Deep Griffin–Lim Iteration: Trainable Iterative Phase Reconstruction Using Neural Network

Yoshiki Masuyama, *Student Member, IEEE*, Kohei Yatabe, *Member, IEEE*, Yuma Koizumi, *Member, IEEE*, Yasuhiro Oikawa, *Member, IEEE*, and Noboru Harada, *Senior Member, IEEE*

***Abstract*—In this paper, we propose a phase reconstruction framework, named Deep Griffin–Lim Iteration (DeGLI). Phase reconstruction is a fundamental technique for improving the quality of sound obtained through some process in the time-frequency domain. It has been shown that the recent methods using deep neural networks (DNN) outperformed the conventional iterative phase reconstruction methods such as the Griffin–Lim algorithm (GLA). However, the computational cost of DNN-based methods is not adjustable at the time of inference, which may limit the range of applications. To address this problem, we combine the iterative structure of GLA with a DNN so that the computational cost becomes adjustable by changing the number of iterations of the proposed DNN-based component. A training method that is independent of the number of iterations for inference is also proposed to minimize the computational cost of the training. This training method, named sub-block training by denoising (SBTD), avoids recursive use of the DNN and enables training of DeGLI with a single sub-block (corresponding to one GLA iteration). Furthermore, we propose a complex DNN based on complex convolution layers with gated mechanisms and investigated its performance in terms of the proposed framework. Through several experiments, we found that DeGLI significantly improved both objective and subjective measures from GLA by incorporating the DNN, and its sound quality was comparable to those of neural vocoders.**

***Index Terms*—Griffin–Lim algorithm, spectrogram consistency, complex neural network, phase reconstruction, sub-block training by denoising (SBTD).**

I. Introduction

PHASE reconstruction of a spectrogram is an active research topic with various applications, such as speech synthesis [1]–[5], voice conversion [6], and sound source enhancement/separation [7]–[14]. As a coefficient of the short-time Fourier transform (STFT) is a complex number, it consists of magnitude and phase. While both of them are necessary for reconstructing the corresponding time-domain signal using the inverse STFT (iSTFT), many acoustical technologies only focus on the amplitude (magnitude). For instance, traditional sound source enhancement applies a real-valued time-frequency (T-F) mask, which modifies amplitude without affecting phase [15]. Another example is a recent speech synthesis approach that generates a time-domain signal by applying iSTFT to the synthesized spectrogram after phase reconstruction [1]–[5]. Phase reconstruction is necessary for such amplitude-based acoustical technologies to obtain a time-domain signal with better sound quality [16]–[19].

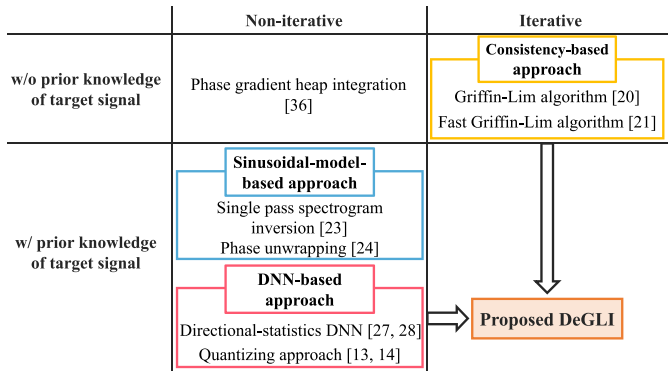


Fig. 1. Overview of the existing and proposed phase reconstruction.

Various phase reconstruction approaches have been presented, such as the consistency-based approach [20]–[22], sinusoidal-model-based approach [23], [24], deep neural network (DNN)–based approach [25]–[32], and others [33]–[38]. An overview of the existing approaches is shown in Fig. 1 to clarify the positions of each approach. Phase reconstruction approaches are categorized based on the assumed prior knowledge of the phase structure of the target signals. The consistency-based approach does not require any prior knowledge of the target signals, and it only exploits the properties of STFT [39], [40]. Since STFT is calculated with overlapping windows, adjacent T-F bins are related to each other, which is called spectrogram consistency. One of the most popular consistency-based phase reconstruction methods is the Griffin–Lim algorithm (GLA) [20] which has been applied to speech and audio synthesis [1]–[4]. However, its performance may not be superb because it does not use any prior knowledge of the target signals. The sinusoidal-model-based approach assumes that the target signal consists of a sum of slowly varying sinusoids to address this problem [23], [24]. This assumption is suitable for a large part of speech and

Manuscript received May 1, 2020; revised August 25, 2020; accepted October 12, 2020. Date of publication October 28, 2020; date of current version January 29, 2021. *(Corresponding author: Yoshiki Masuyama.)*

Yoshiki Masuyama, Kohei Yatabe, and Yasuhiro Oikawa are with the Department of Intermedia Art and Science, Waseda University, Tokyo 169-8555, Japan (e-mail: mas-03151102@akane.waseda.jp; k.yatabe@asagi.waseda.jp; yoikawa@waseda.jp).

Yuma Koizumi and Noboru Harada are with NTT Media Intelligence Laboratories, NTT Corporation, Tokyo 180-8585, Japan (e-mail: koizumi.yuma@ieee.org; noboru@ieee.org).

Color versions of one or more of the figures in this paper are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSTSP.2020.3034486

audio signals. Thanks to the assumption, the sinusoidal-model-based approach outperforms the consistency-based approach in speech enhancement [7], audio restoration [41], and audio source separation [9]. Recently, the DNN-based approach has been studied to use more prior knowledge of the target signals. DNNs can automatically discover the structures of signals in the training dataset and utilize the obtained knowledge for phase reconstruction. This approach has achieved promising results in speech synthesis [27], [28], and it has a high potential for further improvements in various applications because DNNs can acquire the knowledge of any signals [42]. The application of the approach is not restricted to specially structured signals like a sum of sinusoids.

One issue of the DNN-based approach is low flexibility in its computational cost. The number of DNN layers is fixed at the time of training, and thus the required computational cost is not adjustable at the time of inference. This nature of DNN can be restrictive for some applications because the allowable computational cost heavily depends on the device. That is, various DNNs must be designed and trained to apply the DNN-based approach on several types of devices, which is a time-consuming process. Meanwhile, the consistency-based approach is usually given by an iterative algorithm whose computational cost is controllable by changing the number of iterations. To address the issue of flexibility of the DNN-based approach, such iterative nature of the consistency-based approach should be advantageous.

In this paper, we propose a phase reconstruction framework, named *Deep Griffin–Lim Iteration (DeGLI)*, by incorporating a DNN into GLA. The proposed framework stacks a common GLA-based sub-block that contains a DNN. One of the key points of DeGLI is that the number of sub-blocks is adjustable at the time of inference, and the computational cost of DeGLI is controllable by changing the number of sub-blocks, which enables us to apply a single trained DNN to various applications whose allowable computational costs are different. In addition, by exploiting the algorithmic structure of GLA, the properties of STFT are explicitly considered, which reduces the number of parameters to be trained. We also propose a training strategy for the DNN used in DeGLI because the iterative use of a DNN is unusual. The proposed strategy trains the DNN by solving a denoising task with only one sub-block, which is called *sub-block training by denoising (SBTD)*. SBTD significantly reduces the required memory in training and improves stability comparing to end-to-end training. While an arbitrary DNN can be applied to the DeGLI framework, we present a novel complex DNN for handling complex STFT coefficients efficiently. The DNN employs gating mechanisms [43] calculated from the magnitude of complex features and the given amplitude. With the proposed training strategy and complex DNN, DeGLI is trained efficiently and performs well in terms of both objective and subjective measures. Our code and audio samples are available online.[1]

This paper intends to extend our conference paper [31], which outlined DeGLI and verified it with limited experiments. The difference between [31] and this paper is as follows:

- Relations to existing combinations of an iterative algorithm and DNN are clarified (Sections III and IV-D);
- A novel complex DNN is proposed for treating complex STFT coefficients in DeGLI (Section IV-C), and its effectiveness is validated in an experiment (Section V-G);
- The effectiveness of the proposed sub-block is verified in an experiment (Section V-E);
- The efficiency of the proposed training strategy is shown through experiments (Section V-F).

The remainder of this paper is organized as follows. In Section II, we review two phase reconstruction approaches: consistency-based and DNN-based approaches. Section III introduces some combinations of DNNs and iterative algorithms in signal processing because DeGLI combines a DNN and the iterative algorithm of GLA. Section IV presents the DeGLI framework and its effective training strategy named SBTD. A complex DNN for DeGLI is also proposed in Section IV. After investigating the effectiveness and efficiency of DeGLI in Section V, we conclude this paper in Section VI.

II. Phase Reconstruction of Spectrograms

In this section, after stating the problem of phase reconstruction, we briefly review the consistency-based and DNN-based phase reconstruction approaches that are relevant to the proposed phase reconstruction.

A. Phase Reconstruction Problem

Let STFT of a time-domain signal $\mathbf{x} \in \mathbb{R}^N$ with a window $\mathbf{g} \in \mathbb{R}^L$ $(L < N)$ be defined as

$$\mathcal{G}(\mathbf{x})[\omega, \tau] = \sum_{l=0}^{L-1} x[l + a\tau]\, g[l]\, \mathrm{e}^{-2\pi i \omega l / L} = X[\omega, \tau], \quad (1)$$

where i is the imaginary unit, a is the time shifting step, $\tau = 0, \ldots, T-1$ and $\omega = 0, \ldots, K-1$ are the time and frequency indices, respectively, and $K = L$. Here, $X[\omega, \tau]$ is the (ω, τ)th element of $\mathbf{X}$. Then, iSTFT is defined as

$$\mathcal{G}^{\dagger}(\mathbf{X})[n] = \sum_{(l,\tau) \in \mathbb{T}_n} \sum_{\omega=0}^{K-1} X[\omega, \tau]\, \widetilde{g}[l]\, \mathrm{e}^{2\pi i \omega l / L}, \quad (2)$$

where $n = 0, \ldots, N-1$ is the sample index of a time-domain signal, $\mathbb{T}_n = \{(l, \tau) \mid n = l + a\tau,\ l = 0, \ldots, L-1\}$, and $\widetilde{\mathbf{g}} \in \mathbb{R}^L$ is the canonical dual window of $\mathbf{g}$ [44].

The problem of phase reconstruction considered in this paper is to find a tuple of complex STFT coefficients $\mathbf{X}$ from its magnitude $\mathbf{A} \in \mathbb{R}_+^{K \times T}$ based on some criteria. Hereafter, we call $\mathbf{A}$ as a given amplitude. It can be written as follows:

$$\text{Find } \mathbf{X} \text{ s.t. } |X[\omega, \tau]| = A[\omega, \tau]. \quad (3)$$

The most well-accepted criterion considered with (3) is the restriction to the image of STFT. As STFT is redundant in ordinary situations, i.e., $KT > N$, its image denoted by $\mathrm{Im}(\mathcal{G})$ is a linear subspace of the space of all STFT coefficients:[2]

[1] Our project page: https://sites.google.com/view/yoshiki-masuyama/degli.

[2] We call an array of complex numbers $\mathbf{X} \in \mathbb{C}^{K \times T}$ as a tuple of complex STFT coefficients that includes $\mathbf{X} \notin \mathrm{Im}(G)$.

$\mathrm{Im}(\mathcal{G}) \subset \mathbb{C}^{K\times T}$. When $\mathbf{X}$ contains an inconsistent component that lies outside the image of STFT, the magnitude of $\mathbf{X}$ will be different after applying iSTFT $\mathcal{G}^\dagger$ and STFT $\mathcal{G}$. This means that the STFT magnitude of a reconstructed time-domain signal $\mathcal{G}^\dagger(\mathbf{X})$ is different from $\mathbf{A}$, even when $|X[\omega,\tau]| = A[\omega,\tau]$. To avoid such a change, phase reconstruction is often formulated as the following problem:

$$\text{Find } \mathbf{X} \text{ s.t.} \begin{cases} |X[\omega,\tau]| = A[\omega,\tau] \\ \mathbf{X} \in \mathrm{Im}(\mathcal{G}) \end{cases}. \tag{4}$$

Since $\mathcal{G}(\mathbf{x}) \in \mathrm{Im}(\mathcal{G})$ for any time-domain signal $\mathbf{x}$, the optimization problem given in (4) can be interpreted as a problem of finding a time-domain signal $\mathbf{x}$ whose STFT magnitude coincides with the given amplitude. As the magnitude is fixed by the first constraint, its solver attempts to find a tuple of STFT phase components $\boldsymbol{\varphi} \in [-\pi,\pi)^{K\times T}$ such that $\mathbf{A} \odot \exp(i\boldsymbol{\varphi}) \in \mathrm{Im}(\mathcal{G})$, where $\exp(\cdot)$ is the element-wise exponential function, and $\odot$ is the element-wise product.

