A User-centric Approach for Deep Residual-Echo Suppression in Double-talk

Amir Ivry, Israel Cohen, *Fellow, IEEE,* Baruch Berdugo

***Abstract*—We introduce a user-centric residual-echo suppression (URES) framework in double-talk. This framework receives a user operating point (UOP) that consists of two metric values: the residual echo suppression level (RESL) and the desired speech-maintained level (DSML) that the user expects from the RES outcome. Then, the URES pipeline undergoes three stages. Firstly, we consider a deep RES model with a tunable design parameter that balances between the RESL and DSML and utilizes 101 pre-trained instances of this model, each with a different design parameter value. Thus, an identical input is expected to generate a different pair of RESL and DSML values in the prediction of every instance. Second, every prediction is separately fed to a subsequent pre-trained deep model instance that estimates the RESL and DSML of the prediction since these metrics depend on unavailable information in practice. Lastly, each pair of RESL and DSML estimates is compared with the UOP. The pairs that match the UOP up to a given tolerance threshold are narrowed down to the prediction with the maximal acoustic-echo cancellation mean-opinion score (AECMOS), which is the output of the URES system. This suggested framework holds three prominent advantages introduced in this study: it generates an RES output with RESL and DSML that match a UOP, supports near-real-time tracking of UOP changes, and applies AECMOS maximization. Experimental results consider 60 h of varied real and synthetic data. Average results can achieve an AECMOS subjectively considered excellent with RESL and DSML deviations of roughly 2 dB from the UOP. Any UOP adjustment can be tracked in less than 40 ms with a real-time factor of 1.92, but due to the high computational resources demanded by the framework, this is enabled on-edge only with high-end dedicated hardware, which limits general availability.**

***Index Terms*—Residual-echo suppression, user-centric, double-talk, RESL and DSML, AECMOS, deep learning.**

I. Introduction

HANDS-FREE speech communication has become increasingly popular in recent years due to the growing trend of transitioning from face-to-face meetings to online meetings [1], which are characterized by two conversation ends; far-end and near-end. In business calls, for instance, the far-end speaker is commonly a single participant who wears headphones in a close-talk environment, while the near-end is an office conference room. In that setup, speech from the far-end is transmitted to the near-end, which echoes via a nonlinear loudspeaker. In modern conferencing, loudspeakers are frequently not enclosed with but are detached from the near-end microphone, which creates an acoustic coupling between the two [2]. Thus, in double-talk periods, the near-end microphone may capture reverberant echo, desired speech from participants in the near-end, and additional noises. This may cause echo to be transmitted back to the far-end and severely impede the conversation intelligibility [3], [4].

Various linear acoustic echo cancellation (AEC) systems combat this issue [5]–[10]. However, these methods often cannot eliminate echo presence in realistic setups due to nonideal hardware that induces nonlinearity between the echo and the far-end signal [11], the rapidly varying nature of the echo path, and the complicated modeling of echo in double-talk. Residual-echo suppression (RES) systems have achieved impressive results using deep learning to eliminate linear and nonlinear echo patterns that are still present after the linear AEC stage [12]–[21]. In double-talk, RES systems trade-off between residual-echo suppression and desired-speech distortion levels in their output [22]. To evaluate this trade-off, we have introduced two objective performance metrics for RES in double-talk [23]: the residual-echo suppression level (RESL) and the desired-speech maintained level (DSML). In [24], we showed a strong correlation between these metrics and the recent AEC mean-opinion score (AECMOS) objective metric, which predicts subjective human ratings of speech quality of AEC systems with high accuracy in double-talk [2], [25].

Existing studies on RES primarily focus on improving benchmark performance rather than supporting users' inputs. For instance, most RES systems neither offer a framework to trade-off between residual echo and speech-distortion levels at their output nor report performance across various operating points that represent this trade-off. Instead, users employ existing RES systems based on an average benchmark performance, which is frequently reported with metrics that do not distinguish residual-echo presence from desired-speech distortion [23], e.g., signal-to-distortion-ratio [26] or perceptual evaluation of speech quality [27]. Even if an off-the-shelf model is rendered suitable by a user for a specific scenario, adjustments based on user preferences are not supported. Although the AECMOS is currently the most accurate objective assessment for speech quality by humans, no RES system provides a mechanism to maximize the AECMOS. These gaps limit the user experience and flexibility in dynamic environments that often require personalized adjustments. In practice, a business presentation in a near-end conference room may lead the far-end listener towards low speech distortion. In contrast, residual echo suppression may be more important during frequent abrupt echo-path changes when transitioning from the presentation to a near-end multi-participant discussion.

We introduce the user-centric RES (URES) framework in

This research was supported by the Israel Science Foundation (grant no. 1449/23) and the Pazy Research Foundation.

The authors are with the Andrew and Erna Viterbi Faculty of Electrical and Computer Engineering, Technion-Israel Institute of Technology, Haifa 3200003, Israel (e-mail: sivry@campus.technion.ac.il; icohen@ee.technion.ac.il, bbaruch@technion.ac.il).

double-talk. The URES is initiated with a user operation point (UOP) that consists of two performance metrics values: the RESL and DSML [23] that the user wishes to experience from the RES prediction. The URES system then undergoes three stages. Firstly, we utilize an existing deep RES model introduced in [22]. This model embeds a design parameter that controls the trade-off between the RESL and DSML of the RES prediction. We consider 101 pre-trained instances from this model, each with a different design parameter value. Feeding the same input to all instances results in different RESL and DSML values in the prediction of every instance, which covers a wide range of UOPs. Second, each prediction is fed to a separate pre-trained deep model, which maps this prediction to its RESL and DSML estimates. This is essential since these metrics depend on the desired speech signal that is unavailable in double-talk in practice. Third, the estimates from all instances are compared with the UOP. The ones that match it, up to a given tolerance threshold specifying the allowed deviation from the UOP, are narrowed down to the single prediction with the maximal AECMOS transmitted to the far-end. The proposed URES system has three unique advantages: the RESL and DSML of its output match or approach the UOP, changes in the UOP can be tracked in near-real-time in less than 40 ms and with real-time factor (RTF) [28] of 1.92, and the AECMOS of its output is maximized.

The remainder of this paper is organized as follows. In Section II, we formulate the problem. In Section III, we describe the proposed solution. Section IV lays out the experimental setup. In Section V, we present the experimental results. Finally, in Section VI, we conclude.

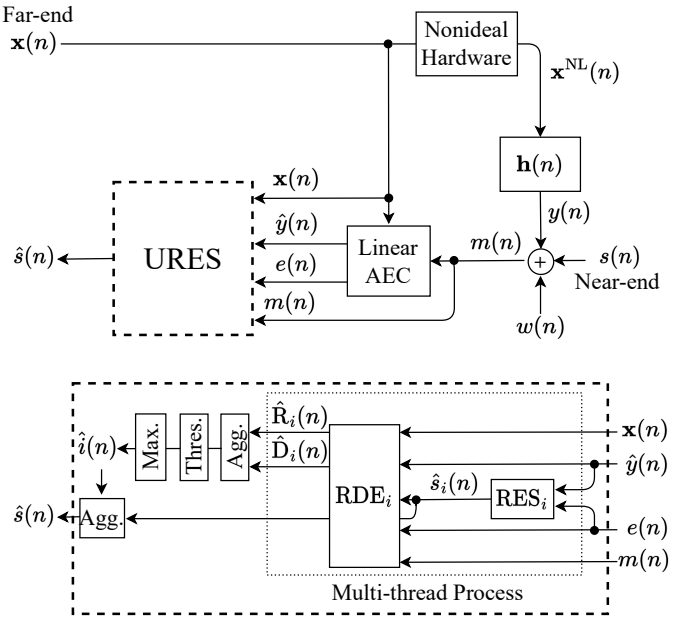

Fig. 1: The three stages of the proposed URES framework at time index n. (1) For $i \in \{0, 1, \ldots, 100\}$, the i^{th} model instance RES$_i$ produces a prediction $\widehat{s}_i(n)$. (2) $\widehat{s}_i(n)$ is inserted to the corresponding i^{th} model instance RDE$_i$, which estimates the RESL and DSML of $\widehat{s}_i(n)$, respectively denoted $\widehat{\mathrm{R}}_i(n)$ and $\widehat{\mathrm{D}}_i(n)$. (3) These estimates are aggregated over all i values and undergo threshold filtering by their proximity to the UOP, followed by an AECMOS maximization. The prediction with the chosen index $\widehat{i}(n)$, namely $\widehat{s}_{\widehat{i}}(n)$, is communicated to the far-end. Notice the RES and RDE models run inference in parallel across all their instances.

II. Problem Formulation

The proposed URES system is depicted in Fig. 1. Scalars are denoted in italics, and vectors are in bold and regarded as column vectors. All acoustic signals are assumed to be zero-mean unless stated otherwise. The near-end microphone signal in time index $n \in \mathbb{Z}$ is given by:

$$m(n) = s(n) + w(n) + y(n), \tag{1}$$

where $m(n), s(n), w(n), y(n) \in \mathbb{R}$. Here, $s(n)$ holds the desired speech and $w(n)$ holds environmental and system noises. The reverberant echo $y(n)$ satisfies

$$y(n) = \mathbf{h}^T(n)\,\mathbf{x}_{\mathrm{NL}}(n), \tag{2}$$

where $\mathbf{x}_{\mathrm{NL}}(n) \in \mathbb{R}^L$ denotes the L most recent samples of the nonlinearly distorted far-end signal, and $\mathbf{h}(n) \in \mathbb{R}^L$ is modeled as a finite impulse response filter with L coefficients that denote the echo path from the loudspeaker to the microphone:

$$\mathbf{x}^{\mathrm{NL}}(n) = \left[x^{\mathrm{NL}}(n), x^{\mathrm{NL}}(n-1), \ldots, x^{\mathrm{NL}}(n-L+1)\right]^T, \tag{3}$$

$$\mathbf{h}(n) = \left[h_0(n), h_1(n), \ldots, h_{L-1}(n)\right]^T. \tag{4}$$

We apply adaptive filtering for the linear AEC system that receives $m(n)$ as input and the L most recent samples of the far-end signal, i.e., $\mathbf{x}(n) \in \mathbb{R}^L$, as reference, and produces the echo-path estimate $\widehat{\mathbf{h}}(n) \in \mathbb{R}^L$:

$$\mathbf{x}(n) = \left[x(n), x(n-1), \ldots, x(n-L+1)\right]^T, \tag{5}$$

$$\widehat{\mathbf{h}}(n) = \left[\widehat{h}_0(n), \widehat{h}_1(n), \ldots, \widehat{h}_{L-1}(n)\right]^T. \tag{6}$$

The echo estimate $\widehat{y}(n) \in \mathbb{R}$ and adaptation error $e(n) \in \mathbb{R}$ in time index n can then be derived by calculating:

$$y(n) = \widehat{\mathbf{h}}^T(n)\,\mathbf{x}(n), \tag{7}$$

$$\begin{aligned} e(n) &= m(n) - \widehat{y}(n) \\ &\overset{(1)}{=} (y(n) - \widehat{y}(n)) + s(n) + w(n). \end{aligned} \tag{8}$$

The signals $\mathbf{x}(n)$, $\widehat{y}(n)$, $e(n)$, and $m(n)$ are the inputs of the URES system that produces the desired-speech estimate $\widehat{s}(n)$ and then communicates it to the far-end. The goal is that $\widehat{s}(n)$ confines to a UOP and achieves the maximal AECMOS value.

