POLYNOMIAL NETWORKS REPRESENTATION OF NONLINEAR MIXTURES WITH APPLICATION IN UNDERDETERMINED BLIND SOURCE SEPARATION

Lu Wang[1] *and Tomoaki Ohtsuki*[2]

[1]Graduate School of Science and Technology, Keio University, Kanagawa, Japan
[2]Department of Information and Computer Science, Keio University, Kanagawa, Japan

ABSTRACT

Similar to the deep architectures, a novel multi-layer architecture is used to extend the linear blind source separation (BSS) method to the nonlinear case in this paper. The approach approximates the nonlinearities based on a polynomial network, where the layer of our network begins with the polynomial of degree 1, up to build an output layer that can represent data with a small bias by a good approximate basis. Relying on several transformations of the input data, with higher-level representation from lower-level ones, the networks are to fulfill a mapping implicitly to the high-dimensional space. Once the polynomial networks are built, the coefficient matrix can be estimated by solving an l_1-regularization on the coding coefficient vector. The experiment shows that the proposed approach exhibits a higher separation accuracy than the comparison algorithms.

Index Terms— Underdetermined BSS, vanishing polynomial networks, nonlinear mixture, sparse coding, time-frequency representation.

1. INTRODUCTION

Recognizing multiple talkers from multiple observations (or mixtures) received by a set of sensors is the task of source separation. The problem is referred to as the underdetermined blind source separation (UBSS) when the number of sensors is less than that of sources [1, 2]. However, without any further constraint, these approaches can not be applied to the nonlinear BSS problem. It always exists infinite solutions without the constriant of nonlinear functions [3].

Various attempts [4, 5, 6] exploiting on some further constraints have been involved, such as representing the distortion based on some unknown parameters [7], extracting the nonlinearity where the mixing function can be approximated by the prior neural network [8], restricting the target function to be smoothness [9], and mapping the nonlinear problem into some feature space [10, 11, 12]. Despite such progress, there are still many important open problems and unexplored areas. For instance, the captured nonlinear features are in fact growing at an enormous rate.

In this paper, we propose to extend the UBSS method [2] to the nonlinear case. The derivation of our algorithm is inspired by ideas from [13], used it for creating a novel network architecture. The approach attempts to generate a polynomial network, which provides a good approximate basis for the values attained by a set of mapping functions. Similar to the principle in deep learning, the layers of our network start with polynomials of degree 1, which has the large bias attained by this simple approximation network. To create the higher level representations of the data to decrease the bias, we next make the network deeper and deeper. Each enhancement of the degree makes the layer deeper into our network. Once the deep polynomial networks are built that can approximate the nonlinearity of the mixing function. Then, we can fulfill a simple linear separation algorithm on top of this output. Thus, our work presents the advantages offered by both, the deep architectures formed by a polynomial network, and the coefficient matrix derives by sparse coding in the underdetermined scenario. In particular, our network can search the number of layers that makes deeper until the candidate dataset becomes empty.

Section 2 reviews the nonlinear BSS, and formulates our problem. Section 3 describes our proposed separation algorithm. Section 4 shows the experimental setup and results. Finally, the conclusions are given in Section 5.

2. NOTATIONS AND PROBLEM SETUP

The general definition of nonlinear BSS addressed in this paper, is given as the following. Given a set of observed data $\mathcal{X} = \{\mathbf{x}(1), \cdots, \mathbf{x}(T)\} \in \mathbb{R}^n$ that are assumed to be generated from a nonlinear, instantaneous and invertible function as

$$\mathbf{x}(t) = \mathcal{F}(\mathbf{s}(t)), \quad t = 1, \cdots, T, \tag{1}$$

where $\mathbf{s}(t) = [s(1), \cdots, s(T)] \in \mathbb{R}^m$ represent the original sources, and the function $\mathcal{F}$ denotes a transformation from $\mathbb{R}^m$ to $\mathbb{R}^n$. To make the nonlinear problem linearly separable, the idea is to generate a polynomial network, which provides a good approximate basis for the values attained by a set of mapping functions. A coefficient matrix that induces an appropriate mapping $\Phi : \mathbb{R}^n \to \mathcal{H}$ is introduced to transform the input data in the high-dimensional space.