B. Consistency-Based Phase Reconstruction

The consistency-based approach relies on the redundancy of STFT, and it attempts to solve (4) directly. A tuple of complex STFT coefficients $\mathbf{X}$ is said to be consistent when $\mathbf{X} \in \mathrm{Im}(\mathcal{G})$. The consistency-based methods force STFT coefficients to be consistent by projecting them onto $\mathrm{Im}(\mathcal{G})$ as [45]

$$P_{\mathcal{C}}(\mathbf{X}) = \mathcal{G}(\mathcal{G}^\dagger(\mathbf{X})), \tag{5}$$

where $\mathcal{C}$ is the set of tuples of complex STFT coefficients satisfying the spectrogram consistency, defined by

$$\mathcal{C} = \left\{ \mathbf{X} \in \mathbb{C}^{K\times T} \mid \mathbf{X} \in \mathrm{Im}(\mathcal{G}) \right\}. \tag{6}$$

By definition, $P_{\mathcal{C}}(\mathbf{X}) \in \mathrm{Im}(\mathcal{G})$ for any $\mathbf{X}$. Note that consistent coefficients are characterized by the fixed points of the projection, i.e., $\mathbf{X} = P_{\mathcal{C}}(\mathbf{X})$ if and only if $\mathbf{X} \in \mathrm{Im}(\mathcal{G})$.

One of the most popular consistency-based phase reconstruction methods is GLA [20]. To find consistent coefficients for the given amplitude, i.e., a solution to the problem given in (4), GLA iteratively applies two projections as follows:

$$\mathbf{X}^{[m+1]} = P_{\mathcal{C}}\big(P_{\mathcal{A}}(\mathbf{X}^{[m]})\big), \tag{7}$$

where m is the iteration count, and $P_{\mathcal{A}}$ is the projection onto the set $\mathcal{A}$ defined by[3]

$$P_{\mathcal{A}}(\mathbf{X})[\omega,\tau] = A[\omega,\tau]\frac{X[\omega,\tau]}{|X[\omega,\tau]|}. \tag{8}$$

Here, $\mathcal{A}$ is the set of complex STFT coefficients whose magnitude coincides with $\mathbf{A}$:

$$\mathcal{A} = \left\{ \mathbf{X} \in \mathbb{C}^{K\times T} \mid |X[\omega,\tau]| = A[\omega,\tau] \right\}. \tag{9}$$

One of the two constraints in (4) is enforced by each projection in GLA. By iterating them, both constraints might be satisfied simultaneously. GLA can be viewed as a projected gradient algorithm [22], and therefore its convergence towards the solution is expected if the initial value is adequate.

[3]When $|X[\omega,\tau]| = 0$ for some (ω,τ), (8) cannot be defined because of zero division. To avoid such a situation, although the projection should be set-valued when both $|X[\omega,\tau]| = 0$ and $A[\omega,\tau] \neq 0$ are satisfied at some (ω,τ), we simply replace the result of zero division by zero in this paper.

C. DNN-Based Phase Reconstruction

DNN-based phase reconstruction has been studied in recent years motivated by the strong modeling capability of DNNs. This approach takes advantage of the rich prior knowledge of the target signals that is automatically learned from a training dataset. In the DNN-based phase reconstruction, a difficulty of training mainly comes from the periodic nature of phase. Phase is wrapped in $[-\pi,\pi)$, and thus it is discontinuous at $\pm\pi$ even when phase varies smoothly. To avoid this discontinuity problem, several techniques have been developed.

One approach is to treat the phase reconstruction problem as a classification problem by quantizing the target STFT phase [13], [14]. In this approach, the DNN estimates the set of indices corresponding to the quantized target phase. By recasting to a classification problem, the DNN training does not suffer from the periodic nature of phase. In another approach [27]–[29], the output of DNN is defined by continuous real numbers whose range is not restricted to $[-\pi,\pi)$. In such a case, the DNN training suffers from the ambiguity of 2π, and therefore this approach utilizes a specific loss function based on circular statistics [27]. This approach potentially achieves higher performance as compared to the quantizing approach, because it does not suffer from the quantization error. Meanwhile, a DNN outputting complex STFT coefficients $\hat{\mathbf{X}}$ instead of STFT phases has also been investigated [30], [31]. In this case, the phase is embedded into the higher-dimensional space, or the complex plane, to avoid the periodicity. Note that the estimate of phase is obtained by taking the complex-argument of $\hat{\mathbf{X}}$, i.e., the amplitude of $\hat{\mathbf{X}}$ is discarded in phase reconstruction.

These existing DNN-based approaches have outperformed GLA [27]–[30]. The architecture of DNNs are fixed at the time of training, and their parameters are optimized for the dataset. Hence, the computational cost for the inference is not controllable afterward. This nature of DNN can be restrictive for applications whose allowable computational time varies.

III. Combinations of Iterative Algorithms and DNN in Signal Processing

Several recent studies have tried to incorporate knowledge from optimization-based signal processing into DNN. The proposed phase reconstruction framework introduced in the next section is based on such combinations of signal processing and deep learning. Thus, some of the studies are reviewed in this section for clarifying the position of the proposed framework.

A. Similarity Between Iterative Algorithm and DNN

The idea of combining an iterative algorithm and DNN is based on the structural similarity between them. In signal processing, one standard methodology is to construct an iterative

algorithm which can be written as the following abstract recursive scheme:

$$\mathbf{u}^{[m+1]} = \mathcal{S}(\mathbf{u}^{[m]}), \tag{10}$$

where $\mathbf{u}^{[m]}$ is a variable at the mth iteration, and $\mathcal{S}$ is an operator representing the update procedure of the algorithm. For example, GLA in (7) can be obtained by setting $\mathcal{S} = P_{\mathcal{C}} \circ P_{\mathcal{A}}$. This procedure can be expanded as

$$\mathbf{u}^{[M]} = \mathcal{S}(\mathcal{S}(\cdots \mathcal{S}(\mathcal{S}(\mathbf{u}^{[0]})))), \tag{11}$$

where the operator $\mathcal{S}$ is applied M times. The operator $\mathcal{S}$ is often designed for each problem based on an optimization algorithm and prior knowledge of the data.

DNN is an artificial neural network consisting of multiple layers with a large number of parameters. Each layer is a relatively simple component such as an affine transform with an element-wise nonlinear transform. Let the mth layer be denoted by $\mathcal{H}_{\boldsymbol{\theta}_m}^{[m]}$. Then, a DNN $\mathcal{F}_{\boldsymbol{\theta}}$ can be represented as

$$\mathcal{F}_{\boldsymbol{\theta}}(\mathbf{u}) = \mathcal{H}_{\boldsymbol{\theta}_M}^{[M]}(\mathcal{H}_{\boldsymbol{\theta}_{M-1}}^{[M-1]}(\cdots \mathcal{H}_{\boldsymbol{\theta}_2}^{[2]}(\mathcal{H}_{\boldsymbol{\theta}_1}^{[1]}(\mathbf{u})))), \tag{12}$$

where the DNN is assumed to have M layers, and $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_M)$ is the tuple of DNN parameters. Each layer $\mathcal{H}_{\boldsymbol{\theta}_m}^{[m]}$ is associated with its parameters $\boldsymbol{\theta}_m$ that is trained with a dataset to minimize the loss function. The architecture of a DNN is often designed empirically, and its layers are not designed specific to problems, even though their choice and combination have an impact on the quality of inference.

From (11) and (12), the similarity of an iterative algorithm and DNN is obvious. An iterative algorithm is usually based on a hand-crafted model of the target signal, and the derived procedure is often the repeated application of the same operator. In contrast, a DNN is usually designed heuristically without any explicit models, and each layer of the DNN is usually different from the others. Some research has attempted to go between them as described below.

B. Designing DNN Based on Iterative Algorithm

One way of borrowing both of the above ideas is to design a DNN architecture based on an iterative algorithm. By unfolding (or unrolling) an iterative algorithm as in (11), it can be interpreted as a DNN with common fixed parameters. By making these parameters uncommon and trainable, a DNN architecture is obtained based on a better-understood algorithm. The parameters are optimized by end-to-end backpropagation to maximize the performance at the fixed number of iterations. This strategy is called deep unfolding [46] which often achieves better performance with fewer iterations as compared to the original iterative algorithms. Its effectiveness has been confirmed in various inverse problems, such as compressed sensing [47], multi-channel input multi-channel output signal processing [48], source separation [49], [50], and others [51].

The idea of deep unfolding was originally proposed in compressed sensing [47] where the iterative shrinkage-thresholding algorithm (ISTA) was unfolded as the learned ISTA (LISTA). Fig. 2 illustrates the iterative scheme of ISTA and the architecture of LISTA. ISTA is a proximal algorithm obtained by inserting

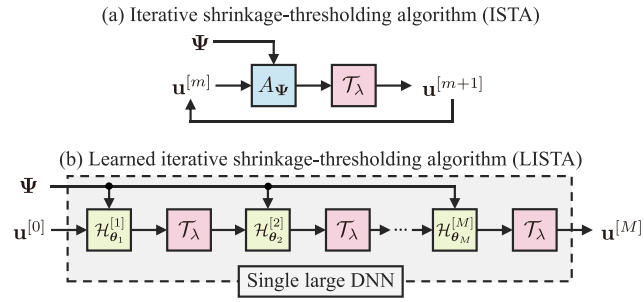


Fig. 2. Illustration of LISTA. ISTA is unfolded to form a single DNN. Blue, red, and green boxes are the affine transform, soft-thresholding operator, and single layer in DNN, respectively.

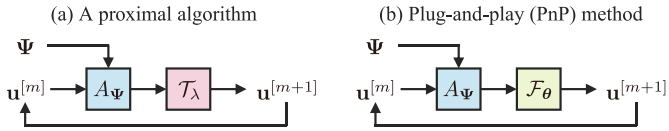


Fig. 3. Concept of PnP. A proximity operator is replaced by a denoising DNN $\mathcal{F}_{\boldsymbol{\theta}}$. Here, we consider ISTA as an example of proximal algorithm. PnP utilizes a DNN with the problem specific transform $A_{\boldsymbol{\Psi}}$.

$\mathcal{S} = \mathcal{T}_\lambda \circ A_{\boldsymbol{\Psi}}$ into (11), where $\mathcal{T}_\lambda$ is the soft-thresholding operator with a parameter $\lambda \in \mathbb{R}_+$, and $A_{\boldsymbol{\Psi}}$ is the affine transform depending on the observation $\boldsymbol{\Psi}$. LISTA unfolds it to design a DNN composed of the affine layers and the soft-thresholding as illustrated in the bottom figure of Fig. 2. As in this example, deep unfolding enables systematically designing an interpretable DNN based on an iterative algorithm whose effectiveness is validated theoretically.

C. Inserting DNN Into Iterative Algorithm

Another way of merging an iterative algorithm and DNN is to utilize DNN within the iteration of an algorithm. In this strategy, one significant challenge is how to insert DNN into an iterative algorithm because an inappropriate use of a DNN can easily collapse the well-constructed optimization algorithm. To solve this problem, several approaches have been presented including regularization by denoising (RED) and plug-and-play (PnP). RED constructs a regularization term based on a denoising DNN $\mathcal{F}_{\boldsymbol{\theta}}$ [52]. The regularization term is formulated by $\mathbf{u}^{\mathsf{T}}(\mathbf{u} - \mathcal{F}_{\boldsymbol{\theta}}(\mathbf{u}))$ that penalizes the noise component $(\mathbf{u} - \mathcal{F}_{\boldsymbol{\theta}}(\mathbf{u}))$ in some sense [53]. An optimization problem with this regularization term can be solved via its gradient. PnP inserts a denoising DNN into a proximal algorithm with the hope that the DNN acts as the proximity operator [54]–[56]. Proximal algorithms can handle multiple regularization terms by utilizing proximity operators [57], $\mathrm{prox}_f : \mathbf{u} \mapsto \arg\min_{\mathbf{v}}[f(\mathbf{v}) + \|\mathbf{v} - \mathbf{u}\|_2^2/2]$, which can be viewed as a Gaussian denoising operation based on the prior distribution $\exp(-f(\cdot))$. PnP replaces a proximity operator of a conventional regularization term by the denoising DNN as illustrated in Fig. 3. The convergence of PnP can be guaranteed by imposing some conditions on the DNN [55].

