Computational Acceleration and Smart Initialization of Full-rank Spatial Covariance Analysis

Hiroshi Sawada, Rintaro Ikeshita, Nobutaka Ito, and Tomohiro Nakatani
NTT Communication Science Laboratories, NTT Corporation, Kyoto, Japan

***Abstract*—Full-rank spatial covariance analysis (FCA) is a method for blind source separation. It is based on a model for observation mixtures with flexible source-related parameters, and an EM algorithm is known to optimize the parameters. FCA has the potential to obtain high-quality separations. However, the algorithm for FCA is computationally demanding and sensitive to initializations. This paper proposes two practical techniques to make effective use of FCA. The first one is to accelerate the execution of the algorithm by using single-instruction-multiple-data (SIMD) instructions run on a GPU. The second one is to initialize the parameters appropriately by scanning the observation mixtures. Experimental results show that high-quality separations were achieved for 6-second real-room speech mixtures (4 sources and 3 microphones) with a computational time of less than 8 seconds.**

***Index Terms*—blind source separation (BSS), full-rank spatial covariance analysis (FCA), expectation-maximization (EM) algorithm, matrix inversion, single instruction multiple data (SIMD)**

I. Introduction

Blind source separation (BSS) has been studied for a long time as various signal processing and machine learning methods [1–5]. Independent component analysis (ICA) [2] is perhaps the most popular one where observation mixtures are linearly transformed into separated signals, and the statistical properties (e.g., independence and non-Gaussianity) of the transformed signals are optimized. ICA assumes that the mixing system is invertible. On the other hand, full-rank spatial covariance analysis (FCA) [6–9] models observed multichannel mixtures with a more general mixing system than ICA does, which consists of full-rank spatial covariance matrices and therefore is not necessarily invertible. Because of this flexibility, FCA has several merits over ICA, such as it can be applied to an underdetermined case where the number N of sources is larger than the number M of observation channels. There have been proposed extensions to richer models [10–15] based on FCA. However, there are prices we have to pay when performing FCA: demanding computation for the optimization and sensitivity to initialization. They prevent us from the practical usage of FCA. In this paper, we propose two practical techniques that solve the two issues of FCA.

The first one is regarding the acceleration of optimization by fully using single-instruction-multiple-data (SIMD) instructions, which are especially effective with a graphics processing unit (GPU). A popular algorithm for FCA has been recognized based on expectation-maximization (EM) [6, 7]. There are many $M \times M$ matrices whose inverse matrices need to be calculated in the EM algorithm, and the proposed SIMD technique accelerates those inverse matrix calculations. An approach [8, 9] has already been proposed for accelerating FCA by assuming that all the spatial covariance matrices can be jointly diagonalized. The assumption exactly holds when the number of sources is equal to two, but in general it approximates the model of FCA. The proposed SIMD approach, on the other hand, optimizes the model parameters exactly by following a full-rank spatial covariance model with no approximations.

The second technique is regarding the initialization of model parameters. A simple but moderately effective way is to randomly initialize the parameters. However, it does not consider and exploit the characteristic of observation mixtures. In this paper, we propose a simple method to initialize spatial covariance matrices, a dominant part of the model parameters. The method is based on online clustering that scans the observation mixtures, and tries to exploit the observation characteristic for obtaining good initial ones that will be converged close to the true spatial covariance matrices of sources.

This paper is organized as follows. Section II first describes an FCA mixture model, and then the corresponding EM algorithm especially with clarifying its computationally demanding parts. Sections III and IV explain the proposed acceleration and initialization techniques, respectively. Section VI reports experimental results from which we recognize how FCA was accelerated especially by a GPU and how the initialization contributed to high-quality separations.

II. Full-rank spatial Covariance Analysis

A. Model and objective function

Suppose that $n = 1, \dots, N$ sources are mixed and observed at $m = 1, \dots, M$ sensors (e.g., microphones in an audio case). Let the sensor observations at a time indexed by t, $t = 1, \dots, T$, be denoted by an M-dimensional complex vector $\mathbf{x}_t \in \mathbb{C}^M$ with $\mathbf{x}_t = [x_{1t}, \dots, x_{Mt}]^\mathsf{T}$. In FCA, a mixture vector $\mathbf{x}_t$ follows a zero-mean multivariate complex Gaussian distribution

$$p(\mathbf{x}_t|\mathbf{0}, \hat{\mathsf{X}}_t) \propto \frac{1}{\det \hat{\mathsf{X}}_t} \exp\left(-\mathbf{x}_t^\mathsf{H} \hat{\mathsf{X}}_t^{-1} \mathbf{x}_t\right) \tag{1}$$

with a covariance matrix

$$\hat{\mathsf{X}}_t = \sum_{n=1}^{N} v_{nt} \mathsf{A}_n + \mathsf{B} \tag{2}$$

parameterized with $M \times M$ matrices $\{\mathsf{A}_n\}_{n=1}^N$, B, and nonnegative scalars $\{\{v_{nt}\}_{n=1}^N\}_{t=1}^T$. Here, A_n is a spatial covariance

