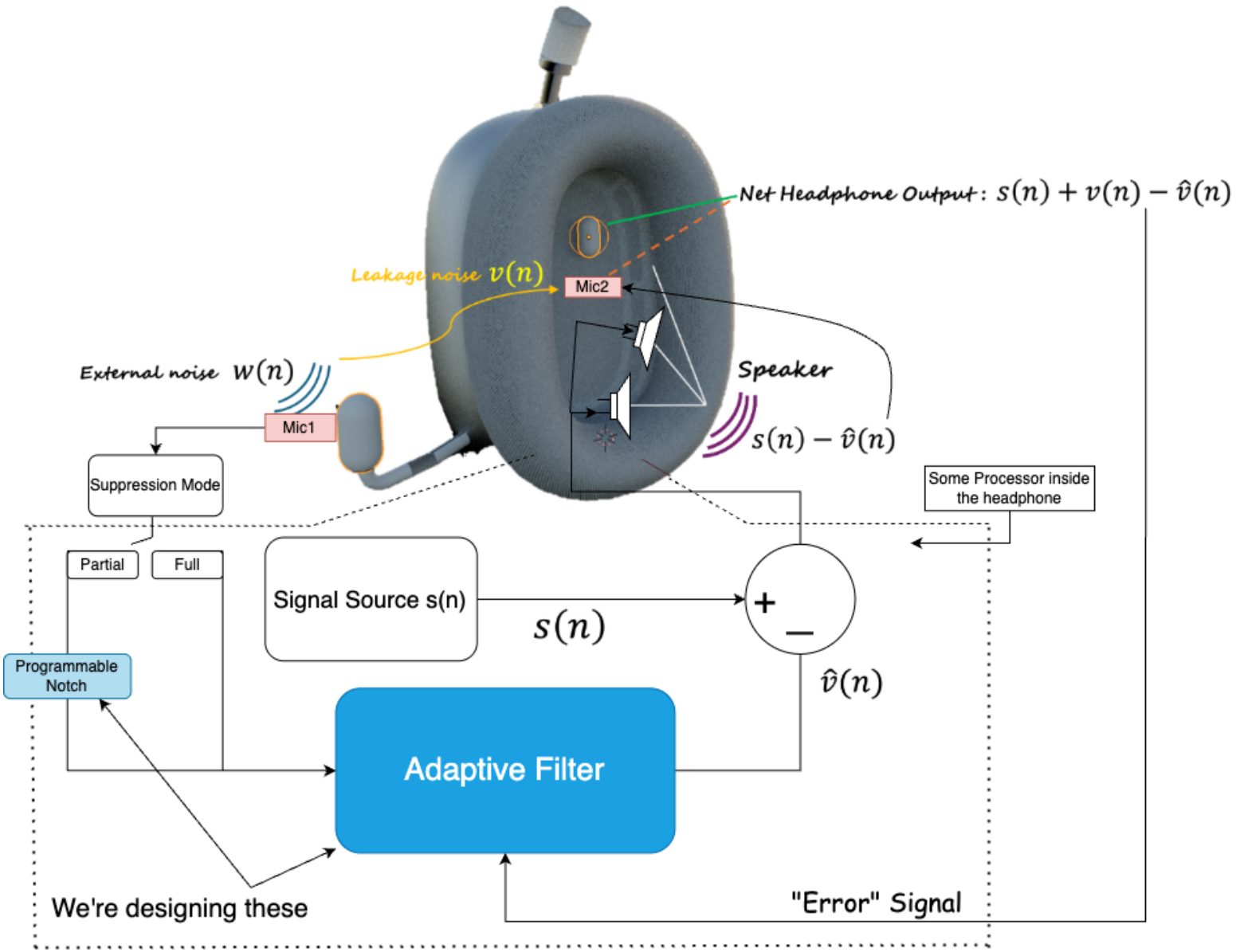
Course Project

Team 5
Dr. Sundaram Vanka

Indian Institute of Technology Hyderabad

April 25, 2025

Block Diagram of The Design



Design Choices and Justification

- We have used **RLS(Recursive Least Squares)** to design the adaptive filter shown in the block diagram.
- In the method of least squares, a deterministic sum of exponentially weighted squared errors is minimised

$$\zeta(n) = \sum_{k=1}^{n} \rho_n(k)\, e_n^2(k) \tag{1}$$

- $e_n(k)$ Error computed using current weights at past time k
- $\rho_n(k)$: Here $\rho_n(k) = \lambda^{n-k}$, where $\lambda \in (0,1]$ is the forgetting factor

WHY RLS

- The ζ here is a better estimate of MSE than per-sample estimates used by LMS which gives more SNR gain.
- RLS uses $3N^2$ + $11N$+ 8 multiplications and $3N^2 + 7N + 4$ additions where as $2N + 3$ multiplications and $2N + 5$ additions in LMS.

