A Split Kernel Adaptive Filtering Architecture for Nonlinear Acoustic Echo Cancellation

Steven Van Vaerenbergh
Dept. Communications Engineering
University of Cantabria, Spain
steven.vanvaerenbergh@unican.es

Luis A. Azpicueta-Ruiz
Dept. Signal Theory and Communic.
Universidad Carlos III de Madrid, Spain
azpicueta@tsc.uc3m.es

Danilo Comminiello
Dept. Information Eng., Electr., Telecom.
Sapienza University of Rome, Italy
danilo.comminiello@uniroma1.it

Abstract—**We propose a new linear-in-the-parameters (LIP) nonlinear filter based on kernel methods to address the problem of nonlinear acoustic echo cancellation (NAEC). For this purpose we define a framework based on a parallel scheme in which any kernel-based adaptive filter (KAF) can be incorporated efficiently. This structure is composed of a classic adaptive filter on one branch, committed to estimating the linear part of the echo path, and a kernel adaptive filter on the other branch, to model the nonlinearities rebounding in the echo path. In addition, we propose a novel low-complexity least mean square (LMS) KAF with very few parameters, to be used in the parallel architecture. Finally, we demonstrate the effectiveness of the proposed scheme in real NAEC scenarios, for different choices of the KAF.**

I. Introduction

Acoustic echo cancellation is still an open problem, mainly due to the widespread use of mobile and hands-free devices. These systems typically include low-cost transducers that generate non-negligible nonlinear distortions. In such cases, exclusively linear solutions are often not able to reduce the echo signals, at the expense of the speech communication. Nonlinear models are therefore becoming a key part of the echo canceller.

A typical single-channel nonlinear acoustic echo cancellation (NAEC) scenario is depicted in Fig. 1. The canceller has access to two electric signals: the audio input signal $x[n]$ generated by the far-end user, which will be transduced and propagated; and the microphone output $d[n]$, modeled as:

$$d[n] = y_h[n] + e_0[n], \tag{1}$$

where $e_0[n]$ represents additive noise at microphone location, and $y_h[n]$ corresponds to the signal radiated by the loudspeaker and propagated throughout the room from the emitter to the microphone. Although the room propagation is an eminently linear process that can be described by the acoustic impulse response (AIR) $h[n]$, the transduction and amplification processes can generate some nonlinear distortion of the signal $x[n]$. The aim of the canceller is to generate a replica of $y_h[n]$, denoted $y[n]$, to be subtracted from the microphone signal $d[n]$ in order to improve the communication.

Several different NAEC approaches have been proposed in the recent literature. Some of them uses state-space models, in particular a particle filter based on evolutionary strategies [1] and Bayesian learning [2]. Another strategy focuses directly on hardware solutions that employ voltage and current measurements [3]. A very important group of techniques for NAEC is the class of linear-in-the-parameters (LIP) nonlinear adaptive filters. These filters consist of a linear combination of nonlinear representations of the input signal, and they include Volterra filters [4], [5], Functional Links Adaptive filters (FLAF) [6], [7], Even Mirror Fourier filters [8], cascaded Kalman filters [9], Hammerstein group models [10], up to schemes that propose physical models of the elements involved in the NAEC scenario [11].

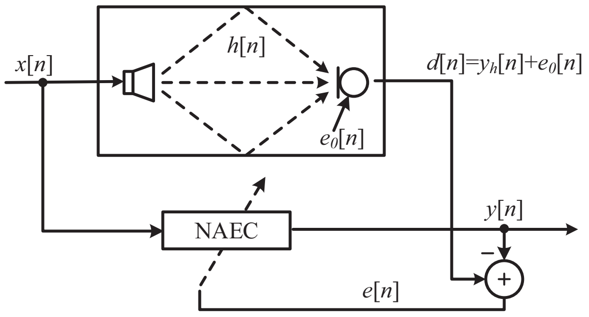

Fig. 1. Typical single-channel NAEC scheme.

