



FORUM ACUSTICUM EURONOISE 2025

LATENT FXLMS: ACCELERATING ACTIVE NOISE CONTROL WITH NEURAL ADAPTIVE FILTERS

Kanad Sarkar^{1*} Austin Lu¹ Manan Mittal²
 Yongjie Zhuang^{2,5} Ryan M. Corey^{3,4} Andrew C. Singer^{1,2}

¹ Electrical and Computer Engineering, University of Illinois Urbana-Champaign, USA
² Electrical and Computer Engineering, Stony Brook University, USA
³ Electrical and Computer Engineering, University of Illinois Chicago, USA
⁴ Discovery Partners Institute, University of Illinois System, USA
⁵ Amazon Web Services[†]

ABSTRACT

Filtered-X LMS (FxLMS) is commonly used for active noise control (ANC), wherein the soundfield is minimized at a desired location. Given prior knowledge of the spatial region of the noise or control sources, we could improve FxLMS by adapting along the low-dimensional manifold of possible adaptive filter weights. We train an auto-encoder on the filter coefficients of the steady-state adaptive filter for each primary source location sampled from a given spatial region and constrain the weights of the adaptive filter to be the output of the decoder for a given state of latent variables. Then, we perform updates in the latent space and use the decoder to generate the cancellation filter. We evaluate how various neural network constraints and normalization techniques impact the convergence speed and steady-state mean squared error. Under certain conditions, our Latent FxLMS model converges in fewer steps with comparable steady-state error to the standard FxLMS.

Keywords: *adaptive filters, active noise control, neural networks, impulse response models*

*Corresponding author: kanads2@illinois.edu

[†]This work does not relate to the author's position at Amazon.

Copyright: ©2025 Sarkar et al. This is an open-access article distributed under the terms of the Creative Commons Attribution 3.0 Unported License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

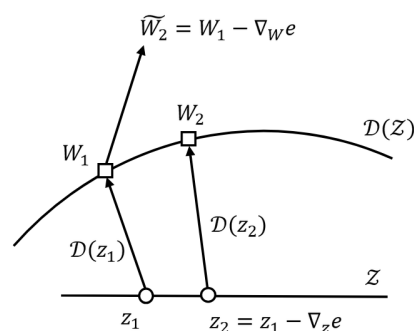


Figure 1. Diagram depicting the core element of Latent FxLMS. Instead of adapting the weights directly, a low-dimensional representation is updated based on the error gradient. The new weights are determined by the decoder's output at the updated latent representation. This method allows for faster convergence by constraining the updated weights to lie on the manifold of learned filters.

1. INTRODUCTION

Active Noise Control (ANC) uses loudspeakers to suppress the effect of an undesirable noise source for a listener at a given location. A standard ANC algorithm is the Filtered-x Least Mean Squares (FxLMS) algorithm, which adaptively filters a reference measurement of the noise signal to form a cancellation signal output by the control speakers. This is an effort to minimize the energy from the noise source at the desired location, where



FORUM ACUSTICUM EURONOISE 2025

an error microphone is ideally placed. The “Filtered- x ” component filters the reference signal with the impulse response between the control speakers and the error microphone, which is then used in the adaptive filter update. A review of ANC methods can be found in [1].

While it is well known that the filter adapts to time-varying noise statistics [1], it can also adapt to time-varying noise source positions when the noise source is farther from the error microphone than the control speaker¹. As the noise source moves within a spatial region, the adaptive filter will change within a subset of \mathbb{R}^L , where L is the filter length. When a manifold of a lower dimension than L can approximate the space of converged adaptive filters, a low-dimensional representation of this space can be exploited to improve the convergence of the filter adaptation. When the possible weights are constrained to a known Riemannian manifold, it can be shown that geometrically constrained adaptive filtering algorithms converge faster to a lower steady-state error than the standard algorithms [2]. However, obtaining a closed-form representation of the motion-based manifold of filter coefficients is difficult.

The proposed Latent FxLMS (LFxLMS) attempts to spatially constrain the adaptive ANC filter by combining the FxLMS algorithm and recent work on neural-network-based system identification (NNSI). The NNSI framework constrains the FIR filter to be the output of a decoder of a pre-trained auto-encoder, and adaptations are performed within the latent space of the decoder. Previous work on acoustic NNSI uses various approaches to track time-varying impulse responses [3, 4]. A topologically aware variational auto-encoder was used to speed up the convergence of RLS [3]. Furthermore, the authors in [4] applied this framework to an extended Kalman filter to improve acoustic echo cancellation performance. The model developed in [5] is similar to our work in that it uses a neural network to provide gradient updates to an adaptive filter in a real-time acoustic system. However, [5] achieves this through a meta-learning framework, where their model output gives the weight gradient, while NNSI provides a gradient in the latent space. This paper also presents novel methods for normalized latent updates and examines their impact on adaptive filter convergence.