Our Design

For full supression, we give the RLS filter the reference noise signal from the external microphone, and the noisy speech signal from the headphones.
For partial suppression, we remove the tones from the reference noise signal, and give it to RLS filter. The tones are removed using a notch filter.

- Filter size, forgetting factor affect the performance of the RLS filter, and the bandwidth of the notch filter affects it's performance.
  - **Forgetting factor** (λ): We chose $\lambda = 0.99$.
  - **Filter Size** (N): We chose $N = 8$.
  - **Bandwidth of Notch Filter**: We chose a bandwidth of 20 Hz.

Assumptions:

- The external noise is uncorrelated with the clean signal.
- The external noise provides a good estimate of the leakage noise reaching the ears.

Performance Metrics:

- **SNR Gain**: We measure the pre and post cancellation SNR, and measure the difference, to quantify the performance of the filter.
- **NRNTN**: We measure the Non tonal noise in Noisy speech, and the residual non tonal noise in the output signal, and compare them.

$$\gamma = \frac{||e(n) - P_{tonal} - P_{clean}||^2}{||d(n) - P_{tonal} - P_{clean}||^2}$$

Time Domain

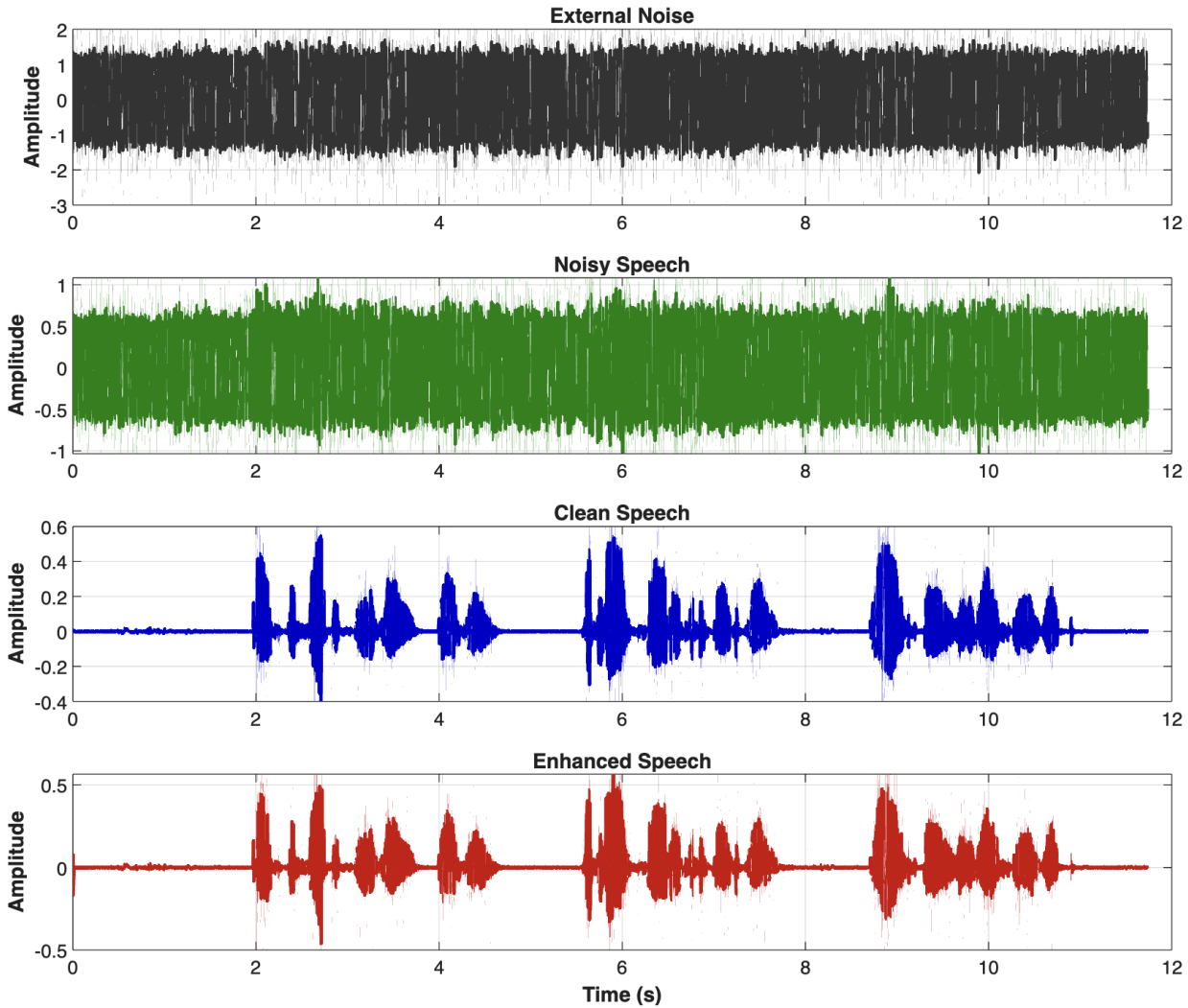


Figura 1: Time Domain Signals

Spectrograms

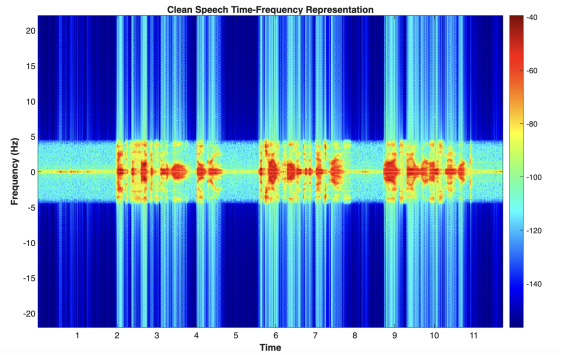


Figura 2: Clean Speech

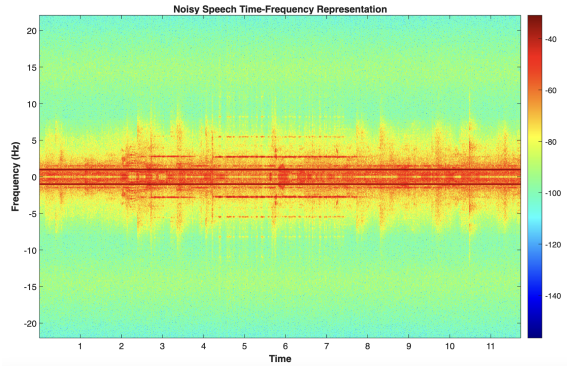


Figura 3: Clean Speech

More Spectrograms

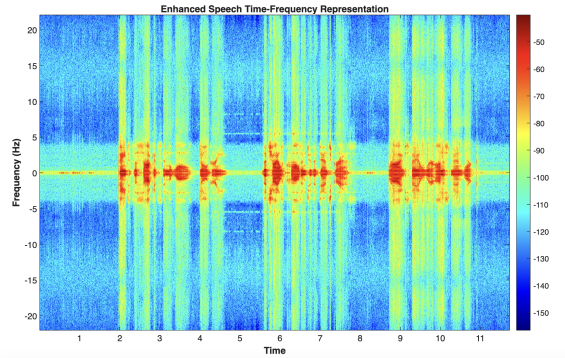


Figura 4: Result with $\lambda = 0.99$

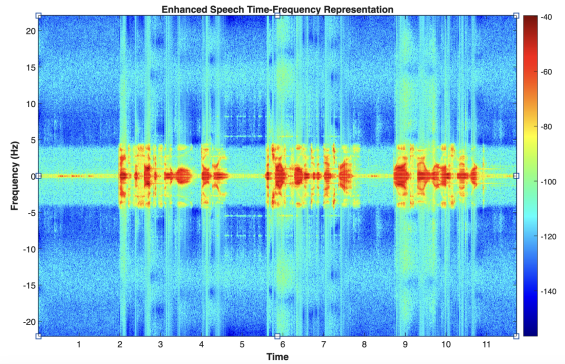


Figura 5: Result with $\lambda = 0.9999$

Comparison: NLMS vs RLS

NLMS (Normalized LMS)

- **Cost Function:**
  $J(n) = e^2(n)$
- **Error:**
  $e(i) = d(i) - \mathbf{w}^T(i)\mathbf{x}(i)$
- **Complexity:**
  - $2N+3$ Multiplications
  - $2N+4$ Additions
  - 1 Division
- **Parameter:** Step size μ
- **Accuracy:** Ensemble average behavior
- **Memory:** Low (only vectors)