An essential aspect of these methods is that the DNN is solely trained as a denoiser. That is, unlike deep unfolding, the training

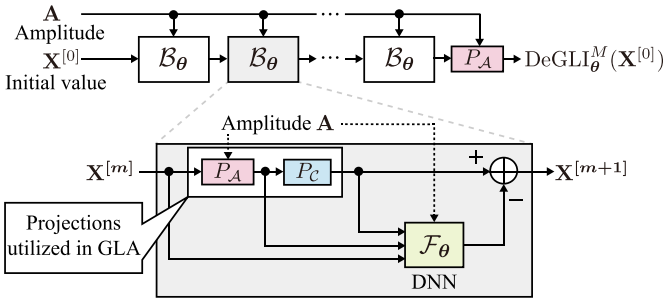


Fig. 4. Illustration of the proposed phase reconstruction framework, DeGLI.

procedure is conducted independently of an original problem and iterative algorithm. The DNN can be trained by utilizing the additive Gaussian noise, which simplifies the preparation of the training data. Applications of the denoising DNN are not restricted to a denoising problem. By inserting the DNN into an iterative algorithm for a task different from denoising, the DNN can improve the performance for that task because of a task-specific structure of the iterative algorithm. The proposed framework follows this concept: utilizing a denoising DNN within a phase-reconstruction-specific algorithm.

IV. Proposed Phase Reconstruction

By incorporating a DNN into the iterative algorithm of GLA, we propose a phase reconstruction framework, named DeGLI, in this section. First, the architecture of DeGLI, shown in Fig. 4, is introduced in Section IV-A. Then, its training strategy and a complex DNN for DeGLI are proposed in Section IV-B and IV-C, respectively. Finally, the relation between the proposed framework and the existing methods is discussed in Section IV-D.

A. Deep Griffin–Lim Iteration (DeGLI)

The motivation of proposing DeGLI is to obtain the benefits from both the DNN-based and consistency-based phase reconstruction approaches. As described in Section II-C, the DNN-based methods reconstruct a high-quality signal by exploiting the rich prior knowledge learned from a training dataset. After fixing the network architecture, its parameters are automatically optimized for a given dataset, which improves the performance of the DNN-based phase reconstruction for signals similar to those in the training dataset. One limitation of DNN-based phase reconstruction is its low flexibility to the computational cost. Since the types and number of layers are fixed at the time of training, the computational cost is not controllable at the time of inference. Therefore, one must design and train another DNN if the trained networks do not meet the computational requirement for some application. This is not the case for iterative algorithms of the consistency-based phase reconstruction (Section II-B). As the computational cost of an iterative algorithm grows linearly with the number of iterations, its cost can be adjusted by selecting the number of iterations based on the computational requirement. To take advantage of both approaches, we propose a DNN-based phase reconstruction framework via the iterative algorithm of GLA.

The proposed framework, DeGLI [4], is illustrated in Fig. 4. It stacks the same sub-block $\mathcal{B}_{\boldsymbol{\theta}}$ containing a DNN $\mathcal{F}_{\boldsymbol{\theta}}$ as

$$\mathbf{X}^{[m+1]} = \mathcal{B}_{\boldsymbol{\theta}}(\mathbf{X}^{[m]}) \tag{13}$$

$$= \mathbf{Z}^{[m]} - \mathcal{F}_{\boldsymbol{\theta}}(\mathbf{X}^{[m]}, \mathbf{Y}^{[m]}, \mathbf{Z}^{[m]}), \tag{14}$$

which is designed based on the algorithmic structure of GLA, where $\mathbf{X}^{[m]}$ is the output of mth sub-block (i.e., the same sub-block $\mathcal{B}_{\boldsymbol{\theta}}$ was applied m times), and $\mathbf{Y}^{[m]}$ and $\mathbf{Z}^{[m]}$,

$$\mathbf{Y}^{[m]} = P_{\mathcal{A}}(\mathbf{X}^{[m]}), \tag{15}$$

$$\mathbf{Z}^{[m]} = P_{\mathcal{C}}(\mathbf{Y}^{[m]}), \tag{16}$$

are intermediate variables keeping the information related to GLA given in (7). The entire architecture of DeGLI is defined by M sub-blocks $\mathcal{B}_{\boldsymbol{\theta}}$ followed by the projection $P_{\mathcal{A}}$ as follows:

$$\mathrm{DeGLI}^{M}_{\boldsymbol{\theta}}(\mathbf{X}^{[0]}) = P_{\mathcal{A}}(\mathcal{B}_{\boldsymbol{\theta}}(\mathcal{B}_{\boldsymbol{\theta}}(\cdots \mathcal{B}_{\boldsymbol{\theta}}(\mathcal{B}_{\boldsymbol{\theta}}(\mathbf{X}^{[0]}))))), \tag{17}$$

where $\mathbf{X}^{[0]}$ is a tuple of complex STFT coefficients whose magnitude coincides with $\mathbf{A}$. Note that the number of sub-blocks M can be adjusted afterward because the same DNN $\mathcal{F}_{\boldsymbol{\theta}}$ is utilized in each $\mathcal{B}_{\boldsymbol{\theta}}$, i.e., the computational cost of DeGLI can be adjusted afterward. Its computational cost grows linearly with the number of sub-blocks as that of an iterative algorithm. Therefore, DeGLI can be applied to various applications and devices, whose computational requirements may vary widely, without designing and training another DNN for each of them. In addition, the iterative use of a single DNN can reduce the number of parameters to be stored, which is desirable for memory-limited devices.

We designed the sub-block based on GLA and its variants, and thus DeGLI can be viewed as an extension of the existing Griffin–Lim type algorithms. DeGLI is reduced to GLA given in (7) when the residual estimated by the DNN $\mathcal{F}_{\boldsymbol{\theta}}$ is always zero. A modified update of GLA presented in [39] is also an instance of DeGLI given by setting $\mathcal{F}_{\boldsymbol{\theta}}(\mathbf{X}^{[m]}, \mathbf{Y}^{[m]}, \mathbf{Z}^{[m]}) = \gamma \mathbf{Y}^{[m]}$, where γ is decided by the parameters of STFT. Another variant of GLA obtained by applying an over-relaxation technique [58] is formulated by

$$\mathbf{Z}^{[m]} = P_{\mathcal{C}}(P_{\mathcal{A}}(\mathbf{X}^{[m]})), \tag{18}$$

$$\mathbf{X}^{[m+1]} = \rho \mathbf{Z}^{[m]} + (1-\rho)\mathbf{X}^{[m]}, \tag{19}$$

where $\rho > 0$. This is also an instance of DeGLI by setting $\mathcal{F}_{\boldsymbol{\theta}}(\mathbf{X}^{[m]}, \mathbf{Y}^{[m]}, \mathbf{Z}^{[m]}) = (1-\rho)(\mathbf{Z}^{[m]} - \mathbf{X}^{[m]})$. In this respect, DeGLI can be interpreted as a general and trainable extension of Griffin–Lim type phase reconstruction. The trainable extension should be able to go beyond a non-trainable Griffin–Lim type algorithm because of exploiting the prior knowledge of the target signal to phase reconstruction. Note that using a DNN in a subtractive form is called residual learning in the

[4] In our conference paper [31], the DNN in DeGLI did not utilize the given amplitude. In Section IV-C, however, we will propose a complex DNN that handles the given amplitude. Because of this difference, the diagram of DeGLI (Fig. 4) is slightly changed from that of Fig. 1 of [31]. This difference also changes the diagram of training strategy of the DNN shown in Fig. 5.

deep learning community [59], and its effectiveness has been confirmed in image denoising [60].

For the residual estimation by the DNN, the input $\mathbf{X}^{[m]}$ and intermediates $\mathbf{Y}^{[m]}$ and $\mathbf{Z}^{[m]}$ should be informative. By comparing $\mathbf{X}^{[m]}$ and $\mathbf{Y}^{[m]}$, related as $\mathbf{Y}^{[m]} = P_{\mathcal{A}}(\mathbf{X}^{[m]})$, the mismatch between the magnitude of $\mathbf{X}^{[m]}$ and the given amplitude $\mathbf{A}$ is known. Similarly, the comparison between $\mathbf{Y}^{[m]}$ and $\mathbf{Z}^{[m]}$, related as $\mathbf{Z}^{[m]} = P_{\mathcal{C}}(\mathbf{Y}^{[m]})$, gives inconsistent components of $\mathbf{Y}^{[m]}$. These mismatches should be zero for the true phase, and therefore a favorable phase reconstruction algorithm should reduce them quickly. Based on this observation, the DNN is informed about all intermediate results of GLA with the hope of generating a suitable direction toward the correct phase.

B. Sub-Block Training by Denoising (SBTD)

One unique feature of DeGLI is the iterative application of a common DNN, and its number of applications M is not fixed. Training of the DNN should be designed to be suitable for this feature. A simple way of training the DNN is the end-to-end strategy, which, in our case, can be formulated as a minimization of the following form:

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{X}^\star}[\mathcal{L}(\mathbf{X}^\star, \mathrm{DeGLI}_{\boldsymbol{\theta}}^{M}(\mathbf{X}^{[0]}))], \tag{20}$$

where $\mathbb{E}$ is the expectation operator, and $\mathcal{L}$ is a loss function. Here, $\mathbf{X}^\star$ is the complex STFT coefficients calculated from a time-domain signal in the training dataset. While such end-to-end training is straightforward, the following problems exist: (i) the number of sub-blocks M must be fixed at the time of training, which is incompatible with the concept of the DeGLI framework allowing M to be adjusted afterward; (ii) backpropagation for such iterative use of the same DNN requires significant computation as M increases; and (iii) it suffers from instability caused by $P_{\mathcal{A}}$ because it consists of element-wise multiplication and division as in (8) which may excessively expand/diminish the magnitude of the gradient calculated in backpropagation.

To circumvent these problems, we focus solely on the sub-block instead of the entire procedure. The desired property in each application of the sub-block is that the output $\mathbf{X}^{[m+1]}$ becomes closer to the solution than the input $\mathbf{X}^{[m]}$. Actually, GLA in (7) fulfills this property in terms of the spectrogram consistency [30], i.e., its iteration does not increase the energy of the inconsistent component $\|P_{\mathcal{A}}(\mathbf{X}^{[m]}) - P_{\mathcal{C}}(P_{\mathcal{A}}(\mathbf{X}^{[m]}))\|_{\mathrm{Fro}}^2$, where $\|\mathbf{X}\|_{\mathrm{Fro}}^2 = \sum_{\omega,\tau} |X[\omega,\tau]|^2$. This favorable property is induced by the fact that the projections are the minimizers of the following problems:

$$P_{\mathcal{A}}(\mathbf{X}) = \arg\min_{\mathbf{Z}} \|\mathbf{X} - \mathbf{Z}\|_{\mathrm{Fro}}^2 \ \text{ s.t. } \ \mathbf{Z} \in \mathcal{A}, \tag{21}$$

$$P_{\mathcal{C}}(\mathbf{X}) = \arg\min_{\mathbf{Z}} \|\mathbf{X} - \mathbf{Z}\|_{\mathrm{Fro}}^2 \ \text{ s.t. } \ \mathbf{Z} \in \mathcal{C}, \tag{22}$$

i.e., the projected result is the closest point to the input within the constraint set. These optimization problems can be seen as denoising problems for the Gaussian contamination model with the prior uniform within the constraint set. In other words, phase reconstruction can be realized by repeatedly applying the Gaussian denoisers that have the desired property. Hence, we propose to train the DNN by a denoising task so that the sub-block becomes a denoising map based on prior knowledge of the target signals.

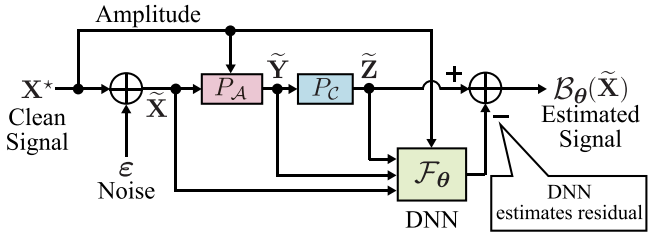


Fig. 5. Diagram of the proposed SBTD.