The LFxLMS algorithm involves training an auto-

¹ If the primary source is closer to the microphone compared to the secondary source, knowledge of future samples of the noise signal is required to cancel sound at the microphone. This goes against the requirement that the adaptive filter is causal.

encoder on a dataset of converged FxLMS adaptive filters, each corresponding to a different noise source location sampled from a region in space. A wide variety of existing literature on impulse response auto-encoder models can be incorporated into this work. Models such as [6, 7] used physics-based and signal-based constraints to interpolate acoustic filter parameters. This paper studies two neural constraints that can be used alongside acoustic-based constraints in future work. The first constraint is the requirement of a disentangled latent space through variational auto-encoders (VAE) [8, 9]. The second constraint requires the convex combination of samples in the impulse response dataset to be reconstructed with the auto-encoder [10].

To our knowledge, this is a novel combination of neural networks and ANC. Early ANC work directly used neural networks as adaptive filters, replacing the adaptive FIR filter with shallow artificial neural networks [11–13]. With advances in machine learning since 2010, progressively deeper neural networks have been employed. These models have been trained on a variety of noise statistics, such as those of hums or impulsive sounds, and either provided gradient updates to the filter or completely replaced the adaptive filter based on the noise source statistics [14–17]. While changing noise statistics are well accounted for, these works do not consider dynamic environments that change *during* ANC, e.g., due to the changing position of the noise source.

This paper provides evidence that when the noise source is spatially constrained to be in a region, the LFxLMS algorithm will converge faster than the traditional FxLMS. The impact of neural-network constraints and normalized latent gradients are also studied.

2. FXLMS AND LATENT FXLMS

This paper assumes a-priori knowledge of the Nyquist sampled noise signal, which we denote as $x = [x_0, x_1, \dots]$. Assume all filters have the same filter length L . Let the primary noise acoustic path $\mathbf{p} = [p_0, \dots, p_{L-1}]^T$ be the impulse response between the noise source and the error microphone. Let the control or secondary path $\mathbf{g} = [g_0, \dots, g_{L-1}]^T$ be the impulse response between the control speaker and the error microphone, and assume there is an estimate of the secondary path denoted as $\hat{\mathbf{g}}$. Furthermore, let $\mathbf{w}[n] = [w_0, \dots, w_{L-1}]^T$ be the adaptive filter coefficients. Finally, let $e = [e_0, e_1, \dots]$ denote the sampled signal the error microphone receives. In this discrete-time, linear,



and time-varying scenario, we want to solve the following minimization problem,

$$\mathbf{w}_{\text{opt}} = \arg \min_{\mathbf{w}} E_x[(e)^2], \quad (1)$$

where $E_x[\cdot]$ is the expectation over the input signal x , and the microphone signal e is given by

$$e = \mathbf{p} * x + \mathbf{g} * \mathbf{w} * x, \quad (2)$$

where $*$ is the linear convolution operator.

2.1 Filtered-X LMS algorithm

The "Filtered-X" component of the FxLMS refers to filtering x with the estimated secondary path $\hat{\mathbf{g}}$ to form the filtered signal $\hat{x} = \hat{\mathbf{g}} * x$ whose samples are given by

$$\begin{aligned} \hat{x}_n &= \hat{\mathbf{g}}^T \mathbf{x}_n \\ &= \sum_{i=0}^{L-1} \hat{g}_i x_{n-i}, \end{aligned} \quad (3)$$

where $\mathbf{x}_n = [x_n \ x_{n-1} \ \dots \ x_{n-L+1}]^T$. This paper uses the normalized update of the FxLMS algorithm, given as

$$\mathbf{w}[n] = \mathbf{w}[n-1] - \mu e_n \frac{\hat{\mathbf{x}}_n}{\epsilon + \|\hat{\mathbf{x}}_n\|^2}, \quad (4)$$

where μ is the user-selected step size, and $\hat{\mathbf{x}}_n = [\hat{x}_n, \dots, \hat{x}_{n-L+1}]$, and $\epsilon > 0$ is a small constant to prevent a division by zero. While the normalized FxLMS requires more computation, it is preferred under time-varying conditions [18]. For computational efficiency, we perform the adaptive filtering in blocks of samples, which means we average the weight update across samples before updating the filter. A diagram of this algorithm is provided in Figure 2.