III. A User-centric Approach for Deep RES

This process is comprised of three main stages. The first stage is described in subsection III-A, where the user chooses a UOP that includes two values: the RESL and the DSML of the RES prediction. In the second stage, as detailed in subsections III-B and III-C, our deep models generate several RES predictions with RESL and DSML values that match the

UOP up to a given tolerance threshold. The third stage in subsection III-D depicts how the prediction with the highest AECMOS is chosen before being communicated to the far-end. At every iteration of the URES framework, it processes new information from frames with M samples that overlap by $\lceil M/2 \rceil$ samples with the previous frames, where $M > 1$.

A. Providing a user operating-point for the URES framework

The UOP consists of a pair of RESL and DSML values. In [23], we introduced the RESL and DSML metrics to separately assess residual echo and speech-distortion levels of RES systems in double-talk. We also provided empirical results of average RESL and DSML values in which the RES system operates, which may guide a UOP selection. Let the UOP in time index n be $(\mathrm{R}(n), \mathrm{D}(n))$, where $\mathrm{R}(n) \in \mathbb{R}$ is the RESL and $\mathrm{D}(n) \in \mathbb{R}$ is the DSML. This study supports $15 \leq \mathrm{R}(n) \leq 30$ and $7.5 \leq \mathrm{D}(n) \leq 15$, in dB.

B. RES with a tunable design parameter

Building upon our earlier work [22], we utilize a deep RES system that at time index n receives the M most recent samples of the outcomes of the linear AEC stage, i.e., the echo estimate $\hat{\mathbf{y}}(n) \in \mathbb{R}^M$ and the adaptation error $\mathbf{e}(n) \in \mathbb{R}^M$:

$$\hat{\mathbf{y}}(n) = [\hat{y}(n), \hat{y}(n-1), \ldots, \hat{y}(n-M+1)]^T, \quad (9)$$

$$\mathbf{e}(n) = [e(n), e(n-1), \ldots, e(n-M+1)]^T. \quad (10)$$

In practice, during training, the RES takes as inputs $\hat{\mathbf{y}}(n)$ and $\mathbf{e}(n)$ when they are concatenated to 29 past time frames of M samples each that overlap one another by $\lceil M/2 \rceil$ samples, to utilize past context. Let these context-dependent inputs be denoted by $\hat{\mathbf{y}}^{\mathrm{c}}(n) \in \mathbb{R}^{30M/2}$ and $\mathbf{e}^{\mathrm{c}}(n) \in \mathbb{R}^{30M/2}$:

$$\hat{\mathbf{y}}^{\mathrm{c}}(n) = [\hat{\mathbf{y}}(n), \hat{\mathbf{y}}(n-M/2), \ldots, \hat{\mathbf{y}}(n-29M/2)]^T, \quad (11)$$

$$\mathbf{e}^{\mathrm{c}}(n) = [\mathbf{e}(n), \mathbf{e}(n-M/2), \ldots, \mathbf{e}(n-29M/2)]^T, \quad (12)$$

where we omit the $\lceil \cdot \rceil$ sign from this point on for sake of clarity. For these inputs, the RES produces $\widehat{\mathbf{s}}(n) \in \mathbb{R}^M$, which aims to estimate $\mathbf{s}(n) \in \mathbb{R}^M$, i.e., the M most recent samples of the desired speech:

$$\widehat{\mathbf{s}}(n) = [\widehat{s}(n), \widehat{s}(n-1), \ldots, \widehat{s}(n-M+1)]^T, \quad (13)$$

$$\mathbf{s}(n) = [s(n), s(n-1), \ldots, s(n-M+1)]^T. \quad (14)$$

The RES system architecture is based on the UNet [29] neural network and is detailed in Appendix A-A. This system aims to remove residual-echo components and preserve the desired speech in the short-time Fourier transform (STFT) domain [30] by using an analysis window of M samples with $M/2$ samples overlap. During training, $\alpha \in \mathbb{R}$ is a non-negative design parameter that governs the trade-off between residual echo and speech distortion levels at the output of the RES system by regularizing the following objective function:

$$J(\alpha) = \left\| \widehat{\mathbf{S}} - \mathbf{S} \right\|_2^2 + \alpha \cdot \left\| \widehat{\mathbf{S}} \right\|_2^2 + \sigma_{\widehat{\mathbf{S}}}^2 \cdot \mathbb{I}_{\alpha>0}, \quad (15)$$

Here, $\widehat{\mathbf{S}} \in \mathbb{R}^F$ and $\mathbf{S} \in \mathbb{R}^F$ represent the STFT amplitudes of the time-domain frames $\widehat{\mathbf{s}}(n)$ and $\mathbf{s}(n)$, respectively. Also, $\|\widehat{\mathbf{S}}\|_2$ is the ℓ_2-norm of $\widehat{\mathbf{S}}$, $\sigma_{\widehat{\mathbf{S}}}^2$ is the variance of $\widehat{\mathbf{S}}$, and $\mathbb{I}_{\alpha>0}$ equals 1 when $\alpha > 0$ and 0 otherwise. For brevity, we neglect time-frequency index notations from (15), but it is explicitly mentioned that $\widehat{\mathbf{S}}$, $\mathbf{S}$, and $\sigma_{\widehat{\mathbf{S}}}^2$ are all functions of time and frequency. The objective function in (15) has been developed by the authors in [22], and in [22], [23] its functionality has been thoroughly investigated and experimental results have shown its inherent ability to create a trade-off between residual-echo suppression and desired-speech distortion levels in RES systems during double-talk. According to (15), when α increases, the training process inclines towards minimizing the norm of the prediction. This creates more residual echo suppression but constrains the speech component in the output to a higher distortion rate. In contrast, as α lowers and reaches $\alpha = 0$, more focus is put on minimizing the distortion between the system prediction and the desired speech for the cost of high residual echo presence.

In [23], we have shown how the average RESL values rise and how the average DSML values lower when α increases, and vice versa. Since higher values mean better performance for both the RESL and the DSML, shifting α can change the operating point of the RES system and match it with the UOP. We exploit this property and separately pre-train 101 identical instances of the RES system, each with a different α value ranging from $\alpha = 0$ to $\alpha = 1$ with increments of 0.01. This large number of α values separated by a thin resolution was empirically shown to cover a wide range of RESL and DSML pairs supporting the UOP. It was also revealed that $\alpha > 1$ causes undesired nullification of sub-bands in the RES prediction. The index $i \in \mathbb{N}_0$, where $i \in \{0, 1, \ldots, 100\}$, is used to denote each pre-trained RES model instance, i.e., RES_i, and each of its corresponding predictions in time index n, i.e., $\widehat{\mathbf{s}}_i(n) \in \mathbb{R}^M$. For all i values, the design parameter value used to pre-train RES_i is calculated by $\alpha_i = i/100$.

C. Estimation of the RESL and DSML metrics

Each prediction from the 101 RES system instances from subsection III-B separately undergoes RESL and DSML estimation. These estimates are then compared with the UOP. Formally, the RESL and DSML metrics [23] depend on the time-varying response of the RES system in double-talk:

$$\mathbf{p}(n) = \left. \frac{\widehat{\mathbf{s}}(n)}{\mathbf{e}(n)} \right|_{\text{Double-talk}}, \quad (16)$$

using element-wise division, where $\mathbf{p}(n) \in \mathbb{R}^M$ and $e(n-j) \neq 0$ in double-talk for all $j \in \mathbb{N}_0$ and $j \in \{0, 1, \ldots, M-1\}$. The expression of $\mathbf{p}(n)$ as the ratio between the output and input signals of the neural network in the time domain allows treating $\mathbf{p}(n)$ as a linear response, apply it separately to different time-domain signals, and inspect its influence on them. We illustrate the functionality of $\mathbf{p}(n)$ as a valid response expression by observing the popular signal-to-distortion-ratio (SDR) metric [26], and substitute (16) into it:

$$\begin{aligned} \mathrm{SDR} &= 10 \log_{10} \left. \frac{\|\mathbf{s}(n)\|_2^2}{\|\mathbf{s}(n) - \widehat{\mathbf{s}}(n)\|_2^2} \right|_{\text{Double-talk}} \\ &= 10 \log_{10} \left. \frac{\|\mathbf{s}(n)\|_2^2}{\|\mathbf{s}(n) - \mathbf{p}^T(n)\,\mathbf{e}(n)\|_2^2} \right|_{\text{Double-talk}}. \end{aligned} \quad (17)$$

Namely, applying $\mathbf{p}(n)$ to the input of the neural network $\mathbf{e}(n)$ in the time domain results is the output of the neural network $\widehat{\mathbf{s}}(n)$, and this relation is represented inside the SDR in (17). Before defining the RESL and DSML metrics, we recognize that deep models may apply inherent bias and compensate for it by defining $\tilde{\mathbf{s}}(n) = \widehat{p}(n)\,\mathbf{s}(n)$, where $\widehat{p}(n) \in \mathbb{R}$ and is given by:

$$\widehat{p}(n) = \frac{\left\langle \mathbf{p}(n)\,\mathbf{s}(n), \mathbf{s}(n) \right\rangle}{\|\mathbf{s}(n)\|_2^2}. \tag{18}$$

Here, $\mathbf{p}(n)\,\mathbf{s}(n)$ is done element-wise and $\langle \cdot, \cdot \rangle$ is the internal product between vectors. Then, by applying the response $\mathbf{p}(n)$ to the desired speech only and calculating the following ratio, the DSML scalar value in time index n is derived by:

$$\text{DSML} = 10\log_{10} \frac{\|\tilde{\mathbf{s}}(n)\|_2^2}{\|\tilde{\mathbf{s}}(n) - \mathbf{p}(n)\,\mathbf{s}(n)\|_2^2}\bigg|_{\text{Double-talk}}. \tag{19}$$

The scalar value of the RESL in time index n is manufactured by considering the noisy residual-echo estimate $\mathbf{r}(n) = \mathbf{e}(n) - \mathbf{s}(n)$, where $\mathbf{r}(n) \in \mathbb{R}^M$ and calculating:

$$\text{RESL} = 10\log_{10} \frac{\|\mathbf{r}(n)\|_2^2}{\|\mathbf{p}(n)\,\mathbf{r}(n)\|_2^2}\bigg|_{\text{Double-talk}}, \tag{20}$$

where $\mathbf{p}(n)\,\mathbf{r}(n)$ is done element-wise. According to (19)–(20), the RESL and DSML metrics cannot be calculated in practice since they require knowledge of the desired speech $\mathbf{s}(n)$. Namely, during inference, the prediction of the RES system cannot be translated into its RESL and DSML values. Thus, we developed a deep model denoted an RESL-DSML estimator (RDE) that estimates the relation between available acoustic signals and the RESL and DSML via implicit evaluation of $\mathbf{s}(n)$. To explain how we choose the inputs of the RDE, we retrieve a scalar-based view instead of a frame-based view and recognize with (1), (2) that for every time index n:

$$s(n) = m(n) - \mathbf{h}^T(n)\,\mathbf{x}_{\text{NL}}(n) - w(n). \tag{21}$$

By considering $m(n)$ as an input to the RDE and by ignoring the noise $w(n)$, it is left to estimate $\mathbf{h}(n)$ and $\mathbf{x}_{\text{NL}}(n)$. Based on the linear relation in (7), inserting both $\widehat{y}(n)$ and $\mathbf{x}(n)$ to the RDE should yield $\widehat{\mathbf{h}}(n)$, which estimates $\mathbf{h}(n)$. Notice that $\widehat{\mathbf{h}}(n)$ is practically available from the linear AEC stage, but its non-speech structure makes it more effective to feed the RDE with speech signals and derive implicit relations between them, which is empirically supported in our internal experiments. By (1) and (2), we estimate $\mathbf{x}_{\text{NL}}(n)$ by using $\mathbf{x}(n)$ and $m(n)$. The former constitutes a linear part of $\mathbf{x}_{\text{NL}}(n)$, and the latter is a mix of signals that includes $\mathbf{x}_{\text{NL}}(n)$. The RDE is also fed with $e(n)$, employed in the RESL and DSML calculations. As a final input, we insert $\widehat{s}(n)$ to the model since it is both an integral component of the RESL and DSML calculations and because it is constructed to approximate $s(n)$. Similarly to subsection III-B, we utilize 101 identical RDE model instances. RDE_i, which denotes the i^{th} RDE instance, receives five channels in the time domain, i.e., $\mathbf{x}(n)$, $\widehat{\mathbf{y}}(n)$, $\mathbf{e}(n)$, $\mathbf{m}(n)$, and $\widehat{\mathbf{s}}_i(n)$, where we now return to the frame-based view.

