Online Dominant Generalized Eigenvectors Extraction via a Randomized Algorithm

Haoyuan Cai, Maboud Kaloorazi, Jie Chen, Wei Chen, Cédric Richard

This article has been accepted for publication in IEEE Transactions on Vehicular Technology. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVT.2023.3243244

Online Dominant Generalized Eigenvectors Extraction via a Randomized Algorithm

Haoyuan Cai, *Student Member, IEEE*, Maboud Kaloorazi, *Member, IEEE*, Jie Chen, *Senior Member, IEEE*, Wei Chen, *Senior Member, IEEE* and Cédric Richard, *Senior Member, IEEE*

***Abstract*—This paper is concerned with online algorithms for the generalized Hermitian eigenvalue problem (GHEP). We first present an algorithm based on randomization, termed alternate-projections randomized eigenvalue decomposition (APR-EVD), to solve the standard eigenvalue problem. The APR-EVD algorithm is computationally efficient and can be computed by making only one pass through the input matrix. We then develop two online algorithms based on APR-EVD for the dominant generalized eigenvectors extraction. Our proposed algorithms use the fact that GHEP is transformed into a standard eigenvalue problem, however to avert computations of a matrix inverse and inverse of the square root of a matrix, which are prohibitive, they exploit the rank-1 strategy for the transformation. Our algorithms are devised for extracting generalized eigenvectors for scenarios in which observed stochastic signals have unknown covariance matrices. The effectiveness and practical applicability of our proposed algorithms are validated through numerical experiments with synthetic and real-world data.**

***Index Terms*—Randomized algorithms, dominant generalized eigenvector extraction, online algorithms, matrix decomposition, fast subspace estimation and tracking, real-time hyperspectral image denoising.**

I. Introduction

GHEP plays a vital role in many signal processing and machine learning applications, such as sound zone generalization [2], nonorthogonal multiple access system [3], hyperspectral image denoising [4], canonical correlation analysis [5], linear discriminant analysis [6], and multichannel Wiener filtering [7], to name a few. Given a matrix pencil $(\mathbf{R}_y, \mathbf{R}_x)$, where $\mathbf{R}_y$, $\mathbf{R}_x \in \mathbb{C}^{N\times N}$ are Hermitian and positive definite, the generalized Hermitian eigenvalue problem (GHEP) [8] is defined follows:

$$\mathbf{R}_y\mathbf{w}_i = \lambda_i\mathbf{R}_x\mathbf{w}_i, i = 1, ..., N, \tag{1}$$

where $\mathbf{w}_1, \cdots, \mathbf{w}_N \in \mathbb{C}^N \backslash \{\mathbf{0}\}$ are generalized eigenvectors corresponding to N generalized eigenvalues $\lambda_1 > \lambda_2 > \cdots > \lambda_N > 0$. The ith generalized eigen-pair is denoted by $(\mathbf{w}_i, \lambda_i)$. To solve the GHEP, in many cases, one can transform (1) into a Hermitian or non-Hermitian eigenvalue problem. For the former case, provided that $\mathbf{R}_x^{1/2}(\mathbf{R}_x^{1/2})^H = \mathbf{R}_x$, the set of eigenvectors $\mathbf{w}$ is obtained by $(\mathbf{R}_x^{-1/2})^H\mathbf{v}$, where $\mathbf{v}$ is the set of eigenvectors of $\mathbf{R}_x^{-1/2}\mathbf{R}_y(\mathbf{R}_x^{-1/2})^H$. Here $\mathbf{v}$ is determined so that $\mathbf{w}^T\mathbf{R}_x\mathbf{w} = \mathbf{I}$. For the latter case, transforming into a non-HEP, (1) reduces to $\mathbf{R}_x^{-1}\mathbf{R}_y\mathbf{w} = \mathbf{\Lambda}\mathbf{w}$, where $\mathbf{\Lambda}$ contains λ_i's, and consequently solved by a relevant standard algorithm [8]. Traditional methods for solving the GHEP include power and inverse iteration based methods, Lanczos method and Jacobi-Davidson method [8]. The bottleneck of these methods, however, lies in the computation of a matrix inverse as well as inverse of the square root of a matrix, which is demanding particularly for large data matrices.

This paper focuses on developing adaptive algorithms in order to fast extract and track the generalized eigenvectors for online applications in which observed stochastic signals have unknown covariance matrices, that is, matrices $\mathbf{R}_y$ and $\mathbf{R}_x$ are time-variant and required to be estimated. The traditional methods (also called batch methods) are inefficient and, in some cases, infeasible to apply for such applications due to their computational workload. To overcome this drawback, adaptively computing generalized eigenvectors methods were proposed [9]–[16]. These methods can be found in a wide variety of applications including blind source separation [17], [18], feature extraction [19], [20], noise filtering [21], fault detection [22], antenna array processing [9]–[11], classification [23], speech enhancement [24]. The adaptive methods presented in [9]–[11] are gradient-based, and extract the first dominant (or principal) generalized eigenvector. These methods, however, are unsuitable for applications where multiple dominant generalized eigenvectors are desired [24], [25]. To address this issue, after extracting the principal eigenvector, [12] used the deflation technique, and the works in [15], [16] proposed to use nested orthogonal complement structure of the generalized eigensubspace in order to compute the remaining generalized eigenvectors in a decreasing order of their importance. However, the shortcomings of gradient-based methods are that (i) they converge slowly, (ii) it is challenging to determine an appropriate learning rate to guarantee tracking speed and numerical stability, and (iii) it is difficult to parallelize them (in order to exploit modern computational platforms), thereby making them unsuitable for large data matrices. The methods in [15], [16] are called coupled methods, and extract generalized eigenvectors by using the orthogonal projection technique, which make them challenging for parallelization. Yang et al. [12] proposed recursive least-square (RLS)-type adaptive algorithms based on the projection

A short and preliminary version of this work appears in the 28th European Signal Processing Conference (EUSIPCO), the Netherlands, Jan. 2021 [1].

H. Cai, M. Kaloorazi and J. Chen are with Research and Development Institute of Northwestern Polytechnical University in Shenzhen, Shenzhen, 518063, China, and with School of Marine Science and Technology, Northwestern Polytechnical University, Xi'an, 710072, China (e-mail: haoyuan.cai@mail.nwpu.edu.cn, kaloorazi@nwpu.edu.cn, dr.jie.chen@ieee.org). W. Chen is with State Key Laboratory of Rail Traffic Control and Safety, Beijing Jiaotong University, Beijing, China (e-mail: weich@bjtu.edu.cn). Cédric Richard is with Université Côte d'Azur, Nice, France (Corresponding author: J. Chen).

This article has been accepted for publication in IEEE Transactions on Vehicular Technology. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVT.2023.3243244

approximation subspace tracking (PAST) technique [26] for $r \geq 1$ dominant generalized eigenvectors extraction. To reduce the computational costs of the RLS-based methods, the work in [13] presented R-GEVE (reduced-rank generalized eigenvector extraction) based on the reduced rank idea [27], which, instead of directly solving the GHEP of size $N \times N$, reduces it into an $r(r+1)$-dimensional subspace by searching the current generalized eigenvectors in the subspace spanned by the previous generalized eigenvectors and the current observed sample vector. The shortcoming of this method, however, is that the dimension of the subspace where eigenvectors are searched is very narrow in the sense that it has limitation in fast tracking [14]. Tanaka [14] developed an adaptive algorithm to extract multiple generalized eigenvectors based on the power iteration scheme. The algorithm uses a rank-1 update strategy, and transforms the GHEP (1) into a HEP. However, the main steps needed in each iteration include $\mathcal{O}(rN^2)$ floating-point operations (flops) when tracking the r-dominant generalized eigenvectors, which is still computationally expensive.

Recently developed algorithms based on randomization have been shown to be highly accurate for low-rank matrix approximation [28]–[36]. They are computationally efficient and easy to implement on parallel machines. However, these algorithms provide approximation for a single matrix and hence can not be directly applied to generalized eigenvectors tracking. The work in [37] proposed a randomized algorithm for GHEP. However it considers a given matrix pencil and matrices are not assumed to be varying over time. The work in [1], which contains primary results of the current work, utilizes randomization for generalized eigenvectors tracking. This work substantially extends [1] by presenting the proof for APR-EVD as well as convergence behavior of the proposed online tracking algorithms. In addition, we present more numerical results, which further demonstrate the effectiveness and efficiency of the algorithms. Our work considers streaming data and seeks a reduced-size problem as in [13], however our algorithms search the generalized eigenvectors in the space comprised the collection of observed sample vectors up to k, i.e., $\sum_{i=1}^{k} \alpha^{k-i}\mathbf{x}(i)\mathbf{x}^H(i)$ and $\sum_{i=1}^{k} \beta^{k-i}\mathbf{y}(i)\mathbf{y}^H(i)$, where α and β are forgetting factors. This naturally extends the subspace where generalized eigenvectors are sought.

A. Summary of Contributions

In this paper, firstly we present an efficient algorithm termed APR-EVD (alternate-projections randomized eigenvalue decomposition) to solve a standard eigenvalue problem. We develop an error bound for this algorithm, and further empirically investigate the tightness of the bound via two matrices. Then, based on APR-EVD, we devise two fast and efficient algorithms for dominant generalized eigenvectors extraction: we harness the rank-1 update strategy to transform the GHEP into HEP and non-HEP, and then apply the APR-EVD algorithm to track the principal eigenvectors. The proposed algorithms have fast tracking speed in a time-varying environment, as will be demonstrated in simulation sections. Our proposed algorithms takes $\mathcal{O}(N^2)$ flops during each necessary iterative computations, which achieves the better balance between fast tracking and lower computation among several existing methods. Further, they can be parallelized on modern computers due to compounding randomized sampling techniques and rank-1 update strategy. In addition, the proposed algorithms are investigated under various settings and scenarios. In particular, they are employed for the task of real-time denoising of hyperspectral images.

The remainder of this paper is organized as follows. In Section II, a novel randomized algorithm for solving standard eigenvalue problems is proposed. In Section III, we develop two online algorithms for the extraction of r-dominant generalized eigenvectors. We provide a detailed description of the implementation and computational complexity of the the proposed algorithms. In Section IV, we establish theoretical analysis for the proposed algorithms under the assumption that the covariance estimations are unbiased. We also provide the low-rank approximation error bound for APR-EVD. In Section V, we present numerical experimental results and discuss the efficacy and efficiency of our proposed algorithms in relation to existing algorithms. Concluding remarks are given in Section VI.

B. Notation

Normal fonts x and X denote scalar. Boldface small letters $\mathbf{x}$ and capital letters $\mathbf{X}$ denote column vectors and matrices, respectively. $\mathbb{C}$ denotes the complex domain. The superscript $(\cdot)^*$ denotes the conjugate of a complex number, $(\cdot)^{\mathrm{H}}$ denotes the Hermitian transpose operator, and $(\cdot)^\dagger$ denotes the pseudo-inverse of a matrix. $\mathbf{I}_N$ denotes an identity matrix of order N. $\mathtt{orth}_{\mathtt{r}}(\cdot)$ constructs an orthonormal basis with r columns for the range of a matrix. The notation $||\cdot||_2$ denotes the ℓ_2-norm.

II. Alternate-Projections Randomized EVD (APR-EVD)

This section presents our algorithm (APR-EVD) for solving the standard eigenvalue problem. APR-EVD utilizes randomization and forms the building block of online generalized eigenvectors extraction algorithms presented in the next section.

In many practical applications, the signal subspace spanned by the dominant generalized eigenvectors lies in a low-dimensional space [24]. This implies that low-rank approximation techniques, which approximate an input matrix by one of lower rank, can be applied to treat GHEPs. Recent low-rank approximation methods based on randomized sampling [33]–[35] are computationally efficient. In addition, they can harness advanced computer architectures. In this section, we propose the APR-EVD algorithm. The proposed online algorithms, presented in the next section, will extract generalized eigenvectors by applying APR-EVD to a transformation of (1).

A. The APR-EVD Algorithm

Our proposed APR-EVD algorithm is described as follows. First, an orthonormal basis for the range of a general square input matrix $\mathbf{A} \in \mathbb{C}^{N\times N}$ is obtained. Next, a representation of $\mathbf{A}$ is computed by the basis. Finally, the left or right dominant

eigenvectors and corresponding eigenvalues are extracted by utilizing the Rayleigh-Ritz process. Before proceeding with the detailed procedure of our algorithm, we assume that $\mathbf{A}$ has only r independent columns, i.e., the numerical rank of $\mathbf{A}$ is r. This implies that the range of $\mathbf{A}$ can be constructed with r independent orthonormal columns.

Our decomposition is constructed as follows: we first generate a random matrix $\mathbf{\Psi} \in \mathbb{C}^{N\times d}$, where $r \leq d < N$; we consider random variables with standard Gaussian distribution (alternative distributions, e.g., the subsampled randomized Fourier transform (SRFT) [38], can also be used). Then, we construct the $N \times d$ matrix:

$$\mathbf{G} = \mathbf{A}^H\mathbf{\Psi}. \tag{2}$$

Matrix $\mathbf{G}$ is a projection onto the row space of $\mathbf{A}$ by $\mathbf{\Psi}$. Next, we form the $N \times d$ matrix:

$$\mathbf{H} = \mathbf{AG}. \tag{3}$$

Matrix $\mathbf{H}$ is a projection onto the column space of $\mathbf{A}$ by $\mathbf{G}$. After, we orthonormalize the columns of $\mathbf{H}$ in order to obtain a basis $\mathbf{Q}$. This operation can efficiently be performed by a call to a packaged QR decomposition. We only keep the first r columns, i.e., $\mathbf{Q}$ is of size $N \times r$:

$$\mathbf{Q} = \mathtt{orth}_{\mathtt{r}}(\mathbf{H}). \tag{4}$$

Note that the rank of $\mathbf{H}$ is at most r [39], and $\mathbf{Q}$ approximates the range of $\mathbf{A}$. Through exploiting $\mathbf{Q}$, we use the Rayleigh-Ritz method [40], [41] to compute a Ritz pair $(\mathbf{\Lambda}, \mathbf{V})$ of the following matrix:

$$\mathbf{T} = \mathbf{Q}^H\mathbf{AQ}. \tag{5}$$

Thus

$$\mathbf{U}_r \triangleq \mathbf{QV}. \tag{6}$$

and $\mathbf{\Lambda}$ constitute the approximate eigenpair of $\mathbf{A}$.