978-9-0827-9703-9/19/$31.00 ©2019 IEEE

matrix that encodes the spatial property from source n to all M sensors, and B is a noise covariance matrix. We assume A_n and B to be Hermitian and positive semidefinite. A scalar v_{nt} represents the temporal power of source n at time index t.

The parameters $\theta = \{\{\mathsf{A}_n\}_{n=1}^N, \mathsf{B}, \{\{v_{nt}\}_{n=1}^N\}_{t=1}^T\}$ can be optimized in a maximum likelihood sense, equivalently by minimizing the negative log-likelihood $\mathcal{C}(\theta) = -\log p(\{\mathbf{x}_t\}_{t=1}^T|\theta)$. We assume the likelihood is decomposed into time samples

$$p(\{\mathbf{x}_t\}_{t=1}^T|\theta) = \prod_{t=1}^T p(\mathbf{x}_t|\mathbf{0}, \hat{\mathsf{X}}_t)\,. \tag{3}$$

Substituting (1) into (3), we have the objective function

$$\mathcal{C}(\theta) = \sum_{t=1}^T \left[\mathbf{x}_t^{\mathsf{H}}\hat{\mathsf{X}}_t^{-1}\mathbf{x}_t + \log\det\hat{\mathsf{X}}_t\right] \tag{4}$$

to be minimized.

B. Source separation

Let $\mathbf{y}_{nt} \in \mathbb{C}^M$ and $\mathbf{b}_t \in \mathbb{C}^M$ be latent variables that satisfy

$$\mathbf{x}_t = \sum_{n=1}^N \mathbf{y}_{nt} + \mathbf{b}_t\,. \tag{5}$$

Once the parameters θ are optimized, separated signals and noises are obtained as the conditional expectations typically by the multichannel Wiener filters

$$\tilde{\mathbf{y}}_{nt} = \mathbb{E}[\mathbf{y}_{nt}|\mathbf{x}_t, \theta] = v_{nt}\mathsf{A}_n\hat{\mathsf{X}}_t^{-1}\mathbf{x}_t\,, \tag{6}$$

$$\tilde{\mathbf{b}}_t = \mathbb{E}[\mathbf{b}_t|\mathbf{x}_t, \theta] = \mathsf{B}\hat{\mathsf{X}}_t^{-1}\mathbf{x}_t\,, \tag{7}$$

respectively.

C. Expectation-Maximization (EM) algorithm

The objective function (4) with (2) can be minimized by an EM algorithm [6, 7].

E-step calculates the conditional expectations $\tilde{\mathsf{Y}}_{nt}$ and $\tilde{\mathsf{B}}_t$ of the outer product of latent vectors $\mathbf{y}_{nt}$ and $\mathbf{b}_t$ as

$$\tilde{\mathsf{Y}}_{nt} = \mathbb{E}[\mathbf{y}_{nt}\mathbf{y}_{nt}^{\mathsf{H}}|\mathbf{x}_t, \theta] = \tilde{\mathbf{y}}_{nt}\tilde{\mathbf{y}}_{nt}^{\mathsf{H}} + (\mathsf{I} - v_{nt}\mathsf{A}_n\hat{\mathsf{X}}_t^{-1})v_{nt}\mathsf{A}_n\,, \tag{8}$$

$$\tilde{\mathsf{B}}_t = \mathbb{E}[\mathbf{b}_t\mathbf{b}_t^{\mathsf{H}}|\mathbf{x}_t, \theta] = \tilde{\mathbf{b}}_t\tilde{\mathbf{b}}_t^{\mathsf{H}} + (\mathsf{I} - \mathsf{B}\hat{\mathsf{X}}_t^{-1})\mathsf{B}\,, \tag{9}$$

respectively.

