# Active noise control with fast array recursive least squares filters using a parallel implementation for numerical stability

Arthur Berkhoff[1,2] and Sjoerd van Ophem[1]
[1] Faculty of Engineering Technology, University of Twente, Drienerlolaan 5, 7522NB Enschede,
P.O. Box 217, 7500AE Enschede, The Netherlands, a.p.berkhoff@utwente.nl
[2] Acoustics and Sonar group, TNO Technical Sciences, Oude Waalsdorperweg 63, 2597AK The Hague,
P.O. Box 96864, 2509JG The Hague, The Netherlands, arthur.berkhoff@tno.nl

**Summary**
Noise reduction in feedforward active noise control systems with a rapidly changing primary path requires rapid convergence and fast tracking. This can be accomplished with a fast-array Kalman method which uses an efficient rotation matrix technique to calculate the filter parameters. However, finite precision effects lead to unstable behavior. In this paper results of a recent algorithm are presented, which exhibits the fast convergence, tracking properties and the linear calculation complexity of the fast array Kalman method, but which does not suffer from the numerical problems. This is achieved by using a convex combination of two parallel finite length growing memory recursive least squares filters. A periodic reset of the filter parameters with proper re-initialization is enforced, preventing the numerical instability. The performance of the algorithm is demonstrated in numerical simulations and in real-time experiments. Convergence rate and tracking performance are similar to that of a fast-array sliding window recursive least squares algorithm, while eliminating the numerical issues. It is shown that the new algorithm provides significantly improved convergence and tracking as compared to more traditional algorithms, such as based on the filtered reference least mean squares algorithm.

**PACS no. 43.50.Ki, 43.60.Ac, 43.60.Mn**

## 1. Introduction[1]

The main reason for the low convergence rate of the fxLMS algorithm is the assumption that both the adaptive filter and the secondary path estimate are Linear Time Invariant (LTI) and therefore can be interchanged [2]. This assumption only holds if the adaptive filter changes slowly in comparison to the secondary path dynamics. Nevertheless, to improve the convergence rate, multiple changes to the fxLMS algorithm have been proposed in the literature, for example the modified fxLMS [3], fast affine projections [4] and preconditioned LMS [2]. Another way to improve the rate of convergence can be obtained by reformulating the ANC problem as a state estimation problem, as has been proposed by Sayyarrodsari et al. [5] .

The assumption of the LTI adaptive filter and secondary path also potentially influences the tracking performance of primary path changes. These changes will occur when the primary noise source is moving relative to the ANC system or when the reference microphone is moving. Some examples of moving noise sources are airplanes and cars. With such noise sources, the primary path can change rapidly, violating the assumption of a system with slowly varying dynamics. Some examples of ANC with moving noise sources are given by Omoto and Fujiwara [6], Berkhoff [7] and Van Ophem and Berkhoff [8].

A real-time implementation of a fast array Kalman filter [9, 10] was presented by Van Ophem and Berkhoff [8]. In this implementation an output normal parameterization of the estimated secondary path was used to reduce the amount of floating point operations and to remove redundancy from the state space model. Although the fast array Kalman filter shows the desired high rate of convergence, it was shown by Van Ophem and Berkhoff [8] that the tracking performance is diminishing with progressing time. The reason for this behavior is that the Kalman filter uses all old data to calculate the estimate of the filter coefficients. Therefore, a logical way to improve tracking would be a mechanism which throws away old data in the recursions. In Fraanje et al. [10] a forgetting factor, which exponentially weights the data, was proposed to improve the tracking performance. Although the improved tracking

[1] Text and figures of the present paper are based on S. van Ophem and A.P. Berkhoff, A numerically stable, finite memory, fast array recursive least squares filter for broadband active noise control, International Journal of Adaptive Control and Signal Processing, submitted (2014) [1]

