An efficient parallel kernel based on Cholesky decomposition to accelerate Multichannel Non-Negative Matrix Factorization

Antonio J. Muñoz-Montoro[1*], Julio J. Carabias-Orti[2†], Daniele Salvati[3†] and Raquel Cortina[1†]

[1*]Department of Computer Science, University of Oviedo, Campus de Gijón, Gijón, 33204, Asturias, Spain.
[2]Departament of Telecommunication Engineering, University of Jaen, Campus Cientifico-Tecnologico de Linares, Linares, 23700, Jaen, Spain.
[3]Department of Mathematics, Computer Science and Physics, University of Udine, Via delle Scienze 206, Udine, 33100, Italy.

*Corresponding author(s). E-mail(s): munozantonio@uniovi.es;
Contributing authors: carabias@ujaen.es; daniele.salvati@uniud.it; cortina@uniovi.es;
†These authors contributed equally to this work.

Abstract

Multichannel Source Separation has been a popular topic, and recently proposed methods based on the local Gaussian model (LGM) have provided promising result despite its high computational cost when several sensors are used. The main reason being due to inversion of a spatial covariance matrix, with a complexity of $\boldsymbol{O(I^3)}$, being $\boldsymbol{I}$ the number of sensors. This drawback limits the practical application of this approach for tasks such as sound field r econstruction o r v irtual r eality, among others. In this paper, we present a numerical approach to reduce the complexity of the Multichannel Non-negative Matrix Factorization (MNMF) to address the task of audio source separation for scenarios with a high number of sensors such as High Order Ambisonics (HOA) encoding. In particular, we propose a parallel multi-architecture driver to compute the multiplicative update rules in MNMF approaches. The proposed driver has been designed to work on both sequential and multi-core computers, as well as Graphics Processing Units (GPUs) and Intel Xeon coprocessors. The proposed software was written in C language and can be called from numerical computing environments. The proposed solution tries to reduce the computational cost of the multiplicative update rules by using the Cholesky decomposition and by solving several triangular equation systems. The proposal has been evaluated for different scenarios with promising results in terms of execution times for both CPU and GPU. To the best of our knowledge, our proposal is the first system that addresses the problem of reducing the computational cost of full-rank MNMF-based systems using parallel and high performance techniques.

Keywords: Multichannel NMF, High Order Ambisonics, Cholesky decomposition, GPU, parallel architecture

1 Introduction

Non-negative matrix factorization (NMF) is a powerful tool for dimensionality reduction in audio processing. It decomposes a non-negative data matrix (i.e., a matrix with non-negative entries) into a product of two lower-rank matrices. This allows for the approximation of the data matrix as a sum of rank-1 non-negative matrices. This technique is popular due to its universality, as it can be applied to a wide range of audio sources, including speech, music, environmental sounds, etc., and its flexibility, as it can be adapted with various constraints such as harmonicity of spectral patterns, smoothness of activation coefficients, pre-trained spectral patterns, etc.

However, when dealing with multichannel data, standard NMF is not suitable. Multichannel audio is becoming increasingly prevalent in today's world, with technologies such as distributed microphone arrays and ambisonic microphones widely adopted for 360-degree videos and Virtual Reality (VR) experiences. Additionally, the object-based audio format is also widely used and regarded as a standard VR format. It consists of pairs of object metadata and corresponding waveform signals. Placing an audio object anywhere in space gives listeners an immersive experience that can serve as the audio part of the six degrees of freedom (6DOF) system proposed in MPEG-I [1]. In this sense, converting from the High Order Ambisonics (HOA) format to the object format is highly desired, but requires further processing of the captured signals.

To address the limitations of standard NMF in handling multichannel data, various extensions have been proposed, such as stacking channels into one matrix structure [2] or considering a parallel factor (PARAFAC) model [3]. However, these methods primarily focus on using amplitude information and do not fully exploit the spatial information present in microphone array recordings. To address this issue, the beamspace data model proposed by Lee et al. [4] incorporates the projection of the input signal onto a set of steered directions and accounts for the inherent phase-difference information present in these recordings. Another approach, as proposed in [5], is to project signals from a uniform linear array onto the Ray Space (RS) domain, where spatial information is also encoded in the magnitude information. Other methods based on the RS domain incorporate the spatial information by using plenacoustic functions [6, 7]. An alternative way for modeling spatial information is through the use of signal representations based on spherical harmonic (SH) domain [8, 9] or the spatial covariance matrix (SCM) [10].

The multichannel NMF (MNMF) modeling treats the complex-valued STFT coefficients as realizations of zero-mean circular complex-valued Gaussian random variables with structured variances (using NMF to model the signal power spectral density (PSD)) and covariances [11]. However MNMF suffers from the strong sensitivity to parameter initialization and a high computational cost. The former drawback can be mitigated by constraining the mixing model as a weighted combination of fixed spatial kernels steering toward a subset of possible spatial directions [10, 12] or by initializing the source PSD using single channel deep learning techniques [13, 14]. The later drawback limits the practical use of this method for tasks involving a high number of sensors for applications such as dereverberation [15], sound field reconstruction [16, 17] or navigation for augmented/extended reality [18]. The main reason owing to the multiple matrix inversions during the parameter estimation procedure [19].

To mitigate this drawback, several diagonalization based methods have been proposed to provide computationally-efficient solutions [9 , 13 , 17 , 20–22] at the cost of limiting to certain array setup [22], reducing only to the SCM diagonal values [9] or relying on the statistical independence between the sources to derive the spatial characteristics [13, 21].

However, to our best knowledge, optimizations of the original full-rank solution have not studied in the literature. In this work, we study the efficiency of the original full-rank MNMF in [23] and develop a novel approach based on Cholesky decomposition to allow novel MNMF extensions for scenarios where a high number of sensors are involved. We focus specifically on the application of the MNMF model in the HOA domain for A and B-formats, which is the most widely adopted sound field representation. Note that the Cholesky decomposition is a powerful tool for efficiently solving linear systems and inverting matrices. In the field of signal processing, it has been used in a variety of applications such as instantaneous fundamental frequency estimation [24], speech enhancement [25], and audio source separation [26, 27]. Additionally, it has also been used in combination with other techniques such as NMF to reduce dimensionality in data [27–29]. However, this is the first work in which the Cholesky decomposition is specifically used to accelerate matrix inversions during the parameter estimation procedure in MNMF-based models.