M-step updates the model parameters by

$$v_{nt} \leftarrow \frac{1}{M}\mathrm{tr}\left(\mathsf{A}_n^{-1}\tilde{\mathsf{Y}}_{nt}\right)\,, \tag{10}$$

$$\mathsf{A}_n \leftarrow \frac{1}{T}\sum_{t=1}^T \frac{1}{v_{nt}}\tilde{\mathsf{Y}}_{nt}\,, \tag{11}$$

$$\mathsf{B} \leftarrow \frac{1}{T}\sum_{t=1}^T \tilde{\mathsf{B}}_t\,. \tag{12}$$

The EM algorithm iterates the **E-step** and **M-step** for a predefined number of times or until convergence with some criterion.

Algorithm 1 Sequential calculation of inverse matrices

```
procedure SeqMatInv
    for $t = 1$ to $T$ do
        $\hat{\mathsf{X}}_t^{-1} \leftarrow \mathrm{inv}(\hat{\mathsf{X}}_t)$
    end for
end procedure
```

D. Demanding Matrix Inverse Calculations

The EM algorithm involves the task of calculating the inverse matrices for all $\hat{\mathsf{X}}_t$, $t = 1, \ldots, T$. With an ordinary sequential computing model, we typically perform the procedure shown in **Algorithm** 1, where $\mathrm{inv}(\cdot)$ is a built-in function that calculates the inverse of a matrix. The procedure is computationally demanding especially when the number T of samples is large.

III. Acceleration using SIMD instructions

In this section, we propose to calculate many inverse matrices and other types of mathematical operations efficiently by using SIMD instructions.

A. Matrix Inverse

Suppose that we have T Hermitian matrices $\hat{\mathsf{X}}_t$, $t = 1, \ldots, T$, each of which has size $M \times M$. Concatenating them, we have a tensor $\mathcal{Q} = [\hat{\mathsf{X}}_1, \ldots, \hat{\mathsf{X}}_T]$ of size $M \times M \times T$. Let the results of calculating the inverses for all the T matrices be stored in a tensor $\mathcal{R} = [\hat{\mathsf{X}}_1^{-1}, \ldots, \hat{\mathsf{X}}_T^{-1}]$.

Let a Hermitian matrix be expressed in a block form

$$\hat{\mathsf{X}}_t = \begin{pmatrix} q & \mathbf{q}^{\mathsf{H}} \\ \mathbf{q} & \mathsf{Q} \end{pmatrix} \tag{13}$$

where q is a scalar, $\mathbf{q}$ is a vector, and Q is a matrix of size $(M-1)\times(M-1)$. We employ a formula regarding the inverse of a block matrix [16]

$$\begin{pmatrix} q & \mathbf{q}^{\mathsf{H}} \\ \mathbf{q} & \mathsf{Q} \end{pmatrix}^{-1} = \begin{pmatrix} q^{-1}\left(1 + q^{-1}\mathbf{q}^{\mathsf{H}}\mathsf{S}^{-1}\mathbf{q}\right) & -q^{-1}\mathbf{q}^{\mathsf{H}}\mathsf{S}^{-1} \\ -q^{-1}\mathsf{S}^{-1}\mathbf{q} & \mathsf{S}^{-1} \end{pmatrix} \tag{14}$$

with

$$\mathsf{S} = \mathsf{Q} - q^{-1}\mathbf{q}\mathbf{q}^{\mathsf{H}}\,, \tag{15}$$

which can be derived from the Sherman–Morrison formula.

Algorithm 2 describes the proposed SIMD procedure SimdMatInv for calculating the inverse matrices. Line 5 calculates (15) for $t = 1, \ldots, T$ in a SIMD manner. Lines from 6 to 9 calculate (14) for $t = 1, \ldots, T$ again in a SIMD manner. In the description, let $\mathcal{Q}(m, l, :)$ and $\mathcal{R}(m, l, :)$ be tensor slices of size $1\times1\times T$ which contain the (m, l)-elements $[\hat{\mathsf{X}}_t]_{ml}$ and $[\hat{\mathsf{X}}_t^{-1}]_{ml}$ of all the T matrices, respectively. On the other hand, let $\mathcal{Q}(\bar{m}, \bar{l}, :)$ and $\mathcal{R}(\bar{m}, \bar{l}, :)$ be tensors of size $(M-1)\times(M-1)\times T$ where the m-th row and l-th column of $\hat{\mathsf{X}}_t$ and $\hat{\mathsf{X}}_t^{-1}$ are eliminated for all the T matrices, respectively. In the element-wise multiplication $\cdot$ and division $/$ operators, so-called Matlab's "singleton expansion" or NumPy's "broadcasting" occur if necessary. The algorithm calls other SIMD procedures SimdMV and SimdVV, which will be explained in the next subsection.