RLS (Recursive Least Squares)

- **Cost Function:**
  $J(n) = \sum_{k=0}^{n} \lambda^{n-k} e^2(k)$
- **Error:** $e(k) = d(k) - \mathbf{w}_{n-1}^T \mathbf{x}(k)$
- **Complexity:**
  - $3N^2 + 11N + 8$ Multiplications
  - $3N^2 + 7N + 4$ Additions
  - 1 Division
- **Parameter:** Forgetting factor λ
- **Accuracy:** Optimized for specific signal, hence more accurate
- **Memory:** High (stores matrices)

References and Design Influence I

References

- Farhang-Boroujeny, B. (2013). *Adaptive Filters: Theory and Applications (2nd ed.)*. Wiley.

- Chapter 12 Method Of Least Squares : The Algorithm was taken from this book and all the derivations are influenced by this book.

RLS DERIVATIONS

We aim to minimize the exponentially weighted least-squares error:

$$\zeta(n) = \sum_{k=1}^{n} \lambda^{n-k} e_n^2(k) \tag{12.1}$$

where $0 < \lambda \leq 1$ is the forgetting factor.

$$y_n(k) = \mathbf{w}^T(n)\mathbf{x}(k) \tag{12.2}$$

$$e_n(k) = d(k) - y_n(k) \tag{12.3}$$

We define the filter output and error at time k using weights at time n.

Vector Notation

$$\mathbf{d}(n) = [d(1)\ d(2)\ \ldots\ d(n)]^T \tag{12.4}$$

$$\mathbf{y}(n) = [y_n(1)\ y_n(2)\ \ldots\ y_n(n)]^T \tag{12.5}$$

$$\mathbf{e}(n) = [e_n(1)\ e_n(2)\ \ldots\ e_n(n)]^T \tag{12.6}$$

$$\mathbf{X}(n) = [\mathbf{x}(1)\ \mathbf{x}(2)\ \ldots\ \mathbf{x}(n)] \tag{12.7}$$

These help simplify the matrix-based formulation.

Matrix Form of Output and Error

$$\mathbf{y}(n) = \mathbf{X}^T(n)\mathbf{w}(n) \tag{12.8}$$

$$\mathbf{e}(n) = \mathbf{d}(n) - \mathbf{y}(n) \tag{12.9}$$

$$\zeta(n) = \mathbf{e}^T(n)\mathbf{e}(n) \tag{12.10}$$

Substitute expressions into the cost function:
The cost function becomes a quadratic in $\mathbf{w}(n)$.

$$\zeta(n) = \mathbf{d}^T(n)\mathbf{d}(n) - 2\theta^T(n)\mathbf{w}(n) + \mathbf{w}^T(n)\Phi(n)\mathbf{w}(n) \tag{12.11}$$

Where:

$$\Phi(n) = \mathbf{X}(n)\mathbf{X}^T(n) \tag{12.12}$$

$$\theta(n) = \mathbf{X}(n)\mathbf{d}(n) \tag{12.13}$$

Optimal Weights from Normal Equation

Set derivative of $\zeta(n)$ w.r.t. $\mathbf{w}(n)$ to zero:

$$\Phi(n)\hat{\mathbf{w}}(n) = \theta(n) \tag{12.14}$$

$$\hat{\mathbf{w}}(n) = \Phi^{-1}(n)\theta(n) \tag{12.15}$$

Introduce forgetting factor:

$$\rho_n(k) = \lambda^{n-k}, \quad 0 < \lambda \leq 1 \tag{12.33}$$

Then define:

$$\Lambda(n) = \text{diag}(\lambda^{n-1}, \lambda^{n-2}, \ldots, 1) \tag{12.35}$$

Exponentially Weighted Normal Equation

$$\Phi_\lambda(n) = \mathbf{X}(n)\Lambda(n)\mathbf{X}^T(n) \tag{12.37}$$
$$\theta_\lambda(n) = \mathbf{X}(n)\Lambda(n)\mathbf{d}(n) \tag{12.38}$$
$$\hat{\mathbf{w}}(n) = \Phi_\lambda^{-1}(n)\theta_\lambda(n) \tag{12.36}$$