In particular, we propose a parallel multi-architecture driver to compute the multiplicative update rules in MNMF approaches. The proposed driver has been designed for sequential and multi-core computers, as well as Graphics Processing Units (GPU) and Intel Xeon coprocessors. Note that our approach can be called from numerical computing environments such as MATLAB or GNU Octave through MEX (MATLAB Executable) interfaces and from Python.

The structure of the rest of the article is as follows. In Sect. 2, we review some of the most common mixtures model representations and present the problem formulation. Then, the proposed approach is presented in Sect. 3. In Sect. 4, the evaluation setup is presented and the proposed system is tested for different scenarios. Finally, we summarize the work in Sect. 5.

2 Background

In this section we will review some of the most common mixtures model representations in the literature. Then, the foundation of the multichannel NMF will be introduced, from the Local Gaussian Model (LGM) formulation to the derivation of the update rules.

2.1 Mixture Models

2.1.1 Microphone Domain

The source separation problem consists in estimating the contribution $\mathbf{s}_{j,t} \in \mathbb{R}^I$ of each source $j = 1, ..., J$ in each microphone $i = 1, ..., I$ and at each time instant $t = 1, ..., T$. In the absence of noise, the mixture can be written as:

$$\mathbf{y}_t = \sum_{j=1}^{J} \mathbf{s}_t^j, \tag{1}$$

where $\mathbf{y}_t = [y_{1,t}, ..., x_{I,t}] \in \mathbb{R}^I$ are microphone array signals. Under reverberant conditions and assuming the hypothesis of point sources, the source signal $\tilde{s}_t^j$ can be related to its contribution $\mathbf{s}_t^j$ through:

$$\mathbf{s}_t^j = [\alpha_i^j * \tilde{s}^j]_t \tag{2}$$

where $*$ denotes the convolution product, α_i^j is the impulse response of the mixing filter between the source j, and the microphone i.

2.1.2 HOA domain

In the HOA domain, any plane wave can be characterized by a sound signal s_t and a direction of arrival of the sound γ. The unit vector γ indicates the direction of arrival of the plane wave (the origin of the sound source). This vector can be decomposed in spherical coordinates as $\gamma = (\cos\phi\cos\theta, \sin\theta\cos\phi, \sin\theta)$, with θ being the azimuth and ϕ the elevation of the sound source. The SH components gains for source at direction (ϕ, θ) can be expressed as,

$$Z_{nm}(\phi,\theta) = \sqrt{(2n+1)\frac{(n- \mid m \mid)!}{(n+ \mid m \mid)}} P_{n|m|}(\sin\theta) z_m(\phi) \ , \tag{3}$$

where the SH order and degree are denoted by n and m, respectively. $P_{n|m|}$ is the associated Legendre function of degree n and

$$z_m(\phi) = \begin{cases} \sqrt{2}\sin \mid m \mid \phi & \text{if } m < 0 \\ 1 & \text{if } m = 0 \\ \sqrt{2}\cos \mid m \mid \phi & \text{if } m > 0 \end{cases} . \tag{4}$$

Each order n has $i = 1, ..., I$ channels (i.e. SH signals) with $I = (n+1)^2$. The SH components are usually ordered using the so-called ACN ambisonics channel ordering[1] as a vector $\mathbf{z}(\gamma) \in \mathbb{R}^K$ containing each $Z_{nm}(\phi, \theta)$.

For a general set of multiple localized sources (multiple plane waves) with signals s_t^j coming from direction γ^j, the anechoic ambisonics mixture $\mathbf{y}_t \in \mathbb{R}^I$ can be expressed as:

$$\mathbf{y}_t = \sum_{j=1}^{J} s_t^j \mathbf{y}(\gamma^j) \ . \tag{5}$$

As explained in [30], reverberant conditions can be modeled considering an image-source model of L images modeling reflections and late reverberation (up to a desired limit). In this way, the mixture model can be expressed as a function of a propagation delay δ_l^j and a propagation filter $h_l^j(\tau)$ that models the absorption and attenuation correspoding to each l-th image of each q-th source. The resulting reverberant model is expressed as:

$$y_t = \sum_{j=1}^{J} s^j(t-\delta^j)y(\gamma^j) + \sum_{j=1}^{J} \sum_{\tau=\delta^j+1}^{\Omega} s^j(t-\tau)h^j(\tau)) \ , \tag{6}$$

where δ^j are the direct path delays and Ω is the chosen maximum sample length for the ambisonics IRs. Note that, for causality, the first non-zero index of $h^j(\tau)$ is set as $\tau = \delta_l^j$.

2.2 Local Gaussian Model

The model assumes that an I-channel vector of a short-time Fourier transform (STFT) bin for j-th source can be modeled as a multivariate complex Gaussian, i.e.,

$$\mathbf{s}_{ft}^j \sim \mathcal{N}_{\mathbb{C}}(0, \mathbf{R}_{ft}^j), \tag{7}$$

where $\mathbf{s}_{ft}^j \in \mathbb{C}^I$ denotes the spatial image of the j-th source in the STFT domain, $\mathbf{R}_{ft}^j = \mathbb{E}[\mathbf{s}_{ft}^j {\mathbf{s}_{ft}^j}^H] \in \mathbb{C}^{I \times I}$ denotes the covariance matrix of the complex Gaussian distribution $\mathbb{C}$, where f is the frequency bin index, and t is the time frame index.