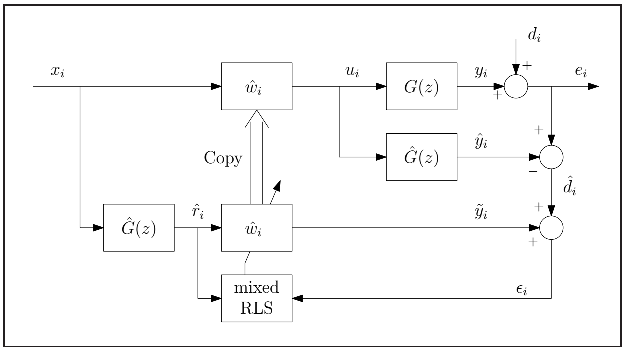

Figure 1. Modified filtered-RLS.

was observed by Van Ophem and Berkhoff [8], a disastrous instability caused by the round-off errors in digital systems, was also observed by the authors [8]. An alternative solution for improving the tracking performance has been proposed by Sayed [9] in the form of a sliding window. A description of this filter in fast array form has been described by Park et al. [11].

The sliding window RLS algorithm works by running two RLS filters in parallel. The first filter works as a standard growing memory RLS filter, but the second filter throws away old information, which results in a finite memory filter. It was found [12] that this filter also suffers from round-off errors, especially in single precision floating point arithmetic, but with a linear error growth, as opposed to the exponential error growth with a forgetting factor.

This paper describes results of a Single Input Single Output (SISO) ANC algorithm presented in Ref. [1] having a rate of convergence and tracking performance similar to that of a fast array sliding window RLS filter, but without the numerical error growth. Results of the algorithm are given in simulations and in real-time experiments.

## 2. Methods

### 2.1. Modified filtered-RLS

In this paper a SISO ANC system with a modified structure is considered. A block diagram of this system is shown Fig. 1. The goal of the adaptive filter is to find a set of Finite Impulse Response (FIR) filter coefficients $\hat{w}_i \in \mathbb{R}^{n_w}$, which minimize the modified error $\epsilon_i$. This modified error will be calculated by summing the estimated disturbance $\hat{d}_i$ and the output of the adaptive filter $\tilde{y}_i$:

$$\epsilon_i = \hat{d}_i + \tilde{y}_i. \tag{1}$$

The output of the adaptive filter is calculated by multiplying the filtered reference signal $\hat{r}_i$ with the filter coefficients $\hat{w}_i$:

$$\tilde{y}_i = -\hat{r}_{n_w,i}^T \hat{w}_i, \tag{2}$$

in which $\hat{r}_{n_w,i}$ is a vector with the last $n_w$ values of the filtered reference signal:

$$\hat{r}_{n_w,i} = \begin{bmatrix} \hat{r}_i \ \hat{r}_{i-1} \ \cdots \ \hat{r}_{i-n_w+1} \end{bmatrix}^T. \tag{3}$$

The filtered reference signal is calculated by filtering the measured reference signal with the estimated secondary path state space model:

$$\theta_{i+1}^r = A_s \theta_i^r + B_s x_i, \tag{4}$$
$$\hat{r}_i = C_s \theta_i^r + D_s x_i, \tag{5}$$

in which $\theta_i^r$ is the internal path state and $A_s,\ B_s,\ C_s$ and $D_s$ are the estimated secondary path state matrices.

The estimated value of the disturbance is calculated by subtracting the estimated output $\hat{y}_i$ of the secondary path $\hat{G}(z)$ from the measured error $e_i$:

$$\hat{d}_i = e_i - \hat{y}_i. \tag{6}$$

The estimated output of the secondary path is calculated by filtering the control signal $u_i$ with the estimated state space model of the secondary path:

$$\hat{\theta}_{i+1} = A_s \hat{\theta}_i + B_s u_i, \tag{7}$$
$$\hat{y}_i = C_s \hat{\theta}_i + D_s u_i, \tag{8}$$

The control signal $u_i$ is calculated by filtering the reference signal $x_i$ with the adaptive filter, as follows

$$u_i = x_{n_w}^T \hat{w}_i, \tag{9}$$
$$x_{n_w} = \begin{bmatrix} x_i \ x_{i-1} \ \cdots \ x_{i-n_w+1} \end{bmatrix}^T. \tag{10}$$

This filter structure is well known in the context of ANC and has been applied both to filtered-reference LMS and RLS algorithms [2], [10].