In this paper, we propose a specific type of LIP nonlinear filters based on kernel methods. Kernel methods are a class of machine learning techniques that provide a framework for nonlinear signal processing, in which several nonlinear techniques can be formulated as convex optimization problems [12]. While traditionally they suffered from high computational requirements that made them unsuitable for real-time applications, several low-complexity techniques have since been developed that allow to design *kernel adaptive filters* (KAF) [13], [14], for instance [15]–[20]. Recently, some of these techniques have been applied to NAEC [21]–[23].

This work includes several contributions. In Section II we provide a brief overview of kernel adaptive filtering and we propose a novel, low-complexity online KAF algorithm. Then, in Section III, we define a NAEC framework based on a parallel structure similar to [6], in which any kernel adaptive filter can be incorporated to model the nonlinear distortion. In Section IV we test the proposed scheme with several KAF algorithms in real NAEC scenarios, followed by our conclusions and some brief future research lines in Section V.

II. Adaptive filtering with kernels

A. Kernel methods

A standard procedure to extend the scope of linear filters to nonlinear processing consists in mapping the input data $\mathbf{x} \in \mathcal{X}$ to a high-dimensional *feature space* by means of a nonlinear transformation $\Phi(\cdot)$ and then applying the linear filtering algorithm in that space [24]. By operating in a high-dimensional space, more degrees of freedom are available to formulate a solution for the filtering problem. Nevertheless, the additional dimensions also increase the computational load of the algorithm, which may become excessive.

Kernel methods provide an efficient solution to this problem by avoiding the explicit mapping of the data [12]. They exploit a property from the theory of reproducing kernel Hilbert spaces (RKHS) that allows to operate implicitly in the high-dimensional feature space by solely replacing inner products by Mercer kernels in the original problem formulation.

A Mercer kernel is any continuous, symmetric and positive definite function $\kappa : \mathcal{X} \times \mathcal{X} \to \mathcal{R}$, see [25]. The polynomial kernel of order p, for instance, is defined as

$$\kappa(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^\top \mathbf{x}' + c)^p, \tag{2}$$

where c is a trade-off parameter. According to Mercer's theorem, any Mercer kernel $\kappa(\mathbf{x}, \mathbf{x}')$ induces a mapping $\Phi(\cdot)$ from the input space to a high-dimensional feature space (which is an inner product space) such that the following relationship holds [26]

$$\kappa(\mathbf{x}, \mathbf{x}') = \Phi(\mathbf{x})^\top \Phi(\mathbf{x}'). \tag{3}$$

In case of the polynomial kernel with $p = 2$, the mapping can be calculated explicitly for L-dimensional data as

$$\Phi(\mathbf{x}) = [x_1 x_1, \ldots, x_1 x_L, \ldots, x_L x_L, \sqrt{2c}x_1, \ldots, \sqrt{2c}x_L, c].$$

The property (3), commonly known as the *kernel trick*, is the building block of kernel methods. It implies that by replacing the inner products by kernels in a linear algorithm, a new algorithm is obtained that is equivalent to performing the original algorithm in the feature space, without the need to perform explicit calculations in this high-dimensional space.

Interestingly, in [27] it was shown that the coefficient space of a Volterra series is an RKHS associated with a polynomial kernel. This implies that the modeling of nonlinear systems through Volterra filters, which require large amounts of data and computation, can be carried out implicitly at much less cost using polynomial kernels. A more commonly used kernel is the Gaussian kernel $\kappa(\mathbf{x}, \mathbf{x}') = \exp(-\|\mathbf{x} - \mathbf{x}'\|^2/(2\sigma^2))$. Nevertheless, this kernel function is much less efficient in applications such as NAEC where the AIR is of very high order.

