Signal Processing 142 (2018) 492–503

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

Polynomial dictionary learning algorithms in sparse representations

Jian Guan[a], Xuan Wang[a,*], Pengming Feng[b], Jing Dong[c], Jonathon Chambers[d], Zoe L. Jiang[a], Wenwu Wang[e]

[a] Computer Application Research Center, Harbin Institute of Technology, Shenzhen, 518055, China
[b] State Key Laboratory of Space-Ground Integrated Information Technology, Beijing Institute of Satellite Information Engineering, Beijing, 100029, China
[c] College of Electrical Engineering and Control Science, Nanjing Tech University, Nanjing, 211800, China
[d] School of Electrical and Electronic Engineering, Newcastle University, Newcastle upon Tyne, NE1 7RU, UK
[e] Centre for Vision, Speech and Signal Processing, University of Surrey, Guildford, GU2 7XH, UK

ARTICLE INFO

Article history:
Received 6 April 2017
Revised 13 July 2017
Accepted 12 August 2017
Available online 14 August 2017

Keywords:
Dictionary learning
Polynomial matrix
Impulse responses
Sparse representation

ABSTRACT

Dictionary learning has been extensively studied in sparse representations. However, existing dictionary learning algorithms are developed mainly for standard matrices (i.e. matrices with scalar elements), and little attention has been paid to polynomial matrices, despite their wide use for describing convolutive signals or for modeling acoustic channels in room and underwater acoustics. In this paper, we propose a polynomial dictionary learning technique to deal with signals with time delays. We present two types of polynomial dictionary learning methods based on the fact that a polynomial matrix can be represented either as a polynomial of matrices (i.e. the coefficient in the polynomial corresponding to each time lag is a scalar matrix) or equally as a matrix of polynomial elements (i.e. each element of the matrix is a polynomial). The first method allows one to extend any state-of-the-art dictionary learning method to the polynomial case; and the second method allows one to directly process the polynomial matrix without having to access its coefficient matrices. A sparse coding method is also presented for reconstructing convolutive signals based on a polynomial dictionary. Simulations are provided to demonstrate the performance of the proposed algorithms, e.g. for polynomial signal reconstruction from noisy measurements.

1. Introduction

Dictionary learning has been widely used in many applications, such as signal denoising [1,2], source separation [3–6], and image super-resolution [7]. Several algorithms have been proposed for this problem, such as method of optimal directions (MOD) [8], K-SVD [9], and simultaneous codeword optimization (SimCO) [10], often with a two-stage process alternating between sparse coding and dictionary update. The sparse coding step aims to find the sparse coefficient matrix of a signal for a given dictionary using algorithms, such as matching pursuit (MP) [11], the least absolute shrinkage and selection operator (LASSO) [12], focal underdetermined system solver (FOCUSS) [13], orthogonal least squares (OLS) [14–16], and orthogonal matching pursuit (OMP) [17–28]. The dictionary update step aims to revise the dictionary at the current iteration to better fit the training signals with the sparse coefficient matrix obtained from the previous iteration.

Although the conventional dictionary learning methods have been studied extensively, they cannot be applied directly to deal with signals with time delays, such as acoustic impulse responses, and reverberant (convolutive) signals. Such signals are often described with polynomials or polynomial matrices, and encountered widely in digital signal processing and communications [29], e.g. for convolutive mixing [30] and multiple-input multiple-output (MIMO) channel modeling [31]. For example, an element of a polynomial matrix can be used to denote a finite impulse response (FIR) filter, e.g. in a MIMO system [31,32].

In this paper, we present a polynomial dictionary learning technique to deal with signals with time delays, where two types of polynomial dictionary learning methods are proposed based on how a polynomial matrix is represented. A polynomial matrix can be expressed in terms of *the polynomial of matrices model* or *the matrix of polynomials model* [30,33]. The first method is proposed based on the polynomial of matrices model, for which the polynomial dictionary learning problem can be converted to a conventional dictionary learning problem by concatenating the coefficient matrices of the polynomial matrix [34]. This allows the conven-

* Corresponding author.
E-mail addresses: j.guan@cs.hitsz.edu.cn (J. Guan), wangxuan@cs.hitsz.edu.cn (X. Wang), p.feng.cn@outlook.com (P. Feng), jingdong@njtech.edu.cn (J. Dong), Jonathon.Chambers@newcastle.ac.uk (J. Chambers), zoeljiang@gmail.com (Z.L. Jiang), w.wang@surrey.ac.uk (W. Wang).

http://dx.doi.org/10.1016/j.sigpro.2017.08.011
0165-1684/

tional dictionary learning methods (e.g. K-SVD, MOD, and SimCO) to be used to solve the polynomial dictionary learning problem. Even though this method can be used in dictionary learning for signals with time delays, it cannot be applied directly to the polynomial matrix (i.e. a matrix of polynomial elements). The second method, on the other hand, is proposed based on the matrix of polynomials model, where an idea similar to the conventional MOD algorithm is applied to the polynomial case. It has an advantage where dictionary learning can be directly performed on the polynomial matrices without having to first resort to their coefficient matrices as in the polynomial of matrices model. In addition, we present a polynomial OMP method by extending the conventional OMP to the polynomial case as a byproduct to calculate the representation coefficients for signal reconstruction.

The proposed polynomial dictionary learning technique can be used for modeling acoustic impulse responses, thereby having potential applications in e.g. denoising, dereverberation, deconvolution, and channel shortening of acoustic impulse responses. Each element of the polynomial matrix can be seen as an FIR filter, and the atoms in the learned dictionary also represent FIR filters. As a result, the polynomial dictionary, which is learned from a set of acoustic impulse responses, can provide an overall description of the acoustic environment. In this paper, we demonstrate the performance of the proposed polynomial dictionary learning algorithms for acoustic impulse response modeling and denoising.

The remainder of the paper is organized as follows: Section 2 briefly introduces the background of conventional dictionary learning and polynomial matrices; Section 3 presents the proposed polynomial dictionary learning methods in detail; Section 4 evaluates the performance of the proposed algorithms, using simulations and experiments on acoustic impulse response modeling and denoising; and Section 5 concludes the paper and discusses potential future work.

2. Background

2.1. Dictionary learning

Dictionary learning aims to learn a dictionary with a set of training signals, so that each training signal can be represented by a small number of atoms chosen from the dictionary. Typically, this can be modeled as

$$\mathbf{Y} = \mathbf{D}\mathbf{X}, \tag{1}$$

where $\mathbf{Y} \in \mathbb{R}^{n\times N}$ is the set of training signals $\{\mathbf{y}_i\}_{i=1}^N$, $\mathbf{D} \in \mathbb{R}^{n\times K}$ $(n \ll K)$ is the overcomplete dictionary containing K atoms $\{\mathbf{d}_j\}_{j=1}^K \in \mathbb{R}^n$, and $\mathbf{X} \in \mathbb{R}^{K\times N}$ is the sparse representation matrix.

To find the dictionary, the following optimization problem is often considered

$$\begin{aligned} &\min_{\mathbf{D},\mathbf{X}} \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2 \\ &\text{subject to } \forall i,\ \|\mathbf{x}_i\|_0 \le \kappa, \end{aligned} \tag{2}$$

where $\mathbf{x}_i$ is the ith column of the matrix $\mathbf{X}$, $\|.\|_0$ denotes the number of nonzero entries in the argument, and κ controls the sparsity level, i.e. the maximum number of the nonzero entries in each column. The Frobenius norm (F-norm) is defined as $\|\mathbf{M}\|_F = \sqrt{\sum_i \sum_j \mathbf{M}_{ij}^2}$, where $\mathbf{M}_{ij}$ is the (i, j)th element of $\mathbf{M}$.

The above optimization problem is usually solved using a two-step iterative process, alternating between sparse coding and dictionary update. In the sparse coding step, given the observation matrix $\mathbf{Y}$ and the dictionary matrix $\mathbf{D}$, $\mathbf{X}$ is estimated, subject to the constraint that each column of $\mathbf{X}$ is sparse (in the level of κ). In the dictionary update step, the dictionary matrix $\mathbf{D}$ is calculated based on the set of training signals $\{\mathbf{y}_i\}_{i=1}^N$ within $\mathbf{Y}$, and the sparse coefficient matrix $\mathbf{X}$ obtained in the previous step. This process is iterated until a pre-defined stopping criterion is met. Examples of such algorithms include MOD [8], K-SVD [9], and SimCO [10]. These algorithms, however, are designed only for scalar dictionary matrices. They are not directly applicable to polynomial matrices that are widely used for representing signals with time lags, such as acoustic impulse response or convolutive signals, as discussed next.

2.2. Polynomial matrices

Polynomial matrices have been widely used for describing transfer functions in MIMO systems [35], e.g. the collection of multiple-path channel impulse responses from the sources to the sensors. In an acoustic system, the polynomial matrix can be used to model the acoustic impulse responses, with each element of the polynomial matrix representing an FIR filter, which can be a segment of the impulse responses with relative short time lags.