Algorithm 2 SIMD calculation of inverse matrices

```
procedure SimdMatInv($\mathcal{Q}$)                    ▷ $\mathcal{Q}$: size $M \times M \times T$
    if $M$ is 1 then
        $\mathcal{R} \leftarrow 1/\mathcal{Q}(1,1,:)$
    else
        $\mathcal{S} \leftarrow \mathcal{Q}(\bar{1},\bar{1},:) - \mathcal{Q}(\bar{1},1,:) \cdot \mathcal{Q}(1,\bar{1},:)/\mathcal{Q}(1,1,:)$
        $\mathcal{R}(\bar{1},\bar{1},:) \leftarrow$SimdMatInv($\mathcal{S}$)
        $\mathcal{R}(\bar{1},1,:) \leftarrow -$SimdMV($\mathcal{R}(\bar{1},\bar{1},:), \mathcal{Q}(\bar{1},1,:)$)$/\mathcal{Q}(1,1,:)$
        $\mathcal{R}(1,\bar{1},:) \leftarrow \mathcal{R}(\bar{1},1,:)^{\mathsf{H}}$
        $\mathcal{R}(1,1,:) \leftarrow (1-$SimdVV($\mathcal{Q}(1,\bar{1},:), \mathcal{R}(\bar{1},1,:)$)$)/\mathcal{Q}(1,1,:)$
    end if
    return $\mathcal{R}$
end procedure
```

Algorithm 3 SIMD vector-vector multiplications (inner product)

```
procedure SimdVV($\mathcal{Q}$, $\mathcal{R}$)    ▷ $\mathcal{Q}$: size $1 \times M \times T$,  $\mathcal{R}$: size
    $M \times 1 \times T$
    $S \leftarrow$ 0's of size $1 \times 1 \times T$
    for $m = 1$ to $M$ do
        $\mathcal{S} \leftarrow \mathcal{S} + \mathcal{Q}(1,m,:) \cdot \mathcal{R}(m,1,:)$
    end for
    return $\mathcal{S}$
end procedure
```

B. Other Mathematical Operations

For other several operations than matrix inversion, we also employ SIMD calculations as much as possible. Due to space limitations, we do not explain them all, but the two SIMD procedures called from SIMDMATINV.

Algorithm 3 describes the procedure SIMDVV. It calculates the inner product of each tth vector of $\mathcal{Q} = [\mathbf{q}_1^{\mathsf{H}}, \dots, \mathbf{q}_T^{\mathsf{H}}]$ and tth vector of $\mathcal{R} = [\mathbf{r}_1, \dots, \mathbf{r}_T]$, and stores the tth result $s_t = \mathbf{q}_t^{\mathsf{H}}\mathbf{r}_t$ in a tensor $\mathcal{S} = [s_1, \dots, s_T]$ of size $1 \times 1 \times T$.

Algorithm 4 describes the procedure SIMDMV. It calculates the matrix-vector multiplications of each tth matrix of $\mathcal{Q} = [\mathsf{Q}_1, \dots, \mathsf{Q}_T]$ and tth vector of $\mathcal{R} = [\mathbf{r}_1, \dots, \mathbf{r}_T]$, and stores the tth result $\mathbf{s}_t = \mathsf{Q}_t\mathbf{r}_t$ in a tensor $\mathcal{S} = [\mathbf{s}_1, \dots, \mathbf{s}_T]$ of size $M \times 1 \times T$.

IV. Initialization of parameters

Before performing the EM algorithm, the parameters $\theta = \{\{\mathsf{A}_n\}_{n=1}^N, \mathsf{B}, \{\{v_{nt}\}_{n=1}^N\}_{t=1}^T\}$ should be initialized. Since the algorithm gradually improves the parameters, the initialization is important for reaching a good convergence point where sources are well separated.

