Published in AJSE, Vol:23, Issue: 3
Received on 14th K_w 2024
Revised on 1st H_l s_pw 2023
Published on 1st H_l s_pw 0.03

AIUB JOURNAL OF SCIENCE AND ENGINEERING
ISSN: 1608 – 3679 (print) 2520 – 4890 (Online)

A Performance Analysis on Matrix Factorisations in Solving Musical Instrument Source Separation Problem

Wen Kai Adrian Tang, Wei Shean Ng, How Hui Liew

***Abstract*—Audio signal decomposition breaks a mixture of musical instrument audio signals into its fundamental musical instrument components. Machine learning is one of the methods widely used in audio signal decomposition. However, the limitation of computer hardware and the complexity of the algorithm may cause the computational speed of machine learning to deteriorate. This paper aims to use the contemporary matrix factorisation to extract the fundamental musical instrument audio signal component from the mixture of musical instrument audio signals. We choose nine contemporary matrix factorisation techniques and compare their performance in separating the mixture of musical instrument audio signals. We create five scenarios with different melody complexity to test the matrix factorisation techniques. Based on the Signal-to-Noise Ratio, Nonnegative Matrix Factorisation with Kullback-Leibler Divergence (NMF-KL) is the best separation performance when the monotonic noise is not added to the mixture of musical instrument audio signals. Initially, NMF-KL has good separation when monotonic noise is added to the simple recurring mixture of musical instrument audio signals, but as the melody complexity increases the NMF-KL separation performance starts to deteriorate. Lastly, the matrix factorisation techniques do not work well when the white noise is added to the mixture of musical instrument audio signals.**

***Index Terms*—Signal processing, Signal decomposition, Matrix decomposition, Mixed source separation, Signal to noise ratio.**

I. Introduction

IMAGINE a rhythmic complex melody played by an orchestra and the output sound is the combination of different musical instruments, each contributing to a part of the melody. Signal decomposition is like breaking down the complex melody into its fundamental components. In short, signal decomposition [1] is a technique to analyse signals by representing the complex signal as the sum of simpler signals. The advantages of using signal decomposition include feature extraction [2], data compression [3], etc. Hence, the research has developed some techniques to perform signal decomposition such as the Fast Fourier Transform [4], Wavelet Transform [5], Empirical Mode Decomposition [6], Nonnegative Matrix Factorisation [7], etc.

This work was supported by Universiti Tunku Abdul Rahman (UTAR) through the UTAR Research Fund (UTARRF) under project number IPSR/RMC/UTARRF/2021-C1/N03.

W. K. A. Tang is with Universiti Tunku Abdul Rahman, Jalan Sungai Long, 43000, Kajang, Selangor, Malaysia, email: adriantwk97@1utar.my

W. S. Ng is with Universiti Tunku Abdul Rahman, Jalan Sungai Long, 43000, Kajang, Selangor, Malaysia, email: ngws@utar.edu.my

H. H. Liew is with Universiti Tunku Abdul Rahman, Jalan Sungai Long, 43000, Kajang, Selangor, Malaysia, email: liewhh@utar.edu.my

Signal decomposition has been applied in many different fields. In the field of physics, it is used in machinery fault diagnosis [8], signal denoising [9], turbulence analysis [10], etc. In civil engineering, signal decomposition determines the safety of bridges, buildings, and tunnels [11]. Other than that, signal decomposition is also applied in biomedical signal processing [12], [13] in separating the heart and lung signals using a combination method consisting of Nonnegative Matrix Factorisation and machine learning to perform the duty of auscultation for cardiopulmonary, cardiovascular and respiratory diseases. In addition, [14] used the Convolutive Nonnegative Matrix Factorisation in extracting several sound sources from monophonic input. On the other hand, [15] showed that Independent Component Analysis can solve the cocktail party problem which is a blind source separation problem.

In the paper, we aim to build a framework using matrix factorisation to perform separation on either simple or complex mixtures of musical instrument audio signals in the single-channel approach. In addition, we also test if our framework has consistent performance when the noise is added to the mixture of musical instrument audio signals. Conversely, the musical instrument source separation problem is part of the signal decomposition. Furthermore, different matrix factorisation techniques are employed to compare the separation quality achieved by each technique. There are roughly two types of matrix factorisation in numerical linear algebra. The first type is the canonical matrix factorisation which can be seen in the textbook, for example, LU factorisation [16], Singular Value Decomposition [17], Cholesky factorisation [18] and QR factorisation [19]. The second type is the contemporary matrix factorisation which is a modern technique, for example, Nonnegative Matrix Factorisation [7], Convolutive Nonnegative Matrix Factorisation [20], Interpolative Decomposition [21] and Independent Component Analysis [22].

Next, the data we use to test the framework in a mixture of musical instrument audio signals with a minimum of two musical instruments and a maximum of five musical instruments. Due to the copyright laws on music signals, our music signals data is generated using open source Musical Instrument Digital Interface (MIDI). We assume that the mixture of musical instrument audio signals is obtained by adding J number of

musical instruments as shown in (1).

$$y = \sum_{j=1}^{J} x_j, \tag{1}$$

where y is the mixture of musical instrument audio signals, x is the $J = 2, 3, 4, 5$. Initially, the mixture of musical instrument audio signals, y, is in time domain representation. Thus, we need to apply the Short-time Fourier Transform (STFT) in Librosa [23] library to transform y into a matrix representation, Y. After that, we use matrix factorisation to break the mixture of musical instrument audio signals. The matrix factorisation techniques used in this paper are Nonnegative Matrix Factorisation (NMF), Convolutive Nonnegative Matrix Factorisation (CNMF), Interpolative Decomposition (ID) and Independent Component Analysis (ICA). On the other hand, Singular Value Decomposition (SVD) from the canonical matrix factorisations is chosen as a benchmark method. Subsequently, we use the Signal-to-Noise Ratio [24], [25] to evaluate the separated musical instrument audio signal quality by each of the matrix factorisation techniques.

II. Methodology

This section discusses the framework for performing the musical instrument audio source separation using matrix factorisation techniques in the single-channel approach. The single-channel approach only needs one mixed audio signal as an input for the framework. Fig. 1 shows an overview flow on obtaining the separated musical instrument audio signals from the mixture of musical instrument audio signals via the signal-channel approach.

A. Step 1: Import the mixture of musical instrument audio

A mixture of musical instrument audio signals in WAV file format with a sample rate of 22050 Hz is input using the Librosa [23] library. 22050 Hz sample rate means there are 22050 samples per second. Python outputs a list of samples as shown below.

$$y = [a_1, a_2, a_3, \ldots, a_t],$$

where y is the mixture of musical instrument audio signals and t is the number of samples. As we input the audio signal, the Librosa library automatically represents the audio signal in the time domain. For the framework to work well, it is recommended that the input audio melody is dynamic where the audio signal amplitude changes over time.

B. Step 2: Transform the audio signal in the time domain to matrix representation using a Short-time Fourier Transform

In this paper, we are using matrix factorisation techniques to analyse the musical instrument audio signals. Thus, Short-time Fourier Transform (STFT) is applied to the mixture of musical instrument audio signals, y. In this paper, we use the Librosa library for STFT calculation, requiring certain parameters to be configured. We set the length of window signals after padding with zeros (n_fft) to 2048 which is the same as the physical duration of 93 milliseconds at a sample rate of 22050 Hz