Given a new datum $\mathbf{x}[n]$, the output of a linear filtering algorithm in feature space can be written as

$$f(\mathbf{x}[n]) = \widetilde{\mathbf{w}}^\top \Phi(\mathbf{x}[n]), \tag{4}$$

where $\widetilde{\mathbf{w}}$ contains the coefficients of the filter. In order to avoid explicitly calculating the involved high-dimensional vectors, one can rely on the Representer theorem [28], which guarantees that the solution in the feature space, $\widetilde{\mathbf{w}}$, can be expressed as a linear combination of all previously observed data $\mathbf{x}[i]$, transformed into feature space

$$\widetilde{\mathbf{w}} = \sum_i \alpha_i \Phi(\mathbf{x}[i]).$$

This expression allows to rewrite (4) as the *kernel expansion*

$$f(\mathbf{x}[n]) = \sum_i \alpha_i \kappa(\mathbf{x}[i], \mathbf{x}[n]) = \boldsymbol{\alpha}^\top \mathbf{k}, \tag{5}$$

in which $\boldsymbol{\alpha}$ and $\mathbf{k}$ are vectors that hold the elements α_i and $\kappa(\mathbf{x}[i], \mathbf{x}[n])$, respectively.

B. A Kernelized Normalized Least Mean Squares Filter

Consider the well-known normalized least mean squares filter (NLMS) [29], which performs the following update of its solution $\widehat{\mathbf{w}}$ each time it processes a new data pair $(\mathbf{x}[n], d[n])$:

$$e[n] = d[n] - \widehat{\mathbf{w}}[n-1]^\top \mathbf{x}[n], \tag{6a}$$

$$\widehat{\mathbf{w}}[n] = \widehat{\mathbf{w}}[n-1] + \frac{\eta}{\|\mathbf{x}[n]\|^2 + \epsilon} e[n]\mathbf{x}[n], \tag{6b}$$

where η is the learning rate and ϵ is a regularization factor. Note that in the NAEC filtering scenario the datum $\mathbf{x}[n]$ refers to the tap-delay input vector at the n-th time instant.

The update equations (6) of the NLMS algorithm can be ported directly to the kernel feature space, where they read

$$e[n] = d[n] - \widetilde{\mathbf{w}}[n-1]^\top \Phi(\mathbf{x}[n]), \tag{7a}$$

$$\widetilde{\mathbf{w}}[n] = \widetilde{\mathbf{w}}[n-1] + \frac{\eta}{\|\Phi(\mathbf{x}[n])\|^2 + \epsilon} e[n]\Phi(\mathbf{x}[n]). \tag{7b}$$

In [17] a set of update equations for $\boldsymbol{\alpha}[n]$ are obtained by casting (7) into a least-squares projection framework, yielding

$$e[n] = d[n] - \boldsymbol{\alpha}[n-1]^\top \mathbf{k}, \tag{8a}$$

$$\boldsymbol{\alpha}[n] = \begin{bmatrix} \boldsymbol{\alpha}[n-1] \\ 0 \end{bmatrix} + \frac{\eta}{\|\mathbf{k}_a\|^2 + \epsilon} e[n]\mathbf{k}_a \tag{8b}$$

in which $\mathbf{k}$ is the $n-1$-dimensional vector whose i-th element is $\kappa(\mathbf{x}[i], \mathbf{x}[n])$, and $\mathbf{k}_a = [\mathbf{k}^\top, \kappa(\mathbf{x}[n], \mathbf{x}[n])]^\top$. By performing the update equations (8), a nonlinear version of NLMS is obtained, which corresponds to linear filtering in the high-dimensional feature space induced by the Mercer kernel. Notice, though, that the computational complexity of KNLMS for the n-th iteration is $\mathcal{O}(n)$. Furthermore, in order to calculate $\mathbf{k}$, all previous data $\mathbf{x}[i]$, $i = 1, \ldots, n-1$ need to be stored. This set is referred to as the *dictionary* $\mathcal{D}$.

C. A Simple and Practical KNLMS Algorithm

The update (8b) implies that the vector $\boldsymbol{\alpha}[n]$ and the dictionary $\mathcal{D}$ grow unboundedly over time. This growth stems from the fact that the kernel expansion (5) relies explicitly on all previously observed data. Kernel adaptive filters therefore require a dictionary control mechanism that decides whether to insert or to remove certain data, with the aim of building a compact dictionary that represents the possible input data as closely as possible.