The proposed training strategy, which we call SBTD, is illustrated in Fig. 5. With the complex Gaussian noise $\boldsymbol{\varepsilon}$, SBTD is defined as the following minimization problem:

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{X}^\star, \widetilde{\mathbf{X}}}[\mathcal{L}(\widetilde{\mathbf{Z}} - \mathbf{X}^\star, \mathcal{F}_{\boldsymbol{\theta}}(\widetilde{\mathbf{X}}, \widetilde{\mathbf{Y}}, \widetilde{\mathbf{Z}}))], \tag{23}$$

where $\widetilde{\mathbf{X}} = \mathbf{X}^\star + \boldsymbol{\varepsilon}$ denotes the complex STFT coefficients contaminated by the complex Gaussian noise, $\widetilde{\mathbf{Y}} = P_{\mathcal{A}}(\widetilde{\mathbf{X}})$, and $\widetilde{\mathbf{Z}} = P_{\mathcal{C}}(\widetilde{\mathbf{Y}})$. It expects the DeGLI sub-block $\mathcal{B}_{\boldsymbol{\theta}}$ to map the noisy STFT coefficients $\widetilde{\mathbf{X}}$ to the clean one $\mathbf{X}^\star$ as

$$\mathbf{X}^\star \approx \mathcal{B}_{\boldsymbol{\theta}}(\widetilde{\mathbf{X}}) \qquad (= \widetilde{\mathbf{Z}} - \mathcal{F}_{\boldsymbol{\theta}}(\widetilde{\mathbf{X}}, \widetilde{\mathbf{Y}}, \widetilde{\mathbf{Z}})\,), \tag{24}$$

or equivalently $\mathbf{X}^\star - \widetilde{\mathbf{Z}} \approx -\mathcal{F}_{\boldsymbol{\theta}}(\widetilde{\mathbf{X}}, \widetilde{\mathbf{Y}}, \widetilde{\mathbf{Z}})$ which leads to the objective function in (23). Since the backpropagation does not pass through $P_{\mathcal{A}}$, SBTD can avoid the instability caused by it. In addition, its computational cost and required memory are significantly reduced compared to the end-to-end method in (20) which repeatedly applies the same DNN. The training data can be easily synthesized by adding the Gaussian noise, which is also an advantage of the proposed strategy.

Although SBTD is defined through the denoising task of complex STFT coefficients, it is actually related to phase reconstruction. Magnitude of the inputted noisy coefficients is always replaced by the true ones by $P_{\mathcal{A}}$. Thus, the latter part of the sub-block, which includes the DNN, mainly focuses on the reduction of phase disturbance. This is not a usual denoising task but rather a phase reconstruction task. Appropriateness of this training strategy will be confirmed in Section V-F.

C. Proposed Complex DNN

While any DNN can be incorporated into the DeGLI framework, this subsection presents a DNN architecture suitable for DeGLI named the *amplitude-informed gated complex convolutional neural network (AI-GCNN)*. AI-GCNN consists of multiple AI-GC layers: complex convolution layers with a gating mechanism that considers the given amplitude.

A 2D complex convolution is formulated by [61]

$$\begin{aligned}\mathrm{Conv}_{\mathbf{W}_{\mathbb{C}}}(\mathbf{C}) = (\mathbf{W}_{\mathrm{Re}} * \mathbf{C}_{\mathrm{Re}} - \mathbf{W}_{\mathrm{Im}} * \mathbf{C}_{\mathrm{Im}}) \\ + i(\mathbf{W}_{\mathrm{Re}} * \mathbf{C}_{\mathrm{Im}} + \mathbf{W}_{\mathrm{Im}} * \mathbf{C}_{\mathrm{Re}}),\end{aligned} \tag{25}$$

where $*$ denotes 2D real convolution, $\mathbf{C} = \mathbf{C}_{\mathrm{Re}} + i\mathbf{C}_{\mathrm{Im}}$ is a complex input variable, $\mathbf{C}_{\mathrm{Re}}$ and $\mathbf{C}_{\mathrm{Im}}$ are its real and imaginary parts, respectively. Here, $\mathbf{W}_{\mathbb{C}} = \mathbf{W}_{\mathrm{Re}} + i\mathbf{W}_{\mathrm{Im}}$ is a complex

convolution filter to be trained, and $\mathbf{W}_{\mathrm{Re}}$ and $\mathbf{W}_{\mathrm{Im}}$ are its real and imaginary parts, respectively. The complex convolution layer is chosen for capturing the relation between adjacent T-F bins efficiently. Furthermore, the projection regarding spectrogram consistency $P_{\mathcal{C}}$ can be written as an operation similar to the complex convolution[5] [39]. Thus, a complex convolution layer can be regarded as a generalization of the projection of spectrogram consistency.

For the nonlinearity in AI-GCNN, we consider a gating mechanism [43] motivated by the following two reasons: requirement of a multiplicative structure and T-F masking. Firstly, data-dependent scalar multiplication is desired. When the output of the sub-block is ideal at the mth iteration, i.e., $\mathbf{X}^{[m]} = \mathbf{X}^{\star}$, the intermediate variable $\mathbf{Z}^{[m]}$ is also $\mathbf{X}^{\star}$, and thus the output of the DNN should be zero. This requirement can be easily fulfilled by applying zero multiplication as the gating mechanism. Secondly, the gating mechanism is motivated by T-F masking. As DeGLI subtracts the DNN output from $\mathbf{Z} = P_{\mathcal{C}}(P_{\mathcal{A}}(\mathbf{X}))$, the DNN should extract the unsuitable components contained in $\mathbf{Z}$. A usual choice for extracting some components in a spectrogram is T-F masking [62], which can be interpreted as a gating mechanism.

A T-F mask is often estimated from STFT magnitude. Hence, we calculate the gate as

$$\mathrm{AmpGate}_{\mathbf{W}_{\mathbb{R}}}(\mathbf{C}) = \mathrm{Sigmoid}(|\mathbf{C}| * \mathbf{W}_{\mathbb{R}}), \tag{26}$$

where $\mathrm{Sigmoid}(\cdot)$ is the sigmoid function, $\mathbf{W}_{\mathbb{R}}$ is a real convolution filter, and $|\mathbf{C}|$ denotes magnitude of the complex numbers calculated element-wise. We define the *amplitude-based gated complex convolution* (AGC) layer as

$$\mathrm{AGC}_{\mathbf{W}_{\mathbb{C}},\mathbf{W}_{\mathbb{R}}}(\mathbf{C}) = \mathrm{Conv}_{\mathbf{W}_{\mathbb{C}}}(\mathbf{C}) \odot \mathrm{AmpGate}_{\mathbf{W}_{\mathbb{R}}}(\mathbf{C}). \tag{27}$$

Finally, as the given amplitude $\mathbf{A}$ is important for phase reconstruction, AI-GCNN employs an amplitude-informed extension of AGC layer, AI-GC layer. It is defined as

$$\text{AI-GC}_{\mathbf{W}_{\mathbb{C}},\mathbf{W}_{\mathbb{R}}}(\mathbf{C}) = \mathrm{Conv}_{\mathbf{W}_{\mathbb{C}}}(\mathbf{C}) \odot \mathrm{AmpGate}_{\mathbf{W}_{\mathbb{R}}}(\check{\mathbf{C}}), \tag{28}$$

$$\check{\mathbf{C}} = [\mathbf{A}, \mathbf{C}], \tag{29}$$

where $[\cdot,\cdot]$ denotes concatenation in the channel direction. AI-GCNN should be possible to estimate the undesired residual component by comparing the given amplitude $\mathbf{A}$ and the magnitude of the complex features.

Before leaving this subsection, we briefly mention an existing complex layer for comparison. Recently, complex DNNs have gained much attention in complex T-F mask estimation for speech enhancement/separation [63], [64]. These studies utilized complex extensions of nonlinear functions related to the rectified linear unit (ReLU), and [63] reported that the following $\mathbb{C}$PReLU achieved promising results:

$$\mathbb{C}\mathrm{PReLU}(\mathbf{C}) = \mathrm{PReLU}_{\alpha_{\mathrm{Re}}}(\mathbf{C}_{\mathrm{Re}}) + i\mathrm{PReLU}_{\alpha_{\mathrm{Im}}}(\mathbf{C}_{\mathrm{Im}}), \tag{30}$$

where $\alpha_{\mathrm{Re}} > 0$ and $\alpha_{\mathrm{Im}} > 0$ are the parameters of parametric ReLU (PReLU). This $\mathbb{C}$PReLU is not isotropic with respect to the origin, which induces bias to the complex output. This bias directly affects the estimated residual component in DeGLI. In contrast, the gating mechanism given in (28) preserves the phase because the sigmoid function takes a real number within 0 to 1. Thus, the proposed nonlinear function does not induce such bias to the estimated residual.

[5]Specifically, the projection regarding the spectrogram consistency is given by the twisted convolution with a single filter [39]. The twisted convolution requires frequency-dependent phase modification in addition to the regular convolution.

D. Relation Between DeGLI and Existing Methods

In this subsection, the relation between DeGLI and existing iterative-algorithm-based DNNs is discussed. The main difference between DeGLI and deep unfolding [46] is whether it is iterative or not. Deep unfolding constructs a single large DNN based on an unrolled iterative algorithm. Each iteration in the algorithm is interpreted as a layer of the DNN, where each layer has its own trainable parameters to be optimized. Therefore, deep unfolding must fix the number of layers before training, and the entire DNN is applied only once at the time of inference. In contrast, DeGLI iterates a single DNN, which can be seen as deep unfolding whose parameters are shared in all layers. Recently, the effectiveness of weight-shared DNNs has been confirmed in natural language processing, and they outperformed weight-unshared DNNs in some sense [65], [66]. DeGLI does not fix the number of iterations in advance, and it can be decided at the time of inference. In addition, the computational cost for the training of DeGLI is less than that for deep unfolding thanks to the proposed SBTD.

The main difference between DeGLI and PnP [54] is whether the training is task-oriented or not. PnP incorporates the trained DNN into a proximal algorithm as a proximity operator which can be interpreted as a Gaussian denoiser. This leads to the idea of training the DNN to eliminate the Gaussian noise in the inputted degraded signal. This training strategy is totally independent of the original task. In contrast, SBTD contains the projections specific to phase reconstruction, and thus the training strategy of the DNN is related to phase reconstruction. In the sub-block, the first projection $P_{\mathcal{A}}$ perfectly removes the noise in terms of amplitude, and the second one $P_{\mathcal{C}}$ eliminates the inconsistent component contained in the result of $P_{\mathcal{A}}$. Because of these projections, the proposed training strategy is different from solving a general Gaussian denoising problem for the DNN.

Note that, while DeGLI is related to methods that combine an iterative algorithm and DNN, it is not directly based on a specific algorithm already proposed for phase reconstruction. In the existing methods, the usual strategy is to replace a predefined operation ($P_{\mathcal{A}}$ or $P_{\mathcal{C}}$ in our case) by a DNN to exploit the learned knowledge (see Figs. 2 and 3). In contrast, DeGLI leaves the predefined operations ($P_{\mathcal{A}}$ and $P_{\mathcal{C}}$) and additionally applies a DNN to their results. This idea is based on the acceleration techniques as in (19) which does not modify the operations of GLA.

V. EXPERIMENTS

In this section, we conducted several experiments to validate the effectiveness of the proposed DeGLI framework. The

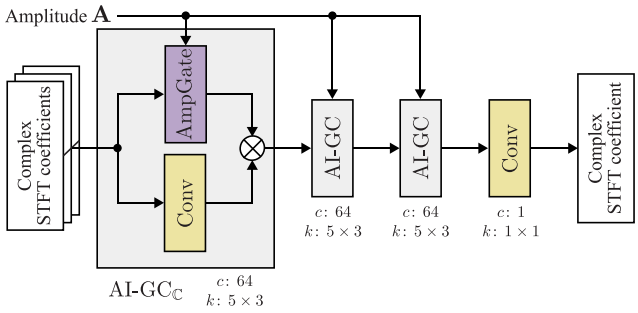


Fig. 6. Illustration of AI-GCNN utilized in the experiments. It maps a concatenation of three tuples of complex STFT coefficients $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ to the undesired component in $\mathbf{Z}$. Here, "AI-GC" indicates the AI-GC layer given in (28) with zero padding to maintain its input size, where k and c are the kernel size and number of channels, respectively. "Conv" represents the complex convolution layer (without bias) in (25). Stride sizes for all convolution layers were set to 1×1.