We next define a multivariate polynomial that allows us to build a polynomial network so that to capture the nonlinearity or distortion caused by a mixing function. The multivariate polynomial [14] performs a mapping $\Phi : \mathbb{R}^n \to \mathbb{R}$ associated with $\mathbf{x} \in \mathbb{R}^n$, as the form

$$\Phi(\mathbf{x}) = \sum_{i=0}^{\Delta} \sum_{\boldsymbol{\alpha}^{(i)}} \omega_{\boldsymbol{\alpha}^{(i)}} \prod_{j=1}^{n} x_j^{\alpha_j^{(i)}}, \tag{2}$$

where Δ is the degree of the polynomial, and $\boldsymbol{\alpha}^{(i)}$ ranges over all n-dimensional vectors of positive integers, such that $\sum_{j=1}^{n} \alpha_j^{(i)} = i$. $\omega \in \mathbb{R}$ is the coefficient matrix.

Problem 1. Given a set of data $\{\mathbf{x}(1), \cdots, \mathbf{x}(T)\} \in \mathbb{R}^n$. The problem is to learn a polynomial network formed by a set of basises $\{\Phi_i(\mathbf{x}(1)), \cdots, \Phi_i(\mathbf{x}(T))\}_{i=1}^{k}$ of k polynomials with a coefficient matrix. For the coefficient matrix with the column vectors $[\mathbf{W}_1, \cdots, \mathbf{W}_n]^\top$, the demixing process can be defined by

$$\hat{s}_i(t) = \sum_{j=1}^{k} W_{ij} \Phi_j(\mathbf{x}(t)), \tag{3}$$

for all $i = 1, \cdots, k$, where the symbol $[\cdot]^\top$ denotes the transpose operator. □

Problem 1 implies that if we generate some basises, using the deep architectures, we can represent the varieties of the nonlinearity. Then we can fulfill a simple linear separation algorithm on top of these outputs.

3. POLYNOMIAL NETWORKS REPRESENTATION BASED NONLINEAR SEPARATION APPROACH

In this paper, we introduce a polynomial network that provides a good approximate basis for the values attained by a mapping function. Then using a linear separation method, we can estimate the coefficient matrix on top of network output.

3.1. Constructing the Polynomial Networks

The polynomial of degree 1, denoted as $\Phi^{(1)}$ is defined by a vector $\{\mathbf{x}_i(t)\}_{i=1}^{n}$ with the coefficient $\boldsymbol{\beta} \in \mathbb{R}^{n+1}$, such that

$$\Phi^{(1)}(\mathbf{x}(t)) = \beta_0 + \sum_{i=1}^{n} \beta_i x_i(t) = \sum_{i=0}^{n} \beta_i \rho_i(\mathbf{x}(t)), \tag{4}$$

where $x_i(t)$ is the i-th channel of the observations $\mathbf{x}(t)$ and we use $\rho_i(\mathbf{x}(t)) = x_i(t)$ for convenience. For each time t, considering all data points from $t = 1, \cdots, T$, we have

$$\boldsymbol{\Phi}^{(1)}(\mathcal{S}) = \begin{bmatrix} \sum_{i=0}^{n} \beta_i \rho_i(\mathbf{x}(1)) \\ \vdots \\ \sum_{i=0}^{n} \beta_i \rho_i(\mathbf{x}(T)) \end{bmatrix} = \sum_{i=0}^{n} \beta_i \boldsymbol{\rho}_i(\mathcal{S}), \tag{5}$$

where $\boldsymbol{\rho}_i(\mathcal{S}) = [\rho_i(\mathbf{x}(1)), \cdots, \rho_i(\mathbf{x}(T))]^\top$.