### 2.2. Mixed windowed RLS

For the adaption of the filter coefficients we propose a filter which behaves like a constant length finite memory RLS algorithm with a linear calculation complexity $O(n_w)$, equivalent to the Chandrasekhar form of the sliding window RLS filter [11], but does not exhibit the round-off error propagation. To achieve this, a convex mixing approach is used to emulate the sliding window RLS filter.

Convex combinations of filters have been a popular topic in recent years, see Refs. [13], [14] and [15]. An example of convex filters in the context of ANC is given by Ferrer [16]. The main difference between these approaches and the proposed filter in this paper, is the way the convex combination is applied. In the literature the convex combinations of the filters are used to mix two filters, which have different filter parameters, such as forgetting factors and convergence coefficients. The optimal mixing parameters, which give the lowest MSE, then will be determined by an extra adaptive filter.

The proposed implementation will use two filters with identical filter parameters and predetermined time-varying mixing coefficients. This means that not necessarily the convex combination with the lowest MSE will be found. Instead, it simulates a filter with a constant memory length, such as the sliding window RLS filter.

Two parallel growing memory filters are mixed in such a way, that the total available information used for calculating the least squares solution will be equal at every time instance.

Firstly, the equations for a recursive update of the mixed solution are presented. The mixing parameters $\alpha_i$ and $\beta_i$ are constrained by $0 \leq \alpha_i \leq 1$ , $0 \leq \beta_i \leq 1$, and sum up to unity:

$$\alpha_i + \beta_i = 1, \; \forall i. \tag{11}$$

A possible choice for the mixing parameters can be found in Fig. 2.

Consider two parallel RLS filters, both with a growing data window bounded to $W$ entries. The first filter will be activated at time instance $U$ and the second filter will be activated after $V = U + W/2$ iterations. The filters have the following data matrices $H_i$, $H_{V:i}$ and measurement vectors $y_i$, $y_{V:i}$, with $V < i < W$ :

$$H_i = \begin{bmatrix} \hat{r}^T_{n_w,U} \\ \hat{r}^T_{n_w,U+1} \\ \vdots \\ \hat{r}^T_{n_w,i} \end{bmatrix}, \; H_{V:i} = \begin{bmatrix} \hat{r}^T_{n_w,V} \\ \hat{r}^T_{n_w,V+1} \\ \vdots \\ \hat{r}^T_{n_w,i} \end{bmatrix}. \tag{12}$$

$$y_i = \begin{bmatrix} \hat{d}_U \\ \hat{d}_{U+1} \\ \vdots \\ \hat{d}_i \end{bmatrix}, \; y_{V:i} = \begin{bmatrix} \hat{d}_V \\ \hat{d}_{V+1} \\ \vdots \\ \hat{d}_i \end{bmatrix}. \tag{13}$$

The cost functions of the parallel RLS filters are [9]:

$$\min_{w_i}[w_i^T \Pi w_i + \|y_i - H_i w_i\|^2], \tag{14}$$

$$\min_{w_{V:i}}[w_{V:i}^T \Pi w_{V:i} + \|y_{V:i} - H_{V:i} w_{V:i}\|^2], \tag{15}$$

in which the matrix $\Pi \in \mathbb{R}^{n_w \times n_w}$ is a positive definitive regularization matrix. In Ref. [1] it is shown that the resulting update equations are:

$$\hat{w}_{mix,i} = \alpha_i(\hat{w}_{i-1} + K_i R_i^{-1} \epsilon_i) + \\ \beta_i(\hat{w}_{V:i-1} + K_{V:i} R_{V:i}^{-1} \epsilon_{V:i}). \tag{16}$$

in which $K_i$ is the Kalman gain, $\epsilon_i$ is the innovation and $R_i$ is the expected value of the innovation.