Provided that the matrix $\mathbf{A}$ is stored in the row-major format or is revealed rows by rows, equations (2) and (3) can be computed through a single pass over $\mathbf{A}$. The APR-EVD algorithm thus needs two passes over the input matrix. However, by approximating equation (5), we devise a single-pass algorithm, which can be directly employed for streaming data processing. We estimate matrix $\mathbf{T}$ through a pre-multiplication of the identity in (5) by $\mathbf{\Psi}^H\mathbf{Q}$:

$$\mathbf{\Psi}^H\mathbf{QT} = \mathbf{\Psi}^H\mathbf{QQ}^H\mathbf{AQ}.$$

Having known that $\mathbf{A} \approx \mathbf{QQ}^H\mathbf{A}$ and by the definition of $\mathbf{G}$ (2), an estimation $\widetilde{\mathbf{T}}$ is given by:

$$\widetilde{\mathbf{T}} = [\mathbf{\Psi}^H\mathbf{Q}]^\dagger[\mathbf{G}^H\mathbf{Q}]. \tag{7}$$

B. Computational Complexity Bound for APR-EVD

To construct an approximation to an $N \times N$ matrix $\mathbf{A}$, APR-EVD incurs the following costs: generating a matrix of Gaussian random variables costs $\mathcal{O}(Nd)$ flops. Forming $\mathbf{G}$ (2) and $\mathbf{H}$ (3) each costs $\mathcal{O}(N^2d)$ flops. Generating $\mathbf{Q}$ (4) costs $\mathcal{O}(Nd^2)$ flops. Considering the estimation to (5), forming $\widetilde{\mathbf{T}}$ and computing an eigenpair costs $\mathcal{O}(Nr^2) + \mathcal{O}(r^3)$ flops. The operation count of the decomposition is dominated by multiplications of $\mathbf{A}$, and we thus have the complexity of order

$$\mathcal{C}_{\text{APR-EVD}} = \mathcal{O}(N^2d).$$

Here d (the sampling size parameter) is very close to r.

III. Proposed Adaptive Algorithms

By transforming (1) into either HEP or non-HEP, the APR-EVD algorithm is capable of extracting the generalized eigenvectors with considerably less computation than batch method. However, integrating the construction of random structured subspace (2) and (3) with online statistic estimation of $\mathbf{R}_x$ and $\mathbf{R}_y$, we have obtained two efficient tracking procedures, which operates in the reduced-size level and avoids prohibitive matrix-matrix multiplication, matrix inverse and square root of inverse of matrix operations. We firstly discuss the online estimation of matrix pencil $(\mathbf{R}_y, \mathbf{R}_x)$ in order to obtain these two tracking procedures.

In many signal and information processing applications, $\mathbf{R}_x$ and $\mathbf{R}_y$ are associated to data covariance matrices. Let the covariance matrices of zero-mean stochastic vectors $\mathbf{x}(k) \in \mathbb{C}^N$ and $\mathbf{y}(k) \in \mathbb{C}^N$ be given by $\mathbf{R}_x = E\{\mathbf{x}(k)\mathbf{x}^H(k)\}$ and $\mathbf{R}_y = E\{\mathbf{y}(k)\mathbf{y}^H(k)\}$. When processing online streaming data, at each instant k scaled versions of these matrices are typically estimated by time averaging with the most recent data by

$$\mathbf{R}_x(k) = \alpha\mathbf{R}_x(k-1) + \mathbf{x}(k)\mathbf{x}^H(k), \tag{8}$$

$$\mathbf{R}_y(k) = \beta\mathbf{R}_y(k-1) + \mathbf{y}(k)\mathbf{y}^H(k), \tag{9}$$

where parameters $\alpha \in (0, 1)$ and $\beta \in (0, 1)$ are smoothing constants.

Considering the above setting, this section describes two algorithms that address the GHEP in an online manner. They first reduce (1) to a standard eigenvalue problem as presented at the beginning of this paper, and then apply APR-EVD to track the generalized principal eigenvectors. In Algorithm 1, the problem (1) is transformed to a HEP by integrating the statistics from streaming data, where we track the random structured subspace of $\mathbf{R}_x^{-1/2}(k)\mathbf{R}_y(k)(\mathbf{R}_x^{-1/2}(k))^H$ to extract the r-dominant generalized eigenvectors. Whereas, the problem (1) is transformed to a non-HEP in Algorithm 2, and the random structured subspace of $\mathbf{R}_x^{-1}(k)\mathbf{R}_y(k)$ is tracked simultaneously.

A. Extracting r-dominant generalized eigenvectors by tracking the random structured subspace of $\mathbf{R}_x^{-1/2}(k)\mathbf{R}_y(k)(\mathbf{R}_x^{-1/2}(k))^H$

By exploiting rank-1 update strategy under the scenario of streaming data, we can efficiently compute the random structured subspace of a input matrix described in equations (2) and (3). It is known that GHEP can be treated as a standard eigenvalue problem. Therefore, one way is to track the random structured subspace $\mathbf{R}_x^{-1/2}(k)\mathbf{R}_y(k)(\mathbf{R}_x^{-1/2}(k))^H\mathbf{\Psi}$. Updating this random structured subspace requires to update $\mathbf{R}_x^{-1/2}(k)\mathbf{R}_y(k)(\mathbf{R}_x^{-1/2}(k))^H$ beforehand. However, directly updating $\mathbf{R}_x^{-1/2}(k)\mathbf{R}_y(k)(\mathbf{R}_x^{-1/2}(k))^H$, involves computations of matrix square root and its inversion, as expounded in Section I. This is computationally prohibitive,

i.e., it requires $\mathcal{O}(N^3)$ operations. We thus adopt the rank-1 update strategy presented in [14] to recursively compute $\mathbf{R}_x^{-1/2}(k)\mathbf{R}_y(k)(\mathbf{R}_x^{-1/2}(k))^H$, where these steps involve equations (10) - (13). For ease of notation, let

$$\begin{aligned}\mathbf{K}(k) &= \mathbf{R}_x^{-1/2}(k),\\ \mathbf{R}(k) &= \mathbf{K}(k)\mathbf{R}_y(k)\mathbf{K}^H(k).\end{aligned} \tag{10}$$

The procedure of this algorithm (hereafter Algorithm 1) involves updating $\mathbf{R}(k)$, recursively forming $\mathbf{G}(k) = \mathbf{R}^H(k)\mathbf{\Psi}(k)$, and finally $\mathbf{H}(k) = \mathbf{R}(k)\mathbf{G}(k)$, which enables the extraction of orthonormal basis $\mathbf{Q}(k)$. $\mathbf{R}(k)$ and $\mathbf{K}(k)$ are updated through the following equations:

$$\begin{aligned}\mathbf{R}(k) =&\frac{1}{\alpha}\Big[\beta\mathbf{R}(k-1) + \tilde{\mathbf{y}}(k)\tilde{\mathbf{y}}^H(k)\\ &+ \tilde{\mathbf{x}}(k)\mathbf{c}^H(k) + \delta_1(k)\mathbf{h}(k)\tilde{\mathbf{x}}^H(k)\Big],\\ \mathbf{K}(k) =& \frac{1}{\sqrt{\alpha}}\mathbf{K}(k-1) + \delta_1(k)\tilde{\mathbf{x}}(k)\bar{\mathbf{x}}^H(k).\end{aligned} \tag{11}$$

where

$$\begin{aligned}\tilde{\mathbf{y}}(k) &= \mathbf{K}(k-1)\mathbf{y}(k),\\ \tilde{\mathbf{x}}(k) &= \frac{1}{\sqrt{\alpha}}\mathbf{K}(k-1)\mathbf{x}(k),\\ \mathbf{c}(k) &= \delta_2^*(k)\tilde{\mathbf{x}}(k) + \delta_1(k)\mathbf{h}(k),\\ \mathbf{h}(k) &= \beta\mathbf{r}_x(k) + a_1(k)\tilde{\mathbf{y}}(k),\\ \bar{\mathbf{x}}(k) &= \frac{1}{\sqrt{\alpha}}\mathbf{K}^H(k-1)\tilde{\mathbf{x}}(k).\end{aligned} \tag{12}$$

In the above relations, $a_1(k)$, $\delta_1(k)$, $\delta_2(k)$ and $\mathbf{r}_x(k)$ are defined by:

$$\begin{aligned}a_1(k) &= \tilde{\mathbf{y}}^{\mathrm{H}}(k)\tilde{\mathbf{x}}(k),\\ \delta_1(k) &= \frac{1}{\|\tilde{\mathbf{x}}(k)\|^2}\Big(\frac{1}{\sqrt{1+\|\tilde{\mathbf{x}}(k)\|^2}} - 1\Big),\\ \delta_2(k) &= |\delta_1(k)|^2\left(\beta\tilde{\mathbf{x}}^{\mathrm{H}}(k)\mathbf{r}_x(k) + |a_1(k)|^2\right),\\ \mathbf{r}_x(k) &= \mathbf{R}(k-1)\tilde{\mathbf{x}}(k).\end{aligned} \tag{13}$$

After updating $\mathbf{R}(k)$ and $\mathbf{K}(k)$, $\mathbf{G}(k)$ is obtained through the recursion:

$$\begin{aligned}\mathbf{G}(k) =&\mathbf{R}^H(k)\mathbf{\Psi}\\ =&\frac{1}{\alpha}\Big[\beta\mathbf{R}(k-1) + \tilde{\mathbf{y}}(k)\tilde{\mathbf{y}}^H(k) + \tilde{\mathbf{x}}(k)\mathbf{c}^H(k)\\ &+ \delta_1^H(k)\mathbf{h}(k)\tilde{\mathbf{x}}^H(k)\Big]^H\mathbf{\Psi}\\ =&\frac{1}{\alpha}\Big[\beta\mathbf{G}(k-1) + \tilde{\mathbf{y}}(k)\mathbf{y}_o^H(k) + \mathbf{c}(k)\mathbf{x}_o^H(k)\\ &+ \delta_1(k)\tilde{\mathbf{x}}(k)\mathbf{h}_o^H(k)\Big],\end{aligned} \tag{14}$$

where

$$\begin{aligned}\mathbf{y}_o(k) &= \mathbf{\Psi}^H\tilde{\mathbf{y}}(k),\\ \mathbf{x}_o(k) &= \mathbf{\Psi}^H\tilde{\mathbf{x}}(k),\\ \mathbf{h}_o(k) &= \mathbf{\Psi}^H\mathbf{h}(k).\end{aligned}$$

We note that $\mathbf{\Psi}$ is generated before the streaming data $\mathbf{x}(k)$ and $\mathbf{y}(k)$ is fed and it is kept as a constant, which implies $\mathbf{\Psi}(0) = \cdots = \mathbf{\Psi}(k)$. Therefore, the index k is omitted for the ease of exposition. Accordingly, $\mathbf{H}(k)$ is obtained through the recursion:

$$\begin{aligned}\mathbf{H}(k) =&\mathbf{R}(k)\mathbf{G}(k)\\ =&\frac{1}{\alpha^2}\Big[\beta^2\mathbf{H}(k-1) + \mathbf{S}_1(k) + \mathbf{S}_2(k) + \mathbf{S}_3(k) + \mathbf{S}_4(k)\Big].\end{aligned} \tag{15}$$

The terms $\{\mathbf{S}_i(k)\}_{i=1}^4$ are given by:

$$\begin{aligned}\mathbf{S}_1(k) &= \beta\mathbf{r}_y(k)\mathbf{y}_o^{\mathrm{H}}(k) + \beta\mathbf{r}_c(k)\mathbf{x}_o^{\mathrm{H}}(k) + \beta\delta_1(k)\mathbf{r}_x(k)\mathbf{h}_o^{\mathrm{H}}(k),\\ \mathbf{S}_2(k) &= \tilde{\mathbf{y}}(k)\big[\beta\mathbf{y}_h^{\mathrm{H}}(k) + a_2(k)\mathbf{y}_o^{\mathrm{H}}(k)\\ &\quad + a_3^*(k)\mathbf{x}_o^{\mathrm{H}}(k) + \delta_1(k)a_1(k)\mathbf{h}_o^{\mathrm{H}}(k)\big],\\ \mathbf{S}_3(k) &= \tilde{\mathbf{x}}(k)\big[\beta\mathbf{c}_h^{\mathrm{H}}(k) + a_3(k)\mathbf{y}_o^{\mathrm{H}}(k)\\ &\quad + a_4(k)\mathbf{x}_o^{\mathrm{H}}(k) + \delta_1(k)a_5(k)\mathbf{h}_o^{\mathrm{H}}(k)\big],\\ \mathbf{S}_4(k) &= \mathbf{h}(k)\big[\delta_1(k)\beta\mathbf{x}_h^{\mathrm{H}}(k) + \delta_1(k)a_1^*(k)\mathbf{y}_o^{\mathrm{H}}(k)\\ &\quad + \delta_1(k)a_5^*(k)\mathbf{x}_o^{\mathrm{H}}(k) + |\delta_1(k)|^2 a_6(k)\mathbf{h}_o^{\mathrm{H}}(k)\big].\end{aligned} \tag{16}$$

where

$$\begin{aligned}\mathbf{r}_y(k) &= \mathbf{R}(k-1)\tilde{\mathbf{y}}(k),\\ \mathbf{r}_c(k) &= \mathbf{R}(k-1)\mathbf{c}(k),\\ \mathbf{y}_h(k) &= \mathbf{G}^{\mathrm{H}}(k-1)\tilde{\mathbf{y}}(k),\\ a_2(k) &= \tilde{\mathbf{y}}^{\mathrm{H}}(k)\tilde{\mathbf{y}}(k),\\ a_3(k) &= \mathbf{c}^{\mathrm{H}}(k)\tilde{\mathbf{y}}(k),\\ \mathbf{c}_h(k) &= \mathbf{G}^{\mathrm{H}}(k-1)\mathbf{c}(k),\\ a_4(k) &= \mathbf{c}^{\mathrm{H}}(k)\mathbf{c}(k),\\ a_5(k) &= \mathbf{c}^{\mathrm{H}}(k)\tilde{\mathbf{x}}(k),\\ \mathbf{x}_h(k) &= \mathbf{G}^{\mathrm{H}}(k-1)\tilde{\mathbf{x}}(k),\\ a_6(k) &= \mathbf{x}^{\mathrm{H}}(k)\tilde{\mathbf{x}}(k).\end{aligned} \tag{17}$$

