IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication 1/4/2024, date of current version 1/4/2024.

Digital Object Identifier 10.1109/ACCESS.2023.0322000

Ensemble Principal Component Analysis

OLGA DORABIALA[1], ALEKSANDR ARAVKIN[1,2], and J. NATHAN KUTZ[1,2]
[1]Department of Applied Mathematics, University of Washington, Seattle, WA 98105 USA
[2]Department of Electrical and Computer Engineering, University of Washington, Seattle, WA 98105 USA

Corresponding author: Olga Dorabiala (e-mail: olgad400@uw.edu).

This material is based upon work supported by the National AI Research Institutes program supported by NSF and DHS under Award No. 2112085. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding agencies.

ABSTRACT Efficient representations of data are essential for processing, exploration, and human understanding, and *Principal Component Analysis* (PCA) is one of the most common dimensionality reduction techniques used for the analysis of large, multivariate datasets today. Two well-known limitations of the method include sensitivity to outliers and noise and no clear methodology for the uncertainty quantification of the principal components or their associated explained variances. Whereas previous work has focused on each of these problems individually, we propose a scalable method called *Ensemble PCA* (EPCA) that addresses them simultaneously for data which has an inherently low-rank structure. EPCA combines boostrapped PCA with k-means cluster analysis to handle challenges associated with sign-ambiguity and the re-ordering of components in the PCA subsamples. EPCA provides a noise-resistant extension of PCA that lends itself naturally to uncertainty quantification. We test EPCA on data corrupted with white noise, sparse noise, and outliers against both classical PCA and Robust PCA (RPCA) and show that EPCA performs competitively across different noise scenarios, with a clear advantage on datasets containing outliers and orders of magnitude reduction in computational cost compared to RPCA.

INDEX TERMS dimensionality reduction, principal component analysis, k-means clustering

I. INTRODUCTION

Across the engineering, physical, biological and social sciences, data science methods are a dominant paradigm for the processing, exploration, and understanding of emerging big data. Indeed, modern sensor technologies are revolutionizing the automated collection of large data sets, which must be processed in order to aid in human understanding and decision making. Critical for such interpretability is the generation of low-dimensional feature spaces, or dominant patterns, which can be extracted from high-dimensional and multivariate data streams [21], [39]. One of the most successful methods for extracting dominant features in data is *principal component analysis* (PCA), which provides a characterization of dominant correlated activity using the underling mathematical algorithm of the *singular value decomposition* (SVD). The SVD has been so prolific, that it has been independently developed for dimensionality-reduction and feature extraction across a number of fields, each with their own name, including *proper orthogonal decomposition* (POD) [3], [4], [26], *empirical orthogonal functions* (EOFs) [19], [28], Karhunen-Loéve expansion [11], [25], and the Hoteling transform [20]. Computational algorithms of the early 1970s [16] allowed for the robust and mature extraction of PCA modes from data, which has made PCA a classical data analysis technique. However, PCA is sensitive to noise and outliers, and the extraction of PCA modes does not come with any uncertainty quantification (UQ) metrics. We address both of these issues in our innovation of *Ensemble PCA*, which builds upon classic PCA in order to stabilize PCA models and provide clear UQ metrics for a given data set.

The motivations for dimensionality reduction are extensive. Often, the features of high-dimensional data exhibit partial redundancy and dependency. Beyond reducing redundancy, extracting the ‘‘most important" features of a data matrix that best summarize the information contained in a signal and removing irrelevant features helps practitioners both to work with limited computational resources and to understand the underlying data structure. Further, projecting high-dimensional data into 2D or 3D space can be extremely beneficial for human visualization. Too many features can complicate data analysis and visualization, and dimensionality reduction helps avoid these pitfalls. There are two main classes of dimensionality reduction techniques: linear and nonlinear. At their core, linear techniques use linear transformations to shift and stretch data. Examples include SVD, PCA, and Fisher’s *linear discriminant analysis* (LDA). These

methods are a cornerstone of analyzing data due to their simple geometric interpretations and typically attractive computational properties [8]. They are particularly useful when the data lies in a linear subspace and where the original variables are replaced by a smaller set of underlying variables. Nonlinear techniques involve more complicated data transformations and include *t-distributed stochastic neighbor embedding* (t-SNE) [37], Isomap [2], and autoencoders [17], [38]. Nonlinear methods are generally more powerful than their linear counterparts, but can be slow to optimize and many are non-deterministic, meaning they get different, locally-optimal solutions on each run [33].

The focus of this work is linear dimensionality reduction, particularly PCA [22], as it is arguably the most common dimensionality reduction technique used for the analysis of large, multivariate datasets today. Innovations in randomized methods now allow PCA to work at massive scales [12], [13], [18]. Extensions of PCA include Kernel PCA, which runs the algorithm on a feature space determined by the kernel [29], probabilistic PCA [35], which is targeted for data with missing entries, and sparse PCA [42], which seeks to summarize data using combinations of only a few input variables.

Regardless of extension, PCA has two well-known shortcomings. The first is sensitivity to outliers, where entire rows of the data matrix may be contaminated, and sparse noise, where individual entries of the data matrix may be affected. The second drawback is that there is no clear methodology for UQ. Previous work has focused on each of these problems individually. Robust PCA extends PCA to the sparse noise domain [5], and bootstrapping has been explored as a method for estimating sampling variability [14]. To our knowledge, there has not been an extensive analysis of bootstrapping for the purpose of developing a noise-resistant PCA. Gabrys et. al [15] suggested that statistical resampling could be applied to PCA to recover the true components of datasets corrupted with large outliers, but failed to test this concept on a sufficient number of datasets and wholly ignored UQ. Drawing inspiration from [15] we propose Ensemble Principal Component Analysis (EPCA), which ensembles bootstrapping with k-means cluster analysis to create a noise-resistant approach. We test EPCA against RPCA and standard PCA on eight datasets corrupted with sparse noise, white noise, and outliers. We show that EPCA achieves maximum performance on datasets with outliers and performs competitively on datasets with other types of noise. EPCA also naturally supports uncertainty quantification and provides an orders of magnitude reduction in computational cost compared to RPCA.