A polynomial matrix can be represented using either a polynomial of matrices model (a polynomial whose coefficients are matrices), or a matrix of polynomials model (i.e. a matrix whose elements are polynomials). More specifically, for a $p \times q$ polynomial matrix $\boldsymbol{A}(z)$, we have

$$\begin{aligned} \boldsymbol{A}(z) &= \sum_{\ell=0}^{L-1} \mathbf{A}(\ell) z^{-\ell} \\ &= \begin{bmatrix} a_{11}(z) & a_{12}(z) & \cdots & a_{1q}(z) \\ a_{21}(z) & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ a_{p1}(z) & \cdots & \cdots & a_{pq}(z) \end{bmatrix}, \end{aligned} \tag{3}$$

where $\mathbf{A}(\ell) \in \mathbb{C}^{p\times q}$ is the coefficient matrix of $z^{-\ell}$, which denotes the impulse response at time lag ℓ, and L is the maximum time lag of each polynomial. Note that, L is set to be a positive integer here, however, the model can be easily extended for a negative L. In this paper, the polynomial matrix, e.g., $\boldsymbol{A}(z)$, is denoted in italic font to avoid confusion with its coefficient matrix, e.g., $\mathbf{A}(\ell)$, which we denote in normal font. We can see from (3) that $\boldsymbol{A}(z)$ can be expressed as a sum of terms with weights $z^{-\ell}$ and coefficient matrices $\mathbf{A}(\ell)$, $\ell = 0, \ldots, L-1$, or alternatively expressed as a matrix whose elements are polynomials. The (i, j)th element of $\boldsymbol{A}(z)$, $a_{ij}(z)$, can be expressed as

$$a_{ij}(z) = \sum_{\ell=0}^{L-1} a_{ij}(\ell) z^{-\ell}, \tag{4}$$

where the coefficient $a_{ij}(\ell)$ can be seen as the magnitude of the (i, j)th impulse response at time lag ℓ. The F-norm of $\boldsymbol{A}(z)$ can be defined as follows

$$\|\boldsymbol{A}(z)\|_F = \sqrt{\sum_{i=1}^{p}\sum_{j=1}^{q}\sum_{\ell=0}^{L-1} |a_{ij}(\ell)|^2}. \tag{5}$$

Note that, setting the filters in (3) to be the same length is mainly for the convenience of modeling and algorithmic implementation. In practice, for the FIR filters $a_{ij}(z)$ that have different lengths, one can set all the elements $a_{ij}(z)$ to be the same length with zero padding, i.e. setting the coefficients of the high-order taps of the shorter filters to be zeros.

There are several algorithms that have already been proposed for polynomial matrix decomposition, such as polynomial eigenvalue decomposition [35,36] and polynomial singular value decomposition [30,32,37]. However, no algorithms have yet been presented for polynomial matrix decomposition in a sparse representation context, which is our focus in this paper, as discussed next.

3. Polynomial dictionary learning

3.1. Proposed model

Based on the conventional dictionary learning model (1), we propose a polynomial dictionary learning model [34] as follows

$$\boldsymbol{Y}(z) = \boldsymbol{D}(z)\mathbf{X}, \tag{6}$$

where the polynomial matrix $\boldsymbol{Y}(z) \in \mathbb{R}^{n\times N}$ contains the signals (e.g. acoustic impulse responses) to be represented, $\boldsymbol{D}(z) \in \mathbb{R}^{n\times K}$ is the polynomial dictionary matrix with polynomial atoms, and $\mathbf{X} \in \mathbb{R}^{K\times N}$ is the sparse representation coefficient matrix of $\boldsymbol{Y}(z)$.

Similar to conventional dictionary learning, the aim here is to find a suitable polynomial dictionary $\boldsymbol{D}(z)$ for sparse representation of the "signals" denoted as polynomials $\boldsymbol{Y}(z)$, such as

$$\begin{aligned} &\min_{\boldsymbol{D}(z),\mathbf{X}} \|\boldsymbol{Y}(z) - \boldsymbol{D}(z)\mathbf{X}\|_F^2 \\ &\text{subject to } \forall i,\ \ \|\mathbf{x}_i\|_0 \le \kappa, \end{aligned} \tag{7}$$

3.2. Polynomial dictionary learning based on the polynomial of matrices model

In this section, we present a polynomial dictionary learning algorithm based on the optimization of (7) and using the polynomial of matrices model. To this end, as in our preliminary work [34], we can convert the polynomial model (6) to a conventional dictionary learning model [34]. As a result, any state-of-the-art dictionary learning methods could be used to address the optimization problem in (7).

According to equation (3), (6) can be rewritten as

$$\sum_{\ell=0}^{L-1} \mathbf{Y}(\ell) z^{-\ell} = \sum_{\ell=0}^{L-1} \mathbf{D}(\ell) z^{-\ell} \mathbf{X}, \tag{8}$$

where $\mathbf{Y}(\ell) \in \mathbb{R}^{n\times N}$ and $\mathbf{D}(\ell) \in \mathbb{R}^{n\times K}$ are the coefficient matrices of the polynomial matrices $\boldsymbol{Y}(z)$ and $\boldsymbol{D}(z)$ at lag ℓ, respectively. $\mathbf{Y}(\ell)$ can be seen as the impulse responses at lag ℓ. For any $\ell \in \{0, \dots, L-1\}$, $\mathbf{Y}(\ell)$ takes the same linear combination of the atoms in its corresponding $\mathbf{D}(\ell)$, and $\mathbf{X}$ is the sparse representation matrix for all these $\mathbf{D}(\ell)$s, i.e.

$$\mathbf{Y}(\ell) = \mathbf{D}(\ell)\mathbf{X}, \tag{9}$$

which means the coefficient matrices of $\boldsymbol{Y}(z)$ at all the time lags can be represented as the linear combination of the coefficient matrices of $\boldsymbol{D}(z)$ at their corresponding lags ℓ with the same sparse representation matrix $\mathbf{X}$. Therefore, (6) can be further rewritten as

$$\underline{\mathbf{Y}} = \underline{\mathbf{D}}\mathbf{X}, \tag{10}$$

where $\underline{\mathbf{Y}} \in \mathbb{R}^{nL\times N}$ and $\underline{\mathbf{D}} \in \mathbb{R}^{nL\times K}$ are defined by concatenating the coefficient matrices of $\boldsymbol{Y}(z)$ and $\boldsymbol{D}(z)$ at all the time lags, respectively, as

$$\underline{\mathbf{Y}} = [\mathbf{Y}(0); \dots; \mathbf{Y}(\ell); \dots; \mathbf{Y}(L-1)], \tag{11}$$

$$\underline{\mathbf{D}} = [\mathbf{D}(0); \dots; \mathbf{D}(\ell); \dots; \mathbf{D}(L-1)]. \tag{12}$$

As a result, the polynomial dictionary learning model (6) is converted to the conventional dictionary learning model (10). Therefore, the polynomial dictionary learning optimization problem (7) can be rewritten as

$$\begin{aligned} &\min_{\underline{\mathbf{D}},\mathbf{X}} \|\underline{\mathbf{Y}} - \underline{\mathbf{D}}\mathbf{X}\|_F^2 \\ &\text{subject to } \forall i,\ \ \|\mathbf{x}_i\|_0 \le \kappa, \end{aligned} \tag{13}$$

where the new dictionary $\underline{\mathbf{D}}$ is overcomplete, and it can be learned by many state-of-the-art dictionary learning methods. Usually, an alternating optimization strategy is employed to solve (13), by iteratively updating the dictionary and sparse coefficients. Assuming the dictionary is fixed, the sparse representation matrix $\mathbf{X}$ can be calculated by optimizing the following equation using methods such as OMP or FOCUSS [13]

$$\begin{aligned} &\min_{\mathbf{X}} \|\underline{\mathbf{Y}} - \underline{\mathbf{D}}\mathbf{X}\|_F^2 \\ &\text{subject to } \forall i,\ \ \|\mathbf{x}_i\|_0 \le \kappa, \end{aligned} \tag{14}$$

In [34], the K-SVD algorithm is employed to learn the dictionary $\underline{\mathbf{D}}$. Here, we assume $\underline{\mathbf{d}}_k$ is the kth column of $\underline{\mathbf{D}}$, and $\mathbf{x}_T^k$ contains its corresponding coefficients from the kth row of $\mathbf{X}_{\Omega_k}$, and Ω_k is the set of indices indicating which atom $\underline{\mathbf{d}}_k$ should be used for representing $\underline{\mathbf{Y}}$. Then $\underline{\mathbf{d}}_k$ and $\mathbf{x}_T^k$ can be updated by optimizing the following cost

$$\min_{\underline{\mathbf{d}}_k, \mathbf{x}_T^k} \|\mathbf{E}_k - \underline{\mathbf{d}}_k \mathbf{x}_T^k\|_F^2, \tag{15}$$

where $\mathbf{E}_k = \underline{\mathbf{Y}}_{\Omega_k} - \sum_{j\neq k} \underline{\mathbf{d}}_j \mathbf{X}_{j,\Omega_k}$ denotes the error matrix in which the kth atom is removed, and the optimization of (15) can be seen as a rank-1 matrix approximation problem, so that SVD can be used for the decomposition of $\mathbf{E}_k$ to minimize (15). The extended K-SVD algorithm for polynomial dictionary learning is given in Algorithm 1.

Algorithm 1 Extended K-SVD.

Input: Signal matrix $\boldsymbol{Y}(z)$, sparsity κ, the number of iterations I_n
Output: $\underline{\mathbf{D}}$ and $\mathbf{X}$
Polynomial Matrix Conversion:
Convert $\boldsymbol{Y}(z) = \sum_{\ell=0}^{L-1} \mathbf{Y}(\ell) z^{-\ell}$ to a scalar matrix $\underline{\mathbf{Y}}$, using (11).
Initialization: $\underline{\mathbf{D}}^{(0)} = \underline{\mathbf{Y}}(:, 1:K)$.
Iterations:
for $n = 1, \dots, I_n$
  Sparse Coding:
  Calculate sparse representations by using conventional OMP to solve (14).
  Dictionary Update:
  for $k = 1, \cdots, K$
    Define the set of indices Ω_k by finding the relevant elements in $\underline{\mathbf{Y}}$ which use atom $\underline{\mathbf{d}}_k$.
    Calculate $\mathbf{E}_k = \underline{\mathbf{Y}}_{\Omega_k} - \sum_{j\neq k} \underline{\mathbf{d}}_j \mathbf{X}_{j,\Omega_k}$.
    Update the dictionary atom and its corresponding sparse representation coefficient by using the SVD decomposition to minimize (15), as $(\underline{\mathbf{d}}_k, \mathbf{x}_T^k) = \mathrm{SVD}(\mathbf{E}_k)$.
  end for
end for

Alternatively, $\underline{\mathbf{D}}$ can also be learned by using other methods such as MOD [8]. Assuming $\mathbf{X}^{(n)}$ is the sparse representation matrix obtained at the nth iteration, the dictionary can then be obtained by solving the following optimization problem

$$\underline{\mathbf{D}}^{(n+1)} = \underset{\underline{\mathbf{D}}}{\operatorname{argmin}} \|\underline{\mathbf{Y}} - \underline{\mathbf{D}}\mathbf{X}^{(n)}\|_F^2, \tag{16}$$

where (16) can be seen as a least-squares problem, therefore the dictionary can be updated in terms of MOD as

$$\underline{\mathbf{D}}^{(n+1)} = \underline{\mathbf{Y}}\mathbf{X}^{(n)^T} \left(\mathbf{X}^{(n)}\mathbf{X}^{(n)^T}\right)^{-1}. \tag{17}$$

The dictionary $\underline{\mathbf{D}}$ can be obtained when the algorithm converges. The extended MOD algorithm is summarized in Algorithm 2.

