2

Multichannel Speech Enhancement

Lino García and Soledad Torres-Guijarro
Universidad Europea de Madrid, Universidad de Vigo
Spain

1. Introduction

1.1 Adaptive Filtering Review

There are a number of possible degradations that can be found in a speech recording and that can affect its quality. On one hand, the signal arriving the microphone usually incorporates multiple sources: the desired signal plus other unwanted signals generally termed as noise. On the other hand, there are different sources of distortion that can reduce the clarity of the desired signal: amplitude distortion caused by the electronics; frequency distortion caused by either the electronics or the acoustic environment; and time-domain distortion due to reflection and reverberation in the acoustic environment.

Adaptive filters have traditionally found a field of application in noise and reverberation reduction, thanks to their ability to cope with changes in the signals or the sound propagation conditions in the room where the recording takes place. This chapter is an advanced tutorial about multichannel adaptive filtering techniques suitable for speech enhancement in multiple input multiple output (MIMO) very long impulse responses. Single channel adaptive filtering can be seen as a particular case of the more complex and general multichannel adaptive filtering. The different adaptive filtering techniques are presented in a common foundation. Figure 1 shows an example of the most general MIMO acoustical scenario.

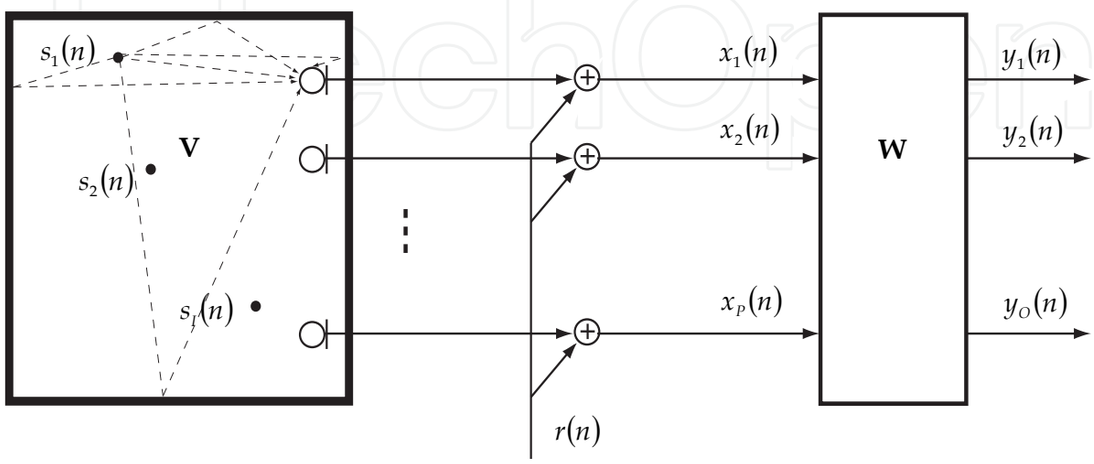

Fig. 1. Audio application scenario.

The box, on the left, represents a reverberant room. $\mathbf{V}$ is a $P \times LI$ matrix that contains the acoustic impulse responses (AIR) between the I sources and P microphones (channels); L is a filters length. Sources can be interesting or desired signals (to enhance) or noise and interference (to attenuate). The discontinuous lines represent only the direct path and some first reflections between the $s_1(n)$ source and the microphone with output signal $x_1(n)$. Each $\mathbf{v}_{pi}(n)$ vector represents the AIR between $i = 1 \ldots I$ and $p = 1 \ldots P$ positions and is constantly changing depending on the position of both: source or microphone, angle between them, radiation pattern, etc.

$$\mathbf{V} = \begin{bmatrix} \mathbf{v}_{11} & \mathbf{v}_{12} & \cdots & \mathbf{v}_{1I} \\ \mathbf{v}_{21} & \mathbf{v}_{22} & \cdots & \mathbf{v}_{2I} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{v}_{P1} & \mathbf{v}_{P2} & \cdots & \mathbf{v}_{PI} \end{bmatrix},$$

$$\mathbf{v}_{pi} = [\, v_{pi1} \; v_{pi2} \; \cdots \; v_{piL} \,]. \tag{1}$$

$r(n)$ is an additive noise or interference signal. $x_p(n)$, $p = 1 \ldots P$ is a corrupted or poor quality signal that wants to be improved. The filtering goal is to obtain a $\mathbf{W}$ matrix so that $y_o(n) \approx \hat{s}_i(n)$ corresponds to the identified signal. The signals in the Fig. 1 are related by

$$\mathbf{x}(n) = \mathbf{V}\mathbf{s}(n) + r(n), \tag{2}$$

$$\mathbf{y}(\mathrm{n}) = \mathbf{W}\mathbf{x}(\mathrm{n}). \tag{3}$$

$s(n)$ is a $LI \times 1$ vector that collects the source signals,

$$\mathbf{s}(n) = \left[\mathbf{s}_1^T(n) \;\; \mathbf{s}_2^T(n) \;\; \cdots \;\; \mathbf{s}_I^T(n)\right]^T, \tag{4}$$
$$\mathbf{s}_i(n) = \left[s_i(n) \;\; s_i(n-1) \;\; \cdots \;\; s_i(n-L+1)\right]^T.$$