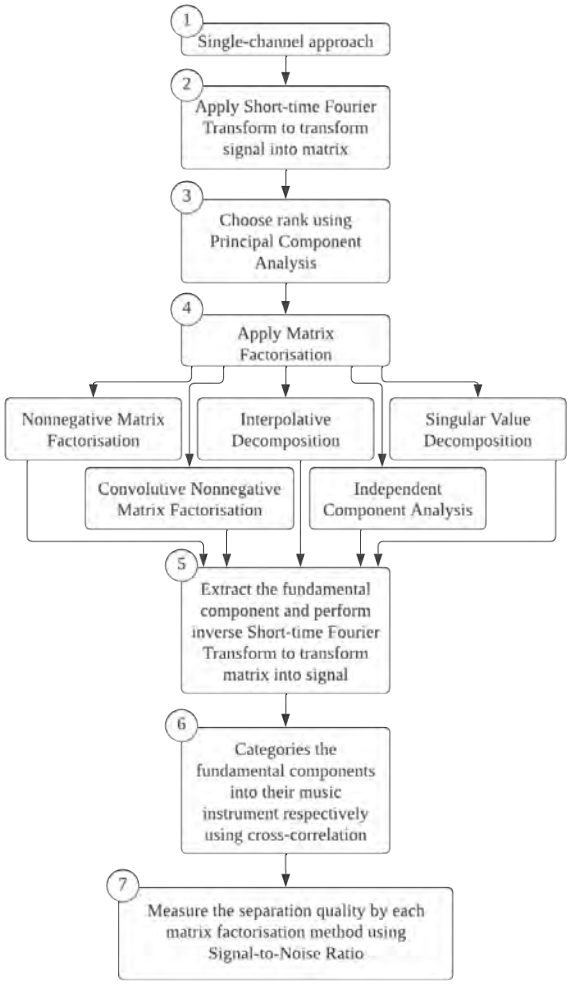


Fig. 1: Flowchart of performing musical instrument source separation using matrix factorisation in single-channel approach.

which is suitable for music signal analysis. The window length (win_length) is set to 2048 and the hop length (hop_length) is set to 512 which is the recommended parameter for music signal analysis as shown in [23]. The STFT is a method that uses the Hanning window [26] to segment y into $l \in \mathbb{R}^+$ window segments, s_l, then apply Fourier Transform on the samples under each window segment separately. Then, each s_l is inserted as a column of the matrix, $Y \in \mathbb{R}^{n \times m}$, as shown below.

$$\begin{aligned} Y &= \begin{bmatrix} | & | & | & | \\ s_1 & s_2 & \cdots & s_l \\ | & | & | & | \end{bmatrix}, \\ &= \begin{bmatrix} a_{11} + b_{11}i & a_{12} + b_{12}i & \cdots & a_{1m} + b_{1m}i \\ a_{21} + b_{21}i & a_{22} + b_{22}i & \cdots & a_{2m} + b_{2m}i \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} + b_{n1}i & a_{n2} + b_{n2}i & \cdots & a_{nm} + b_{nm}i \end{bmatrix}, \end{aligned} \tag{2}$$

where n denotes the number of columns of a matrix and m denotes the number of rows of a matrix. As shown in (2), all

the entries in Y are complex numbers. Hence, we need to take the absolute value of the matrix Y, so that all the entries are positive values and we obtain matrix, Y^{+}. At the same time, we also extract the phase, Φ, information from matrix Y using (3) and in Python, we are using "np.angle".

$$\phi_{cd} = \tan^{-1} \frac{b_{cd}}{a_{cd}}, \tag{3}$$

where $c = 1, 2, \ldots, n$ and $d = 1, 2, \ldots, m$. Thus, we obtain Φ as shown below.

$$\Phi = \begin{bmatrix} \phi_{11} & \phi_{12} & \cdots & \phi_{1m} \\ \phi_{21} & \phi_{22} & \cdots & \phi_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{n1} & \phi_{n2} & \cdots & \phi_{nm} \end{bmatrix}.$$

C. Step 3: Choose rank using explained variance ratio computed using principal component analysis

Rank, $k = \min(n, m)$, is an important parameter when performing dimensional reduction in the matrix factorisation for matrix $Y^{+} \in \mathbb{R}^{n \times m}$ to obtain the approximated matrix, $Y^{+'} \in \mathbb{R}^{n \times m}$. Choosing the wrong rank may lead to a loss of information on matrix $Y^{+'}$ when reconstructing using the product of 2 or more factorise matrices. Therefore, Principal Component Analysis (PCA) is applied to matrix Y^{+} to determine the rank by calculating the explained variance ratio for each principal component. The first principal component always has the highest explained variance ratio indicating it holds more information. The explained variance ratio value decreases from the first principal component to the last principal component. In Python, the calculation for the explained variance ratio using PCA in scikit-learn [27] and output p explained variance ratio values where $p = \min(n, m)$ is the number of principal components. Then, we use the inequality in (4) to determine the rank of the matrix factorisation.

$$\sum_{j=1}^{p} r_j \leq 0.98, \tag{4}$$

where r_j is the explained variance ratio of the jth principal component. The summation stops when the sum of the explained variance ratio exceeds 0.98. For example, if the sum of the explained variance ratio from $j = 1$ to $j = 98$ exceeds 0.98, then we take 97 as our rank. In (4), we choose 0.98 because we want to take 98% information from the matrix Y^{+}.

D. Step 4: Feature extraction by matrix factorisation

After the rank, k, is obtained, we perform matrix factorisation on the matrix Y^{+} to obtain a product of two or more simpler matrices. In this paper, we apply Nonnegative Matrix Factorisation (NMF) and Convolutive Nonnegative Matrix Factorisation (CNMF). Besides that, CNMF is an extended method from NMF which is effective in frequency-time domain analysis [28]. We also chose the randomised version of Interpolative Decomposition (RID) [21] because of its fast computational speed when dealing with large matrix dimensions. Besides that, we also choose Independent Component Analysis (ICA) as it is a popular method in blind source separation [22]. Furthermore, we choose Singular Value Decomposition (SVD) as the benchmark method in this research. Then, we look into each of the matrix factorisation methods.

1) Nonnegative Matrix Factorisation (NMF): [7] defines NMF as a technique to decompose a nonnegative matrix A into $W \in \mathbb{R}^{n \times k}$ and $H \in \mathbb{R}^{k \times m}$ matrices, where k is the rank. Then, the factorised structure is shown below.

$$A \approx A^{'} = WH,$$

where all the entries in W and H matrices are non-negative and $A^{'}$ is an approximated matrix of A by using smaller dimension of W and H matrices. Next, is to compute the W and H matrices. First, we randomly initialise entries value of W and H matrices. Then, we use the multiplicative update rules derived by [29] in (5) and (6) to obtain a new entries value for W and H matrices.

$$W \longleftarrow W \odot \frac{\left([WH + C]^{\odot(\beta-2)} \odot V \right) H^T}{[WH + C]^{\odot(\beta-1)} H^T}, \tag{5}$$

$$H \longleftarrow H \odot \frac{W^T \left([WH + C]^{\odot(\beta-2)} \odot V \right)}{W^T [WH + C]^{\odot(\beta-1)}}, \tag{6}$$

where $\odot$ denotes the element-wise matrix multiplication, $(\cdot)^{\odot}$ denotes the element-wise power, $\frac{(\cdot)}{(\cdot)}$ denotes the element-wise division and C denotes a very small positive number to prevent divisibility by zero. [30] state that there are three versions of multiplicative update rules by substituting $\beta = [0, 1, 2]$ into (5) and (6). We have the Euclidean distance version when $\beta = 2$, the Kullback-Leibler divergence version when $\beta = 1$ and the Itakura-Saito divergence when $\beta = 0$. Lastly, we stop updating the W and H matrices when the tolerance imposed by the user is fulfilled. In Python, the scikit-learn library NMF model is used in this paper.

2) Convolutive Nonnegative Matrix Factorisation (CNMF): CNMF is an extension of the NMF which can capture the short-term temporal dependencies in the time series data [31]. First, we define the operator $\overset{t\rightarrow}{(\cdot)}$ and $\overset{\leftarrow t}{(\cdot)}$ below.

$$H = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \qquad H = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

$$\overset{0\rightarrow}{H} = \begin{bmatrix} 0 & 1 & 2 \\ 0 & 4 & 5 \end{bmatrix} \qquad \overset{\leftarrow 0}{H} = \begin{bmatrix} 2 & 3 & 0 \\ 5 & 6 & 0 \end{bmatrix}$$

$$\overset{1\rightarrow}{H} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 4 \end{bmatrix} \qquad \overset{\leftarrow 1}{H} = \begin{bmatrix} 3 & 0 & 0 \\ 6 & 0 & 0 \end{bmatrix}$$

The factorisation structure of CNMF described by [14] is shown below.