Finally, $\boldsymbol{D}(z)$ can be obtained from $\underline{\mathbf{D}}$ with a reverse operation of (12), and $\underline{\mathbf{Y}}$ can be reconstructed using $\mathbf{X}$ obtained by applying the OMP algorithm, as

$$\hat{\underline{\mathbf{Y}}} = \left[\hat{\mathbf{Y}}(0); \dots; \hat{\mathbf{Y}}(\ell); \dots; \hat{\mathbf{Y}}(L-1)\right], \tag{18}$$

Algorithm 2 Extended MOD.

Input: signal matrix $\boldsymbol{Y}(z)$, sparsity κ, number of iterations I_n
Output: $\underline{\mathbf{D}}$ and $\mathbf{X}$
Polynomial Matrix Conversion:
Convert $\boldsymbol{Y}(z)=\sum_{\ell=0}^{L-1}\mathbf{Y}(\ell)z^{-\ell}$ to scalar matrix $\underline{\mathbf{Y}}$, using (11).
Initialization: $\underline{\mathbf{D}}^{(0)}=\underline{\mathbf{Y}}(:,1:K)$.
Iterations:
for $n=1,\ldots,I_n$
Sparse Coding:
Calculate sparse representations by using conventional OMP to solve (14).
Polynomial Dictionary Update:
Update the polynomial dictionary by solving (16), using (17).
end for

where $\hat{\mathbf{Y}}(\ell)$ is the coefficient matrix of the reconstructed polynomial matrix $\hat{\boldsymbol{Y}}(z)$ at lag ℓ, where $\ell\in\{0,\ldots,L-1\}$. With a reverse operation to equation (11), we can obtain the coefficient matrix $\hat{\mathbf{Y}}(\ell)$ of the reconstructed polynomial matrix $\hat{\boldsymbol{Y}}(z)$ at each time lag ℓ. Finally, $\hat{\boldsymbol{Y}}(z)$ can be obtained by employing (3) and (18).

Note that, $\underline{\mathbf{D}}$ can also be learned by using other state-of-the-art dictionary learning methods based on our proposed model (10). In this paper, both K-SVD and MOD are extended to the polynomial case, hence they are here named extended K-SVD and extended MOD, respectively.

3.3. Polynomial dictionary learning based on the matrix of polynomials model

In this section, we present another polynomial dictionary learning method by directly operating on $\boldsymbol{D}(z)$ and $\boldsymbol{Y}(z)$ based on the matrix of polynomials model. This (partially) avoids the process for converting the polynomial model to a conventional model.

To demonstrate the concept, we employ the same strategy as that in the conventional MOD algorithm. Given $\mathbf{X}^{(n)}$ obtained at the nth iteration, and the "signal" $\boldsymbol{Y}(z)$, where $\mathbf{X}^{(n)}$ is calculated by using the same method as in Section 3.2, then $\boldsymbol{D}(z)$ can be updated by optimizing the following cost function,

$$\boldsymbol{D}(z)^{(n+1)}=\underset{\boldsymbol{D}(z)}{\operatorname{argmin}}\|\boldsymbol{Y}(z)-\boldsymbol{D}(z)\mathbf{X}^{(n)}\|_F^2. \tag{19}$$

Similar to (16), (19) can be solved by extending (17) to the polynomial case, leading to

$$\boldsymbol{D}(z)^{(n+1)}=\boldsymbol{Y}(z)\mathbf{X}^{(n)^T}(\mathbf{X}^{(n)}\mathbf{X}^{(n)^T})^{-1}. \tag{20}$$

With (20), the polynomial dictionary is updated directly rather than operating on the polynomial coefficient matrices as in the methods described in Section 3.2. According to (19) and (20), the proposed polynomial MOD (PMOD) algorithm is summarized in Algorithm 3.

Algorithm 3 Polynomial MOD.

Input: signal matrix $\boldsymbol{Y}(z)$, sparsity κ, the number of iterations I_n
Output: $\boldsymbol{D}(z)$ and $\mathbf{X}$
Initialization: $\boldsymbol{D}(z)^{(0)}=\boldsymbol{Y}(z)(:,1:K)$.
Iterations:
for $n=1,\ldots,I_n$
Sparse Coding:
Calculate sparse representations by using conventional OMP to solve (14).
Polynomial Dictionary Update:
Update the polynomial dictionary by solving (19), using (20).
end for

The dictionary learned by the PMOD algorithm will be compared with the extended MOD in Section 4.

3.4. Polynomial sparse representation

In this section, we aim to find the sparse representation $\mathbf{X}$ of polynomial matrix $\boldsymbol{Y}(z)$ modeled signals, given the polynomial dictionary $\boldsymbol{D}(z)$, based on the polynomial dictionary learning model (6). Here, $\boldsymbol{D}(z)$ can be obtained by using the proposed methods in Section 3.2 or 3.3. As a byproduct, we propose a polynomial sparse representation method by extending the OMP algorithm to the polynomial case.

Assuming $\boldsymbol{y}(z)$ is an arbitrary polynomial "signal" from the set of polynomial signals $\boldsymbol{Y}(z)$, the sparse representation of $\boldsymbol{y}(z)$ can be calculated by optimizing the following cost function

$$\begin{aligned}&\min_{\mathbf{x}}\|\boldsymbol{y}(z)-\boldsymbol{D}(z)\mathbf{x}\|_F^2\\&\text{subject to }\|\mathbf{x}\|_0\le\kappa.\end{aligned} \tag{21}$$

Similar to the discussions in Sections 3.2 and 3.3, in order to optimize (21), we can convert the polynomial sparse representation problem (21) to the conventional sparse coding problem by concatenating the coefficient matrices of $\boldsymbol{Y}(z)$ and $\boldsymbol{D}(z)$ respectively. Therefore, (21) can be converted to

$$\begin{aligned}&\min_{\mathbf{x}}\|\underline{\mathbf{y}}-\underline{\mathbf{D}}\mathbf{x}\|_2^2\\&\text{subject to }\|\mathbf{x}\|_0\le\kappa,\end{aligned} \tag{22}$$

where $\underline{\mathbf{y}}$ denotes the vector obtained by concatenating all the coefficients of $\boldsymbol{y}(z)$ at all lags

$$\underline{\mathbf{y}}=[\mathbf{y}(0);\ldots;\mathbf{y}(\ell);\ldots;\mathbf{y}(L-1)]. \tag{23}$$

Many sparse coding algorithms can be used to optimize (22), such as the OMP algorithm. The OMP algorithm employs a greedy strategy to calculate the sparse coefficients by iteratively estimating the κ-nonzero coefficients to approximate the signal. For each iteration, the residual between the signal and its approximation is updated, where the approximation is calculated by selecting the best-matched atoms from the dictionary which can maximally reduce the ℓ_2-norm residual error between the signal and its approximation. When the error is reduced to below a specified threshold, the optimal sparse representation is obtained.

However, it is not trivial to find the match between the signal and the atoms in the polynomial case, as this involves the similarity measures between two polynomial vectors/matrices. In the conventional OMP algorithm, the similarity between the atom and the current residual is measured by their inner product, where the atom has the maximum inner product with the current residual being selected as the best-matched atom. This is not directly applicable for the polynomial case. Here we use the F-norm as the similarity measure between the polynomial residual and polynomial atoms, i.e. by calculating their distance using the F-norm. For each iteration, we select the polynomial atom (i.e. the column in the polynomial dictionary), which has the smallest F-norm error with the polynomial residual, as the best-matched dictionary atom.

Suppose $\boldsymbol{d}_{k_0}(z)$ is the k_0th column of the polynomial dictionary $\boldsymbol{D}(z)$, which is the best-matched polynomial atom at the current iteration j, then $\boldsymbol{d}_{k_0}(z)$ can be calculated as

$$\boldsymbol{d}_{k_0}(z)=\underset{\boldsymbol{d}_k(z)}{\operatorname{argmin}}\|\boldsymbol{d}_k(z)-\boldsymbol{r}(z)^{(j-1)}\|_F^2,\ \ k=1,\ldots,K, \tag{24}$$

where $\boldsymbol{d}_k(z)$ is the kth column of $\boldsymbol{D}(z)$, $\boldsymbol{r}(z)^{(j-1)}$ is the residual $\boldsymbol{r}(z)$ at the $(j-1)$th iteration, and $\boldsymbol{r}(z)$ is initialized by the signal $\boldsymbol{y}(z)$. The provisional solution $\mathbf{x}$ can then be obtained by optimizing the

following cost

$$\mathbf{x}^{(j)} = \underset{\mathbf{x}}{\operatorname{argmin}} \|\boldsymbol{y}(z) - \boldsymbol{D}(z)_{S^{(j)}}\mathbf{x}\|_F^2 \quad \text{subject to } \|\mathbf{x}\|_0 \le \kappa, \tag{25}$$

where $\boldsymbol{D}(z)_{S^{(j)}}$ contains the best-matched atoms indexed by the set $S^{(j)}$. The formulation (25) can be seen as a polynomial least-squares problem, as the coefficient matrices of $\boldsymbol{y}(z)$ at all different lags should have the same linear combination of the coefficient matrices of $\boldsymbol{D}(z)_{S^{(j)}}$ at their corresponding lags. According to (11), and (12), we can obtain the solution to (25) as

$$\mathbf{x}^{(j)} = (\underline{\mathbf{D}}_{S^{(j)}}^T \underline{\mathbf{D}}_{S^{(j)}})^{-1} \underline{\mathbf{D}}_{S^{(j)}}^T \underline{\mathbf{y}}, \tag{26}$$

where $\underline{\mathbf{D}}_{S^{(j)}}$ and $\underline{\mathbf{y}}$ are constructed by concatenating the coefficient matrices of $\mathbf{D}_{S^{(j)}}$ and $\boldsymbol{y}(z)$ at all lags, respectively. Then, at the jth iteration, the residual $\boldsymbol{r}(z)$ is updated as

$$\boldsymbol{r}^{(j)}(z) = \boldsymbol{y}(z) - \boldsymbol{D}(z)_{S^{(j)}}\mathbf{x}^{(j)}. \tag{27}$$

The proposed polynomial OMP (POMP) algorithm is given in Algorithm 4.