A straightforward dictionary control mechanism for real-time applications consists in pruning one datum from the dictionary and inserting one new datum into it, in each iteration. In order to decide which datum to prune, several algorithms have been proposed [14]. However, most of them have $\mathcal{O}(M^2)$ complexity per iteration, where M is the dictionary size, exceeding the cost of the parameter update of the adaptive filter. A very simple and low-cost pruning strategy consists in discarding the dictionary element with lowest associated weight α_i, as it has the lowest contribution in the functional representation. This criterion is also very appropriate for practical settings, as it does not add any free parameters to the algorithm.

Algorithm 1 Simple Kernel Normalized Least Mean Squares

```
parameters: $\kappa(\cdot,\cdot)$, $M$, $\eta$.
initialize: $\mathcal{D}_1 = \{\mathbf{x}[1]\}$, $\boldsymbol{\alpha}[1] = d[1]/\kappa(\mathbf{x}[1],\mathbf{x}[1])$.
for $n = 2, 3, \ldots$ do
    if $|\mathcal{D}| = M$ then
        Determine basis to remove: $j = \text{argmin}_i\, \alpha_i[n-1]$.
        Remove $j$-th entry from $\mathcal{D}[n-1]$ and $\boldsymbol{\alpha}[n-1]$.
    end if
    Calculate $\mathbf{k}$, containing the kernels between $\mathcal{D}$ and $\mathbf{x}[n]$.
    Calculate $\mathbf{k}_a = [\mathbf{k}^\top, \kappa(\mathbf{x}[n],\mathbf{x}[n])]^\top$.
    $e[n] = d[n] - \boldsymbol{\alpha}[n-1]^\top \mathbf{k}$    (8a)
    $\boldsymbol{\alpha}[n] = \left[\boldsymbol{\alpha}[n-1]^\top, 0\right]^\top + \eta/(\|\mathbf{k}_a\|^2 + \epsilon)\cdot e[n]\mathbf{k}_a$    (8b)
    $\mathcal{D}[n] = \mathcal{D}[n-1] \cup \{\mathbf{x}[n]\}$.
end for
```

We summarize the described simple KNLMS (SKNLMS) algorithm in Alg. 1. While it serves as an illustration of the design of a kernel adaptive filter, it is also a novel algorithm that will be used in the experiments of Section IV.

III. Proposed NAEC scheme

An acoustic echo path always involves linear elements, related to the acoustic propagation described by the AIR and to the linear distortion of the transducer, and, increasingly often, nonlinear distortions caused mainly by the power amplifier and/or the loudspeaker. Different classes of structures have been proposed for NAEC in the literature. Recently, a parallel architecture was expressly proposed for NAEC, highlighting its demonstrated versatility and robustness [6]. This structure allows the decoupled adaptation of the linear and nonlinear parts (which is why it is referred to as "split"), enabling an independent selection of filtering settings such as adaptation speed, input buffer length, etc.

Following this trend, the model we adopt is a parallel construction similar to that of [6], composed of a linear and a nonlinear adaptive filter. However, in this work, instead of using a functional link adaptive filter for the nonlinear branch, we propose to study the use of kernel adaptive filters to model the nonlinearities of the echo path, motivated by their state-of-the-art results in nonlinear adaptive filtering. The proposed scheme is depicted in Fig. 2. We denote it the "split kernel adaptive filter" (SKAF) due to its similarity with the split functional link adaptive filter (SFLAF) from [6].

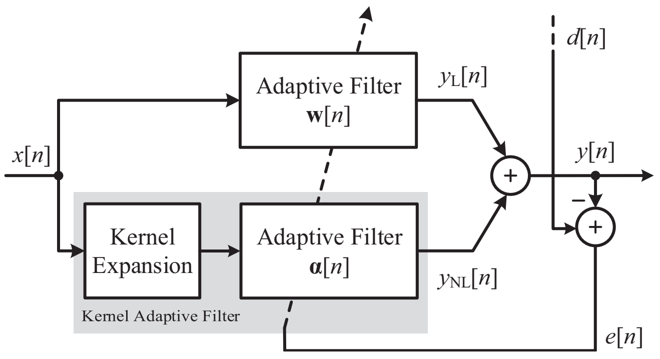

Fig. 2. The proposed split kernel adaptive filtering scheme.

