_i - \bar{\mathcal{S}}) \times_1 \mathbf{U}_1^\top \times_2 \mathbf{U}_2^\top \times_3 \mathbf{U}_3^\top \right\|_F^2. \end{aligned} \quad (3)$$

Here, $\{\hat{\mathbf{U}}_1, \hat{\mathbf{U}}_2, \hat{\mathbf{U}}_3\}$ can be estimated by applying the algorithm given in online Appendix B (supplementary materials) to the training dataset (historical degradation tensors). The algorithm in online Appendix B (supplementary materials) requires the dimensionality of the tensor subspace (i.e., $\{p_d\}_{d=1}^3$) be known in advance. In practice, one widely used criterion to determine the dimensionality is the Q-based method (Lu, Plataniotis, and Venetsanopoulos 2008), which is similar to the fraction-of-variance-explained (FVE) criterion for principal component number selection in regular PCA (Fang, Gebraeel, and Paynabar 2017). Specifically, to determine p_d , the centered training degradation tensors, $\{\tilde{\mathcal{S}}_i = \mathcal{S}_i - \bar{\mathcal{S}}\}_{i=1}^n$, are first unfolded to matrices (denoted by $\{\tilde{\mathcal{S}}_{i(d)}\}_{i=1}^n$) by applying mode- d matricization. Next, the covariance matrix of the unfolded matrices is computed using $\frac{1}{n} \sum_{i=1}^n \{\tilde{\mathcal{S}}_{i(d)} \tilde{\mathcal{S}}_{i(d)}^\top\}$, and its eigen values (denoted by $\{\lambda_j\}_{j=1}^k$, k is the number of eigen values) are calculated via eigen decomposition. Then the variation explained by the first p_d eigen values is $\rho_{p_d} = \sum_{j=1}^{p_d} \lambda_j / \sum_{j=1}^k \lambda_j$, and p_d can be selected such that ρ_{p_d} is greater than a specific number, say 0.95. More details regarding the Q-method and some other algorithms used to determine the dimensionality of the tensor subspace can be found in Lu, Plataniotis, and Venetsanopoulos (2008).

It should be pointed out that if the image stream is noiseless and embedded in a low-dimensional subspace meaning that the corresponding tensor is low rank, then multilinear dimension reduction techniques will not result in any information loss.

However, in practice, image streams often contain noise, so multilinear dimension reduction techniques will cause some information loss. However, the lost information is mainly associated with the noise, which is not important in predicting RUL. Multilinear dimension reduction techniques help reduce the number of parameters to be estimated from $2 + \prod_{d=1}^3 q_d$ in Equation (1) to $2 + \prod_{d=1}^3 p_d$ in Equation (2), where $2 + \prod_{d=1}^3 p_d \ll 2 + \prod_{d=1}^3 q_d$. However, often, the number of reduced parameters (i.e., $2 + \prod_{d=1}^3 p_d$) is still so large that requires further dimension reduction. For example, for a $40 \times 50 \times 100$ tensor, if $p_1 = p_2 = p_3 = 10$, the number of parameters is reduced from 200,002 to 1002. To further reduce the number of parameters so that they can be estimated by using a limited training sample, we utilize two well-known tensor decomposition techniques namely, CP and Tucker decompositions. We briefly review these decompositions in Sections 3.1 and 3.2, and discuss how they are incorporated into our prognostic framework.

3.1. Dimension Reduction via CP Decomposition

In CP decomposition, the coefficient tensor $\tilde{\mathcal{B}}$ in Equation (2) is decomposed into a sum product of a set of rank one vectors. Given the rank of $\tilde{\mathcal{B}}$, which we denote by k , we have the following decomposition,

$$\tilde{\mathcal{B}} = \sum_{r=1}^k \tilde{\beta}_1^{(r)} \circ \tilde{\beta}_2^{(r)} \circ \tilde{\beta}_3^{(r)}, \quad (4)$$

where $\tilde{\beta}_d^{(r)} = [\tilde{\beta}_{d,1}^{(r)}, \dots, \tilde{\beta}_{d,p_d}^{(r)}]^\top \in \mathbb{R}^{p_d}$, and “ \circ ” denotes the outer product operator. It can be easily shown that $\text{vec}(\tilde{\mathcal{B}}) = (\tilde{\mathcal{B}}_3 \odot \tilde{\mathcal{B}}_2 \odot \tilde{\mathcal{B}}_1) \mathbf{1}_k$, where $\tilde{\mathcal{B}}_d = [\tilde{\beta}_d^{(1)}, \dots, \tilde{\beta}_d^{(k)}] \in \mathbb{R}^{p_d \times k}$ for $d = 1, 2, 3$ and $\mathbf{1}_k \in \mathbb{R}^k$ is a k -dimensional vector of ones. Thus, Equation (2) can be reexpressed as follows:

$$\begin{aligned} y_i &= \alpha + \langle \text{vec}(\tilde{\mathcal{B}}), \text{vec}(\tilde{\mathcal{S}}_i) \rangle + \sigma \epsilon_i \\ &= \alpha + \langle (\tilde{\mathcal{B}}_3 \odot \tilde{\mathcal{B}}_2 \odot \tilde{\mathcal{B}}_1) \mathbf{1}_k, \text{vec}(\tilde{\mathcal{S}}_i) \rangle + \sigma \epsilon_i. \end{aligned} \quad (5)$$

The number of parameters in Equation (5) is $2 + \sum_{d=1}^3 p_d \times k$, which is significantly smaller than $2 + \prod_{d=1}^3 p_d$ from (2). In our 3-order tensor example, if $p_1 = p_2 = p_3 = 10$ and the rank $k = 2$, the number of parameters decreases from 1002 to $62 = 2 + 10 \times 2 + 10 \times 2 + 10 \times 2$.

3.1.1. Parameter Estimation for CP Decomposition

To estimate the parameters of Equation (5) using MLE, we maximize the corresponding penalized log-likelihood function:

$$\begin{aligned} & \arg \max_{\theta} \left\{ \ell(\theta) - \sum_{d=1}^3 r(\tilde{\mathcal{B}}_d) \right\} \\ &= \arg \max_{\theta} \left\{ -n \ln \sigma + \sum_{i=1}^n \ln f \left(\frac{y_i - \alpha - \langle (\tilde{\mathcal{B}}_3 \odot \tilde{\mathcal{B}}_2 \odot \tilde{\mathcal{B}}_1) \mathbf{1}_k, \text{vec}(\tilde{\mathcal{S}}_i) \rangle}{\sigma} \right) - \sum_{d=1}^3 r(\tilde{\mathcal{B}}_d) \right\}, \end{aligned} \quad (6)$$

where $\theta = (\alpha, \sigma, \tilde{\mathbf{B}}_1, \tilde{\mathbf{B}}_2, \tilde{\mathbf{B}}_3)$ and $r(\tilde{\mathbf{B}}_d) = \lambda_d \sum_{r=1}^k \sum_{q=1}^{p_d} \|\tilde{\beta}_{d,q}^{(r)}\|_1$. The ℓ_1 -norm penalty term encourages the sparsity of $\tilde{\mathbf{B}}$, which helps avoid over-fitting.

The block relaxation method proposed by De Leeuw (1994) and Lange (2010) is used to maximize expression (6). Specifically, we iteratively update a block of parameters, say $(\tilde{\mathbf{B}}_d, \sigma, \alpha)$, while keeping other components $\{\tilde{\mathbf{B}}_{\neq d}\}$ fixed. In each update, the optimization criterion is reduced to $\arg \max_{\tilde{\mathbf{B}}_d, \sigma, \alpha} \{\ell(\theta) - r(\tilde{\mathbf{B}}_d)\}$.

Next, we show in Proposition 2 that the optimization problem for each block $\tilde{\mathbf{B}}_d$ is equivalent to optimizing the penalized log-likelihood function for $y_i = \alpha + \langle \tilde{\mathbf{B}}_d, \mathbf{X}_{d,i} \rangle + \sigma \epsilon_i$, where $\tilde{\mathbf{B}}_d$ is the parameter matrix; $\mathbf{X}_{d,i}$ is the predictor matrix defined as follows:

$$\mathbf{X}_{d,i} = \begin{cases} \tilde{\mathbf{S}}_{i(1)}(\tilde{\mathbf{B}}_3 \odot \tilde{\mathbf{B}}_2), & \text{if } d = 1 \\ \tilde{\mathbf{S}}_{i(2)}(\tilde{\mathbf{B}}_3 \odot \tilde{\mathbf{B}}_1), & \text{if } d = 2 \\ \tilde{\mathbf{S}}_{i(3)}(\tilde{\mathbf{B}}_2 \odot \tilde{\mathbf{B}}_1), & \text{if } d = 3 \end{cases}, \quad (7)$$

where $\tilde{\mathbf{S}}_{i(d)}$ is the mode- d matricization of $\tilde{\mathbf{S}}_i$ (defined in the Preliminaries Section).

Proposition 2. Consider the optimization problem in (6), given other parameters except $(\tilde{\mathbf{B}}_d, \sigma, \alpha)$, the optimization problem can be reduced to

$$\arg \max_{\tilde{\mathbf{B}}_d, \sigma, \alpha} \left\{ -n \ln \sigma + \sum_{i=1}^n \ln f \left(\frac{y_i - \alpha - \langle \tilde{\mathbf{B}}_d, \mathbf{X}_{d,i} \rangle}{\sigma} \right) - r(\tilde{\mathbf{B}}_d) \right\}. \quad (8)$$

The proof of Proposition 2 is provided in online Appendix C (supplementary materials). As pointed out by Städler, Bühlmann, and Geer (2010), the estimates of $\alpha, \tilde{\mathbf{B}}_d, \sigma$ in optimizing problem (8) are not invariant under scaling of the response. To be specific, consider the transformation $y'_i = by_i, \alpha' = b\alpha, \tilde{\mathbf{B}}'_d = b\tilde{\mathbf{B}}_d, \sigma' = b\sigma$ where $b > 0$. Clearly, this transformation does not affect the regression model $y_i = \alpha + \langle \tilde{\mathbf{B}}_d, \mathbf{X}_{d,i} \rangle + \sigma \epsilon_i$. Therefore, invariant estimates based on the transformed data $(y'_i, \mathbf{X}_{d,i})$, should satisfy $\hat{\alpha}' = b\hat{\alpha}, \hat{\mathbf{B}}'_d = b\hat{\mathbf{B}}_d, \hat{\sigma}' = b\hat{\sigma}$, where $\hat{\alpha}, \hat{\mathbf{B}}_d, \hat{\sigma}$ are estimates based on original data $(y_i, \mathbf{X}_{d,i})$. However, this does not hold for the estimates obtained by optimizing (8). To address this issue, expression (8) is modified by dividing the penalty term by the scale parameter σ :

$$\arg \max_{\tilde{\mathbf{B}}_d, \sigma, \alpha} \left\{ -n \ln \sigma + \sum_{i=1}^n \ln f \left(\frac{y_i - \alpha - \langle \tilde{\mathbf{B}}_d, \mathbf{X}_{d,i} \rangle}{\sigma} \right) - r \left(\frac{\tilde{\mathbf{B}}_d}{\sigma} \right) \right\}. \quad (9)$$

We can show that the resulting estimates possess the invariant property (see online Appendix D, supplementary materials). Note that in the modified problem, the penalty term penalizes the ℓ_1 -norm of the coefficients and the scale parameter σ simultaneously, which has some close relations to the Bayesian Lasso (Park and Casella 2008; Städler, Bühlmann, and Geer 2010). The log-likelihood function in (9) is not concave which

causes computational problems. We use the following reparameterization to transform the optimization function to a concave function: $\alpha_0 = \alpha/\sigma, \mathbf{A}_d = \tilde{\mathbf{B}}_d/\sigma, \rho = 1/\sigma$. Consequently, the optimization problem can be rewritten as:

$$\arg \max_{\mathbf{A}_d, \rho, \alpha_0} \left\{ n \ln \rho + \sum_{i=1}^n \ln f(\rho y_i - \alpha_0 - \langle \mathbf{A}_d, \mathbf{X}_{d,i} \rangle) - r(\mathbf{A}_d) \right\}. \quad (10)$$

The optimization problem in (10) is concave if function $f(\cdot)$ is log-concave, which is the case for most LLS distributions including *normal*, *logistic*, *SEV*, *generalized log-gamma*, *log-inverse Gaussian* (Doray 1994). *Lognormal*, *log-logistic*, and *Weibull* distributions whose density function is not log-concave can easily be transformed to *normal*, *logistic* and *SEV* distributions, respectively, by taking the logarithm of the TTF. Various optimization algorithms such as coordinate descent (Friedman et al. 2007) and gradient descent (Tseng 2001) can be used for solving (10). Algorithm 1 shows the steps of the block relaxation method for optimizing (10) and finding the ML estimates of the parameters.

