OPTIMIZING NEURAL-NETWORK SUPPORTED ACOUSTIC BEAMFORMING BY ALGORITHMIC DIFFERENTIATION

Christoph Boeddeker, Patrick Hanebrink*, Lukas Drude, Jahn Heymann, Reinhold Haeb-Umbach*

Paderborn University, Department of Communications Engineering, Paderborn, Germany

ABSTRACT

In this paper we show how a neural network for spectral mask estimation for an acoustic beamformer can be optimized by algorithmic differentiation. Using the beamformer output SNR as the objective function to maximize, the gradient is propagated through the beamformer all the way to the neural network which provides the clean speech and noise masks from which the beamformer coefficients are estimated by eigenvalue decomposition. A key theoretical result is the derivative of an eigenvalue problem involving complex-valued eigenvectors. Experimental results on the CHiME-3 challenge database demonstrate the effectiveness of the approach. The tools developed in this paper are a key component for an end-to-end optimization of speech enhancement and speech recognition.

Index Terms— Acoustic Beamforming, Deep Neural Network, Complex-Valued Algorithmic Differentiation, Generalized Eigenvalue Problem

1. INTRODUCTION

While neural networks (NNs) have become state-of-the-art in automatic speech recognition (ASR), they are more recently also making inroads in speech enhancement. In particular, it has been shown that a NN is very effective at estimating the masks by which the speech and noise power spectral density (PSD) matrices can be computed from the noisy input signal [1, 2, 3]. The beamforming coefficients are then computed from those PSD matrices. Such a hybrid front-end incorporating both neural-network and model based elements was able to significantly improve the performance of a strong ASR backend, which involved a deep neural network (DNN) for acoustic modeling.

However, it is somewhat awkward that such a system involves two networks, one for mask estimation and one for acoustic modeling, with differing objective functions: the cross entropy between an ideal binary mask and the network output for the first and the cross entropy between the context-dependent phoneme state and its prediction by the network for the latter. An alternative to this is a deep network for acoustic modeling with multi-channel raw speech at its input. It had been shown that such a large network was indeed able to take advantage of the multi-channel input and learn the desired spatial selectivity [4]. A drawback of that approach, however, is that it needs large amounts of data and long training times until convergence. Further, one might argue why the network must learn beamforming operations from data if there exists a decades old theory of optimal beamforming.

The approach taken in this paper is therefore different: we stick to the model-based approach and employ a NN for mask estimation, however the network is optimized w.r.t. to a criterion which is closer to the ultimate goal than the former used criterion on mask purity. In our case this ultimate goal is related to speech enhancement (the output SNR), while we leave the optimization w.r.t. the same criterion as the NN for acoustic modeling for a companion paper [5].

To be specific, we are going to compute the gradient of the output SNR w.r.t. the weights of the NN for mask estimation. This is achieved by means of algorithmic differentiation (AD). In AD a function is decomposed into a series of elementary operations, and for each of these elementary operations a function is implemented which computes its gradient w.r.t. to the desired quantity [6, 7]. In our application this decomposition involves as the most crucial step, the derivative of an eigenvalue problem involving complex-valued eigenvectors. Its solution, which, to the best of our knowledge, is presented here for the first time, is perhaps the theoretically most valuable result of this paper.

The work presented here combines the work of [3, 8, 9] and uses the framework [6]. In [8] complex-valued neuronal networks are presented, especially the complex-valued chain rule. This rule is used to extend the real-valued "forward and reverse mode algorithmic differentiation" [9] to complex-valued matrices. The derivation for the gradient of the eigendecomposition from [9] is generalized to the complex-valued problem. Also an extended form is presented which is required when the following calculations depend on the magnitude of the eigenvectors. Thereby the objective function of the NN from [3] is replaced to train a NN optimized w.r.t. a SNR criterion. This approach improves previous configurations in terms of speech quality.

In the next section we describe our NN supported system for acoustic beamforming. Then we recapitulate some basics on optimization of a real-valued objective function which depends on complex-valued vectors in Section 3. Section 4 sketches the key step of algorithmic differentiation mentioned earlier, and Section 5 presents experimental results on the CHiME-3 challenge dataset demonstrating the effectiveness of our approach. Conclusions are drawn in Section 6.