$x(n)$ is a $P \times 1$ vector that corresponds to the convolutive system output excited by $s(n)$ and the adaptive filter input of order $O \times LP$. $x_p(n)$ is an input corresponding to the channel p containing the last L samples of the input signal x,

$$\mathbf{x}(n) = \left[\mathbf{x}_1^T(n) \;\; \mathbf{x}_2^T(n) \;\; \cdots \;\; \mathbf{x}_P^T(n)\right]^T, \tag{5}$$
$$\mathbf{x}_p(n) = [\, \mathbf{x}_p(n) \;\; \mathbf{x}_p(n\text{-}1) \;\; \cdots \mathbf{x}_p(n\text{-}L\text{+}1)]^T.$$

$\mathbf{W}$ is an $O \times LP$ adaptive matrix that contains an AIRs between the P inputs and O outputs

$$\mathbf{W} = \begin{bmatrix} \mathbf{w}_{11} & \mathbf{w}_{12} & \cdots & \mathbf{w}_{1P} \\ \mathbf{w}_{21} & \mathbf{w}_{22} & \cdots & \mathbf{w}_{2P} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{w}_{O1} & \mathbf{w}_{O2} & \cdots & \mathbf{w}_{OP} \end{bmatrix},$$

$$\mathbf{w}_{op} = [w_{op1} \;\; w_{op2} \;\; \cdots \;\; w_{opL}]. \tag{6}$$

For a particular output $o = 1 \ldots O$, normally matrix $\mathbf{W}$ is rearranged as column vector

$$\mathbf{w} = [\mathbf{w}_1 \quad \mathbf{w}_2 \quad \cdots \quad \mathbf{w}_P]^T. \tag{7}$$

Finally, $y(n)$ is an $O \times 1$ target vector, $\mathbf{y}(n) = [y_1(n) \quad y_2(n) \quad \cdots \quad y_O(n)]^T$.

The used notation is the following: a or α is a scalar, $\mathbf{a}$ is a vector and $\mathbf{A}$ is a matrix in time-domain $\mathbf{a}$ is a vector and $\mathbf{A}$ is a matrix in frequency-domain. Equations (2) and (3) are in matricial form and correspond to convolutions in a time-domain. The index n is the discrete time instant linked to the time (in seconds) by means of a sample frequency F_s according to $t = nT_s$, $T_s = 1/F_s$. T_s is the sample period. Superscript ^{T} denotes the transpose of a vector or a matrix, * denotes the conjugate of a vector or a matrix and superscript ^{H} denotes Hermitian (the conjugated transpose) of a vector or a matrix. Note that, if adaptive filters are $L \times 1$ vectors, L samples have to be accumulated per channel (i.e. delay line) to make the convolutions (2) and (3).

The major assumption in developing linear time-invariant (LTI) systems is that the unwanted noise can be modeled by an additive Gaussian process. However, in some physical and natural systems, noise can not be modelled simply as an additive Gaussian process, and the signal processing solution may also not be readily expressed in terms of mean squared errors (MSE)[1].

From a signal processing point of view, the particular problem of noise reduction generally involves two major steps: *modeling* and *filtering*. The modelling step generally involves determining some approximations of either the noise spectrum or the input signal spectrum. Then, some filtering is applied to emphasize the signal spectrum or attenuate/reject the noise spectrum (Chau, 2001). Adaptive filtering techniques are used largely in audio applications where the ambient noise environment has a complicated spectrum, the statistics are rapidly varying and the filter coefficients must automatically change in order to maintain a good intelligibility of the speech signal. Thus, filtering techniques must be

[1] MSE is the best estimator for random (or stochastic) signals with Gaussian distribution (normal process). The Gaussian process is perhaps the most widely applied of all stochastic models: most error processes, in an estimation situation, can be approximated by a Gaussian process; many non-Gaussian random processes can be approximated with a weighted combination of a number of Gaussian densities of appropriated means and variances; optimal estimation methods based on Gaussian models often result in linear and mathematically tractable solutions and the sum of many independent random process has a Gaussian distribution (central limit theorem) (Vaseghi, 1996).

powerful, precise and adaptive. Most *non-referenced* noise reduction systems have only one single input signal. The task of estimating the noise and/or signal spectra must then make use of the information available only from the single input signal and the noise reduction filter will also have only the input signal for filtering. *Referenced* adaptive noise reduction/cancellation systems work well only in constrained environments where a good reference input is available, and the crosstalk problem is negligible or properly addressed.