Algorithm 1 Block relaxation algorithm for solving problem(6).

Input: $\{\tilde{\mathbf{S}}_i, y_i\}_{i=1}^n$ and rank k

Initialization: Matrices $\tilde{\mathbf{B}}_2^{(0)}, \tilde{\mathbf{B}}_3^{(0)}$ are initialized randomly.

while convergence criterion not met **do**

for $d = 1, \dots, 3$ **do**

$$\mathbf{X}_{d,i}^{(j+1)} = \begin{cases} \tilde{\mathbf{S}}_{i(1)}(\tilde{\mathbf{B}}_3^{(j)} \odot \tilde{\mathbf{B}}_2^{(j)}), & \text{if } d = 1 \\ \tilde{\mathbf{S}}_{i(2)}(\tilde{\mathbf{B}}_3^{(j)} \odot \tilde{\mathbf{B}}_1^{(j+1)}), & \text{if } d = 2 \\ \tilde{\mathbf{S}}_{i(3)}(\tilde{\mathbf{B}}_2^{(j+1)} \odot \tilde{\mathbf{B}}_1^{(j+1)}), & \text{if } d = 3 \end{cases}$$

$$\mathbf{A}_d^{(j+1)}, \rho^{(j+1)}, \alpha_0^{(j+1)} = \arg \max_{\mathbf{A}_d, \rho, \alpha_0} \left\{ n \ln \rho + \sum_{i=1}^n \ln f(\rho y_i - \alpha_0 - \langle \mathbf{A}_d, \mathbf{X}_{d,i}^{(j+1)} \rangle) - r(\mathbf{A}_d) \right\}$$

$$\tilde{\mathbf{B}}_d^{(j+1)} = \mathbf{A}_d^{(j+1)} / \rho^{(j+1)}$$

end for

 Let $j := j + 1$

end while

Output: $\alpha = \alpha_0/\rho, \sigma = 1/\rho, \{\tilde{\mathbf{B}}_d\}_{d=1}^3$

The convergence criterion is defined by $\ell(\tilde{\theta}^{(j+1)}) - \ell(\tilde{\theta}^{(j)}) < \epsilon$, in which $\ell(\tilde{\theta}^{(j)})$, is defined as follows:

$$\begin{aligned} \ell(\tilde{\theta}^{(j)}) = & -n \ln \sigma^{(j)} + \sum_{i=1}^n \ln f \left(\frac{y_i - \alpha^{(j)} - \left((\tilde{\mathbf{B}}_3^{(j)} \odot \tilde{\mathbf{B}}_2^{(j)} \odot \tilde{\mathbf{B}}_1^{(j)}) \mathbf{1}_k, \text{vec}(\tilde{\mathbf{S}}_i) \right)}{\sigma^{(j)}} \right) \\ & - \sum_{d=1}^3 r \left(\frac{\tilde{\mathbf{B}}_d^{(j)}}{\sigma^{(j)}} \right), \end{aligned} \quad (11)$$

where $\tilde{\theta}^{(j)} = (\alpha^{(j)}, \sigma^{(j)}, \tilde{\mathbf{B}}_1^{(j)}, \tilde{\mathbf{B}}_2^{(j)}, \tilde{\mathbf{B}}_3^{(j)})$.

It can be shown that Algorithm 1 exhibits the global convergence property (see Proposition 1 in Zhou, Li, and Zhu 2013). In other words, it will converge to a stationary point for any initial

point. Since a stationary point is only guaranteed to be a local maximum or saddle point, the algorithm is run several times with different initializations while recording the best results.

Algorithm 1 requires the rank of $\tilde{\mathbf{B}}$ to be known in advance for CP decomposition. In this article, the Bayesian information criterion (BIC) is used to determine the appropriate rank. The BIC is defined as $-2\ell(\tilde{\boldsymbol{\theta}}) + m \ln(n)$, where ℓ is the log-likelihood value defined in Equation (11), n is the sample size (number of systems) and m is the number of effective parameters. Here, $m = k(\sum_{d=1}^3 p_d - 2)$, where $k(-2)$ is used for the scaling indeterminacy in the CP decomposition (Li, Zhou, and Li 2013).

3.2. Dimension Reduction via Tucker decomposition

Tucker decomposition is the second tensor decomposition approach used in this article. It is used to reduce the dimensionality of $\tilde{\mathbf{B}}$ as a product of a low-dimensional core tensor and a set of factor matrices as follows:

$$\begin{aligned} \tilde{\mathbf{B}} &= \tilde{\mathcal{G}} \times_1 \tilde{\mathbf{B}}_1 \times_2 \tilde{\mathbf{B}}_2 \times_3 \tilde{\mathbf{B}}_3 \\ &= \sum_{r_1=1}^{k_1} \sum_{r_2=1}^{k_2} \sum_{r_3=1}^{k_3} \tilde{g}_{r_1, r_2, r_3} \tilde{\boldsymbol{\beta}}_1^{(r_1)} \circ \tilde{\boldsymbol{\beta}}_2^{(r_2)} \circ \tilde{\boldsymbol{\beta}}_3^{(r_3)}, \end{aligned} \quad (12)$$

where $\tilde{\mathcal{G}} \in \mathbb{R}^{k_1 \times k_2 \times k_3}$ is the core tensor with element $(\tilde{\mathcal{G}})_{r_1, r_2, r_3} = \tilde{g}_{r_1, r_2, r_3}$, $\tilde{\mathbf{B}}_d = [\tilde{\boldsymbol{\beta}}_d^{(1)}, \dots, \tilde{\boldsymbol{\beta}}_d^{(k_d)}] \in \mathbb{R}^{p_d \times k_d}$ for $d = 1, 2, 3$ is the factor matrix, " \times_d " is the mode- d product operator, and " \circ " is the outer product operator. Using this decomposition, Equation (2) can be reexpressed as follows:

$$\begin{aligned} y_i &= \alpha + \langle \tilde{\mathbf{B}}, \tilde{\mathcal{S}}_i \rangle + \sigma \epsilon_i \\ &= \alpha + \langle \tilde{\mathcal{G}} \times_1 \tilde{\mathbf{B}}_1 \times_2 \tilde{\mathbf{B}}_2 \times_3 \tilde{\mathbf{B}}_3, \tilde{\mathcal{S}}_i \rangle + \sigma \epsilon_i. \end{aligned} \quad (13)$$

3.2.1. Parameter Estimation for Tucker Decomposition

The following penalized log-likelihood function is used to compute the ML estimates of the parameters in expression (13).

$$\begin{aligned} \arg \max_{\boldsymbol{\theta}} & \left\{ \ell(\boldsymbol{\theta}) - r(\tilde{\mathcal{G}}) - \sum_{d=1}^3 r(\tilde{\mathbf{B}}_d) \right\} \\ &= \arg \max_{\boldsymbol{\theta}} \left\{ -n \ln \sigma + \sum_{i=1}^n \ln f \left(\frac{y_i - \alpha - \langle \tilde{\mathcal{G}} \times_1 \tilde{\mathbf{B}}_1 \times_2 \tilde{\mathbf{B}}_2 \times_3 \tilde{\mathbf{B}}_3, \tilde{\mathcal{S}}_i \rangle}{\sigma} \right) \right. \\ & \quad \left. - r(\tilde{\mathcal{G}}) - \sum_{d=1}^3 r(\tilde{\mathbf{B}}_d) \right\}, \end{aligned} \quad (14)$$

where $\boldsymbol{\theta} = (\alpha, \sigma, \tilde{\mathcal{G}}, \tilde{\mathbf{B}}_1, \tilde{\mathbf{B}}_2, \tilde{\mathbf{B}}_3)$, $r(\tilde{\mathcal{G}}) = \lambda \sum_{r_1=1}^{k_1} \sum_{r_2=1}^{k_2} \sum_{r_3=1}^{k_3} \|\tilde{g}_{r_1, r_2, r_3}\|_1$, and $r(\tilde{\mathbf{B}}_d) = \lambda_d \sum_{r_d=1}^{k_d} \sum_{q=1}^{p_d} \|\tilde{\beta}_{d,q}^{(r_d)}\|_1$.

Similar to the CP decomposition model, the block relaxation method is used to solve expression (14). To update the core tensor $\tilde{\mathcal{G}}$ given all the factor matrices, the optimization criterion is reduced to $\arg \max_{\tilde{\mathcal{G}}} \left\{ \ell(\boldsymbol{\theta}) - r(\tilde{\mathcal{G}}) \right\}$. **Proposition 3** shows that this optimization problem is equivalent to optimizing the penalized log-likelihood function of $y_i = \alpha + \langle \text{vec}(\tilde{\mathcal{G}}), \mathbf{x}_i \rangle + \sigma \epsilon_i$,

where $\text{vec}(\tilde{\mathcal{G}})$ is the parameter vector and \mathbf{x}_i is the predictor vector defined by $\mathbf{x}_i = (\tilde{\mathbf{B}}_3 \otimes \tilde{\mathbf{B}}_2 \otimes \tilde{\mathbf{B}}_1)^\top \text{vec}(\tilde{\mathcal{S}}_i)$.