II. RELATED WORK

PCA operates as follows. Let $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_m] \in \mathbb{R}^{N\times m}$ be a data matrix with N samples and m features. Define the mean-centered matrix as $\overline{\mathbf{X}} = \mathbf{X} - \mu_{\mathbf{X}}$, where each feature is centered by its mean. The sample covariance matrix $\mathbf{C} \in \mathbb{R}^{m\times m}$ of $\overline{\mathbf{X}}$ is then defined as

$$\mathbf{C} = \frac{\overline{\mathbf{X}}^T\overline{\mathbf{X}}}{n-1}. \tag{1}$$

Since $\mathbf{C}$ is symmetric, it can be diagonalized as

$$\mathbf{C} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^*, \tag{2}$$

where $\mathbf{V}$ contains the eigenvectors of $\mathbf{C}$ and $\mathbf{\Lambda}$ is a diagonal matrix containing the eigenvalues λ_i on the diagonal, ordered by decreasing magnitude. The principal components of $\mathbf{X}$ are defined as $\overline{\mathbf{X}}\mathbf{V}$. The principal component associated with the largest eigenvalue projects the data onto the direction of greatest variance, while the eigenvalue measures the amount of information that can be explained by that projection [36]. The d largest eigenvalues are referred to as explained variance.

The results of PCA can be unreliable under data corruption. We identify three main classes of noise: white noise, sparse noise, and outliers. Under white noise, all entries of a data matrix $\mathbf{X}$ are slightly perturbed. Under sparse noise, a small subset of the elements of $\mathbf{X}$ are corrupt with some probability p, and with outliers, multiplicative noise is applied to a given percentage of rows of $\mathbf{X}$. PCA will break down if even one entry of $\mathbf{X}$ is grossly corrupted [27].

The sensitivity of PCA has motivated the development of noise-resistant extensions. Directions of research include the use of robust estimators of scatter as opposed to sample covariance [36] and the development of Robust Principal Component Analysis (RPCA). RPCA sets the standard for dimensionality reduction under sparse noise, and is generally applied for foreground detection [5]. Mathematically, RPCA assumes we observe a data matrix $\mathbf{X} \in \mathbb{R}^{N\times m}$ of the following form $\mathbf{X} = \mathbf{L}_0 + \mathbf{S}_0 + \mathbf{Z}_0$, where $\mathbf{L}_0$ is low-rank, $\mathbf{S}_0$ is sparse, and $\mathbf{Z}_0$ is a noise term containing i.i.d. noise on each entry [7], [40]. Generally, RPCA recovers $\mathbf{L}_0$ and $\mathbf{S}_0$ by solving the optimization problem in (3)

$$\min_{\mathbf{L},\mathbf{S}} ||\mathbf{L}||_* + \alpha||\mathbf{S}||_1, \tag{3}$$

subject to some constraint on the $\mathbf{L}$ and $\mathbf{S}$ matrices, such as $\mathbf{X} = \mathbf{L} + \mathbf{S}$ or $||\mathbf{X} - \mathbf{L} - \mathbf{S}|| \leq \delta$, where δ is some tuning parameter and the norm can be customized [5]. In Expression (3), $||\cdot||_*$ denotes the nuclear norm, $||\cdot||_1$ denotes the sum of the absolute values of matrix entries, and $\alpha > 0$ is a tuning parameter controlling the regularization of $\mathbf{S}$.

Equation (3) is referred to as the Principal Component Pursuit (PCP) problem, and a handful of methods have been suggested for solving PCP [5], [27]. PCP can recover $\mathbf{L}$ and $\mathbf{S}$ exactly [7], but is limited to the case where the low-rank component is exactly low-rank and the sparse component is exactly sparse. In addition, existing algorithms for solving PCP are computationally expensive and require extensive parameter tuning [5]. Another set of extensions to PCA focus on missing data, where entries of $\mathbf{X}$ are deleted at random. Most approaches specialized for dealing with missingness focus on data imputation [23], [41]. However, missingness can be

considered a sub-category of sparse noise, and methods such as RPCA continue to work in this context.

Another drawback of PCA is that it offers no clear method for estimating the sampling variability of its descriptive outputs. Though some work has explored analytical, asymptotic confidence intervals (CIs) for principal components, the methods are often either computationally infeasible or require strong assumptions on the data [14]. An alternative approach is using bootstrap-based CIs. Bootstrapping is a way to assess the variability of a statistic of interest through random sampling with replacement. It does not assume any distribution for the estimates of the uncertainties, and can be applied to most statistics [1].

In the context of PCA, boostrapping estimates the variability of PCA across different samples of the population [14]. From a theoretical standpoint, boostrapping is generally not useful unless we are interested in inference concerning only a few large eigenvalues, which are well-separated from the bulk and of multiplicity one [24], i,e. data of inherently low-rank. Though bootstrapping has been applied for the uncertainty quantification of PCA, very little work has been done using bootstrapping as a method for robustification.

Gabrys et. al [15] suggested using statistical resampling as a way to recover the principal components of a data matrix corrupted with outliers, and taking inspiration from their work, we propose EPCA, which ensembles bootstrapping PCA with k-means clustering to create a method for the dimensionality reduction and analysis of low-rank, noisy data that also lends itself to uncertainty quantification. By using k-means to aggregate the output of boostrapped samples, we circumvent the challenges associated with component re-ordering and sign ambiguity. We test the performance of EPCA against classical PCA and Robust PCA, with respect to runtime and accuracy, on datasets corrupted with sparse noise, white noise, and outliers.

III. ENSEMBLE PRINCIPAL COMPONENT ANALYSIS (EPCA)

Given a data matrix $\mathbf{X} \in \mathbb{R}^{N\times m}$, EPCA samples B bags of size n at random with replacement. PCA is run on each of the B samples, the principal components are stored in a matrix $\mathbf{P}^{(j)}$, and the eigenvalues are stored in a matrix $\mathbf{\Lambda}^{(j)}$ for $j \in [1, B]$. The goal is to summarize the results of our B samples to output d dominant modes.