Let us represent the spatial image of a mixture of multiple sources $\mathbf{y}_{f,t} \in \mathbb{C}^I$ as a sum of complex Gaussians, i.e.,

[1] http://ambisonics.ch/

$$\mathbf{y}_{ft} = \sum_{j=1}^{J} s_{ft}^{j}(\omega) \sim \mathcal{N}_{\mathbb{C}}(0, \mathbf{R}_{ft}), \tag{8}$$

where $\mathbf{R}_{f,t}^{j} \in \mathbb{C}^{I \times I}$ denotes the SCM. Under the assumption that the sources are mutually independent, the SCM of the mixture $\mathbf{R}_{f,t}^{j}$ can be modeled as the sum of the SCMs of all sources, i.e.,

$$\mathbf{R}_{ft} = \mathbb{E}[\mathbf{y}_{ft}\mathbf{y}_{ft}^{H}] = \sum_{j=1}^{J} \mathbf{R}_{ft}^{j}(\omega), \tag{9}$$

and the log-likelihood of the spatial image $\mathbf{y}_{ft}$ for the model parameters φ can be expressed as

$$\log \mathbb{P}_{ft} = \sum_{j=1}^{J} \log \mathcal{N}_{\mathbb{C}}(\mathbf{y}_{ft} \mid 0, \hat{\mathbf{R}}_{ft}(\varphi)), \tag{10}$$

where the SCM of the mixture is modeled by $\hat{\mathbf{R}}_{ft}(\varphi))$ and the parameter φ will be defined in the next section. The maximization of this likelihood can be interpreted as the minimization of the log-determinant divergence between the empirical SCM, $\tilde{\mathbf{R}}_{ft} = \mathbf{y}_{ft}\mathbf{y}_{ft}^{H}$, and the estimated SCM, $\hat{\mathbf{R}}_{ft} \in \mathbb{C}^{I \times I}$

$$C(\varphi) = \sum_{ft} D_{LD}\left(\hat{\mathbf{R}}_{ft} \mid \tilde{\mathbf{R}}_{ft}\right) \equiv \sum_{ft} \operatorname{tr}\left(\tilde{\mathbf{R}}_{ft}\hat{\mathbf{R}}_{ft}(\varphi)^{-1}\right) + \log\det\left(\hat{\mathbf{R}}_{ft}(\varphi)\right), \tag{11}$$

where $C(\varphi)$ can be seen as a cost function that we want to minimize with respect to the model parameters φ. We denote the log-determinant divergence by D_{LD}.

2.3 Multichannel NMF (MNMF)

In [23], the authors proposed a MNMF framework where the SCM mixture $\hat{\mathbf{R}}_{ft}$ is assumed to be a positive definite Hermitian and modeled as a superposition of time-invariant SCMs $\mathbf{G}_{f,k} \in \mathbb{C}^{I \times I}$ coupled with a scale value $\lambda_{f,t}$ that represents the PSD and can be modeled using a classical NMF structure. The scale value can be modeled using a classical NMF structure:

$$\lambda_{ft} = \sum_{k=1}^{K} w_{fk} h_{kt}, \tag{12}$$

where k denotes the NMF component index, w_{fk} and h_{kt} represent both the basis functions and their corresponding time-varying gains. The SCM mixture $\hat{\mathbf{R}}_{ft}$ can be expressed as:

$$\hat{\mathbf{R}}_{ft}(\varphi) = \sum_{k=1}^{K} \mathbf{G}_{fk} w_{fk} h_{kt}, \tag{13}$$

where the model parameters $\varphi = \{\mathbf{G}_{fk}, w_{fk}, h_{kt}\}$ can be estimated by minimizing the cost function in Eq. (11) using classical algorithms such as the expectation-maximization (EM) [31] or the majorization-minimization (MM) [23] to derive the update rules. In particular, for the model in Eq. (13), the update rules using MM are as follows:

$$w_{fk} \leftarrow w_{fk} \sqrt{\frac{\sum_t h_{kt} \operatorname{tr}\left(\hat{\mathbf{R}}_{ft}^{-1} \mathbf{R}_{ft} \hat{\mathbf{R}}_{ft}^{-1} \mathbf{G}_{f,k}\right)}{\sum_t h_{kt} \operatorname{tr}\left(\hat{\mathbf{R}}_{ft}^{-1} \mathbf{G}_{f,k}\right)}} \tag{14}$$

$$g_{kt} \leftarrow g_{kt} \sqrt{\frac{\sum_t w_{fk} \operatorname{tr}\left(\hat{\mathbf{R}}_{ft}^{-1} \mathbf{R}_{ft} \hat{\mathbf{R}}_{ft}^{-1} \mathbf{G}_{f,k}\right)}{\sum_f w_{fk} \operatorname{tr}\left(\hat{\mathbf{R}}_{ft}^{-1} \mathbf{G}_{f,k}\right)}} \tag{15}$$

whereas, updating $\mathbf{G}_{fk}$ require to solve an algebraic Riccati equation of the form $\mathbf{G}_{fk}\mathbf{A}\mathbf{G}_{fk} = \mathbf{B}$. Details are omitted for the sake of brevity but could be reviewed in [23].

Unfortunately, updating the model parameters requires a huge computational cost of order $O(I^3)$ owing to the multiple matrix inversions during the parameter updates [19]. This drawback limits the use of this framework when the number of channels increases. For example, in the HOA, modeling a fourth order representation requires at least $(4 + 1) \times 2 = 25$ microphones to avoid spatial aliasing, which makes the method infeasible in practical situations.

As commented in the introduction, several diagonalization based methods have been proposed to provide computationally-efficient solutions [9, 13, 20–22] at the cost of limiting to certain array setup [22], reducing only to the SCM diagonal values [9] or relying on the statistical independence between the sources to derive the spatial characteristics [13, 21]. However, to our best knowledge, optimizations of the original full-rank solution have not studied in the literature. In the next section, we present an approach to optimize the inverse of full-rank matrices.

3 Proposed approach for reducing the complexity of MNMF

In this work, a parallel approach for updating the full-rank matrices of MNMF systems is proposed. In particular, the objective of this work is to provide an efficient solution to the problem described in Section 2.3, i.e., reducing the computational cost of the multiplicative update rules in the MNMF approaches when dealing with high number of microphone signals, such as the case of HOA encoding.