TABLE I
EXPERIMENTAL CONDITIONS

Parameters for Signal Processing	
Sampling rate	22050 Hz
Window type	Hann window
Window length	46.4 ms (1024 samples)
Shift length	11.6 ms (256 samples)
Parameters of DNN Architecture	
# of AI-GC layers	3
# of complex convolution layers	1
# of channels	64
Kernel size of AI-GC layers	5×3
Kernel size of the last convolution layer	1×1
Stride for convolution layers	1×1
# of parameters	380k
Parameters for Training	
Initial step size	0.0004
Batch size	32
# of epochs	300

experimental conditions are described in Section V-A. In Section V-B, we investigate the relation between the number of sub-blocks and the quality of the reconstructed signals. DeGLI is compared with GLA and an existing DNN-based phase reconstruction method. Section V-C compares DeGLI with two neural vocoders in both objective and subjective measures. The generalization ability of DeGLI is shown in Section V-D. Finally, three proposed techniques (the sub-block, SBTD, and AI-GCNN) are validated separately in Sections V-E, V-F, and V-G, respectively.

A. Experimental Condition

1) Dataset, Initialization, and STFT Parameters: We utilized the LJ speech dataset which consists of 13100 English utterances of a single speaker. Audio clips were separated into three subsets: 12500 clips for the training, 300 clips for the validation, and 300 clips for the testing as in [67]. During the training, the utterances were divided into about 1-second-long segments (24064 samples), where the sampling rate was 22050 Hz. As the noise for SBTD, the complex Gaussian noise was added in T-F domain so that the signal-to-noise ratio (SNR) was randomly selected from -6 to 12 dB. At the time of inference, for DeGLI and GLA, the magnitude of $\mathbf{X}^{[0]}$ was set to the given amplitude $\mathbf{A}$ that was calculated from the corresponding time-domain signal. The phase was initialized by the following two methods: sampling from the uniform distribution within $\pm\pi$ or setting to zero. Note that, in the latter case, $\mathbf{X}^{[0]}$ coincided with $\mathbf{A}$. STFT was implemented with the Hann window, whose length was 46.4 ms (1024 samples), with 11.6 ms (256 samples) shifting.

2) DNN Architecture and Training Setup: The architecture of AI-GCNN utilized in the following experiments is depicted in Fig. 6. It consists of 3 AI-GC layers given in (28), and one complex convolution layer follows them. Note that all complex convolutional layers did not contain bias. This DNN estimated a residual component from three tuples of complex STFT coefficients which were concatenated to the channel directions. For SBTD given in (23), the loss function $\mathcal{L}$ was the squared Frobenius norm of the difference. In all training, parameters were optimized by the Adam optimizer [68] with a batch size 32 for 300 epoch, where the step size was decayed by multiplying 0.5 every 100 epoch from 0.0004. Since we observed that the training and validation losses exhibited almost the same behavior in our early experiments, we only utilized the validation dataset to monitor the behavior of training in the experiments. All the above-mentioned conditions are summarized in Table I.

3) Evaluation Metrics: The phase reconstruction performance of DeGLI was compared with those of existing methods by three objective measurements. We evaluated the quality of the reconstructed signal by the wide-band extension of the perceptual evaluation of subjective quality (PESQ) [69]. Note that original PESQ was utilized for evaluation of narrow-band speech in telecommunications. The short-time objective intelligibility (STOI) was also utilized to evaluate the speech intelligibility [70]. These objective measures have been commonly utilized to evaluate the sound quality in speech enhancement [62]. As in ordinary studies of phase reconstruction, we also evaluated the log-spectral convergence (LSC) [71]:

$$\mathrm{LSC}(\hat{\mathbf{X}}, \mathbf{A}) = 20 \log_{10} \frac{\| \mathbf{A} - |P_{\mathcal{C}}(\hat{\mathbf{X}})| \|_{\mathrm{Fro}}}{\| \mathbf{A} \|_{\mathrm{Fro}}}. \tag{31}$$

B. Relation Between Performance and Number of Sub-Blocks

To validate the concept of DeGLI that can balance the performance and computational cost, we first investigated the performance per number of sub-blocks M. The objective measures of signals obtained by DeGLI, GLA, and GLA initialized by a DNN-based phase reconstruction method called recurrent phase unwrapping (RPU) [32] are shown in Fig. 7. Although the DNN was not trained in a end-to-end manner, the objective measures of DeGLI were monotonically improved by increasing the number of sub-blocks M. Since M is adjustable at the time of inference, this result indicates that DeGLI can easily trade the quality of reconstructed signals and the computational cost linearly depending on M. Namely, one can eliminate unnecessary computation or improve the sound quality as long as the computational resource allows. We stress that this favorable feature of DeGLI cannot be achieved by existing DNN-based phase reconstruction, including RPU. As the result of using DNN inside GLA,

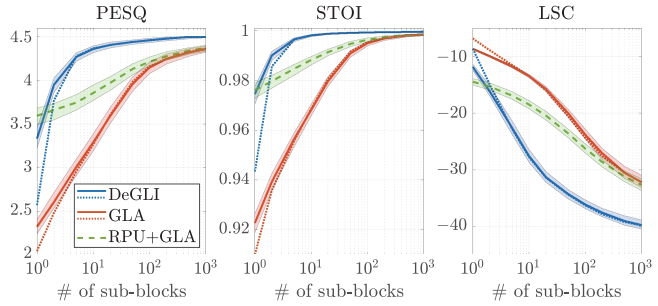


Fig. 7. Relation between the objective measures (PESQ, STOI, and LSC) and the number of sub-blocks M. Higher PESQ and STOI indicate better sound quality. Lower LSC means better phase reconstruction. Lines represent medians, and colored regions indicate the first and third quartiles. The types of lines represent different initialization of phase: solid lines represent uniformly random initialization, and dotted lines represent zero initialization. The quartiles for the zero initialization are omitted for visibility because their appearance was essentially the same with those of random initialization.

DeGLI outperformed the original GLA in terms of all objective measures for both initialization methods of phase. The uniformly random initialization achieved better objective measures than zero initialization when the number of sub-blocks was small, and thus we utilized the random initialization method in the following experiments. It also outperformed RPU followed by GLA. These results indicate that DeGLI is not only efficient but also well-performing. Note that DeGLI significantly improved the sound quality in the first 10 sub-blocks. Perceptually, we also confirmed that the reconstructed signals had reasonable quality at $M = 10$.

Since DeGLI applies the DNN in addition to the projections of GLA, its computational time per iteration is longer than that of GLA. As a reference, DeGLI with 10 sub-blocks took about 0.172 s on a GPU (NVIDIA TITAN V) and 3.089 s on a CPU (Intel Core i9-7980XE, 2.60 GHz) for a 2-second-long signal. GLA with 200 iterations took about 0.297 s on the GPU and 5.97 s on the CPU. We stress that DeGLI with 10 sub-blocks outperformed GLA with 200 iterations in terms of the objective measures as observed in Fig. 7.

C. Comparison to Neural Vocoders

In speech synthesis, it has been widely demonstrated that neural vocoders can generate natural speech signals and outperform GLA in terms of perceptual quality [67], [72], [73]. Since the experimental results in Section V-B showed the reconstruction quality only in terms of objective measures, DeGLI was compared in terms of naturalness in this subsection. In this experiment, we compared DeGLI with two neural vocoders: a popular open source WaveNet (WN) [72], [74] based on a mixture of logistics distribution,[6] and the official WaveGlow (WG) [67]. The audio samples of them were brought from a public folder.[7] Note that GLA and DeGLI reconstruct a time-domain signal from a linear spectrogram while the neural vocoders reconstruct that from a mel-spectrogram. Reconstructed signals were evaluated by PESQ, STOI, and a subjective test like MUSHRA (multiple stimuli with hidden reference and anchor) with 16 participants. In the subjective test, GLA with 60 iterations was used as the anchor. The number of the DeGLI sub-blocks was also set to 60. Each participant evaluated 10 sets that were randomly selected from 40 prepared utterances.[8] Each evaluation set consisted of the original signal, as the hidden reference, and 4 reconstructed signals by GLA, WN, WG, and DeGLI. The participants were asked to rate them in terms of naturalness from 0 to 100. To assess naturalness relatively and clarify the difference, we additionally instructed the participants to rate one of them as 0 (worst among the 5 signals) and another one as 100 (expected to be the hidden reference) for each set.

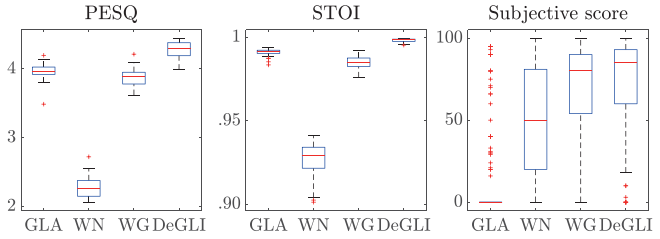


Fig. 8. PESQ, STOI, and the result of subjective test for signals reconstructed by GLA [20], Yamamoto's open-source WN [72], [74], official WG [67], and the proposed DeGLI with 60 sub-blocks. Red lines represent medians, and boxes indicate the first and third quartiles.

PESQ, STOI, and the result of the subjective test are illustrated in Fig. 8. GLA and DeGLI resulted in higher PESQ and STOI compared to the neural vocoders. This should be because the neural vocoders generated signals whose STFT magnitudes were different from the given ones while GLA and DeGLI maintained them by $P_{\mathcal{A}}$. In the subjective test, the official WG outperformed both GLA and the Yamamoto's open-source WN, which is consistent with the results reported in previous studies [67], [73]. Compared to the official WG, DeGLI further improved the subjective scores. One of the advantages of DeGLI is the smallness of the number of parameters to be stored, which is the result of the parameter sharing and GLA-based fixed layers. The number of parameters of DeGLI was less than 0.5% of that of WG.

D. Robustness and Generalization Capability of DeGLI

1) Evaluation using Degraded Spectrograms: In some applications, including speech synthesis, the given amplitude $\mathbf{A}$ may contain estimation errors. To validate the robustness of DeGLI to such errors, in this experiment, we evaluated DeGLI on degraded spectrograms. For degradation of spectrograms, we utilized mel-frequency compression because a mel-spectrogram is a popular acoustic feature in speech synthesis. The degraded spectrogram was calculated by the following procedure. First, a linear spectrogram calculated from a time-domain signal was converted to a mel-spectrogram. Second, the mel-spectrogram

[6]The audio samples of WN were generated by using Yamamoto's open source implementation: https://doi.org/10.5281/zenodo.1472609. This code has been utilized as a reference in several papers [67], [75], [76].

[7]The audio samples were brought from Google Drive: https://drive.google.com/open?id=1SKO4oDmqUy-AEFXCSAMHyhe_iUIG9FKq

[8]The audio samples utilized in the test are available in our project page: https://sites.google.com/view/yoshiki-masuyama/degli.

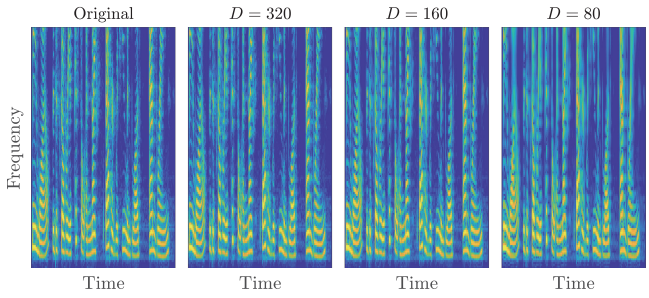


Fig. 9. Example of the original linear spectrogram and degraded ones with different levels of mel-frequency compression.

