Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 18, No. 3, pp. 545–564, 2022

MULTIVARIATE SERIES NOISE REDUCTION VIA SEQUENTIAL MOJORIZATION METHOD AND ITS EXTENSIONS

Aoxiang Chen and Qingna Li*

Abstract: Based on the Hankel structured low-rank approximation and the technique of majorization, the sequential majorization method (SMM-Cadzow) proposed by Qi et al. (2018) has been proven to be an effective and fast method for the signal extraction from noisy time series. This method elevates the approximation of low-rank matrix by designing a new object function. In this paper, we use the idea of multivariate analysis to reconstruct the SMM-Cadzow method and therefore form the multivariate version of SMM-Cadzow (MSMM-Cadzow). We thoroughly describe the problems of selecting two important parameters (window length and the rank of the low-rank matrix). The result of signal extraction largely depends on the two parameters. The condition which perfectly fits the MSMM-Cadzow model is also introduced in the paper. Focusing on denoising the weak signal, we propose two new schemes (recycling MSMM algorithm and variate accumulation MSMM algorithm) using the MSMM-Cadzow. The numerical results demonstrated that MSMM-Cadzow has a significant importance on signal extraction and the denoising of weak signal has been improved by the proposed algorithms (recycling MSMM algorithm and variate accumulation MSMM algorithm).

Key words: *signal extraction, multivariate analysis, weak signal recovery, Hankel matrix*

Mathematics Subject Classification: *62H99, 65K10*

1 Introduction

In this paper, we consider the reconstruction of multivariate signals from its noised series. Let's first describe it mathematically. Let $s^{(i)} \in \mathcal{R}^{N_i}$; $i = 1, \ldots, n$ be the signals with length N_i, let the observed signals $y^{(i)} \in \mathcal{R}^{N_i}$; $i = 1, \ldots, n$, be the mixture of $s^{(i)}$ and some noise $r^{(i)} \in \mathcal{R}^{N_i}$; $i = 1, \ldots, n$, *i.e.*,

$$y^{(i)} = s^{(i)} + r^{(i)}, i = 1, \ldots, n. \tag{1.1}$$

The aim is to reconstruct $s^{(i)}$, $i = 1, \ldots, n$ based on $y^{(i)}$, $i = 1, \ldots, n$. If $N_1 = \cdots = N_n$, the multivariate time series have the same pattern. Otherwise, the data are referred to as unpatterned multivariate time series. Such situation usually happens in climate and weather data where $s^{(i)}$, $i = 1, \ldots, n$ represents different indices in climate, and they might be observed in different frequencies, resulting in different length of $s^{(i)}$. Reconstruction of multivariate data from such kind of unpatterned noised data is the main aim of this paper.

*Corresponding author. This author's research was supported by the National Natural Science Foundation of China (No.12071032).

From the statistics point of view, there are various methods which can be used to reconstruct multivariate series, including the traditional methods: vector auto-regression method and copula method [2, 16]. We also refer to monographs [17, 21] for more details of those traditional methods. Most of the above approaches deal with pattern data, i.e., $N_1 = \cdots = N_n$. To deal with unpatterned multivariate series, there are some approaches, for example, multistage maximum likelihood estimation and multivariate singular spectrum analysis [1, 3].

For univariate time series, besides the above traditional methods, the low-rank Hankel matrix optimization has become the main approaches recently. Singular Spectrum Analysis (SSA) is one of such methods which is based on the structure of Hankel matrix. SSA has been extensively and effectively applied to many areas such as signal processing, economics, image processing, earth science, and so on [6, 9, 10, 18]. The advantage of SSA is that it is a non-parametric method, meaning that there are no previous assumptions (such as whether the series is stationary). However, SSA suffers some drawbacks mainly in the Hankel-type weights. Excellent work has been done on improving SSA [8, 15, 26]. In particular, [8] propose a sequential majorization method (SMM-Cadzow), which deals with more reasonable weights in low-rank Hankel matrix model. SMM-Cadzow is demonstrated to be computationally efficient in denoising time series.

Based on the above analysis, a natural question is whether the SMM-Cadzow method can be adapted to deal with multivariate time series with different patterns. This motivates the work in this paper. We proposed a multivariate noise reduction method, referred to as the multivariate sequential majorization method (MSMM). The contribution of this paper are in three folds. Firstly, MSMM is proposed based on the idea of multivariate analysis. Secondly, we address how to choose the window length and the rank of the Hankel matrix, both of which are key parameters in MSMM. Thirdly, for weak signal extraction, which is an important yet difficult scenario, we propose two variants of MSMM, i.e., the recycling MSMM method and variate accumulation MSMM method. Numerical experiments are conducted to verify the efficiency of the proposed MSMM, especially in denoising weak signals.

The rest of this paper is organized as follows. In Section 2, we introduce some preliminaries, mainly focusing on SMM for the univariate time series. In Section 3, we propose our MSMM for multivariate time series with different patterns. In Section 4, we discuss how to choose the two important parameters: the window length and the rank of the Hankel matrix. In Section 5, we propose two variants: the recycling MSMM method and variate accumulation MSMM to deal with the weak signal extraction. Numerical results are reported in Section 6 including the performance of MSMM for unpatterned multivariate time series as well as two variants of MSMM for weak signal extraction. Final conclusions are given in Section 7.

Notations. We use small letter to represent column vectors and capital letter to represent matrices. The i-th element in $\mathbf{x}$ is denoted as x_i. The inner product of matrix A and B of the same size is defined as $\langle A, B\rangle := Tr(A^\mathsf{T}B)$. Let $\circ$ denote the Hadamard product. Let $\|\cdot\|_F$ denote the Frobenius norm for matrices and l_2 norm for vectors. Let $Diag(x)$ denote the diagonal matrix with x as the diagonal entries. We use $\mathbf{1}_L$ to denote the column vector of length L with all elements one.

2 Preliminaries

In this section, we introduce the mathematical problem for extracting univariate time series, which is mainly summarized from [24]. Then we briefly review the main idea of SMM-Cadzow [8] for univariate noise reduction.

2.1 Hankel Matrix Optimization of Noise Reduction for Univariate Case

Let $\mathbf{x} = (x_1, \ldots, x_N) \in \mathcal{R}^N$ be a time series of length N, and $L > 0$ $(1 < L < N)$ is a given integer called window length. Letting $K = N - L + 1$, we can define the following L-lagged vectors:

$$\mathbf{x}_i^{\mathsf{T}} = (x_i, \ldots, x_{i+L-1}), \quad i = 1, \ldots, K.$$

Define the L-lagged matrix as below

$$X = \mathcal{T}(\mathbf{x}) := (\mathbf{x}_1, \ldots, \mathbf{x}_K) \in \mathcal{R}^{L \times K},$$

which is usually referred to as the trajectory matrix. It is obvious that X is a Hankel matrix. Let $\mathcal{H}$ be the set of Hankel matrix of size $L \times K$. The mapping $\mathcal{T} : \mathcal{R}^N \mapsto \mathcal{H}$ maps time series of length N into a Hankel matrix of size $L \times K$. It is easy to notice that $\mathcal{T}$ is an one to one projection, and the inverse of it: $\mathcal{T}^{-1}$, exists and maps a Hankel matrix of size $L \times K$ to time series of length N.

Let $\mathbb{X}_r^N$ to represent series whose length is N and the rank of corresponding trajectory matrix X is no larger than r, that is,

$$\mathbb{X}_r^N := \{\mathbf{x} \in \mathbb{R}^N | rank(\mathcal{T}(\mathbf{x})) \leq r\}.$$

To denoise the noise corrupted time series $\mathbf{y} \in \mathcal{R}^N$, we can extract the true signal by looking for the nearest approximation to $\mathbf{y}$ from $\mathbb{X}_r^N$. Particularly, we minimize the following weighted norm to denoise the time series $\mathbf{y} \in \mathcal{R}^N$:

$$\min_{\mathbf{x} \in \mathbb{X}_r^N} \ f_w(\mathbf{x}) = \|\mathbf{x} - \mathbf{y}\|_w^2 := \sum_{i=1}^{N} w_i (x_i - y_i)^2. \tag{2.1}$$

The underlying reason to set a low-rank constraint $\mathbf{x} \in \mathbb{X}_r^N$ is that the signal we wish to extract corresponds to a low-rank matrix under the mapping $\mathcal{T}$ [8, Section 1]. In fact, it has a close link to time series that are governed by a linear recurrence relation. We refer the interested reader to [15, 24, 25, 26]. Problem (2.1) is a popular method to extract univariate signals, and it was investigated in many papers such as [15, 26, 25].