Algorithm 4 Polynomial OMP.

Input: signal $\boldsymbol{y}(z)$, dictionary $\boldsymbol{D}(z)$, sparsity κ
Output: $\mathbf{x}_{opt}$
Initialization: residual $\boldsymbol{r}(z)^{(0)} = \boldsymbol{y}(z)$, solution $\mathbf{x} = 0$, solution support $S^0 = \emptyset$, $\epsilon = 10^{-6}$.
Iteration:
for $j = 1, \ldots, \kappa$
Best-Matching Atom Selection:
Optimize $k_0 = \underset{k}{\operatorname{argmin}} \|\boldsymbol{d}_k(z) - \boldsymbol{r}(z)^{(j-1)}\|_F^2$, by calculating the F-norm error $\|\boldsymbol{d}_k(z) - \boldsymbol{r}(z)^{(j-1)}\|_F^2$, where $\boldsymbol{d}_k(z) \in \boldsymbol{D}(z)$, $k = 1, \cdots, K$.
Update Support Set: $S^{(j)} = S^{(j-1)} \bigcup \{k_0\}$.
Update Provisional Solution: Calculate $\mathbf{x}^{(j)}$ by solving (25), using (26).
Update Residual: $\boldsymbol{r}(z)^{(j)} = \boldsymbol{y}(z) - \boldsymbol{D}_{S^{(j)}}(z)\mathbf{x}^{(j)}$.
Stopping Criteria: If $\|\boldsymbol{r}(z)^{(j)}\|_F^2 \le \epsilon$, then $\mathbf{x}_{opt} = \mathbf{x}^{(j)}$, and break, else continue.
end for

3.5. Computational complexity

In this section, we analyse the computational complexity of the proposed algorithms. For the polynomial dictionary learning methods, i.e. extended K-SVD, extended MOD and PMOD, the computational complexities involved in the sparse coding stage in each iteration are dominated by the calculation of $\underline{\mathbf{D}}\mathbf{X}$, which are the same, i.e. $O(nLKN)$. In the dictionary update stage, however, the computational complexity of the extended K-SVD is dominated by the calculation of $\mathbf{E}_k$, which is $O(nLKN)$ for each $\mathbf{E}_k$, and overall at $O(nLK^2N)$. For the extended MOD and PMOD, the complexity is dominated by $\underline{\mathbf{Y}}\mathbf{X}^T$ and $\boldsymbol{Y}(z)\mathbf{X}^T$, as shown in (17) and (20), which require $O(nLNK)$ and $O(nNK)$ respectively, with pre-computed $\mathbf{X}\mathbf{X}^T$.

For the POMP algorithm, although the selection of the best-matching atom is different from that in the conventional OMP method, it requires the same number of iterations for atom selection. For each "signal" $\boldsymbol{y}(z)$, the computational complexity is dominated by the calculation of $\underline{\mathbf{D}}^T\underline{\mathbf{y}}$ with pre-computed $\underline{\mathbf{D}}^T\underline{\mathbf{D}}$ as shown in (26), which requires $O(nLK)$, and for a set of "signals" $\boldsymbol{Y}(z)$, the computational complexity is $O(nLKN)$.

3.6. Recoverability and RIP property

The restricted isometry property (RIP) of sparse recovery algorithms (e.g. the OMP and OLS algorithms) has been studied in the compressed sensing (CS) literature [16,21–28,38–42], and the dictionary learning context [43–47]. For example, in [26,40,41], sufficient conditions required by the OMP method were established for the exact κ-sparse signal recovery in the noiseless case or the exact support set recovery in the noise case, if the sensing matrix satisfies the RIP. The incoherence property has also been studied, for example, in [44] for dictionary learning, and in [39] for compressed sensing.

The proposed dictionary learning methods are the extensions of the conventional K-SVD and MOD methods to the polynomial case. Although the conventional K-SVD and MOD have been successfully used in real applications, these methods lack theoretical guarantees. In other words, the dictionaries learned by these methods cannot guarantee to satisfy the RIP [38,39] and incoherence property, and theoretical results of these methods have not yet been fully justified [43–45]. This is because both the K-SVD and MOD methods used an alternating minimization strategy to learn the dictionary in two steps, namely, sparse coding and dictionary update, by fixing one and updating the other. By using this strategy, the dictionary needs to be initialized, however, the initial guesses may be far from the true dictionary, which leads to the difficulty for providing provable guarantees for these algorithms [44]. In real applications, there is no ground truth dictionary, which makes it is even harder to provide such guarantees in practice. In addition, the algorithms may converge to a coherent dictionary, which can lead to unstable estimation for sparse recovery [43,44].

The extended K-SVD and extended MOD algorithms are based on the polynomial of matrices model, where we converted the polynomial dictionary learning problem to a conventional dictionary learning problem. Thus, similar to conventional K-SVD and MOD methods, the polynomial dictionaries obtained by using the extended K-SVD and extended MOD may not satisfy the RIP or incoherency property. The PMOD algorithm is based on the matrix of polynomials model, which is an extension of the MOD method. The PMOD method used the same strategy and stopping criterion as the MOD method to train the dictionary, where the polynomial dictionary is initialized with the "polynomial signals" (i.e. acoustic signals modeled with a polynomial matrix), which may also be far away from the true dictionary, and the dictionary obtained after convergence may not be incoherent. It is reasonable to deduce that the PMOD method may not be able to guarantee the RIP or incoherency property. However, further efforts are required to provide more precise theoretical results.

The proposed POMP is an extension of the conventional OMP to the polynomial case, and in the extreme scenario where no time delay (i.e. zero time lags) is involved, the proposed POMP degenerates to the conventional OMP (except the measure of similarity between the residual and the atoms in the dictionary). Therefore, the existing theoretical results established for conventional OMP in the literature could be extended to the polynomial case. However, it is not trivial to extend these theoretical results when multiple time lags are involved and extra attention need to be given to several important issues, such as the definition of the RIP property and incoherence measures in the polynomial setting. These are interesting future research directions that are beyond the scope of our current work.

4. Experiments and resluts

In this section, we evaluate the performance of the proposed methods using both synthetic and real data. We use a polynomial matrix to model signals with time lags, and therefore the polyno-

mial dictionaries are learned from training data consisting of signals with time lags. The learned polynomial dictionaries are used to recover the noisy signals. The experiments are first conducted on synthetic polynomial matrices to show how the proposed methods work on polynomial matrices, where the polynomial matrices are generated randomly, and each element of the polynomial matrices can be assumed as an FIR model represented by polynomials. Then, the proposed methods are evaluated for acoustic impulse responses denoising, where the polynomial matrices are used to model acoustic impulse responses (generated by a room image model, and recorded in real rooms). In both cases, white Gaussian noise with zero mean and unit variance is added to the data.

4.1. Experimental setup and data generation

4.1.1. Synthetically generated polynomial matrices

First, we show experiments on synthetically generated data as follows. We generate a random scalar matrix $\underline{\mathbf{D}}$ with uniformly distributed entries, which is then used as the coefficient matrix for the polynomial matrix $\boldsymbol{D}(z)$, where each column of $\underline{\mathbf{D}}$ is normalized to unity norm. The size of $\underline{\mathbf{D}}$ generated is 50×100. Then, $\underline{\mathbf{Y}}$ is generated by the linear combination of different columns in $\underline{\mathbf{D}}$. Finally, the polynomial matrices $\boldsymbol{Y}(z)$ and $\boldsymbol{D}(z)$ are generated by splitting their coefficient matrices according to Eqs. (11) and (12). Here, the training data we generated are 10×2000 polynomial matrices with 5 time lags.

4.1.2. Simulated and real acoustic impulse responses

The second type of data tested contains acoustic impulse responses, described by polynomial matrices. Two types of acoustic impulse responses are tested, respectively, those generated by a room image model [48], and the real acoustic impulse responses taken from [49].

By using the image model, the acoustic impulse responses are generated in a $20 \times 20 \times 3$ m^3 room (a simulated large hall). The reverberation time is set to be 900 ms, and the sampling frequency is 16 kHz, so that the number of time lags for each impulse response is 14,400. We generated 1000 acoustic impulse responses as the training set. Polynomial matrices are used to model the acoustic impulse response signals. Each acoustic impulse response is split into several segments with the same length, thereby each segment can be seen as an FIR filter which is modeled by a polynomial with a certain number of lags. Note that, once the length of each polynomial (FIR) is given, the number of polynomials can be calculated according to the number of acoustic signals and the length of each acoustic signal. These polynomial elements can be used to construct a polynomial matrix, whose dimensions are determined according to the length of the signals and the number of time lags specified in each polynomial element.

For the real data, we take 840 real impulse responses from the database [49] as the training signals, where the length of each impulse response is 192,000 samples. Each element of the polynomial matrix is designed to have 40 lags. Hence, each impulse response signal can be modeled by 4800 polynomial elements. Therefore, the acoustic signals in the training set are designed as a $20 \times 201,600$ polynomial matrix with 40 lags for each element.