Let the predicted RESL and DSML values of RES_i in time index n be respectively denoted as $\widehat{\text{R}}_i(n) \in \mathbb{R}$ and $\widehat{\text{D}}_i(n) \in \mathbb{R}$. Notice that in this specific instance, we utilize only the M most recent samples from $\mathbf{x}(n)$, and not its full L most recent samples as in (5), where $L > M$ since L represents the length of an echo path that is traditionally longer than the length of the analysis time frame M. During training, the ℓ_2 distance is minimized between the pair of estimates $\widehat{\text{R}}_i(n)$ and $\widehat{\text{D}}_i(n)$, and the pair of ground truth calculations of the RESL and DSML obtained from (16)–(20). The architecture of the RDE model is detailed in Appendix A-B.

D. Maximizing the AECMOS

In this stage, we describe how the final prediction of the URES framework is determined before being communicated to the far-end. First, for all i values, $\widehat{\text{R}}_i(n)$ and $\widehat{\text{D}}_i(n)$ are aggregated into one batch that contains 101 pairs of values. Second, the UOP from subsection III-A is being compared against each pair in this batch. Let us respectively define the maximal allowed deviation of $\widehat{\text{R}}_i(n)$ and $\widehat{\text{D}}_i(n)$ from the UOP coordinates $\text{R}(n)$ and $\text{D}(n)$ using the non-negative tolerance threshold values $\text{TH}_{\text{R}}(n) \in \mathbb{R}$ and $\text{TH}_{\text{D}}(n) \in \mathbb{R}$. Consider the set $A = \{0, 1, \ldots, 100\}$ to contain all the possible 101 RDE systems indices, and its subset $A^{\text{TH}}(n) \subseteq A$ that contains only the indices in A that confine to the two following conditions in time index n:

$$\left|\widehat{\text{R}}_i(n) - \text{R}\right| < \text{TH}_{\text{R}}(n), \tag{22}$$

$$\left|\widehat{\text{D}}_i(n) - \text{D}\right| < \text{TH}_{\text{D}}(n), \tag{23}$$

where $\text{TH}_{\text{R}}(n)$ and $\text{TH}_{\text{D}}(n)$ are in dB. We denote the number of indices in $A^{\text{TH}}(n)$, i.e., its cardinality, as $P(n) \in \mathbb{N}_0$, where $P(n) \in \{0, 1, \ldots, 101\}$. Notice that when $P(n) = 0$ this means that in time index n the estimated RESL and DSML of every prediction of the URES system has over-deviated from the UOP beyond $\text{TH}_{\text{R}}(n)$ and $\text{TH}_{\text{D}}(n)$, in dB. In this case, our system falls back to the prediction with the minimal ℓ_2-norm between its estimated RESL and DSML and the UOP and reports to the user with suggestions to increase the threshold values. Our experimental results in Subsection V-C show that while neither $\text{TH}_{\text{R}}(n)$ nor $\text{TH}_{\text{D}}(n)$ fall below 1 dB, then $P(n) > 0$ for every time index n. Third, we turn to the AECMOS and calculate it in time index n using a long past-context window. We denote these inputs to the AECMOS by $\widehat{\mathbf{s}}^{\text{MOS}}(n)$, $\mathbf{e}^{\text{MOS}}(n)$, and $\mathbf{x}^{\text{MOS}}(n)$:

$$\widehat{\mathbf{s}}^{\text{MOS}}(n) = [\widehat{s}(n), \widehat{s}(n-1), \ldots, \widehat{s}(n-N+1)]^T, \tag{24}$$

$$\mathbf{e}^{\text{MOS}}(n) = [e(n), e(n-1), \ldots, e(n-N+1)]^T, \tag{25}$$

$$\mathbf{x}^{\text{MOS}}(n) = [x(n), x(n-1), \ldots, x(n-N+1)]^T, \tag{26}$$

where typically $N >> M$. Let $\widehat{i}(n) \in A^{\text{TH}}(n)$ denote the index of the RES system that produced the prediction with the highest AECMOS scalar value in time index n, namely:

$$\widehat{i}(n) = \underset{i \in A^{\text{TH}}(n)}{\arg\max}\, \text{AECMOS}\left(\widehat{\mathbf{s}}_i^{\text{MOS}}(n), \mathbf{e}^{\text{MOS}}(n), \mathbf{x}^{\text{MOS}}(n)\right), \tag{27}$$

where AECMOS $\in \mathbb{R}$. We denote $\Delta_{\mathrm{R}}(n), \Delta_{\mathrm{D}}(n) \in \mathbb{R}$ as the deviations of outputs of $\mathrm{RDE}_{\widehat{i}}$ from the UOP in time index n:

$$\Delta_{\mathrm{R}}(n) = \left| \widehat{\mathrm{R}}_{\widehat{i}}(n) - \mathrm{R}(n) \right|, \tag{28}$$

$$\Delta_{\mathrm{D}}(n) = \left| \widehat{\mathrm{D}}_{\widehat{i}}(n) - \mathrm{D}(n) \right|, \tag{29}$$

where $0 \leq \Delta_{\mathrm{R}}(n) < \mathrm{TH}_{\mathrm{R}}$ and $0 \leq \Delta_{\mathrm{D}}(n) < \mathrm{TH}_{\mathrm{D}}$ by definition. Finally, for all i, the predictions $\widehat{\mathbf{s}}_i(n)$ are aggregated into one batch, and $\widehat{\mathbf{s}}_{\widehat{i}}(n)$ is communicated to the far-end.

IV. Experimental Settings

Principal information is shared in this section, and remaining details are given in Appendix B; database acquisition is detailed in Appendix B-A, and preprocessing, training, and inference parameters are given in Appendix B-B.

A. Database acquisition

We utilize 50 h from the AEC-challenge database and 10 h of independent recordings performed in our lab. Both corpora contain only double-talk periods, i.e., where far-end and near-end speech overlap.

The AEC-challenge corpus was sampled at 16 KHz and is detailed in [31]. It includes acoustic scenarios when no echo-path change occurs and when it occurs regularly. No echo-path change describes scenarios when neither the near-end speakers nor near-end devices move. In contrast, echo-path change describes scenarios when at least one of the above moves regularly during the recording. We extract from this database 10 h of synthetic data and 40 h of real recordings, where the latter were captured using roughly $1,000$ hands-free devices in various acoustic environments. This data considers a wide range of noise and echo levels, having signal-to-echo-ratio (SER) distributed in $[-10, 10]$ dB and signal-to-noise-ratio (SNR) distributed in $[0, 40]$ dB.

The independent recordings were sampled at 16 KHz and employed clips from the TIMIT [32] and Librispeech [33] databases. This data only includes acoustic segments with no echo-path changes. A mouth simulator played the near-end speech, and a loudspeaker modeled the effect of the nonlinear echo inside the near-end, where both devices were located in various positions in the room during the experiment. Both the speech and echo were captured by a microphone in the near-end.

This database was collected to model especially challenging real-life acoustic scenarios that exhibit high echo levels. The SER levels were distributed in $[-20, -10]$ dB and SNR levels were roughly distributed in $[27, 37]$ dB. Formally, SER=$10\log_{10}\left[\|s(n)\|_2^2 / \|y(n)\|_2^2\right]$ in dB and SNR=$10\log_{10}\left[\|s(n)\|_2^2 / \|w(n)\|_2^2\right]$ in dB.

B. Preprocessing, training, and inference

The training set comprises 45 h from the AEC-challenge database; 35 h was randomly split from the 40 h batch of real recordings, and 10 h of synthetic data were included. The training set also contains 5 h from the real independent recordings. The test set comprises only real recordings: the remaining 5 h from the AEC challenge and the remaining 5 h from the independent recordings. The training and test sets are balanced to avoid bias by following guidelines from the preprocessing stage in [22]. Specifically, they contain equal representation for male and female participants, the far-end and near-end speakers are different, no speaker participates in both the training and test sets, and every speaker has been assigned as the far-end and near-end speaker. The linear AEC stage that precedes the URES system is a sign-error normalized least mean square (SNLMS) adaptive filter [6], [34] that operates in the time domain with a filter length of 150 ms. The training and test sets are each divided into 10 s segments and internally shuffled. This leads to abrupt echo-path changes that create frequent re-convergence of the linear AEC filter, as commonly occurs in real life [35], [36].

During training, each time-domain signal is converted to its STFT amplitude and normalized before being inserted into the RES model. The output of this RES model then undergoes de-normalization and inverse STFT [30] using the overlap-save method [37] by employing the phase from the adaptation error of the linear AEC system. Normalization is done by subtracting its minimal value from the training set and dividing it by its dynamic range. De-normalization is the inverse process. The RES and RDE models share the training samples of the echo estimate and adaptation error. The predictions of the RES models from the training stage are utilized to train the RDE models. During inference, normalization, and de-normalization in the RES stage are applied using the statistics from the training set [38].

C. Performance Measures

We use the AECMOS version number 4 from the API of Microsoft [25] and calculate it using the input signals in (24)–(26). It should be noted that the first AECMOS category is for call quality degradation caused by echo, and the second AECMOS category is for call quality degradation caused by other sources, including noise, missing audio, distortions, and cut-outs. In the scope of our study, we investigate the performance of an RES system in double-talk while ignoring acoustic phenomena such as noise sources, audio packet loss, communication interruption, and the like. Instead, we focus on the user experience when they judge the call quality degradation caused by the echo presence. Therefore, the AECMOS value we include in our calculations is of the first category, which has been trained to predict the human subjective rating to the question "How would you judge the degradation from the echo?".

Let us consider the integration of the second category into the study by either reporting its value on the output signal chosen by maximizing the first AECMOS category, or by changing the functionality of our method to maximize the second category instead. To learn how informative the first is for the user, we conducted an internal experiment that has shown that the correlation is weak between the two categories of the AECMOS. The second option deviates from the prime contribution of this study, which is enabling communication with least quality degradation due to echo from the view of the subjective user.