To derive the equivalent Hankel matrix optimization formulation of (2.1), let $\mathcal{M}$ be the set of matrices of size $L \times K$ and $\mathcal{M}_r \subset \mathcal{M}$ be the set of matrix whose rank is no larger than r. Let $\mathbf{w}^{\mathsf{T}} := (w_1, \ldots, w_N)$, $w_i \geq 0$ and $\mathbf{v} \in \mathbb{R}^N$ (suppose $L \leq K$) be given by:

$$v_i := \begin{cases} \frac{1}{i} & \text{when i=1,...,L-1;} \\ \frac{1}{L} & \text{when i=L,...,K;} \\ \frac{1}{N-i+1} & \text{when i=K+1,...,N,} \end{cases} \tag{2.2}$$

Let $\sqrt{\mathbf{w}} \in \mathcal{R}^N$ denote the component-wise square root vector and define

$$M := \mathcal{T}(\sqrt{\mathbf{v}} \circ \sqrt{\mathbf{w}}). \tag{2.3}$$

The equivalent Hankel matrix optimization model of (2.1) is given by

$$\min_{X \in \mathcal{M}_r \cap \mathcal{H}} \ f(X) := \frac{1}{2} \|M \circ (X - Y)\|_F^2. \tag{2.4}$$

However, solving (2.4) is computationally difficult unless $\mathbf{w}$ has some properties [15]. Even for $\mathbf{w} = (1, \ldots, 1)^{\mathsf{T}} \in \mathcal{R}^N$, (2.4) is still a tricky problem. Below we briefly review the sequential majorization method (SMM) proposed by [8], which solves (2.4) efficiently.

2.2 SMM

SMM-Cadzow aims to solve (2.1) which is indeed a Hankel low-rank approximation problem with arbitrary weights. First, we need to transform (2.1) to the corresponding Hankel matrix form problem (2.4). In order to tackle the computational difficulty in (2.4), we need to approximate the object function of (2.4) with a function which is easier to solve, and SMM-Cadzow method uses the following majorization function to approximate $f(X)$ at a given point Z [8]:

$$f_m(X,Z) := f(Z) + \langle \nabla f(Z), X - Z\rangle + \frac{1}{2}\|(\sqrt{\mathbf{p}}\sqrt{\mathbf{q}}^{\mathsf{T}}) \circ (X - Z)\|_F^2, \quad \forall\, X \in \mathcal{M}, \tag{2.5}$$

where $\mathbf{p} \in \mathcal{R}^L$ and $\mathbf{q} \in \mathcal{R}^K$ are two positive vectors which possess the follow properties:

$$\|M \circ X\|_F \le \|(\sqrt{\mathbf{p}}\sqrt{\mathbf{q}}^{\mathsf{T}}) \circ X\|_F, \quad \forall X \in \mathcal{M}. \tag{2.6}$$

Both sides of (2.6) could be seen as the weighted norm of X, and the weight matrix on the right side is actually a rank-one matrix. Solving (2.5) could benefit us in computing. As a matter of fact, (2.5) could be regarded as an approximation of (2.4) at a given point $Z \in \mathcal{M}$. Moreover, (2.5) is solved by using the following alternating projections.

Let $\Pi_{\mathcal{H}}^{(P,Q)}(Y)$ represent the optimal solution of the following optimization problem:

$$\min_{X\in\mathcal{H}} \ \|P(X - Y)Q\|_F. \tag{2.7}$$

Vectors $\mathbf{p}$ and $\mathbf{q}$ are positive vector in $\mathcal{R}^L$ and $\mathcal{R}^K$ respectively, and $P = diag(\mathbf{p})$, $Q = diag(\mathbf{q})$. The solution of (2.7), denoted as $\widehat{\mathrm{Y}} = \Pi_{\mathcal{H}}^{(P,Q)}(Y)$, takes the following form [26]:

$$\widehat{y}_{ij} = \frac{\sum_{l,k:l+k=i+j} p_l q_k y_{lk}}{\sum_{l,k:l+k=i+j} p_l q_k}. \tag{2.8}$$

Now we let $\Pi_{\mathcal{M}_r}^{(P,Q)}(\mathbf{Y})$ to be the optimal solution of the following optimization problem:

$$\min_{X\in\mathcal{M}_r} \ \|P(X - Y)Q\|_F. \tag{2.9}$$

The solution of (2.9), denoted as $\widehat{\mathrm{Y}} = \Pi_{\mathcal{M}_r}^{(P,Q)}(Y)$, is as follows [15]:

$$\Pi_{\mathcal{M}_r}^{(P,Q)}(Y) = P^{-1}\Pi_{\mathcal{M}_r}(PYQ)Q^{-1}. \tag{2.10}$$

3 Multivariate Sequential Majorization Method (MSMM)

In this section, we will develop MSMM for the noise reduction of multivariate time series with different patterns. We will first establish the Hankel matrix optimization model, then propose MSMM for solving the optimization model. Finally, we will discuss the properties of MSMM.

3.1 The Hankel Matrix Optimization Model

Recall that our aim is to denoise the multivariate time series $\mathbf{y}^{(i)} \in \mathcal{R}^{N_i}$, $i = 1, \ldots, n$, as defined in (1.1). Similar to the case of univariate time series, a natural way to recover the multivariate time series is to directly solve the weighted least squares minimization model:

$$\min_{\mathbf{X}^{(i)}\in\mathbb{X}_r^{N_i},\ i=1,\ldots,n} \ \sum_{i=1}^{n} \|\mathbf{x}^{(i)} - \mathbf{y}^{(i)}\|^2_{\mathbf{w}^{(i)}}, \tag{3.1}$$

where $\mathbf{w}^{(i)} \in \mathcal{R}^{N_i}$, $i = 1, \ldots, n$ are the given corresponding non-negative weighted vectors. Clearly, (3.1) is separable and thus could be solved sequentially by using SMM. However, the underlying structure and features of the multivariate signals might be ignored since one treats them as several univariate time series. The main purpose of our paper is to propose a model which can perform noise reduction simultaneously. The reason is that simultaneously analysis allows us to observe the relations and extract common structures among several time series.

To modify (3.1) as the Hankel matrix optimization model, which enables us to perform noise reduction simultaneously, the main challenge is how to construct the trajectory matrix. Usually there are two ways to construct the trajectory matrix for multivariate series [11, 4]. The only difference between the two ways is how they stack the final block Hankel matrix after each trajectory is formed for each univariate series $\mathbf{y}^{(i)}$ by $\mathcal{T}$. One is to stack horizontally and the other is vertically. Given the fact that the multivariate series have different patterns, we use the horizontal way to construct the trajectory matrix in our paper. After connecting each trajectory matrix horizontally, we obtain the final trajectory matrix for multivariate series. Furthermore, we obtain the column space of the trajectory matrix, referred to as the trajectory space, which is a space spanned by all the L-lagged vectors of the n series. The above analysis leads to the following mathematical statement before which an assumption is needed.

Assumption 3.1. Assume the trajectory matrix of the noise series is of full rank.

Let $\mathcal{H}^B := \mathcal{H}^1 \times \mathcal{H}^2 \times \cdots \times \mathcal{H}^n$ be the set of block Hankel matrices, where $\mathcal{H}^i$ denotes the Hankel matrices of size $L \times K_i$, and $K_i = N_i - L + 1$, $i = 1, \ldots, n$. Let $\mathcal{M}^B$ be the set of matrices of size $L \times K_T$, where $K_T = K_1 + \cdots + K_n$. Define $\mathcal{T}_M : \mathcal{R}^{N_1} \times \mathcal{R}^{N_2} \times \ldots \mathcal{R}^{N_n} \mapsto \mathcal{H}^B$ by

$$\overline{Y} = \mathcal{T}_M(\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(n)}) := [\mathcal{T}(\mathbf{y}^{(1)}), \mathcal{T}(\mathbf{y}^{(2)}), \ldots, \mathcal{T}(\mathbf{y}^{(n)})] \in \mathcal{H}^B. \tag{3.2}$$

Similar to the property of $\mathcal{T}$, $\mathcal{T}_M$ is also a one-to-one mapping and the inverse of $\mathcal{T}_M$, denoted by $\mathcal{T}_M^{-1}$, transferring block Hankel matrices back to multivariate series. That is,

$$\mathcal{T}_M^{-1}(\overline{Y}) := (\mathcal{T}^{-1}(Y^{(1)}), \ldots, \mathcal{T}^{-1}(Y^{(n)})), \tag{3.3}$$

where $Y^{(i)} \in \mathcal{H}^i$ and $\mathcal{T}^{-1}(Y^{(i)}) \in \mathcal{R}^{N_i}$. Similar as in (2.2) and (2.3), define $\mathbf{v}^{(j)} \in \mathcal{R}^{N_j}$ as in (2.2) with $N := N_j$ and let

$$M^{(j)} = \mathcal{T}(\sqrt{\mathbf{v}^{(j)}} \circ \sqrt{\mathbf{w}^{(j)}}), \quad \overline{M} := [M^{(1)}, \ldots, M^{(n)}]. \tag{3.4}$$

Consequently, we reach the following Hankel matrix optimization model to denoise the multivariate time series $\mathbf{y}^{(i)} \in \mathcal{R}^{N_i}$, $i = 1, \ldots, n$:

$$\min_{\overline{X} \in \mathcal{M}_r^B \cap \mathcal{H}^B} \; f(\overline{X}) := \frac{1}{2} \|\overline{M} \circ (\overline{X} - \overline{Y})\|_F, \tag{3.5}$$

where $\mathcal{M}_r^B$ denotes the sets of matrices of size $L \times K_T$, whose rank is no larger than r.