Alg. 2 lists the pseudo-code for the SKAF scheme. The two training steps at the end are equivalent to an adaptation step using the combined error $e[n] = d[n] - y_L[n] - y_{NL}[n]$, which is in line with [6]. However, by explicitly listing the desired output as in Alg. 2, it becomes clear that any nonlinear filtering algorithm with a training and prediction step can be fit into this scheme. The proposed framework is therefore modular and allows to take advantage of open source libraries such as KAFBOX [30] which includes numerous KAF algorithms. A Matlab implementation of the SKAF and SKNLMS algorithms is available at http://gtas.unican.es/people/steven.

Algorithm 2 Split Kernel Adaptive Filtering

```
Initialize NLMS and KAF on data pair$(\mathbf{x}[1], d[1])$.
for $n = 2, 3, \ldots$ do
    Given input $\mathbf{x}[n]$, generate output $y_L[n]$ of NLMS.
    Given input $\mathbf{x}[n]$, generate output $y_{NL}[n]$ of KAF.
    Train NLMS on data pair $(\mathbf{x}[n], d[n] - y_{NL}[n])$.
    Train KAF on data pair $(\mathbf{x}[n], d[n] - y_L[n])$.
end for
```

There is, however, a restriction on the KAF algorithm that is plugged into this scheme. In particular, the algorithm should have tracking capability, which is not always the case (see [14]), and it should have an efficient mechanism for managing the dictionary size in order to allow real-time application. In the experiments we will study the results of two such kernel adaptive filters, one of the LMS class (as proposed in Section II) and one of the recursive least squares (RLS) class.

Finally, we remark that several authors have started studying the application of kernel adaptive filtering in NAEC recently. In particular, [21] modeled the acoustic echo path as a single kernel adaptive filter using a sum of a linear and a nonlinear kernel. In that approach, however, the linear filter is conditioned by the restrictions imposed by the kernel adaptive filter, such as the dictionary size. And in [23] a parallel construction of filters was proposed, similar to our scheme, though the nonlinear part was modelled by a more complex multi-Gaussian kernel adaptive filter.

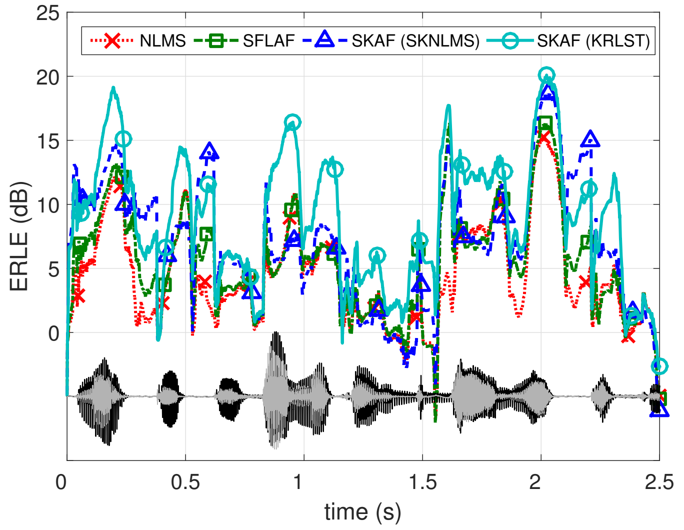

Fig. 3. Performance comparison in terms of ERLE between a linear filter, a SFLAF, a SKAF with SKNLMS and a SKAF with KRLS-T, in case of quiet speech input. The speech is shown in black (original) and grey (distorted).