2.2 Latent Filtered-x Least Mean Squared

The Latent Filtered-x LMS (LFxLMS) algorithm assumes that the primary source position lies within a known room region and that we know the static secondary path. Thus, the space of possible adaptive filters \mathcal{W} can be approximated by a manifold of a lower dimension than L . Under this manifold assumption, say that there exists a differentiable function \mathcal{D} that maps a low-dimensional space \mathcal{Z} to $\mathcal{W} \subset \mathbb{R}^L$. The LFXLMS algorithm requires $\mathbf{w}[n]$ to be in the range space of \mathcal{D} .

$$\mathbf{w}[n] = \mathcal{D}(\mathbf{z}[n]) \quad (5)$$

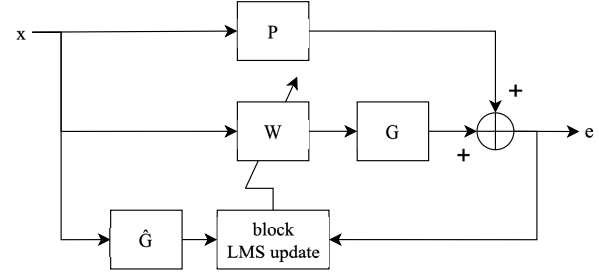


Figure 2. The FxLMS algorithm layout. The goal of ANC is to acoustically cancel the image of the noise source $\mathbf{p} * x$ at a desired location. A control speaker emits a filtered reference of the noise signal, whose image $\mathbf{g} * \mathbf{w} * x$ negates the noise source.

The optimization in Equation (1) is then reformulated as,

$$\begin{aligned} \mathbf{z}_{\text{opt}} &= \arg \min_{\mathbf{z}} E_x[(e)^2] \\ &= \arg \min_{\mathbf{z}} E_x[(\mathbf{p} * x + \mathbf{g} * \mathcal{D}(\mathbf{z}) * x)^2]. \end{aligned} \quad (6)$$

Assuming \mathcal{D} is differentiable, we can adapt the latent variable using the Jacobian Υ evaluated at the latent variable,

$$\Upsilon(\mathbf{z}^*) = \begin{bmatrix} \frac{\partial w_1}{\partial z_1} \big|_{\mathbf{z}=\mathbf{z}^*} & \dots & \frac{\partial w_L}{\partial z_1} \big|_{\mathbf{z}=\mathbf{z}^*} \\ \vdots & \ddots & \vdots \\ \frac{\partial w_1}{\partial z_k} \big|_{\mathbf{z}=\mathbf{z}^*} & \dots & \frac{\partial w_L}{\partial z_k} \big|_{\mathbf{z}=\mathbf{z}^*} \end{bmatrix}, \quad (7)$$

where w_1, \dots, w_L are the filter coefficients. The Jacobian matrix $\Upsilon(\mathbf{z})$ linearly maps a tangent vector at $\mathbf{w}[n] = \mathcal{D}(\mathbf{z}[n])$ to a tangent vector at the latent vector $\mathbf{z}[n]$. Thus, we can map the FxLMS gradient to the corresponding gradient direction in the latent space. The chain rule yields the following Latent FxLMS update rule,

$$\mathbf{z}[n] = \mathbf{z}[n-1] - \mu_z \Upsilon(\mathbf{z}[n-1])(e_n \hat{\mathbf{x}}_n), \quad (8)$$

where μ_z is the step-size in latent space. Figure 3 is a diagram that describes the latent FxLMS algorithm, with Figure 4 defining the latent update block.

2.3 Normalization Schemes for LFXLMS

Using a normalized step size can improve the convergence of adaptive filters. Two possible approaches to normalization exist when adapting in \mathcal{Z} .



FORUM ACUSTICUM EURONOISE 2025

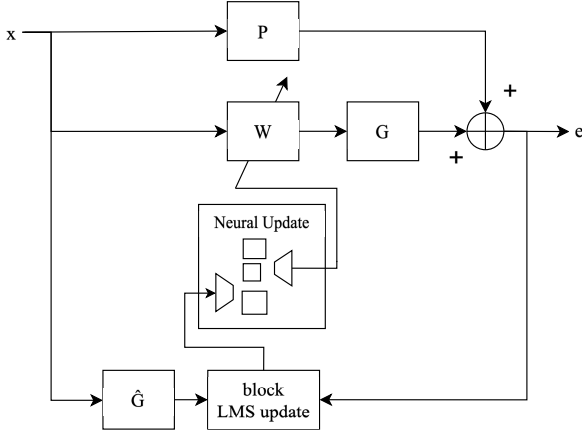


Figure 3. The latent FxLMS algorithm layout, where adaptations occur in the latent space of the decoder. The Neural Update block is expanded in Figure 4.