Proposition 3. Consider the optimization problem in (14), given $\{\tilde{\mathbf{B}}_1, \tilde{\mathbf{B}}_2, \tilde{\mathbf{B}}_3\}$, the optimization problem is reduced to

$$\arg \max_{\tilde{\mathcal{G}}} \left\{ -n \ln \sigma + \sum_{i=1}^n \ln f \left(\frac{y_i - \alpha - \langle \text{vec}(\tilde{\mathcal{G}}), (\tilde{\mathbf{B}}_3 \otimes \tilde{\mathbf{B}}_2 \otimes \tilde{\mathbf{B}}_1)^\top \text{vec}(\tilde{\mathcal{S}}_i) \rangle}{\sigma} \right) - r(\tilde{\mathcal{G}}) \right\}, \quad (15)$$

The proof of **Proposition 3** is given in online Appendix E (supplementary materials). To guarantee the invariance property of the estimates and concavity of the optimization function, we apply the following re-parameterization: $\rho = 1/\sigma$, $\alpha_0 = \alpha/\sigma$, $\mathcal{C} = \tilde{\mathcal{G}}/\sigma$, $r(\mathcal{C}) = \lambda \sum_{r_1=1}^{k_1} \sum_{r_2=1}^{k_2} \sum_{r_3=1}^{k_3} \frac{\|\tilde{g}_{r_1, r_2, r_3}\|_1}{\sigma}$. This enables us to reexpress criterion (15) as follows:

$$\arg \max_{\mathcal{C}, \rho, \alpha_0} \left\{ n \ln \rho + \sum_{i=1}^n \ln f \left(\rho y_i - \alpha_0 - \langle \text{vec}(\mathcal{C}), (\tilde{\mathbf{B}}_3 \otimes \tilde{\mathbf{B}}_2 \otimes \tilde{\mathbf{B}}_1)^\top \text{vec}(\tilde{\mathcal{S}}_i) \rangle \right) - r(\mathcal{C}) \right\}. \quad (16)$$

To update the factor matrix $\tilde{\mathbf{B}}_d$, we fix the core tensor $\tilde{\mathcal{G}}$ and the rest of the factor matrices $\{\tilde{\mathbf{B}}_{\neq d}\}$, and maximize the following criterion $\arg \max_{\tilde{\mathbf{B}}_d} \left\{ \ell(\boldsymbol{\theta}) - r(\tilde{\mathbf{B}}_d) \right\}$. **Proposition 4** shows that this optimization problem is equivalent to optimizing the log-likelihood function of $y_i = \alpha + \langle \tilde{\mathbf{B}}_d, \mathbf{X}_{d,i} \rangle + \sigma \epsilon_i$, where $\tilde{\mathbf{B}}_d$ is the parameter matrix; $\mathbf{X}_{d,i}$ is the predictor matrix defined as follows:

$$\mathbf{X}_{d,i} = \begin{cases} \tilde{\mathcal{S}}_{i(1)} (\tilde{\mathbf{B}}_3 \otimes \tilde{\mathbf{B}}_2) \tilde{\mathbf{G}}_{(1)}^\top, & \text{if } d = 1 \\ \tilde{\mathcal{S}}_{i(2)} (\tilde{\mathbf{B}}_3 \otimes \tilde{\mathbf{B}}_1) \tilde{\mathbf{G}}_{(2)}^\top, & \text{if } d = 2 \\ \tilde{\mathcal{S}}_{i(3)} (\tilde{\mathbf{B}}_2 \otimes \tilde{\mathbf{B}}_1) \tilde{\mathbf{G}}_{(3)}^\top, & \text{if } d = 3 \end{cases}, \quad (17)$$

where $\tilde{\mathcal{S}}_{i(d)}$ and $\tilde{\mathbf{G}}_{(d)}$ are the mode- d matricization of $\tilde{\mathcal{S}}_i$ and $\tilde{\mathcal{G}}$, respectively.

Proposition 4. Consider the problem in (14), given $\tilde{\mathcal{G}}$ and $\{\tilde{\mathbf{B}}_{\neq d}\}$, the optimization problem is reduced to

$$\arg \max_{\tilde{\mathbf{B}}_d} \left\{ -n \ln \sigma + \sum_{i=1}^n \ln f \left(\frac{y_i - \alpha - \langle \tilde{\mathbf{B}}_d, \mathbf{X}_{d,i} \rangle}{\sigma} \right) - r(\tilde{\mathbf{B}}_d) \right\}. \quad (18)$$

The proof of **Proposition 4** is provided in online Appendix F (supplementary materials). Similar to expression (9), we use penalty term $r(\frac{\tilde{\mathbf{B}}_d}{\sigma})$ and let $\rho = 1/\sigma$, $\alpha_0 = \alpha/\sigma$, $\mathbf{A}_d = \tilde{\mathbf{B}}_d/\sigma$. Consequently, we obtain the following optimization subproblem for parameter estimation:

$$\arg \max_{\mathbf{A}_d} \left\{ n \ln \rho + \sum_{i=1}^n \ln f \left(\rho y_i - \alpha_0 - \langle \mathbf{A}_d, \mathbf{X}_{d,i} \rangle \right) - r(\mathbf{A}_d) \right\}. \quad (19)$$

Algorithm 2 Block relaxation algorithm for solving problem (13).

Input: $\{\tilde{S}_i, y_i\}_{i=1}^n$ and rank $\{k_d\}_{d=1}^3$

Initialization: Core tensor $\mathcal{G}^{(0)}$ and matrices $\tilde{\mathbf{B}}_2^{(0)}, \tilde{\mathbf{B}}_3^{(0)}$ are initialized randomly.

while convergence criterion not met **do**

for $d = 1, \dots, 3$ **do**

$$\mathbf{X}_{d,i}^{(j+1)} = \begin{cases} \tilde{S}_{i(1)}(\tilde{\mathbf{B}}_3^{(j)} \otimes \tilde{\mathbf{B}}_2^{(j)})\{\tilde{\mathbf{G}}_{(1)}^{(j)}\}^\top, & \text{if } d = 1 \\ \tilde{S}_{i(2)}(\tilde{\mathbf{B}}_3^{(j)} \otimes \tilde{\mathbf{B}}_1^{(j+1)})\{\tilde{\mathbf{G}}_{(2)}^{(j)}\}^\top, & \text{if } d = 2 \\ \tilde{S}_{i(3)}(\tilde{\mathbf{B}}_2^{(j+1)} \otimes \tilde{\mathbf{B}}_1^{(j+1)})\{\tilde{\mathbf{G}}_{(3)}^{(j)}\}^\top, & \text{if } d = 3 \end{cases}$$

$$\mathbf{A}_d^{(j+1)}, \rho^{(j+1)}, \alpha_0^{(j+1)}$$

$$= \arg \max_{\mathbf{A}_d, \rho, \alpha_0} \left\{ n \ln \rho + \sum_{i=1}^n \ln f(\rho y_i - \alpha_0 - \langle \mathbf{A}_d, \mathbf{X}_{d,i}^{(j+1)} \rangle) - r(\mathbf{A}_d) \right\}$$

$$\tilde{\mathbf{B}}_d^{(j+1)} = \mathbf{A}_d^{(j+1)} / \rho^{(j+1)}$$

end for

$$\mathcal{C}^{(j+1)}, \rho^{(j+1)}, \alpha_0^{(j+1)}$$

$$= \arg \max_{\mathcal{C}, \rho, \alpha_0} \left\{ n \ln \rho + \sum_{i=1}^n \ln f(\rho y_i - \alpha_0 - \langle \text{vec}(\mathcal{C}), (\tilde{\mathbf{B}}_3^{(j+1)} \otimes \tilde{\mathbf{B}}_2^{(j+1)} \otimes \tilde{\mathbf{B}}_1^{(j+1)})^\top \text{vec}(\tilde{S}_i) \rangle) - r(\mathcal{C}) \right\}$$

$$\mathcal{G}^{(j+1)} = \mathcal{C}^{(j+1)} / \rho^{(j+1)}$$

 Let $j := j + 1$

end while

Output: $\alpha = \alpha_0 / \rho, \sigma = 1 / \rho, \mathcal{G}, \{\tilde{\mathbf{B}}_d\}_{d=1}^3$

The pseudocode for the block relaxation algorithm is summarized in Algorithm 2. The convergence criterion is defined by $\ell(\tilde{\boldsymbol{\theta}}^{(j+1)}) - \ell(\tilde{\boldsymbol{\theta}}^{(j)}) < \epsilon$, where $\ell(\tilde{\boldsymbol{\theta}}^{(j)})$, is given by

$$\begin{aligned} \ell(\tilde{\boldsymbol{\theta}}^{(j)}) = & -n \ln \sigma^{(j)} + \sum_{i=1}^n \ln f \\ & \left(\frac{y_i - \alpha^{(j)} - \langle \tilde{\mathcal{G}}^{(j)} \times_1 \tilde{\mathbf{B}}_1^{(j)} \times_2 \tilde{\mathbf{B}}_2^{(j)} \times_3 \tilde{\mathbf{B}}_3^{(j)}, \tilde{S}_i \rangle}{\sigma} \right) \\ & - r(\tilde{\mathcal{G}}^{(j)}) - \sum_{d=1}^3 r(\tilde{\mathbf{B}}_d^{(j)}), \end{aligned} \quad (20)$$

where $\tilde{\boldsymbol{\theta}}^{(j)} = (\alpha^{(j)}, \rho^{(j)}, \mathcal{G}^{(j)}, \tilde{\mathbf{B}}_1^{(j)}, \tilde{\mathbf{B}}_2^{(j)}, \tilde{\mathbf{B}}_3^{(j)})$.

The set of ranks (i.e., k_1, k_2, k_3) used in the Tucker decomposition is an input to Algorithm 2. BIC is also used here to determine the appropriate rank, where ℓ is the log-likelihood value defined in Equation (20), n is the sample size (number of systems) and $m = \sum_{d=1}^3 p_d k_d + \prod_{d=1}^3 k_d - \sum_{d=1}^3 k_d^2$ is the number of effective parameters. Here the term $-\sum_{d=1}^3 k_d^2$ is used to adjust for the non-singular transformation indeterminacy in the Tucker decomposition (Li, Zhou, and Li 2013).

Using BIC for rank selection in the Tucker-based tensor regression model can be computationally prohibitive. For example, for a three-order tensor, there are totally $27 = 3^3$ rank

candidates when the maximum rank in each dimensionality is 3. Increasing the maximum rank to 4 and 5, the number of rank candidates is increased to $64 = 4^3$ and $125 = 5^3$, respectively. To address this challenge, we propose a computationally efficient heuristic method that automatically selects an appropriate rank. First, an initial coefficient tensor is estimated by regressing each pixel against the TTF. Next, high-order singular value decomposition (HOSVD) (De Lathauwer, De Moor, and Vandewalle 2000) is applied to the estimated tensor. HOSVD works by applying regular SVD to matricizations of the initial tensor on each mode. The rank of each mode can be selected by using fraction-of-variance explained (FVE) (Fang, Zhou, and Gebrael 2015) and the resulting eigenvector matrix is the factor matrix for that mode. Given the initial tensor and its estimated factor matrices, we can estimate the core tensor. The core tensor and factor matrices estimated by HOSVD are used for initialization in Algorithm 2. As pointed out by various studies in the literature, HOSVD often performs reasonably well as an initialization method for iterative tensor estimation algorithms (Kolda and Bader 2009; Lu, Plataniotis, and Venetsanopoulos 2008).