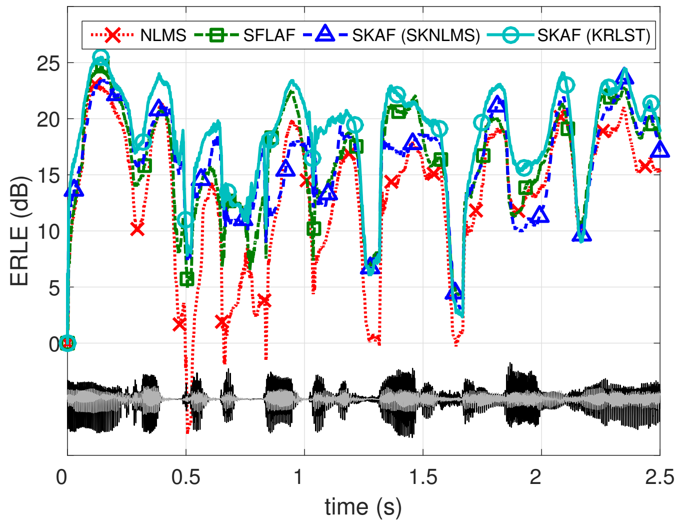

Fig. 4. Performance comparison in terms of ERLE between a linear filter, a SFLAF, a SKAF with SKNLMS and a SKAF with KRLS-T, in case of speech input with a loud volume level of the loudspeaker.

IV. Experimental evaluation

We evaluate the performance of the proposed method on real data from a classic scenario of acoustic echo cancellation, i.e. a hands-free desktop teleconference. Experiments take place in a typical office room with a relatively low level of background noise, thus providing sufficiently high signal-to-noise ratio (SNR). This condition has been chosen in order to evaluate the proposed canceller fairly, avoiding external interferences that could require further processing modules (e.g., double-talk detectors). For the same reason, we used a high-quality microphone (AKG C562 CM with omnidirectional pattern), which allows us to focus on the nonlinearities produced by the loudspeaker. On the other hand, at 40 cm from the microphone, we placed a low-cost commercial loudspeaker, capable of introducing significant distortions. The input signal is male speech recorded at 16 kHz sampling frequency. The duration of the experiments is 20 seconds. We consider two different volume levels, which entail different nonlinearity degrees. In the first experiment, we consider a typical volume level of a quiet speech conversation, without significant fluctuations. In this condition, loudspeaker distortions are mild and they cannot be perceived by the user. However, they do affect the echo cancellation, thus degrading the performance in the absence of a nonlinear modeling. In a second experiment, we evaluate the NAEC performance in more adverse conditions, caused by a louder volume level of the loudspeaker, which produces stronger distortions. In this case, besides provoking a performance decrease, such distortions also generate an annoying audible crackling effect.

We compare the performance of three different algorithms: SFLAF [6] (which has obtained better performance than Volterra filters), the proposed SKAF scheme and linear NLMS. The SKAF scheme is applied with two different kernel adaptive filters: once with the computationally less expensive SKNLMS algorithm, and once with the more sophisticated kernel recursive least-squares tracker (KRLS-T) algorithm from [18], [19]. While RLS-based algorithms have higher complexity than LMS-based algorithms, we equip the KRLS-T algorithm with a far smaller dictionary than SKNLMS so that both run times are similar (and lower than that of SFLAF).

In the first experiment, the parameters are set as follows: the linear filter in all algorithms has 150 taps; NLMS has learning rate $\eta = 0.5$ and regularization $\epsilon = 10^{-3}$; the nonlinear branch of SFLAF uses trigonometric expansion functions and has a buffer of 150 samples, expansion order 5, memory order 2, learning rate 0.1, and regularization 0.001; both configurations of SKAF use a buffer of 20 samples and the polynomial kernel of order $p = 3$ with $c = 1$, and whereas the SKNLMS implementation uses a dictionary size $M = 1000$ and learning rate 0.5, the KRLS-T implementation uses $M = 100$ and forgetting factor $\lambda = 0.999$. In the second experiment the following parameters are changed: the nonlinear branch of SFLAF has a buffer of 100 samples, learning rate 0.2, and regularization 0.1; both configurations of SKAF use the polynomial kernel of order $p = 2$, and KRLS-T has forgetting factor $\lambda = 0.99$.

The performance of the algorithms is evaluated in terms of the *Echo Return Loss Enhancement* (ERLE), defined as

$$\text{ERLE}[n] = 10 \log_{10} \left(\frac{E\{d^2[n]\}}{E\{e^2[n]\}} \right),$$

where the operator $E\{\cdot\}$ denotes the mathematical expectation, which is approximated by smoothing its instantaneous values.