Two challenges are that there is rotational variability in the principal components found by PCA and that the identified components can be re-ordered in the subsamples [34]. Most boostrapping PCA approaches tackle the first issue by using a Procrustean rotation to match the boostrap PCs to the PCs obtained by running PCA on the entire dataset [30] or by rotating the PCs towards some pre-specified target matrix $\mathbf{T}$ [34]. Instead, we create the matrix

$$\widetilde{\mathbf{P}} = \begin{bmatrix} \mathbf{P}^{(1)} \\ \mathbf{P}^{(2)} \\ \vdots \\ \mathbf{P}^{(B)} \\ -\mathbf{P}^{(1)} \\ -\mathbf{P}^{(2)} \\ \vdots \\ -\mathbf{P}^{(B)} \end{bmatrix} \tag{4}$$

by stacking all of the principal components found in the bags and their reflections. In this way, every principal component is stored along with its reflection, regardless of initial orientation. We also stack the corresponding eigenvalues accordingly, creating

$$\widetilde{\mathbf{\Lambda}} = \begin{bmatrix} \mathbf{\Lambda}^{(1)} \\ \mathbf{\Lambda}^{(2)} \\ \vdots \\ \mathbf{\Lambda}^{(B)} \\ \mathbf{\Lambda}^{(1)} \\ \mathbf{\Lambda}^{(2)} \\ \vdots \\ \mathbf{\Lambda}^{(B)} \end{bmatrix}. \tag{5}$$

The next step is to run k-means clustering on $\widetilde{\mathbf{P}}$ to output $2d$ clusters. This approach automatically clusters components that are oriented in the same direction and avoids any challenges associated with re-ordering. We use the normalized cluster centers of our d rotationally unique clusters as our predicted principal components, and the averages of the eigenvalues associated with the members of each cluster as our predicted eigenvalues. We order our final predicted components according to the magnitude of the average predicted eigenvalues. EPCA is visualized in Figure 1 and explained in Algorithm 1.

Algorithm 1 Ensemble Principal Component Analysis

(a) Mean center the data. $\overline{\mathbf{X}} = \mathbf{X} - \mu_{\mathbf{X}}$
(b) Select B bags of size n with replacement to create $\overline{X}^{(j)}$ for $j \in [1, B]$.
(c) Run PCA on each of the B bags to output d eigenvalues $\mathbf{\Lambda}^{(j)}$ and d principal components $\mathbf{P}^{(j)}$
(d) Stack all $\mathbf{P}^{(j)}$ and their reflections to create $\widetilde{\mathbf{P}}$. Stack corresponding $\mathbf{\Lambda}^{(j)}$ to create $\widetilde{\mathbf{\Lambda}}$.
(e) Run k-means clustering on $\widetilde{\mathbf{P}}$ with $2d$ clusters to cluster principal components oriented in the same direction.
(f) Output the directionally unique d average principal components, their corresponding average eigenvalues, and the variances of both.

IV. UNCERTAINTY QUANTIFICATION

Like all other boostrapping PCA methods [1], [14], [30], [34], EPCA lends itself naturally to uncertainty quantifica-

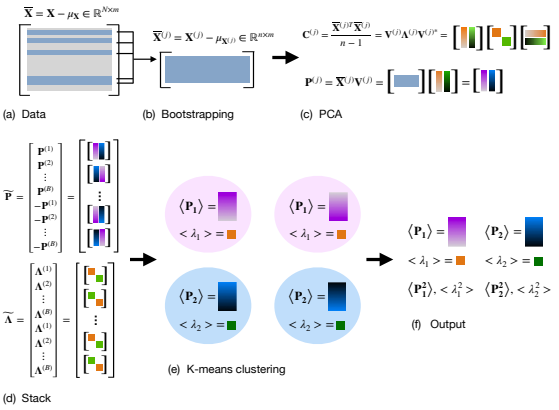


FIGURE 1. Ensemble Principal Component Analysis (EPCA). Given a data matrix with inherently low-rank structure, we sample *B* bags of size *n* at random with replacement. We run PCA and store *d* principal components and their corresponding eigenvalues for each bag. We stack all components, along with their reflections to account for rotational variability. We also stack all eigenvalues, in accordance with the order in the matrix storing components. Next, we run *k*-means clustering on the stacked component matrix with 2*d* clusters.

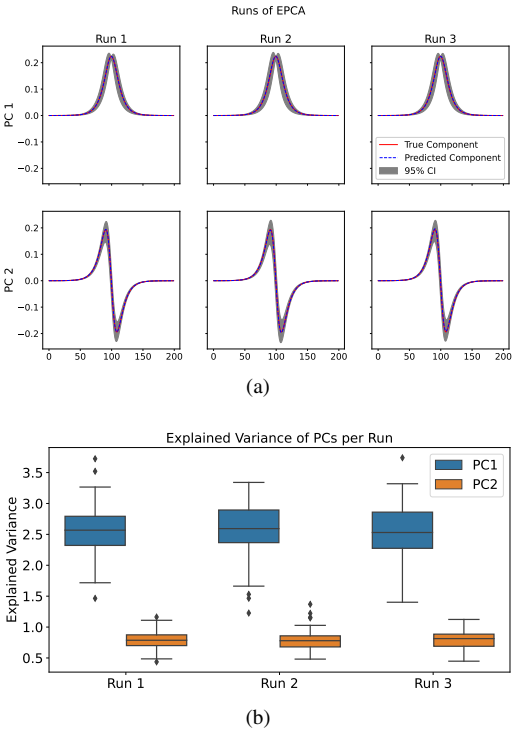


FIGURE 2. Uncertainty Quantification of three runs of EPCA on wave data. (a) The 95% Confidence Intervals (CIs) associated with the top two principal components. The CIs are tight, suggesting high confidence in the predicted components. (b) Boxplots of the distribution of the explained variance of each of the principal components. The true eigenvalues of the covariance matrix are contained within the interquartile ranges (IQRs) of all three trials.

tion. There are many approaches to estimate a confidence interval (CI) from the boostrap distribution, but we focus on the percentile method [34].

Figures 2 and 3 show UQ for three runs of EPCA on a clean dataset and the same dataset containing 5% outliers of scale 10, respectively. We show the results of three runs, since the results of EPCA vary based on the random bootstrap samples. Figures 2(a) and 3(a) show the 95% CIs for the principal components. The tightness of our CIs correlates with the level of confidence in our output. We note that the CIs are significantly tighter for the clean data, and even though there is greater uncertainty on the corrupted data, EPCA is still able to closely identify the true components in two of the three pictured runs. Figures 2(b) and 3(b) show the distributions of the respective eigenvalues, which tell us the amount of variance explained by each of the principal components. On both the original and corrupt data, the interquartile ranges (IQR) of the variances capture the true values, but on the corrupted data, the IQRs are skewed higher and slightly wider. Gross outliers have been removed from the boxplots in Figure 3(b). Even though the explained variances are more difficult to capture on the outlier data, they at least tell us about the order of the principal components and their separation. In practice, the eigenvalues of the covariance matrix are not used in PCA's projection of data into a lower-dimensional space, so their explicit values are not as important as the principal components themselves.