As presented in Section 2.3, the multiplicative update rules (see Eq. 14 and Eq. 15) involve ~~the calculation of~~ the inverse of the SCM $\hat{\mathbf{R}}_{ft}$ for each time-frequency point (f, t) and matrix multiplications. These operations entail a high computational cost for two reasons: 1) it is necessary to repeat them as many times as the number of iterations required for the convergence of the method and as the number of time-frequency points of the input signal spectrogram, and 2) the large size of the matrices when dealing with multichannel recordings with high number of microphones.

In this sense, the design of a driver that allows to efficiently compute the multiplicative update rules is required to develop a feasible MNMF system for real scenarios. Therefore, we propose a parallel multi-architecture driver designed to work on multi-core computers and GPU. The proposed software was written in C language using, as appropriate, OpenMP or the CUDA suite. This driver can also be called from numerical computing environments such as MATLAB or GNU Octave through MEX (MATLAB Executable) interfaces or from Python.

To deal with the goal, we propose to use the Cholesky decomposition for Hermitian and positive semi-definite matrices, as explained below. Without loss of generality, the target operation to speedup can be expressed as:

$$\alpha \leftarrow \alpha \frac{\sum \beta \operatorname{tr}\left(\mathbf{A}^{-1}\mathbf{B}\mathbf{A}^{-1}\mathbf{C}\right)}{\sum \beta \operatorname{tr}\left(\mathbf{A}^{-1}\mathbf{C}\right)} \tag{16}$$

where $\alpha, \beta \in \mathbb{R}$ and $\mathbf{A}, \mathbf{B}, \mathbf{C} \in \mathbb{C}^{I \times I}$. Note that $\mathbf{A}, \mathbf{B}, \mathbf{C}$ are Hermitian and positive semi-definite matrices. As can be observed, the most costly operation in Eq. 16 is

$$\mathbf{Z} = \operatorname{tr}\left(\mathbf{A}^{-1}\mathbf{B}\mathbf{A}^{-1}\mathbf{C}\right) \tag{17}$$

and, in particular, the inverse of $\mathbf{A}$, since the number of channels I is really high in HOA signals. Our proposal consists of reformulating Eq. 17 as a problem of solving several linear equations systems. In this sense, we can first define the following systems of linear equations:

$$\left[\mathbf{X}\ \mathbf{Y}\right] = \mathbf{A}^{-1}\left[\mathbf{B}\ \mathbf{C}\right]. \tag{18}$$

This implies

$$\mathbf{A}\left[\mathbf{X}\ \mathbf{Y}\right] = \left[\mathbf{B}\ \mathbf{C}\right]. \tag{19}$$

To address these systems, we propose to apply the Cholesky decomposition on the matrix $\mathbf{A}$. Thus, if $\mathbf{A} \in \mathbb{C}^{I \times I}$ is a positive semi-definite Hermitian matrix, Cholesky decomposition factorizes it into an upper triangular matrix and its conjugate transpose as follows:

$$\mathbf{A} = \mathbf{U}^{T}\mathbf{U}. \tag{20}$$

In our approach, the Cholesky decomposition has been addressed using the LAPACK implementation based on Level 3 BLAS calls. The total number of floating-point operations is approximately ~~$\frac{4}{3}n^3$~~ $\frac{4}{3}I^3$ for complex flavors according to LAPACK documentation.

Using the Cholesky decomposition on $\mathbf{A}$, Eq. 19 can be redefined as the following triangular systems of linear equations:

$$\mathbf{U}^{T}\left[\mathbf{Z}_1\ \mathbf{Z}_2\right] = \left[\mathbf{B}\ \mathbf{C}\right] \tag{21}$$

Algorithm 1 Pseudocode of the proposed driver

```
Input: Matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ and scalars $\alpha$ and $\beta$.
Compute the Cholesky decomposition of $\mathbf{A}$.
Obtain $\mathbf{Z}_1$ and $\mathbf{Z}_2$ solving the linear equation system in Eq. 21.
Obtain $\mathbf{X}$ and $\mathbf{Y}$ solving the linear equation system in Eq. 22.
Compute the trace of $\mathbf{Y}$.
Compute the trace of $\mathbf{XY}$.
Update $\alpha$ using Eq. 23.
Output: The updated value of $\alpha$.
```

and

$$\mathbf{U}\left[\mathbf{X}\ \mathbf{Y}\right] = \left[\mathbf{Z}_1\ \mathbf{Z}_2\right]. \tag{22}$$

These systems are solved in our approach using again the LAPACK implementation based on Level 3 BLAS calls. These operations (i.e., the Cholesky decomposition and the equation system solving) require ~~$\frac{28}{3}n^3$~~ $\frac{52}{3}I^3$.

Once the equation systems have been solved, Eq. 16 can be reformulated as:

$$\alpha \leftarrow \alpha \frac{\sum \beta \ \mathrm{tr}\left(\mathbf{XY}\right)}{\sum \beta \ \mathrm{tr}\left(\mathbf{Y}\right)}. \tag{23}$$

Finally, the trace of the matrix product $\mathbf{XY}$ has been addressed taking into account that only elements of the diagonal must be computed instead of computing the whole matrix product.

The pseudocode of the proposed approach is detailed in Algorithm 1. The code for the proposed parallel multi-architecture driver is freely available online[2].

4 Evaluation and experimental results

In this section, the proposed system is evaluated in terms of execution times and speedup. In this evaluation, we have conducted several experiments to analyze the performance and reliability of our proposal is a synthetic dataset. For this purpose, we have generated several multi-channel synthetic mixtures with different duration and number of channels. All these mixtures were created with a sampling frequency of 44,1 kHz. In particular, the time-frequency representation used for the mixtures was obtained by using 2048-point short-time Fourier transform (STFT) and half overlap between adjacent frames. ~~The FFTW package [32] was used to compute the STFT.~~ Finally, the number of channels considered varied between 64 and 4096.