2. MODEL

We consider an array of D microphones, which receives a noisy speech signal

$$\mathbf{Y}_{f,t} = \mathbf{X}_{f,t} + \mathbf{N}_{f,t}, \tag{1}$$

where $\mathbf{Y}_{f,t}$, $\mathbf{X}_{f,t}$ and $\mathbf{N}_{f,t}$ are the short-time Fourier transforms (STFT) of noisy speech, clean speech and noise at frequency bin f and time frame t, respectively. Our goal is to recover the clean speech image $X_{f,t,1}$, i.e., the clean speech signal as observed by the first microphone, via acoustic beamforming [10, Ch. 10]:

$$\hat{X}_{f,t,1} = \mathbf{w}_f^{\mathrm{H}} \mathbf{Y}_{f,t}, \tag{2}$$

*Both authors contributed equally.

This work was in part supported by Deutsche Forschungsgemeinschaft (DFG) under contract no. Ha3455/11-1

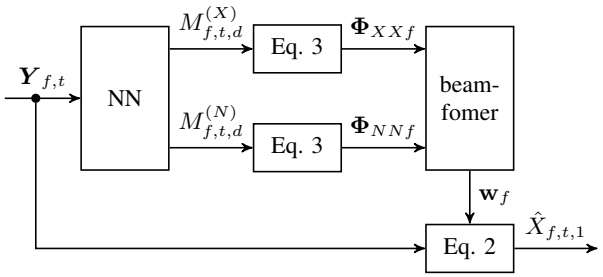

Fig. 1: Block diagram of the described system

where $\mathbf{w}_f$ is the beamformer coefficient vector, which is assumed to be time-invariant.

Fig. 1 gives an overview of the considered NN supported acoustic beamformer for speech enhancement [2].

The NN is applied to each microphone channel separately to estimate a speech and a noise mask $M^{(X)}_{f,t,d}$ and $M^{(N)}_{f,t,d}$, respectively. The D speech and noise masks are averaged to a single speech and a single noise mask, and the masks are used to obtain estimates of the speech and noise PSD matrices:

$$\mathbf{\Phi}_{\nu\nu f} = \frac{\sum_{\check{t}=1}^{T}\left(\sum_{d=1}^{D} M^{(\nu)}_{f,\check{t},d}\right)\mathbf{Y}_{f,\check{t}}\mathbf{Y}^{\mathrm{H}}_{f,\check{t}}}{\sum_{\check{t}=1}^{T}\left(\sum_{d=1}^{D} M^{(\nu)}_{f,\check{t},d}\right)}, \tag{3}$$

where $\nu \in \{X, N\}$ and where $(\cdot)^{\mathrm{H}}$ denotes the conjugate transpose. Once these PSD matrices are given, several statistically optimal beamformers can be computed. In this paper the minimum variance distortionless response (MVDR) beamformer and the generalized eigenvector (GEV) beamformer with an optional distortion reduction filter [11] are considered. For the latter the beamforming coefficients are then obtained by solving the GEV problem

$$\mathbf{\Phi}_{XXf}\mathbf{W}_f = \mathbf{\Phi}_{NNf}\mathbf{W}_f\mathbf{\Lambda}_f \tag{4}$$

with $\mathbf{W}_f$ and $\mathbf{\Lambda}_f$ being the matrix of eigenvectors and the diagonal matrix of eigenvalues, respectively. The beamforming vector is given by the eigenvector with the largest eigenvalue [11]

$$\mathbf{w}_f = \mathcal{P}(\mathbf{\Phi}_{XXf}, \mathbf{\Phi}_{NNf}). \tag{5}$$