2. Multichannel Adaptive Filters

In a multichannel system ($P > 1$) it is possible to remove noise and interference signals by applying sophisticated adaptive filtering techniques that use spatial or redundant information. However there are a number of noise and distortion sources that can not be minimized by increasing the number of microphones. Examples of this are the surveillance, recording, and playback equipment. There are several classes of adaptive filtering (Honig & Messerschmitt, 1984) that can be useful for speech enhancement, as will be shown in Sect. 4. The differences among them are based on the external connections to the filter. In the *estimator* application [see Fig. 2(a)], the internal parameters of the adaptive filter are used as *estimate*. In the *predictor* application [see Fig. 2(b)], the filter is used to filter an input signal, $x(n)$, in order to minimize the output signal, $e(n) = x(n) - y(n)$, within the constrains of the filter structure. A *predictor* structure is a linear weighting of some finite number of past input samples used to *estimate* or *predict* the current input sample. In the *joint-process estimator* application [see Fig. 2(c)] there are two inputs, $x(n)$ and $d(n)$. The objective is usually to minimize the size of the output signal, $e(n) = d(n) - y(n)$, in which case the objective of the adaptive filter itself is to generate an estimate of $d(n)$, based on a filtered version of $x(n)$, $y(n)$ (Honig & Messerschmitt, 1984).

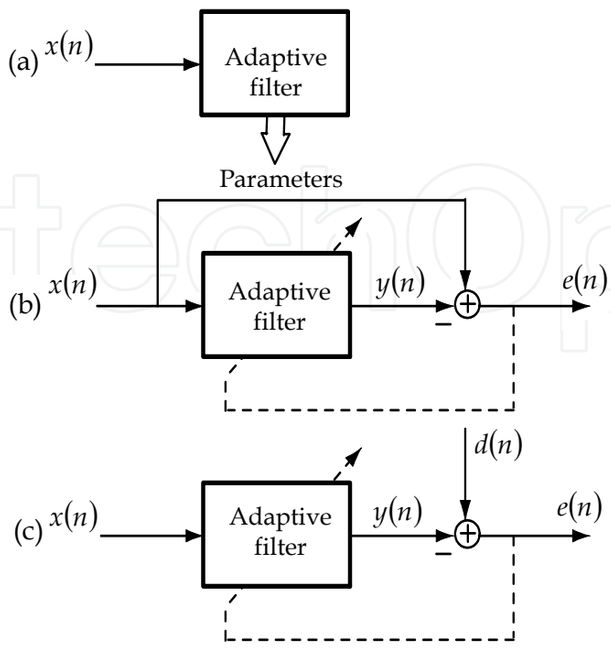

Fig. 2. Classes of adaptive filtering.

2.1 Filter Structures

Adaptive filters, as any type of filter, can be implemented using different structures. There are three types of *linear* filters with finite memory: the *transversal filter*, *lattice predictor* and *systolic array* (Haykin, 2002).

2.1.1 Transversal

The *transversal filter*, *tapped-delay line filter* or *finite-duration impulse response filter* (FIR) is the most suitable and the most commonly employed structure for an adaptive filter. The utility of this structure derives from its simplicity and generality.

The multichannel transversal filter output used to build a joint-process estimator as illustrated in Fig. 2(c) is given by

$$y(n) = \sum_{p=1}^{P}\sum_{l=1}^{L} w_{pl} x_p(n-l+1) = \sum_{p=1}^{P} \left\langle \mathbf{w}_p, \mathbf{x}_p(n) \right\rangle = \left\langle \mathbf{w}, \mathbf{x}(n) \right\rangle . \tag{8}$$

Where $\mathbf{x}(n)$ is defined in (5) and $\mathbf{w}$ in (7). Equation (8) is called *finite convolution sum*.

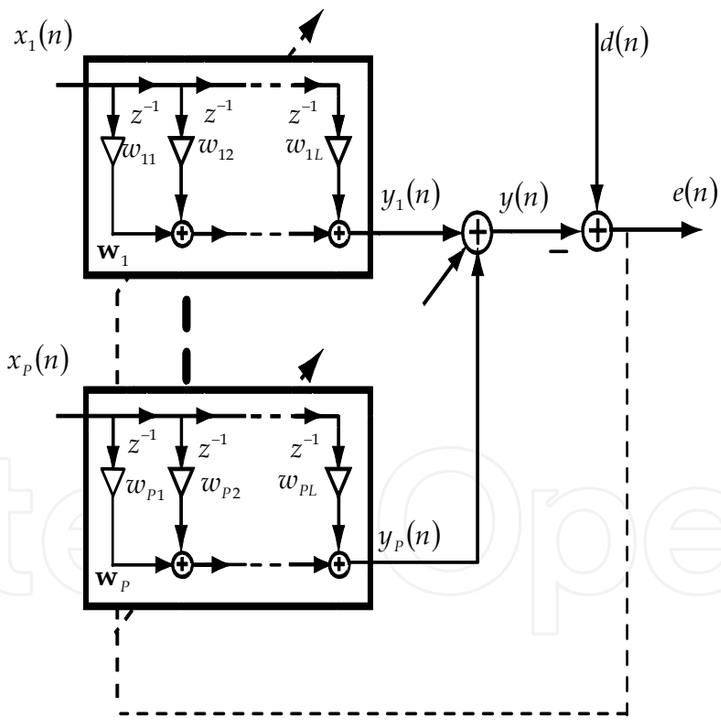

Fig. 3. Multichannel transversal adaptive filtering.