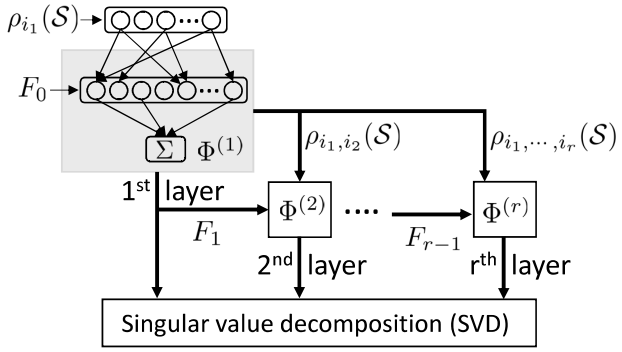

Fig. 1: Schematic diagram of the polynomial networks, with polynomials until degree r.

Theorem 1. The polynomial $\boldsymbol{\Phi}^{(1)}(\mathcal{S})$ vanishes on dataset $\mathcal{S}$ if and only if $\boldsymbol{\Phi}^{(1)}(\mathcal{S}) \leq \boldsymbol{\epsilon}_{T\times 1}$, which requires the vector $\boldsymbol{\beta}$ would be in the null space of the $T \times (n+1)$ matrix $\mathbf{A}_1 = [\boldsymbol{\rho}_0(\mathcal{S}), \cdots, \boldsymbol{\rho}_n(\mathcal{S})]$ as

$$\mathbf{A}_1 \boldsymbol{\beta} = [\boldsymbol{\rho}_0(\mathcal{S}), \cdots, \boldsymbol{\rho}_n(\mathcal{S})]\boldsymbol{\beta} \leq \boldsymbol{\epsilon}_{T\times 1}, \tag{6}$$

where the tolerated value ϵ enables us to relax the effect of noise, which is a vector with the same element closed to 0. □

In this paper, we prefer to search all vanishing polynomials[1] for dataset $\mathcal{S} = \{\mathbf{x}(t)\}_{t=1}^{T} \in \mathbb{R}^n$, so that to build a deep polynomial network to approximate the varieties of the nonlinearity. These polynomials do not achieve the inversion of nonlinear mixing directly, but provide some good approximate for the values attained by the nonlinear mixing.

The process is illustrated in Fig. 1. First, the 1^{st} layer is constructed by the basis that spans all values attained by polynomials of degree 1. Using the Gram-Schmidt algorithm, we can generate some orthogonal polynomials which require

$$\boldsymbol{\gamma}_i^{(1)}(\mathcal{S}) = \boldsymbol{\rho}_i(\mathcal{S}) - \sum_{\boldsymbol{\eta} \in F_0} \langle \boldsymbol{\rho}_i(\mathcal{S}), \boldsymbol{\eta}(\mathcal{S}) \rangle \boldsymbol{\eta}(\mathcal{S}), \tag{7}$$

where the input is dataset $C_1 = \{\boldsymbol{\rho}_i(\mathcal{S})\}$ for all $i = 1, \cdots, n$ and we initialize $F_0 = \{\boldsymbol{\eta}(\mathcal{S}) : \boldsymbol{\eta}(\mathcal{S}) = \boldsymbol{\rho}_0(\mathcal{S})/\|\boldsymbol{\rho}_0(\mathcal{S})\|\}$ and $V_0 = \emptyset$, respectively. If a proper combination can be searched, which lead to $\boldsymbol{\gamma}_i^{(1)}(\mathcal{S})$ almost vanishing on the data $\mathcal{S}$, we update $V_1 \leftarrow V_1 \bigcup \{\boldsymbol{\gamma}_i^{(1)}(\mathcal{S})\}$. Otherwise, $F_1 \leftarrow F_1 \bigcup \{\boldsymbol{\gamma}_i^{(1)}(\mathcal{S})/\|\boldsymbol{\gamma}_i^{(1)}(\mathcal{S})\|\}$ is updated. Thus, V_1 and F_1 are referred to as the sets of vanishing polynomial and non-vanishing polynomial in degree 1, respectively. At the end of this process, F_1 contains a set of non-vanishing linear combinations which will be used for generating the 2^{nd} layer.