Our goal is to derive update rules for the parameters of the NN such that an objective function computed from the beamformer output is optimized. Here we choose the negative SNR of the normalized beamformer output signal as our objective function to minimize:

$$J = -10\log_{10}\frac{P^{(X)}}{P^{(N)}}. \tag{6}$$

The power of the beamformed speech and noise signal is calculated via the estimated beamforming vector which is applied to the normalized clean speech and noise signals:

$$\begin{aligned}\boldsymbol{\nu}^{(\mathrm{Norm})}_{f,t} &= \frac{\boldsymbol{\nu}_{f,t}}{\sqrt{\sum_{\check{t}=1}^{T}\left|\left|\boldsymbol{\nu}_{f,\check{t}}\right|\right|^2}},\\ \nu^{(\mathrm{Beam})}_{f,t} &= \mathbf{w}^{\mathrm{H}}_f\boldsymbol{\nu}^{(\mathrm{Norm})}_{f,t},\\ P^{(\nu)} &= \sum_{t=1}^{T}\sum_{f=1}^{F}\frac{|\nu^{(\mathrm{Beam})}_{f,t}|^2}{T},\end{aligned} \tag{7}$$

where $\nu \in \{X, N\}$ and where T and F are the total number of frames and frequency bins, respectively. The normalization ensures that every frequency contributes equally strong to the objective.

Note that the objective function J is (of course) real-valued, while its computation from the input signal involves complex-valued quantities. For error backpropagation we thus need to compute complex-valued gradients. In the next section we recapitulate how this can be done.

3. COMPLEX-VALUED BACKPROPAGATION

To derive the basic math rules for complex-valued gradients, we consider a chain of two functions:

$$\begin{aligned}J &= f(s) \quad \text{where } s = \sigma + \mathrm{j}\omega,\\ s &= g(z) \quad \text{where } z = x + \mathrm{j}y,\\ s, z &\in \mathbb{C}, \quad J, \sigma, \omega, x, y \in \mathbb{R}\end{aligned} \tag{8}$$

and show how the partial derivative of J w.r.t. x and y can be computed.

It is possible to reformulate each operation as a real-valued operation on the real and imaginary part separately. Instead we want to define a complex derivative w.r.t. z^* (where $(\cdot)^*$ is the complex conjugate) under the assumption that the objective function is real-valued. In [12, Eq. 6] it is defined as

$$\frac{\partial J}{\partial z^*} = \frac{1}{2}\left(\frac{\partial J}{\partial x} + \mathrm{j}\frac{\partial J}{\partial y}\right), \tag{9}$$

which is known as the *Wirtinger calculus*.

The chain rule for complex functions can be derived by inserting the multivariable real-valued chain rule [13, p. 393]

$$\frac{\partial J}{\partial x} = \frac{\partial J}{\partial \sigma}\frac{\partial \sigma}{\partial x} + \frac{\partial J}{\partial \omega}\frac{\partial \omega}{\partial x} \tag{10}$$

in (9), which leads to

$$\begin{aligned}\frac{\partial J}{\partial z^*} &= \frac{1}{2}\left(\frac{\partial J}{\partial \sigma}\frac{\partial \sigma}{\partial x} + \frac{\partial J}{\partial \omega}\frac{\partial \omega}{\partial x} + \mathrm{j}\frac{\partial J}{\partial \sigma}\frac{\partial \sigma}{\partial y} + \mathrm{j}\frac{\partial J}{\partial \omega}\frac{\partial \omega}{\partial y}\right)\\ &= \cdots = \left(\frac{\partial J}{\partial s^*}\right)^*\frac{\partial s}{\partial z^*} + \frac{\partial J}{\partial s^*}\left(\frac{\partial s}{\partial z}\right)^*.\end{aligned} \tag{11}$$