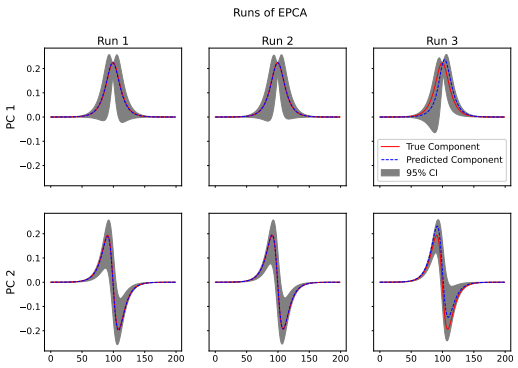
V. EXPERIMENTS

We test EPCA against RPCA (solved using Augmented Lagrange Multipliers [27]) and classical PCA on eight datasets with four types of added noise: sparse, Gaussian white, uniform white, and outliers. We include three small, three medium, and two large datasets in our analysis. Our base datasets, in order of increasing size, are iris [32], wine [32], breast cancer Wisconsin (WBC) [32], a synthetic wave, the digits 0 and 1 from MNIST [9], flow around a cylinder [6], and sea surface temperature (SST) [31].

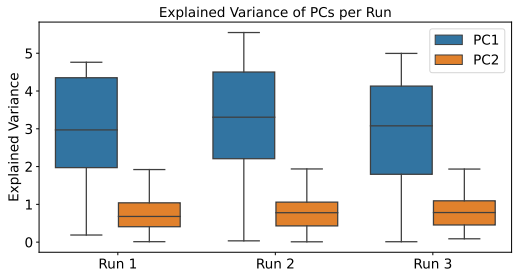
A. PARAMETER SELECTION

The first type of noise is sparse noise, where each entry in the data matrix $\mathbf{X}$ is corrupt with probability p. The value of corrupt entries is set to c. As the number of features grows, the number of corrupted entries will also grow. The next two types of noise are types of white noise, where noise from either the uniform distribution with mean 0 and variance v or the Gaussian distribution with mean 0 and variance v are added to $\mathbf{X}$. We ensure the variance is small with respect to the dominant singular value of $\mathbf{X}$, as this is a level of corruption under which classical PCA should still perform well [10]. In practice, we set $v = \frac{\sigma_1}{f}$, where σ_1 is the dominant singular value of $\mathbf{X}$ and f is some variance divisor. Finally, we create outliers by multiplying a randomly selected $s\%$ of the rows of $\mathbf{X}$ by an outlier scale of S. We expect RPCA to perform best on sparse data, PCA to perform well even in the presence of low-variance white noise [10], and EPCA to perform best on outlier data.

For all datasets, we perform EPCA with $B = 100$ bags, but



(a)



(b)

FIGURE 3. Uncertainty Quantification of three runs of EPCA on wave data corrupt with 5% outliers of scale 10. (a) The 95% Confidence Intervals (CIs) associated with the top two principal components. The CIs are wider than that of the clean data. (b) Boxplots of the distribution of the explained variance of each of the principal components. The true eigenvalues of the covariance matrix are contained within the interquartile ranges (IQRs) of all three trials, but the IQRs are skewed upward and wider than those from the clean data. Gross outliers have been removed from the boxplots for visualization purposes.

vary the size n of the bags. We take n to be larger on datasets with white noise and sparse noise and smaller on datasets with outliers. Since white noise is applied to all entries of $\mathbf{X}$ and sparse noise will impact most entries as $\mathbf{X}$ becomes higher-dimensional, choosing larger bag sizes n helps "mute" the impact of the noise. Contrastingly, since outliers impact only $s\%$ of entries, choosing smaller bags helps prevent the selection of an outlier. In RPCA, the parameter controlling the extent of the regularization on the sparse part of $\mathbf{X}$ is set to $\alpha = 0.20$. The remaining parameters for the Augmented Lagrange Multipliers algorithm used to solve RPCA are set to their default values.

B. EVALUATION

Given an initial data matrix $\mathbf{X}$, we calculate the true PCA modes using classical PCA. We then corrupt $\mathbf{X}$ and run PCA, RPCA, and EPCA on each of the corrupted datasets and quantify the percent relative error for the components using Equation 6.

$$\%\text{Relative Error} = \frac{||t - p||_2}{||t||_2} \times 100, \tag{6}$$

where t are the true components and p the predicted components. As we saw in Section IV, the eigenvalues of the covariance matrix in EPCA are skewed on noisy data. However, the eigenvalues are not involved in PCA's low-dimensional mapping, only in ordering the predicted components. Therefore, we consider only the error in the principal components in our experiments.

C. VARIOUS LEVELS OF NOISE

Our first set of experiments addresses how PCA, EPCA, and RPCA respond to various levels of noise. For sparse noise, we vary both the probability p of an entry being corrupt and the scale c of the corruption. For both normal and uniform white noise, we vary the scale of the variance divisor f. Finally, for outliers, we vary both the percentage s of corrupt rows and the scale S. We take each of six datasets, excluding cylinder and SST, and randomly add noise of a given level five times, resulting in 30 datasets per noise level. We do not include the cylinder and SST datasets in this analysis, as RPCA is unable to produce a result within 120s of runtime. Since the results of PCA and RPCA are deterministic,while the results of EPCA are stochastic due to the randomness in the bagging procedure, we run PCA and RPCA once and EPCA five times on each of the datasets. We average each method's respective relative errors together to get an average percent relative error for every level of corruption.