Next, we orthonormalize the columns of $\mathbf{H}(k)$ (15), obtaining $\mathbf{Q}(k)$:

$$\mathbf{Q}(k) = \mathtt{orth}_{\mathtt{r}}(\mathbf{H}(k)), \tag{18}$$

which gives an approximate basis for the range of $\mathbf{R}(k)$. Then, to lower computational workload, instead of computing $\mathbf{T}(k)$, we make use of the formula in (7) to compute $\widetilde{\mathbf{T}}(k)$:

$$\widetilde{\mathbf{T}}(k) = [\mathbf{\Psi}^H\mathbf{Q}(k)]^\dagger[\mathbf{G}^H(k)\mathbf{Q}(k)].$$

By performing the EVD on $\widetilde{\mathbf{T}}(k)$, we obtain the eigenpair $(\widetilde{\mathbf{\Lambda}}(k), \widetilde{\mathbf{V}}(k))$. Accordingly, an approximation to the r leading generalized eigenvectors of the matrix pencil $(\mathbf{R}_y, \mathbf{R}_x)$ is obtained by:

$$\widetilde{\mathbf{W}}(k) = \mathbf{K}(k)\mathbf{Q}(k)\widetilde{\mathbf{V}}(k). \tag{19}$$

Computational Cost of Algorithm 1. The steps described, due to the fact that they share the same observed vectors in time index k as well as some common variables which have been known through the computation in time index $k-1$, (i) are computationally less expensive compared with matrix-matrix multiplications and matrix square-root inverse, especially for very large N, and (ii) can be executed in parallel. The main steps involve computations of (11)-(17) in each iteration. Steps (18)-(19) are not necessary at each iteration. TABLE I summarizes the dominant cost of Algorithm 1

This article has been accepted for publication in IEEE Transactions on Vehicular Technology. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVT.2023.3243244

TABLE I: Dominant cost of Algorithm 1

Equations	Complexity order
(11)	$10N^2 + \mathcal{O}(N)$
(12)	$3N^2 + \mathcal{O}(N)$
(13)	$N^2 + \mathcal{O}(N)$
(14) - (16)	$2N^2 + \mathcal{O}(Nd)$
(18) - (19)	$N^2r + \mathcal{O}(Nr^2)$
overall	$N^2r + 16N^2 + \mathcal{O}(Nr^2) + \mathcal{O}(Nd)$

B. Extracting r-dominant generalized eigenvectors by tracking the random structured subspace of $\mathbf{R}_x^{-1}(k)\mathbf{R}_y(k)$

Let $\mathbf{P}(k) = \mathbf{Q}_x(k)\mathbf{R}_y(k)$, where $\mathbf{Q}_x(k) = \mathbf{R}_x^{-1}(k)$. The algorithm presented here (hereafter called Algorithm 2) first recursively updates $\mathbf{P}(k)$, then applies it as the input matrix for ARP-EVD to extract the generalized eigenvectors. In doing so, by applying the SM-formula (Sherman-Morrison-formula) [42], we obtain a recursion for $\mathbf{Q}_x(k)$:

$$\mathbf{Q}_x(k) = \mathbf{R}_x^{-1}(k) = \frac{1}{\alpha}\Big[\mathbf{Q}_x(k-1) - \frac{\mathbf{q}_x(k)\mathbf{q}_x^H(k)}{\alpha + \mathbf{x}^H(k)\mathbf{q}_x(k)}\Big], \tag{20}$$

where

$$\mathbf{q}_x(k) = \mathbf{Q}_x(k-1)\mathbf{x}(k).$$

As a result, $\mathbf{P}(k)$ is obtained by:

$$\begin{aligned}\mathbf{P}(k) =& \mathbf{Q}_x(k)\mathbf{R}_y(k)\\ =& \frac{1}{\alpha}\Big[\beta\mathbf{P}(k-1) + \mathbf{q}_y(k)\mathbf{y}^H(k) - \mathbf{q}_x(k)\mathbf{z}^H(k)\Big],\end{aligned} \tag{21}$$

where

$$\mathbf{q}_y(k) = \mathbf{Q}_x(k-1)\mathbf{y}(k),$$

$$\mathbf{z}(k) = \left(\frac{\beta\mathbf{x}^H(k)\mathbf{P}(k-1)}{\alpha + \mathbf{x}^H(k)\mathbf{q}_x(k)} + \frac{\mathbf{q}_x^H(k)\mathbf{y}(k)\mathbf{y}^H(k)}{\alpha + \mathbf{x}^H(k)\mathbf{q}_x(k)}\right)^H.$$

After expressing $\mathbf{P}(k)$ through a recursion, we now update $\mathbf{G}(k)$ as follows:

$$\begin{aligned}\mathbf{G}(k) &= \mathbf{P}^H(k)\mathbf{\Psi}\\ &= \frac{1}{\alpha}\Big[\beta\mathbf{G}(k-1) + \mathbf{y}(k)\mathbf{m}_y^H(k) - \mathbf{z}(k)\mathbf{m}_x^H(k)\Big],\end{aligned} \tag{22}$$

where

$$\mathbf{m}_y(k) = \mathbf{\Psi}^H\mathbf{q}_y(k),$$

$$\mathbf{m}_x(k) = \mathbf{\Psi}^H\mathbf{q}_x(k).$$

We then obtain $\mathbf{H}(k)$ through the following equation:

$$\begin{aligned}\mathbf{H}(k) &= \mathbf{P}(k)\mathbf{G}(k)\\ &= \frac{1}{\alpha^2}\Big[\beta^2\mathbf{H}(k-1) + \mathbf{J}_1(k) + \mathbf{J}_2(k) + \mathbf{J}_3(k)\Big],\end{aligned} \tag{23}$$

The terms $\{\mathbf{J}_i(k)\}_{i=1}^3$ in (23) are given by:

$$\begin{aligned}\mathbf{J}_1(k) &= \beta\mathbf{d}_y(k)\mathbf{m}_y^H(k) - \beta\mathbf{d}_z(k)\mathbf{m}_x^H(k),\\ \mathbf{J}_2(k) &= \mathbf{q}_y(k)\big(\beta\mathbf{n}_y^H(k) + b_1(k)\mathbf{m}_y^H(k) - b_2(k)\mathbf{m}_x^H(k)\big),\\ \mathbf{J}_3(k) &= \mathbf{q}_x(k)\big(b_3(k)\mathbf{m}_x^H(k) - \beta\mathbf{n}_z^H(k) - b_2^*(k)\mathbf{m}_y^H(k)\big).\end{aligned} \tag{24}$$

where

$$\begin{aligned}\mathbf{d}_y(k) &= \mathbf{P}(k-1)\mathbf{y}(k),\\ \mathbf{d}_z(k) &= \mathbf{P}(k-1)\mathbf{z}(k),\\ \mathbf{n}_y(k) &= \mathbf{G}^H(k-1)\mathbf{y}(k),\\ b_1(k) &= \mathbf{y}^H(k)\mathbf{y}(k),\\ b_2(k) &= \mathbf{y}^H(k)\mathbf{z}(k),\\ \mathbf{n}_z(k) &= \mathbf{G}^H(k-1)\mathbf{z}(k),\\ b_3(k) &= \mathbf{z}^H(k)\mathbf{z}(k).\end{aligned}$$

We now orthonormalize the columns of $\mathbf{H}(k)$ (23), obtaining $\mathbf{Q}(k) = \mathtt{orth}_{\mathtt{r}}(\mathbf{H}(k))$, which provides approximation to the range of $\mathbf{P}(k)$. Following the procedure described in Algorithm 1, we form $\widetilde{\mathbf{T}}(k)$ and compute the eigenpair $(\widetilde{\mathbf{\Lambda}}(k), \widetilde{\mathbf{V}}(k))$. The r leading generalized eigenvectors of the matrix pencil $(\mathbf{R}_y, \mathbf{R}_x)$ are then approximated by:

$$\widetilde{\mathbf{W}}(k) = \mathbf{Q}(k)\widetilde{\mathbf{V}}(k). \tag{25}$$

Computational Cost of Algorithm 2. TABLE II summarizes the dominant cost of Algorithm 2. The flop count of Algorithm 2 satisfies $13N^2 + \mathcal{O}(Nr^2) + \mathcal{O}(Nd)$ flops.

TABLE II: Dominant cost of Algorithm 2

Equations	Complexity order
(20) - (21)	$11N^2 + \mathcal{O}(N)$
(22)	$\mathcal{O}(Nd)$
(23)-(24)	$2N^2 + \mathcal{O}(Nd)$
(25)	$\mathcal{O}(Nr^2) + \mathcal{O}(r^3)$
overall	$13N^2 + \mathcal{O}(Nr^2) + \mathcal{O}(Nd)$

IV. Theoretical analysis

In this section, we characterize the performance of the proposed algorithms by providing an error bound which shows the accuracy of the computed basis.

Online updates of sample covariance matrices by (8) and (9) lead to unbiased estimates of

$$\begin{aligned}\mathbf{R}'_x &= \frac{1}{1-\alpha}\mathbf{R}_x,\\ \mathbf{R}'_y &= \frac{1}{1-\beta}\mathbf{R}_y.\end{aligned}$$

Assume that the ℓ_2-norm errors of sample covariance matrix estimates at instant k are bounded, i.e.,

$$E\left\{\left\|\mathbf{R}_x(k) - \mathbf{R}'_x\right\|_2\right\} \le \varepsilon_x(k),$$

and

$$E\left\{\left\|\mathbf{R}_y(k) - \mathbf{R}'_y\right\|_2\right\} \le \varepsilon_y(k).$$

The error upper bounds of $\mathbf{R}_x^{-1}(k)\mathbf{R}_y(k)$ and $\mathbf{R}_x^{-\frac{1}{2}}(k)\mathbf{R}_y(k)(\mathbf{R}_x^{-\frac{1}{2}}(k))^H$ can then be denoted by:

$$E\left\{\left\|\mathbf{R}_x^{-1}(k)\mathbf{R}_y(k) - \mathbf{R}'_x{}^{-1}\mathbf{R}'_y\right\|_2^2\right\} \le \varepsilon_1(k),$$

and

$$E\left\{\left\|\mathbf{R}_x^{-\frac{1}{2}}(k)\mathbf{R}_y(k)\mathbf{R}_x^{-\frac{1}{2}}(k) - \mathbf{R}'_x{}^{-\frac{1}{2}}\mathbf{R}'_y(\mathbf{R}'_x{}^{-\frac{1}{2}})^H\right\|_2^2\right\} \le \varepsilon_2(k).$$

where $\varepsilon_1(k)$ and $\varepsilon_2(k)$ are two constants depending on $\varepsilon_x(k)$, $\varepsilon_y(k)$, and $\mathrm{cond}(\mathbf{R}_x)$ with $\mathrm{cond}(\cdot)$ denoting the condition

number of a matrix. Note that these upper bounds only rely on the statistic properties of random variables $\mathbf{x}(k)$ and $\mathbf{y}(k)$, and they are independent of the proposed algorithms.

The results of proposed online algorithms are equivalent to those from applying APR-EVD to $\mathbf{R}_x^{-1}(k)\mathbf{R}_x(k)$ and $\mathbf{R}_x^{-\frac{1}{2}}(k)\mathbf{R}_y(k)\ \mathbf{R}_x^{-\frac{1}{2}}(k)$ at each instant k. Having known that the accuracy of the covariance estimation is independent of the algorithms, establishing an approximation error bound of APR-EVD plays the crucial role in our analysis. In what follows, we bound from above the error of the low-rank approximation constructed by APR-EVD.

For simplicity, we consider the case where the input matrix $\mathbf{A}$ upon which APR-EVD operates is real-valued. An extension to the complex-valued matrix is straightforward. To bound the error of APR-EVD, we first define the SVD [42], which factors $\mathbf{A}$ as:

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{W}^T = \begin{bmatrix}\mathbf{U}_1 & \mathbf{U}_2\end{bmatrix}\begin{bmatrix}\mathbf{\Sigma}_1 & \mathbf{0}\\ \mathbf{0} & \mathbf{\Sigma}_2\end{bmatrix}\begin{bmatrix}\mathbf{W}_1^T\\ \mathbf{W}_2^T\end{bmatrix}, \tag{26}$$

where orthogonal $\mathbf{U}_1 \in \mathbb{R}^{N\times r}$ and $\mathbf{U}_2 \in \mathbb{R}^{N\times(N-r)}$ are the left singular vectors, $\mathbf{\Sigma}_1 \triangleq \text{diag}(\sigma_1, ..., \sigma_r)$ and $\mathbf{\Sigma}_2 \triangleq \text{diag}(\sigma_{r+1}, ..., \sigma_N)$ are diagonal of order r and $N-r$, respectively, whose entries σ_is are the singular values, and orthogonal $\mathbf{W}_1 \in \mathbb{R}^{N\times r}$ and $\mathbf{W}_2 \in \mathbb{R}^{N\times(N-r)}$ are the right singular vectors of $\mathbf{A}$. For the rank-r approximation of $\mathbf{A}$, the least error is achieved by the SVD such that:

$$\|\mathbf{A} - \mathbf{U}_1\mathbf{U}_1^T\mathbf{A}\|_2 = \|\mathbf{\Sigma}_2\|_2,$$

To derive an error bound in relation to the optimal one, we consider $r+p \le d < N$, where $p \ge 0$ is called oversampling parameter and, considering the interaction of left singular vectors of $\mathbf{A}$ with $\mathbf{\Psi}$ (2), define

$$\mathbf{U}^T\mathbf{\Psi} \triangleq [\widetilde{\mathbf{\Psi}}_1^T \quad \widetilde{\mathbf{\Psi}}_2^T]^T,$$

where $\widetilde{\mathbf{\Psi}}_1$ and $\widetilde{\mathbf{\Psi}}_2$ have $d-p$ and $N-d+p$ rows, respectively. We now put forth our theorem.