This equation is the scalar version of [8, Eq. 7]. The extension of this chain rule to a matrix $\mathbf{S}$ as an intermediate variable is

$$\begin{aligned}\frac{\partial J}{\partial z^*} &= \sum_{n,m}\left(\left(\frac{\partial J}{\partial S^*_{n,m}}\right)^*\frac{\partial S_{n,m}}{\partial z^*} + \frac{\partial J}{\partial S^*_{n,m}}\left(\frac{\partial S_{n,m}}{\partial z}\right)^*\right)\\ &= \mathrm{Tr}\left(\left(\frac{\partial J}{\partial \mathbf{S}^*}\right)^{\mathrm{H}}\frac{\partial \mathbf{S}}{\partial z^*} + \left(\frac{\partial J}{\partial \mathbf{S}^*}\right)^{\mathrm{T}}\left(\frac{\partial \mathbf{S}}{\partial z}\right)^*\right),\end{aligned} \tag{12}$$

where $S_{n,m}$ is the element on the n-th row and m-th column of matrix $\mathbf{S}$, $(\cdot)^{\mathrm{T}}$ denotes the transpose and $\mathrm{Tr}(\cdot)$ is the sum of the diagonal matrix elements. Here, $\frac{\partial J}{\partial \mathbf{S}^*}$ and $\frac{\partial \mathbf{S}}{\partial z^*}$ are matrices with the derivative of J w.r.t. each element of $\mathbf{S}^*$ and the derivative for each element of $\mathbf{S}$ w.r.t. z^*, respectively. Both matrices have same shape as $\mathbf{S}$. This equation is the generalization of the real-valued equation in [9].

In the special case, that the function g is independent of y the gradient w.r.t y is zero and (12) simplifies to

$$\frac{\partial J}{\partial z^*} = \Re\left\{\left(\frac{\partial J}{\partial \mathbf{S}^*}\right)^*\frac{\partial \mathbf{S}}{\partial z^*} + \frac{\partial J}{\partial \mathbf{S}^*}\left(\frac{\partial \mathbf{S}}{\partial z}\right)^*\right\}, \tag{13}$$

where $\Re\{\cdot\}$ is the real part. Note that the gradient is real-valued as expected. This is the case for our problem, where the neural network is real-valued.

4. DERIVATIVE OF EIGENVALUE DECOMPOSITION

4.1. Forward and Backward Mode AD

The crucial step in the chain of derivatives when computing the gradient of the objective function w.r.t. the NN parameters is the eigenvalue problem

$$\mathbf{\Phi W} = \mathbf{W\Lambda}. \tag{14}$$

The following derivation is similar to the real-valued case discussed in [9].

According to the standard procedure in AD, we first consider forward mode AD, which starts at the beginning and differentiates each step of the computation [9]. We consider the derivative of (14) w.r.t. a complex scalar $\zeta \in \{z, z^*\}$:

$$\frac{\partial \mathbf{\Phi}}{\partial \zeta}\mathbf{W} + \mathbf{\Phi}\frac{\partial \mathbf{W}}{\partial \zeta} = \frac{\partial \mathbf{W}}{\partial \zeta}\mathbf{\Lambda} + \mathbf{W}\frac{\partial \mathbf{\Lambda}}{\partial \zeta} \tag{15}$$

$$\mathbf{W}^{-1}\frac{\partial \mathbf{\Phi}}{\partial \zeta}\mathbf{W} - \frac{\partial \mathbf{\Lambda}}{\partial \zeta} = \mathbf{W}^{-1}\frac{\partial \mathbf{W}}{\partial \zeta}\mathbf{\Lambda} - \mathbf{W}^{-1}\mathbf{\Phi}\frac{\partial \mathbf{W}}{\partial \zeta}, \tag{16}$$

from which the following derivatives can be obtained (we have to skip the intermediate steps due to space limitations, the full derivation can be found at [14]):

$$\frac{\partial \mathbf{\Lambda}}{\partial \zeta} = \mathbf{I} \circ \left(\mathbf{W}^{-1}\frac{\partial \mathbf{\Phi}}{\partial \zeta}\mathbf{W}\right). \tag{17}$$

and

$$\frac{\partial \mathbf{W}}{\partial \zeta} = \mathbf{W}\left(\mathbf{F} \circ \left(\mathbf{W}^{-1}\frac{\partial \mathbf{\Phi}}{\partial \zeta}\mathbf{W}\right)\right). \tag{18}$$

Here $\mathbf{I}$ is the identity matrix, $\circ$ denotes the Hadamard product (where $(A \circ B)_{ij} = A_{ij}B_{ij}$), and $F_{ij} = 1/(\lambda_j - \lambda_i)$ for $i \neq j$ and $F_{ii} = 0$.