4. RUL Prediction and Realtime Updating

The goal of this article is to predict and update the RUL of partially degraded systems using in situ degradation image streams. To achieve this, we use the LLS regression modeling framework discussed in Section 3, and update the trained model based on data streams observed from fielded systems. The LLS regression model requires that the degradation image streams of the training systems and the fielded system being monitored to have the same dimensionality. In other words, both should have the same number of degradation images or profile length. In reality, this attribute is difficult to maintain for two reasons; (1) different systems have different failure times, and (2) an equipment is typically shutdown after failure and no further observations can be made beyond the failure time. Assuming the sampling (observation) time intervals are the same for all systems, a system with a longer failure time has more degradation data than a system with a short failure time.

To address this challenge, we adopt the time-varying regression framework used in Fang, Zhou, and Gebrael (2015). The idea of the time-varying regression is that systems whose TTF are shorter than the current observation time (of the fielded system) are excluded from the training dataset. Next, the degradation data of the chosen systems are truncated at the current observation time. By doing this, we ensure that the truncated degradation tensors of the chosen systems and the real-time observed degradation tensors of the fielded system possess the same dimensionality.

We summarize the process of predicting and updating the RUL of a fielded system as follows:

- (i) At each sampling time t_n , a new degradation image is observed from a fielded system. Systems whose TTF are longer than t_n are chosen from the training dataset.
- (ii) The image streams of the chosen systems are then truncated at time t_n by keeping only the images observed at times $\{t_1, t_2, \dots, t_n\}$. The truncated image streams con-

stitutes a new “training dataset,” hereafter referred to as *truncated training dataset*.

- (iii) A dimensionality reduction technique, such as MPCA, is applied to the *truncated training dataset* to obtain a low-dimensional tensor subspace of the *truncated training dataset*. Tensors in the *truncated training dataset* are then projected to the tensor subspace and their low-dimensional approximations are estimated.
- (iv) The low-dimensional approximation tensors are used to fit the regression model in Equation (2), and the parameters (i.e., $\hat{\alpha}_{t_n}$, $\hat{\mathcal{B}}_{t_n}$, and $\hat{\sigma}_{t_n}$) are estimated via one of the methods described in Sections 3.1 and 3.2.
- (v) The image stream from the fielded system is projected onto the tensor subspace estimated in step (3), and its low-dimensional approximation (denoted as $\tilde{\mathcal{S}}_{t_n}$) is also estimated. Next, the approximated tensor is input into the regression model estimated in step (4), and the TTF is predicted as follows:

$$\hat{y}_{t_n} = \hat{\alpha}_{t_n} + \langle \hat{\mathcal{B}}_{t_n}, \tilde{\mathcal{S}}_{t_n} \rangle + \hat{\sigma}_{t_n} \epsilon_i, \quad (21)$$

which implies the predicted TTF follows an LLS distribution with location parameter $\hat{\alpha}_{t_n} + \langle \hat{\mathcal{B}}_{t_n}, \tilde{\mathcal{S}}_{t_n} \rangle$ and scale parameter $\hat{\sigma}_{t_n}$, that is, $\hat{y}_{t_n} \sim \text{LLS}(\hat{\alpha}_{t_n} + \langle \hat{\mathcal{B}}_{t_n}, \tilde{\mathcal{S}}_{t_n} \rangle, \hat{\sigma}_{t_n})$. From this distribution, we can calculate both the point and interval estimate of the predicted TTF. The RUL is then obtained by subtracting the current observation time from the predicted TTF.

Note that Steps (i)–(iv) can be done offline. That is, given a training dataset, we can construct *truncated training datasets* with images observed at time $\{t_1, t_2\}$, $\{t_1, t_2, t_3\}$, $\{t_1, t_2, t_3, t_4\}$, \dots , respectively. Regression models are then estimated based on all the possible *truncated training datasets*. Once a new image is observed at say time t_n , the appropriate regression model with images $\{t_1, \dots, t_n\}$ is chosen, and the RUL of the fielded system is estimated in Step (v). This approach enables real-time RUL predictions.

5. Numerical Study I

In this section, we validate the effectiveness of model and rank selection criteria (BIC and the heuristic method) using

simulated degradation image streams. We assume the underlying physical degradation follows a heat transfer process based on which simulated degradation image streams are generated.

5.1. Data Generation

Suppose for system i , the degradation image stream, denoted by $\mathcal{S}_i(x, y, t)$, $i = 1, \dots, 1000$, is generated from the following heat transfer process:

$$\frac{\partial \mathcal{S}_i(x, y, t)}{\partial t} = \alpha_i \left(\frac{\partial^2 \mathcal{S}_i}{\partial x^2} + \frac{\partial^2 \mathcal{S}_i}{\partial y^2} \right), \quad (22)$$

where $(x, y); 0 \leq x, y \leq 0.05$ represents the location of each image pixel, α_i is the thermal diffusivity coefficient for system i and is randomly generated from a uniform distribution $\mathcal{U}(0.5 \times 10^{-5}, 1.5 \times 10^{-5})$ and t is the time frame. The initial and boundary conditions are set such that $\mathcal{S}|_{t=1} = 0$ and $\mathcal{S}|_{x=0} = \mathcal{S}|_{x=0.05} = \mathcal{S}|_{y=0} = \mathcal{S}|_{y=0.05} = 1$. At each time t , the image is recorded at locations $x = \frac{j}{n+1}, y = \frac{k}{n+1}, j, k = 1, \dots, n$, resulting in an $n \times n$ matrix. Here we set $n = 21$ and $t = 1, \dots, 10$, which leads to 10 images of size 21×21 for each system represented by a $21 \times 21 \times 10$ tensor. Finally, iid noises $\epsilon \sim N(0, 0.01)$ are added to each pixel. Example degradation images observed at time $t = 2, 4, 6, 8, 10$ from a simulated system are shown in Figure 3, in which (a) and (b) show images without and with noise, respectively.

To simulate the TTF of each system two sets of coefficient tensors are used. The first set, denoted by \mathcal{B}_C , is simulated in the form of basis matrices with rank 2 used in CP decomposition. Specifically, three matrices, that is, $\mathcal{B}_{C,1} \in \mathbb{R}^{21 \times 2}$, $\mathcal{B}_{C,2} \in \mathbb{R}^{21 \times 2}$, $\mathcal{B}_{C,3} \in \mathbb{R}^{10 \times 2}$ are generated. To induce sparsity, we randomly set half of elements of each matrix to be 0. The values of the remaining 50% elements are randomly generated from a uniform distribution $\text{unif}(-1, 1)$. The TTF, denoted by $y_{C,i}$, is generated by using $y_{C,i} = \langle \text{vec}(\mathcal{B}_C), \text{vec}(\mathcal{S}_i) \rangle + \sigma \epsilon_i$, where $\text{vec}(\mathcal{B}_C) = (\mathcal{B}_{C,3} \odot \mathcal{B}_{C,2} \odot \mathcal{B}_{C,1}) \mathbf{1}_2$, ϵ_i follows a standard smallest extreme value distribution $\text{SEV}(0, 1)$ and σ is 5% times the standard deviation of the location parameter, that is, $\langle \text{vec}(\mathcal{B}_C), \text{vec}(\mathcal{S}_i) \rangle$.

The second set, denoted by \mathcal{B}_T , is simulated in the form of core and factor matrices with rank $(2, 1, 2)$ used in Tucker decomposition. Specifically, a core tensor $\mathcal{G}_T \in \mathbb{R}^{2 \times 1 \times 2}$ and

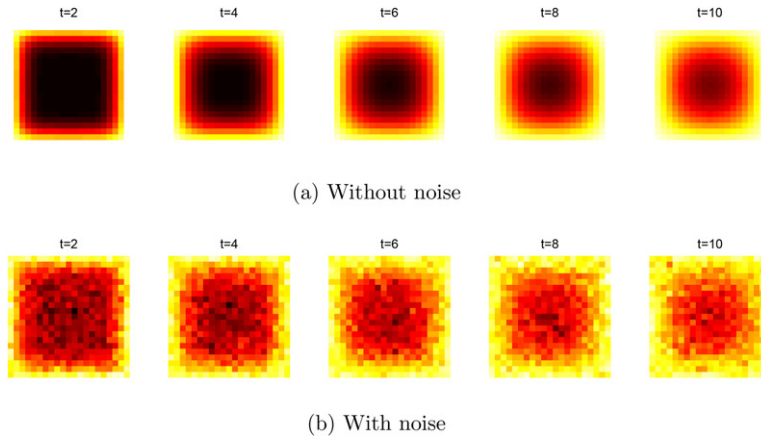


Figure 3. Simulated degradation images based on heat transfer process.

Table 1. BIC values for CP-based tensor regression.

Rank	1	2	3	4	5	6	7
SEV	620.5	−1535.3	−1383.4	−1232.7	−1122.9	−1014.4	−805.9
Normal	550.0	−1422.6	−1273.6	−1153.2	−1064.7	−1013.2	−1114.0
Weibull	618.1	−643.6	−472.6	−301.9	−180.5	−103.5	−54.0
Lognormal	610.5	−336.3	−187.7	−75.5	9.6	67.4	74.3

NOTE. Bold indicates the smallest BIC values achieved by the candidate models.

three factor matrices $\mathbf{B}_{T,1} \in \mathbb{R}^{21 \times 2}$, $\mathbf{B}_{T,2} \in \mathbb{R}^{21 \times 1}$, $\mathbf{B}_{T,3} \in \mathbb{R}^{10 \times 2}$ are generated. All the elements of the core tensor \mathcal{G}_T are set to 1. Furthermore, half of elements of matrices $\mathbf{B}_{T,1}$, $\mathbf{B}_{T,2}$, $\mathbf{B}_{T,3}$ are randomly set to 0 and the remaining elements are randomly generated from $\text{unif}(-1, 1)$. The TTF, $y_{T,i}$, is generated via $y_{T,i} = \langle \text{vec}(\mathbf{B}_T), \text{vec}(\mathcal{S}_i) \rangle + \sigma \epsilon_i$, where $\text{vec}(\mathbf{B}_T) = \mathcal{G}_T \times_1 \mathbf{B}_{T,1} \times_2 \mathbf{B}_{T,2} \times_3 \mathbf{B}_{T,3}$, ϵ_i follows a standard smallest extreme value distribution $\text{SEV}(0, 1)$ and σ is 5% times the SD of the location parameter, that is, $\langle \text{vec}(\mathbf{B}_T), \text{vec}(\mathcal{S}_i) \rangle$.

In the following section, we introduce how to use the BIC criterion and our heuristic rank selection method to identify the LLS distribution and rank. We randomly select 500 of the simulated systems for training and the remaining 500 systems for test.

5.2. Model and Rank Selection

We first apply CP-based tensor regression in Equation (5) to the training dataset, $\{y_{C,i}, \mathcal{S}_i\}_{i=1}^{500}$, and use Algorithm 1 to estimate the model parameters for different ranks and for four LLS distributions, namely, *normal*, *SEV*, *lognormal*, and *Weibull*. The BIC value is then computed for each distribution and rank combination as discussed in Section 3.1. As pointed out earlier, the block relaxation method in Algorithm 1 only guarantees a local optimum and hence, we shall run the algorithm 10 times using randomized initializations and record the smallest BIC. Here, the randomized initializations are achieved by setting each entry of matrices $\tilde{\mathbf{B}}_2^{(0)}$ and $\tilde{\mathbf{B}}_3^{(0)}$ with a random number generated from a uniform distribution $\mathcal{U}(-1, 1)$. The BIC values for all combinations are reported in Table 1. From Table 1, it can be seen that the smallest BIC value is **−1535.3**, which belongs to

Table 3. Distribution and rank selection results by using heuristic rank selection method.