$$A \approx \hat{A} = \sum_{t=0}^{T-1} W_t \overset{t\rightarrow}{H}, \tag{7}$$

where $A \in \mathbb{R}^{+, n \times m}$ is the mixed audio signal in matrix representation as input, $W_t \in \mathbb{R}^{+, n \times k}$ and $\overset{t\rightarrow}{H} \in \mathbb{R}^{+, k \times m}$ are the basis and coefficient matrix. Furthermore, the $\overset{t\rightarrow}{H}$ matrix column shifts t steps to the right and k is the rank. In this

paper, we choose $T = 2$ in (7). Then, matrix A is decompose into summation of $W_0 \cdot \overset{0\rightarrow}{H}$ and $W_1 \cdot \overset{1\rightarrow}{H}$ as shown below.

$$\hat{A} = W_0 \overset{0\rightarrow}{H} + W_1 \overset{1\rightarrow}{H}. \tag{8}$$

The steps are similar to NMF where we first randomly initialise the W and H matrices then calculate new W and H matrices using the multiplicative update rules in (9) and (10).

$$W_t \longleftarrow W_t \odot \frac{\left(\hat{A}^{\odot(\beta-2)} \odot A\right) \overset{t\rightarrow}{H}^T}{\hat{A}^{\odot(\beta-1)} \overset{t\rightarrow}{H}^T} \tag{9}$$

$$H \longleftarrow H \odot \frac{W_t^T \left(\overset{\leftarrow t}{A} \odot \left[\overset{\leftarrow t}{\hat{A}}\right]^{\odot(\beta-2)}\right)}{W_t^T \left(\overset{\leftarrow t}{\hat{A}}\right)^{\odot(\beta-1)}}, \tag{10}$$

where $\odot$ denotes the element-wise multiplication and $(\cdot)^{\odot}$ denotes the element-wise power, $\frac{(\cdot)}{(\cdot)}$ deontes the element-wise division and $t = 0, 1, \ldots, T-1$. [14] recommended to take an average t number of matrix H to form a new matrix $\bar{H}$ that is shared along all the t. In Python, the NMF Toolbox from AudioLabs is used for the CNMF calculation.

3) Interpolative Decomposition (ID): ID is defined by [21] given a matrix $A \in \mathbb{R}^{n \times m}$ and the factorisation structure is shown below.

$$A \approx CZ$$

where matrix $A \in \mathbb{R}^{n \times m}$ is the input matrix, matrix $C \in \mathbb{R}^{n \times k}$ columns are chosen from the column of matrix A and matrix $Z \in \mathbb{R}^{k \times m}$ with some conditions where some size-k subset of the column of Z form the $k \times k$ identity matrix and no entry in Z has an absolute value greater than 2. [21] proposed two different ID algorithms and in this paper Randomised Interpolative Decomposition (Optim RID) is used. The steps of computing Optim RID are as follows. First, a sampling matrix, A_s, is created where its columns are randomly chosen $k + 2$ columns from the input matrix A where k is the rank. Then, we have the column-pivoted QR factorisation which is $A_s P_k = Q_k R_k$ where Q_k is the first k columns of an orthogonal matrix, R_k is the k columns and rows of an upper triangular matrix and P_k is first k columns of a permutation matrix. Let $C = A_s P_k$. Next, we compute the matrix Z by minimising the objective function $||A - CZ||_F$ and obtain $R_k^T R_k Z = C^T A$. Lastly, we apply the diagonal pivoting method to $R_k^T R_k Z = C^T A$ to compute the matrix Z.

4) Independent Component Analysis (ICA): ICA is defined by [22] by giving an observed mixed signal represented by matrix X. Then, we can express it as the unmixing equation.

$$S = XW,$$

where W is the unmixing matrix and S is the independent matrix. Next, we can say that matrix X is decomposed into independent matrix S and mixing matrix as shown in the mixing equation below.

$$X = SA,$$

where A is the mixing matrix such that $A = W^{-1}$. ICA aims to compute the W matrix, therefore we use the Fast ICA algorithm in the scikit-learn Python library to compute it.

5) Singular Value Decomposition (SVD): [17] define SVD by letting $A \in \mathbb{R}^{n \times m}$ matrix, then

$$A = U \Sigma V^T,$$

where U and V^T matrices are orthogonal and Σ is the square diagonal matrix. The steps to calculate the decomposed matrices start by computing matrix AA^T and matrix $A^T A$. Then, the columns of U are eigenvectors of matrix AA^T, the columns of V are eigenvectors of matrix $A^T A$ and matrix Σ are the square roots of the eigenvalues of either AA^T or $A^T A$. When dealing with larger matrix sizes, the SciPy library SVD model is used to compute U, Σ and V^T matrices.

E. Step 5: Extract the fundamental musical instrument audio signal components

In section II-D, we observe that NMF, ID and ICA factorise matrix Y^+ into two matrices which are W and H. On the other hand, CNMF decomposes matrix Y^+ into two pairs of W and H matrices as shown in (8) when $T = 2$. As for our benchmark method, SVD factorises Y^+ into three matrices which are U, Σ and V^T matrices.

Then, we use (11) to compute the fundamental musical instrument audio signal component matrix, F_j, for NMF, ID and ICA where $j = 1, 2, \ldots, k$ and k is the rank found in section II-C. On the other hand, CNMF has two pairs of WH, so we need to use (11) to compute the fundamental musical instrument audio signal component matrix F_j for the first pair of WH and do the same for the second pair of WH. As for SVD, we use (12) to compute the fundamental musical instrument audio signal component matrix, F_j.

$$F_j = \begin{bmatrix} w_{1j} \\ w_{2j} \\ \vdots \\ w_{nj} \end{bmatrix} \otimes \begin{bmatrix} h_{j1} & h_{j2} & \cdots & h_{jm} \end{bmatrix}, \tag{11}$$

$$F_j = \begin{bmatrix} u_{1j} \\ u_{2j} \\ \vdots \\ u_{nj} \end{bmatrix} \otimes \sigma_{jj} \otimes \begin{bmatrix} v_{j1} & v_{j2} & \cdots & v_{jm} \end{bmatrix}, \tag{12}$$

where $\otimes$ denotes the outer product, F_j denotes the jth fundamental musical instrument audio signal component matrix, $j = 1, 2, \ldots, k$ where k is the rank. After obtaining the fundamental musical instrument audio signal component matrix, F_j, we need to reintroduce the phase information, Φ, into F_j using (13).

$$F_{j\phi} = F_j \odot \begin{bmatrix} e^{i\phi_{11}} & e^{i\phi_{12}} & \cdot & e^{i\phi_{1m}} \\ e^{i\phi_{21}} & e^{i\phi_{22}} & \cdot & e^{i\phi_{2m}} \\ \vdots & \vdots & \ddots & \vdots \\ e^{i\phi_{n1}} & e^{i\phi_{n2}} & \cdots & e^{i\phi_{nm}} \end{bmatrix} \tag{13}$$

where $\odot$ denotes the element-wise product. Lastly, we transform the fundamental musical instrument audio signal component matrix, $F_{j\phi}$, into a fundamental musical instrument audio signal component in time-domain representation, f_j, using inverse Short-time Fourier Transform where $j = 1, 2, \ldots, k$ and k is the rank.

F. Step 6: Identify the musical instrument from the fundamental musical instrument audio signal components and categorise them into their respective musical instrument using cross-correlation

From the previous step, we have extracted k fundamental musical instrument audio signal components in the time-domain, f_j where $j = 1, 2, \ldots, k$ and k is the rank. To obtain a similar separated musical instrument audio signal to the original musical instrument audio signal, we need to categorise k fundamental musical instrument audio signal components into their respective musical instrument. Hence, we use the cross-correlation method to measure the similarity between the k fundamental musical instrument audio signal components and the original musical instrument audio signals. Let o_1 and o_2 be the original musical instrument audio signal. Then, we use the correlate function below from the Numpy library [32] to calculate the cross-correlation values.

$$\text{corr1}_j = \text{correlate}\left(f_j, o_1\right),$$
$$\text{corr2}_j = \text{correlate}\left(f_j, o_2\right).$$

Then, we obtain two lists of cross-correlation values which are $\text{corr1}_j = [r_1, r_2, \ldots, r_t]$ and $\text{corr2}_j = [s_1, s_2, \ldots, s_t]$. Next, we extract the maximum value from the two lists as an indicator of which original signals that f_j belong to as $j = 1, 2, \ldots, k$ and k is the rank. If f_1 has a higher corr1_1 than corr2_1, it means that f_1 has high similarity with o_1, else it has high similarity with o_2. Assume that we have the result of f_j where $j = 1, 2, \ldots, k$ is similar to o_1 and o_2 in Table I.