For the computation of the error backpropagation we need the reverse mode AD, though, i.e. we have to work backwards through the sequence of computational steps originally used to compute the scalar output from the input. In our case this refers to computing the derivative of the objective function w.r.t. the PSD matrices.

Using (12) we have

$$\frac{\partial J}{\partial z^*} = \mathrm{Tr}\left(\left(\frac{\partial J}{\partial \mathbf{\Phi}^*}\right)^{\mathrm{H}}\frac{\partial \mathbf{\Phi}}{\partial z^*}\right) + \mathrm{Tr}\left(\left(\frac{\partial J}{\partial \mathbf{\Phi}^*}\right)^{\mathrm{H}}\frac{\partial \mathbf{\Phi}}{\partial z}\right)^*. \tag{19}$$

In this particular case it is sufficient to consider only the first term because a closer investigation reveals that the differentiation w.r.t. z leads to the same solution. We therefore leave out the second and focus on the first term

$$\frac{\partial J}{\partial z^*} = \mathrm{Tr}\left(\left(\frac{\partial J}{\partial \mathbf{\Phi}^*}\right)^{\mathrm{H}}\frac{\partial \mathbf{\Phi}}{\partial z^*}\right) + \ldots, \tag{20}$$

where the dots indicate the disregarded differentiation w.r.t. z. Also (12) can be used with two intermediate matrices, which leads to

$$\frac{\partial J}{\partial z^*} = \mathrm{Tr}\left(\left(\frac{\partial J}{\partial \mathbf{W}^*}\right)^{\mathrm{H}}\frac{\partial \mathbf{W}}{\partial z^*} + \left(\frac{\partial J}{\partial \mathbf{\Lambda}^*}\right)^{\mathrm{H}}\frac{\partial \mathbf{\Lambda}}{\partial z^*}\right) + \ldots, \tag{21}$$

whose terms can be expressed via the forward mode differentiations (17) and (18). To calculate $\frac{\partial J}{\partial \mathbf{\Phi}^*}$ we compare the coefficients of (20) with those of (21), which eventually delivers (again, the reader is referred to [14] for details):

$$\begin{aligned}\frac{\partial J}{\partial \mathbf{\Phi}^*} &= \left(\mathbf{W}\left(\left(\left(\frac{\partial J}{\partial \mathbf{W}^*}\right)^{\mathrm{H}}\mathbf{W}\right) \circ \mathbf{F}^{\mathrm{T}} + \left(\frac{\partial J}{\partial \mathbf{\Lambda}^*}\right)^{\mathrm{H}}\right)\mathbf{W}^{-1}\right)^{\mathrm{H}} \\ &= \mathbf{W}^{-\mathrm{H}}\left(\frac{\partial J}{\partial \mathbf{\Lambda}^*} + \mathbf{F}^* \circ \left(\mathbf{W}^{\mathrm{H}}\frac{\partial J}{\partial \mathbf{W}^*}\right)\right)\mathbf{W}^{\mathrm{H}}\end{aligned} \tag{22}$$

where $(\cdot)^{-\mathrm{H}}$ denotes the inverse of the conjugate transpose. This is the reverse mode gradient of the eigendecomposition. The real-valued version of this equation can be found in [9].

4.2. Extension of the gradient

The eigendecomposition has a degree of freedom for choosing the magnitude of the eigenvectors. Many numerical implementations (including `eig` of the numpy package) set the magnitude to one. To consider this in the backward step we use the normalization for vectors

$$\mathbf{a} = \frac{\mathbf{w}}{\sqrt{\mathbf{w}^{\mathrm{H}}\mathbf{w}}}, \tag{23}$$