2.1.2 Lattice

The *lattice* filter is an alternative to the *transversal* filter structure for the realization of a *predictor* (Friedlander, 1982).

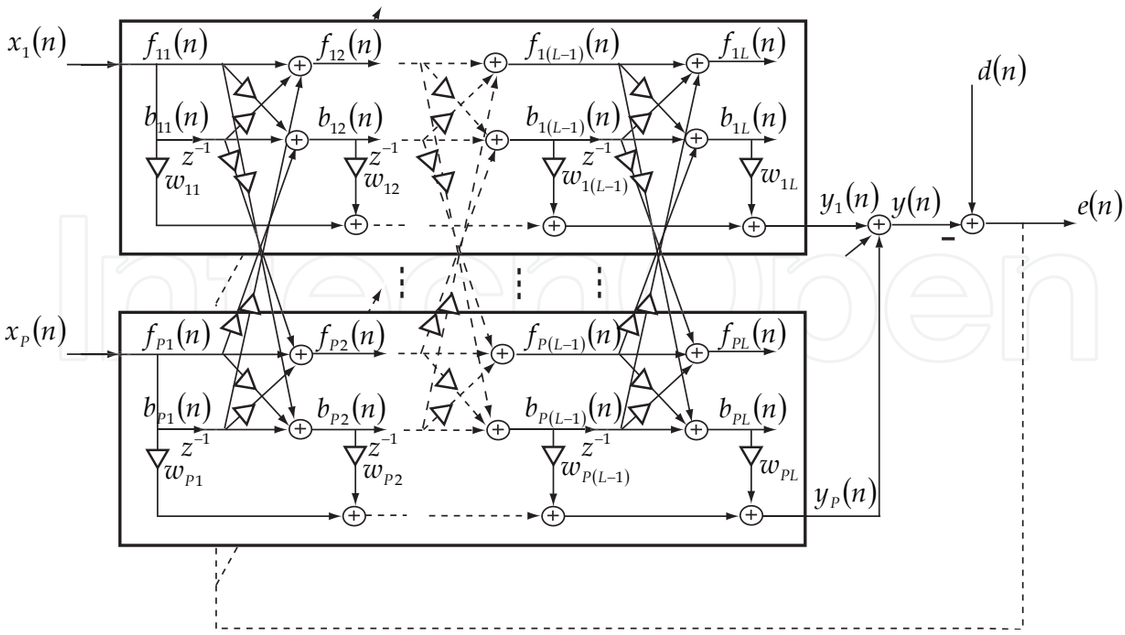

Fig. 4. Multichannel adaptive filtering with lattice-ladder joint-process estimator.

The multichannel version of lattice-ladder structure (Glentis et al., 1999) must consider the interchannel relationship of the reflection coefficients in each stage l.

$$\mathbf{f}_l(n) = \mathbf{f}_{l-1}(n) + \mathbf{K}_l^* \mathbf{b}_{l-1}(n-1), \mathbf{f}_1(n) = \mathbf{x}(n), \tag{9}$$

$$\mathbf{b}_l(n) = \mathbf{b}_{l-1}(n-1) + \mathbf{K}_l \mathbf{f}_{l-1}(n), \mathbf{b}_1(n) = \mathbf{x}(n). \tag{10}$$

Where $\mathbf{f}_l(n) = \begin{bmatrix} f_{1l}(n) & f_{2l}(n) & \cdots & f_{Pl}(n) \end{bmatrix}^T$, $\mathbf{b}_l(n) = \begin{bmatrix} b_{1l}(n) & b_{2l}(n) & \cdots & b_{Pl}(n) \end{bmatrix}^T$,

$\mathbf{x}(n) = \begin{bmatrix} x_1(n) & x_2(n) & \cdots & x_P(n) \end{bmatrix}^T$, and

$$\mathbf{K}_l = \begin{bmatrix} k_{11l} & k_{12l} & \cdots & k_{1Pl} \\ k_{21l} & k_{22l} & \cdots & k_{2Pl} \\ \vdots & \vdots & \ddots & \vdots \\ k_{P1l} & k_{P2l} & \cdots & k_{PPl} \end{bmatrix}^T.$$

The *joint-process estimation* of the *lattice-ladder* structure is especially useful for the adaptive filtering because its predictor *diagonalizes* completely the autocorrelation matrix. The transfer function of a lattice filter structure is more complex than a transversal filter because the reflexion coefficients are involved,

$$\mathbf{b}(n) = \mathbf{A}\mathbf{b}(n-1) + \mathbf{K}\mathbf{f}_1(n)\,, \tag{11}$$

$$y(n) = \mathbf{w}\mathbf{A}\mathbf{b}(n-1) + \mathbf{w}\mathbf{K}\mathbf{f}_1(n)\,. \tag{12}$$