D. FIXED LEVELS OF NOISE

In the second set of experiments, we compare the performance of PCA, RPCA, and EPCA on datasets with fixed levels of corruption to test variability in performance. For sparse noise, we set the probability of an entry being corrupt to $p = 0.01$ and the scale of the corruption to $c = 2$, for both Gaussian and uniform white noise, we set the variance divisor to $f = 1000$, so that the variance is $v = \frac{\sigma_1}{1000}$, and for outliers, we corrupt $s = 5$ percent of rows with scale $S = 5$. For each of our eight datasets and each type of noise, we repeat random corruption and evaluation 100 times, resulting in 800 runs of each method per noise category. In our analysis, we consider only the output of a single run of EPCA. We return boxplots of percent relative error over the 100 trials. Once again, RPCA is unable to produce a result on either the cylinder or SST data before a timeout of 120s; therefore, RPCA's boxplots do not include performance metrics on cylinder or SST. We also carry out a runtime comparison among the three methods.

VI. RESULTS

A. VARIOUS LEVELS OF NOISE

Figure 4 compares the average performance of PCA, EPCA, and RPCA over datasets corrupted with various levels of sparse noise, determined by the probability p of an entry being corrupt and the scale c of the corruption. As we expect, RPCA performs best on sparse noise, most noticeably as the probability and scale of the noise increases. The exception is when the noise scale is set to 0, simulating missing data. However, it is possible that the RPCA regularization parameter α is simply not tuned optimally in this case. When $p = 0.01$, EPCA consistently performs second-best. When

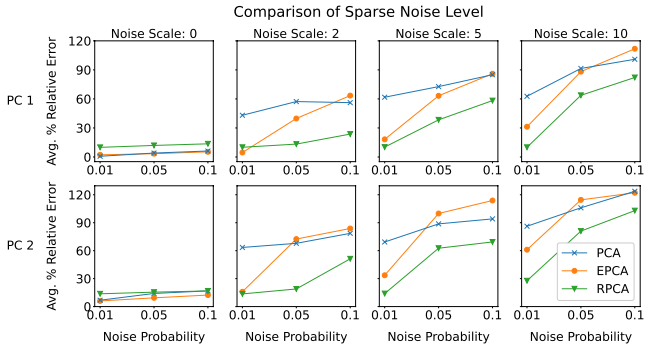


FIGURE 4. Average percent relative error of predicted principal components from PCA, EPCA, and RPCA for data corrupted with sparse noise of different probabilities and scales.

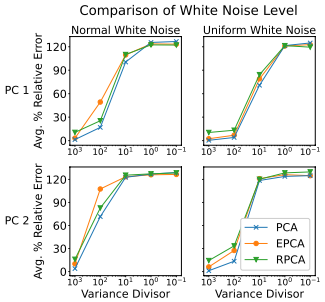


FIGURE 5. Average percent relative error of predicted principal components from PCA, EPCA, and RPCA for data corrupted with Gaussian and uniform white noise sampled from distributions of various variances.

$p = 0.05$, EPCA performs second-best in terms of error in the first principal component and slightly worse than PCA on the second principal component. Finally when $p = .10$, EPCA generally performs worst of the three methods. Ultimately, RPCA is the preferred choice on sparse data, but EPCA is a competitive choice when our noise probability is very small.

Figure 5 explores the performance of the three methods on data with added Gaussian and uniform white noise sampled from distributions with different variances $v = \frac{\sigma_1}{f}$. When the variance divisor f is very large, PCA performs best in terms of the average percent relative error in the first and second principal components. As the variance divisor becomes smaller, PCA, EPCA, and RPCA perform very similarly to one another, but PCA maintains its advantage.

Figure 6 displays the performance of PCA, EPCA, and RPCA on data corrupted with outliers at various percentages s and scales S. We observe that for both principal components, when the percentage of outliers is 15% or lower, EPCA consistently achieves the lowest average percent relative error, regardless of outlier scale. As the percentage of outliers increases, EPCA loses its advantage. We conclude that EPCA will generally outperform PCA and RPCA on outlier data, when the percentage of outliers remains small, regardless of the scale of those outliers.

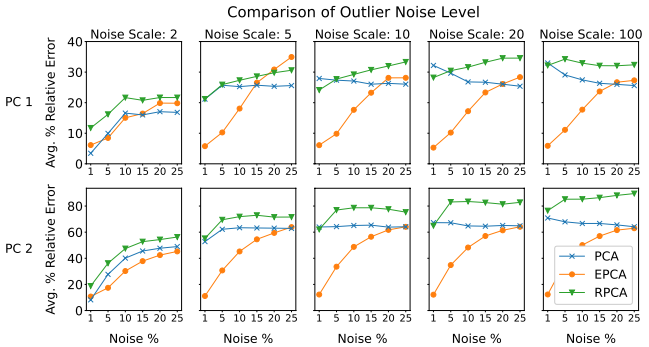


FIGURE 6. Average percent relative error of predicted principal components from PCA, EPCA, and RPCA for data corrupt with outliers of varied percentages and magnitudes.

B. FIXED LEVEL OF NOISE

We investigate the performance of PCA, EPCA, and RPCA over datasets with fixed levels of corruption. The first aspect of performance we consider is runtime. Let $X \in \mathbb{R}^{N\times m}$ be a data matrix with $N \gg m$. Then the complexity of classical PCA is $\mathcal{O}(Nm^2 + m^3)$, which will be dominated by the first term. With respect to RPCA, it is well known that runtime does not scale well with the size of the problem [5]. Each iteration of RPCA solved using ALM has a complexity of $\mathcal{O}(Nm^2)$, and the algorithm takes $\mathcal{O}(\frac{1}{\epsilon})$ iterations to converge to ϵ tolerance [27]. Thus, the overall complexity of RPCA is $\mathcal{O}(Nm^2\frac{1}{\epsilon})$, making it significantly slower than PCA on larger datasets. In contrast, since the runtime of EPCA is influenced by the number B and size n of bootstrapped PCA samples, we can mitigate runtime challenges on larger datasets. The complexity of the PCA runs in an EPCA procedure is $\mathcal{O}(Bmin(n,m)nm + Bm^3)$, while the complexity of the k-means clustering used to search for d unique principal components is $O(4d^2mi)$, where i is the number of iterations. Since we limit the number of iterations and are concerned with inherently low-rank data with only a few large eigenvalues (small d), the runtime of EPCA is in practice dominated by $\mathcal{O}(Bmin(n,m)nm + Bm^3)$. We expect EPCA to run faster than PCA if $\frac{B(n^2+m^2)}{m} < N$ and $n < m$ or if $B(n+m) < N$ and $n > m$. We also expect EPCA to scale significantly better than RPCA when $B\min(n,m)nm < Nm^2\frac{1}{\epsilon}$.

