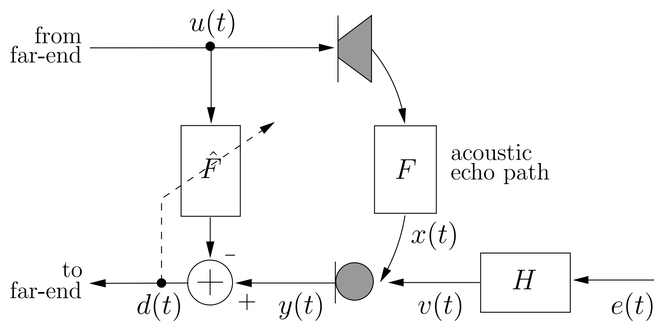
Double-Talk Robust Acoustic Echo Cancellation with Continuous Near-End Activity

Toon van Waterschoot and Marc Moonen
Katholieke Universiteit Leuven, ESAT-SCD, Kasteelpark Arenberg 10, B-3001 Leuven, Belgium
toon.vanwaterschoot@esat.kuleuven.be

1 Introduction

- The acoustic echo cancellation (AEC) problem is defined by
  - the far-end signal $u(t)$,
  - the near-end signal $v(t)$,
  - the room impulse response (RIR) $F(q,t) = f_0(t) + f_1(t)q^{-1} + \ldots + f_{n_F}(t)q^{-n_F}$,
  - the echo signal $x(t) = F(q,t)u(t)$,
  - the microphone signal $y(t) = x(t) + v(t)$,
  - the adaptive filter $\hat{F}(q,t) = \hat{f}_0(t) + \hat{f}_1(t)q^{-1} + \ldots + \hat{f}_{n_F}(t)q^{-n_F}$,
  - the echo-compensated signal $d(t) = y(t) - \hat{F}(q,t)u(t)$.



- A double-talk situation occurs when both the far-end signal $u(t)$ and the near-end signal $v(t)$ are active, and may lead to:
  - slow convergence of the RLS algorithm,
  - divergence of the (N)LMS algorithm.

  The standard solution is to switch off the adaptation during double-talk using a double-talk detector (DTD), which however has two shortcomings:
  - during the time needed to detect double-talk, the convergence of the algorithm may have already been affected considerably,
  - switching off the adaptation does not solve the continuous double-talk problem in which a near-end signal is permanently active.

  Therefore it is desirable to add double-talk robustness to the adaptive algorithms, which is possible by taking into account the near-end signal characteristics.

2 Best linear unbiased estimate

Let us consider the batch linear estimation problem that occurs in AEC, given a data record $\{u(k), y(k)\}_{k=1}^{t}$:

$$\underbrace{\begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(t) \end{bmatrix}}_{\mathbf{y}} = \underbrace{\begin{bmatrix} u(1) & \ldots & u(1-n_F) \\ u(2) & \ldots & u(2-n_F) \\ \vdots & \ddots & \vdots \\ u(t) & \ldots & u(t-n_F) \end{bmatrix}}_{\mathbf{U}} \cdot \underbrace{\begin{bmatrix} f_0 \\ \vdots \\ f_{n_F} \end{bmatrix}}_{\mathbf{f}} + \underbrace{\begin{bmatrix} v(1) \\ v(2) \\ \vdots \\ v(t) \end{bmatrix}}_{\mathbf{v}}. \quad (1)$$

Any linear estimate of parameter vector $\mathbf{f}$ can be written as a linear function of the data vector $\mathbf{y}$:

$$\hat{\mathbf{f}} = \mathbf{Z}^T\mathbf{y}. \quad (2)$$

For this estimate to be unbiased, the $t \times (n_F+1)$ matrix $\mathbf{Z}$ should be subjected to two constraints:

$$\begin{cases} \mathbf{Z}^T\mathbf{U} = \mathbf{I}_{n_F+1} & (a) \\ E\mathbf{Z}^T\mathbf{v} = \mathbf{0}_{(n_F+1)\times 1} & (b) \end{cases} \quad (3)$$