4.1.3. Parameter selection

Assuming the dimension of the polynomial dictionary $\boldsymbol{D}(z)$ is $n \times K$ with L lags, $\boldsymbol{D}(z)$ needs to be overcomplete, that is $n \ll K$. Moreover, as the proposed dictionary learning model (6) can be expressed as the polynomial of matrices model (10), which means the new dictionary $\underline{\mathbf{D}} \in \mathbb{R}^{nL \times K}$ also needs to be overcomplete, which is $nL \ll K$.

As in conventional dictionary learning methods [1,9,50], it is difficult to find theoretically optimal parameters, therefore the parameters used in our polynomial algorithms were set empirically, according to extensive experimental tests. We also carried out some experiments to understand the impact of some important parameters on the performance of the proposed methods, such as the iteration numbers and sparsity in the polynomial dictionary learning process. In the denoising application, we also evaluated the performance of the algorithms for modeling the acoustic impulses using polynomial matrices with different lags, and the polynomial dictionaries with different sizes, which will be discussed in detail later.

4.1.4. Performance metrics

The reconstruction error between the original polynomial matrix $\boldsymbol{Y}(z)$ and the reconstructed polynomial matrix $\hat{\boldsymbol{Y}}(z)$ is used as the performance metric, which is defined as

$$R_{err} = \frac{\|\boldsymbol{Y}(z) - \hat{\boldsymbol{Y}}(z)\|_F^2}{\|\boldsymbol{Y}(z)\|_F^2}. \tag{28}$$

4.2. Experimental results and analysis

The proposed methods are tested on different noise levels, different sparsity levels, different sizes of dictionaries, and different time lags used in the polynomial dictionaries.

4.2.1. Experiments on synthetically generated data

First, we test the convergence of the proposed polynomial dictionary learning methods during the dictionary training procedure. The proposed extended K-SVD, extended MOD and PMOD algorithms are used to train the dictionaries. The size of the dictionaries is set to be identical, which is 10×100 with 5 time lags. Different levels of sparsity are tested (i.e. 3, 5, and 7). The sparse representation coefficients for the reconstruction are found by using the conventional OMP algorithm. In total, 50 realizations are carried out, and for each realization, 200 iterations are tested. The reconstruction errors are calculated at each iteration.

Fig. 1 shows the average reconstruction errors changing at each iteration. From Fig. 1, we can see that both methods can converge within 200 iterations, and the extended K-SVD achieves more accurate polynomial matrix reconstruction results than the extended MOD and PMOD for all levels of sparsity tested. Note that, the PMOD algorithm gives nearly the same average reconstruction accuracy as the extended MOD at each iteration during the dictionary training process. This is reasonable, as both the PMOD and the extended MOD use the same method to calculate the sparse coefficients in the sparse coding stage, although the PMOD operates on the polynomial matrix dictionary directly in the dictionary update stage. Also note that, the proposed methods converge with less iterations when using a lower level of sparsity, this is because less sparse representation coefficients need to be found in the sparse coding stage.

Then, we perform another experiment to evaluate the performance of the proposed methods for recovering a signal (i.e. polynomial matrix) corrupted by noise at different levels. In this case, white Gaussian noise of zero mean and variance chosen to achieve different signal-to-noise ratios (SNRs) is added to the coefficient matrices of the polynomial matrix $\boldsymbol{Y}(z)$. Note that both the size of the input data $\boldsymbol{Y}(z)$ and the dictionary are the same as those in the previous experiments. The numbers of iteration is set to be the same, as 200. Here, the proposed extended K-SVD, extended MOD, and PMOD algorithms are used to learn the dictionaries, and these dictionaries are applied to recover the polynomial matrix from the noise corrupted version. For the extended K-SVD and extended

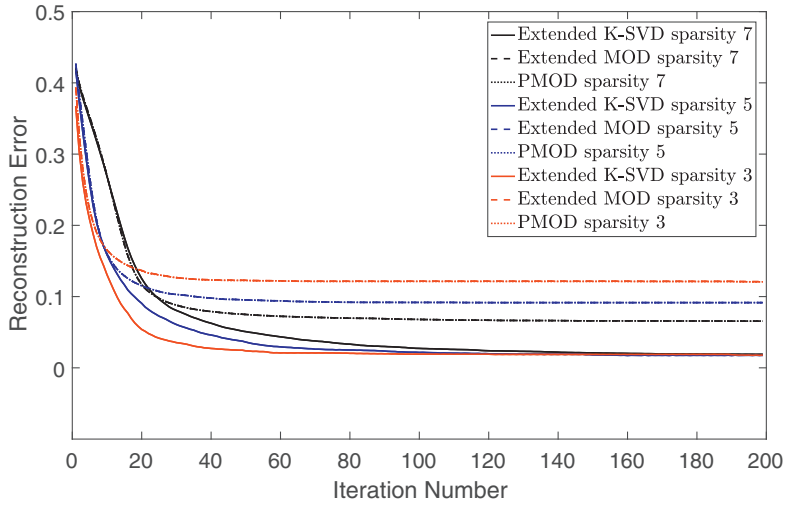

Fig. 1. Reconstruction error changes with the iteration number of the proposed methods during the training process for different levels of sparsity.

Table 1
Performance comparison in terms of the reconstruction error ($\times 10^{-2}$) for polynomial matrix reconstruction at different levels of noise, where different levels of sparsity are tested for denoising the noise corrupted polynomial matrix.

	Sparsity	Noise level (dB)					
		5	10	15	20	25	30
Extended K-SVD	3	16.76	10.29	5.36	2.78	1.47	0.79
	5	23.97	15.59	8.77	4.91	2.83	1.76
	7	28.42	18.88	10.83	6.17	3.73	2.53
Extended MOD	3	20.33	15.42	12.65	11.78	11.53	11.46
	5	25.29	17.26	11.26	8.49	7.49	7.16
	7	29.34	19.76	11.97	7.79	5.93	5.27
PMOD	3	21.25	16.57	14.01	13.22	13.00	12.94
	5	25.29	17.26	11.26	8.49	7.49	7.16
	7	29.34	19.76	11.97	7.79	5.93	5.27
PMOD + POMP	3	65.60	64.11	63.29	62.99	62.90	62.86
	5	62.96	60.58	59.22	58.71	58.55	58.49
	7	62.51	59.57	57.89	57.26	57.04	56.96

MOD, the OMP algorithm is used to calculate the sparse representation coefficients in the sparse coding stage. For the PMOD algorithm, both the OMP and the POMP algorithms are used and their performance is compared. The PMOD and POMP combination is denoted as PMOD + POMP. For each method, one dictionary is learned from the clean "signal" $\boldsymbol{Y}(z)$, and 20 realizations are carried out for the signal recovery by using OMP and POMP accordingly, where different levels of sparsity are tested (i.e. 3, 5, and 7) for training the dictionaries and sparsely representing the polynomial matrix.

Table 1 shows the results of the proposed methods for the noise corrupted polynomial matrix reconstruction. We can see that the extended K-SVD approach can obtain the best recovery accuracy for all levels of sparsity tested, and the extended MOD is slightly better than the PMOD method. POMP performs the worst for recovering the polynomial matrix with the dictionary learned by PMOD. It can be observed that, for the extended K-SVD, better recovery accuracy can be achieved with a lower level of sparsity enforced in reconstruction; whereas for PMOD + POMP, increasing the sparsity tends to give smaller reconstruction errors. For the extended MOD and PMOD, the reconstruction error is increased with the increase in the level of sparsity, for noise in the range of 5 dB–10 dB. In contrast, the reconstruction error becomes smaller with the increase of the sparsity level for noise in the range of 20 dB–30 dB. In comparison, the extended K-SVD method combined with OMP tends to give better accuracy for the reconstruction of noise corrupted polynomial matrix, when using a lower level of sparsity. However, for other methods tested, the denoising performance varies with the change of sparsity and noise level, and there is no clear trend on which sparsity level used will give absolutely better performance than other sparsity levels. Therefore, we only choose one sparsity in our following experiments, where the sparsity for both training dictionaries and reconstructing signal is set as 3. As observed from Fig. 1, all the proposed methods converge approximately after 80 iterations for dictionary training when the sparsity is set as 3, so that we set the maximum iteration number to be 80 in the following experiments.

4.2.2. Experiments on acoustic impulse responses

As the aim of our proposed methods is to process signals with time delays, we test the proposed methods for acoustic signal denoising, where the polynomial matrix is employed to model the acoustic impulse responses. The dictionaries learned by the proposed methods are used for the reconstruction of noise corrupted acoustic signals. In our experiments, the acoustic signals are modeled by polynomial matrices with different time lags, and dictionaries of different size are trained.

First, we conduct experiments on impulse response signals generated by a room image model [48] as mentioned in Section 4.1.2. 1000 clean impulse responses are used as the training set, and the noisy test signal is generated by adding noise to the clean acoustic impulse response. As the length of each impulse response is 14400, the test signal can be split into 720 segments, with the length of each segment as 20, so that the test signal can be modeled as a 10×72 polynomial matrix with 20 lags. In the same way, the training signals can be modeled by a $10 \times 72,000$ polynomial matrix with 20 lags. The size of the polynomial dictionaries is designed as 10×240, 10×320, and 10×400, respectively. The proposed methods are used to recover the noise corrupted impulse response, where different levels of noise are tested. For each method, one dictionary is learned from the clean training signals modeled polynomial matrix, and 20 realizations are carried out for recovering the noise corrupted signal at each noise level.

Table 2 shows the average reconstruction error of the proposed methods for the acoustic signal denoising at different noise levels. From the table, we can see that the proposed methods achieve similar results by using different size of training dictionaries, for low SNR levels (e.g., −10 dB and 0 dB). Dictionaries of smaller size offer better signal reconstruction performance, in contrast, those of

Table 2
Performance comparison in terms of reconstruction error ($\times 10^{-2}$) for room image impulse responses denoising at different noise levels, where dictionaries of different size are tested.