Therefore, the polynomial network starts with polynomials of degree 1, which have the large bias attained by this

[1]The function is referred to as vanishing polynomial of $\mathcal{S}$ iff $\|\Phi(\mathbf{x})\| \leq \epsilon$ for $\forall \mathbf{x} \in \mathcal{S}$, where ϵ is tolerate value and $\|\cdot\|$ denotes the Euclidean norm.

simple approximate network. To create the higher level representations of the data to decrease the bias, we next make the network deeper and deeper. Each enhancement of the degree makes the layer deeper into our network. In particular, our network can search the number of layers that are added until the non-vanishing set F becomes empty.

3.1.1. Generating the Polynomials of Higher Degree

To exploit the layer attained by a higher level representation, the above progress continues to generate the polynomials of higher degree. For a polynomial of degree r, the set $C_r = \{\boldsymbol{\rho}_{i_1,\cdots,i_r}(\mathcal{S})\}_{i_1,\cdots,i_r=1}^{n}$ is formed by $\boldsymbol{\rho}_{i_1,\cdots,i_r}(\mathcal{S}) = [\rho_{i_1,\cdots,i_r}(\mathbf{x}(1)),\cdots,\rho_{i_1,\cdots,i_r}(\mathbf{x}(T))]^\top$, where $\rho_{i_1,\cdots,i_r}(\mathbf{x}(t)) = x_{i_1}(t)x_{i_2}(t)\cdots x_{i_r}(t)$. To obtain the orthogonal polynomial of degree r, we have

$$\boldsymbol{\gamma}_i^{(r)}(\mathcal{S}) = \boldsymbol{\rho}_{i_1,\cdots,i_r}(\mathcal{S}) - \sum_{\boldsymbol{\eta}^{(r-1)}\in F_{r-1}} \langle \boldsymbol{\rho}_{i_1,\cdots,i_r}(\mathcal{S}), \boldsymbol{\eta}^{(r-1)}(\mathcal{S})\rangle \boldsymbol{\eta}^{(r-1)}(\mathcal{S}), \tag{8}$$

where $F_{r-1} = \{\boldsymbol{\eta}_j^{(r-1)} = \frac{\boldsymbol{\rho}_j^{(r-1)}}{\|\boldsymbol{\rho}_j^{(r-1)}\|}\}$ for all $j = 1,\cdots,|F_{r-1}|$, and $|F_{r-1}|$ denotes the number of elements in the set F_{r-1}.

Up to the creation of the output layer, (8) can be batched by using singular value decomposition (SVD). Given a matrix $\mathbf{A}_r$ formed by $\mathbf{A}_r = [\boldsymbol{\gamma}_1^{(r)}(\mathcal{S}),\cdots,\boldsymbol{\gamma}_{|F_{r-1}|}^{(r)}(\mathcal{S})]$. By using SVD, the matrix $\mathbf{A}_r \in \mathbb{R}^{T\times|F_{r-1}|}$ can be decomposed as $\mathbf{A}_r = \mathbf{L}\mathbf{D}\mathbf{U}^\top$. Using a simple matrix operation, we have

$$\mathbf{A}_r\mathbf{U} = \left[\boldsymbol{\gamma}_1^{(r)}(\mathcal{S}),\cdots,\boldsymbol{\gamma}_{|F_t|}^{(r)}(\mathcal{S})\right]\mathbf{U} = \mathbf{L}\mathbf{D}, \tag{9}$$

where $\mathbf{L} = [\boldsymbol{l}_1,\cdots,\boldsymbol{l}_T]$ of $\boldsymbol{l}_i \in \mathbb{R}^T$. The dual representation is given by

$$\boldsymbol{\Phi}_i^{(r)}(\mathcal{S}) = \sum_{j=1}^{|F_{r-1}|} U_{j,i}\boldsymbol{\gamma}_j^{(r)}(\mathcal{S}) = \sum_{j=1}^{T} D_{j,i}\boldsymbol{l}_j = D_{i,i}\boldsymbol{l}_i, \tag{10}$$

where $i = 1,\cdots,|F_{r-1}|$. Thus, $\boldsymbol{\Phi}_i^{(r)}(\mathcal{S})$ is denoted as a vanishing polynomial, we only need to check whether each element of matrix $D_{i,i}$ is less or equal to the tolerate value ϵ.