LLS Distribution	Rank	BIC
SEV	(2,2,2)	−1212.3
Normal	(1,1,1)	−199.0
Weibull	(1,2,2)	36.1
Lognormal	(1,1,1)	−83.7

NOTE. Bold indicates the smallest BIC values achieved by the candidate models.

SEV distribution with rank $r = 2$. This coincides with the rank and the distribution we used to generate the data.

Similarly, the Tucker-based tensor regression model in Equation (13) is applied to the training dataset, $\{y_{T,i}, \mathcal{S}_i\}_{i=1}^{500}$ and Algorithm 2 (see Section 3.2) is used to estimate the parameters. Again, the randomized initializations in Algorithm 2 are achieved by setting each entry of the core tensor \mathcal{G} and matrices $\tilde{\mathbf{B}}_2^{(0)}$ and $\tilde{\mathbf{B}}_3^{(0)}$ with a random number generated from a uniform distribution $\mathcal{U}(-1, 1)$. A total of 27 different rank combinations are tested under four distributions, *normal*, *SEV*, *lognormal*, and *Weibull*. Again, for each distribution-rank combination, Algorithm 2 is run with 10 randomized initializations, and the smallest BIC value is reported in Table 2.

Table 2 indicates that the smallest BIC value (−1313.5) is associated with the *SEV* distribution with rank (2, 1, 2), which again matches the rank and the distribution that was used to generate the data. In Table 3, we also report the results of the heuristic rank selection method for Tucker. It can be seen from Table 3 that the heuristic rank selection method selects rank (1, 1, 1) under *normal* and *lognormal* distributions, while selects rank (2, 2, 2) under *SEV* distribution and (1, 2, 2) under *Weibull* distribution. The smallest BIC values (−1212.3) is achieved under *SEV* distribution with rank (2, 2, 2), which is close to the actual rank.

5.3. Selection Results

In this section, we study the performance of the BIC criterion and our heuristic rank selection method in identifying the correct LLS distribution and the right rank. Specifically, we run the process in Sections 5.1 and 5.2 for 100 times and summarize the probability that both the distribution and rank are correctly

Table 2. BIC values for Tucker-based tensor regression.

Rank	(1,1,1)	(1,1,2)	(1,1,3)	(1,2,1)	(1,2,2)	(1,2,3)	(1,3,1)	(1,3,2)	(1,3,3)
SEV	−163.3	−113.2	−75.8	−44.8	−59.0	−15.7	61.0	52.6	29.6
Normal	−199.0	−149.3	−112.0	−81.0	−82.8	−39.3	24.8	28.9	15.5
Weibull	−73.9	−24.4	13.0	44.1	35.9	79.2	149.6	147.4	133.5
Lognormal	−83.7	−33.9	3.4	34.4	28.6	71.6	140.0	140.2	141.9
Rank	(2,1,1)	(2,1,2)	(2,1,3)	(2,2,1)	(2,2,2)	(2,2,3)	(2,3,1)	(2,3,2)	(2,3,3)
SEV	−44.8	−1313.5	−1269.8	−16.1	−1212.7	−1202.9	95.4	−1115.0	−1106.5
Normal	−80.9	−1259.1	−1215.6	−22.9	−1149.7	−1130.8	89.3	−1048.6	−1028.2
Weibull	44.1	−733.8	−690.2	66.1	−633.2	−607.1	178.1	−543.5	−508.1
Lognormal	34.4	−497.8	−454.3	−85.2	−402.7	−394.4	197.2	−306.2	−292.6
Rank	(3,1,1)	(3,1,2)	(3,1,3)	(3,2,1)	(3,2,2)	(3,2,3)	(3,3,1)	(3,3,2)	(3,3,3)
SEV	60.7	−1201.8	−1224.9	95.5	−1156.4	−1164.0	113.0	−1071.4	−1074.4
Normal	24.9	−1147.2	−1153.2	88.8	−1093.2	−1082.6	129.4	−1009.0	−999.2
Weibull	149.7	−621.9	−613.1	177.8	−572.3	−539.2	205.6	−488.5	−468.2
Lognormal	139.9	−385.9	−391.0	197.5	−337.9	−331.4	238.5	−252.0	−262.3

NOTE. Bold indicates the smallest BIC values achieved by the candidate models.

Table 4. The percentage that both distribution and rank are correctly selected.

	SEV	Lognormal
BIC (CP-based model)	100%	100%
BIC (Tucker-based model)	98%	96%
Heuristic method (Tucker-based model)	86%	85%

selected. Moreover, we also test the selection correctness probability when the underlying distribution for the tensor-based regression model is *lognormal*. This is achieved by following the same process in Sections 5.1 and 5.2 except that the noise term ϵ_i follows a standard lognormal distribution when generating TTFs. We report the percentage that both distribution and rank are correctly selected in Table 4. Table 4 illustrates that the correct selection rate of the BIC criterion is at least 96% for both the CP- and Tucker-based regression models. Therefore, we can conclude that the BIC criterion is effective in selecting an appropriate distribution and the correct rank of the tensors in the LLS regression. Table 4 also indicates that the correct selection probability of the heuristic method is around 85%, which is a relatively high rate but lower than the rate of the BIC criterion. Although the probability that the heuristic method correctly identifying the distribution and rank is smaller than that of the BIC criterion, the Tucker-based regression model integrating the heuristic method achieves the smallest prediction errors (see Sections 6 and 7 for more details). This implies that the heuristic method is still useful in practice.

6. Numerical Study II

In this section, we validate the prediction capability of our methodology with the two types of decomposition approaches using simulated degradation image streams.

6.1. Data Generation

Similar to Section 5, we assume the underlying physical degradation follows a heat transfer process based on which simulated degradation image streams are generated. Specifically, the degradation image streams are generated using Equation (22). Here, the thermal diffusivity coefficient for system i , $i = 1, \dots, 500$, is randomly generated from a uniform distribution $\mathcal{U}(5 \times 10^{-5}, 1 \times 10^{-4})$. Each system has 100 images with size 51×51 . As a result, each system is represented by a $51 \times 51 \times 100$ tensor. Two types of noise are added to each image. The first type of noise is generated from a spatial Gaussian process, $GP(\mathbf{0}, K(\cdot, \cdot))$. Here, the covariance function $K(\mathbf{s}_1, \mathbf{s}_2) = \sigma^2 \exp(-\phi \|\mathbf{s}_1 - \mathbf{s}_2\|)$, where $\sigma^2 = 0.01$, $\phi = 0.25$, and $\|\mathbf{s}_1 - \mathbf{s}_2\| = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$ is the Euclidean distance between pixels $\mathbf{s}_1 = (x_1, y_1)$ and $\mathbf{s}_2 = (x_2, y_2)$. The second type of noise, independent and identically distributed from a normal distribution $\mathcal{N}(0, 0.02^2)$, is added to each pixel.

In prognostic analysis, a TTF is usually defined as the time that the degradation signal crosses a predefined failure threshold. However, for applications with degradation image streams, it is difficult to define such a failure threshold. To address this challenge, in this simulation, the TTF of system i is generated from $\tilde{y}_i = \alpha_i$, where α_i is the thermal diffusivity coefficient. The motivation behind this is that the thermal diffusivity coefficient

α_i controls system i 's degradation rate, which determines the TTF of the system in reality. Therefore, it is reasonable to use the diffusivity coefficient as a proxy of TTF.

In the following two sections, we first introduce the models used to benchmark our proposed methodology. Then, we compare the performance of our method with the benchmarking models in terms of the prediction capability.

6.2. Benchmarks and Validation Settings

Our proposed methods, designated as “CP” and “Tucker,” work by first applying MPCA to the degradation image streams to extract their low-dimensional projected tensors. Next, the projected tensors are regressed against TTFs using the CP- and Tucker-based LLS regression model, respectively. We compare the performance of our methodologies with five baseline models. The first two benchmarking models are similar to our proposed methodologies except that they do not apply MPCA on the degradation image streams. We refer to them as “CP (No MPCA)” and “Tucker (No MPCA),” respectively.

The third baseline model, which we designated as “FPCA,” uses functional principal components analysis (FPCA) to model the overall image intensity. To be specific, we first transform the degradation image stream of each system into a time-series signal by taking the average intensity of each observed image. Next, FPCA is applied to the time-series signals to extract features. FPCA is a popular functional data analysis technique that identifies the important sources of patterns and variations among functional data (time-series signals in our case) (Ramsay and Silverman 2005). The time-series signals are projected to a low-dimensional feature space spanned by the eigen-functions of the signals' covariance function and provides fused features called FPC-scores. Finally, FPC-scores are regressed against the TTFs by using LLS regression. More details about this FPCA prognostic model can be found in (Fang, Zhou, and Gebraeel 2015).

The fourth benchmark is referred to as “PCA,” which uses principal components analysis (PCA) to model the vectorized image intensity. Specifically, we first transform each image (in $\mathbb{R}^{51 \times 51}$) to a vector (in $\mathbb{R}^{1 \times 2601}$). Next, we construct a signal matrix (in $\mathbb{R}^{100n_0 \times 2601}$, n_0 is the number of training systems), each row of which contains a vectorized image. Third, PCA is applied to the signal matrix to reduce dimensionality and extract features. The extracted features from all the images of a system are concatenated to be that system's covariates, which are then regressed against the TTFs by using LLS regression.

We refer to the last baseline method as “B-spline.” This approach is inspired by Yan, Paynabar, and Shi (2017), in which the authors use penalized B-spline to fit the smooth mean function of an image stream. In this simulation, we also use penalized B-spline to fit each degradation image stream, and use the resulting coefficients as that image stream's low-dimensional features. We then regress the features against the TTFs using LLS regression.

We evaluate the performance of our methods and the benchmarks under different training sample sizes: (i) large and (ii) small, where the image streams of the first 400 and 100 systems in the simulated dataset are used for model training, respectively. The image streams of the last 100 systems in the simulated

Table 5. Computational time (unit: second).

Method	CP	Tucker	CP (No MPCA)	Tucker (No MPCA)	FPCA	PCA	B-spline
Time	118	126	113	154	23	656	232

dataset are used for validation. Prediction errors are calculated using the following expression:

$$\text{Prediction Error} = \frac{|\text{Estimated Lifetime}-\text{Real Lifetime}|}{\text{Real Lifetime}}. \quad (23)$$

In this simulation study, the LLS distributions for the regression models are selected by using BIC. The rank for CP-based models is selected using BIC, and the rank for Tucker-based models is chosen by the heuristic method discussed in Section 3.2.1. The number of principal components for “FPCA” and “PCA” is selected using cross-validation (CV). The order of spline basis, knots number, and regularization tuning parameters are chosen using generalized cross-validation (GCV) following the suggestion of Yan, Paynabar, and Shi (2017).

The simulation scenarios were performed using MATLAB 2012b in a 64-bit Unix system with the Xeon X5560 CPU @2.80 GHz processor and 150.0 GB RAM.