$$\frac{\partial J}{\partial \mathbf{w}^*} = \frac{\frac{\partial J}{\partial \mathbf{a}^*} - \mathbf{w}\frac{\Re\{\mathbf{w}^{\mathrm{H}}\frac{\partial J}{\partial \mathbf{a}^*}\}}{\mathbf{w}^{\mathrm{H}}\mathbf{w}}}{\sqrt{\mathbf{w}^{\mathrm{H}}\mathbf{w}}}, \tag{24}$$

where $\mathbf{w}$ has already the magnitude one, which simplifies the equation to:

$$\frac{\partial J}{\partial \mathbf{w}^*} = \frac{\partial J}{\partial \mathbf{a}^*} - \mathbf{w}\Re\left\{\mathbf{w}^{\mathrm{H}}\frac{\partial J}{\partial \mathbf{a}^*}\right\}. \tag{25}$$

The extension to all eigenvectors $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_N]$ leads to

$$\frac{\partial J}{\partial \mathbf{W}^*} = \frac{\partial J}{\partial \mathbf{A}^*} - \mathbf{W}\left(\Re\left\{\mathbf{W}^{\mathrm{H}}\frac{\partial J}{\partial \mathbf{A}^*}\right\} \circ \mathbf{I}\right), \tag{26}$$

and the complete gradient of the eigendecomposition is

$$\begin{aligned}\frac{\partial J}{\partial \mathbf{\Phi}^*} = {} & \mathbf{W}^{-\mathrm{H}}\left(\frac{\partial J}{\partial \mathbf{\Lambda}^*} + \mathbf{F}^* \circ \left(\mathbf{W}^{\mathrm{H}}\frac{\partial J}{\partial \mathbf{W}^*}\right)\right)\mathbf{W}^{\mathrm{H}} \\ & - \mathbf{W}^{-\mathrm{H}}\left(\mathbf{F}^* \circ \left(\mathbf{W}^{\mathrm{H}}\mathbf{W}\left(\Re\left\{\mathbf{W}^{\mathrm{H}}\frac{\partial J}{\partial \mathbf{W}^*}\right\} \circ \mathbf{I}\right)\right)\right)\mathbf{W}^{\mathrm{H}}.\end{aligned} \tag{27}$$

This extension of the gradient is only required if the following calculations depend on the magnitude of the eigenvectors and $\mathbf{\Phi}$ is not a hermitian matrix.

4.3. Verification of the derivatives

For the verification of (22) it can be shown that the gradient is not affected by the identity equation

$$\mathbf{\Lambda}, \mathbf{W} = \mathrm{eig}(\boldsymbol{\phi}), \tag{28}$$

$$\tilde{\boldsymbol{\phi}} = \mathbf{W\Lambda W}^{-1}, \tag{29}$$

for an arbitrary matrix $\boldsymbol{\phi}$ and backward gradient $\frac{\partial J}{\partial \tilde{\boldsymbol{\phi}}^*}$. Since this identity is not sensitive to a change of the eigenvectors magnitude the extension of the gradient is not required.

The test of the extended gradient in (27) is done numerically by calculating the gradient of

$$\mathbf{\Lambda}, [\mathbf{w}_1, \ldots, \mathbf{w}_N] = \mathrm{eig}(\boldsymbol{\phi}), \tag{30}$$

$$\tilde{\mathbf{w}}_n = \mathbf{w}_n e^{-\mathrm{j}\angle w_{n,1}} \quad \text{for } n \in \{1, \ldots, N\}. \tag{31}$$

Table 1: Configuration of the neural network

	Units	Type	Non-Linearity	p_{dropout}
L1	256	BLSTM	Tanh	0.5
L2	513	FF	ReLU	0.5
L3	513	FF	ReLU	0.5
L4	1026	FF	Sigmoid	0.0

The second equation ensures the invariance w.r.t. an absolute phase of the eigenvectors which is necessary for a numerical verification. This test is only successful for the extended gradient.