Specific cases may come to mind to stress the need for the complementary view by the second AECMOS category. Let us consider one, where the near-end microphone signal is muted to achieve a perfect echo removal while also losing the desired signal. By-design, our study supports only double-talk scenarios and user-chosen RESL and DSML values between $[15, 30]$ and $[7.5, 15]$, respectively, in dB. In case the near-end microphone outputs zero, both of these values cannot be defined at all. Also note that the AECMOS has not been trained on muted microphone scenarios. In practical systems it can be automatically detected when the microphone outputs zero, and there should be low probability of activating the proposed system in such scenarios.

The AECMOS is unitless and ranges on a scale of 1 to 5, where 5 is the best score. The AECMOS values are calculated and reported over segments of length N that shift by $M/2$. The AECMOS model was trained using and was optimized for long context windows, which are required to get meaningful results that emulate subjective human ratings. Short windows, e.g., 20 ms, to calculate the AECMOS empirically yield noisy and unreliable values.

Additional evaluation metrics include the RESL and DSML as correspondingly defined in (19) and (20), the value of $P\left(n\right)$ as defined in subsection III-D, and $\Delta_{\mathrm{R}}\left(n\right)$ and $\Delta_{\mathrm{D}}\left(n\right)$ that are respectively calculated using (28) and (29). These metrics are derived by considering a shorter sliding analysis window in the time domain of M samples, with the same step-size as for the AECMOS of $M/2$. This is done to capture the system's behavior with thin resolution, allowing us to dive deep into various interesting data trends during research. However, we recognize that this short window is often noisy, and thus, we report performance in this study by averaging these metrics over long periods, e.g., the test set. The alternative of calculating the response and its dependent metrics using both shorter and longer context windows has been internally examined. It has empirically led to less accurate performance analysis.

To provide the reader with a more holistic perspective of the performance of the URES framework in double-talk, we report three additional evaluation metrics. First is the perceptual evaluation of speech quality (PESQ) metric in wideband mode [27], a unitless measure between 1.5 and 4.5 where a higher value indicates better performance. Second is the deep noise-suppression mean opinion-score (DNSMOS) metric [39], which, similarly to the AECMOS, predicts human subjective ratings, but instead of examining the influence of echo on speech quality as in the AECMOS case, the DNSMOS queries human raters about how noise affects speech quality. Nonetheless, the DNSMOS provides another important estimation of human perception of the output of the URES framework. The DNSMOS is a unitless measure between 1 and 5; higher values indicate better human rating assessment. The PESQ and DNSMOS are calculated using a window size of N samples with a step-size of $M/2$, since they aim to capture perceptual evaluations of human ratings and require a long context window to produce meaningful results. Third, we report the echo-return loss enhancement (ERLE) [17], which measures how echo is removed between the degraded input and enhanced output of the URES system. ERLE is calculated using an analysis window in the time domain with M samples, with a step-size of $M/2$. The ERLE $\in \mathbb{R}$ in time index n is given by the following, in dB:

$$\mathrm{ERLE}\left(n\right) = 10\log_{10}\frac{\|\mathbf{e}\left(n\right)\|_2^2}{\|\widehat{\mathbf{s}}_{\widehat{i}}\left(n\right)\|_2^2}\ . \tag{30}$$

It should be noted that single-talk scenarios are naturally not evaluated in this study since the URES framework was explicitly built to address segments in which both desired speech and residual echo are present. This is expressed technically in the URES functionality, e.g., when the response calculation introduced in (16) cannot have values in its denominator that equal 0, which is a probable possibility in far-end-only events and even more possible in near-end-only events. Even though the URES framework only focuses on double-talk, real-life communication involves a constant shift between double-talk and single-talk scenarios, and integrating a single-talk-supporting module into the URES framework in future research may enhance its practicality.

V. Experimental Results

During the inference stage, every utterance from the test set is inferred with a random UOP pair where the RESL value is uniformly drawn from $[15, 30]$ dB and the DSML value is uniformly drawn from $[7.5, 15]$ dB. Unless stated otherwise, results are reported using mean and standard deviation values of performance metrics across the entire test set. In the tables, the format is mean$\pm\sigma$, where σ stands for standard deviation, and in the figures, the format includes mean values either with or without standard deviation error bars. This section neglects time indices from notation because it reports global results across the test set.

It is worth emphasizing two points regarding the AECMOS calculation during the inference stage of the test-set. First, due to the nature of the AECMOS calculation, the data streaming must accumulate 15 s before results are produced. Second, recall that the URES framework processes present time frames of 20 ms duration, which consist of only a negligible portion of the entire data used to calculate their associated AECMOS values that undergo maximization, calculated over a long past-context window of 15 s. Thus, even though the output frames of the URES framework are accumulated in the far-end, the actual AECMOS that the far-end user experiences is not the accumulation of the AECMOS values used during the URES inference process. To assess the AECMOS performance of the URES framework adequately, we first apply inference to the entire test-set, and then we run AECMOS using the inferred output stream of the URES framework while maintaining the 15 s analysis window size and 10 ms step-size. Following the same logic, the PESQ and the DNSMOS metrics are calculated and reported similarly.

A. Validating the performance of the RDE models

This experiment examines the estimation reliability of the RESL and DSML values by the 101 RDE model instances. Using 10-fold cross-validation [40], 80% of the training set is utilized for training, and the remaining 20% is used for

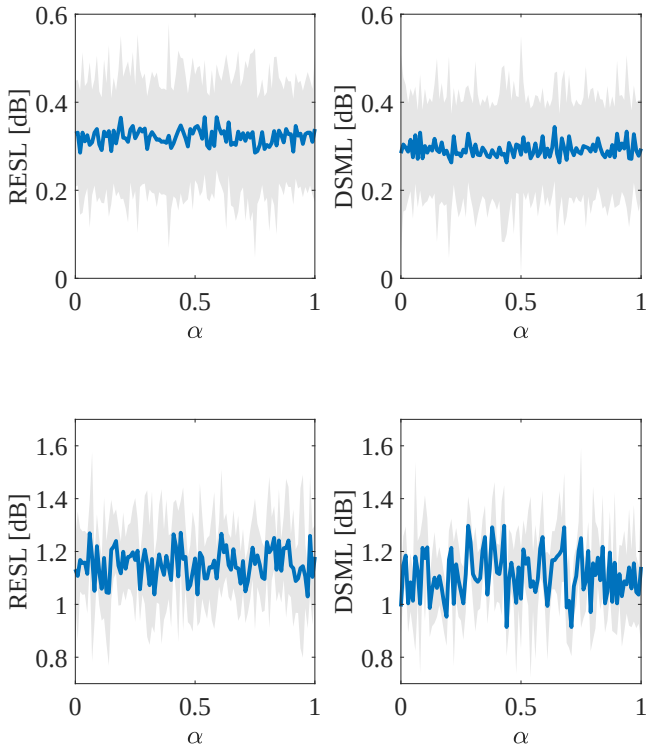

Fig. 2: Top: The ℓ_1 error of the RESL (left) and DSML (right) estimates for each of the 101 RDE model instances versus their α values. Bottom: The ℓ_1 error of the RESL (left) and DSML (right) estimates of a single RDE model instance versus the α values associated with the preceding RES model instances.

validation, where the same bias-free principles between the training and test sets detailed in subsection IV-B are applied between the crossed training and validation sets in every fold.

For every fold and for every i, where $0 \leq i \leq 100$, the crossed training set is used to train the model instances RES_i and RDE_i by following the process in subsection IV-B. Then, RDE_i infers the crossed validation set and produces the corresponding RESL and DSML estimates. These estimates are being compared against the ground-truth RESL and DSML of the validation set. Fig. 2 shows the RESL and the DSML estimation performance of all 101 RDE model instances. For both the RESL and the DSML, the reported values are the mean and standard deviation of the ℓ_1 distance between the estimates and their ground truth across all folds.

Recall that $\alpha_i = i/100$, it is shown that the RESL estimate experiences a maximal mean error of 0.36 dB for $\alpha_{54} = 0.54$, and one standard deviation can bring the error up to 0.57 dB for $\alpha_{39} = 0.39$. The DSML estimate has a maximal mean error of 0.34 dB for $\alpha_{64} = 0.64$, and one standard deviation can bring the error up to 0.5 dB for $\alpha_{54} = 0.54$. Considering this study supports RESL in $[15, 30]$ dB and DSML in $[7.5, 15]$ dB, the maximal mean error values can also be viewed in a relative scale by normalizing them by their corresponding ranges; namely $100 \cdot 0.36/15 = 2.4\%$ and $100 \cdot 0.34/7.5 = 4.5\%$. Based on these results, a subjective view suggests that using 101 RDE model instances produces a consistently reliable average estimation of the RESL and DSML in various acoustic setups.

The following experiment examines the computationally less-heavy possibility of employing a single RDE model for all α values. Similarly to the previous experiment, a 10-fold cross-validation is used to train every RES model instance with its corresponding α value. This time, however, all the outputs of the RES model instances are aggregated, and a single RDE model is used for training and validation for every fold. To ensure bias-free results, the distribution of segments associated with every α value is uniform in every fold's crossed training and validation sets. According to Fig. 2, it is shown that the RESL estimate experiences a maximal mean error of 1.27 dB for $\alpha_{44} = 0.44$, and one standard deviation can bring the error up to 1.57 dB for $\alpha_7 = 0.07$. The DSML estimate has a maximal mean error of 1.29 dB for $\alpha_{68} = 0.68$, and one standard deviation can bring the error up to 1.59 dB for $\alpha_{75} = 0.75$. Again, the maximal mean error values can also be viewed in a relative scale, namely $100 \cdot 1.27/15 = 8.4\%$ and $100 \cdot 1.29/7.5 = 17.2\%$. Based on results, a subjective view suggests that a single RDE model is unreliable in estimating the average RESL and DSML values.

To recap, utilizing a single RDE model may cause an accumulated uncertainty and bias of results, while 101 RDE model instances provide confident results. This renders the computational load of the latter worthy.

B. The effect of the tolerance threshold values on performance

The performance of the URES framework is examined concerning the tolerance threshold parameters TH_R and TH_D. We consider $(\text{TH}_\text{R}, \text{TH}_\text{D})$ pairs that confine to $\text{TH}_\text{R} \in \{1, 2, 3\}$ in dB and $\text{TH}_\text{D} \in \{1, 2, 3\}$ in dB, which yields 9 possible pairs combinations. These sets' values are representative of the URES system behavior but do not significantly deviate from the UOP. For each $(\text{TH}_\text{R}, \text{TH}_\text{D})$ pair, the mean and standard deviation of Δ_R, Δ_D, and the AECMOS are reported.

Table I considers test set utterances only with no echo-path changes. A clear trade-off between the tolerance threshold values and the yielded AECMOS is shown. Limiting the permitted deviation of the RESL and DSML estimates from the UOP to 1 dB leads to a mean AECMOS value of 3.1 out of 5, considered a subjectively mediocre human evaluation. Allowing a larger deviation of $(\text{TH}_\text{R}, \text{TH}_\text{D}) = (3, 3)$ in dB, leads to an AECMOS average of 4.4, which is subjectively considered excellent [25]. The trade-off most probably occurs since increasing the TH_R and TH_D creates a larger set of possible predictions after the threshold stage, which increases the average maximal AECMOS value of these predictions.