Theorem 1: Let a matrix $\mathbf{A}$ be square with order N and numerical rank r. Let $\mathbf{U}_r$ be computed by (6) with APR-EVD. Then,

$$\|\mathbf{A} - \mathbf{U}_r\mathbf{U}_r^T\mathbf{A}\|_2 \le \|\mathbf{\Sigma}_2\|_2 + \sqrt{\frac{C_1^2\|\widetilde{\mathbf{\Psi}}_2\|_2^2\|\widetilde{\mathbf{\Psi}}_1^\dagger\|_2^2}{1 + C_2^2\|\widetilde{\mathbf{\Psi}}_2\|_2^2\|\widetilde{\mathbf{\Psi}}_1^\dagger\|_2^2}}, \tag{27}$$

where $C_1 = \sqrt{r}\frac{\sigma_{d-p+1}^2}{\sigma_r}$, and $C_2 = \frac{\sigma_{d-p+1}^2}{\sigma_1\sigma_r}$.

Proof: Let $\mathbf{P}_{U_r} \triangleq \mathbf{U}_r\mathbf{U}_r^T = \mathbf{Q}\mathbf{V}\mathbf{V}^T\mathbf{Q}^T = \mathbf{Q}\mathbf{Q}^T \triangleq \mathbf{P}_Q$. This relation follows because $\mathbf{V}$ is an orthonormal matrix of order r. We therefore have

$$\|(\mathbf{I} - \mathbf{P}_{U_r})\mathbf{A}\|_2 = \|(\mathbf{I} - \mathbf{P}_Q)\mathbf{A}\|_2. \tag{28}$$

It now suffices to bound the right-hand side of (28). To do so, writing $\mathbf{A} = \mathbf{A}_1 + \mathbf{A}_2$, where $\mathbf{A}_1 = \mathbf{U}_1\mathbf{\Sigma}_1\mathbf{W}_1^T$ and $\mathbf{A}_2 = \mathbf{U}_2\mathbf{\Sigma}_2\mathbf{W}_2^T$, and using the triangle inequality, we will have:

$$\|(\mathbf{I} - \mathbf{P}_Q)\mathbf{A}\|_2 \le \|(\mathbf{I} - \mathbf{P}_Q)\mathbf{A}_1\|_2 + \|(\mathbf{I} - \mathbf{P}_Q)\mathbf{A}_2\|_2. \tag{29}$$

For the first term on the right-hand side of (29), by following the procedure described in the proofs of Theorems 4 and 5 of [32], we have

$$\|(\mathbf{I} - \mathbf{P}_Q)\mathbf{A}_1\|_2 \le \sqrt{\frac{C_1^2\|\widetilde{\mathbf{\Psi}}_2\|_2^2\|\widetilde{\mathbf{\Psi}}_1^\dagger\|_2^2}{1 + C_2^2\|\widetilde{\mathbf{\Psi}}_2\|_2^2\|\widetilde{\mathbf{\Psi}}_1^\dagger\|_2^2}}, \tag{30}$$

where $C_1 = \sqrt{r}\frac{\sigma_{d-p+1}^2}{\sigma_r}$, and $C_2 = \frac{\sigma_{d-p+1}^2}{\sigma_1\sigma_r}$. For the second term on the right-hand side of (29), we will have

$$\|(\mathbf{I} - \mathbf{P}_Q)\mathbf{A}_2\|_2 \le \|\mathbf{I} - \mathbf{P}_Q\|_2\|\mathbf{A}_2\|_2 \le \|\mathbf{\Sigma}_2\|_2. \tag{31}$$

Substituting the results in (30) and (31) into (29), the theorem follows. ∎

Theorem 2: Let matrix $\mathbf{A}$ have an SVD defined in (26), $r+p \le d < N$, where $p \ge 0$, and $\mathbf{U}_r$ be computed by APR-EVD. Let $0 < \Xi \ll 1$, and define

$$C_\Xi = \frac{e\sqrt{d}}{p+1}\Big(\frac{2}{\Xi}\Big)^{\frac{1}{p+1}}\Big(\sqrt{n-d+p} + \sqrt{d} + \sqrt{2\log\frac{2}{\Xi}}\Big).$$

Then, with probability at least $1 - C_\Xi$, we have

$$\|(\mathbf{I} - \mathbf{P}_{U_r})\mathbf{A}\|_2 \le \|\mathbf{\Sigma}_2\|_2 + \sqrt{\frac{r\sigma_{d-p+1}^4}{\sigma_r^2}C_\Xi^2}.$$

Proof: According to [35, Theorem 5.8], for standard Gaussian matrices $\widetilde{\mathbf{\Psi}}_1$ and $\widetilde{\mathbf{\Psi}}_2$, we have

$$\mathbb{P}\Big\{\|\widetilde{\mathbf{\Psi}}_2\|_2\|\widetilde{\mathbf{\Psi}}_1^\dagger\|_2 \ge C_\Xi\Big\} \le \Xi. \tag{32}$$

Moreover, (27) is simplified to

$$\|(\mathbf{I} - \mathbf{P}_{U_r})\mathbf{A}\|_2 \le \|\mathbf{\Sigma}_2\|_2 + \sqrt{C_1^2\|\widetilde{\mathbf{\Psi}}_2\|_2^2\|\widetilde{\mathbf{\Psi}}_1^\dagger\|_2^2}. \tag{33}$$

By substituting the result (32) into (33), the theorem follows. ∎

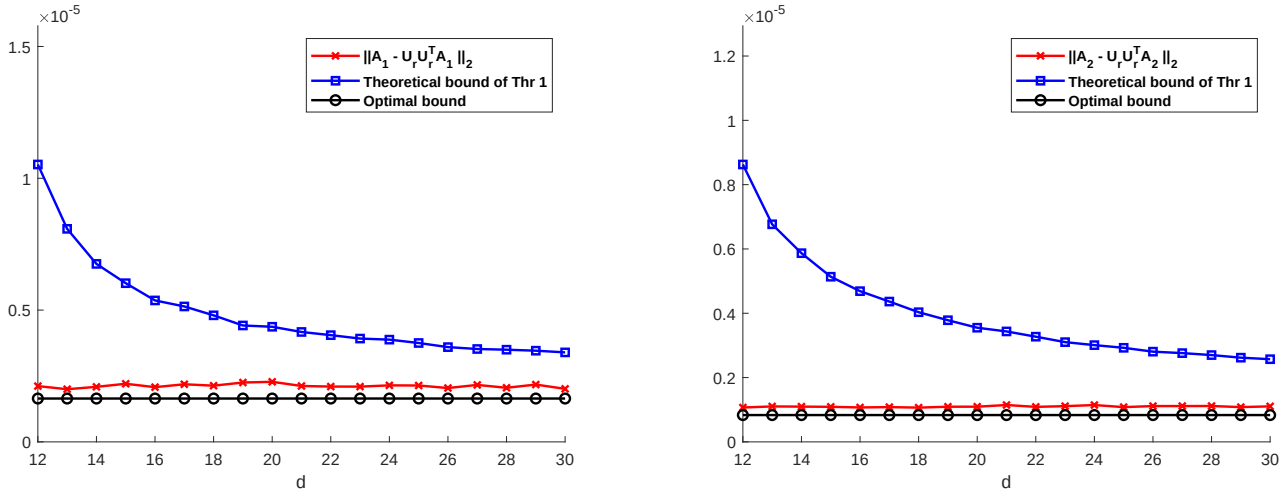
V. Numerical Simulations

This section demonstrates the experimental results of our numerical tests conducted to verify our theoretical findings as well as the performance of our proposed algorithms for tracking the generalized eigenvectors of matrix pencils. The first example involves the evaluation of the error bound of APR-EVD algorithm. Examples 2 to 4 consider the tracking of generalized eigenvectors from streaming data with stationary or time-varying statistics, and the last example considers the application of online hyperspectral image denoising.

In the numerical examples, we compare the results of Algorithms 1 and 2 with those of the following algorithms:

- Power Iteration-based (PI-based) [14]: a fast tracking algorithm based on the power iteration.
- PAST-based: these algorithms are extensions of PAST and PASTd [12]. For examples 2 and 3, we use the sequential version [12, Algorithm 2]. For example 4, we use [12, Algorithm 1], which is used to extract largest generalized eigenvector.
- R-GEVE [13]: this algorithm is based on the reduced-rank technique.

This article has been accepted for publication in IEEE Transactions on Vehicular Technology. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVT.2023.3243244



(a) Comparison of the ℓ_2-norm approximation error for the Hermitian input matrix $\mathbf{A}_1$.

(b) Comparison of the ℓ_2-norm approximation error for the non-Hermitian input matrix $\mathbf{A}_2$.

Fig. 1: Empirical evaluation of the error bound for the APR-EVD algorithm.

- Gradient-based [15, Algorithm 3]: with negative step size to track the principal component and multiple generalized eigenvectors tracking is implemented using orthogonal complement structure.
- GSVD [42]: the batch algorithm for generalized singular value decomposition.

A. Example 1

This subsection empirically investigates via two synthetic matrices the accuracy of APR-EVD in constructing a low-rank approximation, and the tightness of the error bound developed; it compares the ℓ_2-norm approximation error produced by APR-EVD with the theoretical bound presented in (27). In the context of GHEP, the input matrix considered is not an arbitrary square matrix. Instead, it is well structured based on the transformation discussed in Section III. To be precise, the $N \times N$ input matrix is synthesized using two Hermitian matrices $\mathbf{R}_y$ and $\mathbf{R}_x$ satisfying equation (1) and has the form of $\mathbf{R}_x^{-1}\mathbf{R}_y$ and $\mathbf{R}_x^{-1/2}\mathbf{R}_y(\mathbf{R}_x^{-1/2})^H$, as is the case considered in Algorithms 1 and 2. For this purpose, we generate the matrix pencil using the well-known jointly diagonalization [43],

$$\begin{aligned} \mathbf{W}^H\mathbf{R}_y\mathbf{W} &= \mathbf{\Lambda}_{r'}. \\ \mathbf{W}^H\mathbf{R}_x\mathbf{W} &= \mathbf{I}_N. \end{aligned} \tag{34}$$

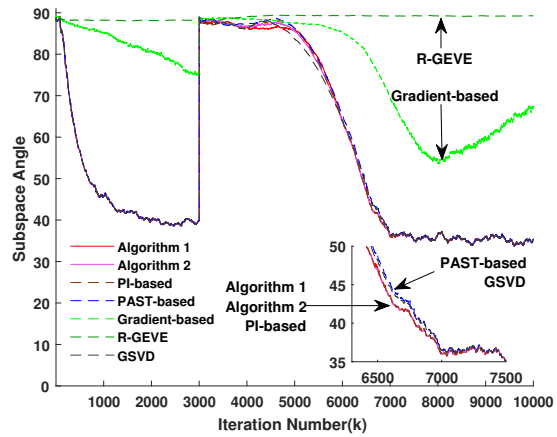
In the above equations, $\mathbf{W}$ is a randomly generated $\mathbf{R}_x$-orthonormal matrix, $\mathbf{R}_x$ is a randomly generated Hermitian positive definite matrix, $\mathbf{\Lambda}_{r'}$ is a diagonal matrix containing r'-dominant generalized eigenvalues that decrease geometrically from 1 to 10^{-5}, and the remaining generalized eigenvalues $\lambda_i = 10^{-1}\lambda_{r'}(i = r'+1, \cdots, N)$. In MATLAB notation, `Ux = randn(N); Rx = Ux'*Ux; smax =1; smin =1e-5; s1 = linspace(smax, smin,N); s1(r+1:N) = 1e-6; W = sqrtm(inv(Rx))*orth(randn(N)); Ry = inv(W)'*diag(s)*inv(W).` This example considers two cases of input matrix: i) Hermitian matrix $\mathbf{A}_1 = \mathbf{R}_x^{-1/2}\mathbf{R}_y(\mathbf{R}_x^{-1/2})^H$, and ii) non-Hermitian matrix $\mathbf{A}_2 = \mathbf{R}_x^{-1}\mathbf{R}_y$. We set $N = 100$ and the practical rank is set to $r' = 10$. We then fix the value $r = r'$ and increase the sampling parameter size d to evaluate the value of $\|\mathbf{A}_i - \mathbf{U}_r\mathbf{U}_r^T\mathbf{A}_i\|_2 \quad (i = 1, 2)$, under the assumption $2 \leq p \leq d - r'$. Figs. 1 and 2 depict the ℓ_2-norm approximation errors $\|\mathbf{A}_1 - \mathbf{U}_r\mathbf{U}_r^T\mathbf{A}_1\|_2$ and $\|\mathbf{A}_2 - \mathbf{U}_r\mathbf{U}_r^T\mathbf{A}_2\|_2$ versus different choices of sampling size d respectively, where the black curve indicates the optimal error bound, i.e., $(r+1)$-th singular value obtained by the SVD of $\mathbf{A}_i$ and the black curve gives the upper bound presented in Theorem 1. Judging from the figures, we conclude that APR-EVD algorithm achieves a highly accurate approximation for both types of input matrices $\mathbf{A}_1$ and $\mathbf{A}_2$ when d is greater than its practical rank r'. Increasing the value of d, however, does not lead to improved approximation performance. Therefore, a moderate choice of $d \geq r' + 2$ is sufficient to yield an accurate approximation basis. We also observe that all the approximation errors are strictly upper-bounded by the theoretical bound under different tested values of d.

B. Example 2

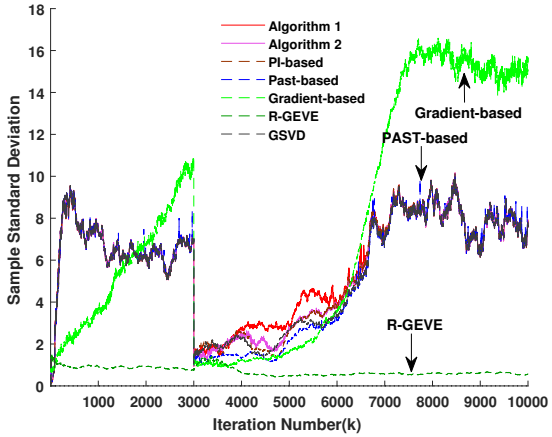
In this example, we simultaneously extract the dominant generalized eigenvectors from two non-stationary real-valued random vector processes $\{\mathbf{y}(k)_{k\in\mathbb{Z}}\}$ and $\{\mathbf{x}(k)_{k\in\mathbb{Z}}\}$. The signal model for generating the random vector processes is synthesized using the MATLAB script described in Example 1. The non-stationary matrix pencil $(\mathbf{R}_y, \mathbf{R}_x)$ can be obtained by varying $\mathbf{W}$ and $\mathbf{R}_x$ over time.