Regarding the testbed, we have focused our interest on two different systems. Firstly, we have used a server with two Intel® Xeon® E5-2603 v3 processor with 6 cores each. It operates at 1.60 GHz and HyperThreading and Turbo Boost are both deactivated. This server has 1 TB of RAM and a

[2] https://github.com/QHPC-SP-Research-Lab/Parallel-Cholesky-MNMF

GPU TESLA P100-PCIe with 16 GB of RAM. ~~The theoretical performance (for floating point operations in double precision) of the CPU is approximately 300 GFLOPS according to the Intel documentation and the theoretical performance of the GPU is approximately 4750 GFLOPS according to the NVIDIA documentation. Therefore, the theoretical speedup of the GPU with respect to the CPU is approximately 16. Secondly, the experiments were conducted on a server with two Intel® Xeon® Silver 4314 processor with 16 cores each. This server operates at 2.40 GHz and has 128 GB of RAM and a GPU RTX A6000-PCIe with 48 GB of RAM. In this case, the theoretical performance (for floating point operations in double precision) of the CPU is approximately 1200 GFLOPS according to the Intel documentation and the theoretical performance of the GPU is approximately 1210 GFLOPS according to the NVIDIA documentation. Therefore, we expect a theoretical speedup on this server of approximately one between the GPU and CPU. Both systems run CentOS Linux 7, the OpenBlas library (release 0.3.20, February 2022), the FFTW library (release 3.3.10, September 2021), the Intel oneAPI (release 2022.2, May 2022) and the GNU C Compiler 7 with the specification 4.5 of OpenMP. OpenBLAS is an optimized BLAS library based on GotoBLAS2 1.13 BSD.~~ The theoretical peak performance for floating point operations in double precision (TPPDP) of this system is 307.2 GFLOPS, according to the Intel documentation[3], and the TPPDP of its GPU is approximately 4.7 TFLOPS according to the NVIDIA documentation[4]. Therefore, the theoretical speedup of the GPU with respect to the CPU is approximately 16. Secondly, the experiments were conducted on a server with two Intel® Xeon® Silver 4314 processor with 16 cores each. This server operates at 2.40 GHz and has 128 GB of RAM and a GPU RTX A6000-PCIe with 48 GB of RAM. In this case, the TPPDP of the CPUs is approximately 2.5 TFLOPS and the TPPDP of the GPU is approximately 1.2 TFLOPS according to the NVIDIA documentation[5]. Therefore, we expect the CPU to achieve a theoretical speedup of approximately two compared to the GPU on this server. Both systems run Linux CentOS Linux 7, Intel oneAPI (release 2022.2, May 2022) and CUDA/cuBLAS (release 11.6).

4.1 Results

The limits of the proposed approach for both testbeds have been explored in this section. Thus, in the experimentation, we have measured the complexity of the developed driver as a function of the size of the target matrices. In this sense, note that the maximum number of operations analogous to Eq. 16 that can be carried out simultaneously is given by the size of the GPU and CPU memory, and thus by the size of the matrices considered. Table 1 summarizes the number of simultaneous operations, depending on the number of channels of the input audio mixtures, that can run in the described testbeds. A column

[3] https://ark.intel.com/

[4] https://www.nvidia.com/en-us/data-center/tesla-p100/

[5] https://www.nvidia.com/en-us/design-visualization/rtx-a6000/

~~I/# channels~~	# operations	# audio frames
64	589824	575
128	147456	144
256	36864	36
512	9216	9
1024	2304	2,2
2048	576	0,6
4096	144	0,1

Table 1: Number of simultaneous operations depending on the number of channels of the input audio mixtures.

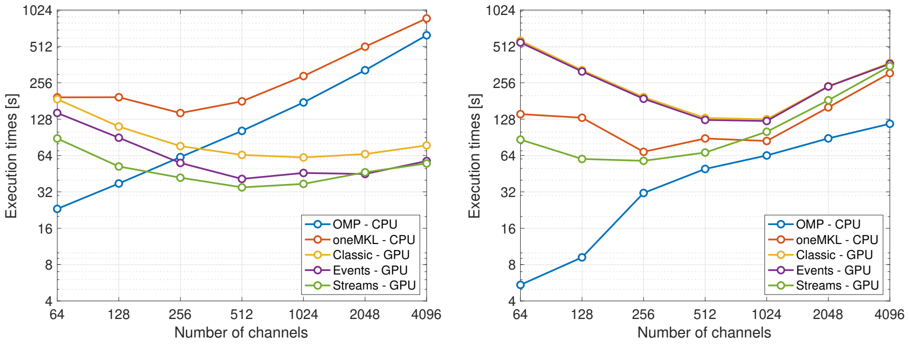
has been included to indicate the approximate number of audio frames processed in each case. ~~Note that this limitation is given by the 128 GB of CPU RAM and 16 GB of GPU RAM available for both systems.~~Note that the maximum problem size is limited by the different memory capacities of the two servers used in the experiments. One server has only 128 GB of CPU RAM and the other has 16 GB of GPU RAM. Thus, both values were chosen to determine the maximum problem size and ensure a fair comparison between both servers.

Different CPU and GPU schemes have been considered for the development of the proposed driver. Regarding the CPU approaches, the first one considered was based on the ~~Intel Math Kernel Library (MKL)~~Intel oneAPI Math Kernel Library (oneMKL). In this approach, all testbed cores were intensely exploited using the extensively parallelized matrix product instructions of the library. On the other hand, a compulsive parallelism approach were implemented based on OpenMP directives. In this case, the problem was tackled by running multiple sequential matrix products in parallel.

Concerning the GPU, several techniques were evaluated. ~~Note that techniques such as zero-copy or unified memory do not provide good results, since the memory reservation for matrices is done outside the proposed driver.~~Note that techniques such as zero-copy or unified memory were not used (or did not provide satisfactory results) in our approach, as the memory allocation for matrices is done outside the proposed driver. The first approach considered was the classical one, based on synchronous communications, in which a continuous flow of data was established to the GPU which processes the data as it arrives. Next, we proposed an approach based on streams. Finally, we combined events and streams to use massively parallel programming on the GPU.

Fig. 1 shows the results obtained in terms of execution times for the Xeon E5-2603 and Xeon Silver 4314 servers as a function of the number of channels of the input audio mixtures and the number of simultaneous operations performed.