### 2.3. Fast array formulation

The parameters in Eq. (16) will be updated with two parallel growing memory (forgetting factor $\lambda = 1$) fast array RLS algorithms. The complexity of these algorithms grows linearly with $n_w$. This linear complexity is achieved by updating the difference of the state error covariance matrices $P_i$ between time instances $i-1$ and $i$. It assumes that this difference can be factorized as follows[9]:

$$dP_i = P_i - \Psi P_{i-1} \Psi^T = L_i M_i L_i^T. \tag{17}$$

In this equation, $\Psi$ is a first diagonal shift matrix. For a system with a shift-invariant input signal, like the proposed ANC system, the rank of the matrix $M$ can be as low as 2. Since an extensive derivation for the update equations of a fast array RLS algorithm is available in the literature, see Refs. [10], [9], we will simply state the results from the literature and the resulting filter equations.

The filter parameters of the two filters will be calculated completely in parallel and no interaction will take place between the filters. Both filters will be reset every time a window length $W$ has passed. A complete description of the algorithm can be found in Ref. [1].

## 3. Results and discussion

The performance of the new approach was tested both numerically and experimentally. Firstly, the numerical performance was tested and compared to a sliding window RLS filter. The numerical experiments were done with both measurement data from a duct and synthetic data. For all the experiments a sampling frequency of $f_s = 2000\ Hz$ was used.

### 3.1. Comparison with growing memory RLS

For the first simulation, the convergence and tracking of the present method were compared to a fast array RLS filter, with a forgetting factor of $\lambda = 1$. Synthetic primary and secondary paths were used to calculate the reference signal, the error signal and the resulting control signal. These paths result from a 1D acoustic model of a duct, with a white noise source. The filter contains $n_w = 250$ coefficients.

The tracking behavior was tested by changing the simulation position of the reference sensor after 10 seconds. For the mixed windowed RLS filter a data window of length $W = 6000$ was chosen. The results are shown in Fig. 3. It is clear that the new algorithm outperforms the growing memory RLS filter when it is used for tracking purposes. However, it is more interesting to compare the results of the present approach with a fast array sliding window RLS algorithm, such as described by Park et al. [11].