To compare the convergence speed and estimation accuracy, we use the angle between subspaces; it measures the closeness of two subspaces (estimated and exact subspaces) [44], [45]. We also use sample standard deviation of subspace angle to measure numerical stability of considered algorithms, which is computed similar to the criteria defined in (36).

This article has been accepted for publication in IEEE Transactions on Vehicular Technology. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVT.2023.3243244



(a) Angle between subspaces of dominant generalized eigenvectors.



(b) Sample standard deviation of dominant generalized eigenvectors.

Fig. 2: Angle between subspaces and sample standard deviation results of dominant generalized eigenvectors for Example 2.

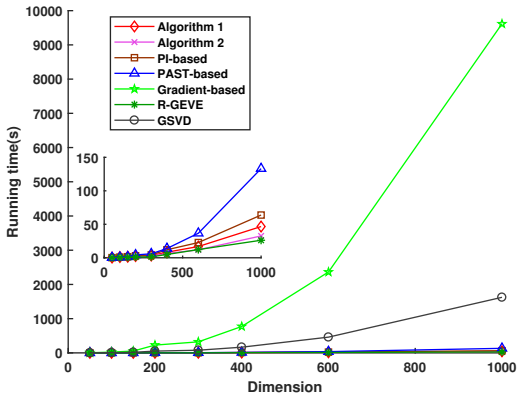


Fig. 3: Runtime versus dimension over 2000 iterations based on the experiment settings in Example 2.

The basic settings of the algorithms considered in this example are set as follows: we set forgetting factors to $\alpha = \beta = 0.998$ as in [15]. Since the explicit knowledge of $(\mathbf{R}_y, \mathbf{R}_x)$ is unknown, the step size of gradient-based algorithm is set to -0.001, which guarantees a considerable convergence speed and numerical stability. For Algorithms 1 and 2, in order to obtain more accurate estimations, we set $r = d = 13$. All parameters of the other algorithms and initial values are the same as in Example 1. For PAST-based algorithm, we set $\beta = 0.92$, $\mathbf{c}(0)_i = 0.1\mathbf{e}_i$, and $d_i(0) = 1$ for $i = 1, 2$ as suggested in [12], where $\mathbf{e}_i$ is the ith column of the $N \times N$ identity matrix. All algorithms are initialized with $\mathbf{R}_x(0) = \mathbf{R}_y(0) = \mathbf{I}_N$, and $\widetilde{\mathbf{w}}_i(0) = \mathbf{e}_i$.

Figs. 2(a) and 2(b) display the subspace angle and sample standard deviation for the estimated r'-dominant generalized eigenvectors, respectively. In terms of tracking the r'-dominant generalized eigenvectors, Fig. 2(a) shows that Algorithm 2, Algorithm 1, PAST-based, PI-based and GSVD exhibit the fastest convergence speed in the initial phase. R-GEVE and gradient-based methods face difficulties to converge. Starting from time instant $k = 3000$, Algorithm 1, Algorithm 2 and PI-based method are able to track the shift of the signal model in the fastest convergence speed. From Fig. 2(b), though R-GEVE has the best numerical stability, this method does not converge. PAST-based method tends to fluctuate in the steady state. In both figures, we observe that Algorithm 1, Algorithm 2 and PI-based method demonstrate similar performance.

C. Runtime comparison

The runtime versus matrix dimension is assessed for the compared algorithms. The simulation is conducted via MATLAB on a PC with a 3.59 GHz AMD Ryzen 5 3600 6-core processor and 16 GB of memory. The results are obtained over 2000 iterations via the same signal model and experimental settings as in example 2.

From Fig. 3, we observe that Algorithm 2 and R-GEVE exhibit the fastest processing speed among all these algorithms as matrix dimension increases. However, the latter encounters difficulties to converge under high-dimensional setting. Algorithm 1 has the second best processing speed among the considered algorithms. Gradient-based and GSVD are computationally prohibitive which limits their practical use for high-dimensional applications.

D. Example 3

In this example, we simultaneously extract the dominant generalized eigenvectors from two real-valued random vector processes $\{\mathbf{y}(k)_{k\in\mathbb{Z}}\}$ and $\{\mathbf{x}(k)_{k\in\mathbb{Z}}\}$. These signals are generated by two sinusoids in additive noise defined in the time domain [15], [16], [46]:

$$y(k) = \sqrt{2}\sin(0.62\pi k + \theta_1) + n_1(k),$$
$$x(k) = \sqrt{2}\sin(0.46\pi k + \theta_2) + \sqrt{2}\sin(0.74\pi k + \theta_3) + n_2(k),$$

where θ_i, for $i = 1, 2, 3$, are the initial phases, which follow uniform probability distributions within $[0, 2\pi]$. $n_1(k)$ and $n_2(k)$ are zero-mean white Gaussian noises with variance

$\sigma_1^2 = \sigma_2^2 = 0.1$. The input vectors $\{\mathbf{y}(k)\}$ and $\{\mathbf{x}(k)\}$ are arranged in blocks of size $N = 8$, that is,

$$\mathbf{y}(k) = [y(k), \cdots, y(k-N+1)]^T,$$
$$\mathbf{x}(k) = [x(k), \cdots, x(k-N+1)]^T,$$

and $k \geq N$. Define the $N \times N$ matrix pencil $\left(\bar{\mathbf{R}}_y, \bar{\mathbf{R}}_x\right)$ with the (m,n) entry $(m,n = 1,\ldots,N)$ of $\bar{\mathbf{R}}_y$ and $\bar{\mathbf{R}}_x$ given by

$$\left[\bar{\mathbf{R}}_y\right]_{mn} = \cos[0.62\pi(n-m)] + \delta_{mn}\sigma_1^2,$$
$$\left[\bar{\mathbf{R}}_x\right]_{mn} = \cos[0.46\pi(n-m)] + \cos[0.74\pi(n-m)] + \delta_{mn}\sigma_2^2.$$

Where δ_{mn} is the Kronecker delta function. When the forgetting factors α and β are set to 1, we have

$$\lim_{k\to\infty} \frac{1}{k}\mathbf{R}_y(k) = \bar{\mathbf{R}}_y,$$
$$\lim_{k\to\infty} \frac{1}{k}\mathbf{R}_x(k) = \bar{\mathbf{R}}_x.$$

The generalized eigenvalues of the matrix pencil $(\bar{\mathbf{R}}_y, \bar{\mathbf{R}}_x)$ are calculated as $\bar{\lambda}_1 = 16.0680$, $\bar{\lambda}_2 = 6.8302$, $\bar{\lambda}_3 = 1.0$, $\bar{\lambda}_4 = 1.0$, $\bar{\lambda}_5 = 0.1592$, $\bar{\lambda}_6 = 0.0708$, $\bar{\lambda}_7 = 0.0254$, and $\bar{\lambda}_8 = 0.0198$. For the first two generalized eigenvalues $\bar{\lambda}_1, \bar{\lambda}_2$, the ratio $\frac{\bar{\lambda}_1+\bar{\lambda}_2}{\sum_{i=1}^{N}\bar{\lambda}_i} = 90.96\%$, showing that the summation of first two generalized eigenvalues $\bar{\lambda}_1$ and $\bar{\lambda}_2$ accounts for a large proportion. Therefore, the first two generalized eigenvectors are dominant.

To compare the convergence speed and estimation accuracy, we use the direction cosine, which measures the similarity between the ith estimated and reference generalized eigenvectors of matrix pencil $\left(\bar{\mathbf{R}}_y, \bar{\mathbf{R}}_x\right)$:

$$\mathrm{DC}_i(k) = \frac{\left|\widetilde{\mathbf{w}}_i^H(k)\bar{\mathbf{w}}_i\right|}{\|\widetilde{\mathbf{w}}_i(k)\|\|\bar{\mathbf{w}}_i\|}, \tag{35}$$

where $\widetilde{\mathbf{w}}_i$ and $\bar{\mathbf{w}}_i$ are the ith estimated and reference generalized eigenvectors, respectively. When $\widetilde{\mathbf{w}}_i$ converges to $\bar{\mathbf{w}}_i$, $\mathrm{DC}_i(k) = 1$. Here the result of (35) is averaged over 100 independent trials. Moreover, to measure the numerical stability of considered algorithms, we use the sample standard deviation (SSD) of the direction cosine defined as:

$$\mathrm{SSD}_i(k) = \sqrt{\frac{1}{L-1}\sum_{j=1}^{L}\left[\mathrm{DC}_{i,j}(k) - \overline{\mathrm{DC}_i}(k)\right]^2}, \tag{36}$$

where $\mathrm{DC}_{i,j}(k)$ is the direction cosine of jth independent trial, where $j = 1, \cdots, L$, of the ith estimated generalized eigenvector, and $\overline{\mathrm{DC}_i}(k)$ is the direction cosine of ith estimated generalized eigenvector averaged over L trials. Here $L = 100$.

The basic settings of the algorithms considered in this example are as follows: for PI-based algorithm, we set $\alpha = \beta = 0.998$ as suggested in [14]. For PAST-based algorithm, we start with $\mu = 0.998$ as suggested in [12, Experiment 1]. For R-GEVE, we set $\beta = 0.998$ as suggested in [13]. For Gradient-based algorithm $\alpha = \beta = 0.998$ as suggested in [15]. For GSVD, we set $\alpha = \beta = 0.998$. Finally, for the proposed Algorithms 1 and 2, we set $\alpha = \beta = 0.998$. By the current setting, all algorithms use the same forgetting factors to estimate the covariance matrices. For Algorithms 1 and 2, in order to obtain more accurate estimations, we set $r = 3$ or $r = 4$ and, accordingly the sample size parameter $d = 5$, as our simulation results show better estimated dominant generalized eigenvectors. Based on smallest and largest generalized eigenvalue of $\left(\bar{\mathbf{R}}_y, \bar{\mathbf{R}}_x\right)$, for gradient-based algorithm, the step size is set to $\eta = -0.0005 \in \left(2/\left(\bar{\lambda}_N - \bar{\lambda}_1\right), 0\right)$ to guarantee a considerable convergence speed and numerical stability. The other settings follow those of example 2.

Figs. 4(a) and 4(b) display the direction cosine for the first two estimated dominant generalized eigenvectors. In terms of tracking the first dominant generalized eigenvector, Fig. 4(a) shows that Algorithm 2 has the fastest convergence speed and the best estimation accuracy. In terms of tracking the second dominant generalized eigenvector, Fig. 4(b) shows that Algorithm 2 also has the fastest convergence speed and a similar estimation accuracy compared to the gradient-based method. However, the gradient-based method has a much slower convergence speed. In both figures, we observe that Algorithm 1 shows similar performance as the PI-based method, however it is computationally more efficient. Algorithm 1 has the lowest computational complexity among the algorithms. Figs. 4(c) and 4(d) display the sample standard deviation for the first two estimated dominant generalized eigenvectors. In both figures, we observe that Algorithm 2 establishes the best numerical stability.

E. Example 4

In this example, the input vectors $\{\mathbf{y}(k)\}$ and $\{\mathbf{x}(k)\}$ are complex-valued vectors defined in the spatial domain, which represent a linear antenna array with multiple fading paths [9]:

$$\mathbf{y}(k) = [y_1(k), \ldots, y_N(k-N+1)]^H,$$
$$\mathbf{x}(k) = [x_1(k), \ldots, x_N(k-N+1)]^H,$$

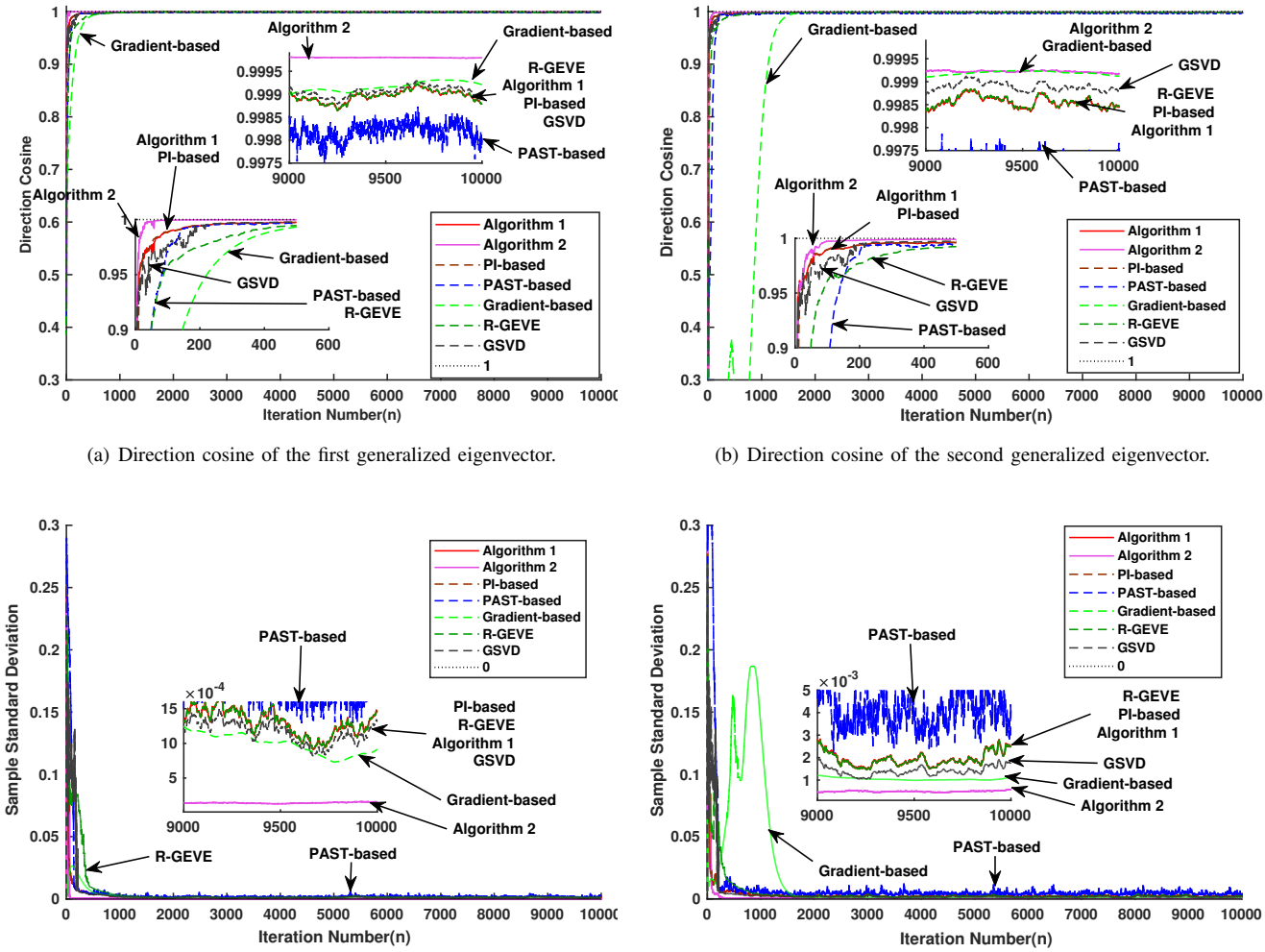
with $k \geq N$, where $y_i(k)$ and $x_i(k)$ for $i = 1, ..., N$ stand for the complex-valued signal impinging the ith element of antenna array, with N being the number of array elements. This model considers the case where an antenna array receives a multipath signal plus a multipath interference. In this scenario, in order to maximize the output signal power and minimize the output interference-plus-noise power, the optimal array weight is found by searching the largest generalized eigenvector of matrix pencil $(\bar{\mathbf{R}}_y, \bar{\mathbf{R}}_x)$, where the entries (m,n) for $m,n = 1, ..., N$ of $(\bar{\mathbf{R}}_y, \bar{\mathbf{R}}_x)$ are given by