	Dictionary size	Noise level (dB)				
		−10	0	10	20	30
Extended K-SVD	10×240	237.55	36.47	17.34	17.20	17.19
	10×320	244.76	37.03	16.67	16.53	16.52
	10×400	249.71	37.19	15.62	15.45	15.43
Extended MOD	10×240	237.18	36.25	17.33	17.20	17.18
	10×320	243.82	36.55	16.13	15.99	15.98
	10×400	249.40	36.98	15.43	15.27	15.26
PMOD	10×240	237.18	36.25	17.33	17.20	17.18
	10×320	243.82	36.55	16.13	15.99	15.98
	10×400	248.92	37.05	15.23	15.07	15.05
PMOD + POMP	10×240	215.74	36.64	22.03	21.93	21.92
	10×320	223.30	38.24	23.10	23.01	23.00
	10×400	228.63	37.26	20.27	20.16	20.15

larger size tend to give higher recovery accuracy for higher SNR levels (e.g., 10 dB, 20 dB, and 30 dB).

The extended MOD and PMOD give almost the same average reconstruction error in the case when the size of dictionaries is 10×240 and 10×320, whereas the average reconstruction errors are different when the size of dictionaries is 10×400, and the PMOD performs better in this case. The reason why the performance is different when the dictionaries had size 10×400 is that the learned dictionaries have redundant atoms, which lead to multiple sparse representations for the signal reconstruction. When the size of the dictionary is larger than a certain number, some learned atoms may become redundant. The extended MOD can get slightly better recovery accuracy than the extended K-SVD. Interestingly, although the PMOD + POMP performs worse than the other three methods when the noise is added over ranges from 0 dB to 30 dB, it gives the best recovery accuracy when the SNR ratio is lower than 0 dB. It is especially worth noting that the performance of PMOD + POMP for acoustic signal denoising is better than that for denoising the polynomial matrices generated randomly in our last experiment, while the reconstruction error is similar to those obtained by the other three methods.

An illustration of the polynomial matrix modeled acoustic impulse response denoising is given in Fig. 2, where a 2×2 polynomial sub-matrix is randomly selected from the polynomial matrix and used to model the entire test acoustic room impulse response. Each element can be seen as a polynomial modeled FIR filter with 20 lags, which is a segment from the test acoustic signal. Fig. 2 shows the clean FIRs in the subplot (a), the corresponding noise added FIRs in (b) (5 dB noise), the recovered FIRs by the extended K-SVD, extended MOD, PMOD, and PMOD+POMP methods in the subplots (c), (d), (e), and (f), respectively. The size of the polynomial dictionaries used is the same, which is 10×320 with 20 lags. We can see from Fig. 2 that all the proposed methods can recover the noise corrupted FIRs in a certain level. Fig. 3 shows an example of the entire acoustic impulse response denoising by using the proposed extended K-SVD method. We can see that the proposed method can recover the noise corrupted signal very well.

Then, another experiment is carried out by using polynomial matrices with different lags to model the acoustic impulse responses. In order to find out how the impulse responses modeled polynomial matrix influences the performance of the proposed methods, the lags of the polynomial matrices used to model the acoustic impulse responses are set to be as 10, 20, and 30, respectively, so that the same 1000 training impulse responses as used in the previous experiment can be modeled as 10×144000 with 10 time lags, 10×72000 with 20 lags, and 10×48000 with 30 lags polynomial matrices, respectively. The size of the dictionaries in training is set to be 10×400 with 10, 20, and 30 lags, respectively. As the previous experiments have shown that the extended MOD and PMOD methods can obtain nearly the same performance during the dictionary training process for acoustic impulse response denoising, here, we only compare the performance of the extended K-SVD, PMOD and PMOD + POMP. For each time lag tested, one dictionary is trained by each method, and 20 realizations are carried out for each noise level. The average reconstruction errors are given in Table 3.

Table 3
Performance comparison in terms of the reconstruction error ($\times 10^{-2}$) for room image impulse responses denoising at different noise levels, where acoustic signals are modeled by polynomial matrices with different lags.

	Lags	Noise level (dB)				
		−10	0	10	20	30
Extended K-SVD	10	353.41	50.06	15.30	14.84	14.83
	20	248.46	37.22	15.70	15.54	15.53
	30	204.23	31.61	15.27	15.18	15.17
PMOD	10	352.48	50.05	15.09	14.64	14.63
	20	248.91	37.19	15.47	15.31	15.34
	30	203.51	31.30	15.20	15.11	15.10
PMOD + POMP	10	324.93	48.21	19.26	18.94	18.93
	20	228.27	37.40	20.43	20.32	20.34
	30	186.11	30.05	16.25	16.18	16.16

From Table 3, we can see that when applying the proposed methods, the acoustic signals modeled by polynomial matrices of greater time lags tend to give better recovery accuracy, for relatively lower SNR levels (i.e. −10 and 0 dB) for both cases. The best denoising result can be obtained by using the PMOD + POMP method. In contrast, the extended K-SVD and PMOD techniques can get better performance at higher SNR levels (i.e. 20, and 30 dB) for acoustic signals modeled by polynomial matrices with 10 lags, and the PMOD has the best denoising performance in this case.

In the above experiments, the polynomial dictionaries are all learned from clean signals. Here, we carry out additional experiments to evaluate the performance of the proposed methods for learning polynomial dictionaries from noise corrupted data. To this end, we add white Gaussian noise at different SNR levels (e.g. 10 dB, 20 dB, and 30 dB) to the same 1000 clean impulse responses used in the previous experiment. The polynomial matrix (containing the training signals) and the polynomial dictionary are obtained in a similar way to the case where the training signals are clean. More specifically, the noisy training signals are modeled by a $10 \times 48,000$ polynomial matrix with 30 lags, and the size of the dictionaries is 10×400 with 30 lags. The test signals are the same as those in the previous experiment. We run 20 realizations in which we train a dictionary for each noise level. The performance comparison of the extended K-SVD, PMOD, and PMOD+POMP methods is given in Table 4. As compared with the results in Table 3, we can see that the extended K-SVD and PMOD algorithms perform slightly better when using noise corrupted training signals, whereas the PMOD+POMP method performs worse than in the noise free case. This shows that the extended K-SVD and PMOD methods have better noise robustness as compared with the PMOD+POMP method. This is probably because the POMP uses the F-norm distance as the measurement for the selection of best-matching atoms, and the F-norm distance may not be as reliable as the inner product for similarity measure between the residual and the atoms for atom selection.

4.2.3. Experiments on real acoustic impulse responses

Finally, an experiment is carried out for real acoustic impulse response signal denoising. The POMP method is used to recover

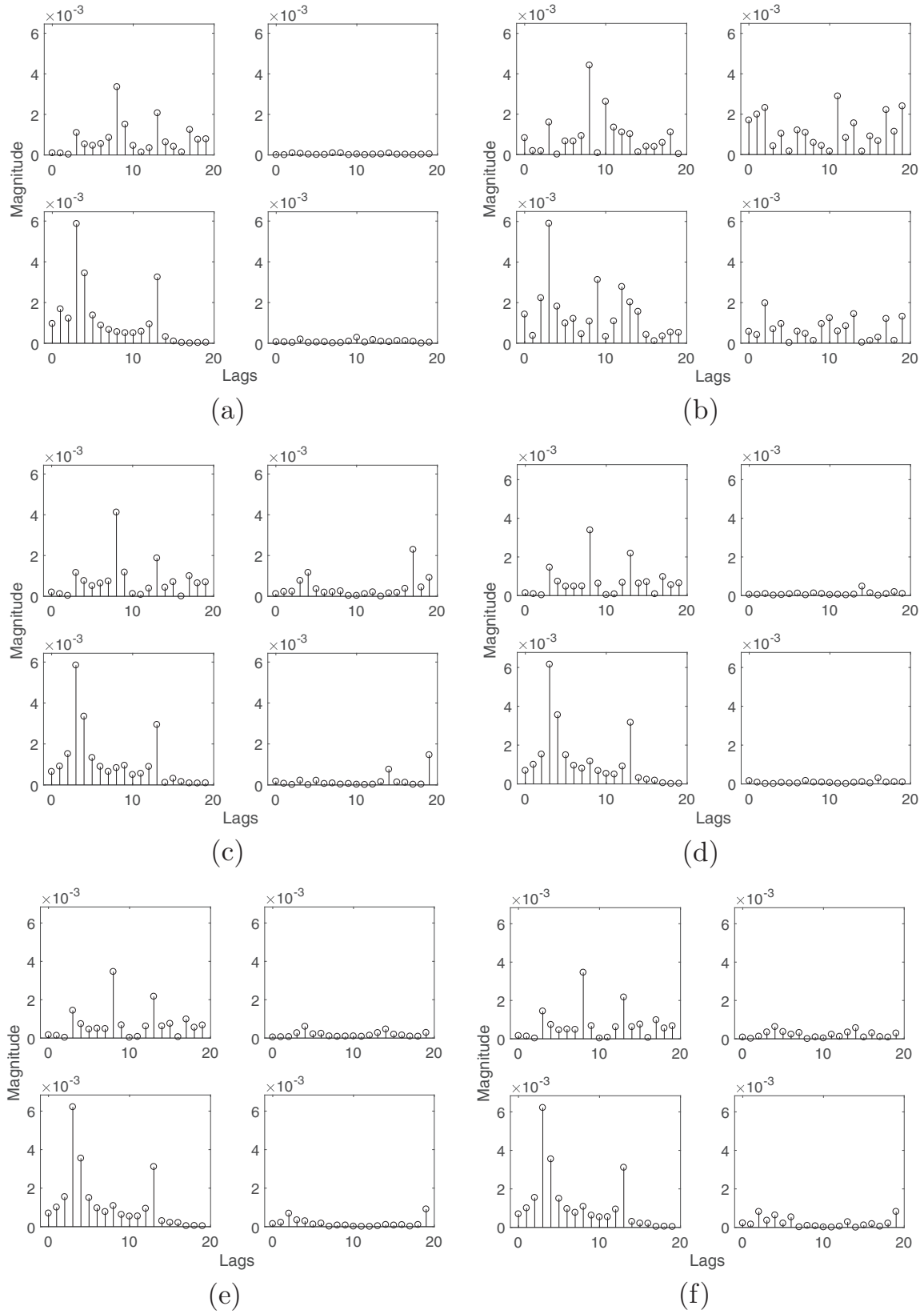

Fig. 2. Illustration of the proposed methods for acoustic impulse response denoising, where a polynomial matrix is used to model the test room impulse response, a sub-matrix with four polynomials is randomly selected from the polynomial matrices, where each polynomial is an FIR filter denoting a segment of the test room impulse response. (a) Clean FIRs; (b) Noisy FIRs; (c) Denoised FIRs by the extended K-SVD; (d) Denoised FIRs by the extended MOD; (e) Denoised FIRs by PMOD; (f) Denoised FIRs by PMOD + POMP.