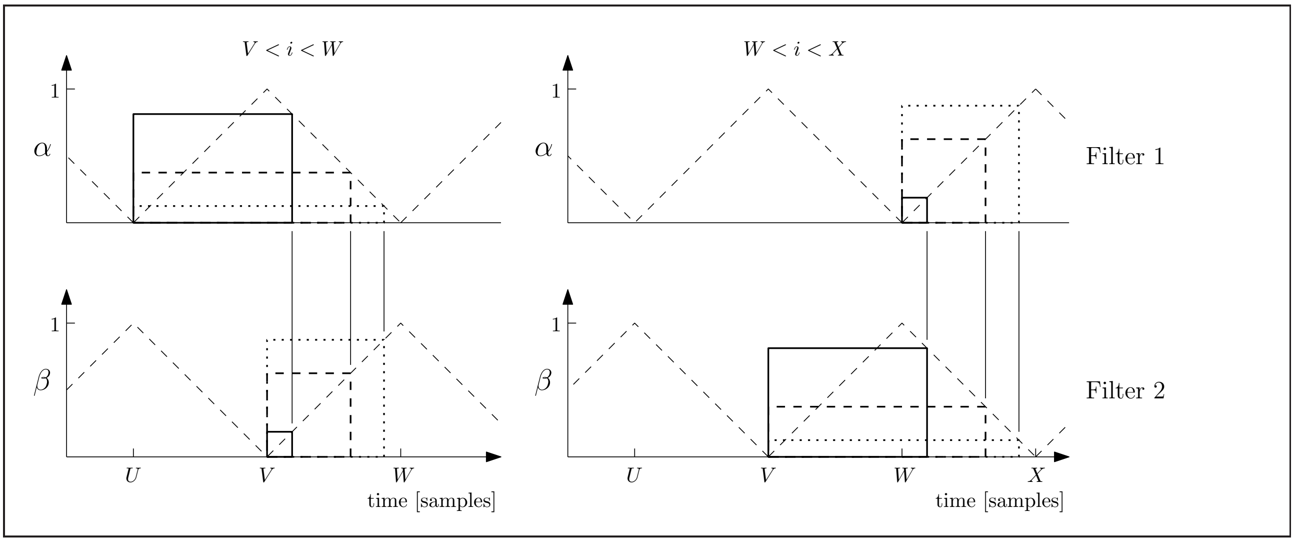

Figure 2. Example of the mixing parameters $\alpha_i$ and $\beta_i$.

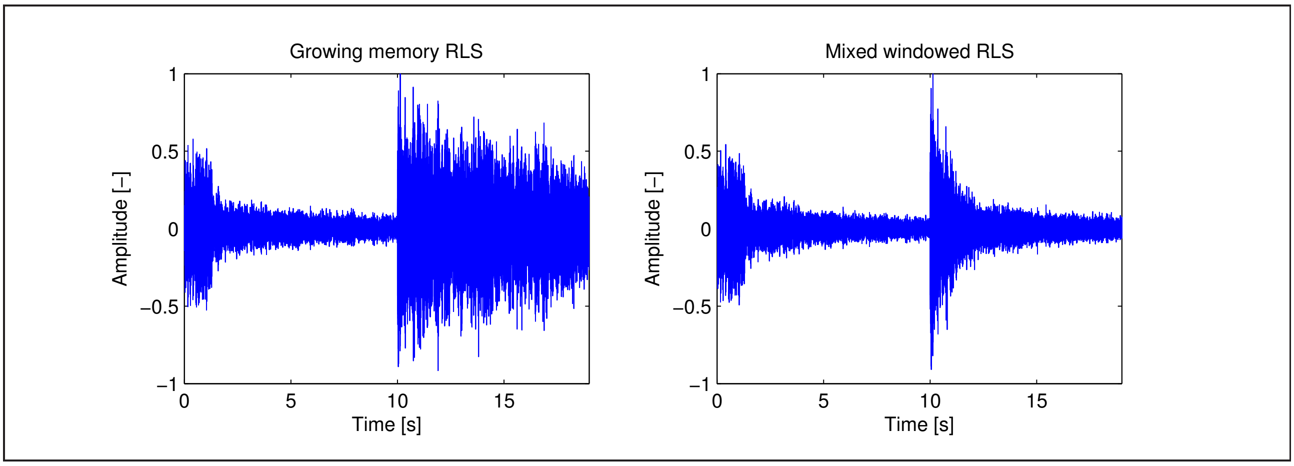

Figure 3. Convergence and tracking performance of the growing memory RLS filter (left) and the mixed windowed RLS filter of data length $W = 6000$ (right). The filter is activated after 1 second and the tracking performance is checked by shifting the reference signal after 10 seconds.

### 3.2. Comparison with fast-array sliding window RLS

For the comparison of the mixed windowed RLS with the fast-array sliding window RLS two cases were considered: A comparison of the convergence and tracking properties and a comparison of the long term numerical behavior. The rotation matrix of the sliding window RLS filter has been calculated with the orthogonal diagonal method [17], because of its improved numerical behavior as compared to hyperbolic Givens rotations.

In Fig. 4 the fast array sliding window RLS and the new filter are compared. For these simulations the data window of the new filter was set to two times the length of the fast array sliding window RLS filter. Just as with the comparison of the new filter with the growing memory, the tracking performance was tested by changing the reference signal after 10 seconds and the results were averaged over 200 simulations. To obtain a good comparison, the filter coefficients of the new filter were set to zero at every reset point. This was done to emulate the behavior of the downdate step of the sliding window algorithm.