Table II addresses segment only with echo-path changes. The trade-off described above remains, but with a consistent reduction in the average AECMOS values across all $(\text{TH}_\text{R}, \text{TH}_\text{D})$ pairs. This is associated with the linear AEC stage struggle with tracking and modeling linear echo in changing echo-path scenarios, which affects the average performance of the successive RES system [22]. Thus, the output of the URES pipeline that relies on the predictions of the RES system instances degrades in its overall subjective evaluation of speech quality that the AECMOS quantifies.

TABLE I: The effect of tolerance threshold values on the URES performance for segments with no echo-path change.

	$\text{TH}_\text{R} = 1$ [dB]			$\text{TH}_\text{R} = 2$ [dB]			$\text{TH}_\text{R} = 3$ [dB]		
	Δ_R [dB]	Δ_D [dB]	AECMOS	Δ_R [dB]	Δ_D [dB]	AECMOS	Δ_R [dB]	Δ_D [dB]	AECMOS
$\text{TH}_\text{D} = 1$ [dB]	0.4 ± 0.3	0.55 ± 0.25	3.1 ± 0.3	1.15 ± 0.45	0.6 ± 0.15	3.35 ± 0.3	1.75 ± 0.65	0.7 ± 0.15	3.5 ± 0.5
$\text{TH}_\text{D} = 2$ [dB]	0.55 ± 0.25	1.3 ± 0.2	3.45 ± 0.4	1.25 ± 0.45	1.45 ± 0.3	3.6 ± 0.4	1.85 ± 0.6	1.55 ± 0.25	4.0 ± 0.3
$\text{TH}_\text{D} = 3$ [dB]	0.65 ± 0.25	1.9 ± 0.2	3.7 ± 0.5	1.3 ± 0.4	2.05 ± 0.3	4.2 ± 0.5	1.95 ± 0.65	2.1 ± 0.3	4.4 ± 0.2

	$\text{TH}_\text{R} = 1$ [dB]			$\text{TH}_\text{R} = 2$ [dB]			$\text{TH}_\text{R} = 3$ [dB]		
	ERLE [dB]	PESQ	DNSMOS	ERLE [dB]	PESQ	DNSMOS	ERLE [dB]	PESQ	DNSMOS
$\text{TH}_\text{D} = 1$ [dB]	11.6 ± 1.3	2.9 ± 0.3	2.95 ± 0.3	14.2 ± 1.35	3.0 ± 0.25	3.0 ± 0.35	16.7 ± 1.55	3.25 ± 0.25	3.2 ± 0.55
$\text{TH}_\text{D} = 2$ [dB]	13.5 ± 1.45	3.05 ± 0.3	3.1 ± 0.45	16.4 ± 1.6	3.1 ± 0.35	3.35 ± 0.5	18.3 ± 1.75	3.4 ± 0.35	3.65 ± 0.5
$\text{TH}_\text{D} = 3$ [dB]	15.7 ± 1.85	3.3 ± 0.4	3.55 ± 0.5	17.8 ± 1.95	3.45 ± 0.4	3.8 ± 0.5	19.5 ± 2.05	3.65 ± 0.35	4.05 ± 0.4

TABLE II: The effect of tolerance threshold values on the URES performance for segments with echo-path change.

	$\text{TH}_\text{R} = 1$ [dB]			$\text{TH}_\text{R} = 2$ [dB]			$\text{TH}_\text{R} = 3$ [dB]		
	Δ_R [dB]	Δ_D [dB]	AECMOS	Δ_R [dB]	Δ_D [dB]	AECMOS	Δ_R [dB]	Δ_D [dB]	AECMOS
$\text{TH}_\text{D} = 1$ [dB]	0.5 ± 0.25	0.65 ± 0.2	2.95 ± 0.3	1.25 ± 0.4	0.65 ± 0.2	3.05 ± 0.4	1.85 ± 0.65	0.75 ± 0.2	3.35 ± 0.5
$\text{TH}_\text{D} = 2$ [dB]	0.65 ± 0.35	1.45 ± 0.3	3.2 ± 0.45	1.3 ± 0.45	1.55 ± 0.3	3.3 ± 0.5	1.9 ± 0.6	1.65 ± 0.2	3.7 ± 0.3
$\text{TH}_\text{D} = 3$ [dB]	0.7 ± 0.1	2.05 ± 0.45	3.5 ± 0.6	1.45 ± 0.45	2.2 ± 0.3	3.8 ± 0.5	2.05 ± 0.6	2.2 ± 0.35	3.9 ± 0.3

	$\text{TH}_\text{R} = 1$ [dB]			$\text{TH}_\text{R} = 2$ [dB]			$\text{TH}_\text{R} = 3$ [dB]		
	ERLE [dB]	PESQ	DNSMOS	ERLE [dB]	PESQ	DNSMOS	ERLE [dB]	PESQ	DNSMOS
$\text{TH}_\text{D} = 1$ [dB]	11.1 ± 1.45	2.9 ± 0.3	2.95 ± 0.3	13.6 ± 1.4	3.0 ± 0.25	3.05 ± 0.35	15.4 ± 1.75	3.25 ± 0.25	3.2 ± 0.55
$\text{TH}_\text{D} = 2$ [dB]	12.9 ± 1.65	3.05 ± 0.3	3.1 ± 0.45	15.3 ± 1.7	3.1 ± 0.35	3.35 ± 0.5	17.5 ± 2.0	3.4 ± 0.35	3.65 ± 0.5
$\text{TH}_\text{D} = 3$ [dB]	14.9 ± 2.0	3.3 ± 0.4	3.55 ± 0.5	16.9 ± 2.1	3.45 ± 0.5	3.8 ± 0.5	18.1 ± 2.45	3.65 ± 0.35	4.05 ± 0.4

Interestingly, results are consistently not symmetric in both tables. E.g., $(\text{TH}_\text{R}, \text{TH}_\text{D}) = (2, 3)$ in dB and $(\text{TH}_\text{R}, \text{TH}_\text{D}) = (3, 2)$ in dB have respective average AECMOS values of 4.2 and 4 in Table I. Namely, having a more extensive range for the DSML to deviate from the UOP, i.e., controlling more of the speech distortion rate, enhances the average AECMOS more than symmetrically applying this logic to the RESL. An interesting future research may involve investigating the possible inherent bias of the AECMOS towards more echo suppression over speech distortion, whether during the human subjective evaluation or in the following automation of it into an objective measure. These tables also give an intuition of how the objective Δ_R and Δ_D empirically relate to the subjective human rating prediction in the AECMOS. Therefore, relying on Tables I and II may allow an educated choice by the user regarding TH_R and TH_D.

It is highlighted that while an estimation error, as discussed in subsection V-A of 1 dB, for instance, may cause uncertainty and bias in the results, the human perception of 1 dB deviation from the UOP tends to be imperceptible [41]. Overall, the URES framework can enable a deviation from the UOP that is subjectively low-perceived [41] along with a subjectively excellent AECMOS, on average, in various acoustic scenarios.

Complementary evaluation metrics in the DNSMOS, PESQ, and ERLE are also evaluated in no echo-path change scenarios in Table I and with echo path-change scenarios in Table II. In both scenarios, all metrics follow the pattern of presenting improved performance as TH_R and TH_D increase. Specifically, the DNSMOS is correlated with the AECMOS in both Table I and Table II, a property that has been previously presented by the authors outside of the context of the URES framework [23], [24]. However, the DNSMOS values are consistently lower on average than the AECMOS values. This may occur because the URES framework has not been optimized to remove noise, while the DNSMOS has been optimized to predict how humans would judge speech quality degradation from noise.

The PESQ scores are also correlated with and are consistently lower than the AECMOS values in both Table I and Table II. Even though the PESQ metric is not as comparable to

the AECMOS as the DNSMOS, the PESQ values still provide a supportive indication of how speech quality may be perceived in the outcome of the URES framework. For the ERLE, given a $(\mathrm{TH_R}, \mathrm{TH_D})$ pair, better performance is achieved when $\mathrm{TH_R} > \mathrm{TH_D}$ than the opposite in both Table I and Table II. This might be observed because as $\mathrm{TH_R}$ increases and $\mathrm{TH_D}$ remains fixed, the AECMOS might achieve maximization by considering wider deviations of RESL values from the UOP. The larger the RESL, the more residual echo suppression has been achieved by the URES system, which is assumed to be correlated with larger ERLE values since the latter measures residual echo loss by the URES system.

C. The effect of the tolerance threshold values on P

This experiment includes scenarios with and without echo-path changes. It reports the average P value for every $(\mathrm{TH_R}, \mathrm{TH_D})$ pair that confines to $\mathrm{TH_R} \in \{1, 2, 3, 4, 5\}$ in dB and $\mathrm{TH_D} \in \{1, 2, 3, 4, 5\}$ in dB, which totals to 25 pairs combinations. By observing Fig. 3, P increases as the tolerance threshold values increase, and vice versa. This is expected since the construction of the URES framework ensures that, on average, the higher $\mathrm{TH_R}$ and $\mathrm{TH_D}$ become, the larger amount of RES predictions are available to undergo AECMOS maximization after the threshold stage, namely P increases, and vice versa.

An important case is where $(\mathrm{TH_R}, \mathrm{TH_D}) = (1, 1)$ in dB, which averages approximately $P = 2$. This indicates that these tolerance threshold values are the lowest valid for the URES framework. A deeper dive reveals that $P = 0$ did not occur for this scenario, and $P = 1$ was reported 17% of the time. On the other hand, $(\mathrm{TH_R}, \mathrm{TH_D}) = (5, 5)$, in dB, achieve an average of $P > 60$. Another observation is the proximity between the results with and without echo-path changes. Namely, even though in Tables I and II the presence of echo-path changes degraded the average AECMOS, it does not narrow the number of possible predictions that arrive at the AECMOS maximization stage. Conclusively, the URES framework supports even very narrow margins of 1 dB from the UOP. However, lightly relaxing this constraint enlarges P significantly, which increases the AECMOS, on average, as supported in Tables I and II and in subsection V-B.

D. The effect of echo and noise levels on performance

We recognize that the dynamic environment of hands-free speech communication exhibits various levels of echo and noise. Considering only segments with no echo-path changes and focusing on a tolerance threshold pair of $(\mathrm{TH_R}, \mathrm{TH_D}) = (2, 2)$ in dB, we report the average performance of the URES framework for SER levels from the set $\{-20, -10, 0, 10\}$ dB and for SNR levels from the set $\{0, 10, 20, 30, 40\}$ dB. It can be shown in Fig. 4 that in severe acoustic setups of -20 dB SER or of 0 dB SNR, the URES framework achieves average AECMOS values close to 3. In contrast, friendly acoustics of 20 dB SER or 40 dB SNR allow an average AECMOS that approaches 4 or even exceeds it.