TABLE I: Cross-correlation value for each f_j compare to o_1 and o_2.

f_j	o_1	o_2
f_1	High cross-correlation	Low cross-correlation
f_2	Low cross-correlation	High cross-correlation
f_3	High cross-correlation	Low cross-correlation
f_4	Low cross-correlation	High cross-correlation
$\vdots$	$\vdots$	$\vdots$
f_{j-3}	Low cross-correlation	High cross-correlation
f_{j-2}	High cross-correlation	Low cross-correlation
f_{j-1}	Low cross-correlation	High cross-correlation
f_j	High cross-correlation	Low cross-correlation

After categorising each of the f_j into their respective musical instrument, we can construct the separated o_1 and o_2 audio signals. We add up all the f_j which has a high cross-correlation with the o_1 audio as the separated o_1 audio signal, then we do the same for the separated o_2 audio signal as shown below.

$$\text{Separated } o_1 \text{ audio signal} = f_1 + f_3 + \cdots + f_{j-2} + f_j$$
$$\text{Separated } o_2 \text{ audio signal} = f_2 + f_4 + \cdots + f_{j-3} + f_{j-1}$$

When dealing with three musical instruments in the mixture audio signal, we just need to calculate additional cross-correlation value between f_j and o_3, i.e., $\text{corr3}_j = \text{correlate}\left(f_j, o_3\right)$.

G. Step 7: Performance evaluation of each matrix factorisation technique in musical instrument audio separation

From section II-F, we obtain the separated musical instrument audio signal. To evaluate the separation quality of each matrix factorisation technique, we use the Signal-to-Noise Ratio [33] as an evaluation method. The formula of SNR is shown below,

$$\text{SNR} = 10 \log \frac{\sum o^2}{\sum \left(o - x\right)^2},$$

where o denotes the original audio signal and x denotes the separated musical instrument audio signal. The higher SNR value indicates that the audio quality of the separated instrument audio signal is good.

III. Result and Discussion

[34] have discussed the computational time required to decompose a matrix for each matrix factorisation technique in this paper. We discovered that Convolutive Nonnegative Matrix Factorisation (CNMF) have the longest computational time followed by Nonnegative Matrix Factorisation (NMF). On the other hand, Interpolative Decomposition (ID) and Independent Component Analysis (ICA) have the lowest computational time. As we increase the rank, the computational time of both NMF and CNMF also increases but ID and ICA do not have a significant increase in the computational time.

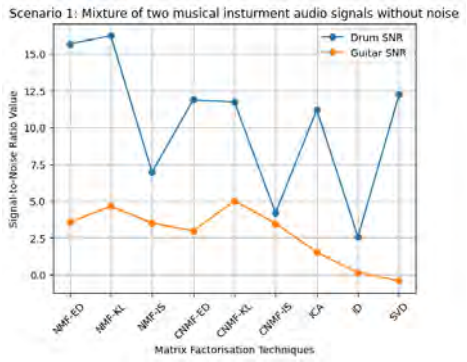
In this paper, we create five scenarios to test the framework in section II. Then, Signal-to-Noise Ratio (SNR) is used to determine the separation quality of each matrix factorisation technique. Table II lists the matrix factorisation techniques and their corresponding abbreviations used in this paper.

TABLE II: Matrix Factorisation Techniques.

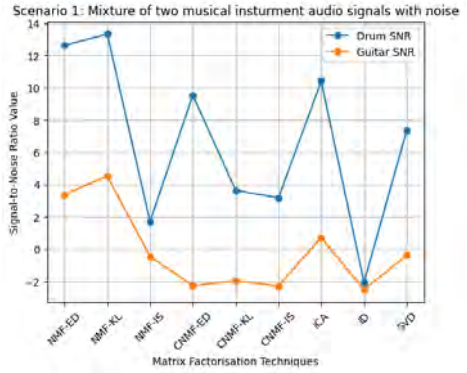
Matrix Factorisation Techniques	**Abbreviations**
Nonnegative Matrix Factorisation with Euclidean Distance	NMF-ED
Nonnegative Matrix Factorisation with Kullback-Leibler Divergence	NMF-KL
Nonnegative Matrix Factorisation with Itakura-Saito Divergence	NMF-IS
Convolutive Nonnegative Matrix Factorisation with Euclidean Distance	CNMF-ED
Convolutive Nonnegative Matrix Factorisation with Kullback-Leibler Divergence	CNMF-KL
Convolutive Nonnegative Matrix Factorisation with Itakura-Saito Divergence	CNMF-IS
Interpolative Decomposition	ID
Independent Component Analysis	ICA
Singular Value Decomposition	SVD

A. Scenario 1: Separate the mixture of two and three musical instrument audio signals with simple and recurring melody

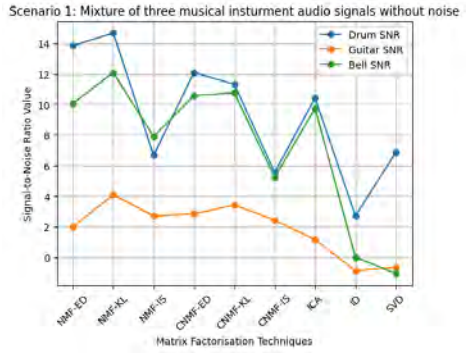
In scenario 1, we mix two independent musical instruments that is drum and guitar audio signals [35]. In this paper,



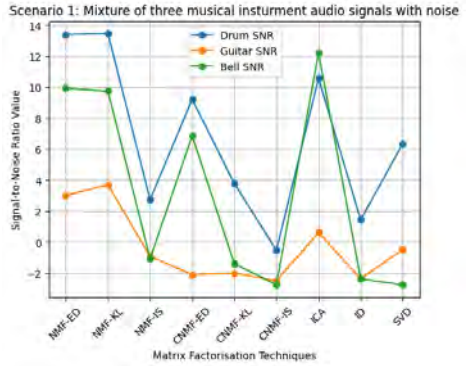
(a) Separation quality for the mixture of two musical instrument audio signals without noise.



(b) Separation quality for the mixture of two musical instrument audio signals with noise.



(c) Separation quality for the mixture of three musical instrument audio signals without noise.



(d) Separation quality for the mixture of three musical instrument audio signals with noise.

Fig. 2: Performance evaluation for separation of mixtures of musical instrument audio signals from each of the matrix factorisation techniques for scenario 1.

independent musical instruments mean that drum and guitar audio signals are not from the same musical melody. The description of this mixed musical instrument audio signal is a simple, short and recurring melody. The melody is recurring for every four seconds. From section II-C, we compute our rank as 15. First, we observe the separation quality for the mixture of two musical instrument audio signals without noise as shown in Fig. 2(a). We can observe that NMF-KL has the highest Drum SNR value and CNMF-KL has the highest guitar SNR value. However, we cannot conclude that CNMF-KL has good separation quality because it has a lower drum SNR value as compared to NMF-KL. Therefore, we can say that NMF-KL has the best separation quality for both the separated drum and guitar audio signals. On the other hand, our benchmark method SVD has a negative guitar SNR value, this indicates that there soft drum melody in the separated guitar audio signal that is not able to be separated. Similarly, ID has a bad separation quality on the guitar audio signal as the SNR value is close to zero.

Most techniques perform well in separating the two musical instruments mixed audio signals as shown in Fig, 2(a). Hence, we add one more musical instrument audio signal into the mixed audio signal and turn it into a mixture of three musical instrument audio signals. The reason is to observe whether the framework can still perform well in separating the mixture of three musical instrument audio signals as shown in Fig. 2(c). We need to compute the rank again as we have inserted additional musical instrument audio signals into the mixed audio signal. Hence, We computed the rank and found it to be 17. As we compare Fig, 2(c) with Fig. 2(a), we can observe that the pattern for both drum and guitar SNR values is similar. Hence, we can say that NMF-KL continues to be the best-performing technique as the SNR value for drum, guitar and bell is high. Even though from the y-axis in Fig. 2(c) and Fig. 2(a), we can observe the SNR value drop for each matrix factorisation technique. This means that the separation quality drops when separating the mixture of three musical instrument audio signals, however, it does not affect that the separated drum, guitar and bell audio melodies are similar to their original counterpart when we hear with human ears for each of the melodies. Human ears are not sensitive enough to identify the minor changes between the original musical instrument audio signals and the separated musical instrument audio signals. On the other hand, the separation performance of ID and SVD is shown in Fig. 2(c) as the separated guitar and bell audio signals have a negative SNR value.

