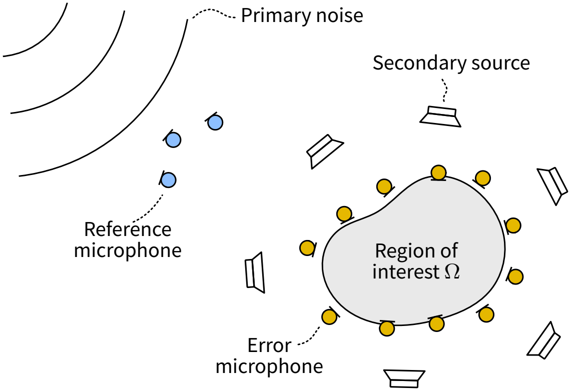
Kernel-Interpolation-based Filtered-x Least Mean Square for Spatial Active Noise Control in Time Domain

Jesper Brunnström & Shoichi Koyama

The University of Tokyo

IEEE ICASSP 2021

Spatial Active Noise Control



Goals

- Broadband noise reduction over a continuous region
- Flexible array placement
- Low computational cost

Problem formulation

Cost function for multipoint pressure control

$$\mathcal{J} = \mathbb{E}\left[\boldsymbol{e}^\top(n)\,\underbrace{\boldsymbol{e}(n)}_{\text{Error signal}}\right]$$

- Minimizing noise only near error microphones

Problem formulation

Cost function for multipoint pressure control

$$\mathcal{J} = \mathbb{E}\left[\boldsymbol{e}^\top(n)\, \underbrace{\boldsymbol{e}(n)}_{\text{Error signal}}\right]$$

- Minimizing noise only near error microphones

Cost function for spatial active noise control

$$\mathcal{L} = \mathbb{E}\left[\int_\Omega \underbrace{u(n,\boldsymbol{r})}_{\text{Sound pressure at } \boldsymbol{r}\in\Omega}{}^2 \, d\boldsymbol{r}\right]$$

- Minimizing noise over the region Ω.
- Sound pressure $u(n,\boldsymbol{r})$ is not directly measurable.

Kernel interpolation of sound field

- Estimate $u(n, \boldsymbol{r})$ from error microphone measurements $\boldsymbol{e}(n)$.
- Interpolation filter $\boldsymbol{z}$ obtained in closed form in frequency domain.

$$u(n, \boldsymbol{r}) = \sum_{i=-\infty}^{\infty} \boldsymbol{z}^\top(i, \boldsymbol{r}) \boldsymbol{e}(n - i).$$

$$\boldsymbol{z}(i, \boldsymbol{r}) = \underbrace{\mathcal{F}^{-1}}_{\text{Inverse Fourier transform}} \Big[\boldsymbol{z}(\omega, \boldsymbol{r})\Big] = \mathcal{F}^{-1} \Big[\big[(\boldsymbol{K}(\omega) + \lambda \boldsymbol{I})^{-1}\big]^\top \boldsymbol{\kappa}(\omega, \boldsymbol{r}) \Big]$$

Kernel interpolation of sound field

- Estimate $u(n, \boldsymbol{r})$ from error microphone measurements $\boldsymbol{e}(n)$.
- Interpolation filter $\boldsymbol{z}$ obtained in closed form in frequency domain.

$$u(n, \boldsymbol{r}) = \sum_{i=-\infty}^{\infty} \boldsymbol{z}^\top(i, \boldsymbol{r}) \boldsymbol{e}(n - i).$$

$$\boldsymbol{z}(i, \boldsymbol{r}) = \underbrace{\mathcal{F}^{-1}}_{\text{Inverse Fourier transform}} \left[\boldsymbol{z}(\omega, \boldsymbol{r}) \right] = \mathcal{F}^{-1} \left[\left[(\boldsymbol{K}(\omega) + \lambda \boldsymbol{I})^{-1} \right]^\top \boldsymbol{\kappa}(\omega, \boldsymbol{r}) \right]$$

Defined by kernel function κ

- Restricts interpolating function to satisfy Helmholtz equation.