$$\left[\overline{\mathbf{R}}_y\right]_{mn} = \frac{1}{I}\sum_{i=0}^{I-1} e^{j\left(\phi_{m,i}^{(1)} - \phi_{n,i}^{(1)}\right)}$$

$$\left[\overline{\mathbf{R}}_x\right]_{mn} = \delta_{mn}\sigma^2 + \frac{1}{I}\sum_{i=0}^{I-1} e^{j\left(\phi_{m,i}^{(2)} - \phi_{n,i}^{(2)}\right)}.$$

In this example, we consider an antenna array with $N = 8$ elements. σ^2 is the variance of the complex Gaussian white noise, $\phi_{m,i}^{(q)} = m\pi\sin\left[\theta_q - \Delta_q/2 + i\Delta_q/(I-1)\right] (q = 1, 2)$ represent the phases of the multipath signals for a given nominal angle-of arrival θ_q, and I is the number of paths for each signal. The signal and interference scatter are uniformly distributed over sectors (angles-of-arrival) of extent Δ_q. We evaluate the tracking performance of the compared algorithms under SNR = 20 dB.

This article has been accepted for publication in IEEE Transactions on Vehicular Technology. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVT.2023.3243244



(a) Direction cosine of the first generalized eigenvector.

(b) Direction cosine of the second generalized eigenvector.

(c) Sample standard deviation of the first generalized eigenvector.

(d) Sample standard deviation of the second generalized eigenvector.

Fig. 4: Direction cosine and sample standard deviation results for Example 3.

We give the following settings for generating the signal, $I = 12, \theta_1 = 5, \Delta_1 = 30, \theta_2 = 25, \Delta_2 = 30$. Since the power of signal has been normalized, we simply adjust noise power to adjust SNR [9], [46]. Here, we set $\sigma^2 = 0.01$ to guarantee SNR = 20 dB. With the setting, the generalized eigenvalues of $(\bar{\mathbf{R}}_y, \bar{\mathbf{R}}_x)$ are calculated as $\bar{\lambda}_1 = 187.9995$, $\bar{\lambda}_2 = 14.2617$, $\bar{\lambda}_3 = 1.3983$, $\bar{\lambda}_4 = 0.6141$, $\bar{\lambda}_5 = 0.0447$, $\bar{\lambda}_6 = 0.0015$, $\bar{\lambda}_7 = 0.0000$, $\bar{\lambda}_8 = 0.0000$.

We are interested in extracting the largest eigenvectors, that is, $r = 1$. We can set $d = r+1$, with 1 being an oversampling parameter. However, our simulation results showed that by searching the first two generalized eigenvectors, i.e., by setting $d = r = 2$ and an oversampling parameter of zero, with the same computational costs, Algorithms 1 and 2 yield slightly better results. For the gradient-based method, the step size is set to $\eta = -0.0002 \in \left(2/\left(\bar{\lambda}_N - \bar{\lambda}_1\right), 0\right)$. The settings of other algorithms follow those of example 2.

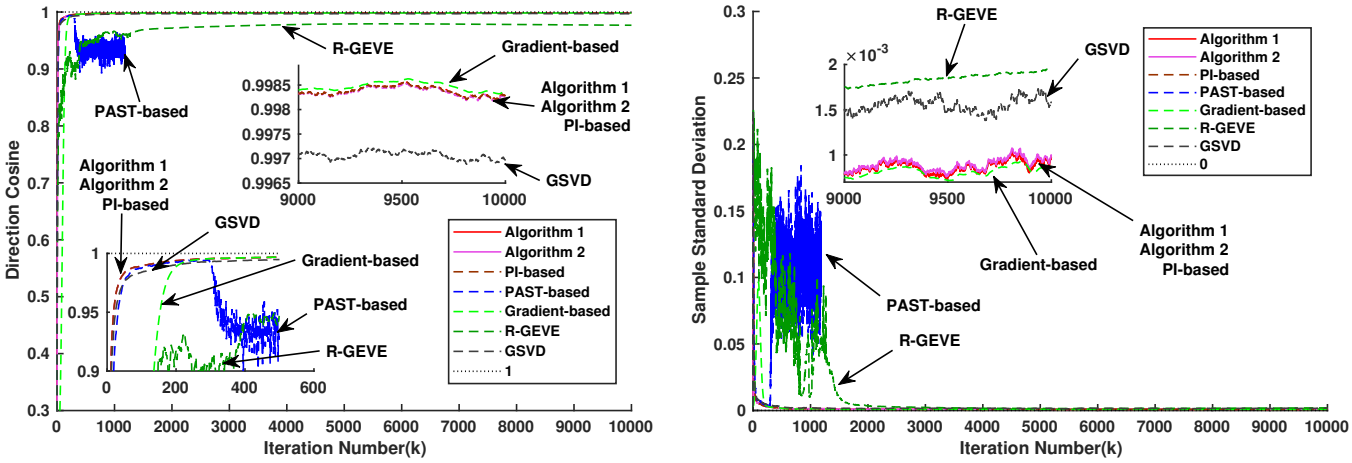
Fig. 5 shows the direction cosine and sample standard deviation for the principal generalized eigenvector. It is observed that Algorithms 1 and 2, gradient-based and PI-based methods show similar performance in terms of estimation accuracy as well as numerical stability. While, the gradient-based method has a slow convergence rate due to a small learning rate, which is required for the algorithm to produce highly accurate results. In addition, Algorithms 1 and 2, PI-based method and the batch GSVD exhibit the fastest convergence speed. However, the latter shows the worse performance in estimation accuracy among these four algorithms. It is further observed that the PASTd-based R-GEVE faces difficulties in convergence, and the extracted principal eigenvector diverges during a specific time instant.

F. Computational complexity comparison

In this subsection, we detail the computational complexity of the algorithms considered in this work.

PI-based [14]: PI-based method is comprised of a 8-steps process. The first four steps are to compute the matrix $\mathbf{R}(k)$, namely $\mathbf{K}(k)\mathbf{R}_y(k)\mathbf{K}^H(k)$. These steps takes $12N^2 + \mathcal{O}(N)$ floating-point operations (flops). Step 5 performs power iteration, with a cost of $N^2 r$ flops. Step 6 reorthonormalizes the subspace derived from Step 5 using QR factorization, which requires $\mathcal{O}(Nr^2)$ flops. Step 7 takes $3N^2 + \mathcal{O}(N)$ flops to

This article has been accepted for publication in IEEE Transactions on Vehicular Technology. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVT.2023.3243244



(a) Direction cosine of the first generalized eigenvector

(b) Sample standard deviation of the first generalized eigenvector

Fig. 5: Direction cosine and sample standard deviation results of the first generalized eigenvector for Example 4.

update matrix $\mathbf{K}(k)$. The last step constrains the extracted generalized eigenvectors to satisfy $\mathbf{w}^H(k)\mathbf{R}_x(k)\mathbf{w}(k) = 1$, which requires N^2r flops. PI-based method costs $2N^2r + 15N^2 + \mathcal{O}(Nr^2)$ flops.

PAST-based [12]: In our paper, the sequential version of PAST-based method is used for comparison. This technique is based on the deflation procedure, by projecting the current searching subspace onto a new subspace that is orthogonal to the subspace spanned by extracted generalized eigenvectors. This method starts by recursively updating the inverse matrix of $\mathbf{R}_x$. This step requires $4N^2 + \mathcal{O}(N)$ flops. The subsequent stage is the deflation procedure. In each loop, updating the generalized eigenvector $\mathbf{w}_i$ and some necessary variables requires $2N^2 + \mathcal{O}(N)$ flops. Hence, $2N^2r + 4N^2 + \mathcal{O}(Nr)$ flops is indispensable for PAST-based method to extract r-dominant generalized eigenvectors.

R-GEVE [13]: This method proceeds in the reduced-rank subspace. Computing the covariance matrix $\mathbf{R}_x(k)$ and $\mathbf{R}_y(k)$ costs $6N^2 + \mathcal{O}(N)$ flops. Updating the other necessary variables requires $2N^2 + \mathcal{O}(Nr^2) + \mathcal{O}(Nr)$ flops. Hence, the dominant cost of R-GEVE is $8N^2 + \mathcal{O}(Nr^2) + \mathcal{O}(Nr)$.

Gradient-based [15]: Updating covariance matrix $\mathbf{R}_y(k)$, $\mathbf{R}_x(k)$ and its inverse $\mathbf{Q}_x(k)$ costs $10N^2 + \mathcal{O}(N)$ flops. In each loop, updating each generalized eigenvector requires approximately $3N^2 + \mathcal{O}(N)$ flops. Note that exploiting the orthogonal complement structure in order to extract remaining generalized eigenvectors is very computationally costly, as it needs $\mathcal{O}(Nr^2)$ in each loop. Hence, the dominant cost of Gradient-based method is $3N^2r + 10N^2 + \mathcal{O}(Nr^3)$.

GSVD [42]: GSVD is implemented using singular value decomposition and QR factorization. Hence, the cost of this method is $\mathcal{O}(N^3)$ flop counts.

TABLE III: Computational complexity for different methods

Method	Complexity
Algorithm 1	$N^2r + 16N^2 + \mathcal{O}(Nr^2) + \mathcal{O}(Nd)$
Algorithm 2	$13N^2 + \mathcal{O}(Nr^2) + \mathcal{O}(Nd)$
PI-based	$2N^2r + 15N^2 + \mathcal{O}(Nr^2)$
PAST-based	$2N^2r + 4N^2 + \mathcal{O}(Nr)$
R-GEVE	$8N^2 + \mathcal{O}(Nr^2) + \mathcal{O}(Nr)$
Gradient-based	$3N^2r + 10N^2 + \mathcal{O}(Nr^3)$
GSVD	$\mathcal{O}(N^3)$

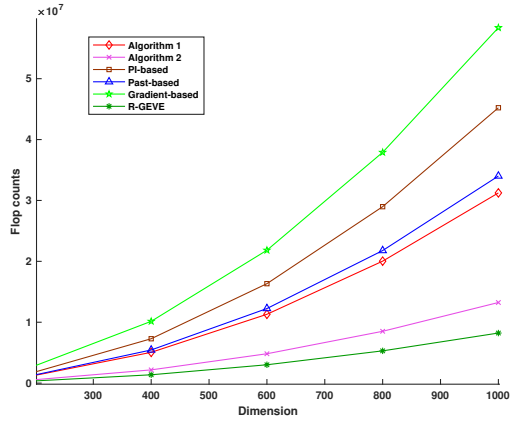


Fig. 6: Flop counts of the recursive algorithms versus the matrix dimension N, where r and d is fixed to $r = d = 15$.

Computational complexity of compared algorithms are shown in Table III. Though R-GEVE requires the lowest order of computation among these algorithms, it may not be of practical use due to the difficulty in convergence. Algorithm 2 shares the same computational order, however it achieves better balance between fast tracking and lower computational cost.

Fig. 6 shows the flop counts of the recursive algorithms versus the matrix dimension N, with $N \gg r$, in which case the terms associated with $\mathcal{O}(\cdot)$ can be ignored. It is observed that Algorithms 1 and 2 are among the most efficient algorithms. Although R-GEVE requires slightly less computations, it does not converge over finite iterations as has been demonstrated by our numerical examples 4.

This article has been accepted for publication in IEEE Transactions on Vehicular Technology. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVT.2023.3243244

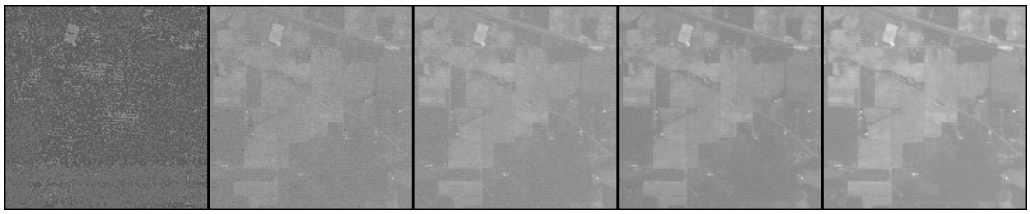


Fig. 7: Illustration of AVIRIS scenes from spectral band 1 (left) to band 5 (right).