In this section, we propose a new but simple method that initializes the spatial covariance matrices $\{\mathsf{A}_n\}_{n=1}^N$ by scanning the observations $\mathbf{x}_t, t = 1, \dots, T$. It is based on online clustering with a similarity measure $sim(\mathbf{x}_t\mathbf{x}_t^{\mathsf{H}}, \mathsf{A}_n)$ between two matrices. **Algorithm** 5 shows the procedure. First, all $\mathsf{A}_n, n = 1, \dots N$, are initialized as identity matrices. Then, for each time frame t, the similarities between $\mathbf{x}_t\mathbf{x}_t^{\mathsf{H}}$ and all $\mathsf{A}_n, n = 1, \dots N$, are calculated, and the most similar A_{n^*} is selected (line 6) and updated (line 7). Regarding similarity measures $sim(\mathbf{x}_t\mathbf{x}_t^{\mathsf{H}}, \mathsf{A}_n)$, we propose to employ

$$sim(\mathbf{x}_t\mathbf{x}_t^{\mathsf{H}}, \mathsf{A}_n) = \frac{\mathrm{tr}(\mathbf{x}_t\mathbf{x}_t^{\mathsf{H}}\mathsf{A}_n)}{\mathrm{tr}(\mathbf{x}_t\mathbf{x}_t^{\mathsf{H}})\,\mathrm{tr}(\mathsf{A}_n)} = \frac{\mathbf{x}_t^{\mathsf{H}}\mathsf{A}_n\mathbf{x}_t}{||\mathbf{x}_t||^2\,\mathrm{tr}(\mathsf{A}_n)}\,, \quad (16)$$

Algorithm 4 SIMD matrix-vector multiplications

```
procedure SimdMV($\mathcal{Q}$, $\mathcal{R}$)  ▷ $\mathcal{Q}$: size $M \times M \times T$,  $\mathcal{R}$: size
    $M \times 1 \times T$
    $\mathcal{S} \leftarrow$ 0's of size $M \times 1 \times T$
    for $m = 1$ to $M$ do
        $\mathcal{S} \leftarrow \mathcal{S} + \mathcal{Q}(:,m,:) \cdot \mathcal{R}(m,1,:)$
    end for
    return $\mathcal{S}$
end procedure
```

Algorithm 5 Initializing full-rank covariance matrices

```
 1: procedure InitMat($\{\mathbf{x}_t\mathbf{x}_t^{\mathsf{H}}\}_{t=1}^T$, $N$)
 2:     for $n = 1$ to $N$ do
 3:         $\mathsf{A}_n \leftarrow \mathsf{I}$                    ▷ identity matrix of size $M$
 4:     end for
 5:     for $t = 1$ to $T$ do
 6:         $n^* = \mathrm{argmax}_{n=1}^N sim(\mathbf{x}_t\mathbf{x}_t^{\mathsf{H}}, \mathsf{A}_n)$
 7:         $\mathsf{A}_{n^*} = \mathsf{A}_{n^*} + \mathbf{x}_t\mathbf{x}_t^{\mathsf{H}}$
 8:     end for
 9:     return $\{\mathsf{A}_n\}_{n=1}^N$
10: end procedure
```

which is slightly modified from matrix cosine similarity [17].

Having the matrices $\{\mathsf{A}_n\}_{n=1}^N$ initialized in the above manner, we do not consider a special strategy for the initialization of the other parameters, i.e., B and $\{\{v_{nt}\}_{n=1}^N\}_{t=1}^T$. In the experiment reported in Sect. VI, they were simply initialized as $\mathsf{B} = \lambda\mathsf{I}$ with $\lambda = 10^{-3}$ and with I being an identity matrix, and $v_{nt} = 1$ for all $n = 1, \dots, N$ and $t = 1, \dots, T$.

V. Full-band case

So far, we have considered a time sequence of observation vectors $\mathbf{x}_t, t = 1, \dots, T$. Let us call this a narrow-band case. To separate real-room audio mixtures with delay and reverberations, we typically apply a short-time Fourier transformation (STFT) to the time-domain mixtures. In this case, as the result of STFT, we have observation vectors $\mathbf{x}_{tf}$ for time index t and frequency-bin index f. Let us call this a full-band case where there are many frequency bins $f = 1, \dots, F$.

The two techniques described so far can also be employed in a full-band case by the notation change summarized in Table I. The acceleration technique proposed in Sect. III would benefit more in a full-band case than in a narrow-band case, because of a larger SIMD data size $T \times F$ as shown in the last line of the table.