Remark 1. Notice that although the set $\mathcal{H}^B$ is the set of block Hankel matrices, the set $\mathcal{M}_r^B$ requires that the matrix $\overline{X}$ must have the rank no larger than r, which is a restriction from the matrix point of view rather than regarding X as some horizontally block matrices. Consequently, one of the constraints $\overline{X} \in \mathcal{M}_r^B$ actually tackles the univariate time series uniformly, whereas, the constraint $\overline{X} \in \mathcal{H}^B$ deals with the unpatternness of each time series, by allowing each block Hankel matrix to admit different sizes.

3.2 MSMM for (3.5)

To solve (3.5), similar to SMM in [8], we first give the following majorization function of f at a given point $\overline{Z}$:

$$f_m(\overline{X},\overline{Z}) := f(\overline{Z}) + \langle \nabla f(\overline{Z}), \overline{X}-\overline{Z}\rangle + \frac{1}{2}\|(\sqrt{\mathbf{u}}\sqrt{\mathbf{v}}^{\mathsf{T}})\circ(\overline{X}-\overline{Z})\|_F^2, \quad \forall\, \overline{X}\in\mathcal{M}_r^B, \tag{3.6}$$

where $\mathbf{u}\in\mathcal{R}^L$ and $\mathbf{v}\in\mathcal{R}^{K_T}$ are two positive vectors share the following properties:

$$\|\overline{M}\circ\overline{X}\|_F \le \left\|\left(\sqrt{\mathbf{u}}\sqrt{\mathbf{v}}^{\mathsf{T}}\right)\circ\overline{X}\right\|_F \quad \forall\, \overline{X}\in\mathcal{M}_r^B. \tag{3.7}$$

The majorization function $f_m(\cdot,\cdot)$ satisfies the following conditions:

$$f_m(\overline{X},\overline{X}) = f(\overline{X}) \quad \forall\, \overline{X}\in\mathcal{M}_r^B, \text{ and, } f(\overline{X}) \le f_m(\overline{X},\overline{Z}) \quad \forall\, \overline{Z},\overline{X}\in\mathcal{M}_r^B.$$

Consequently, given $\overline{X}^k$, one can minimize the majorization function $f_m(\overline{X},\overline{X}^k)$ over the feasible set $\mathcal{M}_r^B\cap\mathcal{H}^B$, to get the next iterate X^{k+1}. The details of MSMM is given in Algorithm 1.

Algorithm 1: MSMM-Cadzow algorithm

1 Give the multivariate series: $\mathbf{y}^{(1)},\ldots,\mathbf{y}^{(n)}$, and the corresponding weight vectors $\mathbf{w}^{(1)},\ldots,\mathbf{w}^{(n)}$, choose the window length L, the rank r. Compute $\overline{Y}=\mathcal{T}_M(\mathbf{y}^{(1)},\ldots,\mathbf{y}^{(n)})$ and $\overline{M}$ as in (3.4). Let $\overline{X}^0=\overline{Y}$, $k:=0$;

2 Compute the $\mathbf{u}$ and $\mathbf{v}$ that satisfy (3.7);

3 Get $\overline{X}^{k+1}$ by solving the following subproblem approximately:

$$\min_{\overline{X}\in\mathcal{M}_r^B\cap\mathcal{H}^B} f_m(\overline{X},\overline{X}^k) = \min_{\overline{X}\in\mathcal{M}_r^B\cap\mathcal{H}^B} \frac{1}{2}\|U(\overline{X}-\triangle_k)V\|_F, \tag{3.8}$$

where $U=diag(\mathbf{u})$, $V=diag(\mathbf{v})$ and $\triangle_k$ is defined as $\triangle_k := \overline{X}^k - \overline{P}^{-2}(\overline{M}\circ\overline{M}\circ(\overline{X}^k-\overline{Y}))\overline{Q}^{-2}$;

4 If the stopping criteria is satisfied, output $\mathcal{T}_M^{-1}(\overline{X}^{k+1})$; otherwise, $k=k+1$, go to Step 3;

Similar to the convergence of SMM-cadzow in [8], it is easy to derive the following convergence result.

Theorem 3.1. *Let $\overline{X}^k$ be generated by MSMM. If $\overline{X}^k$ satisfies the following condition*

$$f_m(\overline{X}^{k+1},\overline{X}^k) \le f_m(\overline{X}^k,\overline{X}^k), \tag{3.9}$$

the object function sequence $f(\overline{X}^k)$ is non-decreasing and converges.

The above theorem implies that to achieve the convergence result for the object function, it is not necessary to get the exact solution of (3.8) which is very demanding due to the non-convexity of the problem. Instead, we only need to get an inexact solution which fulfills (3.9). It would render our computation relatively easy.

Remark 2. Theorem 3.1 is a convergence result in terms of $f(\overline{X}^k)$. As pointed in [8], such convergence result for the object function $f(\overline{X}^k)$ would be good enough for our model to work. Due to the non-convexity of the problem (3.5), it is very difficult to analyze whether the limit points of $\{\overline{X}^k\}$ is the critical point of the problem (3.5). We will leave it as a further topic to study in the future.

Remark 3. Here we would like to further highlight three reasons that we use the horizontal way to construct the trajectory matrix in deriving (3.2). Firstly, it enables us to maintain the structure of the trajectory space as it is in the unitary case. In construct, if we choose the vertical way, the structure of the resulting block trajectory matrix will change dramatically. Secondly, the horizontal way allows us to deal with the case of unpatterned multivariate time series, where each time series may have different lengths. One can allow different length of $\mathbf{y}^{(i)},\ i = 1, \ldots, n$, without worrying about the disturbance it may bring to the dimension of the trajectory space. Last but not the least, it can save the computational time and memory when running the algorithm MSMM. The main computational cost of MSMM lies in the spectral value decomposition in (3.8). Constructing the trajectory matrix horizontally will lead to similar size of matrices which then save the computational time and memory.

We end up this section by discussing some implementation issues.

3.3 Implementations of MSMM

In this part, we mainly discuss the choices of $(\mathbf{u}, \mathbf{v})$ and how to solve the sub-problem (3.8) inexactly.

Recall that $(\mathbf{u}, \mathbf{v})$ has to satisfy the condition in (3.7). A good choice of them will narrow the difference of the right hand side and the left hand side in (3.7) as much as possible so that the approximation f_m in (3.6) can be more accurate. For the univariate case, the choice of $(\mathbf{u}, \mathbf{v})$ is fully discussed in [8]. For the multivariate case, recall that $\mathbf{u} \in \mathcal{R}^L$, $\mathbf{v} \in \mathcal{R}^{K_T}$. To generate $(\mathbf{u}, \mathbf{v})$, let us first define $(\mathbf{u}^{(i)}, \mathbf{v}^{(i)})$ to be the corresponding parameters for the univariate time series $\mathbf{y}^{(i)} \in \mathcal{R}^{N_i}$ with $\mathbf{u}^{(i)} \in \mathcal{R}^L$ and $\mathbf{v}^{(i)} \in \mathcal{R}^{K_i},\ i = 1, \ldots, n$, specified later. $(\mathbf{u}, \mathbf{v})$ can be generated by:

$$u_l := max\{u_l^{(i)} | i = 1, \ldots, n\},\ l = 1, \ldots, L,\ \mathbf{v} = [\mathbf{v}^{(1)}; \ldots; \mathbf{v}^{(n)}], \tag{3.10}$$

where for the *l-th* element in $\mathbf{u} \in \mathcal{R}^L$ we choose the largest one among all the *l-th* element of $\mathbf{u}^{(i)} \in \mathcal{R}^L,\ i = 1, \ldots, n$, and for $\mathbf{v} \in \mathcal{R}^{K_T}$, we simply connect $\mathbf{v}^{(i)} \in \mathcal{R}^{K_i},\ i = 1, \ldots, n$. We need the following property of $(\mathbf{u}^{(i)}, \mathbf{v}^{(i)})$.