2.3.1 Data-normalized Latent LMS

The first approach treats the normalized gradient of the normal FxLMS as the input gradient into $\Upsilon(z)$,

$$\mathbf{z}[n] = \mathbf{z}[n-1] - \frac{\mu_z}{\|\hat{\mathbf{x}}_n\|^2 + \epsilon} \Upsilon(\mathbf{z}[n-1])(e_n \hat{\mathbf{x}}_n). \quad (9)$$

2.3.2 Latent-normalized Latent LMS

The second approach involves normalizing the latent gradient,

$$\mathbf{z}[n] = \mathbf{z}[n-1] - \frac{\mu_z \Upsilon(\mathbf{z}[n-1])(e_n \hat{\mathbf{x}}_n)}{\|\Upsilon(\mathbf{z}[n-1]) \hat{\mathbf{x}}_n\|^2 + \epsilon}. \quad (10)$$

The data-normalized LFXLMS normalizes the weight gradient before being input Υ , while the latent-normalized FxLMS normalizes the latent gradient. As shown in Section 4, the normalization schemes can affect the convergence rate of the LFXLMS algorithm.

3. AUTO-ENCODER MODELS

The NNSI framework requires an auto-encoder trained on the space of possible filter coefficients. An auto-encoder is the composition of an encoder \mathcal{E} , which maps $\mathcal{W} \subset \mathbb{R}^L$ to a lower-dimensional latent space $\mathcal{Z} \subset \mathbb{R}^k$, with a decoder \mathcal{D} , which maps the latent variable to an element of the input space. The LFXLMS algorithm constrains the weights to be in the range space of the decoder \mathcal{D} . Thus,

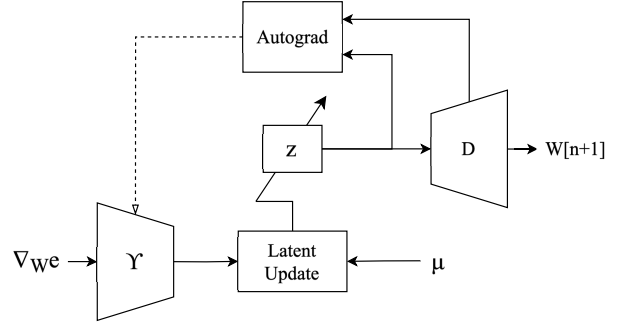


Figure 4. This model defines the latent variable update block, which inputs the weight gradient and outputs the next adaptive filter coefficients. The weight gradient is the input of Υ , which is the Jacobian of the decoder at z .

LFxLMS performance is heavily influenced by how the auto-encoder is trained.

This paper uses the same structure for all models presented to verify the LFXLMS performance using various neural-based constraints. The encoder first transforms the time-domain filter coefficients into the concatenation of the real and imaginary components of the Real Fast Fourier Transform (RFFT) of the filter coefficients, then feeds this representation into the following fully connected model.

$$\begin{aligned} \bar{\mathbf{w}} &= [\Re(\text{RFFT}(\mathbf{w})), \Im(\text{RFFT}(\mathbf{w}))] \\ \mathbf{z} &= \mathcal{E}(\mathbf{w}) = (\mathcal{E}_2 \circ \sigma \circ f_{\text{layernorm}} \circ \mathcal{E}_1)(\bar{\mathbf{w}}), \end{aligned} \quad (11)$$

where $\mathcal{E}_1, \mathcal{E}_2$ are fully connected layers, z is that latent variable that gets adapted in LFXLMS, w is the time-domain representation of the filter coefficients, \circ is the composition map, $f_{\text{layernorm}}$ is the layer normalization, σ is the Sigmoid Linear Unit (SiLU) as the activation function, and \Re, \Im represents the real and imaginary components of a complex number.

The decoder takes a similar structure using the fully connected layers $\mathcal{D}_1, \mathcal{D}_2$ and transforms their output to time-domain filter coefficients.

$$\begin{aligned} [\bar{\mathbf{w}}_1, \bar{\mathbf{w}}_2] &= (\mathcal{D}_2 \circ \sigma \circ f_{\text{layernorm}} \circ \mathcal{D}_1)(\mathbf{z}) \\ \hat{\mathbf{w}} &= \mathcal{D}(\mathbf{z}) = \text{IRFFT}(\bar{\mathbf{w}}_1 + j\bar{\mathbf{w}}_2), \end{aligned} \quad (12)$$

where j is the imaginary constant. All the models are designed using PyTorch, which allows for the transformation between filter representations using automatic differentiation [19].