Recall that all experimental datasets are formed by adding one of four types of noise to eight base datasets 100 times. For each of our eight corrupted datasets, we create boxplots of the spread of the runtime of each method, as seen in Figure 7. We summarize runtime for digits 0 and 1 in MNIST together. Across all datasets but one, PCA achieves the fastest runtime. On the smallest datasets, iris ($\mathbf{X} \in \mathbb{R}^{140\times4}$), wine ($\mathbf{X} \in \mathbb{R}^{178\times13}$), and WBC ($\mathbf{X} \in \mathbb{R}^{569\times30}$), RPCA and EPCA run in time on the same order of magnitude, while PCA runs one to two orders of magnitude faster. On the medium-sized datasets wave ($\mathbf{X} \in \mathbb{R}^{6000\times200}$) and MNIST 0 and 1 ($\mathbf{X} \in \mathbb{R}^{5923\times784}$, $\mathbf{X} \in \mathbb{R}^{6742\times784}$), RPCA runs two orders of magnitude slower than EPCA, but PCA and EPCA run on the same order of magnitude. Finally, on the largest datasets

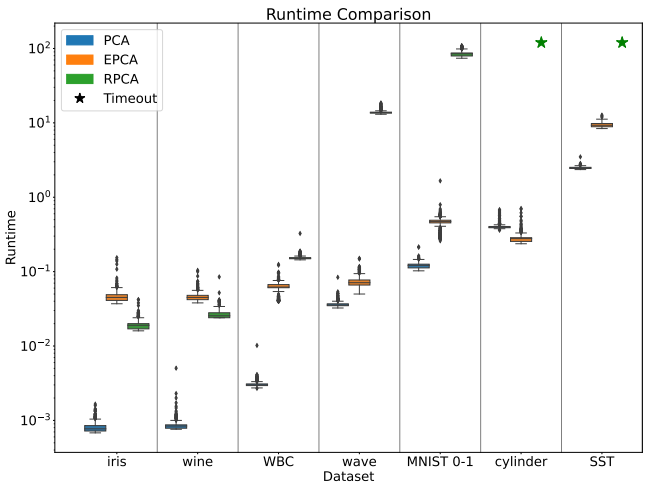


FIGURE 7. **Runtime Comparison of PCA, EPCA, and RPCA. For every corrupted dataset, we create boxplots of the spread of the runtime of 400 runs of each method. For MNIST, we summarize runtime for the digits 0 and 1 together. PCA consistently achieves the fastest runtime. On the smallest datasets, iris, wine, and WBC, RPCA and EPCA run in time on the same order of magnitude, while PCA runs one to two orders of magnitude faster. On the medium-sized datasets, wave and MNIST, RPCA runs two orders of magnitude slower than EPCA, but PCA and EPCA run on the same order of magnitude. Finally, on the largest cylinder and SST data, RPCA is unable to provide an output, timing out after 120s, while EPCA and PCA run on the same order of magnitude.**

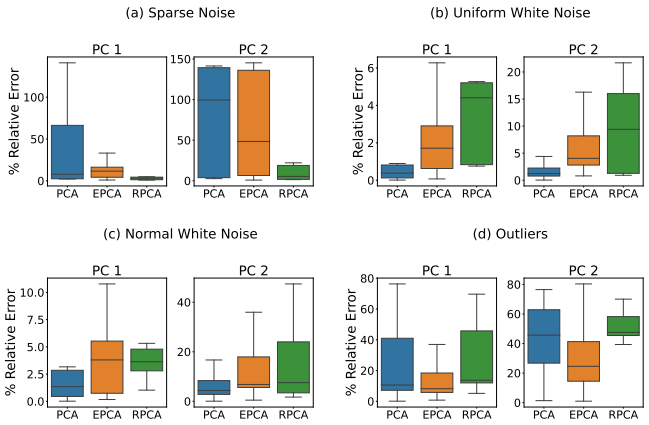


FIGURE 8. **Summary of the performance of PCA, EPCA, and RPCA on noisy data following 100 trials of random corruption and evaluation on eight datasets. RPCA's boxplots do not include performance metrics from the cylinder or SST datasets, due to timeout. (a) For sparse noise ($p = 0.01$, $c = 2$), RPCA outperforms the competitors with both lower and tighter IQRs for both principal components. EPCA performs second best with lower median error and tighter IQRs than PCA. (b) and (c) For data corrupted with uniform and Gaussian white noise ($v = \frac{\sigma}{1000}$), PCA performs best with tighter and lower IQRs. EPCA achieves the second lowest percent relative error on both PCs for uniform white noise and on the second PC for Gaussian white noise. (d) For outliers ($s = 5$, $S = 5$), EPCA outperforms the competitors with lower median errors for both principal components. EPCA also achieves the tightest IQR on the first and second tightest IQR on the second PC.**

cylinder ($\mathbf{X} \in \mathbb{R}^{89351\times151}$) and SST ($\mathbf{X} \in \mathbb{R}^{1726\times64800}$), RPCA is unable to provide an output, timing out after 120s, while EPCA and PCA run on the same order of magnitude. We conclude that on small datasets, RPCA and EPCA have similar runtime. However, unlike RPCA, which is infeasible to run on larger datasets, our method EPCA scales no worse than classical PCA.

The second aspect of performance that we consider is percent relative error in the predicted first and second components. Figure 8 shows boxplots of error for each of the three methods for 100 runs of each type of data corruption over our eight datasets. Outliers in the boxplots have been removed for easier visualization. As expected, on datasets where sparse noise is added, RPCA is able to identify the true principal components with the lowest median percent relative errors and the least variability in its results, as evidenced by tighter interquartile ranges (IQRs). Recall that this performance comes at the cost of a much higher runtime. Though EPCA performs with significantly more error than RPCA, we note that EPCA achieves a similar median error to PCA, as well as a tighter IQR for error in the first PC and both a lower median error and slightly tighter IQR for the second PC.

For both types of white noise, classical PCA outperforms the other methods with both the smallest IQRs and lowest median errors for both principal components. On uniform white noise, EPCA performs second best in both categories, while on normal white noise, RPCA performs second best for the first PC and EPCA for the second PC in both categories.