Proposition 3.2. *Let* $(\boldsymbol{u}^{(i)}, \boldsymbol{v}^{(i)}) \in \mathcal{R}^L \times \mathcal{R}^{K_i}$ *satisfy*

$$\|M^{(i)} \circ X^{(i)}\|_F \leq \|(\sqrt{\boldsymbol{u}^{(i)}}\sqrt{\boldsymbol{v}^{(i)}}^{\mathsf{T}}) \circ X^{(i)}\|_F,\ i = 1, \ldots, n.$$

Then $(\boldsymbol{u}, \boldsymbol{v}) \in \mathcal{R}^L \times \mathcal{R}^{K_T}$ *defined by* (3.10) *satisfies* (3.7).

The proof can be done easily by using the definitions of Hadamard product and Frobenius norm.

By Proposition 3.2, the choice of $(\mathbf{u}, \mathbf{v})$ is connected with the choices of $(\mathbf{u}^{(i)}, \mathbf{v}^{(i)}),\ i = 1, \ldots, n$. Consequently, one can use different schemes to generate $(\mathbf{u}^{(i)}, \mathbf{v}^{(i)})$ following the univariate case in [8, Section 4]. To be self contained, below we briefly describe two ways: simple way and a more complicated way to generate $(\mathbf{u}^{(i)}, \mathbf{v}^{(i)})$. More details can be found in [8, Section 4].

A simple way to calculate [8, Section 4] is based on $\overline{M}^{(i)}$, which is given by (recall that $K_i = N_i - L + 1$):

$$\begin{aligned} u_l^{(i)} &:= max\{\overline{M}_{lk}^{(i)} | k = 1, \ldots, K_T\},\ l = 1, \ldots, L, \\ v_k^{(i)} &:= max\{\overline{M}_{lk}^{(i)} | l = 1, \ldots, L\},\ k = 1, \ldots, K_i. \end{aligned} \tag{3.11}$$

A more complicated way to generate $(\mathbf{u}^{(i)}, \mathbf{v}^{(i)})$ is to solve a linear programming problem, which will leads to a more accurate estimation and therefore narrow the gap between $\|(\sqrt{\mathbf{u}^{(i)}}\sqrt{\mathbf{v}^{(i)}}^{\mathsf{T}}) \circ X^{(i)}\|$ and $\|M^{(i)} \circ X^{(i)}\|$ for any $X^{(i)}$. Such scheme is given below, which comes from Algorithm 4.1 in [8].

Algorithm 2: Generation of $(\mathbf{u}^{(i)}, \mathbf{v}^{(i)})$

1 Initialization: $(\mathbf{a}, \mathbf{b}) \in \mathcal{R}^L \times \mathcal{R}^{K_i}$, weight vector $\mathbf{w}^{(i)}$, window length L, $K_i = N_i - L + 1$;

2 Solve the following linear programming problem to get $(\mathbf{s}, \mathbf{t}) \in \mathcal{R}^L \times \mathcal{R}^{K_i}$:

$$\begin{aligned} \max_{\mathbf{s}\in\mathcal{R}^L, \mathbf{t}\in\mathcal{R}^{K_i}} \quad & \alpha\gamma + \beta\gamma = (\mathbf{1}_L^{\mathsf{T}}\mathbf{s})(\mathbf{1}_{K_i}^{\mathsf{T}}\mathbf{b}^{(i)}) + (\mathbf{1}_L^{\mathsf{T}}\mathbf{a}^{(i)})(\mathbf{1}_{K_i}^{\mathsf{T}}\mathbf{t}) \\ \text{s.t.} \quad & \sum_{l+k=j+1} (\mathbf{a}_l t_k + \mathbf{b}_k s_l) \le \sum_{l+k=j+1} \mathbf{a}_l \mathbf{b}_k - w_j^{(i)},\ j = 1, \ldots, N_i, \\ & 0 \le \mathbf{s} \le \rho\mathbf{a}, 0 \le t \le \rho\mathbf{b} \end{aligned} \tag{3.12}$$

with prescribed ρ;

3 Output: $\mathbf{u}^{(i)} = \mathbf{a} - \mathbf{s}$, $\mathbf{v}^{(i)} = \mathbf{b} - \mathbf{t}$.

Remark 4. The parameter ρ in Algorithm 2 is used to adjust $(\mathbf{s}, \mathbf{t})$ to avoid too large $(\mathbf{s}, \mathbf{t})$. The initial input $(\mathbf{s}, \mathbf{t})$ for each $i = 1, \ldots, n$ can be chosen as the $(\mathbf{u}^{(i)}, \mathbf{v}^{(i)})$ defined in the simple way (3.11).

Remark 5. Both of the simple way (3.11) and the complicated way in Algorithm 2 have their own advantages. The simple way will lead to small computational cost in generating $(\mathbf{u}, \mathbf{v})$. In contrast, Algorithm 2 takes some computational cost, however, it will lead to a more accurate approximation for the majorization of function f.

The other implementation issue is how to solve sub-problem (3.8) in Step 3 of Algorithm 1. We use the idea of alternating projection to get an approximate solution of (3.8). Specifically, we can generate the iteration $\check{X}^i$ in the following way when solving (3.8):

$$\check{X}^{i+1} = \Pi_{\mathcal{H}^B}^{(\mathbf{u},\mathbf{v})} \left(\Pi_{\mathcal{M}_r^B}^{(\mathbf{u},\mathbf{v})} (\check{X}^i) \right),\ \check{X}^0 = \overline{\triangle}_k,\ i = 0, 1, \ldots. \tag{3.13}$$

Here the $\Pi_{\mathcal{H}^B}^{(\mathbf{u},\mathbf{v})}(\overline{X})$ is the projection of each block $X^{(j)}$ in $\overline{X}$ onto its corresponding Hankel matrix space $\mathcal{H}^j$. That is,

$$\Pi_{\mathcal{H}^B}^{(\mathbf{u},\mathbf{v})}(\overline{X}) = [\Pi_{\mathcal{H}^1}^{(\mathbf{u},\mathbf{v}_{(1)})}(X^{(1)}), \ldots, \Pi_{\mathcal{H}^n}^{(\mathbf{u},\mathbf{v}_{(n)})}(X^{(n)})],$$

where $\mathbf{v}_{(i)}$ is the sub-vector of $\mathbf{v}$ consisting the $(K_1 + \cdots + K_{(i-1)} + 1)$-*th* to the K_i-*th* elements.

We stop the alternating projection if the following conditions hold

$$\frac{|f(\check{X}^{i+1}) - f(\check{X}^i)|}{max\{1, f(\check{X}^i)\}} \leq tol1, \quad or \quad \frac{\|\check{X}^{i+1} - \check{X}^i\|_F}{\|\check{X}^i\|_F} \leq tol2.$$

We end this section by the following remark.

Remark 6. One needs to normalize or standardize each series $\mathbf{y}^{(i)}$ before applying MSMM. This is because different scales will make effect in the analysis of the result. Furthermore, the series with large scales would have more effect on the noise reduction and therefore will render the noise reduction effect of other series with smaller scales.

4 Choosing Two Parameters: the Window Length L and the Rank of Hankel Matrix r

In this section, we will show how to choose two parameters in both SMM and MSMM: the window length L and the rank of the Hankel matrix.

4.1 The Window Length L

In both SMM and MSMM, a crucial step is to conduct singular value decomposition for the trajectory matrix. As a result, we want such decomposition to take less computational time. This can be achieved by choosing an L that makes the trajectory matrix more squared-like by full-rank assumption in Assumption 1. Below we present the optimal L for SMM and MSMM.

For the univariate case, the optimal window length with regard to recovering the true signal is achieved when $L = \lfloor (N+1)/2 \rfloor$, where $\lfloor a \rfloor$ denotes the largest integer that is no larger than a [23, Chapter 2].

For multivariate case, the proper window length varies dramatically due to different research purpose. Mathematically speaking, in our multivariate series $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(n)}$, the window length can be chosen between 2 and $\min_{i=1,\ldots,n} (N_i - 1)$. If all time series $\mathbf{y}^{(i)}$, $i = 1, \ldots, n$ are the same length, i.e., $N_1 = \cdots = N_n = N$, we have the following result which comes from Section 4.1 of [11]. To save space, we omit the proof here.