3.2. Coefficient Matrix Identification

Once the basis $\{\Phi_i(\mathbf{x}(1)),\cdots,\Phi_i(\mathbf{x}(T))\}_{i=1}^{k}$ was generated, the nonlinear problem can be linearly separable in (3). We use the UBSS method [1] in these high-dimensional spaces spanned by the basis, we can obtain the coefficient matrix $\mathbf{W}$. Using discrete-time short-time Fourier transform (STFT), the linear BSS (3) can be transformed into the time-frequency (TF) domain

$$\mathcal{D}_{\boldsymbol{\Phi}}(t,\omega) = \tilde{\mathbf{W}}\hat{\mathcal{D}}_{\mathbf{s}_i}(t,\omega), \tag{11}$$

where $\mathcal{D}_{\boldsymbol{\Phi}}(t,\omega) = [\mathcal{D}_{\Phi_1}(t,\omega),\cdots,\mathcal{D}_{\Phi_n}(t,\omega)]^\top$ is the mixture signals in the TF domain and $\hat{\mathcal{D}}_{\mathbf{s}}(t,\omega) = [\hat{\mathcal{D}}_{s_1}(t,\omega),\cdots,\hat{\mathcal{D}}_{s_m}(t,\omega)]^\top$ is the STFT vector of the source signals.

Assumption 1. Given a source signal $\mathbf{s}_i$, its STFT transformation is denoted as $\mathcal{D}_{\mathbf{s}_i}$ in the TF domain. There always exists $\mathcal{D}_{\mathbf{s}_i}$ that is dominant at all (t,ω) TF points, i.e., $|\mathcal{D}_{\mathbf{s}_i}(t,\omega)| \gg |\mathcal{D}_{\mathbf{s}_j}(t,\omega)|$ for $\forall j \neq i$. □

The assumption implies that all sources are disjoint in the TF domain, i.e., there only one source is active on the TF point (t,ω). Then, (11) can be rewritten as

$$\mathcal{D}_{\boldsymbol{\Phi}}(t,\omega) = \hat{\mathcal{D}}_{\mathbf{s}_i}(t,\omega)\tilde{\mathbf{W}}_i, \tag{12}$$

where the TF feature matrix $\mathcal{D}_{\boldsymbol{\Phi}}(t,\omega)$ can be represented by the i-th column vector $\tilde{\mathbf{W}}_i$ with a multiplicative coefficient $\mathcal{D}_{\hat{\mathbf{s}}_i}(t,\omega)$. This implies that the target matrix $\tilde{\mathbf{W}}_i$ can be a linear combination of a few numbers of sample points from the matrix $\mathcal{D}_{\boldsymbol{\Phi}}(t,\omega)$ with the coefficient $\hat{\mathcal{D}}_{\mathbf{s}_i}(t,\omega)$.

We next formulate the problem of (12) by using a sparse direction for TF representation of the mixture TF matrix $\mathcal{D}_{\boldsymbol{\Phi}}(t,\omega)$. Let $\boldsymbol{\pi}_1,\boldsymbol{\pi}_2,\cdots,\boldsymbol{\pi}_L$ be the reshaped vector of all the mixture TF matrix $\mathcal{D}_{\boldsymbol{\Phi}}$, and L is the number of TF points (t,ω). We can define a one row vector $\boldsymbol{\Pi} \triangleq [\boldsymbol{\pi}_1,\cdots,\boldsymbol{\pi}_L]$ that is row-wise stacked together to be generated by the mixture TF matrix $\mathcal{D}_{\boldsymbol{\Phi}}$ at all (t,ω).