Finally, on datasets containing outliers, EPCA outperforms the other methods, achieving a lower median percent relative error for both principal components. EPCA also has the tightest IQR for the first PC and the second-tightest for the second PC. We showcase this performance advantage on our two largest datasets. Figure 9(a) visually compares the predicted PCs of PCA and EPCA on the flow around a cylinder data corrupt with outliers. We note that although EPCA underperforms in the first PC, EPCA identifies the second and third PCs much closer than PCA, which is unable to capture the temporal shift. We also provide a comparison of the spatial modes, as calculated from the predicted PCs in Figure 9(b). Figure 10 visualizes the predicted PCs of PCA and EPCA on corrupted SST data. Both methods perform similarly in the first PC, but EPCA has a clear advantage on the second PC. In both figures, the results of RPCA are omitted, as the method times out before producing an output.

VII. CONCLUSION

We propose EPCA as a scalable extension of PCA that operates well in the presence of various types of noise and lends itself naturally to uncertainty quantification. Our innovative ensembling of boostrapping and k-means clustering allows us to automatically handle the challenges of principal component re-ordering and sign ambiguity in boostrapping PCA. We test the performance of EPCA against RPCA, which is specialized for datasets corrupt with sparse noise, and classical PCA, which should operate well even in the presence of low variance white noise. Further, we carry out a runtime analysis of all three methods on datasets of various sizes.

Overall, EPCA is unable to outperform RPCA on sparsely corrupted data or classical PCA on data with added white

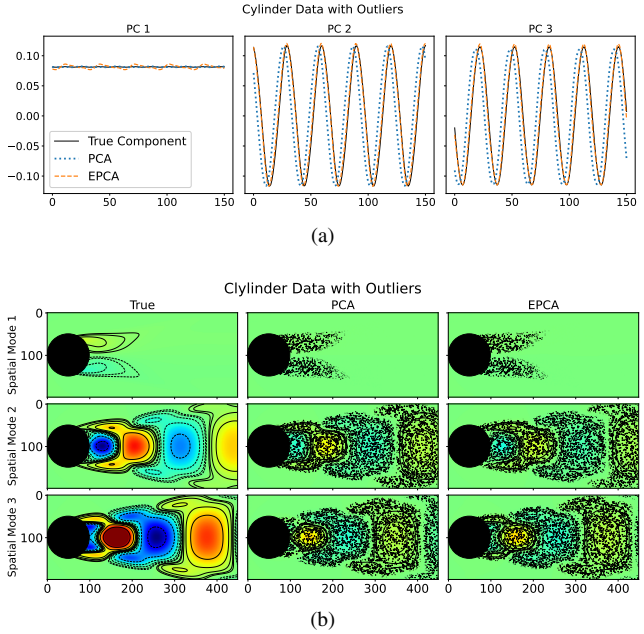


FIGURE 9. Comparison of output of classical PCA and EPCA on the flow around a cylinder dataset corrupt with outliers ($s = 5, S = 5$). (a) Although EPCA has more error than PCA in the first PC, EPCA much better identifies the second and third PCs. PCA is unable to capture the time shift. (b) We use the predicted PCs of PCA and EPCA to generate the spatial modes, and note that EPCA's spatial modes more closely match the ground truth.

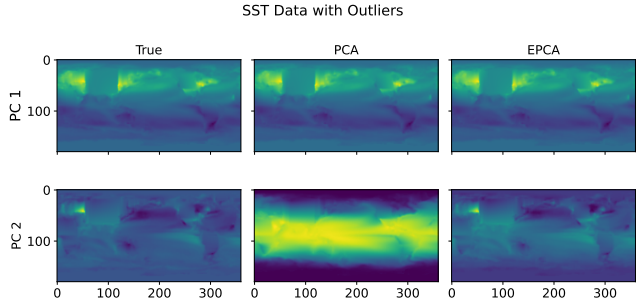


FIGURE 10. Comparison of principal components of classical PCA and EPCA on the SST dataset corrupt with outliers ($s = 5, S = 5$). EPCA clearly identifies the second PC more accurately than PCA.

noise. However, EPCA performs second-best in both noise domains. On datasets containing gross outliers, EPCA significantly outperforms both PCA and RPCA and maintains this performance advantage, regardless of the scale of the outliers, as long as the percentage of outliers remains low. We conclude that although EPCA has a noise domain where it performs best, the method remains useful for all types of data corruption. An added bonus of using EPCA is that unlike classical PCA or RPCA, confidence intervals for the components and the explained variance of those components can be computed naturally. Finally, as data size increases, EPCA is significantly faster than RPCA and scales no worse than PCA, making it particularly attractive for the analysis of larger datasets. As each PCA subroutine in EPCA is independent of the rest, the potential for the parallelization of EPCA should be explored as future work.

Code Availablity: https://github.com/OlgaD400/EPCA

REFERENCES

[1] Hamid Babamoradi, Frans van den Berg, and Åsmund Rinnan. Bootstrap based confidence limits in principal component analysis—a case study. *Chemometrics and Intelligent Laboratory Systems*, 120:97–105, 2013.

[2] Mukund Balasubramanian and Eric L Schwartz. The isomap algorithm and topological stability. *Science*, 295(5552):7–7, 2002.

[3] Peter Benner, Serkan Gugercin, and Karen Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM review*, 57(4):483–531, 2015.

[4] Gal Berkooz, Philip Holmes, and John L Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual review of fluid mechanics*, 25(1):539–575, 1993.

[5] Thierry Bouwmans and El Hadi Zahzah. Robust PCA via principal component pursuit: A review for a comparative evaluation in video surveillance. *Computer Vision and Image Understanding*, 122:22–34, 2014.

[6] Steven L Brunton and J Nathan Kutz. *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press, 2022.

[7] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):1–37, 2011.

[8] John P Cunningham and Zoubin Ghahramani. Linear dimensionality reduction: Survey, insights, and generalizations. *The Journal of Machine Learning Research*, 16(1):2859–2900, 2015.

[9] Li Deng. The MNIST database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[10] Xinghao Ding, Lihan He, and Lawrence Carin. Bayesian robust principal component analysis. *IEEE Transactions on Image Processing*, 20(12):3419–3430, 2011.

[11] R Dony et al. Karhunen-loeve transform. *The transform and data compression handbook*, 1(1-34):29, 2001.