After observing the performance of each matrix factorisation technique in the separation for the mixture of two and three musical instrument audio signals. We know that the mixed audio signal data is not always clean as they have noise in it. Hence, we want to test if the framework can obtain the separated musical instrument audio signal by creating a monotonic noise signal using a sine wave with 1000 Hz. First, we look into the separation for a mixture of two musical instrument audio signals with noise in Fig. 2(b). Initially from Fig. 2(a), all the techniques' SNR values are positive except for SVD. However, techniques like NMF-IS, CNMF-ED, CNMF-KL, CNMF-IS, ID and SVD have a poor performance in

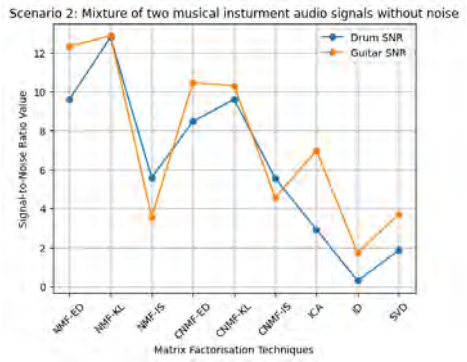
obtaining the separated guitar audio signal as the SNR value is negative. This means that we can still hear a soft noise sound in the separated guitar audio signal. Furthermore, the separation quality of separated drum audio signal drops significantly for NMF-IS, CNMF-KL, CNMF-IS and SVD. As for ID, it has the worst performance as both separated drum and guitar audio signals' SNR values are in the negative region. On the other hand, NMF-KL is the best-performing technique as it has the highest drum SNR and guitar SNR followed by NMF-ED. Even though we can see a drop in SNR value for NMF-KL when comparing Fig. 2(b) and Fig. 2(a), it does not affect that the audio when listening is similar to the original musical instrument audios.

Similarly, we add a monotonic noise to the mixture of three musical instrument audio signals. From Fig. 2(d), NMF-IS is the worst-performing technique as all the SNR values for the separated drum, guitar and bell audio signals are in the negative regions. Besides that, NMF-IS, CNMF-KL, ID and SVD have a bad separation quality for both separated guitar and bell audio signals. However, they have a decent separation quality for the drum audio signal. On the other hand, we can observe that NMF-KL is still the best-performing technique followed by NMF-ED. In conclusion, NMF-KL is the best-performing technique in the separation of the mixture of two and three musical instrument audio signals with or without noise as the mixture audio signal melody is simple and recurring. Even though we can observe a drop in SNR value for NMF-KL when a monotonic noise is added to the mixed audio signal it does not affect that the separated musical instrument audio is similar to the original audio signal when we hear with our human ears.

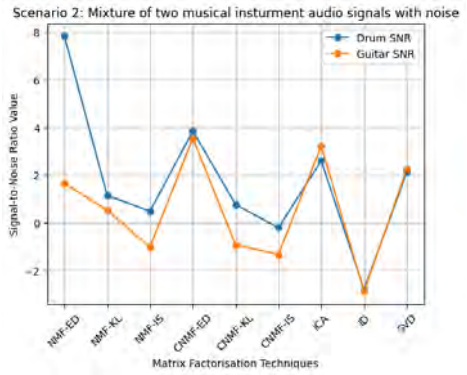
B. Scenario 2: Separate the mixture of two and three musical instrument audio signals with complex, recurring and non-recurring melody

In scenario 2, we mix two dependent drum and guitar musical instrument audio signals [36]. In this paper, dependent musical instruments denote the drum and guitar audio signals from the same musical melody. We describe this mixed musical instrument audio signal as a complex combination of recurring melody with non-recurring melody. After the computation process in section II-C, we take 94 as our rank. First, we observe the separation quality of each technique on the mixture of two musical instrument audio signals as shown in Fig. 3(a). Overall, we can observe that all the drum and guitar SNR values are positive. From Fig. 3(a), we can observe NMF-KL is still the best-performing technique as it has the highest drum and guitar SNR values indicating that the separated drum and guitar audio signal is similar to the original counterpart. On the other hand, separated drum and guitar audio signals produced by ID and SVD have low separation quality. We are still able to hear the separated audio melody, but the separated audio signal melody is not as smooth as the original melody.

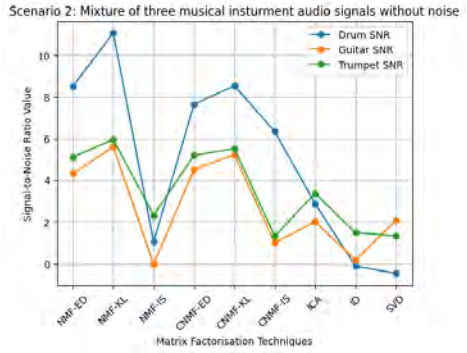
Similar to scenario 1, we add a musical instrument audio signal which is trumpet audio into the mixed audio signal and make it a mixture of three musical instrument audio



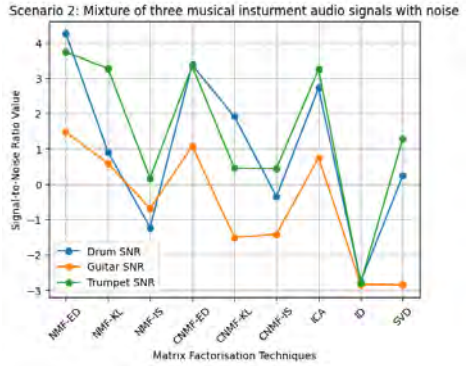
(a) Separation quality for the mixture of two musical instrument audio signals without noise.



(b) Separation quality for the mixture of two musical instrument audio signals with noise.



(c) Separation quality for the mixture of three musical instrument audio signals without noise.



(d) Separation quality for the mixture of three musical instrument audio signals with noise

Fig. 3: Performance evaluation for separation of mixtures of musical instrument audio signals from each of the matrix factorisation techniques for scenario 2.

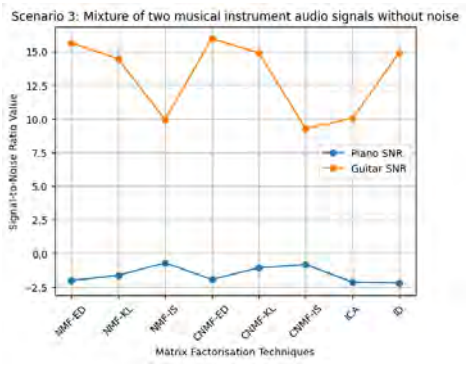
signals. The difference between scenario 1 and scenario 2 is the loudness of the added musical instrument audio signal. The newly added trumpet audio is louder as compared to the drum and guitar audio. This shows that the trumpet is the dominating melody in the mixture of three musical instrument audio signals. Since an additional musical instrument is added, we recalculate the rank to 94. From Fig. 3(c), we observe that NMF-KL continue to be the best-performing technique as the drum, guitar and trumpet SNR value is the highest followed by NMF-ED. On the other hand, the separation quality for the separated guitar audio signal produced by ID and SVD drops from the positive region to the negative region as shown in Fig. 3(c). This shows that ID and SVD do not perform well in the musical instrument audio signal source separation problem.