3.1 Variational Auto-Encoders

The Variational Auto-Encoder (VAE) aims to produce a latent space distributed as a standardized Gaussian [8]. This allows the latent variables to be uncorrelated (also known as disentangled) and the decoder to be used as a generator of input samples. The encoder of the VAE \mathcal{E}_{VAE} is slightly different from that of Equation (11) because the layer \mathcal{E}_2 outputs statistical parameters from which the latent variable is sampled. Using an auto-encoder model with a disentangled latent space could benefit latent adaptation because otherwise the latent gradient vector could point orthogonally to inherent correlations between latent variables.

However, in [9], they demonstrate that maximizing the VAE ELBO can sometimes lead to a posterior mode collapse, where the decoder and encoder of the VAE fail to output a diverse set of samples. The Information Variational Auto-encoder (InfoVAE) [9] addresses the issue of posterior mode collapse by adding the mutual information term $I(x; z)$ in the objective. This leads to replacing the K-L divergence loss of a VAE [8] with the Maximum-Mean Discrepancy [20], which is a kernel-based loss. This paper implements the [9] model with parameters $\alpha = 1, \lambda = 1000$.

3.2 Auto-encoder with uniform mixup

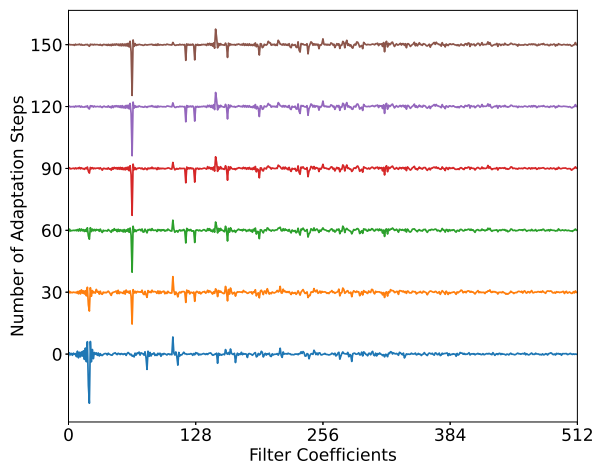


Figure 5. The FxLMS adaptive filter weights as the source position moves. Notice that the filter progressively removes the initial values of the filter while adding in the filter's final values.

When the noise source location changes, the FxLMS algorithm will adapt w to account for this change, removing components of the filter values at the beginning while adding components of the filter values at the end, as shown in Figure 5. The intermediate stages of the adaption process will likely not correspond to the space of converged w filters that would normally be used to train the auto-encoder. Thus, we examine what happens when the auto-encoder is trained on convex combinations of random data pairs. This gives the following loss, to be added to the reconstruction loss of the auto-encoder

$$\mathcal{L}_C = E_{\mathbf{w}_1, \mathbf{w}_2 \in \mathcal{W}, \gamma \in U[0,1]} [(\gamma \mathbf{w}_1 + (1 - \gamma) \mathbf{w}_2 - \mathcal{D} \circ \mathcal{E}(\gamma \mathbf{w}_1 + (1 - \gamma) \mathbf{w}_2))^2], \quad (13)$$

where $U[0, 1]$ is a uniform distribution. On top of this, we also enforce that the latent variables match the convex combination of samples of \mathcal{W} with the following loss

$$L_z = E_{\mathbf{w}_1, \mathbf{w}_2 \in \mathcal{W}, \gamma \in U[0,1]} [\gamma \mathcal{E}(\mathbf{w}_1) + (1 - \gamma) \mathcal{E}(\mathbf{w}_2) - \mathcal{E}(\gamma \mathbf{w}_1 + (1 - \gamma) \mathbf{w}_2)]^2, \quad (14)$$

This expectation is approximated within each training batch of \mathcal{W} . Note that when a Dirichlet distribution instead models γ , the combined loss is a technique called 'mixup' [10]. In mixup, this process is used for data augmentation, i.e., to generate synthetic samples to produce a richer mapping than what the dataset provides.