In a full-band case, all frequency bins are linked regarding the SIMD computation. However, in the statistical model (1) and (2), the parameters of different frequency bins are not linked, and there exist permutation ambiguities of the bin-wise separated signals $\tilde{\mathbf{y}}_{ntf}$. To align the ambiguities, in this paper, we simply follow the ideas [18, 19] based on calculating

$$dom_{nf}(t) = \frac{\mathrm{tr}(\mathsf{A}_{nf})\,v_{ntf}}{\sum_{o=1}^N \mathrm{tr}(\mathsf{A}_{of})\,v_{otf}}\,, \quad (17)$$

$0 \leq dom_{nf}(t) \leq 1$, indicating how the nth separation dominates the mixture at time-frequency slot (t, f). Then, we cluster the time sequences dom_{nf}, $n = 1, \dots, N$ and $f = 1, \dots F$, to align permutations.

TABLE I
EXTENSION FROM NARROW-BAND TO FULL-BAND

	narrow-band	full-band
observation mixture	$\mathbf{x}_t$	$\mathbf{x}_{tf}$
separation	$\tilde{\mathbf{y}}_{nt}$	$\tilde{\mathbf{y}}_{ntf}$
Gaussian cov. matrix	$\hat{\mathsf{X}}_t$	$\hat{\mathsf{X}}_{tf}$
source spatial cov. matrix	A_n	A_{nf}
noise cov. matrix	B	B_f
source temporal power	v_{nt}	v_{ntf}
SIMD data size	T	$T \times F$

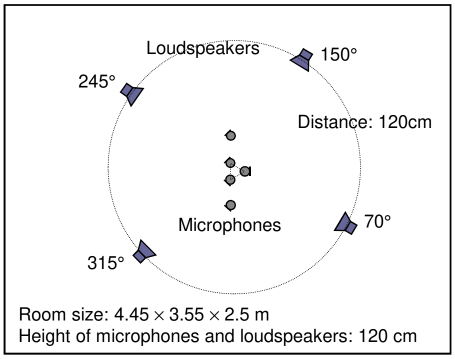
Loudspeakers
150°
245°
Distance: 120cm
Microphones
70°
315°
Room size: 4.45 × 3.55 × 2.5 m
Height of microphones and loudspeakers: 120 cm

Fig. 1. Experimental setup

VI. EXPERIMENTS

We performed experiments to separate three or four speech sources ($N = 3$ or 4) with from two to five microphones ($M = 2, 3, 4, 5$). We measured the impulse responses from the sources (the loudspeakers) to the microphones under the room conditions shown in Fig. 1. The room reverberation time was 130 ms. The mixtures at the microphones were constructed by convolving the impulse responses and 6-second English speech sources. The sampling frequency was 8 kHz. The frame width and shift of STFT were 128 ms and 32 ms, respectively. Consequently, the numbers of time frames and frequency bins were $T = 201$ and $F = 513$, respectively.

A. Acceleration

Figure 2 shows how FCA was accelerated by using the SIMD-based calculations. The EM algorithm was coded with Matlab R2018a and run on an Intel Core i7-8700K (3.70GHz) processor together with GeForce GTX 1080 Ti as a GPU. The computational time was measured by using Matlab's tic and toc for each case. We observe that the proposed SIMD-based acceleration with a GPU, GPU (SIMD all), was considerably effective for all $M = 2, 3, 4, 5$. The SIMD-based matrix inversion (**Algorithm 2**) was effective even without GPU, CPU (SIMD all), as long as the number of microphones was small, but not so effective as M increases.

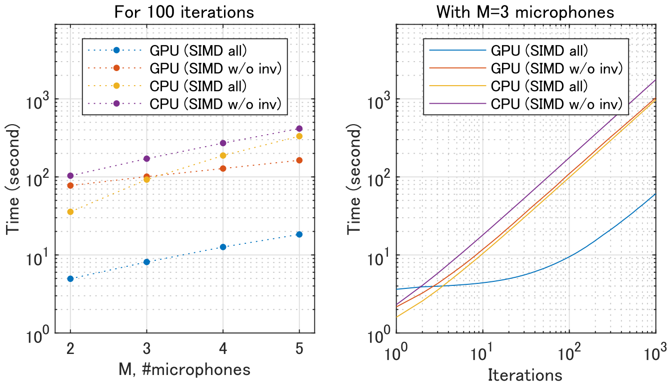

Fig. 2. Computational times for $N = 4$ sources: for 100 iterations with varying number M of microphones (left), and with $M = 3$ microphones up to 1000 iterations (right). GPU and CPU indicate the use of GPU or not. SIMD all corresponds to situations where we employed SIMD calculations as much as possible as Sect. III explains. On the other hand, in SIMD w/o inv situations, we employed the sequential **Algorithm 1** instead of the SIMD **Algorithm 2**.