- A typical AEC adaptive algorithm is based on the least-squares (LS) estimator,

$$\hat{\mathbf{f}}_{\mathbf{LS}} = (\mathbf{U}^T\mathbf{U})^{-1}\mathbf{U}^T\mathbf{y}, \quad (4)$$

  which is unbiased but not necessarily minimum-variance.
- Minimizing the variance $E(\hat{\mathbf{f}} - E\hat{\mathbf{f}})(\hat{\mathbf{f}} - E\hat{\mathbf{f}})^T$ of the estimate (2) under the unbiasedness constraint (3(a)) yields the best linear unbiased estimate (BLUE):

$$\hat{\mathbf{f}}_{\mathbf{BLUE}} = (\mathbf{U}^T\mathbf{R}^{-1}\mathbf{U})^{-1}\mathbf{U}^T\mathbf{R}^{-1}\mathbf{y}, \quad (5)$$

  with $\mathbf{R}$ the near-end signal correlation matrix, defined by

$$\mathbf{R} \triangleq E\mathbf{v}\mathbf{v}^T. \quad (6)$$

References

[1] G. Rombouts, T. van Waterschoot, K. Struyve, and M. Moonen, "Acoustic feedback cancellation for long acoustic paths using a nonstationary source model," in *Proceedings of the 13th European Signal Processing Conference (EUSIPCO-2005)*, Antalya, Turkey, September 4-8, 2005.

3 Prediction error identification

- Assume that the near-end signal $v(t)$ is generated as

$$v(t) = H(q,t)e(t) \quad \text{with} \quad Ee(t)e(t-k) = \delta(k)\sigma_t^2. \quad (7)$$

The BLUE in (5) can then be realized as

$$\hat{\mathbf{f}}_{\mathbf{BLUE}} = (\mathbf{U}^T\mathbf{H}^{-T}\mathbf{\Lambda}^{-1}\mathbf{H}^{-1}\mathbf{U})^{-1}\mathbf{U}^T\mathbf{H}^{-T}\mathbf{\Lambda}^{-1}\mathbf{H}^{-1}\mathbf{y}, \quad (8)$$

with the prefiltering matrix $\mathbf{H}$ and weighting matrix $\mathbf{\Lambda}$ defined as

$$\mathbf{H} = \mathbf{H}^T \triangleq \begin{bmatrix} H(q,1) & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & H(q,t) \end{bmatrix} \quad \text{and} \quad \Lambda \triangleq \begin{bmatrix} \sigma_1^2 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \sigma_t^2 \end{bmatrix}.$$

If the near-end signal is described using an autoregressive model,

$$H(q,t) = \frac{1}{A(q,t)} = \frac{1}{1 + a_1(t)q^{-1} + \ldots + a_{n_A}(t)q^{-n_A}},$$

then the prefilters $H^{-1}(q,k) = A(q,k)$, $k = 1 \ldots t$ are FIR filters of order n_A.

- The BLUE is then also the minimizing estimator of the prediction error criterion

$$V_{PE}(t, \mathbf{f}(t), \mathbf{a}(t), \sigma_t^2) = \frac{1}{2N}\sum_{k=1}^{t} \frac{\lambda^{t-k}}{\sigma_k^2}\big(A(q,k)[y(k) - F(q,t)u(k)]\big)^2,$$

with $\mathbf{a}(t) \triangleq [a_1(t) \ \ldots \ a_{n_A}(t)]^T$. This criterion can be minimized recursively using the two-stage PEM-AFROW algorithm [1]:

First stage: linear prediction of the echo-compensated signal $d(t, \hat{\mathbf{f}}(t-1))$, calculated using the previous estimate $\hat{\mathbf{f}}(t-1)$ on a rectangular hopping window of length M that 'looks ahead' $P-1$ samples:

$$\mathbf{d}(t) = \begin{bmatrix} y(t+P-1) \\ \vdots \\ y(t+P-M) \end{bmatrix} - \begin{bmatrix} u(t+P-1) & \ldots & u(t+P-1-n_F) \\ \vdots & \ddots & \vdots \\ u(t+P-M) & \ldots & u(t+P-M-n_F) \end{bmatrix} \hat{\mathbf{f}}(t-1).$$