It can be seen that the mixed window RLS filter approximates the sliding window RLS filter, but that the MSE is not as smooth. Closer inspection shows that this variation in the MSE is related to the window length, so this is an artifact of the mixing scheme. Further tests show that the amplitude of the fluctuation depends on the magnitude of the regularization coefficient $\delta$. A high value of $\delta$ causes an overshoot when the filter converges and since at every reset one of the parallel filters has to converge again, this can cause a higher MSE. A possible solution to overcome this overshoot would be to incorporate the uncertainty in the secondary path estimates, as described by Fraanje et al. [10].

### 3.3. Numerical behavior

Even when the stable orthogonal diagonal method is used to perform the hyperbolic rotations, the fast array sliding

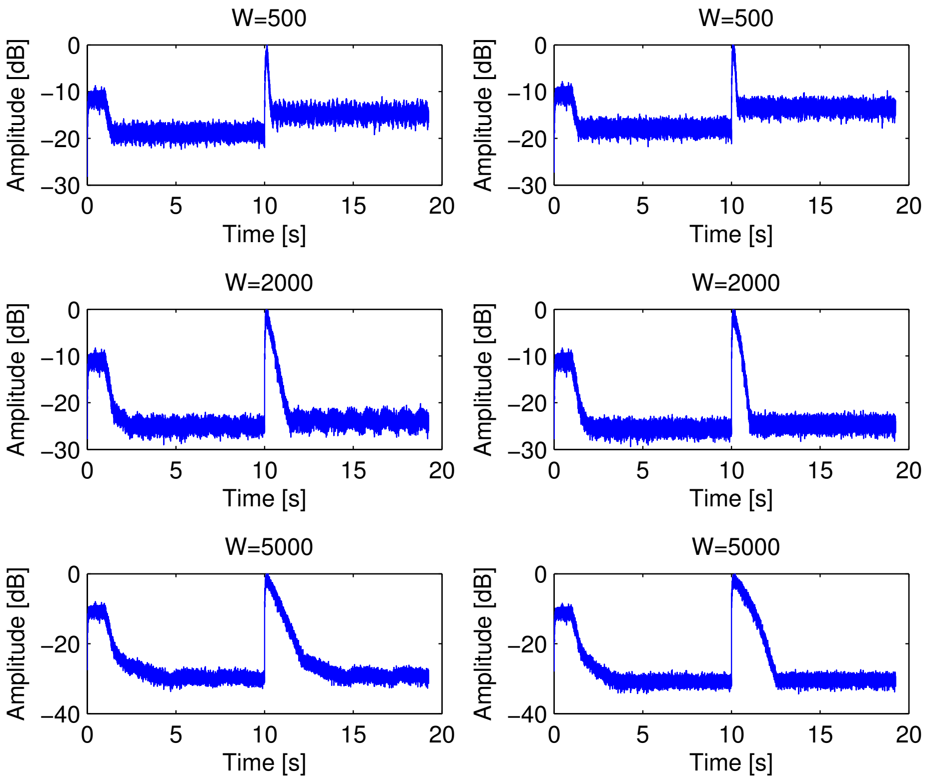

Figure 4. The convergence and tracking performance of the mixed windowed RLS filter (left) and the fast array sliding window RLS (right) for different window lengths, averaged over 200 simulations. The filter coefficients are reset to zero.

window RLS filter exhibits a linear error growth, which means that the performance of the algorithm will deteriorate, when it runs through a large number of recursions. To keep this numerical inaccuracy within bounds, a reset of the algorithm is needed. This means that a third filter has to run in parallel with the sliding window filter, when the reset is applied. The mixed windowed RLS filter does not need this extra filter.