In degraded acoustic conditions, both the lowest average AECMOS and the most significant average deviations from the

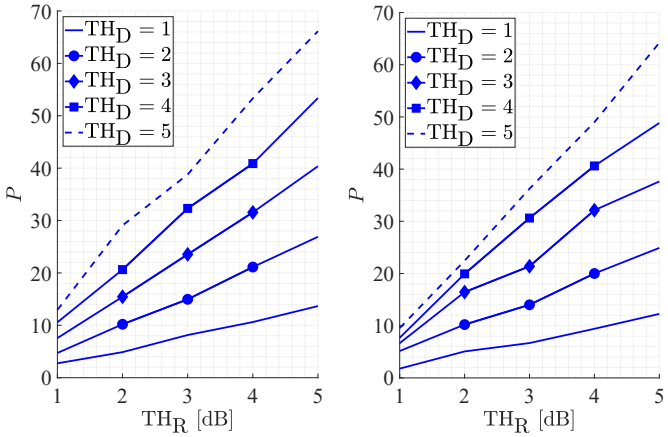

Fig. 3: Average P values for various $(\mathrm{TH_R}, \mathrm{TH_D})$ pairs for scenarios with no echo-path change (left) and with echo-path change (right). The units of $\mathrm{TH_D}$ values in the legend are dB.

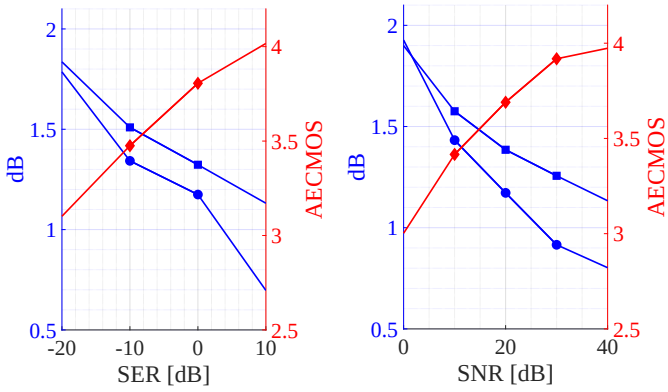

Fig. 4: Average values of the AECMOS (diamonds), Δ_R in dB (circles) and Δ_D in dB (squares) for various levels of SER (left) and SNR (right) values with no echo-path change scenarios and $(\mathrm{TH_R}, \mathrm{TH_D}) = (2, 2)$ in dB.

UOP occur. One assumption is that in conditions of high echo and noise levels, the subjective quality rating is maximized when the RESL and DSML are taken to their allowed extreme to suppress most echo and distort the minor speech possible. Another observation is that the Δ_D is almost consistently higher on average than Δ_R across all SER and SNR levels.

In summary, challenging but practical conditions, e.g., SER $= 0$ dB and SNR $= 20$ dB, are handled well by the URES system, which allows a broad support of this framework in various acoustic environments.

E. The effect of the number of RES instances on performance

The URES system initially employs 101 pre-trained RES model instances, where every instance corresponds to an α value between 0 and 1 with 0.01 increments. In this experiment, we examine how lowering the computational load by

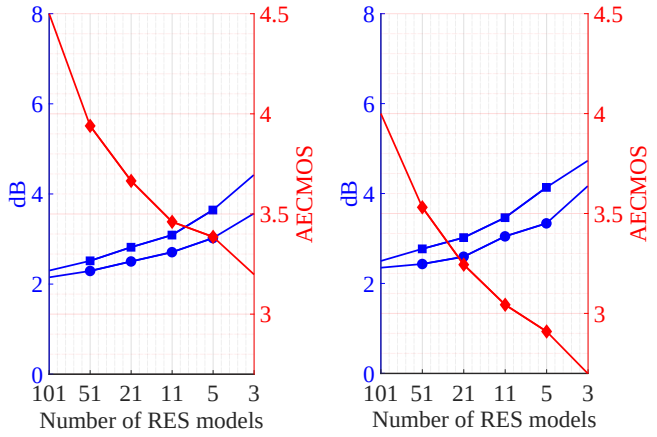

Fig. 5: Average values of the AECMOS (diamonds), Δ_{R} in dB (circles) and Δ_{D} in dB (squares) versus number of trained RES model instances in scenarios without (left) and with (right) echo-path changes, considering $(\mathrm{TH_R}, \mathrm{TH_D}) = (5, 5)$ in dB.

considering fewer RES model instances affects the URES performance. This is done by applying identical training and testing processes as for the original URES framework but with α increments now taken from the set $\{0.02, 0.05, 0.1, 0.25, 0.5\}$. In correspondence, the number of RES model instances examined is the set $\{51, 21, 11, 5, 3\}$, where, for example, taking an increment of 0.25 includes $\alpha \in \{0, 0.25, 0.5, 0.75, 1\}$ and an increment of 0.5 has $\alpha \in \{0, 0.5, 1\}$. The number of RES and RDE model instances is identical, preserving the framework's functionality.

Across all increments, we fix the tolerance threshold pairs to $(\mathrm{TH_R}, \mathrm{TH_D}) = (5, 5)$ in dB. The motivation for this choice relates to how using fewer RES model instances, i.e., larger α increments, intrinsically decreases the average value of P per $(\mathrm{TH_R}, \mathrm{TH_D})$ pair. We wish to mitigate this bias and isolate the effect of how the α increment changes the AECMOS in the URES output.

Based on Fig. 5, the average AECMOS degrades by more than 0.5 points when transitioning from 101 to 51 model instances. Narrowing down the number of instances even further lowers the average AECMOS to subjectively mediocre and below, reaching as low as 2.7 for scenarios with echo-path changes. The increase in the average Δ_{R} and average Δ_{D} values is also significant, almost doubling its size as the number of RES instances lowers from 101 to only 3. To summarize, employing the entire 101 RES model instances significantly impacts the URES framework performance, mainly in terms of the average AECMOS.

F. The effect of computational complexity on practicality

We recognize that the proposed URES framework introduces a high computational burden. So, this subsection is dedicated to resource analysis and discussion on the practicality of the framework. First, Table III reports the computational resources of the proposed system using three measures, i.e., the number of trainable parameters, floating-point operations [42] per 10 ms, and total memory required for both the instructions and the architecture [43]. For each of these measures, we report the resources of a single instance for each of the four components that compose the URES framework, i.e., the linear AEC filter and the deep RES, RDE, and AECMOS models. In addition, we regard the accumulation of these resources and analyze them for the entire URES framework both in the minimal case, i.e., when only one instance is considered per component, and in the maximal case, i.e., when 101 instances of the RES and RDE models are considered, along with 85 instances of the AECMOS. The number of 85 AECMOS instances has been chosen since the experiments we introduced in subsection V-C have revealed that the maximal value of $P(n)$ was 85 when $(\mathrm{TH_R}, \mathrm{TH_D}) = (5, 5)$ in dB. For clarity, it is mentioned that regardless of the number of deep models used, there is merely one linear AEC adaptive filter, which, instead of requiring memory for the architecture, requires memory for allocations.

From Table III, we focus on the most computationally-heavy scenario, where the URES framework requires 43.7×10^6 trainable parameters, 129×10^9 floating-point operations per 10 ms, and 2×10^9 bytes of total memory. Even in this case, we illustrate the practicality of the proposed framework to perform on-edge using existing hardware by taking as an example the NVIDIA Jetson Xavier NX system-on-module (SoM) [44] that is dedicated to neural speech processing. We first note that this SoM has both 8×10^9 bytes and 16×10^9 bytes versions available, which are sufficient for the instructions and the architecture memory needed by the URES framework. Second, this SoM allows for 12.6×10^{12} floating-point operations per second (FLOPs) for 16-bit precision in floating-point format [45], which is the case in our calculations.

We now regard the inference times of the URES framework both on standard processing hardware, e.g., the 11th Gen Intel Core™ i7-11850H @ 2.50 GHz processor, and on the dedicated hardware, taken as the SoM above. As detailed in Appendix B-B, the analysis of the frame size equals $M = 20$ ms and the step-size equals 10 ms. We initially lay out only the buffering latency that every 20 ms analysis frame undergoes during the URES pipeline, from the linear AEC filter's input to the URES framework's output. In the first stage, the linear AEC system inserts a negligible delay by computation. Still, it does accumulate 8 ms of latency, which is the needed time to align the inputs and the outcomes of the linear AEC filter before inserting those into the URES pipeline in a synced manner. This delay is not affected by the type of hardware. In the second stage, every RES model instance requires its input to undergo STFT, RES inference, and inverse STFT. This STFT-related delay is also not affected by the type of hardware. It causes algorithmic latency of 10 ms since we use the overlap-save method that does not introduce additional algorithmic latency [37]. Overall, every 20 ms frame undergoes an algorithmic delay of 18 ms excluding the inference time by the URES system components.

To calculate the inference time, we first notice that the functionality of the URES framework dictates that the outcomes of all RDE instances are aggregated before the AEC-

TABLE III: Computational complexity of the URES framework and a single instance of all its four components.

	Linear AEC filter	RES model (one instance)	RDE model (one instance)	AECMOS model (one instance)	URES framework (minimal compute)	URES framework (maximal compute)
Number of parameters	2400	136×10^3	45×10^3	300×10^3	483.4×10^3	43.7×10^6
Floating-point operations per 10 ms	720×10^3	92×10^6	8×10^6	1400×10^6	1500×10^6	129×10^9
Memory in bytes	115×10^3	10.6×10^6	2.2×10^6	9.3×10^6	22×10^6	2×10^9

MOS layer can perform. Thus, we divide the calculation to two; the inference time by the RES and the RDE instances, and the inference time by the AECMOS. In this calculation, the inference time by the linear AEC filter is negligible and is not regarded. Turning to Table III, the floating-point operations needed for inference of every 20 ms frame by the 101 instances of the RES and RDE models respectively equal $101 \times 92 \times 10^6 = 9.292 \times 10^9$ floating-point operations and $101 \times 8 \times 10^6 = 0.808 \times 10^9$ floating-point operations. Due to the step-size of 10 ms we use, each frame undergoes inference $1000/10 = 100$ times per second. Thus, the RES and RDE layers require $100 \times (9.292 + 0.808) \times 10^9 = 1.01 \times 10^{12}$ FLOPs. By assuming a theoretical capability of the standard processor that performs inference with 100% efficiency, we may utilize all the 2.457×10^{12} FLOPs of the processor and contain this calculation using parallel computing [46].

Next, we need to consider the AECMOS layer that accumulates $85 \times 1400 \times 10^6 = 119 \times 10^9$ floating-point operations, and thus $100 \times 119 \times 10^9 = 11.9 \times 10^{12}$ FLOPs, which cannot be contained by the standard hardware using parallel computing. Overall, every 1 s of input frames take $(11.9 + 1.01) \times 10^{12}/2.457 \times 10^{12} = 5.25$ s to be inferred by the URES framework on a standard processor. Meaning, every 20 ms input frame takes a total latency, including buffering time and inference time, of $8 + 10 + 20 \times 5.25 = 123$ ms. The RTF [28], [47] of the URES framework is the ratio between the actual time necessary for all the computations in the framework to infer the 20 ms input frame, and the duration of the input frame to process, i.e., 20 ms. Using a standard processor, the RTF equals $123/20 = 6.15$ by assuming 100% processor efficiency. Of course, in realistic scenarios where this standard processor is for general purpose, its efficiency can go as low as 10%, which can dramatically increase the inference time and RTF. An interesting complementary view of the RTF focuses only on the inference time by the models, and excludes buffering and algorithmic delays. In that case, the inference time is $20 \times 5.25 = 105$ ms and the RTF equals 5.25. Either way, the standard processor cannot offer real-time capabilities to run the URES framework.