6.3. Results and Analysis

We first evaluate the computational time of our proposed methodologies and the benchmarking models. To do this, we take the image streams from the first 400 systems in the simulated dataset for training and randomly select 1 system from the remaining 100 systems for test. The computational time is reported in Table 5, which illustrates that our proposed methodologies are computationally less efficient than “FPCA” but more efficient than “PCA” and “B-spline.” In addition, it also indicates that the computational time for “CP” and “Tucker” are similar to the computational time of “CP (No MPCA)” and “Tucker (No MPCA).”

Next, we report boxplots of the prediction errors of different approaches at different training sample sizes in Figures 4 and 5. In the following sections, we first evaluate the performance of our proposed tensor-based models by comparing them with

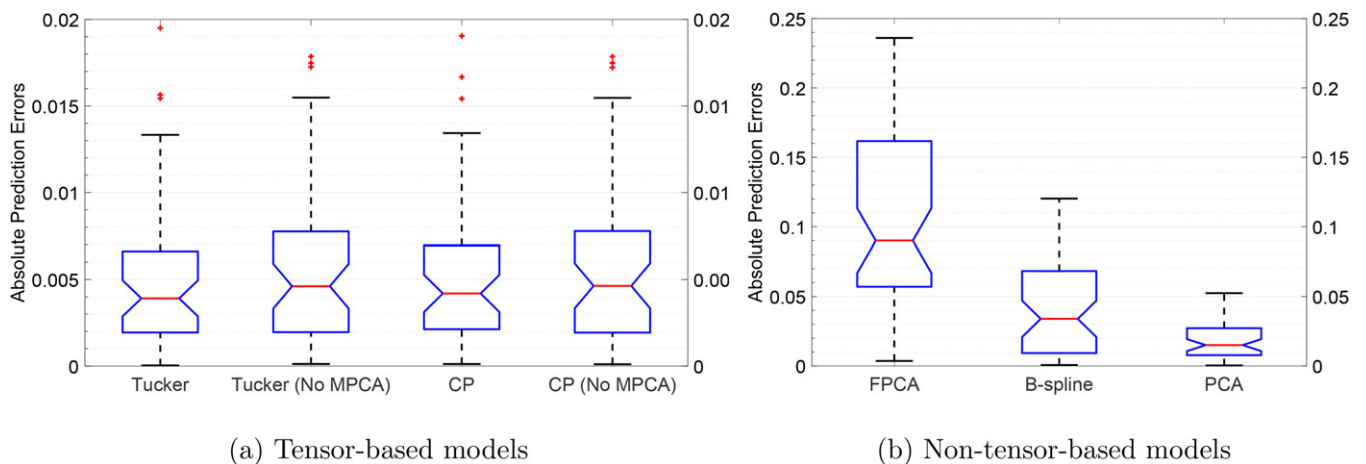
non-tensor-based benchmarks. Then, we evaluate the effectiveness of the multilinear dimension reduction technique incorporated in our tensor-based regression models.

6.3.1. Performance Comparison Between Tensor-Based and Non-Tensor-Based Models

Figures 4 and 5 indicate that, at both large and small training sample size scenarios, our proposed tensor-based regression models (i.e., “CP,” “Tucker,” “CP, No MPCA,” and “Tucker, No MPCA”) achieve smaller absolute prediction errors (in terms of both mean and variance) than the non-tensor-based benchmarks (i.e., “FPCA,” “PCA,” and “B-spline”). For example, in Figure 4 (a), the median absolute prediction errors (and the interquartile range, i.e., IQR) of the tensor-based models are around 0.5%(1.5%), while they are around 9%(25%), 3.2%(12%), and 1.5%(5%) for “FPCA,” “PCA,” and “B-spline,” respectively. Similar phenomenon can also be observed in Figure 5. This implies that our tensor-based models outperform other benchmarks in terms of both prediction accuracy and precision. We believe this is because our tensor-based models have the following characteristics: (i) Our methodologies are capable of capturing the spatio-temporal correlation structure in each image stream; (ii) the multilinear dimensionality reduction technique used in our methods is able to maximize the stream-to-stream (system-to-system) variation captured in the low-dimensional projection space; (iii) our methods can be seen as supervised dimension reduction methodologies, which use TTF information to supervise the dimension reduction of the coefficient tensor.

Compared with our models, none of the benchmarking models, that is, “FPCA,” “PCA,” or “B-spline,” uses TTF information to supervise the dimension reduction process. In addition, the benchmarking models have some other limitations that compromise their prediction capabilities. For example, “FPCA” transforms the degradation image stream of each system into a time-series signal by taking the average intensity of each observed image. This results in a significant loss of spatial information in each image, and thus, compromising the prediction accuracy.

The “PCA” first extracts low-dimensional features (i.e., PC-scores) by applying regular linear PCA to the set of vector-

**Figure 4.** Prediction errors with large training sample size.

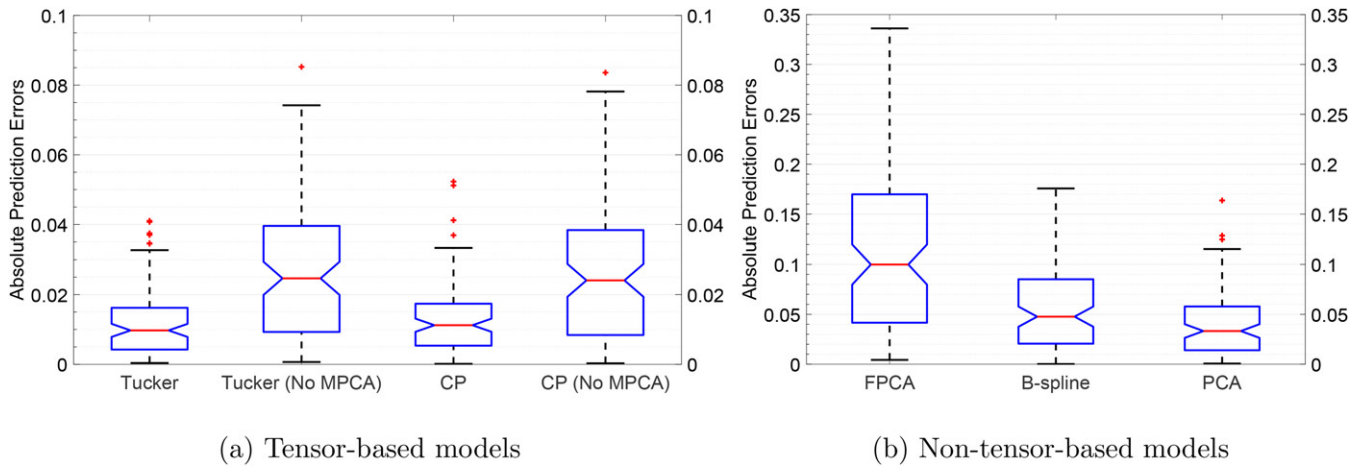


Figure 5. Prediction errors with small training sample size

ized images without considering their temporal dependency. Next, the PC-scores from all the images of each system are concatenated as the system's regression covariates. By doing so, "PCA" hierarchically captures the spatial and temporal correlation among image streams. However, the dimension reduction achieved by "PCA" is insufficient since "PCA" maximizes the image-to-image variation but not the system-to-system variation captured in the low-dimensional projection space. Furthermore, for image streams containing non-separable spatio-temporal correlation, "PCA" breaks the correlation structure by modeling spatial and temporal correlation separately. In addition, for applications where each system has numerous images, "PCA" may result in a large number of covariates for each system, and thus will pose a parameter estimation challenge (the number of parameters is equal to the number of PC-scores extracted from each image times the number of images in each system).

The "B-spline" benchmark individually fits each image stream using penalized B-spline, and the fitting coefficients are treated as low-dimensional features of that system. "B-spline" is capable of capturing the spatio-temporal correlation in each image stream. However, it fails to consider the stream-to-stream (system-to-system) variation, and thus it is an insufficient dimensionality reduction technique for our prognostic application.

All these aforementioned limitations of the benchmarking models compromise their prediction capabilities, so they did not perform well as our proposed tensor-based regression methodologies.

6.3.2. Performance Evaluation of Multilinear Dimension Reduction Methodology

In this subsection, we evaluate the performance of the multilinear dimension reduction technique incorporated in our methodologies.

Figure 4(a) illustrates that when the training sample size is large, the prediction errors of "CP" and "CP (No MPCA)" are close to each other. The median (IQR) of the prediction errors for "CP" and "CP (No MPCA)" are around 0.41%(1.4%) and 0.47%(1.5%), respectively. A similar phenomenon can also be observed between "Tucker" and "Tucker (No MPCA),"

where the median (IQR) of the prediction errors are around 0.4%(1.3%) and 0.45%(1.5%), respectively. This implies that, when the training sample size is large enough, the prediction capabilities of our proposed tensor-based regression models are not affected too much by incorporating the multilinear dimension reduction technique. Figure 5(a) indicates that when the training sample size is small, "CP" and "Tucker" achieve better prediction results than "CP (No MPCA)" and "Tucker (No MPCA)," respectively. For instance, the median (IQR) of the prediction errors for "CP" and "Tucker" are 1.2%(3.5%) and 1%(3.5%), respectively. However, they are 2.5%(8%) and 2.5%(7.5%) for "CP (No MPCA)" and "Tucker (No MPCA)" respectively. Therefore, we conclude that the multilinear dimension reduction technique can help improve the prediction capabilities of our tensor-based regression model when the training sample size is not large enough. This is reasonable since multilinear dimensional reduction techniques help reduce the number of parameters to be estimated in the tensor-based models.

7. Case Study: Degradation Image Streams From Rotating Machinery

In this section, we validate the effectiveness of our methodology using degradation image streams obtained from a rotating machinery. The experimental test bed, which was described in detail in Gebrael, Elwany, and Pan (2009), is designed to perform accelerated degradation tests on rolling element thrust bearings. The test bearings are run from a brand new state until failure. Vibration sensors are used to monitor the health of the rotating machinery. Failure is defined once the amplitude of defective vibration frequencies crosses a prespecified threshold based on ISO standards for machine vibration. Meanwhile, infrared images that capture temperature signatures of the degraded component throughout the degradation test are acquired using an FLIR T300 infrared camera. Infrared images with 40×20 pixels are stored every 10 seconds. Four different experiments were run to failure. The resulting degradation-based image streams contained 375, 611, 827, and 1478 images, respectively.