4.4. GEV beamformer

We now apply the above results to the generalized eigenvalue problem (4). Assuming that the inverse of $\mathbf{\Phi}_{NN\,f}$ exists, the generalized eigenvalue problem can be rewritten as a special eigenvalue problem

$$\mathbf{\Phi}_{NN\,f}^{-1}\mathbf{\Phi}_{XX\,f}\mathbf{W}_f = \mathbf{W}_f\mathbf{\Lambda}_f. \tag{32}$$

With the backward gradient of the matrix inverse product [9]

$$\mathbf{\Phi}_f = \mathbf{\Phi}_{NN\,f}^{-1}\mathbf{\Phi}_{XX\,f}, \tag{33}$$

$$\frac{\partial J}{\partial \mathbf{\Phi}_{XX\,f}^*} = \mathbf{\Phi}_{NN\,f}^{-\mathrm{H}}\frac{\partial J}{\partial \mathbf{\Phi}_f^*}, \tag{34}$$

$$\frac{\partial J}{\partial \mathbf{\Phi}_{NN\,f}^*} = -\frac{\partial J}{\partial \mathbf{\Phi}_{XX\,f}^*}\mathbf{\Phi}_f^{\mathrm{H}}, \tag{35}$$

we can obtain the gradient of the generalized eigenvalue problem.

4.5. MVDR beamformer

The coefficients of the MVDR beamformer are given by

$$\mathbf{w}_f^{(\mathrm{MVDR})} = \frac{\mathbf{\Phi}_{NN\,f}^{-1}\mathbf{w}_f^{(\mathrm{PCA})}}{\mathbf{w}_f^{(\mathrm{PCA})\mathrm{H}}\mathbf{\Phi}_{NN\,f}^{-1}\mathbf{w}_f^{(\mathrm{PCA})}}, \tag{36}$$

where $\mathbf{w}_f^{(\mathrm{PCA})} = \mathcal{P}(\mathbf{\Phi}_{XX\,f}, \mathbf{I})$, i.e., the eigenvector corresponding to the largest eigenvalue (see (5)). The forward and backward mode AD involves only standard operations. The formulas for the real-valued case found in [9] can be transformed to the complex-valued case needed here using the definitions of section 3 above.

5. EVALUATION

We tested our approach on the CHiME-3 dataset [15]. The NN for mask estimation has four hidden layers, a bi-directional long short-term memory (BLSTM) layer followed by three fully connected feedforward (FF) layers, see Table 1.

The network was trained on the simulated training set (*tr05_simu* with 7138 utterances) using *dt05_simu* with 1640 utterances for cross validation. Evaluation was also conducted on the simulated data, because the separated speech and noise signals were needed to compute the PESQ score, which was used for performance evaluation, in addition to the SNR gain. The perceptual evaluation of speech quality [16] (PESQ) value on the MOS-LQO scale is calculated for wideband signals.

Table 2: Overview of the results for different system configurations. The first row shows the scores of the input noisy signal.

mask source	eval. beamf.	post filter	PESQ	SNR
-	-	-	1.21	3.85
Oracle [3]	GEV	-	1.73	15.35
		BAN	1.76	18.02
[3]	GEV	-	1.64	14.08
		BAN	1.70	16.03
NN-GEV	GEV	-	1.71	14.59
		BAN	1.75	16.05
NN-MVDR	MVDR	-	1.43	10.58
		$\|\| \cdot \|\|_2$	1.60	13.84
	GEV	-	1.60	14.21
		BAN	1.50	14.51

Table 2 shows the results for different training scenarios, beamformer configurations and post filters. The blind analytic normalization (BAN) post filter is defined in [11] and $|| \cdot ||_2$ indicates the normalization to unit length. In the first column the training configuration of the NN is denoted, while column two stated the beamformer used for evaluation. Both can be different since the NN only estimates the spectral masks and not the beamfoming vector directly.

Using the same GEV beamformer and NN architecture as in [3], however replacing the previously used objective function on mask purity with the SNR criterion derived here, is able to achieve a gain in PESQ and SNR (lines denoted by NN-GEV). The PESQ score reaches nearly the value of the oracle masks from [3], while the gap for the SNR is clearly larger. On the other hand the MVDR NN, which does not require the extended gradient from (27), is not able to produce a gain, irrespective of which beamformer is used for evaluation. Similar to [3], the normalizing post filter BAN improves the results for nearly all configurations.