We now examine the dedicated hardware in the NVIDIA Jetson Xavier NX SoM, which is able to perform 12.6×10^{12} FLOPs. Using the same type of inference calculations as before, this means that every 1 s of input frames takes $(11.9 + 1.01) \times 10^{12}/12.6 \times 10^{12} = 1.02$ s to be inferred by the URES framework. Meaning, every 20 ms input frame takes a total latency, including buffering time and inference time, of $8 + 10 + 20 \times 1.02 = 38.4$ ms, which is less than 40 ms and meets the standard timing requirements of hands-free communication [48]. The RTF equals $38.4/20 = 1.92$. If we consider merely the inference time by adopting the discussed complementary view, then the inference time is $20 \times 1.02 = 20.04$ ms and the RTF equals 1.02. It is important to consider a complementary view of real-time, where the overall processing time of every 20 ms frame does not exceed the frame shift time of 10 ms [49]. The URES framework does not meet this real-time criteria, even on dedicated hardware.

To recap, the URES framework can produce, on average, speech quality that is subjectively estimated as excellent while also confining to the UOP by roughly 2 dB deviation, while allowing UOP adjustments in less than 40 ms with an RTF of 1.92 given the availability of dedicated on-edge hardware. However, these capabilities come at the expense of an immensely high computational burden that can be contained today only by specifically dedicated hardware, which limits the general availability of the URES framework to the typical user and preserves it primarily for high-end users and customers.

VI. Conclusions

RES in double-talk periods is an integral requirement of many hands-free speech communication systems, and recent RES methods have shown impressive advancements in average benchmark performance. However, existing studies do not support specific user inputs, which has crucial practical and commercial implications. In this work, we developed a user-centric framework for RES in double-talk, which introduces three attributes that aim to enhance user experience. First, the RESL and DSML of the RES output are confined to a UOP up to a given tolerance threshold. Second, our framework supports tracking of changes in the UOP with less than 40 ms and with RTF of 1.92, which is essential in a dynamic acoustic environment of rapidly varying user preferences from a wide spectrum. Third, AECMOS maximization is applied to enhance the subjective speech quality of the output signal. However, the developed framework demands immense computational resources, which practically limit it to a specific market share of high-end users and customers. Future work may involve a learning framework that maps acoustic information to UOP recommendations in real-time, an extension of the objective function in (15) that aims to optimize its trade-off functionality between desired-speech distortion and residual-echo suppression levels, and a release of a lean version to the URES framework that enables the URES framework to run on standard hardware.

APPENDIX A
DEEP MODEL ARCHITECTURES

A. The deep RES architecture

For the deep RES architecture, we employ the following layers: Conv2D for two-dimensional convolution [50], MaxPooling2D to calculate the maximal patch value [51], BatchNorm2D for two-dimensional batch normalization [52], Upsampling [53], and the ReLU activation function [54]. The traditional regularizing dropout [55] layer is replaced with BatchNorm2D. In Table IV, the DoubleConv unit is described, which is the core of the RES architecture. DoubleConv(I, O, M) receives I input channels, O output channels, and M middle channels. Table V details the RES architecture. Subscripts 'c' and 'k' denote the number of channels and kernel size. For example, the first layer in Table IV is a Conv2D layer with I input channels and I output channels that employs a 3×3 kernel.

TABLE IV: The DoubleConv(I, O, M) unit

Conv2D: $I_c \rightarrow I_c$, $(3 \times 3)_k$
Conv2D: $I_c \rightarrow M_c$, $(3 \times 3)_k$
BatchNorm2D: $M_c \rightarrow M_c$
ReLU: $M_c \rightarrow M_c$
Conv2D: $M_c \rightarrow M_c$, $(3 \times 3)_k$
Conv2D: $M_c \rightarrow O_c$, $(3 \times 3)_k$
BatchNorm2D: O_c
ReLU: $O_c \rightarrow O_c$

TABLE V: The deep RES architecture

Layer Description	Output Dimensions
Input: $(2, 30, 161)$	
DoubleConv $(2, 16, 16)$	$(16, 30, 161)$
MaxPooling2D: $(2 \times 2)_k$	$(16, 15, 80)$
DoubleConv $(16, 32, 32)$	$(32, 15, 80)$
MaxPooling2D: $(2 \times 2)_k$	$(32, 7, 40)$
DoubleConv $(32, 64, 64)$	$(64, 7, 40)$
MaxPooling2D: $(2 \times 2)_k$	$(64, 3, 20)$
DoubleConv $(64, 128, 128)$	$(128, 3, 20)$
MaxPooling2D: $(2 \times 2)_k$	$(128, 1, 10)$
DoubleConv $(128, 128, 128)$	$(128, 1, 10)$
UpSampling: scale factor 2	$(128, 2, 20)$
DoubleConv $(256, 64, 128)$	$(64, 3, 20)$
UpSampling: scale factor 2	$(64, 6, 40)$
DoubleConv $(128, 32, 64)$	$(32, 7, 40)$
UpSampling: scale factor 2	$(32, 14, 80)$
DoubleConv $(64, 16, 32)$	$(16, 15, 80)$
UpSampling: scale factor 2	$(16, 30, 160)$
DoubleConv $(32, 16, 16)$	$(16, 30, 161)$
Conv2D: $16_c \rightarrow 1_c$, $(1 \times 1)_k$	$(1, 30, 161)$

B. Deep RDE architecture

For the deep RDE architecture that operates in the waveform domain, we utilize the long short-term memory (LSTM) [56] neural network. The architecture also employs the Flatten layer [57] and the fully-connected linear layer [58], in addition to the ReLU activation function. Following Pytorch convention [59], the LSTM (N, L, H) layer receives batch size of N, sequence length of L, and input size of H.

TABLE VI: The deep RDE architecture

Layer Description	Output Dimensions
Input: $(320, 5)$	
LSTM $(5, 20, 10)$	$(320, 20)$
Flatten	$(6400, 1)$
Linear: $6400_c \rightarrow 2_c$	$(2, 1)$
ReLU: $2_c \rightarrow 2_c$	$(2, 1)$

APPENDIX B
EXPERIMENTAL SETTINGS

A. Database acquisition from independent recordings

Mouth simulator	4227-A^{TM}, Brüel&Kjaer
Loudspeaker	Z120^{TM}, Logitech
Microphone	MT503^{TM}, Spider
Mouth-simulator-to-mic distance	1m, 1.5m, 2m
Loudspeaker-to-mic distance	1m, 1.5m, 2m
Number of rooms	4
Smallest room size	$3 \times 3 \times 2.5$ m^3
Largest room size	$5 \times 5 \times 4$ m^3
Range of RT$_{60}$ [60]	$0.3 - 0.6$ s
Sampling frequency	16×10^3 Hz

B. Preprocessing, training, and inference parameters

Sampling frequency	16×10^3 Hz
Bits precision	16-bit floating-point
L (time, samples)	RT$_{60}$ s, RT$_{60} \times 16 \times 10^3$ samples
M (time, samples)	20 ms, 320 samples
N (time, samples)	15 s, 240×10^3 samples
Step-size time, samples	10 ms, 160 samples
RES past frames	29
RES past frames indices	$1 - 29$
RES learning rate	0.0005
RES mini-batch size	4
RES epochs	10
RES optimizer	Adam [61]
RES training duration	8 minutes / epoch
RDE batch size	5
RDE learning rate	0.001
RDE mini-batch size	4
RDE epochs	10
RDE optimizer	Adam
RDE training duration	12 minutes / epoch

REFERENCES

[1] M. Schmidtner, C. Doering, and H. Timinger, "Agile working during COVID-19 pandemic," *IEEE Engineering Management Review*, vol. 49, no. 2, pp. 18–32, 2021.

[2] K. Sridhar, R. Cutler, A. Saabas, T. Parnamaa, M. Loide, H. Gamper, *et al.*, "ICASSP 2021 acoustic echo cancellation challenge: Datasets, testing framework, and results," in *Proc. ICASSP*. IEEE, 2021, pp. 151–155.

[3] J. Benesty, T. Gänsler, D. R. Morgan, M. M. Sondhi, S. L. Gay, *et al.*, *Advances in network and acoustic echo cancellation*. New York: Springer, 2001.

[4] M. M. Sondhi, D. R. Morgan, and J. L. Hall, "Stereophonic acoustic echo cancellation-an overview of the fundamental problem," *IEEE Signal Processing Letters*, vol. 2, no. 8, pp. 148–151, 1995.

[5] S. H. Pauline, D. Samiappan, R. Kumar, A. Anand, and A. Kar, "Variable tap-length non-parametric variable step-size NLMS adaptive filtering algorithm for acoustic echo cancellation," *Applied Acoustics*, vol. 159, p. 107074, 2020.

[6] A. Ivry, I. Cohen, and B. Berdugo, "Deep adaptation control for acoustic echo cancellation," in *Proc. ICASSP*. IEEE, 2022, pp. 741–745.

[7] M. Salah, M. Dessouky, and B. Abdelhamid, "Design and implementation of an improved variable step-size NLMS-based algorithm for acoustic noise cancellation," *Circuits, Systems, and Signal Processing*, vol. 41, no. 1, pp. 551–578, 2022.

[8] H. Zhao, Y. Gao, and Y. Zhu, "Robust subband adaptive filter algorithms-based mixture correntropy and application to acoustic echo cancellation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 31, pp. 1223–1233, 2023.

[9] Y. Yu, T. Yang, H. Chen, R. C. de Lamare, and Y. Li, "Sparsity-aware SSAF algorithm with individual weighting factors: Performance analysis and improvements in acoustic echo cancellation," *Signal Processing*, vol. 178, p. 107806, 2021.

[10] G. Imen, A. Benallal, M. Mekarzia, and I. Hassani, "The NP-VSS NLMS algorithm with noise power estimation methods for acoustic echo cancellation," in *Proc. International Conference on Advanced Electrical Engineering (ICAEE)*. IEEE, 2022, pp. 1–6.

[11] A. Ivry, I. Cohen, and B. Berdugo, "Nonlinear acoustic echo cancellation with deep learning," in *Proc. Interspeech*, 2021, pp. 4773–4777.

[12] S. Zhang, Z. Wang, J. Sun, Y. Fu, B. Tian, Q. Fu, and L. Xie, "Multi-task deep residual echo suppression with echo-aware loss," in *Proc. ICASSP*. IEEE, 2022, pp. 9127–9131.

[13] J. Franzen and T. Fingscheidt, "Deep residual echo suppression and noise reduction: A multi-input FCRN approach in a hybrid speech enhancement system," in *Proc. ICASSP*. IEEE, 2022, pp. 666–670.

[14] N. K. Desiraju, S. Doclo, M. Buck, and T. Wolff, "Joint online estimation of early and late residual echo PSD for residual echo suppression," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 31, pp. 333–344, 2022.

[15] K. Xie, Z. Yang, and J. Chen, "Nonlinear residual echo suppression based on gated dual signal transformation LSTM network," in *Proc. Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2022, pp. 1696–1701.

[16] C. M. Lee, J. W. Shin, and N. S. Kim, "DNN-based residual echo suppression," in *Proc. Sixteenth Annual Conference of the International Speech Communication Association*, 2015, pp. 1775–1779.

[17] G. Carbajal, R. Serizel, E. Vincent, and E. Humbert, "Multiple-input neural network-based residual echo suppression," in *Proc. ICASSP*. IEEE, 2018, pp. 231–235.