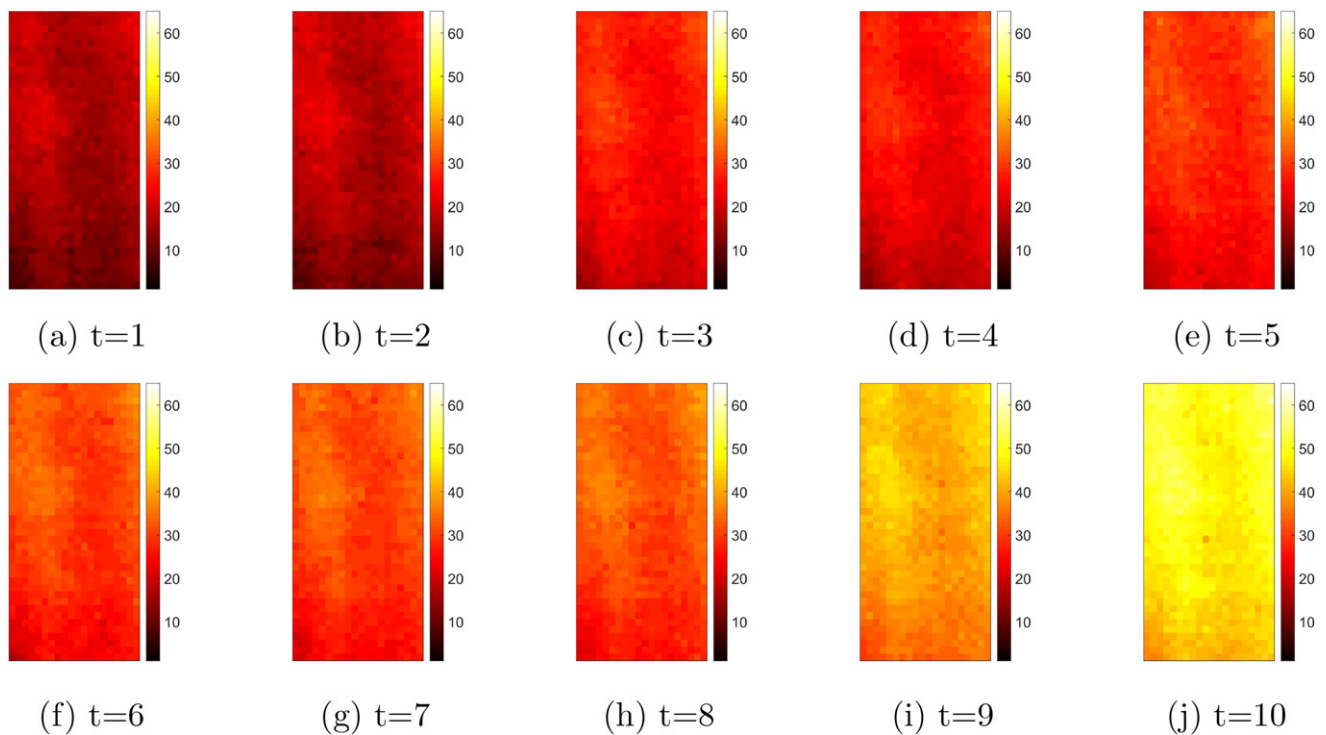


Figure 6. An illustration of one infrared degradation image stream.

Due to the high cost of running degradation experiments, additional degradation image streams were generated by resampling from the original image database obtained from the four experiments discussed earlier. In total 284 image data streams were generated. As an illustration, a sequence of images obtained at different (ordered) time periods are shown in Figure 6.

7.1. Model Selection

In this section, we discuss how to select an appropriate LLS tensor regression model for our dataset. This is achieved by applying different LLS tensor regression candidate models to a training set consisting of multiple image data streams. The model with the smallest BIC is selected as the best candidate model.

To account for the variability in the length of the image streams (as illustrated earlier in Section 4), we generate multiple subsamples based on different TTFs. Specifically, we sort the TTFs in ascending order such that $TTF_1 < TTF_2 < \dots < TTF_n$, where $n \leq 284$ is the number of unique TTFs (or equivalent the number of subsamples). Next, we define subsample i as the systems whose TTFs are greater than or equal to TTF_i , for $i = 1, \dots, n$. For example, subsample 1 includes all the 284 image streams, and subsample 2 includes all the image streams excluding the ones with the smallest TTF, and so forth. Third, each subsample is truncated by only keeping images observed on time domain $[0, TTF_i]$ epochs. By doing so, we ensure that all the image streams in a subsample have the same dimensionality. This is important when applying the LLS tensor regression model. After truncation, the following steps are applied to select the best candidate regression model:

- *Step 1: Dimension reduction.* MPCA is applied to each subsample i (truncated image stream). The fraction-of-variance-explained, which is used to select the number of multilinear principal components (see Lu, Platanotis, and Venetianopoulos (2008) for details), is set to be 0.95. Using this criterion, a low-dimensional tensor is extracted from each image stream (or each system).
- *Step 2: Fitting LLS model.* The low-dimensional tensors extracted from Step 1 are regressed against TTFs using an LLS regression model. Similar to the Simulation study, we evaluate four types of distributions: *normal*, *lognormal*, *SEV*, and *Weibull*. Tucker-based estimation method with heuristic rank selection is used for parameter estimation.
- *Step 3: Comparing BIC values.* BIC values are then computed for each of the four fitted models. The model with the smallest BIC is selected as the most appropriate one for the subsample.
- *Step 4: Distribution selection.* Steps 1, 2, and 3 are applied to all the subsamples. The distribution with the highest selected frequency is considered as the best candidate distribution.

After applying the aforementioned selection procedures to all the subsamples, we summarize the percentage of times each distribution was selected. Table 6 summarizes these results and shows that the *Weibull* distribution was selected on average 74.4% while the *lognormal* was selected 25.6% of the time. We expect to have some overlap in the models that have been selected because for specific parameter values, different distributions may exhibit reasonable fits for the same datasets. In our case, it is clear that the *Weibull* distribution dominates most of the selections and will, therefore, be considered as the suitable distribution for this dataset.

7.2. Performance Evaluation

The Weibull tensor regression model is chosen for evaluating the accuracy of predicting lifetime. Similar to the simulation study in Section 6, we compare the performance of our methods with the “FPCA,” “PCA,” and “B-spline.” Time-series degradation signals corresponding to the infrared images of the experimental test bed were obtained in a similar manner to what was discussed in Section 6. Figure 7 shows a sample of these transformed time-series signals.

The accuracy and precision of the predictions made by the proposed model as well as the benchmarking models are evaluated using a leave-one-out cross-validation study. For each validation, 283 systems are used for training and the remaining one system is used for testing. The RULs of the test system are predicted at each time epoch. The time-varying regression framework presented in Section 4 is used to enable the integration of newly observed image data (from the test data).

As pointed out in Section 4, the proposed LLS tensor regression models are capable of providing interval estimate for the predicted TTFs. To illustrate this, we take one of the systems as an example and report its prediction errors with a 95% confidence interval. To be specific, the predicted TTF (at each observation time epoch) and its upper and lower limits at a 95% confidence level are computed. The prediction errors of the predicted TTF and its upper and lower limits are, respectively, calculated using Equation (23), and the results are then reported in Figure 8. Figure 8 indicates that both the Tucker- and CP-based models achieve higher prediction accuracy when more observations are available, which is reasonable since more observation implies more information regarding the degradation condition of the system and thus, a more accurate prediction result. Another observation from Figure 8 is that both the models accomplish a smaller prediction error interval at a higher observation time epoch. This is also reasonable since more images (more information) are involved in the prediction

Table 6. Distribution selection results.

LLS Distribution	Normal	Lognormal	SEV	Weibull
Selection (%)	0%	25.6%	0%	74.4%

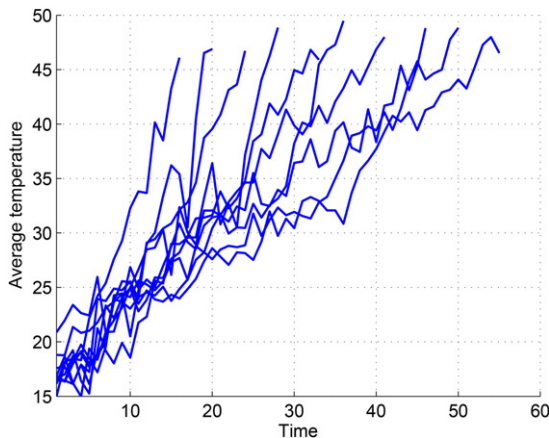


Figure 7. A sample of transformed time-series signals.

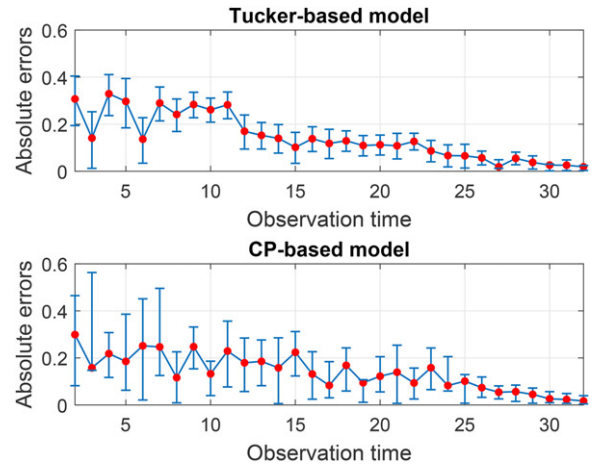


Figure 8. Prediction errors with a 95% CI.

at a higher observation time, and thus the uncertainty of the prediction result is reduced.

To evaluate the performance of our proposed models and the benchmarking methods over all the 284 systems, we summarize the mean and variance of the absolute prediction errors (of the 284 systems) in Figure 9 where 10% represents prediction errors evaluated at life percentiles in the interval of (5%, 15%), 20% for the interval of (15%, 25%), etc.

Figure 9 indicates that all the five methodologies have smaller prediction errors at higher observation percentiles. This is because at higher observation percentiles more degradation-based image data has been observed, which provide more information about the underlying physical degradation process. This results in better prediction accuracy. Figure 9 also reveals that the proposed CP-based and Tucker-based regression models outperform “FPCA,” “PCA,” and “B-spline” in terms of mean and variance of the absolute prediction errors. For example, at the 50th percentile, the mean (variance) of the absolute prediction errors for FPCA, PCA, B-spline, CP-based, and Tucker-based models are 0.12(0.02), 0.09(0.009), 0.08(0.009), 0.07(0.006), and 0.05(0.003), respectively. A similar pattern can also be seen at most of the remaining prediction percentiles. This is consistent with the results in Section 6. Similarly, we believe this is because our tensor-based regression models are capable of (i) capturing the spatio-temporal correlation structure in each image stream, (ii) maximizing the captured stream-to-stream (system-to-system) variation in the low-dimensional projection space, and (iii) using TTF information to supervise the dimension reduction of the coefficient tensor. Again, comparing with our methods, none of the benchmarks uses the TTF information for dimension reduction. In addition, “FPCA” results the loss of spatial information by averaging the pixel intensities. “PCA” is not able to maximize the captured system-to-system variation, breaks spatio-temporal correlation, and may lead to a large number of parameters. “B-spline” also fails to maximize the system-to-system variation. All these limitations compromise the prediction capabilities of the benchmarks.