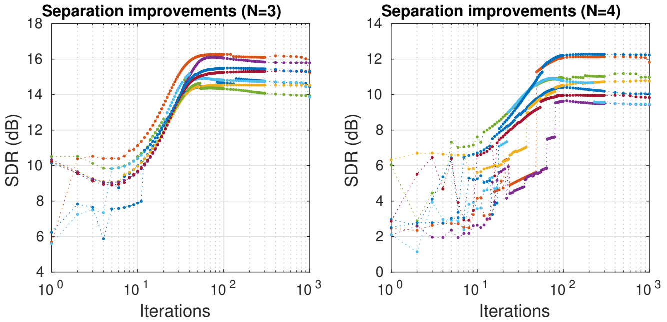

Fig. 3. Improvements of separation performance as iterations went on for eight combinations of sources in each $N = 3$ (left) and $N = 4$ (right) cases.

Figure 3 shows the separation performance with three microphones ($M = 3$) measured in signal-to-distortion ratios (SDRs) [20] for the number of iterations. The discontinuities of the SDR values were due to the non-smoothness of the permutation alignment results. We observe that 100 iterations were sufficient for obtaining good separations. Coming back to Fig. 2, we see that for 100 iterations the proposed technique GPU (SIMD all) took less than 8 seconds for the 6-second mixture. This was practical merit over the others that took more than 100 seconds.

B. Parameter Initialization

We compared three methods Pro, Sep, and Ran for parameter initialization. Pro corresponds to the proposed method described in Sect. IV. In Sep, we executed a source separation method [19] to obtain soft masks w_{ntf}, $0 \leq w_{ntf} \leq 1$, and then the source covariance matrices and the temporal powers were initialized as

$$\mathsf{A}_{nf} \leftarrow \frac{1}{T}\sum_{t=1}^{T} w_{ntf}\mathbf{x}_{tf}\mathbf{x}_{tf}^{\mathsf{H}}\,, \tag{18}$$

$$v_{ntf} \leftarrow \frac{w_{ntf}}{M}\mathbf{x}_{tf}^{\mathsf{H}}\mathsf{A}_{nf}^{-1}\mathbf{x}_{tf}\,, \tag{19}$$

respectively. In Ran, the temporal powers v_{ntf} were randomly initialized with nonnegative numbers and the covariance matri-

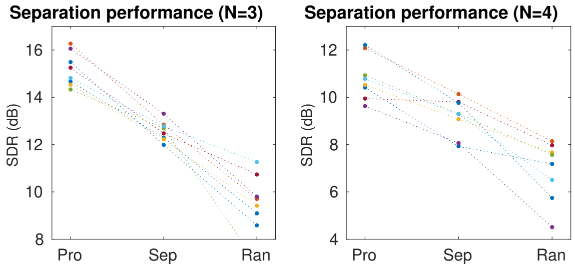

Fig. 4. The effects of parameter initializations examined for eight combinations of sources in each source-number case. The number of iterations was 100 for all cases. The proposed method Pro performed the best in all combinations.

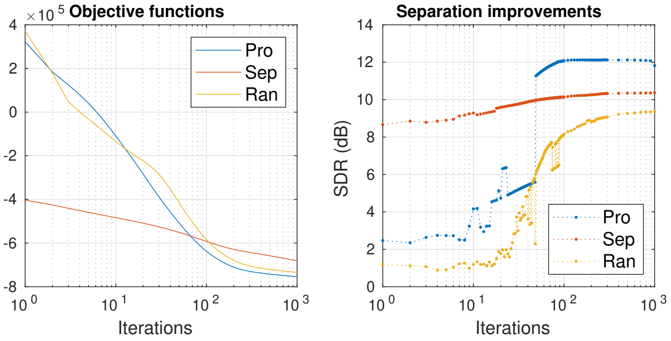

Fig. 5. The effects of parameter initializations examined for a four-source case ($N = 4$). Sep started from a situation where sources were separated by another method [19]. However, the separations were not well improved even after many iterations. The proposed method Pro started from a situation where sources were not separated but converged to a well-separated situation. The random initialization Ran converged to a slightly-separated situation.

ces A_{nf} were initialized as identity matrices. In both Sep and Ran cases, the noise covariance matrices B_f were initialized in the same manner with the proposed method Pro.

Figures 4 and 5 show the effects of parameter initializations. The number of microphones was three ($M = 3$). The proposed method Pro outperformed the other two. See the figure captions for the details.