Let us start by analyzing the performance of the Xeon E5-2603 server. ~~As can be observed, the best results for both small and large matrices are obtained by the OMP scheme regarding CPU approaches~~. As can be observed, regarding CPU approaches, the best results for both small and large matrices



(a) Execution times measured in the Xeon E5-2603 server.

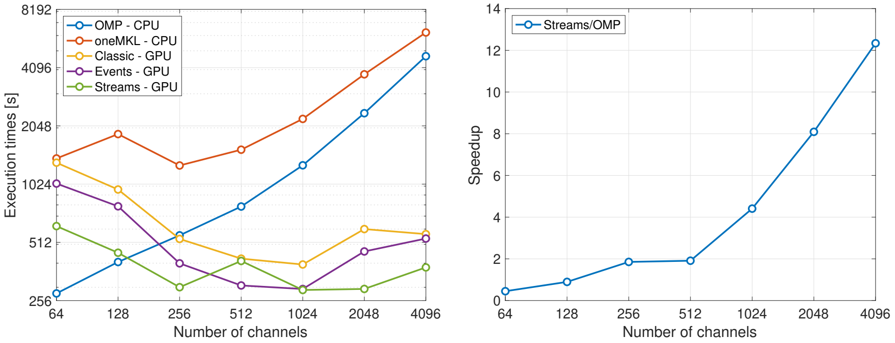
(b) Execution times measured in the Xeon Silver 4314 server.

Fig. 1: Experimental results as a function of the number of channels of the input audio mixtures and the number of simultaneous operations performed in the testbeds.

are obtained when multiple sequential matrix products are run in parallel using OpenMP directives. This strategy is referred to as OMP throughout the rest of the paper. In the particular case of very large matrices, the results obtained by the ~~MKL~~oneMKL approach are close to those obtained by OMP. These results indicates that tackling the problem with ~~MKL~~oneMKL versus OMP is only suitable when the matrices are of significant size. Concerning GPU schemes, we can observe a very similar trend for all the described approaches. In general, the stream scheme outperforms all the other approaches in terms of execution times. Note that for matrices with 2048 and 4096 channels the times obtained for the event-based approach are very similar to those based on streams. Finally, mention that for small sizes the OMP version provides a better performance than the GPU versions.

Results for the Xeon Silver 4314 are illustrated in Fig. 1b. In general, the behavior observed is similar to the one obtained by the Xeon E5-2603 server. However, in this case the OMP approach provides the best performance compared to the other systems, both for CPU and GPU. This is basically due to the use of a more powerful CPU (1200 GFLOPS) and a GPU with worse performance (1210 GFLOPS). For large sizes, the speedup of the GPU respect to the CPU is very close to one, as we already expected. Again we can see that the ~~MKL~~oneMKL approach achieves better results than the OMP approach for large matrices (i.e., 4096 channels). Finally, among all GPU approaches, the stream-based scheme obtains the best results.

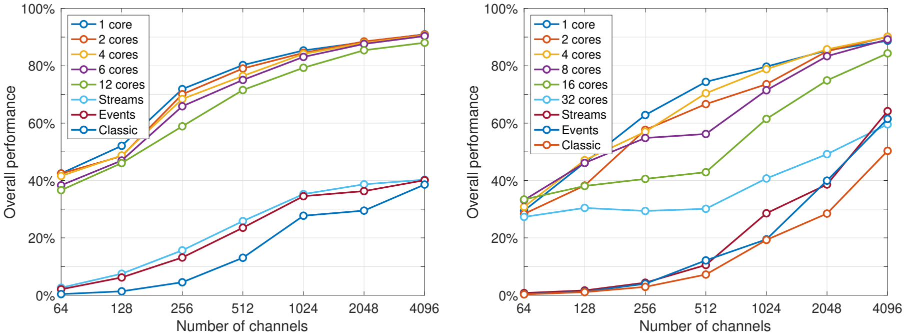
Fig. 2a shows the results obtained by the Xeon E5-2603 server limiting RAM consumption to 750 GB. This implies that approximately seven times more simultaneous operations analogous to Eq. 16 can be run. In this case we can see that the times obtained have increased by approximately an order of magnitude. In addition, all approaches have scaled as expected. As can be



(a) Execution times measured in the Xeon E5-2603 server.

(b) Speedup of the stream version compared to the OMP version.

Fig. 2: Experimental results as a function of the number of channels of the input audio mixtures in the Xeon E5-2603 server. These results were obtained by limiting RAM usage to 750 GB.



(a) Overall performance respect to the peak performance measured in the Xeon E5-2603 server.

(b) Overall performance respect to the peak performance measured in the Xeon Silver 4314 server.

Fig. 3: Overall performance respect to the peak performance as a function of the number of channels of the input audio mixtures.

observed, the stream scheme again obtains the best results. On the ohter hand, Fig. 2b depicts the speedup of the stream version compared to the OMP version for the Xeon E5-2603 server. These two approaches have been chosen because they provide the best results for GPU and CPU, respectively. As can be seen, for audio with few channels, the speedup provided by the GPU scheme with respect to the CPU scheme is not very high. However, the speedup increases as the size of the problem grows, indicating that it is more suitable to use GPU approaches for these cases. This was the expected behavior in view of the computed theoretical speedup.

Finally, to assess the resource utilization of our method, we have measured its performance in relation to the theoretical peak performance of both servers. Figure 3 displays the results as a function of the number of channels in the input audio mixtures. For the CPU results, we have chosen the OMP strategy, as it provided the best results, and we have varied the number of cores used. Additionally, we have included results for the GPU using the Streams, Events, and Classics strategies. Figure 3a shows the results for the Xeon E5-2603 server. As can be observed, the obtained performance increases as the number of channels increases, which makes sense since the problem dimension also increases. For large sizes, the performance for the CPU reaches 90%. In the case of the GPU, the performance is worse and only for large sizes is a 40% performance achieved. This can be attributed to the fact that the problem size may not be large enough to fully utilize the capabilities of the GPU, leading to lower performance compared to the CPU. On the other hand, Figure 3b shows the results for the Xeon Silver 4314 server. The behavior is similar to the other system. However, in this case, it can be observed that the performance for 32 cores decreases. This may be due to resource saturation, energy constraints, or thermal issues [33]. In terms of the GPU, the results obtained for large sizes are close to 70%.