The autocorrelation functions $\phi_{dd}(\tau)$, $\tau = 0 \ldots n_A$, of $d(t, \hat{\mathbf{f}}(t-1))$ are estimated using the autocorrelation method:

$$\begin{bmatrix} \hat{\phi}_{dd}(0) \\ \hat{\phi}_{dd}(1) \\ \vdots \\ \hat{\phi}_{dd}(n_A) \end{bmatrix} = \begin{bmatrix} 0 & \ldots & d(t) & \ldots & d(t-M+1) \\ 0 & \ldots & d(t-1) & \ldots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ d(t) & \ldots & d(t-n_A) & \ldots & 0 \end{bmatrix} \begin{bmatrix} 0 \\ \vdots \\ d(t) \\ \vdots \\ d(t-M+1) \end{bmatrix}$$

The near-end signal AR coefficients $\mathbf{a}(t)$ and the near-end excitation signal variance σ_t^2 are then estimated from $\hat{\phi}_{dd}(\tau)$, $\tau = 0 \ldots n_A$, using the Levinson-Durbin recursion.

Second stage: recursive update of the RIR estimate using the loudspeaker and microphone signals prefiltered with the estimated coefficients $\hat{\mathbf{a}}(t)$ from the first stage:

$$y_A(t) = \begin{bmatrix} y(t) & \ldots & y(t-n_A) \end{bmatrix} \begin{bmatrix} 1 \\ \hat{\mathbf{a}}(t) \end{bmatrix},$$

$$\mathbf{u_A}(t) = \begin{bmatrix} u(t) & \ldots & u(t-n_A) \\ \vdots & \ddots & \vdots \\ u(t-n_F) & \ldots & u(t-n_F-n_A) \end{bmatrix} \begin{bmatrix} 1 \\ \hat{\mathbf{a}}(t) \end{bmatrix}.$$

The RIR estimate $\hat{\mathbf{f}}(t-1)$ can then be updated recursively, either with the Gauss-Newton method:

$$\begin{aligned} \hat{\mathbf{f}}(t) &= \hat{\mathbf{f}}(t-1) + \frac{1}{\hat{\sigma}_t^2}\mathbf{R_f}^{-1}(t)\mathbf{u_A}(t)\varepsilon_p(t), \\ \mathbf{R_f}(t) &= \lambda\mathbf{R_f}(t-1) + \frac{1}{\hat{\sigma}_t^2}\mathbf{u_A}(t)\mathbf{u_A}^T(t), \end{aligned} \quad (8)$$

or with the stochastic gradient method:

$$\hat{\mathbf{f}}(t) = \hat{\mathbf{f}}(t-1) + \mu\frac{\mathbf{u_A}(t)\varepsilon_p(t)}{\mathbf{u_A}^T(t)\mathbf{u_A}(t) + (n_F+1)\hat{\sigma}_t^2} \quad (9)$$

where in both cases weighting is performed using the estimated variance $\hat{\sigma}_t^2$ from the first stage, and the a priori prediction error is calculated as