4. EXPERIMENTS

We compare the performance of FxLMS with the LFXLMS model under various neural network constraints in a simulated acoustic environment. This experiment focuses on three considerations for LFXLMS: the impact of the VAE and infoVAE decoder, the improvement of convergence when the decoder is trained with the mixup constraint, and a comparison of the two normalized updates in the latent space. Currently, it is difficult to theoretically assess latent adaptive filtering. However, empirical methods can still yield valuable insights. This paper shows similar plots of ANC error signals and provides empirical measurements of convergence times.

4.1 Simulated ANC with the Image Source Model

We simulated ANC in a reverberant room using *pyroomacoustics* [21], a Python implementation of the Image Source Model (ISM) [22]. In a $6 \times 6.2 \times 3$ m room with 0.15 s RT60 reverberation time, a secondary speaker is



FORUM ACUSTICUM EURONOISE 2025

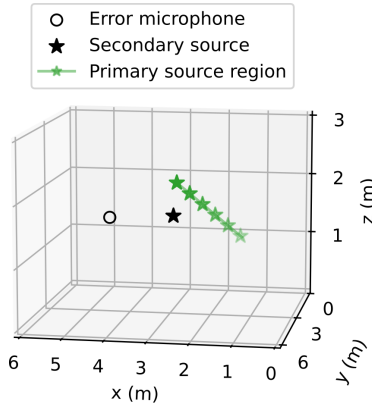


Figure 6. A room layout with the experimental parameters. The source positions are constrained to a line as the LFXLMS algorithm cancels the sound at the error microphone.

placed at the coordinates $[3, 2.5, 1.5]$ m, and an error microphone is placed at $[4.5, 3, 1.5]$ m. The region of primary source location is a line segment from $[1.5, 1, 1]$ m to $[3, 2, 2]$ m, of which we uniformly sample 2,048 primary source positions. Note that each primary source position has a roughly 2 mm spacing and that the primary source is farther away from the error microphone than the secondary source at every point. We obtain impulse responses of length 512 samples from the secondary and primary paths at a sampling rate of 16,000 Hz. The FxLMS algorithm is performed at each primary source position, and we obtain the converged filters to form \mathcal{W} .

From there, we train various auto-encoders with different constraints on \mathcal{W} . As mentioned in Section 3, we use a two-layer network for the encoder and decoder across all examined models. The hidden size was 256, and the latent dimension was 32. The mixup constraint was implemented with 256 convex combinations per batch with a batch size of 64. The InfoVAE constraint is run with using a Gaussian kernel with kernel variance 0.01.

The FxLMS experiments are performed in blocks of 100 samples corresponding to 6.25 ms, with the weights starting from the 0 vector. The maximum stable step sizes must be found for each LFXLMS model to examine convergence speed across models. Determining the upper limits of the latent step-size for LFXLMS is difficult, so the maximum stable step size was determined empirically for each model. The latent-normalized models tend to require step-sizes within $[.05, .3]$, and the data-normalized mod-

els used step-sizes within $[5, 100]$. The primary source position is randomly initialized for each trial and switches to another random position at the 100th block to see how each model adapts to the rapid change. The error signals are collected and averaged from 50 trials per model, which is then analyzed.

4.2 Results on Simulated Data

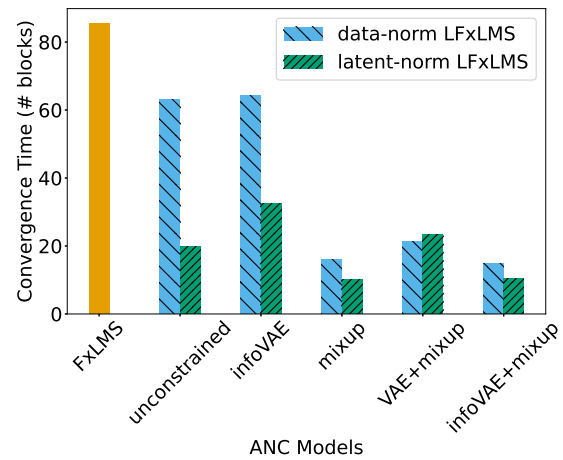


Figure 7. The convergence time of FxLMS and LFXLMS under various neural network constraints and update normalization schemes measured in blocks of samples. Data-norm stands for data-normalization, and lat-norm stands for latent-normalization. The convergence time of LFXLMS with the VAE decoder is not shown.

This paper uses empirical methods of measuring the convergence rate of the LFXLMS algorithms. Section 23A of [18] gives the following empirical measure of convergence,

$$C(e) = \arg \min_k e_k^2 \leq (1 + \rho) e_\infty^2, \quad (15)$$

where e is the error signal, and ρ is a small positive constant. If $\rho = 0.1$, it means the earliest sample in which the squared error is within 10% of the steady-state mean squared error.