5 Conclusion

In this paper, we present a numerical approach to address the audio source separation task based on MNMF for recordings with high number of channels, such as HOA encoding. In particular, we propose a parallel multi-architecture driver to compute the multiplicative update rules in MNMF approaches, optimizing the computation requirements and mitigating the effect of the inverse operation in the full-rank model. The proposed driver has been designed to work on both sequential and multi-core computers, as well as GPU and Intel Xeon coprocessors. The proposed software was written in C language using, as appropriate, OpenMP or the CUDA suite. The driver can also be called from numerical computing environments such as MATLAB or GNU Octave through MEX (MATLAB Executable) interfaces and from Python. The proposed solution tries to reduce the computational cost of the multiplicative update rules by using the Cholesky decomposition and by solving several triangular equation systems.

The proposal has been evaluated for different scenarios with promising results in terms of execution times for both CPU and GPU. To the best of our knowledge, our proposal is the first system that addresses the problem of reducing the computational cost of MNMF-based systems using parallel and high performance techniques.

In conclusion, our proposed solution shows promising results in terms of computational cost reduction for MNMF-based systems. In future work, we plan to integrate this driver into a sound source separation model based on MNMF and investigate its use in the ambisonic and/or spherical harmonics domain. This would allow for a more comprehensive evaluation of the proposed method and its potential applications in audio processing. Additionally, we plan to investigate the potential of the proposed driver to be integrated into other audio processing pipelines, such as audio object-based format conversion and sound field reconstruction. Overall, this work contributes to the advancement of audio processing techniques, particularly in the field of multichannel audio source separation.

Declarations

Ethical Approval. Not applicable.

Competing interests. The authors declare that there is no conflict of interest.

Authors' contributions. All authors contributed equally to this work.

Funding. This work was supported by MCIN/AEI/10.13039/501100011033 under the projects grant *PID2020-119082RB-{C21,C22}* and by Gobierno del Principado de Asturias under grant *AYUD/2021/50994*.

Availability of data and materials. Data generated during the current study are available from the corresponding author on reasonable request.

References

[1] Wien M, Boyce JM, Stockhammer T, Peng WH. Standardization Status of Immersive Video Coding. IEEE Journal on Emerging and Selected Topics in Circuits and Systems. 2019;9(1):5–17. https://doi.org/10.1109/JETCAS.2019.2898948.

[2] Parry RM, Essa I. Estimating the Spatial Position of Spectral Components in Audio. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics); 2006. p. 666–673. Available from: http://link.springer.com/10.1007/11679363_83.

[3] FitzGerald D. Non-negative tensor factorisation for sound source separation. In: IEE Irish Signals and Systems Conference 2005. vol. 2005. IEE; 2005. p. 8–12. Available from: https://digital-library.theiet.org/content/conferences/10.1049/cp_20050279.

[4] Lee S, Park SH, Sung KM. Beamspace-Domain Multichannel Nonnegative Matrix Factorization for Audio Source Separation. IEEE Signal Processing Letters. 2012 1;19(1):43–46. https://doi.org/10.1109/LSP.2011.2173192.

[5] Pezzoli M, Carabias-Orti JJ, Cobos M, Antonacci F, Sarti A. Ray-Space-Based Multichannel Nonnegative Matrix Factorization for Audio Source

Separation. IEEE Signal Processing Letters. 2021;28:369–373. https://doi.org/10.1109/LSP.2021.3055463.

[6] Bianchi L, D'Amelio F, Antonacci F, Sarti A, Tubaro S. A plenacoustic approach to acoustic signal extraction. In: 2015 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA); 2015. p. 1–5.

[7] Marković D, Antonacci F, Bianchi L, Tubaro S, Sarti A. Extraction of Acoustic Sources Through the Processing of Sound Field Maps in the Ray Space. IEEE/ACM Transactions on Audio, Speech, and Language Processing. 2016;24(12):2481–2494. https://doi.org/10.1109/TASLP.2016.2615242.

[8] Peled Y, Rafaely B. Linearly-Constrained Minimum-Variance Method for Spherical Microphone Arrays Based on Plane-Wave Decomposition of the Sound Field. IEEE Transactions on Audio, Speech, and Language Processing. 2013;21(12):2532–2540. https://doi.org/10.1109/TASL.2013.2277939.

[9] Mitsufuji Y, Takamune N, Koyama S, Saruwatari H. Multichannel Blind Source Separation Based on Evanescent-Region-Aware Non-Negative Tensor Factorization in Spherical Harmonic Domain. IEEE/ACM Transactions on Audio, Speech, and Language Processing. 2021;29:607–617. https://doi.org/10.1109/TASLP.2020.3045528.

[10] Nikunen J, Virtanen T. Direction of Arrival Based Spatial Covariance Model for Blind Sound Source Separation. IEEE/ACM Transactions on Audio, Speech, and Language Processing. 2014 3;22(3):727–739. https://doi.org/10.1109/TASLP.2014.2303576.

[11] Ozerov A, Févotte C, Vincent E. In: Makino S, editor. An Introduction to Multichannel NMF for Audio Source Separation. Cham: Springer International Publishing; 2018. p. 73–94. Available from: https://doi.org/10.1007/978-3-319-73031-8_4.

[12] Carabias-Orti JJ, Nikunen J, Virtanen T, Vera-Candeas P. Multichannel Blind Sound Source Separation Using Spatial Covariance Model With Level and Time Differences and Nonnegative Matrix Factorization. IEEE/ACM Transactions on Audio, Speech, and Language Processing. 2018 9;26(9):1512–1527. https://doi.org/10.1109/TASLP.2018.2830105.