Theorem 4.1 ([11]). *The optimal value of window length L for multivariate case with each series has the same length N is:*

$$L = \frac{n}{n+1}(N+1).$$

In practical situation, it is pointed out that the choice of $L \approx N/2$ is still applicable for most situations [23, Chapter 4].

Based on above analysis, we conclude that for multivariate series with same patterns, we can choose $L = n(N+1)/(n+1)$ as suggested by Theorem 4.1, wheres for unpatterned multivariate case, we can still use $L \approx N/2$ as the window length.

4.2 The Upper Bound for Low-Rank Matrices

In this part, we discuss how to choose the upper bound r for low-rank matrices.

In both SMM and MSMM, we require the trajectory matrix to be a low-rank matrix. The underlying reason is that the signal we wish to extract corresponds to a low-rank matrix,

in other words, such series is a finite rank series. The definition of finite-rank series is given below.

Definition 4.2 ([23])**.** For time series $\mathbf{s}$ of arbitrary length, we call it finite-rank series if and only if the rank of its trajectory matrix does not change with the variation of series length or window length.

Definition 4.2 implies that the rank of the trajectory matrix for a finite-rank series is finite and fixed.

The importance of choosing a proper upper of the rank r in our noise reduction method is as follows. If the upper bound is smaller than the true rank, then significant information might be lost. If it is much larger than the true rank, the one may include some noises, which will not make the noise reduction method work well.

For univariate case, the rank of trajectory matrix could be estimated by several ways, such as singular values, scatter diagram of eigenvectors or weighted correlation and so on [24]. To analyze the multivariate case, we further need the property of linear recurrence relationship (LRR), which is defined as follows.

Definition 4.3 ([13])**.** We call $\mathbf{s} \in \mathcal{R}^N$ is of LRR if and only if there are $a_1, \ldots, a_t$, which satisfies:

$$s_{k+t} = \sum_{i=1}^{t} a_i s_{k+t-i},\ 1 \le k \le N-1. \tag{4.1}$$

Moreover, we refer t as the order of this LRR and $a_1, \ldots, a_t$ are the coefficients. If there is a minimal t, denoted as r, that makes LRR hold, we call this LRR the minimum linear recurrence relationship.

Another important fact is that, for infinite series, LRR is equivalent to finite-rank series [24, Chapter 5], which is stated below.

Proposition 4.4. *For infinite time series with finite rank d, it satisfies minimal LRR with order d as well.*

Some notations related to LRR include the polynomial $P_t(\mu) = \mu^t - \sum_{i=1}^{t} a_i \mu^{t-i}$, which is called the characteristic polynomial function of the LRR (4.1), and the roots of it, which are called the characteristic roots [13]. In addition, the roots of the characteristic polynomial corresponding to the minimum LRR is referred to as signal roots [23]. Since the time series that our paper works on is the finite-rank series, which is series governed by LRR, we can check the similarity among the trajectory space of our series collection $\{\mathbf{s}^{(1)}, \ldots, \mathbf{s}^{(n)}\}$ by using some characteristics of LRR. Below we demonstrate the relation between the trajectory space and characteristics of LRR for finite-rank series. Let $\mathbf{s}$ be the time series governed by LRR and Ω be the trajectory space of $\mathbf{s}$. We denote the dimension of Ω by r which is the rank of the trajectory matrix S. We write the minimal LRR of $\mathbf{s}$ as follows:

$$s_{k+r} = \sum_{i=1}^{r} a_i s_{k+r-i},$$

where the coefficients $a_r \neq 0$. The corresponding characteristic polynomial is: $P_r(\mu) = \mu^r - \sum_{i=1}^{r} a_i \mu^{r-i}$, and the roots of the characteristic polynomial is called signal roots. Let p be the number of different roots of the polynomial $P_r(\mu)$, μ_i be the *i-th* root of $P_r(\mu)$ and k_i be the multiplicity of the root μ_i. Then using the theory of finite rank time series [24], there is:

$$k_1 + \cdots + k_p = r,$$

which means the sum of the multiplicity of different signal roots equals to the rank of the trajectory matrix of $\mathbf{s}$. In fact, the structure of each series could be characterized by its signal roots, which is reflected in its own trajectory space. As a result, we can determine the structure of multivariate series by checking the signal roots and its multiplicity of each series in $\mathbf{s}^{(1)}, \ldots, \mathbf{s}^{(n)}$. Such observation leads to the following approach to determine the rank of trajectory matrix for multivariate signals.

To determine the rank of the trajectory matrix for multivariate time series $\{\mathbf{s}^{(1)}, \ldots, \mathbf{s}^{(n)}\}$. Denote $\overline{S} = [S^{(1)}, \ldots, S^{(n)}]$ as the trajectory matrix for $\{\mathbf{s}^{(1)}, \ldots, \mathbf{s}^{(n)}\}$, where $S^{(i)} \in \mathcal{R}^{L \times K_i}$ is the trajectory matrix for $\mathbf{s}^{(i)}$. Let $\Omega_i = span\{\xi_1^{(i)}, \ldots, \xi_{l_i}^{(i)}\}$ be the trajectory space of time series $\mathbf{s}^{(i)}$, with dimension $l_i, i = 1, \ldots, n$. Here $\xi_1^{(i)}, \ldots, \xi_{l_i}^{(i)}$ are linearly independent vector which spans Ω_i.

In fact, Ω_i is the column space of the trajectory matrix $S^{(i)}$ of $\mathbf{s}^{(i)}$. Let Ω be the trajectory space of the multivariate series $\{\mathbf{s}^{(1)}, \ldots, \mathbf{s}^{(n)}\}$. There is

$$\Omega = span\{\xi_1^{(1)}, \ldots, \xi_{l_1}^{(1)}, \xi_1^{(2)}, \ldots, \xi_{l_2}^{(2)}, \ldots, \xi_1^{(n)}, \ldots, \xi_{l_n}^{(n)}\}.$$

The dimension of Ω is exactly the rank of the trajectory matrix $\overline{S}$.

We need the following assumption.

Assumption 4.1. Each time series $\mathbf{s}^{(i)}, i = 1, \ldots, n$ satisfies its own LRR.

Let r_i be the rank of $\mathbf{s}^{(i)}, i = 1, \ldots, n$. There is, $r_i = rank(S^{(i)}) = dim(\Omega_i) = l_i,\ i = 1 \ldots, n$. Let $P_{r_i}^{(i)}$ be the characteristic polynomial function of minimum LRR of $\mathbf{s}^{(i)}$. Let $\mu_1, \ldots, \mu_t$ be the distinct signal roots of all characteristic polynomial function $P_{r_1}^{(1)}, \ldots, P_{r_n}^{(n)}$, and the corresponding largest multiplicity of the signal roots are $k_1, \ldots, k_t$. In other words, k_i is the largest multiplicity of μ_i among all characteristic polynomial functions. Then the rank of $\{\mathbf{s}^{(1)}, \ldots, \mathbf{s}^{(n)}\}$ is given by (see [22] for more details):

$$r = \sum_{i=1}^{l} k_i. \tag{4.2}$$

In other words, r is the sum of the maximum multiplicity of each signal root.

We summarize the above process in the following algorithm.

Algorithm 3: The upper bound r of low-rank matrices for multivariate series

1 Initialization $\mathbf{s}^{(1)}, \ldots, \mathbf{s}^{(n)}$;
2 Calculate the distinct signal roots of $P_{r_i}^{(i)},\ i = 1, \ldots, n$, denoted as $\mu_1, \ldots, \mu_t$;
3 Calculate the largest multiplicity of each μ_i, denoted as $k_i,\ i = 1, \ldots, t$;
4 Let r be given by (4.2). Output r.

Algorithm 3 shows how to determine the rank of the trajectory matrix for true signals $\mathbf{s}^{(1)}, \ldots, \mathbf{s}^{(n)}$. When dealing with noised signals $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(n)}$, we need to give an estimation of r of the underlying true signals $\mathbf{s}^{(1)}, \ldots, \mathbf{s}^{(n)}$. In this case, we can first find the corresponding rank of each series $\mathbf{y}^{(i)}$, and estimate the signal roots by ESPRIT method and Algorithm 3.2 in [23]. Then use (4.2) to obtain the upper bond of the rank of the trajectory matrix of $\mathbf{s}^{(1)}, \ldots, \mathbf{s}^{(n)}$.

5 Two Variants of MSMM

In this part, we will propose two variants of MSMM, which can be used for weak signal extraction.