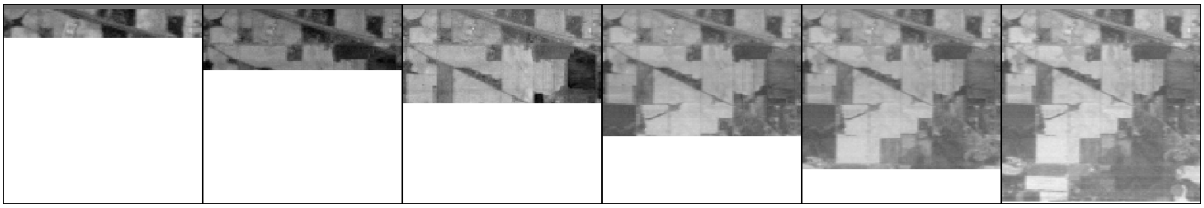


Fig. 8: Illustration of real-time processing results of Algorithm 1 for the first principal component.

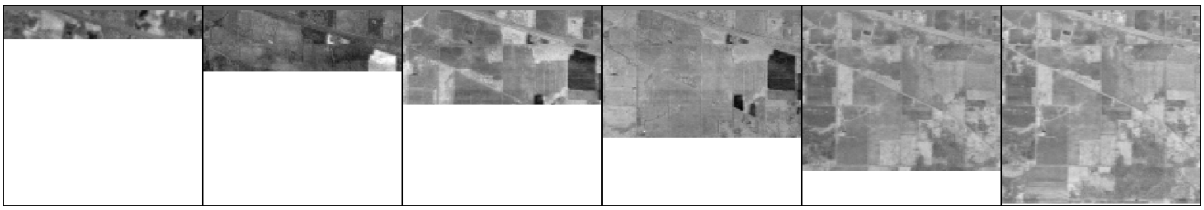


Fig. 9: Illustration of real-time processing results of Algorithm 1 for the second principal component..

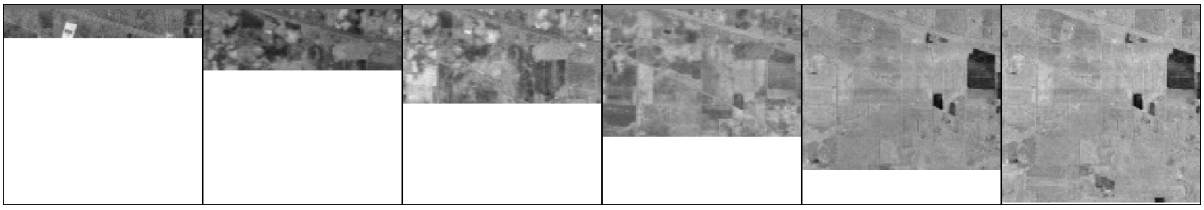


Fig. 10: Illustration of real-time processing results of Algorithm 1 for the third principal component.

G. Example 5: real-time hyperspectral image denoising

This subsection considers an application of real-time hyperspectral image denoising to evaluate the practical usability of the proposed algorithms. As far as we know, hyperspectral features provide distinctive information for real-time tracking of the moving vehicle due to different reflectance characteristics from the target [47]. To capture the information required for this purpose, line-scanning hyperspectral cameras are widely used in aerial platforms because they are cost-effective and able to achieve the same resolution as the megapixel area scan camera. A major concern that arises in rapid scanning by the line-scanning cameras, however, is how to implement

This article has been accepted for publication in IEEE Transactions on Vehicular Technology. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVT.2023.3243244

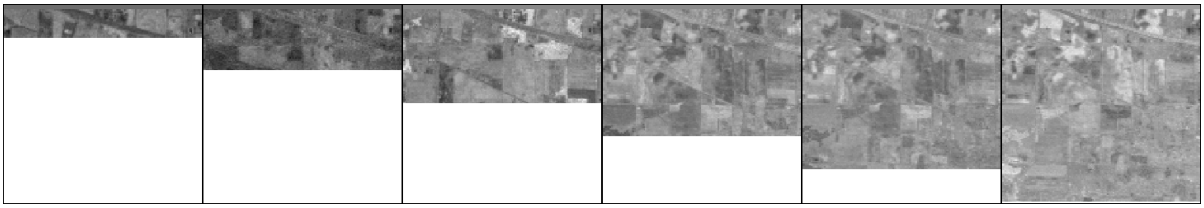


Fig. 11: Illustration of real-time processing results of Algorithm 1 for the fourth principal component.

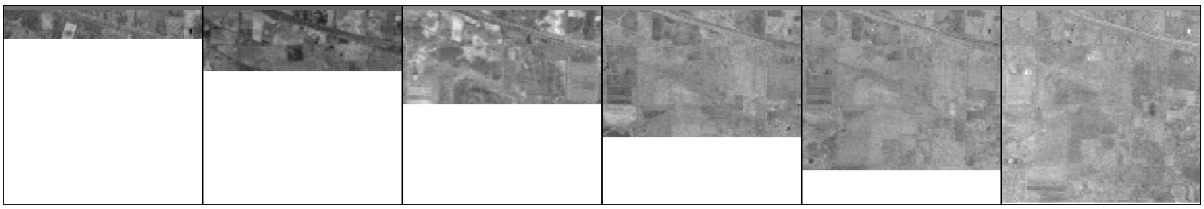


Fig. 12: Illustration of real-time processing results of Algorithm 1 for the fifth principal component.

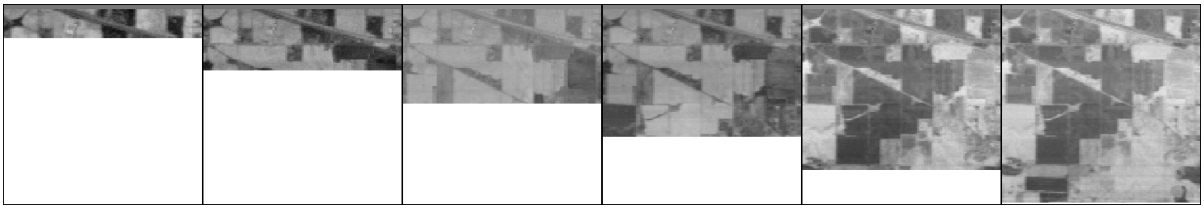


Fig. 13: Illustration of real-time processing results of Algorithm 2 for the first principal component.

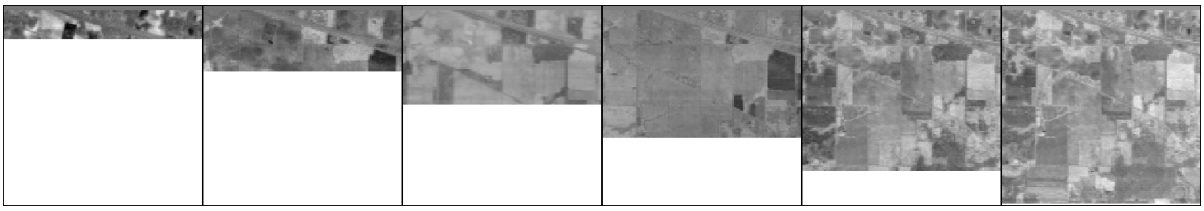


Fig. 14: Illustration of real-time processing results of Algorithm 2 for the second principal component.

noise removal in real-time [48]. Removal of noisy information can be realized, e.g., by arranging the image component according to the quality of the image. This can be done using the minimum noise fraction (MNF) transform [4]. The filter basis involved in the MNF transform is constructed by the generalized eigenvectors of the image covariance and noise covariance [49], [50]. We therefore make use of our proposed algorithms to perform the online NMF transform for hyperspectral image denoising.

A real-world hyperspectral image data from the 1992

This article has been accepted for publication in IEEE Transactions on Vehicular Technology. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVT.2023.3243244

Fig. 15: Illustration of real-time processing results of Algorithm 2 for the third principal component.

Fig. 16: Illustration of real-time processing results of Algorithm 2 for the fourth principal component.

Fig. 17: Illustration of real-time processing results of Algorithm 2 for the fifth principal component.

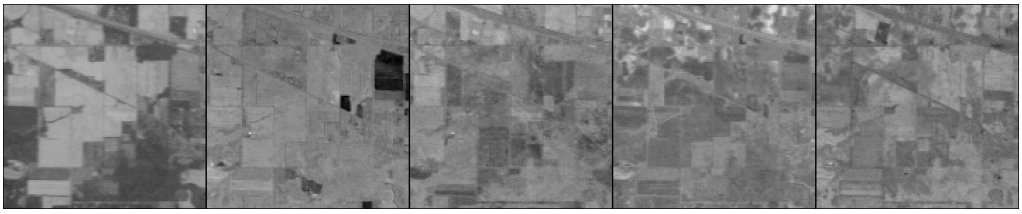


Fig. 18: Illustration of the first five principal components obtained by performing NMF over global data.

AVIRIS India Pines dataset is considered for this example [51]. This AVIRIS scene was taken using 224 spectral reflectance bands covering the wavelength range $0.4 - 2.5$ μm with a spatial resolution of 145×145 pixels. Fig. 7 shows the first five spectral bands of the AVIRIS image. In this experiment, we assume that the line-scan camera builds the rectangular area of the scene from top to down. The noise samples are estimated by using a simple subtraction between the neighbouring lines of image samples. We then use Algorithms 1 and 2 to track the generalized eigenvectors from the image covariance $\mathbf{R}_y$ and noise covariance $\mathbf{R}_x$ for real-time denoising. For both algorithms, the parameters r and d are set to $d = r = 10$. The image covariance $\mathbf{R}_y$ and noise covariance $\mathbf{R}_x$ are initialized using the first three lines of samples to ensure a good initial-

ization. The forgetting factors are set to $\alpha = \beta = 0.9999$ to capture the sample and noise statistics as much as possible.

Figs. 8 to 12 show the real-time processing results of Algorithm 1 for the first principal component to the fifth principal component, respectively. Figs. 13 to 17 show the real-time processing results obtained by Algorithm 2. We only demonstrate some of intermediate results due to space constraints. For comparison, the denoising results obtained by performing NMF over global data is also shown in Fig. 18. From these figures, we observe that both Algorithms achieve considerably improved quality of the image and can clean the pixels from the noisy samples. Besides, Algorithm 1 and Algorithm 2 achieve comparable performance with that of the global NMF over the local regions of the processed image. However, the adaptive nature and high computational efficiency of our proposed algorithms make them more suitable for online hyperspectral image denoising applications. Nonetheless, the applications of the algorithms proposed in this work extend beyond real-time hyperspectral denosing. Future work can be extended to, e.g., sound zone generalization for cocktail party effect [2], design of efficient receiver for non-orthogonal multiple access system in wireless communication [3], and design of multichannel wiener filter [7].

VI. Conclusion

In this paper, we firstly proposed the APR-EVD algorithm that solves the standard eigenvalue decomposition through randomization. We also provided a theoretical analysis for APR-EVD. Then, by exploiting the rank-1 update strategy, we developed two computationally efficient algorithms for online dominant generalized eigenvectors extraction. We conducted several experiments: in Example 1, we experimentally investigated the error bound of APR-EVD algorithm. Simulation results demonstrate that APR-EVD is highly accurate for low-rank approximation. In Example 2, time-varying environment with randomly generated matrix pencil was taken into consideration. Simulation results demonstrate that our proposed algorithms has fastest tracking speed among competing algorithms. In addition, the runtime results suggest that Algorithm 1 and Algorithm 2 are advantageous over the other algorithms under high-dimensional consideration. In Example 3, we evaluated the applicability of our proposed algorithms in recovering multiple dominant generalized eigenvectors from two real-valued random processes defined in time domain. Our results show that Algorithm 2 outperforms the existing algorithms in terms of convergence speed, estimation accuracy and numerical stability. In Example 4, we extracted the largest generalized eigenvector for two complex-valued random processes defined in spatial domain. Our results demonstrate that our proposed algorithms exhibit fastest convergence speed among the considered algorithms, while rendering comparable results. We also considered the application of real-time denoising for hyperspectral imaging. Experimental results suggest that our proposed online algorithms can provide high-quality images.

Acknowledgement

The work of J. Chen was supported in part by NSFC (Grants 62192713, 62192710 and 62171380), Shenzhen Science and Technology Program JCYJ20220530161606014, Department of Nature Resources of Guangdong Province (Oceanic Economy Development Project: An active system for long-distance detection of small underwater targets), Guangdong International Cooperation Project 2022A0505050020, Shaanxi Key Industrial Innovation Chain Project 2022ZDLGY01-02, and Xi'an Technology In dustrialization Plan XA2020-RGZNTJ-0076. The work of W. Chen was supported in part by NSFC (Grant 62122012) and Beijing Natural Science Foundation (L202019, L211012). The work of C. Richard was supported by the French government through the 3IA Côte dAzur Investments in the Future project managed by the National Research Agency (ANR) with the reference number ANR-19-P3IA-0002.

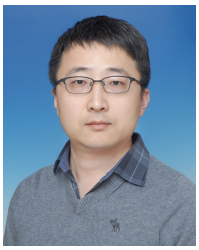
Haoyuan Cai (Student Member, IEEE) received the B.S. degree in electronic and information engineering in 2019 and the M.S. degree in signal and information processing in 2022 from Northwestern Polytechnical University, Xi'an, China. He is currently working towards the Ph.D. degree in the School of Engineering, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland. His research interests include machine learning, distributed learning and network science.

Maboud F. Kaloorazi received the B.Sc. degree from Guilan University, Rasht, Iran, in 2007, the M.Sc. degree from Blekinge Institute of Technology, Karlskrona, Sweden, in 2013, and the Ph.D. degree from Pontifical Catholic University of Rio de Janeiro (PUC-Rio), Brazil, in 2018, all in electrical engineering. From October 2018 to 2020, he was a Postdoctoral Researcher with the School of Marine Science and Technology, Northwestern Polytechnical University, Xi'an, China. He is currently a Professor with the School of Electronic Engineering, Xi'an Shiyou University, Xi'an, China. His research interests include matrix computations through leveraging randomization for high-dimensional data analysis and machine learning methods for large scientific problems.