Figure 7 shows the convergence times of the various LFXLMS models compared to FxLMS. For each model, the convergence times were estimated with $\rho = 0.4$ and averaged over the 50 trials. To obtain an estimation of e_∞^2 , the error signal was averaged over the last 40 blocks.



FORUM ACUSTICUM EURONOISE 2025

For all LFXLMS models shown, the convergence time is faster than the baseline. For most of the models, the latent-normalization leads to faster convergence. As for neural constraints, the models with the mixup constraint lead to faster convergence compared to models without this constraint. The LFXLMS model with the shortest convergence time is the latent-normalized LFXLMS with an InfoVAE decoder trained with mixup. Figure 8 shows the averaged error signal, which supports the results from Figure 7.

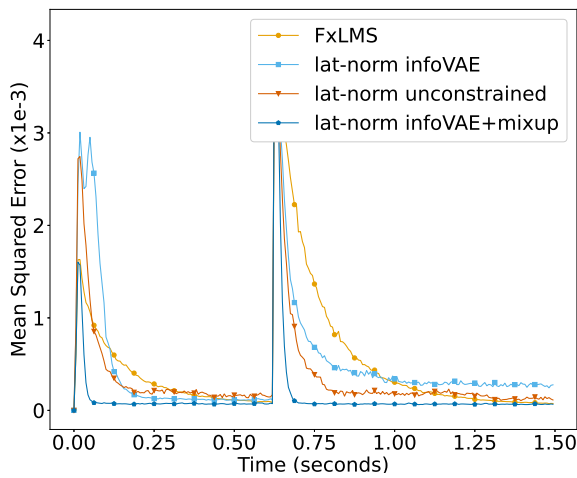


Figure 8. The averaged error signal is taken from latent-normalized models. The latent-normalized mixup LFXLMS is not shown because it is visually indistinguishable from the mixup-constrained infoVAE model.

The steady-state ANC performance per model and trial is calculated as

$$10 \log \frac{e_{\infty, \text{ANC OFF}}^2}{e_{\infty, \text{ANC ON}}^2}, \quad (16)$$

where e_{∞}^2 was estimated using the last 40 blocks of the error signal. Comparing across the models examined in Figure 7, the ANC performance was averaged through the trials were all comparable. FxLMS had an average 37dB drop in error energy, and all other models were within 3dB of the FxLMS performance. This is surprising considering that the neural adaptive filters did not seem to follow the general tradeoff between convergence rate and steady state performance with traditional adaptive filters [18].

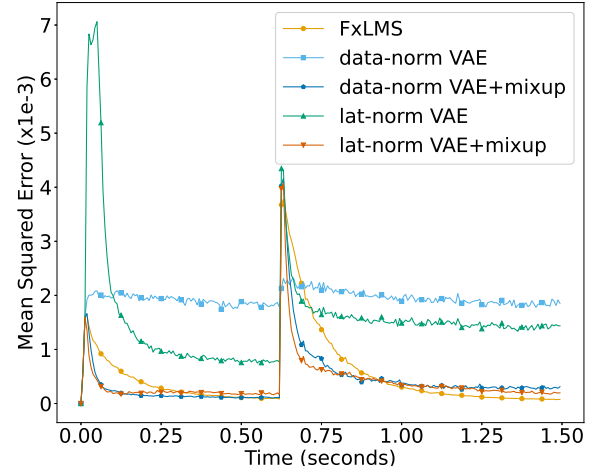


Figure 9. The averaged error signal is taken from all standard VAE models compared to FxLMS.

The LFXLMS model that uses the VAE decoder was not included in Figure 7 because the VAE LFXLMS had a significantly worse convergence time and steady-state mean squared error than all other models. Figure 9 shows the averaged error signal of every VAE model compared to FxLMS. While the mixup loss combined with the VAE will still improve the overall ANC performance, Figure 7 shows better alternatives.

5. CONCLUSION

We have found that performing the FxLMS update in the latent space of the decoder of an auto-encoder yields significant improvements in convergence speed. This is especially true when LFXLMS is performed with latent-normalized updates with a decoder trained with the mixup constraint. The improvements on ANC convergence using shallow neural networks show the potential promise of its capability for hardware integration with ANC devices. More work needs to be done on evaluating the interpolation capabilities of the LFXLMS algorithm. If an ANC application constrains the primary source position to a spatial region, the LFXLMS algorithm should be considered based on the work shown.