Spherical Bessel function

$$\kappa(\omega, \boldsymbol{r}, \boldsymbol{r}') = j_0\left(\frac{\omega}{c} \|\boldsymbol{r} - \boldsymbol{r}'\|\right)$$

Speed of sound

$$\boldsymbol{\kappa}(\omega, \boldsymbol{r}) = \left[\kappa(\omega, \boldsymbol{r}, \boldsymbol{r}_1), \ldots, \kappa(\omega, \boldsymbol{r}, \boldsymbol{r}_M) \right]^\top$$

$$\boldsymbol{K}(\omega) = \begin{bmatrix} \kappa(\omega, \boldsymbol{r}_1, \boldsymbol{r}_1) & \ldots & \kappa(\omega, \boldsymbol{r}_M, \boldsymbol{r}_1) \\ \vdots & \ddots & \vdots \\ \kappa(\omega, \boldsymbol{r}_1, \boldsymbol{r}_M) & \ldots & \kappa(\omega, \boldsymbol{r}_M, \boldsymbol{r}_M) \end{bmatrix}$$

Kernel-interpolation-based FxLMS

- Rewrite cost function using interpolation filter

$$\mathcal{L} = \mathbb{E}\left[\int_{\Omega} u(n, \boldsymbol{r})^2 \, d\boldsymbol{r}\right] = \mathbb{E}\left[\sum_{i,j=-\infty}^{\infty} \boldsymbol{e}^{\top}(n-i) \underbrace{\boldsymbol{\Gamma}(i,j)}_{\int_{\Omega} \boldsymbol{z}(i,\boldsymbol{r}) \boldsymbol{z}^{\top}(j,\boldsymbol{r}) \, \mathrm{d}\boldsymbol{r}} \boldsymbol{e}(n-j)\right]$$

Kernel-interpolation-based FxLMS

- Rewrite cost function using interpolation filter

$$\mathcal{L} = \mathbb{E}\left[\int_\Omega u(n,\boldsymbol{r})^2\, d\boldsymbol{r}\right] = \mathbb{E}\left[\sum_{i,j=-\infty}^{\infty} \boldsymbol{e}^\top(n-i)\underbrace{\boldsymbol{\Gamma}(i,j)}_{\int_\Omega \boldsymbol{z}(i,\boldsymbol{r})\boldsymbol{z}^\top(j,\boldsymbol{r})\,\mathrm{d}\boldsymbol{r}}\boldsymbol{e}(n-j)\right]$$

- Simplify interpolation weighting filter

$$\boldsymbol{A}(k) = \sum_{\nu=-\infty}^{\infty} \boldsymbol{\Gamma}(\nu,\nu+k) = \mathcal{F}^{-1}\left[\int_\Omega \boldsymbol{z}^*(\omega,\boldsymbol{r})\boldsymbol{z}^\top(\omega,\boldsymbol{r})d\boldsymbol{r}\right]$$

Kernel-interpolation-based FxLMS

- Rewrite cost function using interpolation filter

$$\mathcal{L} = \mathbb{E}\left[\int_\Omega u(n,\boldsymbol{r})^2\, d\boldsymbol{r}\right] = \mathbb{E}\left[\sum_{i,j=-\infty}^{\infty} \boldsymbol{e}^\top(n-i)\, \underbrace{\boldsymbol{\Gamma}(i,j)}_{\int_\Omega \boldsymbol{z}(i,\boldsymbol{r})\boldsymbol{z}^\top(j,\boldsymbol{r})\,\mathrm{d}\boldsymbol{r}}\, \boldsymbol{e}(n-j)\right]$$

- Simplify interpolation weighting filter

$$\boldsymbol{A}(k) = \sum_{\nu=-\infty}^{\infty} \boldsymbol{\Gamma}(\nu,\nu+k) = \mathcal{F}^{-1}\left[\int_\Omega \boldsymbol{z}^*(\omega,\boldsymbol{r})\boldsymbol{z}^\top(\omega,\boldsymbol{r})d\boldsymbol{r}\right]$$

- Compute gradient

$$\nabla\mathcal{L}(i) = \sum_{k=-\infty}^{\infty}\sum_{j=0}^{J-1} \underbrace{\boldsymbol{G}^\top(j)}_{\text{Secondary path}}\, \boldsymbol{A}(k)\mathbb{E}\left[\boldsymbol{e}(n)\, \underbrace{\boldsymbol{x}^\top(n-i-j-k)}_{\text{Reference signal}}\right]$$

Practical algorithm

- Truncate and delay $\boldsymbol{A}$ to obtain causal finite impulse response filter $\bar{\boldsymbol{A}}$
- Delay error signal $\boldsymbol{e}$ to obtain correct cross-correlation
- Combine interpolation filter and secondary path

$$\boldsymbol{H}(i) = \sum_{j=-\infty}^{\infty} \bar{\boldsymbol{A}}^\top(j)\boldsymbol{G}(i-j)$$

- Use instantaneous gradient estimate for update of control filter $\boldsymbol{W}$

$$\boldsymbol{W}_{n+1}(i) = \boldsymbol{W}_n(i) - \mu \sum_{j=0}^{J+2K-1} \boldsymbol{H}^\top(j)\boldsymbol{e}(n-K)\boldsymbol{x}^\top(n-i-j)$$