Where $\mathbf{w} = \left[\mathbf{w}_1^T \quad \mathbf{w}_2^T \quad \cdots \quad \mathbf{w}_L^T\right]^T$ is a $LP \times 1$ vector of the joint-process estimator coefficients, $\mathbf{w}_l = \left[w_{1l} \quad w_{2l} \quad \cdots \quad w_{Pl}\right]^T$. $\mathbf{b}(n) = \left[\mathbf{b}_1^T(n) \quad \mathbf{b}_2^T(n) \quad \cdots \quad \mathbf{b}_L^T(n)\right]^T$ is a $LP \times 1$ backward predictor coefficients vector. $\mathbf{A}$ is a $LP \times LP$ matrix obtained with a recursive development of (9) and (10),

$$\mathbf{A} = \begin{bmatrix} \mathbf{0}_{P\times P} & \mathbf{0}_{P\times P} & \cdots & \mathbf{0}_{P\times P} & \mathbf{0}_{P\times P} & \mathbf{0}_{P\times P} \\ \mathbf{I}_{P\times P} & \mathbf{0}_{P\times P} & \cdots & \mathbf{0}_{P\times P} & \mathbf{0}_{P\times P} & \mathbf{0}_{P\times P} \\ \mathbf{K}_1^*\mathbf{K}_2 & \mathbf{I}_{P\times P} & \cdots & \mathbf{0}_{P\times P} & \mathbf{0}_{P\times P} & \mathbf{0}_{P\times P} \\ \mathbf{K}_1^*\mathbf{K}_3 & \mathbf{K}_2^*\mathbf{K}_3 & \cdots & \mathbf{0}_{P\times P} & \mathbf{0}_{P\times P} & \mathbf{0}_{P\times P} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \mathbf{K}_1^*\mathbf{K}_{L-3} & \mathbf{K}_2^*\mathbf{K}_{L-3} & \cdots & \mathbf{0}_{P\times P} & \mathbf{0}_{P\times P} & \mathbf{0}_{P\times P} \\ \mathbf{K}_1^*\mathbf{K}_{L-2} & \mathbf{K}_2^*\mathbf{K}_{L-2} & \cdots & \mathbf{I}_{P\times P} & \mathbf{0}_{P\times P} & \mathbf{0}_{P\times P} \\ \mathbf{K}_1^*\mathbf{K}_{L-1} & \mathbf{K}_2^*\mathbf{K}_{L-1} & \cdots & \mathbf{K}_{L-2}^*\mathbf{K}_{L-1} & \mathbf{I}_{P\times P} & \mathbf{0}_{P\times P} \end{bmatrix}. \tag{13}$$

$\mathbf{I}_{P\times P}$ is a matrix with only ones in the main diagonal and $\mathbf{0}_{P\times P}$ is a $P \times P$ zero matrix. $\mathbf{K} = \left[\mathbf{I}_{P\times P} \quad \mathbf{K}_1 \quad \mathbf{K}_2 \quad \cdots \quad \mathbf{K}_{L-1}\right]^T$ is a $LP \times P$ reflection coefficients matrix.

2.2 Adaptation Algorithms

Once a filter structure has been selected, an *adaptation algorithm* must also be chosen. From control engineering point of view, the speech enhancement is a system identification problem that can be solved by choosing an optimum criteria or cost function $J(\mathbf{w})$ in a *block* or *recursive* approach. Several alternatives are available, and they generally exchange increased complexity for improved performance (speed of adaptation and accuracy of the transfer function after adaption or misalignment defined by $\varepsilon = \|\mathbf{v} - \mathbf{w}\|^2 / \|\mathbf{v}\|^2$).

2.2.1 Cost Functions

Cost functions are related to the statistics of the involved signals and depend on some error signal

$$J(\mathbf{w}) = f\{e(n)\}\,. \tag{14}$$

The error signal $e(n)$ depends on the specific structure and the adaptive filtering strategy but it is usually some kind of similarity measure between the target signal $s_i(n)$ and the

estimated signal $y_o(n) \approx \hat{s}_i(n)$, for $I = O$. The most habitual cost functions are listed in Table1.

$J(\mathbf{w})$	Comments
$\|e(n)\|^2$	Mean squared error (MSE). Statistic mean operator
$\frac{1}{N}\sum_{i}^{N-1} e^2(n)$	MSE estimator. MSE is normally unknown
$e^2(n)$	Instantaneous squared error
$\|e(n)\|$	Absolute error. Instantaneous module error
$\sum_{i}^{n} \lambda^{n-m} e^2(m)$	Least squares (Weighted sum of the squared error)
$E\left\{\|\mathbf{f}_i(n)\|^2 + \|\mathbf{b}_i(n)\|^2\right\}$	Mean squared predictor errors (for a lattice structure)

Table 1. Cost functions for adaptive filtering.

2.2.2 Stochastic Estimation

Non-recursive or *block* methods apply batch processing to a transversal filter structure. The input signal is divided into time blocks, and each block is processed independently or with some overlap. This algorithms have *finite memory*.