We also experimented with pre-training the network with the objective function in [3] and then continuing with the SNR criterion, but this resulted in no further improvement.

6. CONCLUSIONS AND OUTLOOK

Using algorithmic differentiation this paper shows how a NN for mask estimation can be trained to optimize an objective function related to the beamformer output signals. As a key theoretical result we derived the complex gradient for an eigendecomposition. While the improvements in PESQ and SNR compared to an earlier mask-purity related criterion are not tremendous, the main contribution of this paper is that it paves the way for an end-to-end training of a system composed of a single channel speech/noise estimation network, a multichannel beamformer and an ASR network that can be jointly optimized w.r.t. the same criterion.

7. REFERENCES

[1] H. Erdogan, J. R. Hershey, S. Watanabe, M. Mandel, and J. Le Roux, "Improved MVDR Beamforming Uuing Single-Channel Mask Prediction Networks," *Proc. INTERSPEECH*, 2016.

[2] J. Heymann, L. Drude, A. Chinaev, and R. Haeb-Umbach, "BLSTM supported GEV Beamformer Front-End for the 3rd CHiME Challenge," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Dec 2015, pp. 444–451.

[3] J. Heymann, L. Drude, and R. Haeb-Umbach, "Neural Network Based Spectral Mask Estimation for Acoustic Beamforming," in *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[4] Y. Hoshen, R. J. Weiss, and K. W. Wilson, "Speech Acoustic Modeling from Raw Multichannel Waveforms," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2015, pp. 4624–4628.

[5] J. Heymann, L. Drude, C. Boeddeker, P. Hanebrink, and R. Haeb-Umbach, "BEAMNET: End-to-End Training of a Beamformer-Supported Multi-Channel ASR System," in *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[6] S. Tokui, K. Oono, S. Hido, and J. Clayton, "Chainer: a Next-Generation Open Source Framework for Deep Learning," in *Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Twenty-ninth Annual Conference on Neural Information Processing Systems (NIPS)*, 2015.

[7] D. Maclaurin, D. Duvenaud, M. Johnson, and R. P. Adams, "Autograd: Reverse-mode Differentiation of Native Python," 2015.

[8] L. Drude, B. Raj, and R. Haeb-Umbach, "On the Appropriateness of Complex-Valued Neural Networks for Speech Enhancement," in *Proc. INTERSPEECH*, 2016.

[9] M. Giles, "An Extended Collection of Matrix Derivative Results for Forward and Reverse Mode Automatic Differentiation," 2008.

[10] J. Li, L. Deng, R. Haeb-Umbach, and Y. Gong, *Robust Automatic Speech Recognition*, Elsevier, October 2015.

[11] E. Warsitz and R. Haeb-Umbach, "Blind Acoustic Beamforming based on Generalized Eigenvalue Decomposition," *IEEE Transactions on audio, speech, and language processing*, vol. 15, no. 5, pp. 1529–1539, 2007.

[12] D. H. Brandwood, "A Complex Gradient Operator and its Application in Adaptive Array Theory," in *IEE Proceedings F: Communications Radar and Signal Processing*, 1983, vol. 130, pp. 11–16.

[13] E. Kreyszig, *Advanced Engineering Mathematics*, John Wiley & Sons, 2010.

[14] C. Boeddeker, P. Hanebrink, L. Drude, J. Heymann, and R. Haeb-Umbach, "On the Computation of Complex-valued Gradients with Application to Statistically Optimum Beamforming," *arXiv:1701.00392 [cs.NA]*, 2017.

[15] J. Barker, R. Marxer, E. Vincent, and S. Watanabe, "The third 'CHiME' Speech Separation and Recognition Challenge: Dataset, Task and Baselines," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 504–511.

[16] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra, "Perceptual Evaluation of Speech Quality (PESQ) - a new Method for Speech Quality Assessment of Telephone Networks and Codecs," in *Acoustics, Speech and Signal Processing (ICASSP), 2001 IEEE International Conference on*. IEEE, 2001, vol. 2, pp. 749–752.