[13] Sekiguchi K, Nugraha AA, Bando Y, Yoshii K. Fast Multichannel Source Separation Based on Jointly Diagonalizable Spatial Covariance Matrices. In: 2019 27th European Signal Processing Conference (EUSIPCO). IEEE; 2019. p. 1–5. Available from: https://ieeexplore.ieee.org/document/8902557/.

[14] Muñoz-Montoro AJ, Carabias-Orti JJ, Cabañas-Molero P, Cañadas-Quesada FJ, Ruiz-Reyes N. Multichannel Blind Music Source Separation Using Directivity-Aware MNMF With Harmonicity Constraints. IEEE Access. 2022;10:17781–17795. https://doi.org/10.1109/ACCESS.2022.3150248.

[15] Sekiguchi K, Bando Y, Nugraha AA, Fontaine M, Yoshii K. Autoregressive Fast Multichannel Nonnegative Matrix Factorization For Joint Blind Source Separation And Dereverberation. In: ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP); 2021. p. 511–515.

[16] Koyama S, Daudet L. Sparse Representation of a Spatial Sound Field in a Reverberant Environment. IEEE Journal of Selected Topics in Signal Processing. 2019;13(1):172–184. https://doi.org/10.1109/JSTSP.2019.2901127.

[17] Mitsufuji Y, Takamune N, Koyama S, Saruwatari H. Multichannel Blind Source Separation Based on Evanescent-Region-Aware Non-Negative Tensor Factorization in Spherical Harmonic Domain. IEEE ACM Trans Audio Speech Lang Process. 2021;29:607–617. https://doi.org/10.1109/TASLP.2020.3045528.

[18] Borra F, Krenn S, Gebru ID, Marković D. 1ST-Order Microphone Array System for Large Area Sound Field Recording and Reconstruction: Discussion and Preliminary Results. In: 2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA); 2019. p. 378–382.

[19] Boyd SP, Lieven V. Introduction to Applied Linear Algebra : Vectors Matrices and Least Squares. Cambridge UK: Cambridge University Press; 2018. .

[20] Ito N, Nakatani T. FastMNMF: Joint Diagonalization Based Accelerated Algorithms for Multichannel Nonnegative Matrix Factorization. In: ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE; 2019. p. 371–375. Available from: https://ieeexplore.ieee.org/document/8682291/.

[21] Sekiguchi K, Bando Y, Nugraha AA, Yoshii K, Kawahara T. Fast Multichannel Nonnegative Matrix Factorization With Directivity-Aware Jointly-Diagonalizable Spatial Covariance Matrices for Blind Source Separation. IEEE/ACM Transactions on Audio, Speech, and Language Processing. 2020;28:2610–2625. https://doi.org/10.1109/TASLP.2020.3019181.

[22] Mitsufuji Y, Uhlich S, Takamune N, Kitamura D, Koyama S, Saruwatari H. Multichannel Non-Negative Matrix Factorization Using Banded Spatial Covariance Matrices in Wavenumber Domain. IEEE/ACM Transactions on Audio, Speech, and Language Processing. 2020;28:49–60. https://doi.org/10.1109/TASLP.2019.2948770.

[23] Sawada H, Kameoka H, Araki S, Ueda N. Multichannel Extensions of Non-Negative Matrix Factorization With Complex-Valued Data. IEEE Transactions on Audio, Speech, and Language Processing. 2013 5;21(5):971–982. https://doi.org/10.1109/TASL.2013.2239990.

[24] Nørholm SM, Jensen JR, Christensen MG. Instantaneous Fundamental Frequency Estimation With Optimal Segmentation for Nonstationary Voiced Speech. IEEE/ACM Transactions on Audio, Speech, and Language Processing. 2016;24(12):2354–2367. https://doi.org/10.1109/TASLP.2016.2608948.

[25] Kuklasiński A, Doclo S, Jensen SH, Jensen J. Maximum Likelihood PSD Estimation for Speech Enhancement in Reverberation and Noise. IEEE/ACM Transactions on Audio, Speech, and Language Processing. 2016;24(9):1599–1612. https://doi.org/10.1109/TASLP.2016.2573591.

[26] Liutkus A, Badeau R, Richard G. Gaussian Processes for Underdetermined Source Separation. IEEE Transactions on Signal Processing. 2011;59(7):3155–3167. https://doi.org/10.1109/TSP.2011.2119315.

[27] Yoshii K, Tomioka R, Mochihashi D, Goto M. Beyond NMF: Time-Domain Audio Source Separation without Phase Reconstruction. In: ISMIR; 2013. p. 369–374.

[28] Kim J, Park H. Fast Nonnegative Matrix Factorization: An Active-Set-Like Method and Comparisons. SIAM Journal on Scientific Computing. 2011 1;33(6):3261–3281. https://doi.org/10.1137/110821172.

[29] Alostad JM. Reducing Dimensionality Using NMF Based Cholesky Decomposition. In: Proceedings of the International Conference on Research in Adaptive and Convergent Systems. RACS '17. New York, NY, USA: Association for Computing Machinery; 2017. p. 49–55. Available from: https://doi.org/10.1145/3129676.3129697.

[30] Nikunen J, Politis A. Multichannel NMF for Source Separation with Ambisonic Signals. In: 2018 16th International Workshop on Acoustic Signal Enhancement (IWAENC). IEEE; 2018. p. 251–255. Available from: https://ieeexplore.ieee.org/document/8521344/.

[31] Sawada H, Araki S, Makino S. Underdetermined Convolutive Blind Source Separation via Frequency Bin-Wise Clustering and Permutation Alignment. IEEE Transactions on Audio, Speech, and Language Processing. 2011 3;19(3):516–527. https://doi.org/10.1109/TASL.2010.2051355.

[32] Frigo M, Johnson SG. The Design and Implementation of FFTW3. Proceedings of the IEEE. 2005 feb;93(2):216–231. https://doi.org/10.1109/JPROC.2004.840301.

[33] Ferreira Lima JV, Raïs I, Lefevre L, Gautier T. Performance and energy analysis of OpenMP runtime systems with dense linear algebra algorithms. The International Journal of High Performance Computing Applications. 2019;33(3):431–443.