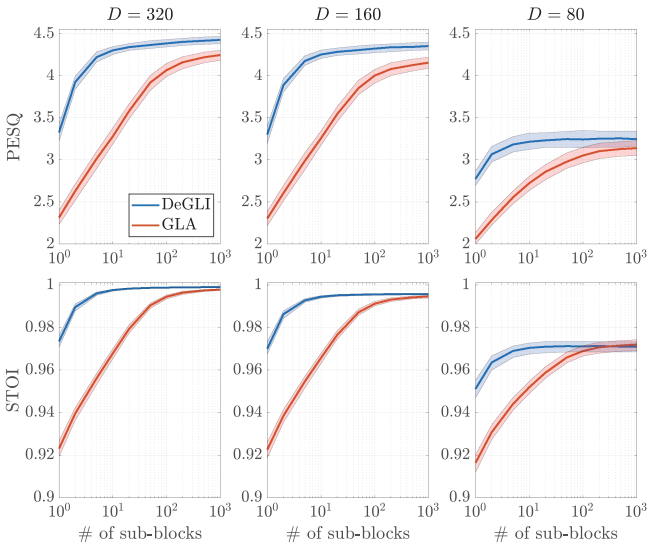


Fig. 10. PESQ and STOI of speech signals reconstructed from the degraded spectrograms. Solid lines represent medians, and colored regions indicate the first and third quartiles.

was converted back to the linear scale by the pseudo inverse of the forward transform. Finally, the negative elements of the reconstructed spectrogram were replaced by zero. To change the level of degradation, the dimension of mel-spectrograms D was chosen from $\{80, 160, 320\}$. Examples of the linear spectrograms are shown in Fig. 9. Note that the DNN is trained only with clean linear spectrograms as described in Section V-A.

PESQ and STOI of reconstructed signals are shown in Fig. 10. In all cases, the final performance of DeGLI became worse than that using the original linear spectrograms. Even so, DeGLI still outperformed GLA when $D \in \{160, 320\}$ and was comparable to GLA when $D = 80$. Note that the monotonic nature of improvement brought by DeGLI was essentially preserved, which indicates the robustness of DeGLI to the error in the given amplitude. SBTD using degraded spectrograms may improve the performance of DeGLI in such a case, but an appropriate way of degradation should depend on the application.

2) Evaluation using Different Speakers: To investigate the generalization capability in terms of speakers, DeGLI trained on the LJ speech dataset was evaluated by using the VCTK corpus. The training dataset, the LJ speech dataset, contains utterances of a single female speaker. For the evaluation, we utilized 600

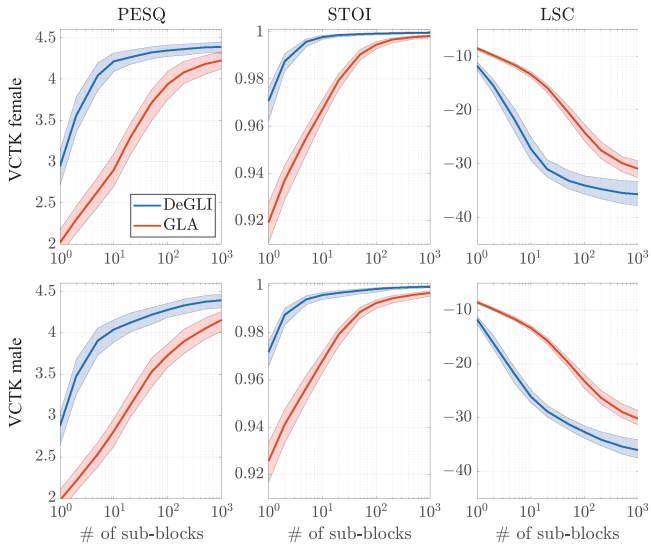


Fig. 11. PESQ and STOI of reconstructed signals in the VCTK corpus, where DeGLI was trained by the LJ speech dataset. Solid lines represent medians over 300 utterances, and colored regions indicate the first and third quartiles.

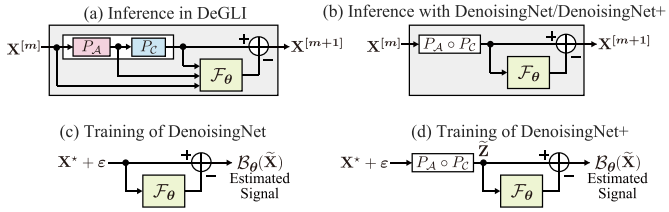


Fig. 12. (a) Sub-block of DeGLI. (b) Sub-block for the inference using DenoisingNet/DenoisingNet+. (c) Diagram of training of DenoisingNet. (d) Diagram of training of DenoisingNet+.

utterances from the VCTK corpus where each 100 utterances were selected from those of three females (`p225`, `p228`, `p229`) and three males (`p226`, `p227`, `p232`). Those utterances were resampled at 22050 Hz.

The experimental results are illustrated in Fig. 11. DeGLI outperformed GLA even for the utterances of speakers not contained in the training dataset. This result suggests the generalization capability of DeGLI to multiple speakers. Comparing Fig. 11 to Fig. 7, the final objective measures for the VCTK corpus were slightly worse than those for the LJ speech dataset due to the mismatch between training and testing datasets. Recent speaker adaptation techniques [77]–[79] may reduce this performance deterioration, which is a possible direction of future works.

E. Effectiveness of Auxiliary Inputs $\mathbf{X}^{[m]}$ *and* $\mathbf{Y}^{[m]}$

In DeGLI, the DNN estimates the residual component in $\mathbf{Z}^{[m]}$ from three tuples of complex STFT coefficients, $\mathbf{X}^{[m]}$, $\mathbf{Y}^{[m]}$, and $\mathbf{Z}^{[m]}$. To clarify the effectiveness of the auxiliary inputs, $\mathbf{X}^{[m]}$ and $\mathbf{Y}^{[m]}$, we compared DeGLI in Fig. 12-(a) with its variant illustrated in Fig. 12-(b), where the DNN only takes $\mathbf{Z}^{[m]}$ as input. We considered two DNNs trained by different strategies for comparison. DenoisingNet was trained with the general Gaussian denoising task as in Fig. 12-(c), while DenoisingNet+

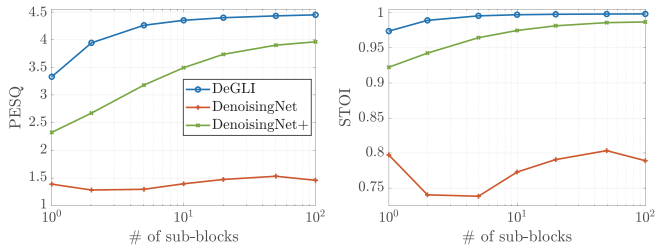


Fig. 13. Medians of PESQ and STOI of reconstructed speech signals for comparing different sub-block architectures.

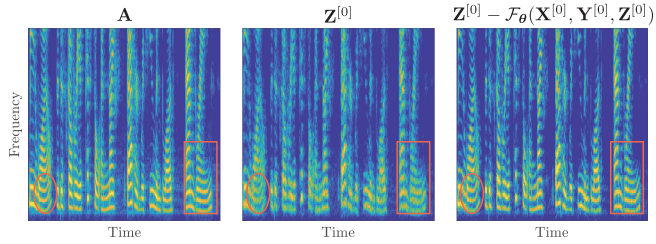


Fig. 14. Example of the given amplitude $\mathbf{A}$, result of GLA $\mathbf{Z}^{[0]}$, and result of refinement by DNN $\mathbf{Z}^{[0]} - \mathcal{F}(\mathbf{X}^{[0]}, \mathbf{Y}^{[0]}, \mathbf{Z}^{[0]})$ (from left to right). In the circle and box, corrupted harmonic components were recovered.

TABLE II
LSCs OF $\mathbf{Z}^{[m]}$ AND $\mathbf{Z}^{[m]} - \mathcal{F}(\mathbf{X}^{[m]}, \mathbf{Y}^{[m]}, \mathbf{Z}^{[m]})$.

m	0	1	2
$\mathbf{Z}^{[m]}$	-3.79	-12.24	-17.64
$\mathbf{Z}^{[m]} - \mathcal{F}(\mathbf{X}^{[m]}, \mathbf{Y}^{[m]}, \mathbf{Z}^{[m]})$	$\mathbf{-8.29}$	$\mathbf{-14.37}$	$\mathbf{-18.44}$

utilized the projections specific to phase reconstruction as in Fig. 12-(d). In this experiment, their architecture was similar to that of AI-GCNN, but AI-GC layers were replaced by AGC layers because the clean STFT magnitude is usually unknown in the general Gaussian denoising task. To set the number of parameters roughly the same, they contained an additional AGC layer before the DNN. The results are summarized in Fig. 13, where DeGLI outperformed the other two. This result indicates that auxiliary variables $\mathbf{X}^{[m]}$ and $\mathbf{Y}^{[m]}$ are informative to estimate the residual component. In addition, it can be seen that the projections in the training stage is indispensable because DenoisingNet, which did not utilize the projections in its training, failed to improve the performance by increasing the number of sub-blocks M.

To illustrate the effectiveness of the residual estimation by the DNN, an example of $\mathbf{A}$, $\mathbf{Z}^{[0]}$, and $\mathbf{Z}^{[0]} - \mathcal{F}(\mathbf{X}^{[0]}, \mathbf{Y}^{[0]}, \mathbf{Z}^{[0]})$ are shown in Fig. 14. It can be seen that the corrupted harmonic components were enhanced by subtracting the residual component estimated by the DNN, e.g., in the orange circle and box. This favorable result was also confirmed in terms of LSCs. LSCs of $\mathbf{Z}^{[m]}$ and $\mathbf{Z}^{[m]} - \mathcal{F}(\mathbf{X}^{[m]}, \mathbf{Y}^{[m]}, \mathbf{Z}^{[m]})$ of the signal corresponding to the spectrogram shown in Fig. 14 for $m = 0, 1, 2$ are listed in Table II. The DNN improved LSCs from the result of GLA $\mathbf{Z}^{[m]}$. These results indicate the effectiveness of the residual estimation by the DNN.

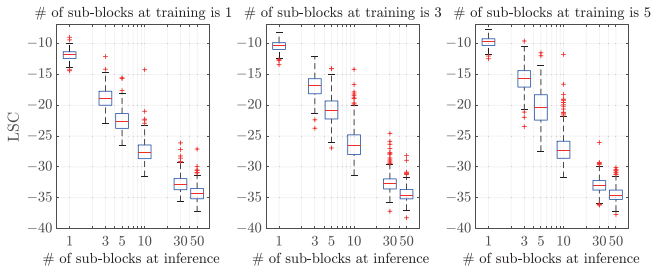


Fig. 15. Comparison of LSCs among training strategies whose number of sub-blocks at the time of training is different. The proposed SBTD, which uses one sub-block in training, is illustrated in the leftmost figure.

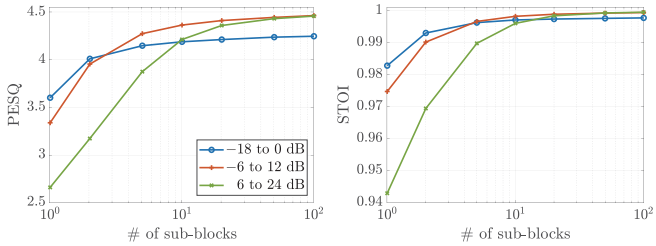


Fig. 16. Medians of PESQ and STOI of reconstructed speech signals for comparing SBTDs with different SNR ranges.

F. Detailed Experiments of SBTD

1) Efficiency of SBTD: To illustrate the sufficiency of the single-block training, SBTD (using 1 sub-block) was compared with its variants using 3 and 5 sub-blocks. In those variants, the model parameters were shared in all sub-blocks to align the number of parameters. In the training with multiple sub-blocks, the complex Gaussian noise was added to the clean STFT coefficients before the first sub-block. Then, the DNN was trained to minimize the difference between the clean STFT coefficients and the output of the last sub-block.

LSCs of the reconstructed signals are shown in Fig. 15. DeGLI trained by SBTD achieved performance comparable to those trained with 3 and 5 sub-blocks. As the computational cost of the training increases as M increases, that of SBTD is the lowest among all variants using more than one sub-blocks. Hence, SBTD minimizes the computational cost for training while maintaining the performance.

2) Effect of Noise Level in SBTD: In the all experiments thus far, DeGLI was trained by SBTD whose SNR of inputted STFT coefficients was varied from -6 to 12 dB. In this experiment, we investigated how this range of SNR affects the performance. To this end, two DNNs were additionally trained by SBTD with a lower SNR range (from -18 to 6 dB) or a higher range (from 6 to 24 dB).