The further solution of (13) is the sparse representation of the TF feature vector $\mathcal{D}_{\boldsymbol{\Pi}}$, that will later construct the estimation of the coefficient matrix in the TF domain.

$$\mathcal{J}(\mathbf{c}_i,\eta) = \frac{1}{2}\|\boldsymbol{\pi}_i - \mathcal{D}_{\boldsymbol{\Pi}}\mathbf{c}_i\|_2^2 + \eta\|\mathbf{c}_i\|_1, \tag{13}$$

subject to $\mathbf{c}_{ii} = 0$, where $\eta > 0$ is a scalar parameter to balance the trade-off between the sparsity and reconstruction error. Once a sparse coding problem is built, the solution can be obtained by solving the convex optimization problem. Here, we use the l_1-Homotopy method in [15] to calculate the redundant dictionary $\mathbf{c}_i$ of (13).

3.3. Source Recovery

Since the mixing matrix is not irreversible in the UBSS [16], the recovered sources also need to be estimated even though the mixing matrix has been known. Therefore, we derive the sub-matrix $\hat{\mathbf{W}}$ on the following assumption.

Assumption 2. At most $n-1$ sources among m sources are active at each TF point for $m > n$. □

Definition 1. Given a matrix $\mathbf{W}$ of size $n \times m$, for any submatrices $\hat{\mathbf{W}}_i$ composed of size $n \times (n-1)$, there are $\binom{m}{n-1}$ elements included in the set of $\hat{\mathbf{W}}$, that is

$$\hat{\mathbf{W}} = \{\hat{\mathbf{W}}_i | \hat{\mathbf{W}}_i = [\hat{\mathbf{W}}_{\lambda_1},\cdots,\hat{\mathbf{W}}_{\lambda_{m-1}}]\}. \tag{14}$$

The condition is easily met and hence not restrictive for audio data [1].

Thus, for any given mixture TF vector $\mathcal{D}_{\mathbf{\Phi}}(t,\omega)$, there must exist an optimal sub-matrix $\hat{\mathbf{W}}_* = [\hat{\mathbf{W}}_{\lambda_1}, \cdots, \hat{\mathbf{W}}_{\lambda_{m-1}}]$ at each TF point (t,ω), such that

$$\hat{\mathbf{W}}_* = \arg\min_{\hat{\mathbf{W}}_i \in \hat{\mathbf{W}}} \left\| \mathcal{D}_{\mathbf{\Phi}}(t,\omega) - \hat{\mathbf{W}}_i \hat{\mathbf{W}}_i^{\dagger} \mathcal{D}_{\mathbf{\Phi}}(t,\omega) \right\|_2, \quad (15)$$

where $\hat{\mathbf{W}}_i^{\dagger}$ is the pseudo-inverse of $\hat{\mathbf{W}}_i$, which is defined as $\hat{\mathbf{W}}^{\dagger} = (\hat{\mathbf{W}}^{\top}\hat{\mathbf{W}})^{-1}\hat{\mathbf{W}}^{\top}$.

Thus, each source in the TF domain can be estimated by

$$\hat{\mathcal{D}}_{s_j}(t,\omega) = \begin{cases} \hat{\mathbf{W}}_*^{\dagger} \mathcal{D}_{\mathbf{\Phi}}(t,\omega), & \text{if } j = \lambda_i, \\ 0, & \text{otherwise}, \end{cases} \quad (16)$$

where λ_i is the index number of the optimal sub-matrix that implies the non-zero element of $\hat{\mathcal{D}}_{s_j}$ at each TF point. The source estimator $\hat{s}_i(t)$ is then obtained by converting $\hat{\mathcal{D}}_{s_i}(t,\omega)$ to the time domain using the inverse STFT.

4. EVALUATION

4.1. Experimental Setup

The experiments are designed on the audio data of real-world that are available from the literature [2]. The mixture signals are assumed to be mixed nonlinearly. Each observation is a linear mixture of the nonlinear distortion of sources, such as an exponential transformation $e^{s_1(t)}, \cdots, e^{s_m(t)}$. The linear mixture is derived from a matrix that randomly generated from the uniform distribution $U[-1,1]$. Example 1 separates $n = 3$ observations transformed from $m = 4$ independent speech signals. The noise is assumed to be generated from a white and Gaussian distribution with some uncorrelated data points whose variance is usually assumed to be uniform. The results are shown on the signal-to-noise power ratio (SNR) with the range from 5 dB to 45 dB. Example 2 uses the observations generated from the enhancement of the undetermined level, i.e., the number of sources is increased from 4 to 7 while the number of observations is kept as 3. All recordings were sampled to 16 kHz. The STFT frame size is set as 1024 points, time frame shift equals to 256, and Hanning window is used as the weighting function. To reduce the randomness effect, the simulation is repeated 20 times.