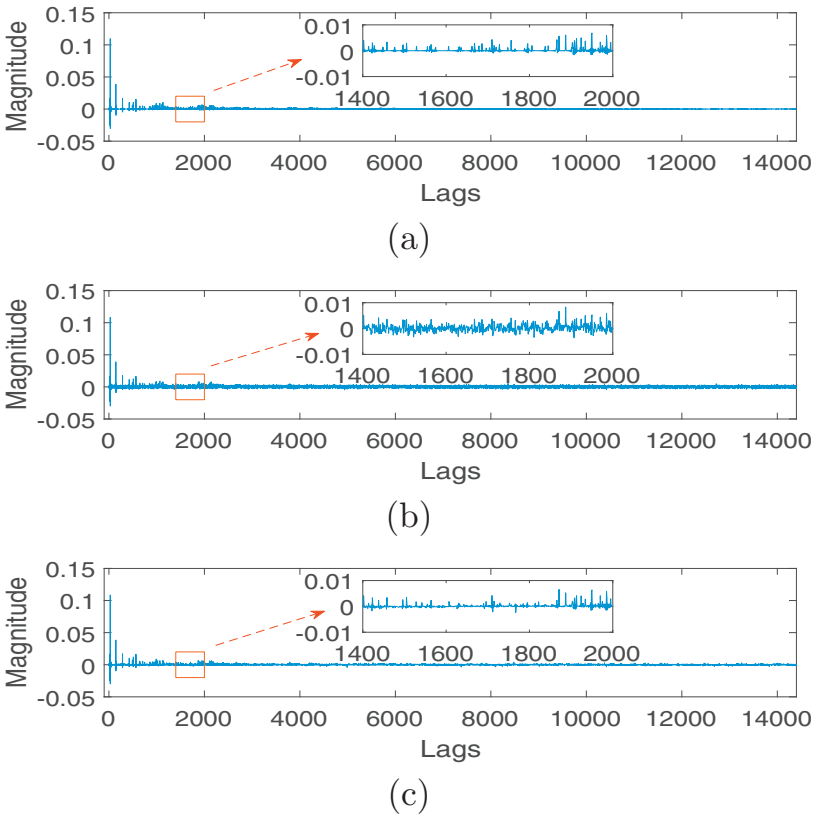

Fig. 3. An example of room image impulse response signal denoising, where the extended K-SVD is used. (a) The clean acoustic signal; (b) The noisy acoustic signal; and (c) The reconstructed acoustic signal.

Table 4
Performance comparison in terms of the reconstruction error ($\times 10^{-2}$) for room image impulse responses denoising at different noise levels, where the dictionaries are learned from training signals with different noise levels, and the size of the dictionaries is 10×400 with 30 lags.

	Training signal noise level(dB)	Test signal noise level (dB)				
		−10	0	10	20	30
Extended K-SVD	10	203.32	31.16	15.09	15.00	14.98
	20	202.63	31.16	15.12	15.03	15.01
	30	203.10	31.28	15.16	15.06	15.05
PMOD	10	203.00	31.39	14.94	14.84	14.83
	20	202.57	31.23	15.01	14.92	14.90
	30	202.61	31.16	14.85	14.75	14.74
PMOD + POMP	10	188.26	33.89	21.78	21.71	21.70
	20	186.82	33.27	20.83	20.76	20.75
	30	186.44	32.92	20.56	20.49	20.48

the noisy real acoustic impulse response, where the polynomial dictionary is learned by the PMOD. Here, the OMP is also used to reconstruct the impulse responses for comparison purpose. The test signal is corrupted by 5 dB noise.

As mentioned in Section 4.1.2, 840 real impulse responses are used as the training signals, which are modeled by a $20 \times 201,600$ polynomial matrix with 40 lags. The size of dictionary is set to be 20×1200 with 40 lags. Fig. 4 shows the clean signal in the subplot (a), its corresponding noisy signal in the subplot (b), and the reconstructed signals by OMP and POMP methods in the subplots (c) and (d), respectively. It can be observed that both reconstructed signals are similar to the clean test signal. The experiments show that our proposed methods can obtain fairly good performance for denoising real acoustic signals.

5. Conclusions

We introduced a polynomial dictionary learning technique to deal with signals with time lags, where the polynomial matrix was employed to model the signals. This provided a way for learning a dictionary for signals with time lags, such as acoustic impulse responses. Two types of polynomial dictionary learning methods were proposed based respectively on the polynomial of matrices model and the matrices of polynomial model. By using the polynomial of matrices model based dictionary learning method, any conventional dictionary learning methods can be used to represent the signals with time lags; whereas the matrices of polynomial dictionary learning model provided a potential way to deal with the polynomial dictionary matrix directly without having to explicitly

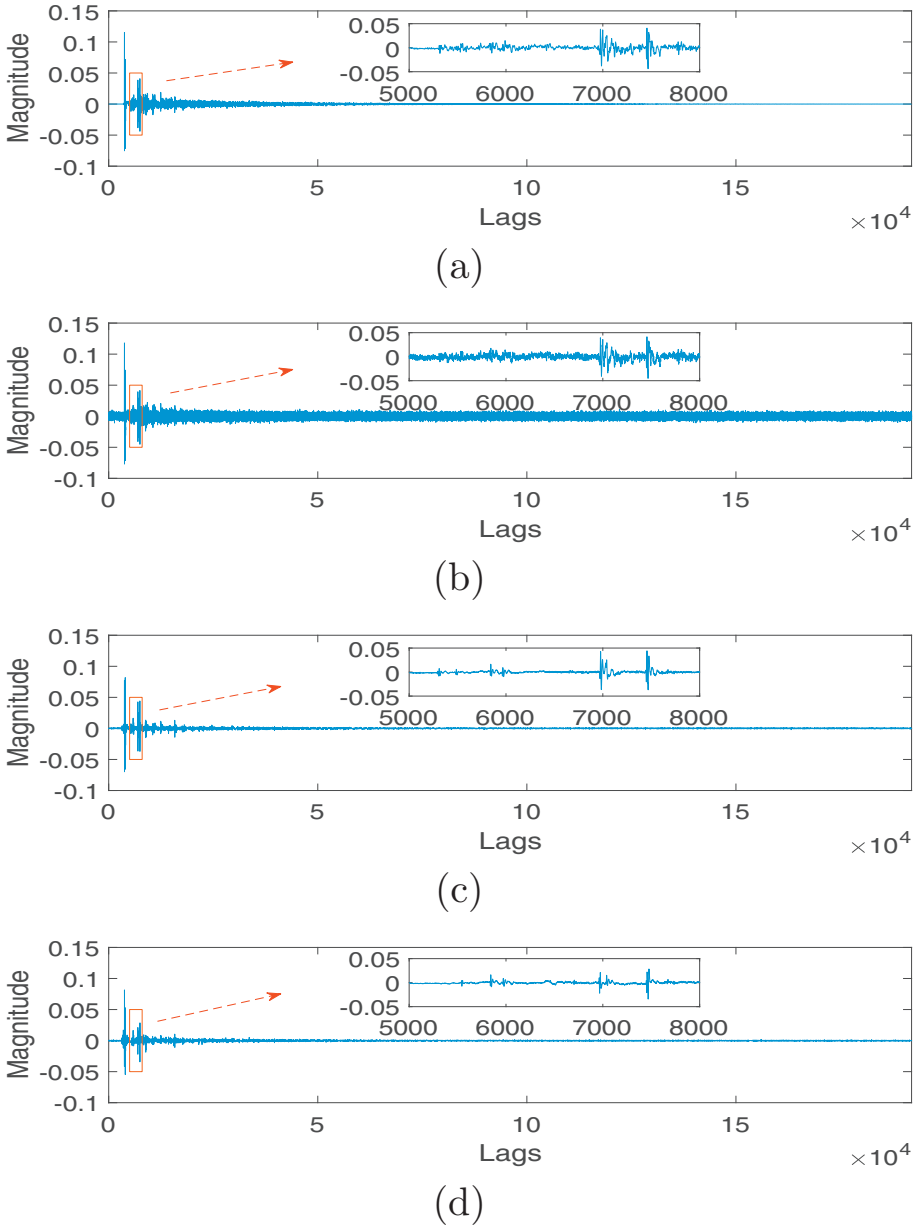

Fig. 4. Denoising of noisy real acoustic impulse response signal. (a) The clean acoustic signal; (b) The noisy acoustic signal (c) The reconstructed acoustic signal by the PMOD; (d) The reconstructed acoustic signal by PMOD + POMP.

access the polynomial coefficient matrices, where the sparse coefficient matrix was still a scalar matrix, rather than a polynomial matrix. As a byproduct, a polynomial OMP algorithm was also proposed. The experiments show that our proposed methods can be used to model signals with time lags, such as acoustic impulse responses, and to reconstruct such signals from noise corrupted samples. Moreover, the experiments also show that we can obtain better performance by carefully designing the polynomial matrix and choosing the size of dictionary according to the tasks at hand.

Acknowledgments

This work was conducted when J. Guan was visiting the Centre for Vision, Speech and Signal Processing (CVSSP) of the University of Surrey, UK, and partially supported by International Exchange and Cooperation Foundation of Shenzhen City, China (No. GJHZ20150312114149569). W. Wang and J. Chambers were supported in part by the Engineering and Physical Sciences Research Council (EPSRC) Grant Number EP/K014307 and the MOD University Defence Research Collaboration in Signal Processing. The authors thank the anonymous reviewers for their helpful suggestions.