Jie Chen (Senior Member, IEEE) received the B.S. degree from Xi'an Jiaotong University, Xi'an, China, in 2006, the Dipl.-Ing. and M.S. degrees in information and telecommunication engineering from the University of Technology of Troyes (UTT), Troyes, France, and Xi'an Jiaotong University, respectively, in 2009, and the Ph.D. degree in systems optimization and security from UTT in 2013.

From 2013 to 2014, he was with the Lagrange-Laboratory, University of Nice Sophia Antipolis, Nice, France. From 2014 to 2015, he was with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI, USA. He is a Professor with Northwestern Polytechnical University, Xi'an, China. His research interests include adaptive signal processing, distributed optimization, hyperspectral image analysis, and acoustic signal processing.

Dr. Chen was the Technical Co-Chair of IWAENC'16, IEEE ICSPCC'21 and 22 held in Xi'an, China. He serves as a Distinguished Lecturer for Asia-Pacific Signal and Information Processing Association (2018-2019), a Co-Chair of the IEEE Signal Processing Society Summer School 2019 in Xi'an, and the General Co-Chair of IEEE MLSP 2022.



Wei Chen (Senior Member, IEEE) received the B.Eng. degree and M.Eng. degree from Beijing University of Posts and Telecommunications, China, in 2006 and 2009, respectively, and the Ph.D. degree in Computer Science from the University of Cambridge, UK, in 2013. Later, he was a Research Associate with the Computer Laboratory, University of Cambridge from 2013 to 2016. He is currently a Professor with Beijing Jiaotong University, Beijing, China. He is the recipient of the 2013 IET Wireless Sensor Systems Premium Award and the 2017 International Conference on Computer Vision (ICCV) Young Researcher Award. His current research interests include AI/ML enabled wireless communications and intelligent information processing.

Cdric Richard (Senior Member, IEEE) received the Dipl.-Ing. and the M.S. degrees in 1994, and the Ph.D. degree in 1998, from the Compiègne University of Technology, Compiègne, France, all in electrical and computer engineering. He is a Full Professor with Université Côte dAzur, Nice, France. He is the author of more 350 journal and conference papers. His current research interests include statistical signal processing and machine learning.

Dr. Richard is the Director of the French federative CNRS research association ISIS (Information, Signal, Image, Vision). In 2010-2015, he was distinguished as a Junior Member of the Institut Universitaire de France. He is head of a Chair on AI for smart and secure territories at the 3IA Cte d'Azur Interdisciplinary Institute for Artificial Intelligence.

Dr. Richard, since 2020, has served as a Senior Area Chair for the IEEE Signal Processing Letters and, since 2019, as an Associate Editor for the IEEE Open Journal of Signal Processing. In 2015-2018, he was also a Senior Area Chair for the IEEE Transactions on Signal Processing, and an Associate Editor for the IEEE Transactions on Signal and Information Processing over Networks. He was an Associate Editor for the IEEE Transactions on Signal Processing (2006-2010). He is the Chair of the IEEE Signal Processing Theory and Methods Technical Committee. In 2019-2020, Prof. Richard served as the Director-at-Large of Region 8 (Europe, Middle East, and Africa) of the IEEE Signal Processing Society (IEEE-SPS) and as a Member of the Board of Governors of the IEEE-SPS.

REFERENCES

[1] H. Cai, M. F. Kaloorazi, J. Chen, W. Chen, and C. Richard, "Online dominant generalized eigenvectors extraction via a randomized method," in *EUSIPCO, Netherland*, 2020.

[2] L. Shi, T. Lee, J. Nielsen, and M. Christensen, "Subspace-based methods for the generation of personal sound zones with physically meaningful constraints," in *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, 2020.

[3] H. Iimori, G. T. F. De Abreu, T. Hara, K. Ishibashi, R.-A. Stoica, D. González, and O. Gonsa, "Robust symbol detection in large-scale overloaded noma systems," *IEEE Open Journal of the Communications Society*, vol. 2, pp. 512–533, 2021.

[4] A. A. Green, M. Berman, P. Switzer, and M. D. Craig, "A transformation for ordering multispectral data in terms of image quality with implications for noise removal," *IEEE Transactions on geoscience and remote sensing*, vol. 26, no. 1, pp. 65–74, 1988.

[5] R. Ge, C. Jin, P. Netrapalli, A. Sidford *et al.*, "Efficient algorithms for large-scale generalized eigenvector computation and canonical correlation analysis," in *International Conference on Machine Learning*. PMLR, 2016, pp. 2741–2750.

[6] K. Lee and J. Kim, "On the equivalence of linear discriminant analysis and least squares," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, no. 1, 2015.

[7] A. Hassani, A. Bertrand, and M. Moonen, "GEVD-based low-rank approximation for distributed adaptive node-specific signal estimation in wireless sensor networks," *IEEE Transactions on Signal Processing*, vol. 64, no. 10, pp. 2557–2572, 2015.

[8] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, *Templates for the solution of Algebraic Eigenvalue Problems: A Practical Guide*. SIAM, 2000.

[9] D. R. Morgan, "Adaptive algorithms for solving generalized eigenvalue signal enhancement problems," *Signal Processing*, vol. 84, no. 10, pp. 957–968, Jun 2004.

[10] S. Choi, J. Choi, H. J. Im, and B. Choi, "A novel adaptive beamforming algorithm for antenna array CDMA systems with strong interferers," *IEEE Transactions on Vehicular Technology*, vol. 51, no. 5, pp. 808–816, Sep 2002.

[11] T. F. Wong, T. M. Lok, J. S. Lehnert, and M. D. Zoltowski, "A linear receiver for direct-sequence spread-spectrum multiple-access systems with antenna arrays and blind adaptation," *IEEE Transactions on Information Theory*, vol. 44, no. 2, pp. 659–676, Mar 1998.

[12] J. Yang, H. Xi, F. Yang, and Y. Zhao, "RLS-based adaptive algorithms for generalized eigen-decomposition," *IEEE Trans. Signal Process.*, vol. 54, no. 4, pp. 1177–1188, Apr 2006.

[13] S. Attallah and K. Abed-Meraim, "A fast adaptive algorithm for the generalized symmetric eigenvalue problem," *IEEE Signal Processing Letters*, vol. 15, no. 11, pp. 797–800, Nov 2008.

[14] T. Tanaka, "Fast generalized eigenvector tracking based on the power method," *IEEE Signal Processing Letters*, vol. 16, no. 11, pp. 969–972, Nov 2009.

[15] T. D. Nguyen, N. Takahashi, and I. Yamada, "An adaptive extraction of generalized eigensubspace by using exact nested orthogonal complement structure," *Multidimensional Systems and Signal Processing*, vol. 24, no. 3, pp. 457–483, Sep 2013.

[16] X. Feng, X. Kong, Z. Duan, and H. Ma, "Adaptive generalized eigenpairs extraction algorithms and their convergence analysis," *IEEE Trans. Signal Process.*, vol. 64, no. 11, pp. 2976–2989, Jun 2016.

[17] L. Parra and P. Sajda, "Blind source separation via generalized eigenvalue decomposition," *Journal of Machine Learning Research*, vol. 4, pp. 1261–1269, Dec 2003.

[18] G. Yingbin, K. Xiangyu, Z. Zhengxin, and H. Li'an, "An adaptive self-stabilizing algorithm for minor generalized eigenvector extraction and its convergence analysis," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 10, pp. 4869–4881, Oct 2018.

[19] X. Han and L. Clemmensen, "Regularized generalized eigen-decomposition with applications to sparse supervised feature extraction and sparse discriminant analysis," *Pattern Recognition*, vol. 49, pp. 43–54, Dec 2016.

[20] G. Yuan, L. Shen, and W. Zheng, "A Decomposition Algorithm for the Sparse Generalized Eigenvalue Problem," in *CVPR*, 2019, pp. 6113–6122.

[21] A. Valizadeh and M. Najibi, "A constrained optimization approach for an adaptive generalized subspace tracking algorithm," *Computers & Electrical Engineering*, vol. 36, no. 4, pp. 596–602, 2010.

[22] H. Chen, G. Jiang, and K. Yoshihira, "Failure detection in large-scale internet services by principal subspace mapping," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 10, pp. 1308–1320, 2007.

[23] S. Ding, X. Hua, and J. Yu, "An overview on nonparallel hyperplane support vector machine algorithms," *Neural computing and applications*, vol. 25, no. 5, pp. 975–982, 2014.

[24] J. Benesty, M. G. Christensen, and J. R. Jensen, *Signal enhancement with variable span linear filters*. Springer, 2016, vol. 7.

[25] M. M. A. Hassani, A. Bertrand, "LCMV beamforming with subspace projection for multi-speaker speech enhancement," in *ICASSP, China*, Mar 2016, pp. 91–95.

[26] B. Yang, "Projection approximation subspace tracking," *IEEE Trans. Signal Process.*, vol. 43, no. 1, pp. 95–107, Jan 1995.

[27] C. E. Davila, "Efficient, high performance, subspace tracking for time-domain data," *IEEE Trans. Signal Process.*, vol. 48, no. 12, pp. 3307–3315, 2000.

[28] A. Frieze, R. Kannan, and S. Vempala, "Fast monte-carlo algorithms for finding low-rank approximations," *Journal of the ACM (JACM)*, vol. 51, no. 6, pp. 1025–1041, 2004.

[29] P. Drineas, R. Kannan, and M. W. Mahoney, "Fast monte carlo algorithms for matrices ii: Computing a low-rank approximation to a matrix," *SIAM Journal on computing*, vol. 36, no. 1, pp. 158–183, 2006.

[30] V. Rokhlin, A. Szlam, and M. Tygert, "A randomized algorithm for principal component analysis," *SIAM Journal on Matrix Analysis and Applications*, vol. 31, no. 3, pp. 1100–1124, 2010.

[31] T. Sarlos, "Improved approximation algorithms for large matrices via random projections," in *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*. IEEE, 2006, pp. 143–152.

[32] M. F. Kaloorazi and R. C. de Lamare, "Subspace-Orbit Randomized Decomposition for Low-Rank Matrix Approximations," *IEEE Trans. Signal Process.*, vol. 66, no. 16, pp. 4409–4424, Aug 2018.

[33] M. F. Kaloorazi and J. Chen, "Randomized Truncated Pivoted QLP Factorization for Low-Rank Matrix Recovery," *IEEE Signal Processing Letters*, vol. 26, no. 7, pp. 1075–1079, Jul 2019.

[34] N. Halko, P.-G. Martinsson, and J. Tropp, "Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions," *SIAM Review*, vol. 53, no. 2, pp. 217–288, Jun 2011.

[35] M. Gu, "Subspace Iteration Randomization and Singular Value Problems," *SIAM J. Sci. Comput.*, vol. 37, no. 3, pp. A1139–A1173, 2015.

[36] M. F. Kaloorazi and J. Chen, "Efficient low-rank approximation of matrices based on randomized pivoted decomposition," *IEEE Transactions on Signal Processing*, vol. 68, pp. 3575–3589, 2020.

[37] A. K. Saibaba, J. Lee, and P. K. Kitanidis, "Randomized algorithms for generalized hermitian eigenvalue problems with application to computing karhunen–loève expansion," *Numerical Linear Algebra with Applications*, vol. 23, no. 2, pp. 314–339, 2016.

[38] F. Woolfe, E. Liberty, V. Rokhlin, and M. Tygert, "A Fast Randomized Algorithm for the Approximation of Matrices," *Applied and Computational Harmonic Analysis*, vol. 25, no. 3, pp. 335–366, Nov 2008.

[39] G. W. Stewart, *Matrix Algorithms: Volume 1: Basic Decompositions*, SIAM, Philadelphia, PA, (1998).

[40] Y. Saad, *Numerical Methods for Large Eigenvalue Problems*. SIAM, 2nd Ed., 2011.

[41] G. Stewart, "A generalization of Saad's theorem on Rayleigh–Ritz approximations," *Linear Algebra and its App.*, vol. 327, no. 1-3, pp. 115–119, Apr 2001.

[42] G. H. Golub and C. F. van Loan, *Matrix Computations*, 3rd ed., Johns Hopkins Univ. Press, Baltimore, MD, (1996).

[43] K. Hermus, P. Wambacq, and H. Van Hamme, "A review of signal subspace speech enhancement and its application to noise robust speech recognition," *EURASIP journal on advances in signal processing*, vol. 2007, pp. 1–15, 2006.

[44] A. Björck and G. H. Golub, "Numerical methods for computing angles between linear subspaces," *Mathematics of computation*, vol. 27, no. 123, pp. 579–594, 1973.

[45] P. Å. Wedin, "On angles between subspaces of a finite dimensional inner product space," in *Matrix Pencils*. Springer, 1983, pp. 263–285.

[46] T. D. Nguyen and I. Yamada, "Adaptive normalized quasi-Newton algorithms for extraction of generalized eigen-pairs and their convergence analysis," *IEEE Trans. Signal Process.*, vol. 61, no. 6, pp. 1404–1418, Mar 2012.

[47] B. Uzkent, M. J. Hoffman, and A. Vodacek, "Real-time vehicle tracking in aerial video using hyperspectral features," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016, pp. 36–44.

[48] A. Bjorgan and L. L. Randeberg, "Real-time noise removal for line-scanning hyperspectral devices using a minimum noise fraction-based approach," *Sensors*, vol. 15, no. 2, pp. 3362–3378, 2015.

[49] C.-I. Chang and Q. Du, "Interference and noise-adjusted principal components analysis," *IEEE transactions on geoscience and remote sensing*, vol. 37, no. 5, pp. 2387–2396, 1999.

[50] N. Yokoya and A. Iwasaki, "A maximum noise fraction transform based on a sensor noise model for hyperspectral data," in *Proceedings of the 31st Asian Conference on Remote Sensing (ACRS),(Tokyo: The University of Tokyo)*, 2010.

[51] M. F. Baumgardner, L. L. Biehl, and D. A. Landgrebe, "220 band aviris hyperspectral image data set: June 12, 1992 indian pine test site 3," *Purdue University Research Repository*, vol. 10, p. R7RX991C, 2015.