[12] Petros Drineas and Michael W Mahoney. RandNLA: randomized numerical linear algebra. *Communications of the ACM*, 59(6):80–90, 2016.

[13] N Benjamin Erichson, Sergey Voronin, Steven L Brunton, and J Nathan Kutz. Randomized matrix decompositions using R. *arXiv preprint arXiv:1608.02148*, 2016.

[14] Aaron Fisher, Brian Caffo, Brian Schwartz, and Vadim Zipunnikov. Fast, exact bootstrap principal component analysis for p> 1 million. *Journal of the American Statistical Association*, 111(514):846–860, 2016.

[15] Bogdan Gabrys, Bruno Baruque, and Emilio Corchado. Outlier resistant PCA ensembles. In *Knowledge-Based Intelligent Information and Engineering Systems: 10th International Conference, KES 2006, Bournemouth, UK, October 9-11, 2006. Proceedings, Part III 10*, pages 432–440. Springer, 2006.

[16] Gene H Golub and Christian Reinsch. Singular value decomposition and least squares solutions. In *Handbook for Automatic Computation: Volume II: Linear Algebra*, pages 134–151. Springer, 1971.

[17] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[18] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

[19] Abdel Hannachi, Ian T Jolliffe, and David B Stephenson. Empirical orthogonal functions and related techniques in atmospheric science: A review. *International Journal of Climatology: A Journal of the Royal Meteorological Society*, 27(9):1119–1152, 2007.

[20] Harold Hotelling and Lester R Frankel. The transformation of statistics to simplify their distribution. *The Annals of Mathematical Statistics*, 9(2):87–96, 1938.

[21] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, et al. *An introduction to statistical learning*, volume 112. Springer, 2013.

[22] Ian Jolliffe. Principal component analysis. *Encyclopedia of statistics in behavioral science*, 2005.

[23] Julie Josse and François Husson. Handling missing values in exploratory multivariate data analysis methods. *Journal de la Société Française de Statistique*, 153(2):79–99, 2012.

[24] Noureddine El Karoui and Elizabeth Purdom. The bootstrap, covariance matrices and PCA in moderate and high-dimensions. *arXiv preprint arXiv:1608.00948*, 2016.
[25] Michael Kirby and Lawrence Sirovich. Application of the karhunen-loeve procedure for the characterization of human faces. *IEEE Transactions on Pattern analysis and Machine intelligence*, 12(1):103–108, 1990.
[26] J Nathan Kutz. *Data-driven modeling & scientific computation: methods for complex systems & big data*. OUP Oxford, 2013.
[27] Zhouchen Lin, Minming Chen, and Yi Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *arXiv preprint arXiv:1009.5055*, 2010.
[28] Edward N Lorenz. *Empirical orthogonal functions and statistical weather prediction*, volume 1. Massachusetts Institute of Technology, Department of Meteorology Cambridge, 1956.
[29] Sebastian Mika, Bernhard Schölkopf, Alex Smola, Klaus-Robert Müller, Matthias Scholz, and Gunnar Rätsch. Kernel PCA and de-noising in feature spaces. *Advances in neural information processing systems*, 11, 1998.
[30] Luis Milan and Joe Whittaker. Application of the parametric bootstrap to models that incorporate a singular value decomposition. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 44(1):31–49, 1995.
[31] NASA Earth Observations. Sea surface temperature. Technical report, NASA, 2012.
[32] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
[33] Vin Silva and Joshua Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. *Advances in neural information processing systems*, 15, 2002.
[34] Marieke E Timmerman, Henk AL Kiers, and Age K Smilde. Estimating confidence intervals for principal component loadings: a comparison between the bootstrap and asymptotic results. *British Journal of Mathematical and Statistical Psychology*, 60(2):295–314, 2007.
[35] Michael E Tipping and Christopher M Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 61(3):611–622, 1999.
[36] S Van Aelst and G Willems. PCA based on multivariate MM-estimators with fast and robust bootstrap. *Preprint. MR2085872*, 2004.
[37] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of machine learning research*, 9(11), 2008.
[38] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.
[39] John Wright and Yi Ma. *High-dimensional data analysis with low-dimensional models: Principles, computation, and applications*. Cambridge University Press, 2022.
[40] Zihan Zhou, Xiaodong Li, John Wright, Emmanuel Candes, and Yi Ma. Stable principal component pursuit. In *2010 IEEE international symposium on information theory*, pages 1518–1522. IEEE, 2010.
[41] Ziwei Zhu, Tengyao Wang, and Richard J Samworth. High-dimensional principal component analysis with heterogeneous missingness. *arXiv preprint arXiv:1906.12125*, 2019.
[42] Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal component analysis. *Journal of computational and graphical statistics*, 15(2):265–286, 2006.

OLGA DORABIALA received the B.S. degree in mathematics from Pennsylvania State University, University Park, PA, USA in 2018, and the M.S. and Ph.D. degrees in applied mathematics from the University of Washington, Seattle, WA, USA in 2019 and 2023, respectively. She is currently a post-doctoral scholar at the University of Washington and a visiting researcher at the Alan Turing Institute, London, UK.

ALEKSANDR Y. ARAVKIN (Member, IEEE) received the B.S. degree in mathematics and computer science, the M.S. degree in statistics, and the Ph.D. degree in mathematics (optimization) from the University of Washington, Seattle, WA, USA, in 2004, 2010, and 2010, respectively. He is currently an Associate Professor of applied mathematics, the Director of mathematical sciences with the Institute for Health Metrics Sciences, an Associate Adjunct Professor of mathematics, statistics, and health metrics sciences, and a Data Science Fellow with the eScience Institute, University of Washington.

J. NATHAN KUTZ (Senior Member, IEEE) is the Yasuko Endo and Robert Bolles Professor of Applied Mathematics and Electrical and Computer Engineering at the University of Washington, having served as chair of applied mathematics from 2007-2015. He is also the Director of the NSF AI Institute in Dynamic Systems (dynamicsAI.org) where he leads a research effort in developing machine learning and AI algorithms for physics and engineering based systems where parsimony and explainability are critical for the design and understanding of such systems. He also has a wide range of interests, including neuroscience to fluid dynamics where he integrates machine learning with dynamical systems and control. During the academic year 2023-2024, he is a visiting professor at Imperial College London and the Alan Turning Institute.