We compared with two algorithms, the UBSS method based on the sparse coding [2], and the underdetermined convolutive BSS (UCBSS) method[2] based on the subspace representation [17]. The normalized mean squared error (NMSE) [2] is used to measure the separation accuracy, which is defined by

$$\text{NMSE}(\mathbf{s}, \hat{\mathbf{s}}) = 10 \log_{10} \left(\frac{1}{m} \sum_{i=1}^{m} \min_{\delta} \frac{\|\mathbf{s}_i - \delta \hat{\mathbf{s}}_i\|_2^2}{\|\mathbf{s}_i\|_2^2} \right), \quad (17)$$

The scalar δ is used for controlling the scalar ambiguity.

[2]https://slsp.kaist.ac.kr/xe/index.php?mid=software

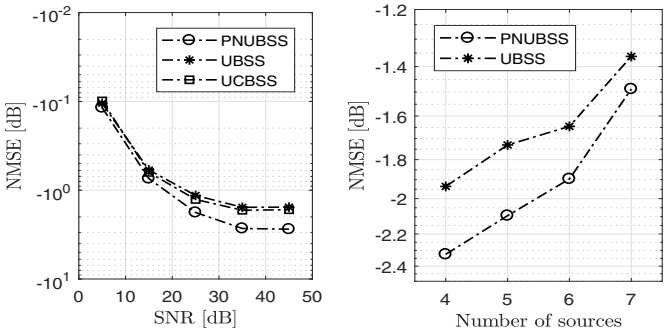

Fig. 2: The separation results on comparisons of the proposed PNUBSS method, the UBSS method [2], and the UCBSS method [17]. (a) Over the different SNR levels, and (b) over the different number of sources.

4.2. Results

In Fig. 2 (a), the results show the NMSE on the different SNR levels. Benefiting from the polynomial networks formed of the good approximate basis, the proposed polynomial networks-based underdetermined blind source separation (PNUBSS) algorithm can provide more accurate performance on the source recovery. Due to the polynomial networks, the nonlinearity can be approximated by a set of basis. As we can see, the NMSE measure on both the UBSS and the UCBSS methods also decreases with SNR being increased. The proposed PNUBSS consistently provides good results over the whole SNR range, suggesting the proposed algorithm is robust.

In Fig. 2 (b), the performance on the source recovery is decreasing as the underdetermined level is enhancing, i.e., more sources are available. The UCBSS algorithm is not available when the number of sources exceeds two than that of observations.

5. CONCLUSIONS

This paper introduces a novel nonlinear BSS algorithm. The main contribution of the novel separation approach is to propose a polynomial network to approximate the varieties of the nonlinearity. The approach attempts to generate a polynomial network, which provides a good approximate basis for the values attained by a set of mapping functions. The layers of our network start with polynomials of degree 1, which have the large bias attained by this simple approximate network. To create a higher level representation of the data to decrease the bias, we next make the network deeper and deeper. Each enhancement of the degree makes the layer deeper into our network. We then exploit the linear separation on top of these outputs. Thus, our work presents the advantages offered by both, the deep architectures formed by a polynomial network, and the coefficient matrix derived by sparse coding in the underdetermined scenario. In particular, our network can search the number of layers that makes deeper until the candidate dataset becomes empty.

6. REFERENCES

[1] A. Aissa-El-Bey, N. Linh-Trung, K. Abed-Meraim, A. Belouchrani, and Y. Grenier, "Underdetermined Blind Separation of Nondisjoint Sources in the Time-Frequency Domain," *IEEE Trans. Signal Process.*, vol. 55, pp. 897–907, Mar. 2007.