The results are illustrated in Fig. 16. First of all, with all SNR ranges, PESQ and STOI were improved as the number of sub-blocks increased. DeGLI trained with the lower SNR range improved the objective measures quickly but resulted in the lowest performance at the end. In contrast, DeGLI trained with the higher SNR range improved PESQ and STOI slowly but surpassed those with the lower SNR range. These results indicate that there exists a trade-off between the speed of improvements

TABLE III
MEDIAN OF PESQ AND STOI OF RECONSTRUCTED SIGNALS WITH 1, 2, 3, 5, 10, 100 SUB-BLOCKS FOR COMPARING VARIOUS DNN ARCHITECTURES. THE BOLD FONT AND UNDERLINE INDICATE HIGHEST AND SECOND HIGHEST SCORES.

	PESQ						STOI						# of params.
M	1	2	3	5	10	100	1	2	3	5	10	100	
rPReLUC	**3.47**	3.93	4.09	4.21	4.32	4.44	0.974	0.988	0.992	0.995	0.997	**0.999**	504k
rGC	2.99	3.71	4.01	4.22	4.33	4.43	0.962	0.985	0.992	0.996	**0.998**	**0.999**	1007k
$\mathbb{C}$PReLUC	3.40	3.89	4.03	4.17	4.26	4.40	0.972	0.987	0.991	0.995	0.997	**0.999**	252k
ReImGC	2.73	3.43	3.79	4.10	4.28	4.41	0.954	0.980	0.988	0.994	0.997	**0.999**	503k
AGC	3.23	3.87	4.08	4.25	**4.37**	**4.47**	0.971	0.989	0.993	0.996	**0.998**	**0.999**	378k
AI-GC	3.33	**3.95**	**4.14**	**4.27**	4.36	4.46	**0.975**	**0.990**	**0.994**	**0.997**	**0.998**	**0.999**	380k

and the final performance, and it can be adjusted by the range of SNR in SBTD.

G. Effectiveness of AI-GC Layer

DeGLI iteratively utilizes a single DNN that directly handle complex STFT coefficients. Such iterative use is unusual, and it is unclear what kind of DNN is suitable for that. Hence, we conducted an experiment for comparing DNN architectures in terms of the DeGLI framework. In addition to AI-GCNN, we considered three complex DNNs and two real DNNs. Two complex DNNs were constructed by replacing AI-GC layers with AGC layers or the following *real-imaginary-based gated complex convolution (ReImGC) layers*:

$$\mathrm{ReImGate}_{\mathbf{W}_{\mathbb{R}}}(\mathbf{C}) = \mathrm{Sigmoid}([\mathbf{C}_{\mathrm{Re}}, \mathbf{C}_{\mathrm{Im}}] * \mathbf{W}_{\mathrm{Re}}), \quad (32)$$

$$\mathrm{ReImGC}_{\mathbf{W}_{\mathbb{C}},\mathbf{W}_{\mathbb{R}}}(\mathbf{C}) = \mathrm{Conv}_{\mathbf{W}_{\mathbb{C}}}(\mathbf{C}) \odot \mathrm{ReImGate}_{\mathbf{W}_{\mathbb{R}}}(\mathbf{C}). \quad (33)$$

Another complex DNN was obtained by combining the complex convolution layers with $\mathbb{C}$PReLU given in (30) (abbreviated as $\mathbb{C}$PReLUC). We also considered real gated convolution layer (abbreviated as rGC) as in our conference paper [31] and real convolution layer with PReLU (abbreviated as rPReLUC) as in [30]. The number of channels for real convolution layers was set to 128.

The results are summarized in Table III. First of all, with all DNNs, the performance of DeGLI improved as the number of sub-blocks M increased. This indicates that the DeGLI framework can cooperate with various types of DNNs. When $M = 1$, DeGLI with the rPReLUC layer achieved the best PESQ. Note that the rPReLUC layer was utilized in a DNN-based phase reconstruction method that directly generates complex STFT coefficients [30]. Therefore, our experimental result confirmed the validity of the previous study. When $M > 1$, AI-GC outperformed rPReLUC for all M. This result indicates that the AI-GC layer is suitable for the iterative use in DeGLI, and its benefit is provided by stacking at least more than one sub-block.

Comparing AGC and ReImGC layers, the gating mechanism based on the magnitude of inputted variables $|\mathbf{C}|$ achieved better performance than that based on the real and imaginary parts of inputs $[\mathbf{C}_{\mathrm{Re}}, \mathbf{C}_{\mathrm{Im}}]$. Furthermore, AGC outperformed rGC, which was utilized in our conference paper [31], with much small number of parameters. This result also indicates the effectiveness of the combination of complex convolution and the amplitude-based gating mechanism. By additionally utilizing the given amplitude, AI-GC outperformed AGC when $M \in \{1, 2, 3, 5\}$, but their difference became smaller as the number of sub-blocks M increased. This might be because the first AGC layer can also consider the target amplitude $\mathbf{A}$ by extracting the amplitude of $\mathbf{Y} = P_{\mathcal{A}}(\mathbf{X})$.

VI. CONCLUSION

We have presented the DNN-based phase reconstruction framework, called DeGLI, which stacks the common GLA-based sub-block containing a DNN. The DNN aims to remove the undesired components from the result of projections of GLA. An advantage of DeGLI is that its computational cost is adjustable at the time of inference by changing the number of sub-blocks. This allows us to use a single model for applications on various devices whose allowable computational cost is different. We further proposed the effective training strategy, named SBTD, that minimizes the computational cost of the training while keeping the phase reconstruction performance. Our experimental results confirmed that DeGLI enables us to trade the quality of a reconstructed signal and computational cost. In addition, DeGLI with 60 sub-blocks resulted in better sound quality comparing to the neural vocoders. Furthermore, we presented a complex DNN for DeGLI, and its effectiveness was validated by comparing various real and complex DNNs.

REFERENCES

[1] S. Takaki, H. Kameoka, and J. Yamagishi, "Direct modeling of frequency spectra and waveform generation based on phase recovery for DNN-based speech synthesis," in *Proc. INTERSPEECH* Aug. 2017, pp. 1128–1132.

[2] T. Kaneko, S. Takaki, H. Kameoka, and J. Yamagishi, "Generative adversarial network-based postfilter for STFT spectrograms," in *Proc. INTERSPEECH*, Aug. 2017, pp. 3389–3393.

[3] Y. Saito, S. Takamichi, and H. Saruwatari, "Text-to-speech synthesis using STFT spectra based on low-/multi-resolution generative adversarial networks," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Apr. 2018, pp. 5299–5303.

[4] Y. Wang *et al.*, "Tacotron: Towards end-to-end speech synthesis," in *Proc. INTERSPEECH*, Aug. 2017, pp. 4006–4010.

[5] A. Marafioti, N. Holighaus, N. Perraudin, and P. Majdak, "Adversarial generation of time-frequency features with application in audio synthesis," in *Proc. Int. Conf. Mach. Learn.*, vol. 97, Jun. 2019, pp. 4352–4362.

[6] H. Suda, D. Saito, and N. Minematsu, "Voice conversion without explicit separation of source and filter components based on non-negative matrix factorization," in *Proc. ISCA Speech Synth. Workshop*, Sep. 2019, pp. 69–74.

[7] M. Krawczyk and T. Gerkmann, "STFT phase reconstruction in voiced speech for an improved single-channel speech enhancement," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 22, no. 12, pp. 1931–1940, Dec. 2014.

[8] P. Mowlaee and J. Kulmer, "Harmonic phase estimation in single-channel speech enhancement using phase decomposition and SNR information," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 23, no. 9, pp. 1521–1532, Sep. 2015.

[9] P. M. R. Badeau and B. David, "Model-based STFT phase recovery for audio source separation," *IEEE/ACM Trans. Audio, Speech Lang. Process.*, vol. 26, no. 6, pp. 1095–1105, Jun. 2018.

[10] T. Afouras, J. S. Chung, and A. Zisserman, "The conversation: Deep audio-visual speech enhancement," in *Proc. INTERSPEECH*, Sep. 2018, pp. 3244–3248.

[11] Z.-Q. Wang, J. L. Roux, D. Wang, and J. R. Hershey, "End-to-end speech separation with unfolded iterative phase reconstruction," in *Proc. INTERSPEECH*, Sep. 2018, pp. 2708–2712.

[12] Z.-Q. Wang, K. Tan, and D. Wang, "Deep learning based phase reconstruction for speaker separation: A trigonometric perspective," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, May 2019, pp. 71–75.

[13] N. Takahashi, P. Agrawal, N. Goswami, and Y. Mitsufuji, "PhaseNet: Discretized phase modeling with deep neural networks for audio source separation," in *Proc. INTERSPEECH*, Sep. 2018, pp. 2713–2717.

[14] J. Le Roux, G. Wichern, S. Watanabe, A. Sarroff, and J. R. Hershey, "Phasebook and friends: Leveraging discrete representations for source separation," *IEEE J. Sel. Top. Signal Process.*, vol. 13, no. 2, pp. 370–382, May 2019.

[15] Y. Ephraim and D. Malah, "Speech enhancement using a minimum-mean square error short-time spectral amplitude estimator," *IEEE/ACM Trans. Acoust., Speech, Signal Process.*, vol. 32, no. 6, pp. 1109–1121, Dec. 1984.

[16] K. Paliwal, K. Wójcicki, and B. Shannon, "The importance of phase in speech enhancement," *Speech Commun.*, vol. 53, no. 4, pp. 465–494, Apr. 2011.

[17] T. Gerkmann, M. Krawczyk-Becker, and J. Le Roux, "Phase processing for single-channel speech enhancement: History and recent advances," *IEEE Signal Process. Mag.*, vol. 32, no. 2, pp. 55–66, Mar. 2015.

[18] P. Mowlaee, R. Saeidi, and Y. Stylianou, "Advances in phase-aware signal processing in speech communication," *Speech Commun.*, vol. 81, pp. 1–29, Jul. 2016.

[19] K. Yatabe, Y. Masuyama, T. Kusano, and Y. Oikawa, "Representation of complex spectrogram via phase conversion," *Acoust. Sci. Tech.*, vol. 40, no. 3, pp. 170–177, May 2019.

[20] D. Griffin and J. Lim, "Signal estimation from modified short-time Fourier transform," *IEEE Trans. Acoust. Speech Signal Process.*, vol. 32, no. 2, pp. 236–243, Apr. 1984.

[21] N. Perraudin, P. Balazs, and P. L. Søndergaard, "A fast Griffin–Lim algorithm," in *Proc. IEEE Workshop Appl. Signal Process. Audio Acoust.*, Oct. 2013, pp. 1–4.

[22] Y. Masuyama, K. Yatabe, and Y. Oikawa, "Griffin–Lim like phase recovery via alternating direction method of multipliers," *IEEE Signal Process. Lett.*, vol. 26, no. 1, pp. 184–188, Jan. 2019.

[23] G. T. Beauregard, M. Harish, and L. Wyse, "Single pass spectrogram inversion," in *Proc. IEEE Int. Conf. Digit. Signal Process.*, Jul. 2015, pp. 427–431.

[24] P. Magron, R. Badeau, and B. David, "Phase reconstruction of spectrograms with linear unwrapping: Application to audio signal restoration," in *Proc. Eur. Signal Process. Conf.*, Aug. 2015, pp. 1–5.

[25] D. S. Williamson, Y. Wang, and D. Wang, "Complex ratio masking for monaural speech separation," *IEEE/ACM Trans. Audio, Speech Lang. Process.*, vol. 24, no. 3, pp. 483–492, Mar. 2016.

[26] D. S. Williamson and D. Wang, "Time-frequency masking in the complex domain for speech dereverberation and denoising," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 25, no. 7, pp. 1492–1501, Jul. 2017.

[27] S. Takamichi, Y. Saito, N. Takamune, D. Kitamura, and H. Saruwatari, "Phase reconstruction from amplitude spectrograms based on von–Mises-distribution deep neural network," in *Proc. Int. Workshop Acoust. Signal Enhance.*, Sep. 2018, pp. 286–290.

[28] S. Takamichi, Y. Saito, N. Takamune, D. Kitamura, and H. Saruwatari, "Phase reconstruction from amplitude spectrograms based on directional-statistics deep neural networks," *Signal Process.*, vol. 169, pp. 107368, Apr. 2020.