Recursive updates:

$$\Phi_\lambda(n) = \lambda\Phi_\lambda(n-1) + \mathbf{x}(n)\mathbf{x}^T(n) \tag{12.41}$$
$$\theta_\lambda(n) = \lambda\theta_\lambda(n-1) + \mathbf{x}(n)d(n) \tag{12.42}$$

To recursively compute inverse we use matrix inversion lemma :

$$(\mathbf{A} + \alpha\mathbf{a}\mathbf{a}^T)^{-1} = \mathbf{A}^{-1} - \frac{\alpha\mathbf{A}^{-1}\mathbf{a}\mathbf{a}^T\mathbf{A}^{-1}}{1 + \alpha\mathbf{a}^T\mathbf{A}^{-1}\mathbf{a}} \tag{12.43}$$

Gain Vector and Weight Update

Using the matrix inversion lemma, the inverse of the autocorrelation matrix is updated as:

$$\mathbf{P}(n) = \lambda^{-1}\mathbf{P}(n-1) - \frac{\lambda^{-2}\mathbf{P}(n-1)\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{P}(n-1)}{1+\lambda^{-1}\mathbf{x}^T(n)\mathbf{P}(n-1)\mathbf{x}(n)} \tag{12.44}$$

Define gain vector:

$$\mathbf{k}(n) = \frac{\lambda^{-1}\mathbf{P}(n-1)\mathbf{x}(n)}{1+\lambda^{-1}\mathbf{x}^T(n)\mathbf{P}(n-1)\mathbf{x}(n)} \tag{12.45}$$

Substitute the gain vector to simplify the inverse update:

$$\mathbf{P}(n) = \lambda^{-1}\left[\mathbf{P}(n-1) - \mathbf{k}(n)\mathbf{x}^T(n)\mathbf{P}(n-1)\right] \tag{12.46}$$

Rewriting the Gain Vector

The gain vector can also be expressed in terms of current inverse:

$$\mathbf{k}(n) = \lambda^{-1}\left[\mathbf{P}(n-1) - \mathbf{k}(n)\mathbf{x}^T(n)\mathbf{P}(n-1)\right]\mathbf{x}(n) \tag{12.47}$$

Another simplification gives:

$$\mathbf{k}(n) = \mathbf{P}(n)\mathbf{x}(n) \tag{12.48}$$

This is sometimes used directly in the RLS weight update.
From recursive forms of $\theta_\lambda(n)$ and $\mathbf{P}(n)$:

$$\hat{\mathbf{w}}(n) = \lambda\mathbf{P}(n)\theta_\lambda(n-1) + \mathbf{k}(n)d(n) \tag{12.49}$$

Final Recursive Weight Update Form

Rewriting with previous weight vector:

$$\hat{\mathbf{w}}(n) = \hat{\mathbf{w}}(n-1) + \mathbf{k}(n)\left[d(n) - \hat{\mathbf{w}}^T(n-1)\mathbf{x}(n)\right] \tag{12.50}$$

This is the core RLS weight update formula. Simplifying further

$$\hat{\mathbf{w}}(n) = \hat{\mathbf{w}}(n-1) + \mathbf{k}(n)e_{n-1}(n) \tag{12.52}$$

Initialization of the RLS Algorithm

Before starting the RLS iterations, we need to initialize the key variables:

- **Initial inverse correlation matrix:**

$$\mathbf{P}(0) = \delta^{-1}\mathbf{I} \tag{12.55}$$

  where δ is a small positive constant (e.g., 10^{-2}).

- **Initial weight vector:**

$$\hat{\mathbf{w}}(0) = \mathbf{0} \tag{12.57}$$

These settings ensure numerical stability and allow recursive updates to begin.

Second-Order IIR Notch Filter

A **notch filter** removes narrowband tonal interference.
For normalized notch frequency w_0 and bandwidth bw, the pole radius:

$$r = 1 - \frac{bw}{2}$$

The transfer function of a second-order IIR notch filter is:

$$H(z) = \frac{1 - 2\cos(\pi w_0)z^{-1} + z^{-2}}{1 - 2r\cos(\pi w_0)z^{-1} + r^2 z^{-2}}$$

From $H(z) = \frac{B(z)}{A(z)}$, we get the recursive form:

$$y[n] = b_0 x[n] + b_1 x[n-1] + b_2 x[n-2] - a_1 y[n-1] - a_2 y[n-2]$$

with:

$$b_0 = 1, \quad b_1 = -2\cos(\pi w_0), \quad b_2 = 1$$
$$a_1 = -2r\cos(\pi w_0), \quad a_2 = r^2$$