After that, we also added a monotonic noise created from a sine wave with 1000 Hz to test the framework performance. By comparing Fig. 3(b) with Fig. 3(a), we can see that the separated guitar audio signal produced by NMF-IS and CNMF-KL is not similar to its original counterpart as the SNR value drop from the positive region to the negative region. Moreover, the separation quality of CNMF-IS and ID deteriorate for the separated drum and guitar audio signals when the monotonic noise is in the mixture of three musical instrument audio signals. Besides that, we have an interesting observation as NMF-KL is not the best-performing technique when it is applied to separate the mixture of two musical instrument audio signals with monotonic noise. Therefore, we continue to observe and see that NMF-ED has the highest drum SNR value and mediocre guitar SNR value shown in Fig. 3(b). Unfortunately, we cannot conclude that NMF-ED is the best-performing technique because it cannot reproduce the cymbal melody in the original drum audio signal. Hence, we can conclude that CNMF-ED is the best-performing technique as it has the second-highest drum SNR value and the highest guitar SNR value. Besides that, it can reproduce the cymbal melody in the separated drum audio signal. When comparing the y-axis from Fig. 3(a) and Fig. 3(b) there is a significant drop because the monotonic noise is not fully separated. Thus, we can hear a soft monotonic noise from each of the separated musical instrument audio signals

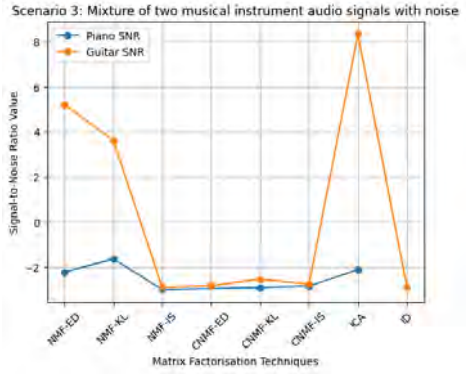
Similarly, we add the noise to the complex mixture of three musical instrument audio signals and obtain the result in Fig. 3(d). We can observe from Fig. 3(d) that the ID has the lowest separation quality with negative SNR values. This implies that the sound of the separated audio signals is not similar to their original counterpart. On the other hand, NMF-ED in Fig. 3(d) also face the same problem of reproducing the cymbal sound, even though it has the highest SNR values for drum, guitar and trumpet. Therefore, we can say that CNMF-ED is considered a high-performing technique in separation for the mixture of three musical instrument audio signals. Then, we compare the y-axis of Fig. 3(c) and Fig. 3(d) and can see the significant drop which indicates the monotonic noise is not fully separated. Thus, we can still hear with our ears that there is a monotonic noise from the separated musical instrument audio signal.

In conclusion, NMF-KL is the best-performing method when monotonic noise is not added to the complex mixture of two and three musical instrument audio signals. When noise is added, CNMF-ED produces a better separation quality for the separated musical instrument audio signals. Occasionally, we encounter problems like NMF-ED in scenario 2 as it is not able to reproduce the cymbal audio in the separated drum audio signal.

C. Scenario 3: Separate the mixture of two musical instrument audio signals where the two musical instrument sound is similar



(a) Separation quality for the mixture of two musical instrument audio signals without noise.



(b) Separation quality for the mixture of two musical instrument audio signals with noise.

Fig. 4: Performance evaluation for separation of mixtures of musical instrument audio signals from each of the matrix factorisation techniques for scenario 3.

In scenario 3, we want to test our framework performance in separating the mixture of two musical instrument audio signals which are under the same category. Hence, we mix two dependent musical instruments which are the piano and guitar [37]. This is because both piano and guitar are considered a string instrument. We take the rank to be 23 which is calculated from section II-C. From Fig. 4(a), we can observe that the matrix factorisation techniques are not able to generate a good separation quality for the separated piano audio signal as the SNR values are negative. This is because the separated piano audio signal still has the guitar audio that is not fully separated. On the other hand, all the matrix factorisation techniques produce a better separation quality for the separated guitar audio signals. From Fig. 4(a), we can see that CNMF-ED and

NMF-ED can generate better-separated guitar audio signals as the SNR value is high.

In this scenario, we also add a monotonic noise to the mixture of two musical instrument audio signals. There are some interesting observations from Fig. 4(b) that both CNMF-ED and ID are not able to generate the separated piano audio signal. We tried to increase the rank from 23 to 80 but were still not able to generate the separated piano audio signal. In the case of the separated guitar audio signal, we can observe that NMF-IS, CNMF-ED, CNMF-KL, CNMF-IS and ID have bad separation quality as shown in Fig. 4(b). This implies the monotonic noise does not fully separate from the separated guitar audio signal and we can hear the monotonic noise with our ears. On the other hand, the separated guitar audio signal produced by NMF-ED, NMF-KL and ICA are mediocre because we can still hear a soft monotonic noise and the melody is not very similar to the original guitar audio signal. This is due to the significant drop in the SNR value when compared with Fig. 4(a) and Fig. 4(b). In conclusion, this scenario shows that the framework does not perform well in separating the piano and guitar audio signals with or without the monotonic noise. This also shows that there are other possible combinations of the mixture of musical instrument audio signals that the matrix factorisation techniques are not able to separate.

D. Scenario 4: Separate the mixture of two musical instrument audio signals with simple recurring melody and white noise

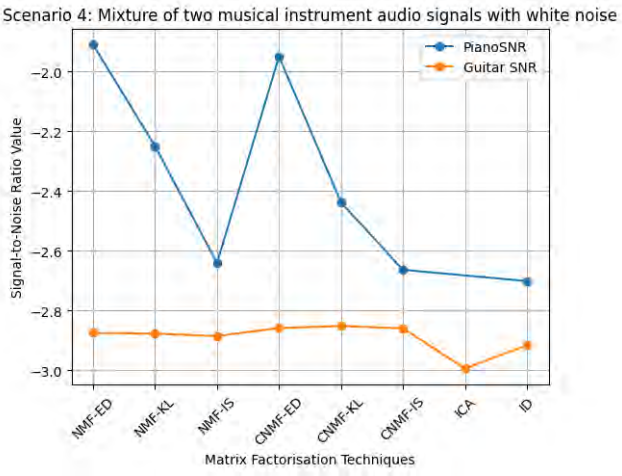


Fig. 5: Performance evaluation for separation of mixtures of musical instrument audio signals from each of the matrix factorisation techniques for scenario 3.

Previously from section III-A to section III-C, we chose our noise as a monotonic noise created using a sine wave with 1000 Hz. According to the result, we can see that the best-performing technique NMF-KL can separate the monotonic noise when the mixture of musical instrument audio signal melody is simple and recurring. However, NMF-KL not able to separate the monotonic noise when the mixture of musical instrument audio signal melody is complex, recurring and non-recurring. In this scenario, we change from monotonic noise to white noise. We create the white noise using a random sample from a normal (Gaussian) distribution [32]. Then, we add the white noise into the mixture of two musical instrument audio signals used in section III-A. When we listen to the mixed audio, the sound of the white noise is louder than the drum and guitar sound indicate the drum and guitar sound hardly can be heard. The result in Fig. 5 shows a bad separation quality for each of the matrix factorisation techniques to generate the separated drum and guitar audio signals. This scenario also shows that the matrix factorisation techniques do not perform well if one of the musical audio signal sound intensities is higher than other musical instrument audio signals in the mixture of musical instrument audio signals.

E. Scenario 5: Separate the mixture of musical instrument audio signals on different music genres

TABLE III: List of music genres.

Music Genre	Combination of Music Instrument	Rank	Citation
Blues	Drum + Guitar + Organ + Bass	92	[38]
Classical	Alto Saxophone + Piano	21	[39]
Country	Guitar + Piano + Drum + Bass	40	[40]
Disco	Bass + Drum + Guitar + Piano + Tensor Saxophone	60	[41]
Hip-Hop	Bass + Drum + Flute	14	[42]
Jazz	Bass + Drum + Piano	56	[43]
Metal	Bass + Drum + Guitar	13	[44]
Pop	Bass + Drum + Flute + Guitar + Vibraphone	41	[45]
Reggae	Bass + Drum + Electric Guitar + Marimba	54	[46]
Rock	Alto Saxophone + Bass + Drum + Piano	45	[47]

TABLE IV: Signal-to-Noise Ratio for each music genre in scenario 5.