VII. Conclusion

To make FCA easier to use and more effective for a real-world BSS task, we proposed two practical techniques. The SIMD-based acceleration was shown to be effective in the experiments especially with a GPU, achieving speedups of more than 10 times over the other implementations. The parameter initializations for spatial covariance matrices have shown to effective for the EM algorithm, reaching good separations after a hundred of iterations. Future work will include the application of such an acceleration technique to the other BSS methods, e.g., multichannel NMF [7, 10, 11].

References

[1] C. Jutten and J. Herault, "Blind separation of sources, part I: An adaptive algorithm based on neuromimetic architecture," *Signal processing*, vol. 24, no. 1, pp. 1–10, 1991.

[2] A. Hyvärinen, J. Karhunen, and E. Oja, *Independent Component Analysis*. John Wiley & Sons, 2001.

[3] A. Cichocki and S. Amari, *Adaptive Blind Signal and Image Processing*. John Wiley & Sons, 2002.

[4] S. Makino, T.-W. Lee, and H. Sawada, Eds., *Blind Speech Separation*. Springer, 2007.

[5] E. Vincent, T. Virtanen, and S. Gannot, *Audio source separation and speech enhancement*. John Wiley & Sons, 2018.

[6] N. Duong, E. Vincent, and R. Gribonval, "Under-determined reverberant audio source separation using a full-rank spatial covariance model," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 18, no. 7, pp. 1830–1840, Sep. 2010.

[7] S. Arberet, A. Ozerov, N. Duong, E. Vincent, R. Gribonval, F. Bimbot, and P. Vandergheynst, "Nonnegative matrix factorization and spatial covariance model for under-determined reverberant audio source separation," in *Proc. ISSPA 2010*, May 2010, pp. 1–4.

[8] N. Ito, S. Araki, and T. Nakatani, "FastFCA: Joint diagonalization based acceleration of audio source separation using a full-rank spatial covariance model," in *Proc. EUSIPCO*, 2018, pp. 1667–1671.

[9] N. Ito and T. Nakatani, "Fastfca-as: Joint diagonalization based acceleration of full-rank spatial covariance analysis for separating any number of sources," in *Proc. IWAENC*, 2018.

[10] A. Ozerov and C. Févotte, "Multichannel nonnegative matrix factorization in convolutive mixtures for audio source separation," *IEEE Trans. Audio, Speech and Language Processing*, vol. 18, no. 3, pp. 550–563, Mar. 2010.

[11] H. Sawada, H. Kameoka, S. Araki, and N. Ueda, "Multichannel extensions of non-negative matrix factorization with complex-valued data," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 21, no. 5, pp. 971–982, May 2013.

[12] M. Togami, Y. Kawaguchi, R. Takeda, Y. Obuchi, and N. Nukaga, "Optimized speech dereverberation from probabilistic perspective for time varying acoustic transfer function," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 7, pp. 1369–1380, 2013.

[13] T. Otsuka, K. Ishiguro, H. Sawada, and H. G. Okuno, "Bayesian nonparametrics for microphone array processing," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 2, pp. 493–504, 2014.

[14] J. Nikunen and T. Virtanen, "Direction of arrival based spatial covariance model for blind sound source separation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 3, pp. 727–739, 2014.

[15] A. Liutkus, D. Fitzgerald, Z. Rafii, B. Pardo, and L. Daudet, "Kernel additive models for source separation," *IEEE Transactions on Signal Processing*, vol. 62, no. 16, pp. 4298–4310, 2014.

[16] W. W. Hager, "Updating the inverse of a matrix," *SIAM review*, vol. 31, no. 2, pp. 221–239, 1989.

[17] H. J. Seo and P. Milanfar, "Detection of human actions from a single example," in *Proc. ICCV*, 2009, pp. 1965–1970.

[18] H. Sawada, S. Araki, and S. Makino, "Measuring dependence of bin-wise separated signals for permutation alignment in frequency-domain BSS," in *Proc. ISCAS 2007*, 2007, pp. 3247–3250.

[19] ——, "Underdetermined convolutive blind source separation via frequency bin-wise clustering and permutation alignment," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 19, no. 3, pp. 516–527, Mar. 2011.

[20] E. Vincent, S. Araki, F. Theis, G. Nolte, P. Bofill, H. Sawada, A. Ozerov, V. Gowreesunker, D. Lutter, and N. Duong, "The signal separation evaluation campaign (2007–2010): Achievements and remaining challenges," *Signal Processing*, vol. 92, no. 8, pp. 1928–1936, Aug. 2012.