References

[1] M. Elad, M. Aharon, Image denoising via sparse and redundant representations over learned dictionaries, IEEE Trans. Image Process. 15 (12) (2006) 3736–3745.
[2] M.G. Jafari, M.D. Plumbley, Fast dictionary learning for sparse representations of speech signals, IEEE J. Sel. Topics Signal Process. 5 (5) (2011) 1025–1031.
[3] M. Zibulevsky, B.A. Pearlmutter, Blind source separation by sparse decomposition in a signal dictionary, Neural Comput. 13 (4) (2001) 863–882.
[4] R. Gribonval, Sparse decomposition of stereo signals with matching pursuit and application to blind separation of more than two sources from a stereo mixture, in: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 3, IEEE, 2002, pp. III–3057.
[5] O. Yilmaz, S. Rickard, Blind separation of speech mixtures via time-frequency masking, IEEE Trans. Signal Process. 52 (7) (2004) 1830–1847.

[6] T. Xu, W. Wang, Methods for learning adaptive dictionary in underdetermined speech separation, in: Proceedings of IEEE International Workshop on Machine Learning for Signal Processing (MLSP), IEEE, 2011, pp. 1–6.
[7] S. Wang, L. Zhang, Y. Liang, Q. Pan, Semi-coupled dictionary learning with applications to image super-resolution and photo-sketch synthesis, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2012, pp. 2216–2223.
[8] K. Engan, S.O. Aase, J.H. Husoy, Method of optimal directions for frame design, in: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 5, IEEE, 1999, pp. 2443–2446.
[9] M. Aharon, M. Elad, A. Bruckstein, K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation, IEEE Trans. Signal Process. 54 (11) (2006) 4311–4322.
[10] W. Dai, T. Xu, W. Wang, Simultaneous codeword optimization (SimCO) for dictionary update and learning, IEEE Trans. Signal Process. 60 (12) (2012) 6340–6353.
[11] S.G. Mallat, Z. Zhang, Matching pursuits with time-frequency dictionaries, IEEE Trans. Signal Process. 41 (12) (1993) 3397–3415.
[12] R. Tibshirani, Regression shrinkage and selection via the LASSO, J. R. Stat. Soc. Ser. B (Methodological) (1996) 267–288.
[13] I.F. Gorodnitsky, B.D. Rao, Sparse signal reconstruction from limited data using focuss: a re-weighted minimum norm algorithm, IEEE Trans. Signal Process. 45 (3) (1997) 600–616.
[14] S. Chen, S.A. Billings, W. Luo, Orthogonal least squares methods and their application to non-linear system identification, Int. J. Control 50 (5) (1989) 1873–1896.
[15] B.K. Natarajan, Sparse approximate solutions to linear systems, SIAM J. Comput. 24 (2) (1995) 227–234.
[16] J. Wang, P. Li, Recovery of sparse signals using multiple orthogonal least squares, IEEE Trans. Signal Process. 65 (8) (2017) 2049–2062.
[17] Y.C. Pati, R. Rezaiifar, P. Krishnaprasad, Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition, in: Proceedings of the 27th Asilomar Conference on Signals, Systems and Computers, IEEE, 1993, pp. 40–44.
[18] A.M. Bruckstein, D.L. Donoho, M. Elad, From sparse solutions of systems of equations to sparse modeling of signals and images, SIAM Rev. 51 (1) (2009) 34–81.
[19] D.L. Donoho, Y. Tsaig, I. Drori, J.-L. Starck, Sparse solution of underdetermined systems of linear equations by stagewise orthogonal matching pursuit, IEEE Trans. Inf. Theory 58 (2) (2012) 1094–1121.
[20] D. Needell, R. Vershynin, Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit, Found. Comput. Math. 9 (3) (2009) 317–334.
[21] T.T. Cai, L. Wang, Orthogonal matching pursuit for sparse signal recovery with noise, IEEE Trans. Inf.Theory 57 (7) (2011) 4680–4688.
[22] S.K. Sahoo, A. Makur, Signal recovery from random measurements via extended orthogonal matching pursuit., IEEE Trans. Signal Process. 63 (10) (2015) 2572–2581.
[23] N.B. Karahanoglu, H. Erdogan, Improving A*OMP: theoretical and empirical analyses with a novel dynamic cost model, Signal Process. 118 (2016) 62–74.
[24] J. Wen, Z. Zhou, J. Wang, X. Tang, Q. Mo, A sharp condition for exact support recovery of sparse signals with orthogonal matching pursuit, IEEE Trans. Signal Process. 65 (6) (2016) 1370–1382.
[25] J. Wang, S. Kwon, P. Li, B. Shim, Recovery of sparse signals via generalized orthogonal matching pursuit: a new analysis, IEEE Trans. Signal Process. 64 (4) (2016) 1076–1089.
[26] J. Wang, Support recovery with orthogonal matching pursuit in the presence of noise, IEEE Trans. Signal Process. 63 (21) (2015) 5868–5877.
[27] C. Herzet, A. Drémeau, C. Soussen, Relaxed recovery conditions for OMP/OLS by exploiting both coherence and decay, IEEE Trans. Inf. Theory 62 (1) (2016) 459–470.
[28] A. Cohen, W. Dahmen, R. DeVore, Orthogonal matching pursuit under the restricted isometry property, Constr. Approx. 45 (1) (2017) 113–127.
[29] T. Saramäki, R. Bregovic, Multirate systems and filter banks, in: Multirate Systems: Design and Applications, 2, 2001, pp. 27–85.
[30] J.A. Foster, J.G. McWhirter, M.R. Davies, J.A. Chambers, An algorithm for calculating the QR and singular value decompositions of polynomial matrices, IEEE Trans. Signal Process. 58 (3) (2010) 1263–1274.
[31] L. Rota, P. Comon, S. Icart, Blind MIMO paraunitary equalizer, in: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 4, IEEE, 2003, pp. 285–288.
[32] R. Brandt, M. Bengtsson, Wideband mimo channel diagonalization in the time domain, in: Proceedings of IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, IEEE, 2011, pp. 1958–1962.
[33] J. Foster, J. McWhirter, S. Lambotharan, I. Proudler, M. Davies, J. Chambers, Polynomial matrix QR decomposition for the decoding of frequency selective multiple-input multiple-output communication channels, IET Signal Process. 6 (7) (2012) 704–712.
[34] J. Guan, J. Dong, X. Wang, W. Wang, A polynomial dictionary learning method for acoustic impulse response modeling, in: Latent Variable Analysis and Signal Separation, in: Lecture Notes in Computer Science, 9237, Springer, 2015, pp. 211–218.
[35] J.G. McWhirter, P.D. Baxter, T. Cooper, S. Redif, J. Foster, An EVD algorithm for para-Hermitian polynomial matrices, IEEE Trans. Signal Process. 55 (5) (2007) 2158–2169, doi:10.1109/TSP.2007.893222.
[36] Z. Wang, J.G. McWhirter, S. Weiss, Multichannel spectral factorization algorithm using polynomial matrix eigenvalue decomposition, in: Proceedings of 49th Asilomar Conference on Signals, Systems and Computers, 2015, pp. 1714–1718, doi:10.1109/ACSSC.2015.7421442.
[37] A. Ahrens, A. Sandmann, S. Lochmann, Z. Wang, Decomposition of optical MIMO systems using polynomial matrix factorization, in: Proceedings of 2nd IET International Conference on Intelligent Signal Processing 2015 (ISP), 2015, pp. 1–6, doi:10.1049/cp.2015.1758.
[38] E.J. Candes, T. Tao, Decoding by linear programming, IEEE Trans. Inf. Theory 51 (12) (2005) 4203–4215.
[39] E.J. Candès, M.B. Wakin, An introduction to compressive sampling, IEEE Signal Process. Mag. 25 (2) (2008) 21–30.
[40] M.A. Davenport, M.B. Wakin, Analysis of orthogonal matching pursuit using the restricted isometry property, IEEE Trans. Inf. Theory 56 (9) (2010) 4395–4401.
[41] J. Wang, B. Shim, On the recovery limit of sparse signals using orthogonal matching pursuit, IEEE Trans. Signal Process. 60 (9) (2012) 4973–4976.
[42] M.M. Abo-Zahhad, A.I. Hussein, A.M. Mohamed, Compressive sensing algorithms for signal processing applications: a survey, Int. J. Commun., Netw. Syst. Sci. 8 (06) (2015) 197–216.
[43] A. Agarwal, A. Anandkumar, P. Jain, P. Netrapalli, R. Tandon, Learning sparsely used overcomplete dictionaries, in: Proceedings of Conference on Learning Theory, 2014, pp. 123–137.
[44] S. Arora, R. Ge, A. Moitra, New algorithms for learning incoherent and overcomplete dictionaries, in: Proceedings of Conference on Learning Theory, 2014, pp. 779–806.
[45] S. Barman, A. Bhattacharyya, S. Ghoshal, The dictionary testing problem, arXiv preprint arXiv:1608.01275 (2016).
[46] A. Agarwal, A. Anandkumar, P. Netrapalli, Exact recovery of sparsely used overcomplete dictionaries, Stat 1050 (2013) 8–39.
[47] A. Agarwal, A. Anandkumar, P. Jain, P. Netrapalli, Learning sparsely used overcomplete dictionaries via alternating minimization, SIAM J. Optim. 26 (2016) 2775–2799.
[48] J.B. Allen, D.A. Berkley, Image method for efficiently simulating small-room acoustics, J. Acoust. Soc. Am. 65 (4) (1979) 943–950.
[49] R. Stewart, M.B. Sandler, Database of omnidirectional and b-format impulse responses, in: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2010, pp. 165–168.
[50] J. Dong, W. Wang, W. Dai, M.D. Plumbley, Z.-F. Han, J. Chambers, Analysis SimCO algorithms for sparse analysis model based dictionary learning, IEEE Tran. Signal Process. 64 (2) (2016) 417–431.