[29] A. A. Nugraha, K. Sekiguchi, and K. Yoshii, "A deep generative model of speech complex spectrograms," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, May 2019, pp. 905–909.

[30] K. Oyamada, H. Kameoka, T. Kaneko, K. Tanaka, N. Hojo, and H. Ando, "Generative adversarial network-based approach to signal reconstruction from magnitude spectrograms," in *Proc. Eur. Signal Process. Conf.*, Sep. 2018, pp. 2514–2518.

[31] Y. Masuyama, K. Yatabe, Y. Koizumi, Y. Oikawa, and N. Harada, "Deep Griffin–Lim iteration," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, May 2019, pp. 61–65.

[32] Y. Masuyama, K. Yatabe, Y. Koizumi, Y. Oikawa, and N. Harada, "Phase reconstruction based on recurrent phase unwrapping with deep neural networks," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, May 2020, pp. 826–830.

[33] I. Waldspurger, A. d'Aspremont, and S. Mallat, "Phase recovery, maxcut and complex semidefinite programming," *Math. Programm.*, vol. 149, no. 1, pp. 47–81, Feb. 2015.

[34] E. J. Candes, Y. Eldar, and T. S. V. Voroninski, "Phase retrieval via matrix completion," *SIAM Rev.*, vol. 57, no. 2, pp. 225–251, May 2015.

[35] E. J. Candès, T. Strohmer, and V. Voroninski, "Phaselift: Exact and stable signal recovery from magnitude measurements via convex programming," *Commun. Pure Appl. Math.*, vol. 66, no. 8, pp. 1241–1274, Nov. 2013.

[36] Z. Průša, P. Balazs, and P. L. Søndergaard, "A noniterative method for reconstruction of phase from STFT magnitude," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 25, no. 5, pp. 1154–1164, May 2017.

[37] K. Yatabe, Y. Masuyama, and Y. Oikawa, "Rectified linear unit can assist Griffin–Lim phase recovery," in *Proc. Int. Workshop Acoust. Signal Enhance.*, Sep. 2018, pp. 555–559.

[38] T. Bendory, Y. C. Eldar, and N. Boumal, "Non-convex phase retrieval from STFT measurements," *IEEE Trans. Inf. Theory*, vol. 64, no. 1, pp. 467–484, Aug. 2018.

[39] J. Le Roux, H. Kameoka, N. Ono, and S. Sagayama, "Fast signal reconstruction from magnitude STFT spectrogram based on spectrogram consistency," in *Proc. Int. Conf. Digit. Audio Effects*, Sep. 2010, pp. 397–403.

[40] J. Le Roux and E. Vincent, "Consistent Wiener filtering for audio source separation," *IEEE Signal Process. Lett.*, vol. 20, no. 3, pp. 217–220, Mar. 2013.

[41] Y. Masuyama, K. Yatabe, and Y. Oikawa, "Model-based phase recovery of spectrograms via optimization on Riemannian manifolds," in *Proc. Int. Workshop Acoust. Signal Enhance.*, Sep. 2018, pp. 126–130.

[42] J. Engel, K. K. Agrawal, S. Chen, I. Gulrajani, C. Donahue, and A. Roberts, "GANSynth: Adversarial neural audio synthesis," in *Proc. Int. Conf. Learn. Represent.*, Apr. 2019.

[43] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," 2016, *arXiv:1612.08083*.

[44] O. Christensen, "Gabor frames and duality," in *An Introduction to Frames and Riesz Bases*, May 2016, Boston, MA, USA: Birkhäuser, pp. 287–325. [Online]. Available: https://doi.org/10.1007/978-3-319-25613-9_12

[45] O. Christensen, "Gabor frames and duality," in *An Introduction to Frames and Riesz Bases*, May 2016, Boston, MA, USA: Birkhäuser, pp. 1–46. [Online]. Available: https://doi.org/10.1007/978-3-319-25613-9_12

[46] J. R. Hershy, J. Le Roux, and F. Weninger, "Deep unfolding: Model-based inspiration of novel deep architectures," 2014, *arXiv:1409.2574*.

[47] K. Greger and Y. LeCun, "Learning fast approximations of sparse coding," in *Proc. Int. Conf. March. Learn.*, Jun. 2010, pp. 399–406.

[48] S. Takabe, M. Imanishi, T. Wadayama, and R. Hayakawa, "Trainable projected gradient detector for massive overloaded MIMO channels: Data-driven tuning approach," *IEEE Access*, vol. 7, pp. 93 326–93 338, Jul. 2019.

[49] S. Wisdom, J. Hershey, J. Le Roux, and S. Watanabe, "Deep unfolding for multichannel source separation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Mar. 2016, pp. 121–125.

[50] G. Wichern and J. Le Roux, "Phase reconstruction with learned time-frequency representations for single-channel speech separation," in *Proc. Int. Workshop Acoust. Signal Enhance.*, Sep. 2018, pp. 396–400.

[51] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing," 2019, *arXiv:1912.10557*.

[52] Y. Romano, M. Elad, and P. Milanfar, "The little engine that could: Regularization by denoising (RED)," *SIAM J. Imag. Sci.*, vol. 10, no. 4, pp. 1804–1844, Oct. 2017.

[53] E. T. Reehorst and P. Schniter, "Regularization by denoising: Clarifications and new interpretations," *IEEE Trans. Comput. Imag.*, vol. 5, no. 1, pp. 52–67, Mar. 2018.

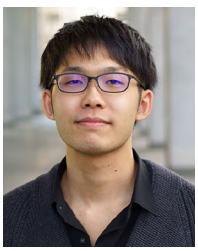
[54] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg, "Plug-and-Play priors for model based reconstruction," in *Proc. IEEE Glob. Conf. Signal, Inf. Process.*, Dec. 2013, pp. 945–948.

[55] S. H. Chan, X. Wang, and O. A. Elgendy, "Plug-and-Play ADMM for image restoration: Fixed-point convergence and applications," *IEEE Trans. Comp. Imag.*, vol. 3, no. 1, pp. 84–98, Mar. 2017.

[56] T. Meinhardt, M. Moller, C. Hazirbas, and D. Cremers, "Learning proximal operators: Using denoising networks for regularizing inverse imaging problems," in *Proc. Int. Conf. Comput. Vis.*, Oct. 2017, pp. 1799–1808.

[57] N. Parikh and S. Boyd, "Proximal algorithms," *Found. Trends Opt.*, vol. 1, no. 3, pp. 127–239, Jan. 2014.

[58] L. Condat, D. Kitahara, A. Contreras, and A. Hirabayashi, "Proximal splitting algorithms: Relax them all!," 2019, *arXiv:1912.00137*.
[59] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comp. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.
[60] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3142–3155, Jul. 2017.
[61] C. Trabelsi *et al.*, "Deep complex networks," in *Proc. Int. Conf. Learn. Represent.*, Apr. 2018.
[62] D. Wang and J. Chen, "Supervised speech separation based on deep learning: An overview," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 26, no. 10, pp. 1702–1726, Oct. 2018.
[63] A. Pandey and D. Wang, "Exploring deep complex networks for complex spectrogram enhancement," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2019, pp. 6885–6889.
[64] Y. Hu *et al.*, "DCCRN: Deep complex convolution recurrent network for phase-aware speech enhancement," Aug. 2020, *arXiv:2008.00264*.
[65] S. Bai, J. Z. Kolter, and V. Koltun, "Deep equilibrium models," in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2019, pp. 690–701.
[66] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "ALBERT: A lite BERT for self-supervised learning of language representations," in *Proc. Int. Conf. Learn. Represent.*, Apr. 2020.
[67] R. Prenger, R. Valle, and B. Catanzaro, "Waveglow: A flow-based generative network for speech synthesis," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, May 2019, pp. 3617–3621.
[68] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, May 2015.
[69] *P.862.2: Wideband Extension to Recommendation P.862 for Assessment Wideband Telephone Networks and Speech Codecs*, ITU-T Std. P.862.2, 2007.
[70] C. H. Taal, R. C. Hendriks, R. Heusdens, and J. Jensen, "An algorithm for intelligibility prediction of time-frequency weighted noisy speech," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 19, no. 7, pp. 2155–2136, Sep. 2011.
[71] N. Strumel and L. Daudet, "Signal reconstruction from STFT magnitude: A state of the art," in *Proc. Int. Conf. Digit. Audio Effects*, Sep. 2011, pp. 375–386.
[72] J. Shen *et al.*, "Natural tts synthesis by conditioning wavenet on mel spectrogram predictions," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Apr. 2018, pp. 4779–4783.
[73] K. Kumar *et al.*, "MelGAN: Generative adversarial networks for conditional waveform synthesis," in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2019, pp. 14 910–14 921.
[74] A. van den Oord *et al.*, "Wavenet: A generative model for raw audio," pp. 4779–4783, Apr. 2016, *arXiv:1609.03499*.
[75] K. Kastner, J. F. Santos, Y. Bengio, and A. Courville, "Representation mixing for TTS synthesis," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, May 2019, pp. 5906–5910.
[76] S. Maiti and M. I. Mandel, "Parametric resynthesis with neural vocoders," in *Proc. IEEE Workshop Appl. Signal Process. Audio Acoust.*, Oct. 2019, pp. 303–307.
[77] L. J. Liu, Z. H. Ling, Y. Jiang, M. Zhou, and L. R. Dai, "Wavenet vocoder with limited training data for voice conversion," in *Proc. INTERSPEECH*, Sep. 2018, pp. 1983–1987.
[78] S. Arik, J. Chen, K. Peng, W. Ping, and Y. Zhou, "Neural voice cloning with a few samples," in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2018, pp. 10 019–10 029.
[79] Y. Koizumi, K. Yatabe, M. Delcroix, Y. Masuyama, and D. Takeuchi, "Speech enhancement using self-adaptation and multi-head self-attention," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, May 2020, pp. 181–185.



Yoshiki Masuyama (Student Member, IEEE) received the B.E. degree from Waseda University in 2019. He is currently working toward the M.E. degree from the Department of Intermedia Art and Science, Waseda University. Mr. Masuyama is a member of the Acoustical Society of Japan (ASJ).

Kohei Yatabe (Member, IEEE) received the B.E., M.E., and Ph.D. degrees from Waseda University, in 2012, 2014, and 2017, respectively. He is currently an Assistant Professor with the Department of Intermedia Art and Science, Waseda University.

Yuma Koizumi (Member, IEEE) received the B.S. and M.S. degrees from Hosei University, in 2012 and 2014, respectively and the Ph.D. degree from the University of Electro-Communications in 2017. Since 2014, he has been with Nippon Telegraph and Telephone Corporation (NTT), where he has been involved with research on acoustic signal processing and machine learning. He is a member of the ASJ and the Institute of Electronics, Information, and Communication Engineers.

Yasuhiro Oikawa (Member, IEEE) received the B.E, M.E., and Ph.D. degrees in electrical engineering from Waseda University, in 1995, 1997, and 2001, respectively. He is currently a Professor with the Department of Intermedia Art and Science, Waseda University. His main research interests include communication acoustics and digital signal processing of acoustic signals. Prof. Oikawa is a member of ASJ, ASA, IEICE, IPSJ, VRSJ, and AIJ.

Noboru Harada (Senior Member, IEEE) received the B.S. and M.S. degrees in computer science from the Department of Computer Science and Systems Engineering, Kyushu Institute of Technology, Kitakyushu, Japan, in 1995 and 1997, respectively, and the Ph.D. degree in computer science from the Graduate School of Systems and Information Engineering, University of Tsukuba, Tsukuba, Japan, in 2017. Since joining NTT Corporation, Tokyo, Japan, in 1997, he has been involved with research on speech and audio signal processing, such as high efficiency coding and lossless compression. His current research interests include acoustic signal processing and machine learning for acoustic event detection, including ADS. Dr. Harada was the recipient of the Technical Development Award from the Acoustical Society of Japan (ASJ) in 2016, the Industrial Standardization Encouragement Awards from the Ministry of Economy Trade and Industry of Japan in 2011, and the Telecom System Technology Paper Encouragement Award from the Telecommunications Advancement Foundation of Japan in 2007. He is a member of the ASJ, the Institute of Electronics, Information and Communication Engineers, and the Information Processing Society of Japan.