This is shown in Fig. 5, where the results of simulations with both double and single floating point precision are shown for both filters. This simulation uses time-invariant data, so it is expected that the average of the filter parameters should converge to a certain solution and must not deviate. It can be seen that the value $R_i^{(2)}$ of the sliding window fast array RLS filter starts to deviate after about $5e5$ iterations. The weighted average value of $R_{i,new} = \alpha_i R_i^{(1)} + \beta_i R_i^{(2)}$ stays constant in both single and double floating point precision. Similar comparisons of the numerical accuracy of the fast array sliding window RLS algorithm with and without a third parallel filter have been done by Van Ophem and Berkhoff [12].

From the numerical simulations it can be concluded that both the new filter and the fast-array sliding window RLS filter have their benefits, but it is has to be noted that the numerical problems of the sliding window fast array RLS cannot be overcome without adding a third parallel filter, while the fluctuations in the MSE of the new filter are controllable by tuning the filter parameters, such as the regularization term $\delta$.

### 3.4. Experimental results

The algorithm was also tested in a real-time environment. For the experiment a duct was used, which was closed on the left hand side and open on the other side. A sound source, emitting white noise, was placed in the pipe on the left hand side and the goal of the experiment was to minimize the sound pressure at the open end of the pipe by using feed forward control. This was done by sending a control signal to the secondary loudspeaker, placed in duct at about 45 cm from the end of the duct. An error microphone was placed at the open end. The details of the control platform are specified in [4]. The secondary path identification was done off-line, by using a sub-space identification algorithm in the SLICOT libraries. This led to estimates with a variance accounted for value of about $99.8\%$. A digital reference signal was used, so that no feedback from the actuator to the reference signal would occur. The experimental results were in agreement with the simulation results [1].

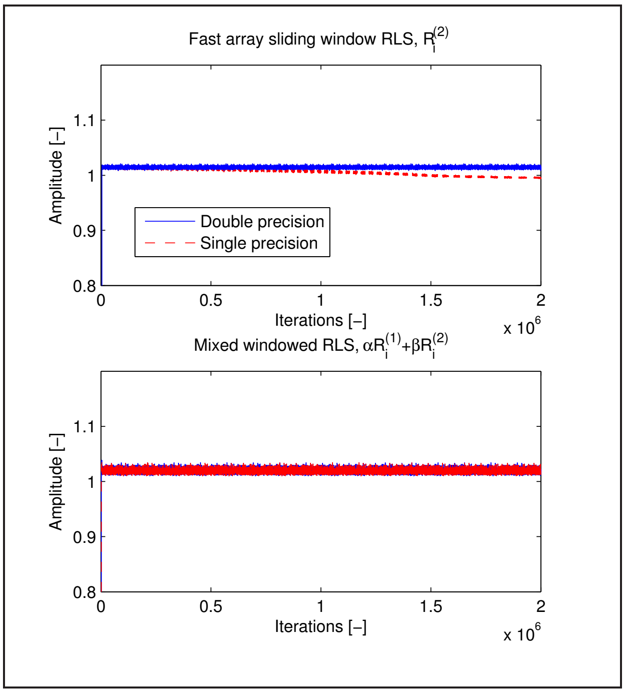

Figure 5. The numerical performance of the sliding window fast-array RLS algorithm, indicated by $R_i^{(2)}$ (top) and the mixed windowed RLS filter, indicated by $\alpha_i R_i^{(1)} + \beta_i R_i^{(2)}$ (bottom) in single and double point floating precision.

## 4. Conclusions

A new algorithm for feedforward active noise control has been presented which has the fast convergence and tracking properties of the sliding window Resursive Least Squares filter and which has a stable numerical implementation. The algorithm has a calculation complexity which is linear with the number of filter parameters. The stable numerical implementation is obtained by using a convex mixing scheme of the filter coefficients, resulting from two parallel finite length, growing memory fast-array RLS filters. This mixing scheme is chosen in such a way that the amount of data used for the calculation of the control signal remains constant.

Although the approximation of the fast-array sliding window RLS is not perfect, especially for longer window lengths, the advantage of this algorithm is the elimination of long term round-off error propagation without adding any redundancy, in contrast to the fast-array sliding window RLS filter. The performance of the filter has been validated in both numerical and experimental tests.