The ERLE performances for both experiments are displayed in Figs. 3 and 4, respectively. For better readability of the figures, we show a window of 2.5 out of 20 seconds of the ERLE behavior. The results for SKAF (SKNLMS) are similar to those obtained for SFLAF. This is an interesting result, given the simplicity and lower run time of SKAF (SKNLMS). By replacing SKNLMS with KRLS-T in SKAF a very significant performance boost is obtained: This algorithm consistently outperforms all other algorithms, often by a margin of more than 5 dB in the first experiment. The gain in the second experiment is smaller, as expected in the presence of stronger non-linearities, though it is still noticeable.

V. Conclusions

We have studied a split kernel adaptive filtering architecture in which a kernel adaptive filtering algorithm is employed in parallel with a linear adaptive filter, for application in nonlinear acoustic echo cancellation. We have designed a low-complexity LMS-type kernel filter that fits into this framework. Two experiments on real data show that the proposed framework with the novel SKNLMS filter yields good results. By using the more sophisticated KRLS-T kernel adaptive filter in the proposed framework we achieve results that surpass the state-of-the-art SFLAF algorithm.

Several future research directions lie ahead. First, new and more robust filtering schemes can be studied, such as collaborative schemes with convex combinations of filters. Second, the proposed ideas can be applied in other applications such as equalization and non-real-time applications.

Acknowledgments

The work of Steven Van Vaerenbergh is supported by the Spanish Ministry of Economy and Competitiveness (under project TEC2014-57402-JIN).

The work of Luis A. Azpicueta-Ruiz is partially supported by Comunidad de Madrid under grant ‘CASI-CAM-CM’ (id. S2013/ICE-2845), by the Spanish Ministry of Economy and Competitiveness (under grant DAMA (TIN2015-70308-REDT) and grant TEC2014-52289-R), and by the European Union.

References

[1] C. Huemmer, C. Hofmann, R. Maas, A. Schwarz, and W. Kellermann, “The elitist particle filter based on evolutionary strategies as novel approach for nonlinear acoustic echo cancellation,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Florence, Italy, May 4-9 2014, pp. 1329–1333.

[2] S. Malik and G. Enzner, “A variational Bayesian learning approach for nonlinear acoustic echo control,” *IEEE Transactions on Signal Processing*, vol. 61, no. 23, pp. 5853–5867, Dec. 2013.

[3] P. Shah, I. Lewis, S. Grant, and S. Angrignon, “Nonlinear acoustic echo cancellation using voltage and current feedback,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 10, pp. 1589–1599, Oct. 2015.

[4] V. Mathews and G. Sicuranza, *Polynomial Signal Processing*. Wiley, 2000.

[5] L. A. Azpicueta-Ruiz, M. Zeller, A. R. Figueiras-Vidal, J. Arenas-García, and W. Kellermann, “Adaptive combination of Volterra kernels and its application to nonlinear acoustic echo cancellation,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 1, pp. 97–110, Jan. 2011.

[6] D. Comminiello, M. Scarpiniti, L. A. Azpicueta-Ruiz, J. Arenas-García, and A. Uncini, “Functional link adaptive filters for nonlinear acoustic echo cancellation,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 7, pp. 1502–1512, Jul. 2013.

[7] ——, “Nonlinear acoustic echo cancellation based on sparse functional link representations,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 7, no. 22, pp. 1172–1183, Jul. 2014.

[8] A. Carini and G. L. Sicuranza, “Fourier nonlinear filters,” *Signal Processing*, vol. 94, pp. 183–194, Jan. 2014.

[9] M. Z. Ikram, “Non-linear acoustic echo cancellation using cascaded Kalman filtering,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Florence, Italy, May 4-9 2014, pp. 1334–1338.

[10] C. Hofmann, C. Huemmer, and W. Kellermann, “Significance-aware Hammerstein group models for nonlinear acoustic echo cancellation,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Florence, Italy, May 4-9 2014, pp. 5975–5979.