6. ACKNOWLEDGMENTS

The authors acknowledge support from the Foxconn Interconnect Technology sponsored Center for Networked In-



FORUM ACUSTICUM EURONOISE 2025

telligent Components and Environments (C-NICE) at the University of Illinois at Urbana-Champaign.

7. REFERENCES

- [1] L. Lu, K.-L. Yin, R. C. de Lamare, Z. Zheng, Y. Yu, X. Yang, and B. Chen, “A survey on active noise control in the past decade—Part I: Linear systems,” *Signal Processing*, vol. 183, p. 108039, 2021.
- [2] J. Mejia, B. Mederos, N. Gordillo, and L. Ortega, “Adaptive filter with riemannian manifold constraint,” *Scientific Reports*, vol. 13, no. 1, p. 9014, 2023.
- [3] K. Helwani, P. Smaragdis, and M. M. Goodwin, “Generative Modeling Based Manifold Learning for Adaptive Filtering Guidance,” in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, 2023.
- [4] T. Hardenbicker and P. Jax, “Online System Identification on Learned Acoustic Manifolds Using an Extended Kalman Filter,” in *2024 18th International Workshop on Acoustic Signal Enhancement (IWAENC)*, pp. 339–343, 2024.
- [5] J. Casebeer, N. J. Bryan, and P. Smaragdis, “Meta-AF: Meta-Learning for Adaptive Filters,” 2022.
- [6] A. Luo, Y. Du, M. J. Tarr, J. B. Tenenbaum, A. Torralba, and C. Gan, “Learning Neural Acoustic Fields,” 2023.
- [7] A. Brendel, J. Zeitler, and W. Kellermann, “Manifold learning-supported estimation of relative transfer functions for spatial filtering,” 2021.
- [8] D. P. Kingma and M. Welling, “An Introduction to Variational Autoencoders,” *Foundations and Trends® in Machine Learning*, vol. 12, no. 4, p. 307–392, 2019.
- [9] S. Zhao, J. Song, and S. Ermon, “InfoVAE: Information Maximizing Variational Autoencoders,” 2018.
- [10] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “mixup: Beyond Empirical Risk Minimization,” in *International Conference on Learning Representations*, 2018.
- [11] L. Lu, K.-L. Yin, R. C. de Lamare, Z. Zheng, Y. Yu, X. Yang, and B. Chen, “A survey on active noise control in the past decade—Part II: Nonlinear systems,” *Signal Processing*, vol. 181, p. 107929, 2021.
- [12] G. L. Sicuranza and A. Carini, “A Generalized FLANN Filter for Nonlinear Active Noise Control,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 8, pp. 2412–2417, 2011.
- [13] C.-Y. Chang, “Neural filtered-U algorithm for the application of active noise control system with correction terms momentum,” *Digital Signal Processing*, vol. 20, no. 4, pp. 1019–1026, 2010.
- [14] Y.-J. Cha, A. Mostafavi, and S. S. Benipal, “DNoiseNet: Deep learning-based feedback active noise control in various noisy environments,” *Engineering Applications of Artificial Intelligence*, vol. 121, p. 105971, 2023.
- [15] Z. Luo, H. Ma, D. Shi, and W.-S. Gan, “GFANC-RL: Reinforcement Learning-based Generative Fixed-filter Active Noise Control,” *Neural Networks*, vol. 180, p. 106687, 2024.
- [16] B. Raeisy and S. Golbahar Haghighi, “Active Noise Controller with reinforcement learning,” in *The 16th CSI International Symposium on Artificial Intelligence and Signal Processing (AISIP 2012)*, pp. 074–079, 2012.
- [17] H. Zhang and D. Wang, “Deep MCANC: A deep learning approach to multi-channel active noise control,” *Neural Networks*, vol. 158, pp. 318–327, 2023.
- [18] A. H. Sayed, *Adaptive Filters*. Hoboken, NJ: Wiley-Blackwell, 2008.
- [19] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, “Automatic differentiation in pytorch,” 2017.
- [20] A. Gretton, K. Borgwardt, M. J. Rasch, B. Scholkopf, and A. J. Smola, “A Kernel Method for the Two-Sample Problem,” 2008.
- [21] R. Scheibler, E. Bezzam, and I. Dokmanic, “Pyroomacoustics: A Python Package for Audio Room Simulation and Array Processing Algorithms,” in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, p. 351–355, IEEE, Apr. 2018.
- [22] J. Allen and D. Berkley, “Image method for efficiently simulating small-room acoustics,” *The Journal of the Acoustical Society of America*, vol. 65, pp. 943–950, 04 1979.