Figure 9 also shows that Tucker-based regression performs better than CP-based. The mean and variance for the Tucker-based model are consistently lower than those of the CP-based

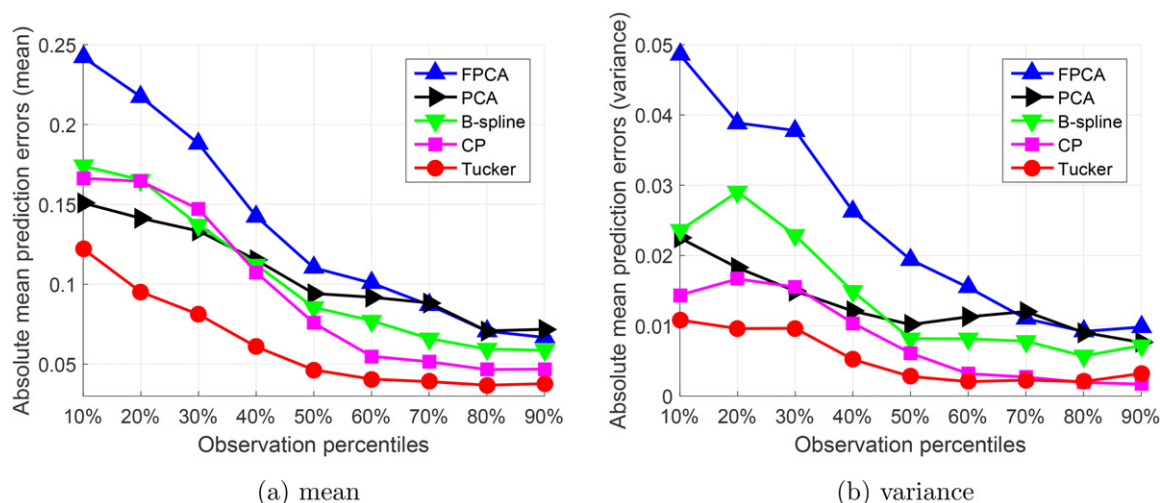


Figure 9. The mean and variance of the absolute prediction errors.

regression model. This difference may be attributed to the fact that the Tucker-based model allows the tensor to have a different rank for each of the three orders (directions). This enhances the flexibility of the regression model. In contrast, the CP-based model requires the rank on each direction to be equal, which may have an impact on the model's flexibility.

8. Conclusions

Degradation tensors such as image streams and profiles often contain rich information about the physical degradation process and can be utilized for prognostics and predicting the RUL of functioning systems. However, the analysis of degradation tensors is challenging due to their high-dimensionality and complex spatial-temporal structure. In this paper, we proposed a penalized (log)-location-scale regression model that can utilize high-dimensional tensors to predict the RUL of systems. Our method first reduces the dimensionality of tensor covariates by projecting them onto a low-dimensional tensor subspace that preserves the useful information of the covariates. Next, the projected low-dimensional covariate tensors are regressed against TTFs via an LLS regression model. To further reduce the number of parameters, the coefficient tensor is decomposed by utilizing two tensor decompositions, CP and Tucker. The CP decomposition decomposes the coefficient tensor as a product of low-dimensional basis matrices, and Tucker decomposition expresses it as a product of a low-dimensional core tensor and factor matrices. Instead of estimating the coefficient tensor, we only estimate its corresponding core tensors and factor/basis matrices. By doing so, the number of parameters to be estimated is dramatically reduced. Two numerical block relaxation algorithms with global convergence property were developed for the model estimation. The block relaxation algorithms iteratively estimate only one block of parameters (i.e., one factor/basis matrix or core tensor) at each time while keeping other blocks fixed until convergences.

We evaluated the performance of our proposed methodology through numerical studies. The results indicated that our methodology outperformed the benchmarks in terms of both prediction accuracy and precision. In addition, the results

showed that the multilinear dimension reduction technique could help improve the prediction accuracy and precision, especially when the training sample size is small. We also validated the effectiveness of our proposed tensor regression model using a case study on degradation modeling of bearings in a rotating machinery. The results indicated that both CP-based and Tucker-based models outperformed the benchmarks in terms of prediction accuracy as well as precision at almost all the life percentiles. The results also indicated that Tucker-based model achieved better prediction accuracy than the CP-based model. This is reasonable since Tucker-based model is more flexible as it allows different modes to have different ranks, while the CP-based model requires all the modes have the same rank. The model developed in this article only works on a single computer. Development of a tensor-based prognostics model that can run on a distributed computing system is an important topic for future research.

Acknowledgment

The authors thank the reviewers and editors for their constructive comments and suggestions, which have considerably improved the article.

Funding

This work was supported by National Science Foundation Grants CMMI-1536555.

Supplementary Material

The proofs of all propositions and code for simulation studies are given in the online supplementary materials.

References

- Bagavathiappan, S., Lahiri, B. B., Saravanan, T., Philip, J., and Jayakumar, T. (2013), "Infrared Thermography for Condition Monitoring—A Review," *Infrared Physics & Technology*, 60, 35–55. [369]
- Carroll, J. D., and Chang, J. J. (1970), "Analysis of Individual Differences in Multidimensional Scaling via an N-way Generalization of 'Eckart-Young' Decomposition," *Psychometrika*, 35, 283–319. [370,371]

- Chen, N., Ye, Z. S., Xiang, Y., and Zhang, L. (2015), "Condition-Based Maintenance Using the Inverse Gaussian Degradation Model," *European Journal of Operational Research*, 243, 190–199. [369]
- Cressie, N., and Wile, C. K. (2015), *Statistics for Spatio-Temporal Data*, Hoboken, NJ: Wiley. [369]
- De Lathauwer, L., De Moor, B., and Vandewalle, J. (2000), "A Multilinear Singular Value Decomposition," *SIAM Journal on Matrix Analysis and Applications*, 21, 1253–1278. [375]
- De Leeuw, J. (1994), "Block-Relaxation Algorithms in Statistics," in *Information Systems and Data Analysis*. Berlin: Springer, pp. 308–324. [371,373]
- Doray, L. (1994), "IBNR Reserve Under a Loglinear Location-Scale Regression Model," *Casualty Actuarial Society Forum*, 2, 607–652. [370,371,373]
- Fang, X., Zhou, R., and Gebraeel, N. (2015), "An Adaptive Functional Regression-Based Prognostic Model for Applications With Missing Data," *Reliability Engineering & System Safety*, 133, 266–274. [369,375,378]
- Fang, X., Paynabar, K., and Gebraeel, N. (2017), "Multistream Sensor Fusion-Based Prognostics Model for Systems With Single Failure Modes," *Reliability Engineering & System Safety*, 159, 322–331. [369]
- Fang, X., Gebraeel, N. Z., and Paynabar, K. (2017), "Scalable Prognostic Models for Large-Scale Condition Monitoring Applications," *IIE Transactions*, 49, 698–710. [372]
- Friedman, J., Hastie, T., Höfling, H., and Tibshirani, R. (2007), "Pathwise Coordinate Optimization," *The Annals of Applied Statistics*, 1, 302–332. [373]
- Gebraeel, N. Z., Lawley, M. A., Li, R., and Ryan, J. K. (2005), "Residual-Life Distributions From Component Degradation Signals: A Bayesian Approach," *IIE Transactions*, 37, 543–557. [369]
- Gebraeel, N., Elwany, A., and Pan, J. (2009), "Residual Life Predictions in the Absence of Prior Degradation Knowledge," *IEEE Transactions on Reliability*, 58, 106–117. [380]
- Kolda, T. G., and Bader, B. W. (2009), "Tensor Decompositions and Applications," *SIAM Review*, 51, 455–500. [370,371,375]
- Lange, K. (2010), *Numerical Analysis for Statisticians*. Berlin: Springer Science & Business Media. [371,373]
- Li, X., Zhou, H., and Li, L. (2013), "Tucker Tensor Regression and Neuroimaging Analysis," arXiv preprint arXiv:1304.5637. [374,375]
- Liu, X., Yeo, K., and Kalagnanam, J. (2016), "Statistical Modeling for Spatio-Temporal Degradation Data," arXiv preprint arXiv:1609.07217. [369]
- Lu, H., Plataniotis, K. N., and Venetsanopoulos, A. N. (2008), "MPCA: Multilinear Principal Component Analysis of Tensor Objects," *IEEE Transactions on Neural Networks*, 19, 18–39. [370,372,375,381]
- Meola, C. (2007), "Infrared Thermography of Masonry Structures," *Infrared Physics & Technology*, 49, 228–233. [369]
- Neogi, N., Mohanta, D. K., and Dutta, P. K. (2014), "Review of Vision-Based Steel Surface Inspection Systems," *EURASIP Journal on Image and Video Processing*, 2014, 50. [369]
- Park, T., and Casella, G. (2008), "The Bayesian Lasso," *Journal of the American Statistical Association*, 103, 681–686. [373]
- Pastor, M. L., Balandraud, X., Grédiac, M., and Robert, J. L. (2008), "Applying Infrared Thermography to Study the Heating of 2024-T3 Aluminium Specimens Under Fatigue Loading," *Infrared Physics & Technology*, 51, 505–515. [369]
- Ramsay J. O., and Silverman, B. W. (2005), *Functional Data Analysis*, New York: Springer. [378]
- Seo, J. J., Yoon, H., Ha, H., Hong, D. P., and Kim, W. (2011), "Infrared Thermographic Diagnosis Mechanism for Fault Detection of Ball Bearing Under Dynamic Loading Conditions," *Advanced Materials Research*, 295, 1544–1547. [369]
- Shu, Y., Feng, Q., and Coit, D. W. (2015), "Life Distribution Analysis Based on Lévy Subordinators for Degradation With Random Jumps," *Naval Research Logistics*, 62, 483–492. [369]
- Städler, N., Bühlmann, P., and Van De Geer, S. (2010), " ℓ_1 -Penalization for Mixture Regression Models," *Test*, 19, 209–256. [373]
- Tseng, P. (2001), "Convergence of a Block Coordinate Descent Method for Nondifferentiable Minimization," *Journal of Optimization Theory and Applications*, 109, 475–494. [373]
- Tucker, L. R. (1966), "Some Mathematical Notes on Three-Mode Factor Analysis," *Psychometrika*, 31, 279–311. [370,371]
- Vellvehi, M., Perpina, X., Lauro, G. L., Perillo, F., and Jorda, X. (2011), "Irradiance-Based Emissivity Correction in Infrared Thermography for Electronic Applications," *Review of Scientific Instruments*, 82, 114901. [369]
- Yan, H., Paynabar, K., and Shi, J. (2017), "Real-Time Monitoring of High-Dimensional Functional Data Streams via Spatio-Temporal Smooth Sparse Decomposition," *Technometrics*, 60, 181–197. [378,379]
- Yan, H., Paynabar, K., and Shi, J. (2015), "Image-Based Process Monitoring Using Low-Rank Tensor Decomposition," *IEEE Transactions on Automation Science and Engineering*, 12, 216–227. [369]
- Ye, Z. S., Chen, N., and Shen, Y. (2015), "A New Class of Wiener Process Models for Degradation Analysis," *Reliability Engineering & System Safety*, 139, 58–67. [369]
- Ye, Z. S., and Chen, N. (2014), "The Inverse Gaussian Process as a Degradation Model," *Technometrics*, 56, 302–311. [369]
- Zhang, Y., and Liao, H. (2015), "Analysis of Destructive Degradation Tests for a Product With Random degradation Initiation Time," *IEEE Transactions on Reliability*, 64, 516–527. [369]
- Zhou, H., Li, L., and Zhu, H. (2013), "Tensor Regression With Applications in Neuroimaging Data Analysis," *Journal of the American Statistical Association*, 108, 540–552. [373]
- Zhou, R. R., Serban, N., Gebraeel, N., and Müller, H. G. (2014), "A Functional Time Warping Approach to Modeling and Monitoring Truncated Degradation Signals," *Technometrics*, 56, 67–77. [369]