Music Genre	SNR value of each musical instrument
Blues	Drum: 9.87; Guitar: 6.06; Organ: 2.66; Bass: 6.05
Classical	Piano: 0.13; Alto Saxophone: 10.76
Country	Drum: 1.31; Guitar: 8.65; Piano: 2.793327; Bass: 7.056608
Disco	Drum: 6.33; Guitar: -1.04; Piano: 2.10; Bass: 3.00; Tensor Saxophone: 4.00
Hip-Hop	**Drum: 9.91; Flute: 11.98; Bass: 11.24**
Jazz	Drum: 2.38; Piano: 3.78; Bass: 7.11
Metal	Drum: 13.70; Guitar: -0.70; Bass: 1.09
Pop	Drum: 1.56; Guitar: 5.66; Flute: 4.08; Bass: 3.95; Vibraphone -1.16
Reggae	Drum: 9.12; Electric Guitar: 8.50; Marimba: 6.02; Bass: 4.33
Rock	Drum: 11.58; Bass: 3.96; Alto Saxophone: 1.65; Piano: 2.86

In the previous scenario, we obverse that NMF-KL is the best-performing technique to separate the mixture of musical instrument audio signals without any noise. In scenario 5, we use NMF-KL in 10 distinct music genre melodies similar to the real-world problem. Each of the music genres is a mixture of musical instrument audio signals with a minimum of two musical instruments and a maximum of five musical instruments. The music genres, musical instrument combination and rank are presented in Table III and the result is shown in Table IV. Most of the musical instrument melodies in Table III

are medium-pace melodies where some musical instruments' audio sound intensity is higher compared to other musical instruments in the mixed audio signal. From Table IV, we observe that NMF-KL has better separation performance in the Hip-Hop melody. This is because the sound intensity of the drum, flute and bass audio signals are similar. Besides that, NMF-KL also performed mediocrely in the music genres of blues, country and jazz. The separated musical instrument audio signals in these three music genres are not as clear as the original musical instrument audio signals but the overall melody pattern when listening is similar to their original counterpart. Furthermore, we observe that some original musical instrument audio signals are not played throughout the melody. For example, the bass and guitar audio signals in metal are only played in the first three minutes. When we listen to the separated bass and guitar audio signals generated by NMF-KL, we can hear some audio sound after three minutes of the melody. This also shows that NMF-KL limitation in the separation of musical instrument audio signals which is not played throughout the melody.

F. Limitation of Interpolative Decomposition

In this paper, we are using the randomised version of the Interpolative Decomposition proposed by [21]. The advantage of ID is fast computation time and computational time does not have a significant increase when we increase the size of the matrix and rank. From section III-A and III-B, ID does not have good separation performance to generate the separated musical instrument audio signal. The reason for the bad performance is the ID needs to create the signal sampling matrix Y_s^+ where the columns of the matrix are randomly chosen from the matrix Y^+. We already know that some musical instrument audio signals are not played throughout the melody. This implies that the randomly chosen columns may not contain information about that particular musical instrument. Thus, ID is not able to generate that particular musical instrument's audio signal.

IV. Conclusion

This paper aims to investigate the performance of matrix factorisation techniques when applied to solve musical instrument audio signal source separation problems. We have created five different scenarios to test the matrix factorisation techniques' performance. We found out that Nonnegative Matrix Factorisation with Kullback-Leibler divergence (NMF-KL) is the best-performing technique. However, adding noise into the complex mixture of musical instrument audio signals affects the separation performance of NMF-KL. In addition, the matrix factorisation techniques do not perform well when white noise is added to the mixture of musical instrument audio signals. In addition, NMF-KL fails to provide a good separation quality for which the musical instrument is not played throughout the melody. In future, we would like to study on depth of choosing the correct T value in (7). The separation performance of CNMF may increase if the correct T is chosen. We also would like to compare the separation performance of matrix factorisation techniques with the existing techniques in the future. Lastly, the limitation of the framework is the original musical instrument audio signal sources must be known. However, this may not be applicable in real-world problems as the original sources are unknown most of the time. In future studies, we plan to include open music data such as OpenMIC-2018 [48] to test performance of the framework. We also plan to include other noise data such as the real-world noisy environments, in our coming research. Besides the matrix factorisation methods discussed in this paper, we will incorporate deep learning methods such as Convolutional Neural Network to observe their performance in solving musical instrument source separation problem.

References

[1] M. Civera and C. Surace, "A comparative analysis decomposition techniques for structural health monitoring on an experimental benchmark," *Sensors*, vol. 21, no. 5, p. 1825, Mar. 2021. DOI: doi:https://doi.org/10.3390/s21051825.

[2] Y. Li, S. Jiao and X. Gao, "A novel signal feature extraction technology based on empirical wavelet transform and reverse dispersion entropy," *Def. Technol.*, vol. 17, no. 5, pp. 1625-1635, Oct. 2021. DOI: doi:https://doi.org/10.1016/j.dt.2020.09.001.

[3] S. Lin, Y. Liu and Y. Li, "Seismic Data Compression By Signal Decomposition," *SEG Tech. Program 14 Expand. Abstr.*, Jan. 1997. DOI: doi:https://doi.org/10.1190/1.1885654.

[4] P. Duhamel and M. Vetterli, "Fast Fourier transform: A tutorial review and a state of the art," *Signal Process.*, vol. 19, no. 4, pp. 259-299, Apr. 1990. DOI: doi:https://doi.org/10.1016/0165-1684(90)90158-U.

[5] D. Zhang, "Wavelet Transform," in *Fundamentals of Image Data Mining*, Switzerland: Springer, 2019, pp. 35-44. [Online]. Available: doi:https://doi.org/10.1007/978-3-030-17989-2_3.

[6] A. Zeiler, R. Faltermeier, I. R. Keck, A. M. Tomé, C. G. Puntonet, and E. W. Lang, "Empirical Mode Decomposition - an introduction," *IEEE Xplore*, pp. 1-8, Jul. 2010. DOI: doi:https://doi.org/10.1109/IJCNN.2010.5596829.

[7] D. Lee and H. S. Seung, "Algorithms for Non-negative Matrix Factorization", in *NIPS*, 2000. [Online] Available: https://proceedings.neurips.cc/paper_files/paper/2000/file/f9d1152547c0bde01830b7e8bd60024c-Paper.pdf.

[8] Z. Feng, D. H. Zhang, and M. J. Zuo, "Adaptive Mode Decomposition Methods and Their Applications in Signal Analysis for Machinery Fault Diagnosis: A Review With Examples," *IEEE Access*, vol. 5, pp. 24301-24331, Oct. 2017. DOI: doi:https://doi.org/10.1109/access.2017.2766232.

[9] D. Iatsenko, P. V. E. McClintock and A. Stefanovska, "Nonlinear Mode Decomposition: a noise-robust, adaptive decomposition method," *Phys. Rev. E*, vol. 92, p. 032916, Sep. 2015. DOI: doi:https://doi.org/10.1103/PhysRevE.92.032916.

[10] T. Liu, Z. Luo, J.-H. Huang, and S. Yan, "A Comparative Study of Four Kinds of Adaptive Decomposition Algorithms and Their Applications," *Sensors*, vol. 18, no. 7, pp. 2120-2120, Jul. 2018. DOI: doi:https://doi.org/10.3390/s18072120.

[11] R. Ganguli, "Introduction," in *Structural Health Monitoring: A Non-Deterministic Framework*, Singapore: Springer Singapore, 2020, pp. 1-5. [Online]. Available: https://doi.org/10.1007/978-981-15-4988-5_1.

[12] Y. Torabi, S. Shirani, and J. P. Reilly, "A New Non-Negative Matrix Factorization Approach for Blind Source Separation of Cardiovascular and Respiratory Sound Based on the Periodicity of Heart and Lung Function," *arXiv.org*, May 2023. DOI: doi:https://doi.org/10.48550/arXiv.2305.01889.

[13] W. Wang, S. Wang, D. Qin, Y. Fang, and Y. Zheng, "Heart-lung sound separation by nonnegative matrix factorization and deep learning," *Biomed. Signal Process. Control.*, vol. 79, pp. 104180-104180, Jan. 2023. DOI: doi:https://doi.org/10.1016/j.bspc.2022.104180.

[14] P. Smaragdis, "Convolutive speech bases and their application to supervised speech separation," *TASLP*, vol. 15, no. 1, pp. 1-12, 2007. DOI: doi:https://dx.doi.org/10.1109/TASL.2006.876726.

[15] P. G. Parande and T. Thomas, "A study of the cocktail party problem," in *ICECTA*, 2017, pp. 1-5. [Online]. Available: doi:https://doi.org/10.1109/ICECTA.2017.8251979.