Weak signal means that the true signal is inundated by the noise. In other words, it means the signal noise ratio (SNR) is low. Weak signal arises in many situations: radar, sonar, earthquake detection as so on [14, 29, 12]. To extract weak signals, many special models and numerical algorithms are proposed [28, 7, 20]. In this paper, we will discuss how to extend our proposed MSMM for weak signal extraction.

Assume the noised univariate weak signal is $\mathbf{y} \in \mathcal{R}^N$, which has low SNR.

To denoise the weak signals of MSMM, the main idea is to use the historical denoised series as the additional time series, and combine them with the noisy signal to obtain a multivariate series, then we can apply MSMM on the resulting multivariate series. The reason we try to make use of the historical denoised series is that MSMM has the ability to extract similar structure among series, and the historical denoised series can provide more accurate information about the true signals.

Specifically, let $\mathbf{d}^{(k)}$ be the denoised data provided by the current iteration. To make use of the information of current iteration, we can construct a multivariate series $(\mathbf{y}, \mathbf{d}^{(k)}) \in \mathcal{R}^N \times \mathcal{R}^N$, and apply the proposed MSMM to do further noise reduction. This will lead the first variant of MSMM: recycling MSMM. In other words, we do not drop the information $\mathbf{d}^{(k)}$, but recycle it to make further use of it. The resulting algorithm is demonstrated in Algorithm 4.

The other way is to make use of all the previous information to construct a multiplicity series. That is, at k-*th* iteration, we already have the previous denoise series $\mathbf{d}^{(k)}, \mathbf{d}^{(k-1)}, \ldots, \mathbf{d}^{(1)}$, and we construct a multivariate series $(\mathbf{y}, \mathbf{d}^{(k)}, \mathbf{d}^{(k-1)}, \ldots, \mathbf{d}^{(1)}) \in \mathcal{R}^N \times \mathcal{R}^N \times \mathcal{R}^N \times \ldots \mathcal{R}^N$. Then we apply MSMM to the $(k+1)$ dimension series. It leads to the second variant: the variate accumulation MSMM, which is demonstrated in Algorithm 5.

Algorithm 4: recycling MSMM algorithm

1 Initialization: $\mathbf{y} \in \mathcal{R}^N$ to be denoised; window length L, rank r;
2 Apply SMM to $\mathbf{y}$, and let the returned signal be $\mathbf{d}^{(0)} \in \mathcal{R}^N$, $k := 0$;
3 Apply MSMM to $(\mathbf{y}, \mathbf{d}^{(k)})$, to get the denoised univariate signal $\mathbf{d}^{(k+1)} \in \mathcal{R}^N$;
4 If the stopping criterion is satisfied, stop, output $\mathbf{d}^{(k+1)}$; otherwise, $k := k+1$, go to Step 2;

Algorithm 5: variate accumulation MSMM algorithm

1 Initialization: $\mathbf{y} \in \mathcal{R}^N$ to be denoised; window length L, rank r;
2 Apply SMM to $\mathbf{y}$, and let the returned signal be $\mathbf{d}^{(0)} \in \mathcal{R}^N$, $k := 0$;
3 Combine the noisy signal $\mathbf{y}$ and all the previous denoised signals $\mathbf{d}^{(0)}, \ldots, \mathbf{d}^{(k)}$ to make a multivariate series $(\mathbf{y}, \mathbf{d}^{(0)}, \ldots, \mathbf{d}^{(k)})$. Apply MSMM to the above multivariate time series. Let the returned signal be $\mathbf{d}^{(k+1)}$;
4 If the stopping criterion is satisfied, stop, output $\mathbf{d}^{(k+1)}$; otherwise, $k := k+1$, go to 2;

We end this section by some remarks.

Remark 7. As for the stopping criteria, we compare the difference of two consecutive denoised signals:

$$\frac{\|\mathbf{d}^{(k)} - \mathbf{d}^{(k-1)}\|_F}{max\{1, \|\mathbf{d}^{(k)}\|_F\}} \leq tol,$$

where tol is specified by users.

Remark 8. In step 2 of Algorithm 4 and Algorithm 5, one can use any method for univariate noise reduction, such as SMM. Other univariate noise reduction can also be used such as Wavelet denoising and random phase noise reduction and so on [5, 19, 20, 27, 30, 31, 32, 33].

6 Numerical Results

In this part, we will conduct numerical experiments to see the performance of the proposed methods. It is divided into two parts. In the first part, we test MSMM method and compare with other existing methods [22]. In the second part, we test recycling MSMM and variate accumulation MSMM. All the experiments in this section is conducted in Matlab R2018b on a MacBook Air with a Intel Core i5 CPU at 1.6 GHz and 8GB of RAM.

6.1 Performance of MSMM

In this part, we compare MSMM with three methods: SSA (Singular Spectrum Analysis) [25], SMM [8] and MSSA (Multivariate Singular Spectrum Analysis) [11]. The multivariate series $\mathbf{y}^{(i)}$ is constructed as in (1.1). In our test, we set $N_1 = N_2 = 71$, $n = 2$, $\mathbf{r}^{(1)}$ and $\mathbf{r}^{(2)}$ are the realization of two independent Gaussian white noise with variance $\sigma^2 = 25$. $\mathbf{s}^{(1)}$ and $\mathbf{s}^{(2)}$ are two harmonic signals generated in the following ways.
E1: The two signals have the same period, different amplitude, and the difference between the two phases is not $\frac{\pi}{2}$:

$$\mathbf{s}_k^{(1)} = 30cos(2\pi k/12), \quad \mathbf{s}_k^{(2)} = 30cos(2\pi k/12 + \pi/4),\ k = 1, \ldots, N.$$

E2: The two signals have the same period and amplitude, and the difference between phases is $\frac{\pi}{2}$:

$$\mathbf{s}_k^{(1)} = 30cos(2\pi k/12), \quad \mathbf{s}_k^{(2)} = 30cos(2\pi k/12 + \pi/2),\ k = 1, \ldots, N.$$

E3: The two signals have different period:

$$\mathbf{s}_k^{(1)} = 30cos(2\pi k/12), \quad \mathbf{s}_k^{(2)} = 30cos(2\pi k/8 + \pi/2),\ k = 1, \ldots, N.$$

The dimension of the trajectory space of each univariate signal in the above tests is 2, implying that the dimension of trajectory matrix is 2 for each univariate signals. However, the trajectory spaces of $(\mathbf{s}^{(1)}, \mathbf{s}^{(2)})$ are different. Table 1 shows the rank of the trajectory matrix in different methods.

We use RMSE (root mean square error) to represent the error between the estimated signals $\widehat{\mathbf{s}^{(i)}}$ and the true signals $\mathbf{s}^{(i)}$, which is given by $RMSE^{(i)} = \sqrt{\sum_{j=1}^{N}(\widehat{\mathbf{s}_j^{(i)}} - \mathbf{s}_j^{(i)})^2}$ $i = 1, 2$. $RMSE = \frac{1}{2}RMSE^{(1)} + \frac{1}{2}RMSE^{(1)}$.

Table 1: the dimension of signal trajectory space

	E1	E2	E3
SSA	2	2	2
SMM	2	2	2
MSSA	2	2	4
MSMM	2	2	4

We randomly generated 1000 realizations of $(\mathbf{y}^{(1)}, \mathbf{y}^{(2)})$ for each test and report the averaged results in Table 2, where the best noise reduction for each method over different window lengths is marked in bold. The best result among different methods and window lengths for each fixed experiment is underlined. We also report the cputime in second for each method.

Table 2: RMSE for E1-E3

		L=12	L=24	L=36	L=48	L=60	time
E1	SSA	3.25	2.01	**2.00**	2.01	2.01	0.08
	SMM	1.22	**1.16**	**1.16**	**1.16**	1.22	0.09
	MSSA	3.18	1.82	1.58	**1.46**	1.99	0.08
	MSMM	1.20	1.11	1.11	**1.09**	1.15	0.11
E2	SSA	3.25	2.01	**2.00**	2.01	3.25	0.08
	SMM	1.22	**1.16**	1.17	**1.16**	1.22	0.08
	MSSA	3.18	1.81	1.59	**1.48**	1.98	0.08
	MSMM	1.16	1.01	0.99	**0.98**	1.03	0.12
E3	SSA	3.22	2.01	**2.00**	2.01	3.22	0.09
	SMM	1.21	**1.16**	**1.16**	**1.16**	1.21	0.09
	MSSA	6.89	3.75	3.09	**2.86**	3.85	0.08
	MSMM	2.04	1.51	1.45	**1.43**	1.47	0.13

From Table 2, we have the following observations.