[2] L. Zhen, D. Peng, Z. Yi, Y. Xiang, and P. Chen, "Underdetermined Blind Source Separation Using Sparse Coding," *IEEE Trans. on Neural Networks and Learning Systems*, vol. 28, no. 12, pp. 3102–3108, Dec. 2017.

[3] A. Hyvärinen and P. Pajunen, "Nonlinear Independent Component Analysis: Existence and Uniqueness Results," *Neural Networks*, vol. 12, no. 3, pp. 429–439, Sep. 1999.

[4] D. Martinez and A. Bray, "Nonlinear Blind Source Separation Using Kernels," *IEEE Trans. on Neural Networks*, vol. 14, no. 1, pp. 228–235, Jan. 2003.

[5] A. Hyvärinen and H. Morioka, "Nonlinear ICA of Temporally Dependent Stationary Sources," in *Proc. of Int. Conf. on Artificial Intelligence and Statistics (AISTATS)*, vol. 54, 20–22 Apr. 2017, pp. 460–469.

[6] L. Wang and T. Ohtsuki, "Nonlinear Blind Source Separation Unifying Vanishing Component Analysis and Temporal Structure," *IEEE Access*, vol. 6, pp. 42 837–42 850, July 2018.

[7] G. Burel, "Blind Separation of Sources: A Nonlinear Neural Algorithm," *Neural Networks*, vol. 5, no. 6, pp. 937–947, Dec. 1992.

[8] Y. Tan, J. Wang, and J. M. Zurada, "Nonlinear Blind Source Separation Using a Radial Basis Function Network," *IEEE Trans. on Neural Networks*, vol. 12, no. 1, pp. 124–134, Jan. 2001.

[9] B. Ehsandoust, M. Babaie-Zadeh, B. Rivet, and C. Jutten, "Blind Source Separation in Nonlinear Mixtures: Separability and a Basic Algorithm," *IEEE Trans. on Signal Processing*, vol. 65, no. 16, pp. 4339–4352, Aug. 2017.

[10] S. Harmeling, A. Ziehe, M. Kawanabe, B. Blankertz, and K.-R. Müller, "Nonlinear Blind Source Separation Using Kernel Feature Spaces," in *Proc. Int. Workshop on Independent Component Analysis and Blind Signal Separation*, Dec. 2001, pp. 102–107.

[11] S. Harmeling, A. Ziehe, M. Kawanabe, and K.-R. Müller, "Kernel Feature Spaces and Nonlinear Blind Source Separation," in *Advances in neural information processing systems (NIPS)*, vol. 14, Dec. 2002, pp. 761–768.

[12] S. Harmeling, A. Ziehe, and M. Kawanabe, "Kernel-based Nonlinear Blind Source Separation," *Neural Computation*, vol. 15, no. 5, pp. 1089–1124, May 2003.

[13] R. Livni, D. Lehavi, S. Schein, H. Nachlieli, S. Shalev-Shwartz, and A. Globerson, "Vanishing Component Analysis," in *Proc. of Int. Conf. on Machine Learning (ICML)*, Jun. 2013, pp. 597–605.

[14] R. Livni, S. Shalev-Shwartz, and O. Shamir, "An Algorithm for Training Polynomial Networks," in *arXiv preprint arXiv:1304.7045*, 2013.

[15] M. S. Asif and J. Romberg, "Sparse Recovery of Streaming Signals Using l_1-Homotopy," *IEEE Trans. on Signal Process.*, vol. 62, no. 16, pp. 4209–4223, Aug. 2014.

[16] D. Peng and Y. Xiang, "Underdetermined Blind Separation of Non-Sparse Sources Using Spatial Time-Frequency Distributions," *Digital Signal Process.*, vol. 20, no. 2, pp. 581–596, Mar. 2010.

[17] J. Cho and C. D. Yoo, "Underdetermined Convolutive BSS: Bayes Risk Minimization Based on a Mixture of Super-Gaussian Posterior Approximation," *IEEE/ACM Trans. on Audio, Speech, and Language Process.*, vol. 23, no. 5, pp. 828–839, Mar. 2015.