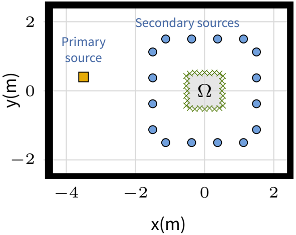
Fast block KI-FxLMS

- Perform costly filter update block-wise in frequency domain
- Compute convolution and correlation operations with overlap-save and fast Fourier transform
- Filter update is delayed by block length
- Control filter is obtained in time domain
- Loudspeaker signals are computed each sample to maintain causality

Evaluation

- Simulation in reverberant space, $\mathrm{RT}_{60} \approx 0.35s$
- 28 error microphones and 16 loudspeakers.

Simulated room



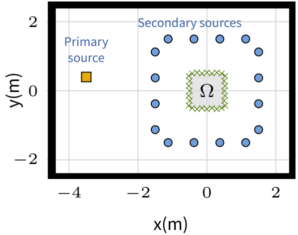
Noise sources

- *Noise* - White noise bandlimited to 100-500Hz.
- *Song* - High Horse by artist Secret Mountains.
- *Instrumental* - Instrumental version of *Song*.

Evaluation

- Simulation in reverberant space, $\mathrm{RT}_{60} \approx 0.35s$
- 28 error microphones and 16 loudspeakers.

Simulated room



Noise sources

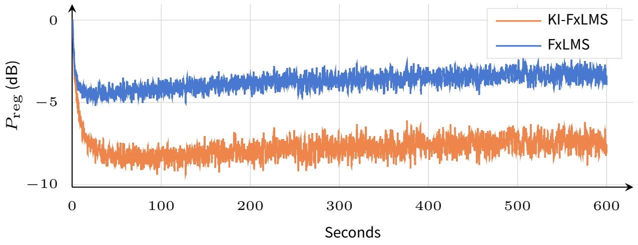
- *Noise* - White noise bandlimited to 100-500Hz.
- *Song* - High Horse by artist Secret Mountains.
- *Instrumental* - Instrumental version of *Song*.

Performance metric: *regional power reduction*

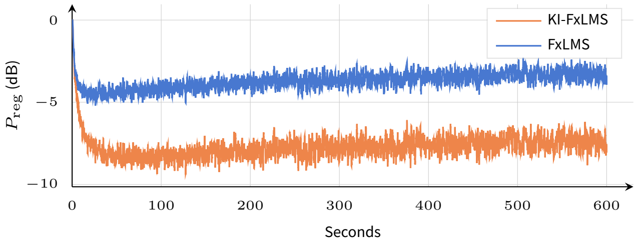
- Ratio of sound power with ANC compared to without ANC.
- Discretize region of interest Ω with 400 equally spaced points $\boldsymbol{r}_\nu$.

$$P_{\mathrm{reg}}(n) = \frac{\sum_\nu \sum_\tau u(n-\tau, \boldsymbol{r}_\nu)^2}{\sum_\nu \sum_\tau u_{\mathrm{p}}(n-\tau, \boldsymbol{r}_\nu)^2}$$

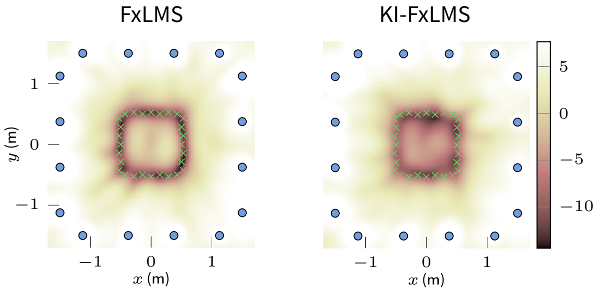
Comparison of KI-FxLMS and FxLMS for *Noise*



Comparison of KI-FxLMS and FxLMS for *Noise*



Power distribution after 600 seconds



Comparison of KI-FxLMS and FxLMS for all source types

Average noise reduction from 120 to 240 seconds.

	Noise	**Song**	**Instrumental**
FxLMS	-3.81	-3.28	-3.13
Fast Block FxLMS	-3.88	-3.17	-3.12
KI-FxLMS	-7.88	$\mathbf{-3.99}$	$\mathbf{-4.39}$
Fast Block KI-FxLMS	$\mathbf{-8.00}$	-3.60	-4.08

- Both proposed methods outperform the conventional methods for all source types.
- For non-stationary noises, fast block KI-FxLMS performance is slightly degraded.

Conclusions

- The proposed KI-FxLMS outperforms FxLMS with regards to regional power reduction.
- Zero to marginal increase in computational cost.
- Fast block implementation significantly more efficient at low cost in performance.