1. One can observe that for univariate methods: SSA and SMM, the window length L and $N - L + 1$ have the same noise reduction effect in terms of RMSE. However, for multivariate methods MSSA and MSMM, the RMSEs are different when choosing different window lengths. Moreover, we can see that the noise reduction result is better when window length L is larger than $N/2$. In particular, for multivariate methods, the best RMSE is obtained when $L = 48$, which is $2N/3$.

2. Comparing SMM with SSA, it seems that SMM leads to smaller RMSE than the traditional method SSA. Same conclusions are also valid for multivariate methods, meaning that MSMM performs better than MSSA.

3. In terms of different examples, in E1 and E2, the two series $\mathbf{s}^{(1)}$ and $\mathbf{s}^{(2)}$ have the same data structure, meaning that the trajectory spaces of $\mathbf{s}^{(1)}$ and $\mathbf{s}^{(2)}$ coincide with each other. Comparing the results in Table 2 for E1 and E2, the multivariate methods have better noise reduction than the univariate methods. In particularly, MSMM has the best performance in denoising. Based on such observation, we can see that when there are some similarities among series in multivariate time series, our MSMM will produce better denoising effect and is more efficient.

4. For E3, the structures of $\mathbf{s}^{(1)}$ and $\mathbf{s}^{(2)}$ are completely different. In this case, SMM is the best.

5. In terms of cputime, MSMM takes a bit longer time than other methods since it deals with multivariate series which involves a larger matrix than other methods.

We are also interested in how MSMM performs with unpatterned series. In order to compare how well the MSMM works with unpatterned series, we use the series in E4 and E5 as shown below with $N_1 = 72$ and $N2 = 60$.

E4: (Unpatterned case of E1). The two signals have the same period, different amplitude, and the difference between the two phases is not $\frac{\pi}{2}$:

$$\mathbf{s}_k^{(1)} = 30cos(2\pi k/12),\ k = 1, \ldots, N_1, \quad \mathbf{s}_k^{(2)} = 30cos(2\pi k/12 + \pi/4),\ k = 1, \ldots, N_2.$$

E5: (Unpatterned case of E2). The two signals have the same period and amplitude, and the difference between phases is $\frac{\pi}{2}$:

$$\mathbf{s}_k^{(1)} = 30cos(2\pi k/12),\ k = 1, \ldots, N_1, \quad \mathbf{s}_k^{(2)} = 30cos(2\pi k/12 + \pi/2),\ k = 1, \ldots, N_2.$$

We randomly generated 1000 realizations of $(\mathbf{y}^{(1)}, \mathbf{y}^{(2)})$ for each test and report the averaged results in Table 1.

Table 3: RMSE for E4 and E5

		L=24	L=36	L=48
E4	SMM	1.19	1.20	1.20
	MSMM	1.13	1.10	1.11
E5	SMM	1.21	1.22	1.21
	MSMM	1.02	1.01	1.01

From Table 3, we observe that MSMM still outperforms SMM even with the unpatterned series. Comparing the result in Table 3 and the corresponding MSMM result in Table 2 from our manuscript, we find that the unpatterned ones performs slightly worse than the patterned ones, and this is reasonable since when working with unpatterned series there are more challenges in the sense that it is very difficult to choose the optimal window length. However, MSMM still works well on the unpatterned series (the differences between the RMSE result for unpatterned and patterned ones are at most 0.01).

6.2 Weak signal extraction

In this part, we test the recycling MSMM (MSMM-C1) and variate accumulation MSMM (MSMM-C2) on weak signal extraction.

We generate super-positional sine signals with amplitude 0.1 and frequency 5 and 10 respectively. The noise is the Gaussian noise with mean 0 and variance 0.5. We sample the signal in 20 seconds with interval 0.01s, resulting in a length of $N = 2000$ time series. The SNR is -32.188dB in this case. Figure 1 displays the real signal $\mathbf{s}$ and noisy signal $\mathbf{y}$ respectively.

We apply SMM, MSMM-C1 and MSMM-C2 to the signal $\mathbf{y}$. The results are reported in Figure 2 and Table 4. It can be seen that MSMM-C1 and MSMM-C2 show an significant improvement over SMM. In particular, MSMM-C2 achieves an RMSE 0.0211. The denoised results are further shown in Figure 2, where one can see that the signal returned by MSMM

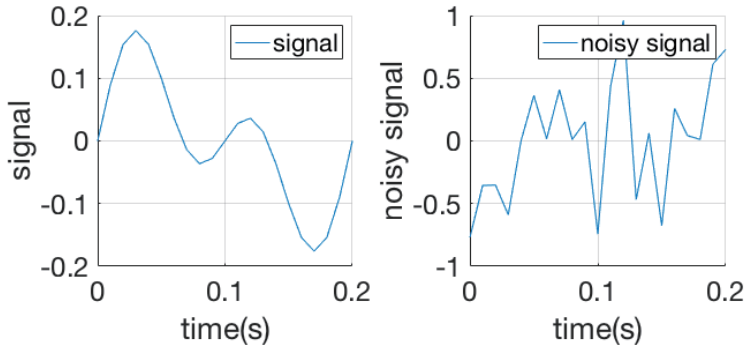

Figure 1: The original signal and the noisy signal

is close to that true signal. The reason that MSMM-C2 outperforms MSMM-C1 is that MSMM-C2 makes use of all previous denoised information, which enable us to better extract the structure of the true signal. However, MSMM-C2 maybe time-consuming for large series. In such situation, one need a tradeoff between MSMM-C1 and MSMM-C2.

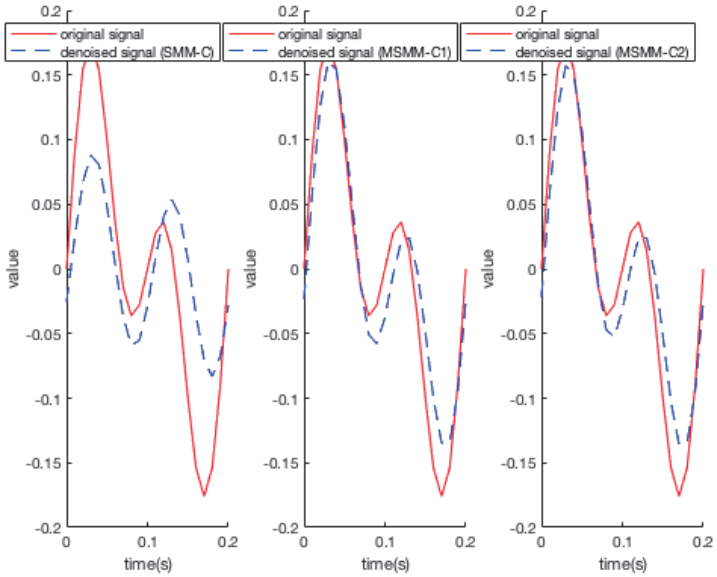

Figure 2: The noise reduction results for three methods

In some situations of weak signal recovery, one may want to recover the frequency from the weak siganls. Figure 4 demonstrates the frequency of recovered signals, where one can see that both MSMM-C1 and MSMM-C2 can return a good estimation of frequency.