The use of memory (vectors or matrice blocks) improves the benefits of the adaptive algorithm because they emphasize the variations in the crosscorrelation between the channels. However, this requires a careful structuring of the data, and they also increase the computational exigencies: memory and processing. For channel p, the input signal vector defined in (5) happens to be a matrix of the form

$$\mathbf{X}_p(n) = \left[\mathbf{x}_p^T(n-N+1) \quad \mathbf{x}_p^T\left(n-(N-1)+1\right) \quad \cdots \quad \mathbf{x}_p^T(n)\right]^T, \tag{15}$$

$$\mathbf{X}_p(n) = \begin{bmatrix} x_p(n-N+1) & x_p\left(n-(N-1)+1\right) & \cdots & x_p(n) \\ x_p(n-N) & x_p\left(n-(N-1)\right) & \cdots & x_p(n-1) \\ \vdots & \vdots & \ddots & \vdots \\ x_p(n-N-L+2) & x_p\left(n-(N-1)-L+2\right) & \cdots & x_p(n-L+1) \end{bmatrix},$$

$$\mathbf{d}(n) = \left[d(n-N+1) \quad d\left(n-(N-1)+1\right) \quad \cdots \quad d(n)\right]^T, \tag{16}$$

where N represents the *memory size*. The input signal matrix to the multichannel adaptive filtering has the form

$$\mathbf{X}(n) = \left[\mathbf{X}_1^T(n) \quad \mathbf{X}_2^T(n) \quad \cdots \quad \mathbf{X}_P^T(n)\right]^T. \tag{17}$$

In the most general case (with order memory N), the input signal $\mathbf{X}(n)$ is a matrix of size $LP \times N$. For $N = 1$ (memoryless) and $P = 1$ (single channel) (17) is reduced to (5).
There are adaptive algorithms that use memory $N > 1$ to modify the coefficients of the filter, not only in the direction of the input signal $x(n)$, but within the hyperplane spanned by the $x(n)$ and its $N-1$ immediate predecessors $\left[x(n) \quad x(n-1) \quad \cdots \quad x(n-N+1)\right]$ per channel. The block adaptation algorithm updates its coefficients once every N samples as

$$\mathbf{w}(m+1) = \mathbf{w}(m) + \Delta\mathbf{w}(m), \tag{18}$$

$$\Delta\mathbf{w}(m) = \arg\min J(\mathbf{w}).$$

The matrix defined by (15) stores $K = L + N - 1$ samples per channel. The time index m makes reference to a single update of the weights from time n to $n+N$, based on the K accumulated samples.
The *stochastic recursive* methods, unlike the different optimization *deterministic iterative* algorithms, allow the system to approach the solution with the partial information of the signals using the general rule

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \Delta\mathbf{w}(n), \tag{19}$$

$$\Delta\mathbf{w}(n) = \arg\min J(\mathbf{w}).$$

The new estimator $\mathbf{w}(n+1)$ is updated from the previous estimation $\mathbf{w}(n)$ plus the *adapting-step* or *gradient* obtained from the cost function minimization $J(\mathbf{w})$. These algorithms have an *infinite memory*. The trade-off between convergence speed and the accuracy is intimately tied to the length of memory of the algorithm. The error of the joint-process estimator using a transversal filter with memory can be rewritten like a vector as

$$\mathbf{e}(n) = \mathbf{d}(n) - \mathbf{y}(n) = \mathbf{d}(n) - \mathbf{X}^T(n)\mathbf{w}(n). \tag{20}$$

The unknown system solution, applying the MSE as the cost function, leads to the *normal* or *Wiener-Hopf equation*. The Wiener filter coefficients are obtained by setting the gradient of the square error function to zero, this yields

$$\mathbf{w} = \left[\mathbf{X}\mathbf{X}^H\right]^{-1}\mathbf{X}\mathbf{d}^* = \mathbf{R}^{-1}\mathbf{r}. \tag{21}$$

$\mathbf{R}$ is a correlation matrix and $\mathbf{r}$ is a cross-correlation vector defined by

$$\mathbf{R} = \mathbf{X}\mathbf{X}^H = \begin{bmatrix} \mathbf{X}_1\mathbf{X}_1 & \mathbf{X}_1\mathbf{X}_2 & \cdots & \mathbf{X}_1\mathbf{X}_P \\ \mathbf{X}_2\mathbf{X}_1 & \mathbf{X}_2\mathbf{X}_2 & \cdots & \mathbf{X}_2\mathbf{X}_P \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{X}_P\mathbf{X}_1 & \mathbf{X}_P\mathbf{X}_2 & \cdots & \mathbf{X}_P\mathbf{X}_P \end{bmatrix}, \tag{22}$$

$$\mathbf{r} = \mathbf{X}\mathbf{d}^* = \begin{bmatrix} \mathbf{X}_1\mathbf{d}^* & \mathbf{X}_2\mathbf{d}^* & \cdots & \mathbf{X}_P\mathbf{d}^* \end{bmatrix}^T. \tag{23}$$

For each $i = 1 \ldots I$ input source, $P(P-1)/2$ relations are obtained: $\mathbf{x}_p^H \mathbf{w}_q = \mathbf{x}_q^H \mathbf{w}_p$ for $p, q = 1 \ldots P$, with $p \neq q$. Given vector $\mathbf{u} = \begin{bmatrix} \sum_{p=2}^{P} \mathbf{w}_p^T & -\mathbf{w}_1^T & \cdots & -\mathbf{w}_1^T \end{bmatrix}^T$, due to the nearness with which microphones are placed in scenario of Fig. 1, it is possible to verify that $\mathbf{R}\mathbf{u} = \mathbf{0}_{PL \times 1}$, thus $\mathbf{R}$ is not invertible and no unique problem solution exists. The adaptive algorithm leads to one of many possible solutions which can be very different from the target $\mathbf{v}$. This is known as a *non-unicity problem*.