[11] S. S. Payal, V. J. Mathews, A. Iyer, R. Lambert, and J. Hutchings, “Equalization of excursion and current-dependent nonlinearities in loudspeakers,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Florence, Italy, May 4-9 2014, pp. 4–9.

[12] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. The MIT Press, Dec. 2002.

[13] W. Liu, J. Principe, and S. Haykin, *Kernel Adaptive Filtering: A Comprehensive Introduction*. Wiley, 2011, vol. 57.

[14] S. Van Vaerenbergh and I. Santamaría, *Online Regression with Kernels*. New York: Chapman and Hall/CRC, 2014, no. Machine Learning & Pattern Recognition Series, ch. 21, pp. 477–501.

[15] J. Kivinen, A. J. Smola, and R. C. Williamson, “Online learning with kernels,” *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2165–2176, Aug. 2004.

[16] Y. Engel, S. Mannor, and R. Meir, “The kernel recursive least squares algorithm,” *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2275–2285, Aug. 2004.

[17] C. Richard, J. C. M. Bermudez, and P. Honeine, “Online prediction of time series data with kernels,” *IEEE Transactions on Signal Processing*, vol. 57, no. 3, pp. 1058–1067, Mar. 2009.

[18] M. Lázaro-Gredilla, S. Van Vaerenbergh, and I. Santamaría, “A Bayesian approach to tracking with kernel recursive least-squares,” in *2011 IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, Sep. 2011, pp. 1–6.

[19] S. Van Vaerenbergh, M. Lázaro-Gredilla, and I. Santamaría, “Kernel recursive least-squares tracker for time-varying regression,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 8, pp. 1313–1326, Aug. 2012.

[20] S. Scardapane, D. Comminiello, M. Scarpiniti, and A. Uncini, “Online sequential extreme learning machine with kernels,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 9, pp. 2214–2220, Sep. 2015.

[21] J. M. Gil-Cacho, M. Signoretto, T. Van Waterschoot, M. Moonen, and S. H. Jensen, “Nonlinear acoustic echo cancellation based on a sliding-window leaky kernel affine projection algorithm,” *IEEE Trans. on Audio, Speech, and Lang. Proc.*, vol. 21, no. 9, pp. 1867–1878, Sep. 2013.

[22] S. Van Vaerenbergh and L. A. Azpicueta-Ruiz, “Kernel-based identification of Hammerstein systems for nonlinear acoustic echo-cancellation,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Florence, Italy, May 4-9 2014, pp. 3767–3771.

[23] K. Nishikawa and F. Albu, “Implementation method of kernel adaptive filter as an add-on for a linear adaptive filter,” in *23rd European Signal Processing Conf. (EUSIPCO)*, Nice, France, Sep. 2015, pp. 2736–2740.

[24] D. S. Broomhead and D. Lowe, “Multivariable functional interpolation and adaptive networks,” *Complex Systems*, vol. 2, no. 3, pp. 321–355, 1988.

[25] N. Aronszajn, “Theory of reproducing kernels,” *Transactions of the American Mathematical Society*, vol. 68, pp. 337–404, 1950.

[26] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer-Verlag New York, Inc., 1995.

[27] M. O. Franz and B. Schölkopf, “A unifying view of wiener and volterra theory and polynomial kernel regression,” *Neural computation*, vol. 18, no. 12, pp. 3097–3118, 2006.

[28] B. Schölkopf, R. Herbrich, and A. J. Smola, “A generalized representer theorem,” in *Computational learning theory*. Springer, 2001, pp. 416–426.

[29] A. H. Sayed, *Fundamentals of Adaptive Filtering*. Wiley-IEEE Press, 2003.

[30] S. Van Vaerenbergh and I. Santamaría, “A comparative study of kernel adaptive filtering algorithms,” in *2013 IEEE Digital Signal Proc. (DSP) Workshop and IEEE Signal Proc. Education (SPE)*, Napa, CA, USA, Jul. 2013, software available at http://sourceforge.net/projects/kafbox/.