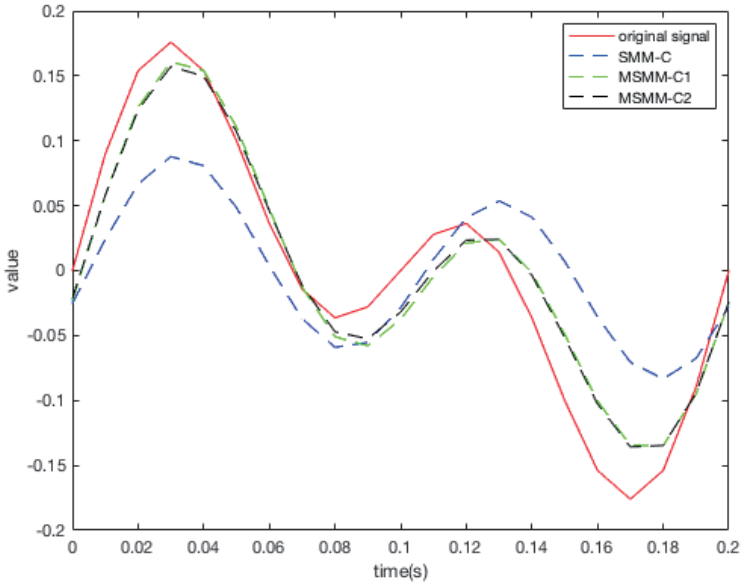

Figure 3: The noise reduction result from 0 to 0.2s

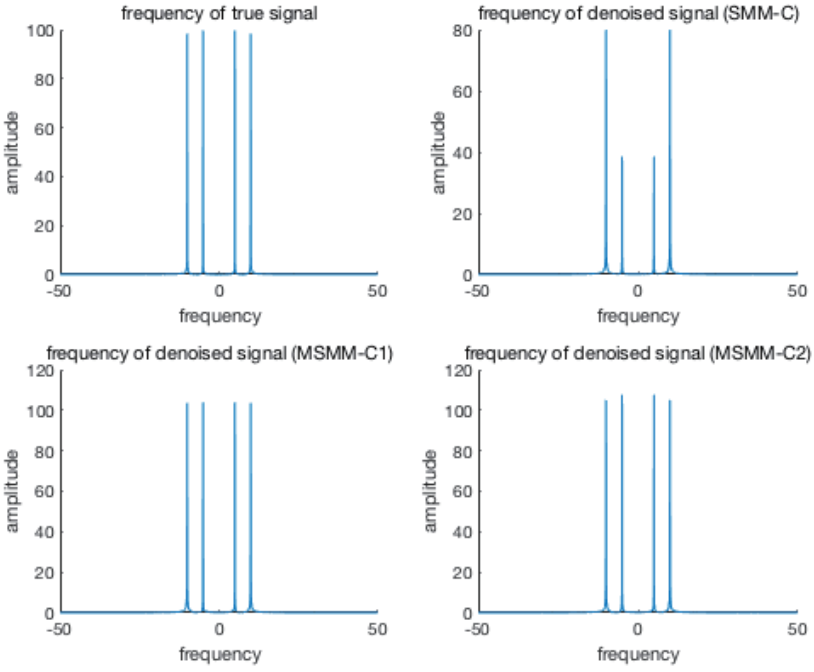

Figure 4: The frequency of noise series and denoised signal

Table 4: RMSE of noise reduction

	SMM-C	MSMM-C1	MSMM-C2
RMSE	0.0536	0.0267	0.0211

7 Conclusions

In this paper, we proposed MSMM to denoise multivariate signals. Based on MSMM, we designed two new denosing schemes: recycling MSMM and variate accumulation MSMM, which are particularly suitable for weak signal extraction. We also provide how to choose the window length and the upper bound for the rank of the trajectory matrices in MSMM. Numerical results revealed that MSMM outperforms the traditional denoising methods, and the two variants have impressive performance in recovering weak signals.

8 Acknowledgement

We would like to thank the two anonymous referees who provided valuable comments based on which we further improved our paper.

References

[1] A. Groth and M. Ghil, Multivariate singular spectrum analysis and the road to phase synchronization, *Phys. Rev. E.* 3 (2011) 036206.

[2] A. Patton, Copula methods for forecasting multivariate time series, *Handbook of economic forecasting.* (2013) 899–960.

[3] A. J. Patton, Estimation of multivariate models for time series of possibly different lengths, *J Appl Econ.* 2 (2006) 147–173.

[4] A. Hannachi, I. Jolliffe and D. Stephenson, Empirical orthogonal functions and related techniques in atmospheric science: A review, *Int J Climatol.* 9 (2007) 1119–1152.

[5] D. Griffin and J. Lim, Signal estimation from modified short–time fourier transform, *IEEE Trans. Signal Process.* 2 (1984) 236–243.

[6] F. Alonso, J. Del Castillo and P. Pintado, Application of singular spectrum analysis to the smoothing of raw kinematic signals, *J. Biomech.* 5 (2005) 1085–1092.

[7] G.-Q. Zhu, K. Ding, Y. Zhang and Y. Zhao, Experimental research of weak signal detection based on the stochastic resonance of nonlinear system, *Acta Physica Sinica* 5 (2010) 3001–3006.

[8] H.-D. Qi, J. Shen and N. Xiu, A sequential majorization method for approximating weighted time series of finite rank, *Stat. Interface* 4 (2018) 615–630.

[9] H. Hassani and D. Thomakos, A review on singular spectrum analysis for economic and financial time series, *Stat. Interface.* 3 (2010), 377–397.

[10] H. Hassani, S. Heravi and A. Zhigljavsky, Forecasting european industrial production with singular spectrum analysis, *Int. J. Forecast.* 1 (2009) 103–118.

[11] H. Hassani and R. Mahmoudvand, Multivariate singular spectrum analysis: A general view and new vector forecasting approach, *International Journal of Energy and Statistics.* 01 (2013) 55–83.

[12] H. Li, R. Wang, S. Cao, Y. Chen, N. Tian and X. Chen, Weak signal detection using multiscale morphology in microseismic monitoring, *J Appl Geophy.* (2016) 39–49.

[13] J. D. Hamilton, *Time series analysis.* Princeton university press, 2020.

[14] J. Yan and L. Lu, Improved hilbert–huang transform based weak signal detection methodology and its application on incipient fault diagnosis and ecg signal analysis, *Signal Processing.* (2014) 74–87.

[15] J. Gillard and A. A. Zhigljavsky, Weighted norms in subspace–based methods for time series analysis, *Numer.* 5 (2016) 947–967.

[16] K. Holden, Vector auto regression modeling and forecasting, *J Forecast.* 3 (1995) 159–166.

[17] K. H. Esbensen, D. Guyot, F. Westad and L. P. Houmoller, *Multivariate data analysis: in practice: an introduction to multivariate data analysis and experimental design.* Multivariate Data Analysis, 2002.

[18] K. W. Chau and C. Wu, A hybrid model coupled with singular spectrum analysis for daily rainfall prediction, *J. Hydroinformatics.* 61 (2019) 195–203.

[19] M. Akin, Comparison of wavelet transform and fft methods in the analysis of eeg signals, *J. Med. Syst.* 3 (2002) 241–247.

[20] M. Lang, H. Guo, J. E. Odegard, C. S. Burrus and R. O. Wells, Noise reduction using an undecimated discrete wavelet transform, *IEEE Signal Process. Lett.* 1 (1996) 10–12.

[21] N. Cliff, *Analyzing multivariate data,* Harcourt Brace Jovanovich, 1987.

[22] N. Golyandina, A. Korobeynikov, A. Shlemov and K. Usevich, Multivariate and 2d extensions of singular spectrum analysis with the rssa package, *arXiv preprint arXiv:1309.5050* (2013).

[23] N. Golyandina, A. Korobeynikov and A. Zhigljavsky, *Singular spectrum analysis with R.* Springer, 2018.

[24] N. Golyandina, A. Korobeynikov and A. Zhigljavsky, *Analysis of time series structure: SSA and related techniques.* CRC press, 2001.

[25] N. Golyandina and A. Zhigljavsky, Singular Spectrum Analysis for time series, *Springer Science & Business Media,* 2013.

[26] N. Zvonarev and N. Golyandina, Iterative algorithms for weighted and unweighted finite–rank time–series approximations, *Stat Interface.* 1 (2017) 5–18

[27] P. Flandrin, G. Rilling and P. Goncalves, Empirical mode decomposition as a filter bank, *IEEE Signal Process. Lett.* 2 (2004) 112–114.

[28] Q. He and J. Wang, Effects of multiscale noise tuning on stochastic resonance for weak signal detection, *Digital Signal Processing.* 4 (2012) 614–621.

[29] S. M. Mousavi, W. Zhu, Y. Sheng and G. C. Beroza, Cred: A deep residual network of convo- lutional and recurrent units for earthquake signal detection, *Sci. Rep.* 1 (2019) 1–14.

[30] T.-W. Lee, M. S. Lewicki and T. J. Sejnowski, Ica mixture models for unsupervised classification of non–gaussian classes and automatic context switching in blind signal separation. *IEEE PAMI* 10 (2000) 1078–1089.

[31] S. Nawab, T. Quatieri and J. Lim, Signal reconstruction from short-time fourier transform magnitude, *IEEE Trans Acoust* 4 (1983) 986–998.

[32] S. Sardy, P. Tseng and A. Bruce, Robust wavelet denoising, *AADA* 6 (2001) 1146–1152.

[33] Z. Wu, and N. E. Huang, Ensemble empirical mode decomposition: a noise-assisted data analysis method, *Optim. Lett.* 01 (2009) 1–41.

Manuscript received 21 June 2021
revised 30 August 2021
accepted for publication 19 September 2021

Aoxiang Chen
School of Mathematics and Statistics
Beijing Institute of Technology
Beijing, 100081, P. R. China
E-mail address: e0572388@u.nus.edu

Qingna Li
School of Mathematics and Statistics
Beijing Key Laboratory on MCAACI
Key Laboratory of Mathematical Theory and
Computation in Information Security
Beijing Institute of Technology. Beijing
100081, P. R. China
E-mail address: qnl@bit.edu.cn