For a prediction application, the cross-correlation vector $\mathbf{r}$ must be slightly modified as $\mathbf{r} = \mathbf{X}\mathbf{x}(n-1)$, $\mathbf{x}(n-1) = \begin{bmatrix} x(n-1) & x(n-2) & \cdots & x(n-N) \end{bmatrix}^T$ and $P = 1$.

The optimal Wiener-Hopf solution $\mathbf{w}_{\text{opt}} = \mathbf{R}^{-1}\mathbf{r}$ requires the knowledge of both magnitudes: the correlation matrix $\mathbf{R}$ of the input matrix $\mathbf{X}$ and the cross-correlation vector $\mathbf{r}$ between the input vector and desired answer $\mathbf{d}$. That is the reason why it has little practical value. So that the linear system given by (21) has solution, the correlation matrix $\mathbf{R}$ must be nonsingular. It is possible to estimate both magnitudes according to the windowing method of the input vector.

The *sliding window* method uses the sample data within a window of finite length N. Correlation matrix and cross-correlation vector are estimated averaging in time,

$$\mathbf{R}(n) = \mathbf{X}(n)\mathbf{X}^H(n)/N, \tag{24}$$

$$\mathbf{r}(n) = \mathbf{X}(n)\mathbf{d}^*(n)/N.$$

The method that estimates the autocorrelation matrix like in (24) with samples organized as in (15) is known as the *covariance method*. The matrix that results is positive semidefinite but it is not Toeplitz.

The *exponential window* method uses a recursive estimation according to certain forgetfulness factor λ in the rank $0 < \lambda < 1$,

$$\mathbf{R}(n) = \lambda\mathbf{R}(n-1) + \mathbf{X}(n)\mathbf{X}^H(n), \tag{25}$$

$$\mathbf{r}(n) = \lambda\mathbf{r}(n-1) + \mathbf{X}(n)\mathbf{d}^*(n).$$

When the excitation signal to the adaptive system is not stationary and the unknown system is time-varying, the exponential and sliding window methods allow the filter to forget or to eliminate errors happened farther in time. The price of this forgetfulness is deterioration in the fidelity of the filter estimation (Gay & Benesty, 2000).
A recursive estimator has the form defined in (19). In each iteration, the update of the estimator is made in the $\Delta\mathbf{w}(n)$ direction. For all the optimization deterministic iterative schemes, a stochastic algorithm approach exists. All it takes is to replace the terms related to the cost function and calculate the approximate values by each new set of input/output samples. In general, most of the adaptive algorithms turn a stochastic optimization problem into a deterministic one and the obtained solution is an approximation to the one of the original problem.

The gradient $\mathbf{g} = \nabla J(\mathbf{w}) = \frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = -2\mathbf{X}\mathbf{d}^* + 2\mathbf{X}\mathbf{X}^H\mathbf{w}$, can be estimated by means of $\mathbf{g} = -2(\mathbf{r} + \mathbf{R}\mathbf{w})$, or by the equivalent one $\mathbf{g} = -\mathbf{X}\mathbf{e}^*$, considering $\mathbf{R}$ and $\mathbf{r}$ according to (24) or (25). It is possible to define recursive updating strategies, per each l stage, for lattice structures as

$$\mathbf{K}_l(n+1) = \mathbf{K}_l(n) + \Delta\mathbf{K}_l(n), \tag{26}$$

$$\Delta\mathbf{K}_l(n) = \arg\min J(\mathbf{K}_l).$$

2.2.3 Optimization strategies

Several strategies to solve $\Delta\mathbf{w} = \arg\min J(\mathbf{w})$ are proposed (Glentis et al., 1999) (usually of the least square type). It is possible to use a quadratic (second order) approximation of the error-performance surface around the current point denoted $\mathbf{w}(n)$. Recalling the second-order *Taylor series* expansion of the cost function $J(\mathbf{w})$ around $\mathbf{w}(n)$, with $\Delta\mathbf{w} = \mathbf{w} - \mathbf{w}(n)$, you have

$$J(\mathbf{w} + \Delta\mathbf{w}) \cong J(\mathbf{w}) + \Delta\mathbf{w}^H\nabla J(\mathbf{w}) + \frac{1}{2}\Delta\mathbf{w}^H\nabla^2 J(\mathbf{w})\Delta\mathbf{w} \tag{27}$$

Deterministic iterative optimization schemes require the knowledge of the cost function, the *gradient* (first derivatives) defined in (29) or the *Hessian* matrix (second order partial derivatives) defined in (45,52) while *stochastic recursive* methods replace these functions by impartial estimations.