[16] J. M. Elble and N. V. Sahinidis, “A review of LU factorisation in the simplex algorithm,” *Int. J. Math. Oper. Res.*, vol. 4, no. 4, pp. 319-365, Sep. 2012. DOI: doi:http://dx.doi.org/10.1504/IJMOR.2012.048900.
[17] G. W. Stewart, “On the Early History of the Singular Value Decomposition”, *SIAM Rev. Soc. Ind. Appl. Math.*, vol. 35, no. 4, pp. 551-566, Dec. 1993. DOI: doi:https://doi.org/10.1137/1035134.
[18] N. J. Higham, “Cholesky factorisation,” *Wiley Interdiscip. Rev. Comput. Stat.*, vol. 1, no. 2, pp. 251–254, Jun. 2009. DOI: doi:https://doi.org/10.1002/wics.18.
[19] Nugraha and T. Basaruddin, ”Analysis and comparison of QR decomposition algorithm in some types of matrix,” in *FedCSIS*, 2012, pp. 561-565. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6354470.
[20] H. Wu, A. Marmoret and J. E. “Cohen, Semi-Supervised Convolutive NMF for Automatic Piano Transcript”, *arXiv (Cornell University)*, Jan. 2022. DOI: doi:https://doi.org/10.48550/arXiv.2202.04989.
[21] R. Advani and S. O’Hagan, ”Efficient Algorithms for Constructing an Interpolative Decomposition,” *arXiv (Cornell University)*, Jan. 2021. DOI: doi:https://doi.org/10.48550/arXiv.2105.07076.
[22] R. M. Parry and I. Essa, “Estimating the Spatial Position of Spectral Components in Audio,” in *ICA*, 2006, pp. 666-673. [Online]. Available: doi:https://doi.org/10.1007/11679363_83.
[23] B. Mcfee et al., “librosa: Audio and Music Signal Analysis in Python,” in *Proc. Python Sci. Conf.*, 2015, pp. 18-24. [Online]. Available: doi:https://doi.org/10.25080/Majora-7b98e3ed-003.
[24] G. Carleo et al., “Machine learning in the physical sciences,” *Rev. Mod. Phys.*, vol. 91, no. 4, p. 045002, Dec. 2019. DOI: doi:https://doi.org/10.1103/RevModPhys.91.045002.
[25] E. Vincent, R. Gribonval and C. Fevotte, “Performance measurement in blind audio source separation,” *TASLP*, vol. 14, no. 4, pp. 1462-1469, Jul. 2006. DOI: doi:https://doi.org/10.1109/TSA.2005.858005.
[26] P. Podder, T. Zaman Khan, M. Haque Khan, and M. Muktadir Rahman, “Comparative Performance Analysis of Hamming, Hanning and Blackman Window,” *Int. J. Comput. Appl.*, vol. 96, no. 18, pp. 1–7, Jun. 2014, DOI: doi:https://doi.org/10.5120/16891-6927.
[27] F. Pedregosa et al., “Scikit-learn: Machine Learning in Python,” *J. Mach. Learn. Res.*, vol. 12, no. 85, pp. 2825-2830, 2011. Available: http://jmlr.org/papers/v12/pedregosa11a.html.
[28] Y. X. Wang and Y. J. Zhang, “Nonnegative matrix factorisation: A comprehensive review,” *IEEE T. Knowl. Data Eng.*, vol. 25, no. 6, pp. 1336-1353, 2013. DOI: doi:https://doi.org/10.1109/TKDE.2012.51.
[29] J. J. Burred, ”Detailed derivation of multiplicative updates rules for NMF.” J.J. Burred. https://www.jjburred.com/research/pdf/jjburred_nmf_updates.pdf.
[30] C. Févotte, N. Bertin, and J.-L. Durrieu, “Nonnegative Matrix Factorization with the Itakura-Saito Divergence: With Application to Music Analysis,” *Neural. Comput.*, vol. 21, no. 3, pp. 793–830, Mar. 2009. DOI: doi:https://doi.org/10.1162/neco.2008.04-08-771.
[31] P. D. O’Grady and B. A. Pearlmutter, “Convolutive non-negative matrix factorisation with a sparseness constraint,” in *MLSP*, 2006, pp. 427-432. [Online]. Available: doi:https://doi.org/10.1109/MLSP.2006.275588.
[32] C. R. Harris et al., “Array programming with NumPy,” *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020, DOI: doi:https://doi.org/10.1038/s41586-020-2649-2.
[33] N. Detlefsen et al., “TorchMetrics - Measuring Reproducibility in PyTorch,” *JOSS*, vol. 7, no. 70, p. 4101, Feb. 2022. DOI: doi:https://doi.org/10.21105/joss.04101.
[34] W. K. A. Tang, W. S. Ng and H. H. Liew, “Separation of two musical instruments using matrix factorisation techniques,” *Int. J. Appl. Math.*, vol. 36, no. 3, pp. 425-436, 2023. DOI: doi:https://dx.doi.org/10.12732/ijam.v36i3.8.
[35] S. Tjoa, “musicinformationretrieval.” Github. https://github.com/stevetjoa/musicinformationretrieval.com/tree/gh-pages/audio.
[36] “Micheal Jackson – beat it.mid – free midi.” BitMidi. https://bitmidi.com/michael-jackson-beat-it-mid#google_vignette.
[37] “taylor swift-ours.mid — free midi.” BitMidi. https://bitmidi.com/taylor_swift-ours-mid.
[38] Freddie king midi files - download for free, midiworld.com. Available from: https://www.midiworld.com/files/921/
[39] Boccherini’s Minuet arranged for Alto Sax and Piano, mflies. Available from: https://www.mfiles.co.uk/classical-midi.htm
[40] I’m So Lonesome I Could Cry Midi, FreeMidi.org. Available from: https://freemidi.org/download3-9934-im-so-lonesome-i-could-cry-hank-williams
[41] BEEGEES.Night fever.mid, BitMidi, 2018. Available from: https://bitmidi.com/beegees-night-fever-mid#google_vignette
[42] DJ Khaled ft. Justin Bieber - I’m the One MIDI, Carlo’s MIDI, 2017. Available from: DJKhaledft.JustinBieber-I’mtheOneMIDI
[43] mo’ better blues - archivio midi files Jazz, mobetterblues. Available from: http://www.mobetterblues.altervista.org/midi_jazz/standard.html
[44] Black Sabbath-Behind The Wall of Sleep, Metal MIDI. Available from: https://metal-midi.grahamdowney.com/midi.html
[45] Shawn Mendes & Camila Cabello - Señorita MIDI, Carlo’s MIDI, 2019. Available from: https://www.cprato.com/en/midi/details/352/shawn-mendes-camila-cabello-senorita
[46] REGGAE.MID, BitMidi, 2018. Available from: https://bitmidi.com/reggae-mid
[47] Coldplay-clocks_version_2.mid, BitMidi, 2018. Available from: https://bitmidi.com/coldplay-clocks_version_2-mid
[48] Papers with Code - OpenMIC-2018 Dataset, Paperwithcode, 2018. Available from: https://paperswithcode.com/dataset/openmic-2018

Tang Wen Kai , Adrian was born in Selangor, Malaysia in 1997. He hold a Bachelor’s Degree (Honours) in Applied Mathematics with Computing from Universiti Tunku Abdul Rahman (UTAR), Malaysia, earned in 2021, and a Master of Science degree, from Universiti Tunku Abdul Rahman (UTAR), Malaysia, completed in 2024. His research focuses on matrix decomposition, image processing, audio processing, etc.

Dr. Ng Wei Shean is an Assistant Professor at the Department of Mathematical and Actuarial Sciences, Universiti Tunku Abdul Rahman (UTAR), Malaysia. She received a BSc. (Hons) degree in Mathematics from University of Malaya, Malaysia, an MSc. Mathematics from the National University of Singapore, Singapore, and a PhD in Mathematics from the University of Malaya, Malaysia. Her research interests include preserver problems, matrix decomposition, combinatorics, graph theory, etc.

Dr. Liew How Hui was born in Kuala Lumpur, Malaysia in 1976. He received the BEngg (Honours) degree in Electronics Engineering from University of Sussex, in 1997 and the Master of Science, PhD in Science in applied mathematics from Tsinghua University, in 2000 and 2003 respectively. He has joined the Department of Computing and IT in Taylor’s College from 2004 to 2006. Since 2007, he has been an Assistant Professor with the Department of Mathematical and Actuarial Sciences, Universiti Tunku Abdul Rahman. His research interests include applied numerical methods, statistical simulations, formal mathematics and applied computing. He has worked with the colleagues to publish works on applied computing, universal portfolios, etc. since 2015.