[18] H. Zhang, K. Tan, and D. Wang, "Deep learning for joint acoustic echo and noise cancellation with nonlinear distortions." in *Proc. Interspeech*, 2019, pp. 4255–4259.

[19] L. Pfeifenberger and F. Pernkopf, "Nonlinear residual echo suppression using a recurrent neural network." in *Proc. Interspeech*, 2020, pp. 3950–3954.

[20] X. Zhou and Y. Leng, "Residual acoustic echo suppression based on efficient multi-task convolutional neural network," *preprint arXiv:2009.13931*, 2020.

[21] A. Ivry, I. Cohen, and B. Berdugo, "Off-the-shelf deep integration for residual-echo suppression," in *Proc. ICASSP*. IEEE, 2022, pp. 746–750.

[22] ——, "Deep residual echo suppression with a tunable tradeoff between signal distortion and echo suppression," in *Proc. ICASSP*. IEEE, 2021, pp. 126–130.

[23] ——, "Objective metrics to evaluate residual-echo suppression during double-talk," in *Proc. WASPAA*. IEEE, 2021, pp. 101–105.

[24] ——, "Objective metrics to evaluate residual-echo suppression during double-talk in the stereophonic case," *Proc. Interspeech*, pp. 5348–5352, 2022.

[25] M. Purin, S. Sootla, M. Sponza, A. Saabas, and R. Cutler, "AECMOS: A speech quality assessment metric for echo impairment," in *Proc. ICASSP*. IEEE, 2022, pp. 901–905.

[26] E. Vincent, R. Gribonval, and C. Févotte, "Performance measurement in blind audio source separation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 4, pp. 1462–1469, 2006.

[27] 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 *Proc. ICASSP*, vol. 2. IEEE, 2001, pp. 749–752.

[28] M. Malik, M. K. Malik, K. Mehmood, and I. Makhdoom, "Automatic speech recognition: a survey," *Multimedia Tools and Applications*, vol. 80, pp. 9411–9457, 2021.

[29] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. Medical Image Computing and Computer-Assisted Intervention*. Springer, 2015, pp. 234–241.

[30] H. Zhivomirov, "On the development of STFT-analysis and ISTFT-synthesis routines and their practical implementation," *Technology, Education, Management, Informatics (TEM) Journal*, vol. 8, no. 1, pp. 56–64, 2019.

[31] R. Cutler, A. Saabas, T. Parnamaa, M. Purin, H. Gamper, S. Braun, *et al.*, "ICASSP 2022 acoustic echo cancellation challenge," in *Proc. ICASSP*. IEEE, 2022, pp. 9107–9111.

[32] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, and D. S. Pallett, "DARPA TIMIT acoustic-phonetic continous speech corpus CD-ROM. NIST speech disc 1-1.1," *NASA STI/Recon technical report*, vol. 93, p. 27403, 1993.

[33] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an ASR corpus based on public domain audio books," in *Proc. ICASSP*. IEEE, 2015, pp. 5206–5210.

[34] N. L. Freire and S. C. Douglas, "Adaptive cancellation of geomagnetic background noise using a sign-error normalized LMS algorithm," in *Proc. ICASSP*, vol. 3. IEEE, 1993, pp. 523–526.

[35] L. Dogariu, C. Paleologu, J. Benesty, and S. Ciochină, "An efficient kalman filter for the identification of low-rank systems," *Signal Processing*, vol. 166, p. 107239, 2020.

[36] I. Fîciu, J. Benesty, L. Dogariu, C. Paleologu, and S. Ciochină, "Efficient algorithms for linear system identification with particular symmetric filters," *Applied Sciences*, vol. 12, no. 9, p. 4263, 2022.

[37] S. Muramatsu and H. Kiya, "Extended overlap-add and-save methods for multirate signal processing," *IEEE Transactions on Signal Processing*, vol. 45, no. 9, pp. 2376–2380, 1997.

[38] L. Huang, J. Qin, Y. Zhou, F. Zhu, L. Liu, and L. Shao, "Normalization techniques in training dnns: Methodology, analysis and application," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[39] C. K. A. Reddy, V. Gopal, and R. Cutler, "DNSMOS P. 835: A non-intrusive perceptual objective speech quality metric to evaluate noise suppressors," in *Proc. ICASSP*. IEEE, 2022, pp. 886–890.

[40] J. D. Rodriguez, A. Perez, and J. A. Lozano, "Sensitivity analysis of k-fold cross validation in prediction error estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 3, pp. 569–575, 2009.

[41] W. A. Yost, "Fundamentals of hearing: An introduction," 2001.

[42] T. Nguyen, D. Hicks, D. Moss, *et al.*, "11 Tera-FLOP per second photonic convolutional accelerator for deep learning optical neural networks," *preprint arXiv:2011.07393*, 2020.

[43] K. J. Lee, "Architecture of neural processing unit for deep neural networks," in *Advances in Computers*. Elsevier, 2021, vol. 122, pp. 217–245.

[44] NVIDIA, "Jetson Xavier NX: The world's smallest AI supercomputer," https://developer.nvidia.com/blog/jetson-xavier-nx-the-worlds-smallest-ai-supercomputer/, 2023.

[45] A. Agrawal, S. M. Mueller, B. M. Fleischer, X. Sun, N. Wang, J. Choi, *et al.*, "DLFloat: A 16-b floating point format designed for deep learning training and inference," in *Proc. Symposium on Computer Arithmetic (ARITH)*. IEEE, 2019, pp. 92–95.

[46] T. Ben-Nun and T. Hoefler, "Demystifying parallel and distributed deep learning: An in-depth concurrency analysis," *preprint arXiv:1802.09941*, 2018.

[47] S. Gulzar Ahmad, H. Ullah Khan, S. Ijaz, and E. Ullah Munir, "Use case-based evaluation of workflow optimization strategy in real-time computation system," *The Journal of Supercomputing*, vol. 76, pp. 708–725, 2020.

[48] *ETSI ES 202 740: Speech and multimedia Transmission Quality (STQ); Transmission requirements for wideband VoIP loudspeaking and hands-free terminals from a QoS perspective as perceived by the user*, ETSI Std., 2016.

[49] R. Cutler, A. Saabas, T. Parnamaa, M. Purin, E. Indenbom, N. Ristea, *et al.*, "ICASSP 2023 acoustic echo cancellation challenge," in *Proc. ICASSP*, 2023, p. to appear.

[50] Y. Wu, F. Yang, Y. Liu, X. Zha, and S. Yuan, "A comparison of 1-D and 2-D deep convolutional neural networks in ECG classification," *preprint arXiv:1810.07088*, 2018.

[51] P. Zhou, Z. Qi, S. Zheng, J. Xu, H. Bao, and B. Xu, "Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling," *preprint arXiv:1611.06639*, 2016.

[52] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry, "How does batch normalization help optimization?" *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[53] J. Pons, S. Pascual, G. Cengarle, and J. Serrà, "Upsampling artifacts in neural audio synthesis," in *Proc. ICASSP*. IEEE, 2021, pp. 3005–3009.

[54] A. F. Agarap, "Deep learning using rectified linear units (ReLU)," *preprint arXiv:1803.08375*, 2018.

[55] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[56] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of recurrent neural networks: LSTM cells and network architectures," *Neural Computation*, vol. 31, no. 7, pp. 1235–1270, 2019.

[57] J. Jin, A. Dundar, and E. Culurciello, "Flattened convolutional neural networks for feedforward acceleration," *preprint arXiv:1412.5474*, 2014.

[58] A. G. Schwing and R. Urtasun, "Fully connected deep structured networks," *preprint arXiv:1503.02351*, 2015.

[59] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[60] M. R. Schroeder, "New method of measuring reverberation time," *The Journal of the Acoustical Society of America*, vol. 37, no. 6, pp. 1187–1188, 1965.

[61] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *preprint arXiv:1412.6980*, 2014.

Amir Ivry (M'24) earned both his B.Sc. and Ph.D. degrees (direct track) in Electrical and Computer Engineering from the Technion - Israel Institute of Technology, Haifa, Israel, completing the program in 2016 and 2023, respectively.

In 2022, he assumed the role of Principal Researcher at Microsoft, focusing on spoken language processing. This followed a tenure as a Research Scientist in the Prime Minister's Office from 2015 to 2021, and a research internship at Microsoft from 2021 to 2022.

Dr. Ivry has been recognized with the Jacobs Award twice and received the Outstanding Research Award in Data Science from the Israel Council for Higher Education. He served as the chief editor for the book "Deep Learning Interviews". His research interests span data-centric AI, automatic speech recognition, out-of-distribution language processing, spoken language understanding, analysis of machine learning and deep learning systems, and performance measurements.

In 2022, Dr. Ivry was listed in Forbes 30 under 30.

Israel Cohen (M'01-SM'03-F'15) received the B.Sc. (*Summa Cum Laude*), M.Sc. and Ph.D. degrees in electrical engineering from the Technion – Israel Institute of Technology, Haifa, Israel, in 1990, 1993 and 1998, respectively. He is the Louis and Samuel Seidan Professor in electrical and computer engineering at the Technion – Israel Institute of Technology.

From 1990 to 1998, he was a Research Scientist with RAFAEL Research Laboratories, Haifa, Israel Ministry of Defense. From 1998 to 2001, he was a Postdoctoral Research Associate with the Computer Science Department, Yale University, New Haven, CT, USA. In 2001 he joined the Electrical Engineering Department of the Technion. He is a coeditor of the Multichannel Speech Processing Section of the *Springer Handbook of Speech Processing* (Springer, 2008), and the coauthor of *Fundamentals of Signal Enhancement and Array Signal Processing* (Wiley-IEEE Press, 2018). His research interests are array processing, statistical signal processing, deep learning, analysis and modeling of acoustic signals, speech enhancement, noise estimation, microphone arrays, source localization, blind source separation, system identification and adaptive filtering.

Dr. Cohen was awarded an Honorary Doctorate from Karunya Institute of Technology and Sciences, Coimbatore, India (2023), the Norman Seiden Prize for Academic Excellence (2017), the SPS Signal Processing Letters Best Paper Award (2014), the Alexander Goldberg Prize for Excellence in Research (2010), and the Muriel and David Jacknow Award for Excellence in Teaching (2009). He was as Associate Editor of the IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING and IEEE SIGNAL PROCESSING LETTERS, a Member of the IEEE Audio and Acoustic Signal Processing Technical Committee and the IEEE Speech and Language Processing Technical Committee, and a Distinguished Lecturer of the IEEE Signal Processing Society.

Baruch Berdugo received the B.Sc. (Cum Laude) and M.Sc. degrees in electrical engineering, and Ph.D. degree in biomedical engineering from the Technion - Israel Institute of Technology, Haifa, Israel, in 1978, 1987 and 2001, respectively.

He is currently a Research Associate in electrical engineering at the Technion - Israel Institute of Technology. From 1982 to 2001, he was a Research Scientist with RAFAEL Research Laboratories, Haifa, Israel Ministry of Defense. From 2001 to 2003, he was the General Manger of Lamar Signal Processing, Yoqneam Ilit, Israel. From 2004 to 2018, he was the General Manager of MRD technologies, Kibbutz Usha, Israel, and the CTO of Phoenix Audio Technologies, Irvine, CA, USA.

His research interests are array processing, statistical signal processing, deep learning, analysis, speech enhancement, noise estimation, microphone arrays, source localization, and adaptive filtering.