### References

[1] van Ophem S, Berkhoff A. A numerically stable, finite memory, fast array recursive least squares filter for broadband active noise control. *International Journal of Adaptive Control and Signal Processing* 2014 (submitted); .

[2] Elliott SJ. Signal processing for active control. Academic Press: London, 2001; 124.

[3] Bjarnason E. Active noise cancellation using a modified form of the filtered-x LMS algorithm. *Proceedings of Eusipco 92, 6th European Signal Processing Conference* 1992; :1053–1056.

[4] Wesselink JM, Berkhoff AP. Fast affine projections and the regularized modified filtered-error algorithm in multichannel active noise control. *The Journal of the Acoustical Society of America* 2008; **124**(2):949–960, doi: 10.1121/1.2945169.

[5] Sayyarrodsari B, How JP, Hassabi B, Carrier A. Estimation-based synthesis of $h_\infty$-optimal adaptive FIR filters for filtered-LMS problems. *IEEE Transactions on Signal Processing* 2001; **49**(1):164–178, doi:10.1109/78.890358.

[6] Omoto A, Fujiwara K. Behavior of adaptive algorithms in active noise control systems with moving noise sources. *Acoustical Science and Technology* 2002; **23**(2):84–89, doi: 10.1250/ast.23.84.

[7] Berkhoff AP. Control strategies for active noise barriers using near-field error sensing. *The Journal of the Acoustical Society of America* 2005; **118**(3):1469, doi: 10.1121/1.1992787.

[8] Van Ophem S, Berkhoff AP. Multi-channel kalman filters for active noise control. *The Journal of the Acoustical Society of America* 2013; **133**(4):2105–2115, doi: 10.1121/1.4792646.

[9] Sayed AH. Fundamentals of adaptive filtering. Wiley: NY, 2003; 732–873.

[10] Fraanje R, Sayed AH, Verhaegen M, Doelman NJ. A fast-array kalman filter solution to active noise control. *International Journal of Adaptive Control and Signal Processing* 2005; **19**(2-3):125–152, doi:10.1002/acs.841.

[11] Park P, Cho YM, Kailath T. Chandrasekhar recursion for structured time-varying systems and its application to recursive least squares problems. *Second IEEE Conference on Control Applications*, 1993; 797–803 vol.2, doi: 10.1109/CCA.1993.348233.

[12] Van Ophem S, Berkhoff AP. Active control of time-varying broadband noise and vibrations using a sliding-window kalman filter. *Proceedings of the International Conference on Noise and Vibration Engineering, ISMA*, Leuven, Belgium, 2014.

[13] Arenas-Garcia J, Figueiras-Vidal A, Sayed A. Mean-square performance of a convex combination of two adaptive filters. *IEEE Transactions on Signal Processing* 2006; **54**(3):107–1090, doi:10.1109/TSP.2005.863126.

[14] Silva MTM, Nascimento VH. Improving the tracking capability of adaptive filters via convex combination. *IEEE Transactions on Signal Processing* Jul 2008; **56**(7):3137–3149, doi:10.1109/TSP.2008.919105.

[15] Bershad N, Bermudez J, Tourneret JY. An affine combination of two LMS adaptive filters - transient mean-square analysis. *IEEE Transactions on Signal Processing* May 2008; **56**(5):1853–1864, doi:10.1109/TSP.2007.911486.

[16] Ferrer M, Gonzalez A, De Diego M, Pinero G. Convex combination filtered-x algorithms for active noise control systems. *IEEE Transactions on Audio, Speech, and Language Processing* 2013; **21**(1):156–167, doi: 10.1109/TASL.2012.2215595.

[17] Chandrasekaran S, Sayed AH. Stabilizing the generalized schur algorithm. *SIAM Journal on Matrix Analysis and Applications* Oct 1996; **17**(4):950–983, doi: 10.1137/S0895479895287419.