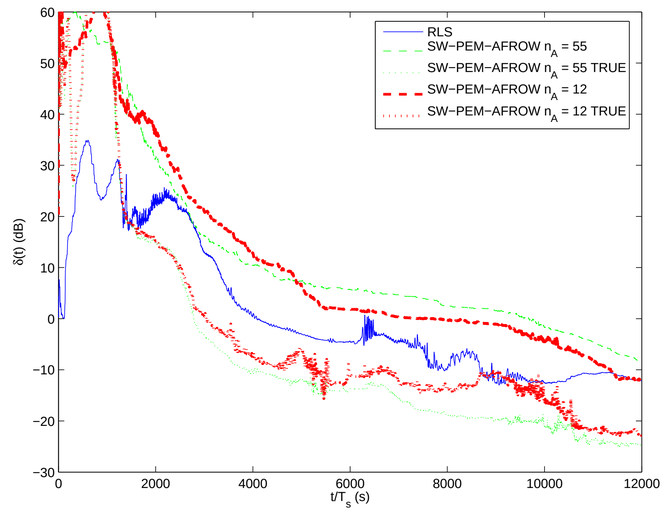
$$\varepsilon_p(t) = \varepsilon(t, \hat{\mathbf{f}}(t-1), \hat{\mathbf{a}}(t)) = y_A(t) - \mathbf{u_A}^T(t)\hat{\mathbf{f}}(t-1).$$

4 Simulation results

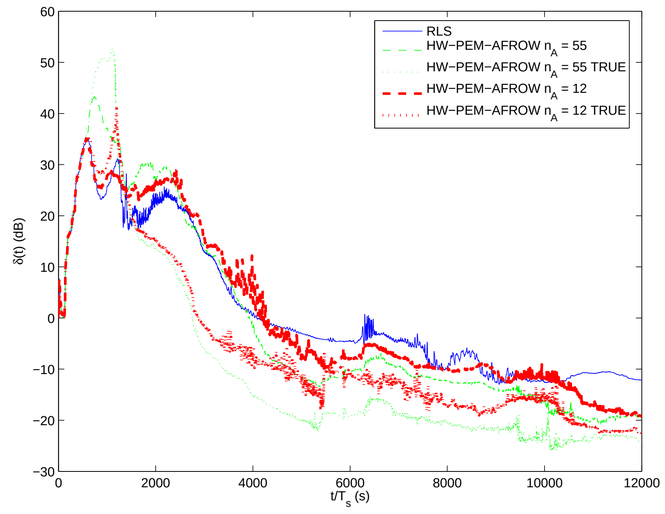
- Simulation parameters: $f_s = 8kHz$, $n_F + 1 = 1000$, $n_A = 12$ or 55, $\lambda = 0.9997$, $\mu = 0.5$, $M = 215$.
- Echo-to-background ratio: $EBR \triangleq \frac{\sum_{k=1}^{N}|x(k)|^2}{\sum_{k=1}^{N}|v(k)|^2} = 10dB$.
- Performance measure: $\delta(t) = 20\log_{10}\frac{\|\hat{\mathbf{f}}(t) - \mathbf{f}\|}{\|\mathbf{f}\|}$.
- 'TRUE': knowledge of the clean near-end signal is assumed

4.1 Gauss-Newton type algorithms

- Sliding window: $P = 1$

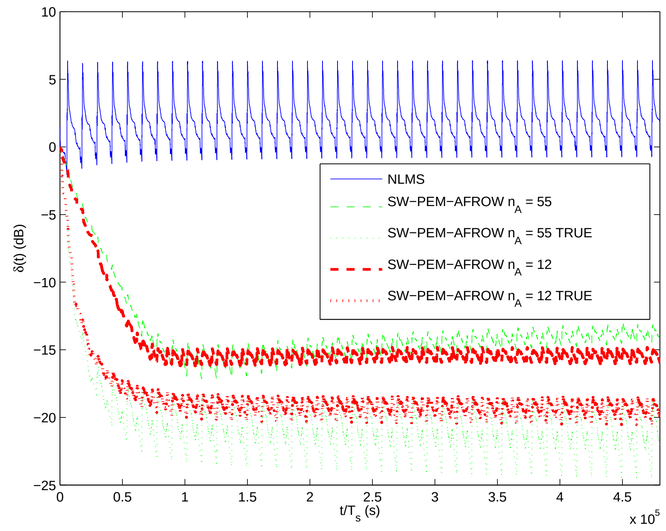


- Hopping window: $P = M - n_A$



4.2 Stochastic gradient algorithms

- Sliding window: $P = 1$



- Hopping window: $P = M - n_A$