$$\nabla J(\mathbf{w}) = \left[\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}_1} \quad \frac{\partial J(\mathbf{w})}{\partial \mathbf{w}_2} \quad \cdots \quad \frac{\partial J(\mathbf{w})}{\partial \mathbf{w}_L}\right]^T, \tag{28}$$

$$\nabla^2 J(\mathbf{w}) = \begin{bmatrix} \frac{\partial^2 J(\mathbf{w})}{\partial \mathbf{w}_1 \partial \mathbf{w}_1} & \frac{\partial^2 J(\mathbf{w})}{\partial \mathbf{w}_1 \partial \mathbf{w}_2} & \cdots & \frac{\partial^2 J(\mathbf{w})}{\partial \mathbf{w}_1 \partial \mathbf{w}_L} \\ \frac{\partial^2 J(\mathbf{w})}{\partial \mathbf{w}_2 \partial \mathbf{w}_1} & \frac{\partial^2 J(\mathbf{w})}{\partial \mathbf{w}_2 \partial \mathbf{w}_2} & \cdots & \frac{\partial^2 J(\mathbf{w})}{\partial \mathbf{w}_2 \partial \mathbf{w}_L} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 J(\mathbf{w})}{\partial \mathbf{w}_L \partial \mathbf{w}_1} & \frac{\partial^2 J(\mathbf{w})}{\partial \mathbf{w}_L \partial \mathbf{w}_2} & \cdots & \frac{\partial^2 J(\mathbf{w})}{\partial \mathbf{w}_L \partial \mathbf{w}_L} \end{bmatrix}^T . \tag{29}$$

The vector $\mathbf{g}(n) = \nabla J(\mathbf{w})$ is the *gradient* evaluated at $\mathbf{w}(n)$, and the matrix $\mathbf{H}(n) = \nabla^2 J(\mathbf{w})$ is the *Hessian* of the cost function evaluated at $\mathbf{w}(n)$.

Several first order adaptation strategies are: to choose a starting initial point $\mathbf{w}(0)$, to increment election $\Delta\mathbf{w}(n) = \mu(n)\mathbf{g}(n)$; two decisions are due to take: movement direction $\mathbf{g}(n)$ in which the cost function decreases fastest and the step-size in that direction $\mu(n)$. The iteration stops when a certain level of error is reached $\Delta\mathbf{w}(n) < \xi$,

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu(n)\mathbf{g}(n) . \tag{30}$$

Both parameters $\mu(n)$, $\mathbf{g}(n)$ are determined by a cost function. The second order methods generate values close to the solution in a minimum number of steps but, unlike the first order methods, the second order derivatives are very expensive computationally. The adaptive filters and its performance are characterized by a selection criteria of $\mu(n)$ and $\mathbf{g}(n)$ parameters.

Method	Definition	Comments
SD	$\mu(n) = -\frac{\|\mathbf{g}\|^2}{\mathbf{g}^H \mathbf{R} \mathbf{g}}$	Steepest-Descent
CG	(See below)	Conjugate Gradient
NR	$\mu(n) = \alpha \mathbf{Q}$	Newton-Raphson

Table 2. Optimization methods.

The *optimization methods* are useful to find the minimum or maximum of a quadratic function. Table 2 summarizes the optimization methods. SD is an iterative optimization procedure of easy implementation and computationaly very cheap. It is recommended with cost functions that have only one minimum and whose gradients are isotropic in magnitude respect to any direction far from this minimum. NR method increases SD performance using a carefully selected weighting matrix. The simplest form of NR uses $\mathbf{Q} = \mathbf{R}^{-1}$. *Quasy-Newton*

methods (QN) are a special case of NR with $\mathbf{Q}$ simplified to a constant matrix. The solution to $J(\mathbf{w})$ is also the solution to the *normal equation* (21). The *conjugate gradient* (CG) (Boray & Srinath, 1992) was designed originally for the minimization of convex quadratic functions but, with some variations, it has been extended to the general case. The first CG iteration is the same that the SD algorithm and the new successive directions are selected in such a way that they form a set of vectors mutually conjugated to the Hessian matrix (corresponding to the autocorrelation matrix, $\mathbf{R}$), $\mathbf{q}_i^H \mathbf{R} \mathbf{q}_j = 0, \forall i \neq j$. In general, CG methods have the form

$$\mathbf{q}_l = \begin{cases} -\mathbf{g}_l, & l = 1 \\ -\mathbf{g}_l + \beta_l \mathbf{q}_{l-1}, & l > 1 \end{cases} \tag{31}$$

$$\mu_l = \frac{\langle \mathbf{g}_l, \mathbf{q}_l \rangle}{\langle \mathbf{q}_l, \mathbf{g}_l - \mathbf{p}_l \rangle}, \tag{32}$$

$$\beta_l = \frac{\|\mathbf{g}_l\|^2}{\|\mathbf{g}_{l-1}\|^2}, \tag{33}$$

$$\mathbf{w}_{l+1}(n) = \mathbf{w}_l(n) + \mu_l(n)\mathbf{q}_l . \tag{34}$$

CG spans the search directions from the gradient in course, $\mathbf{g}$, and a combination of previous $\mathbf{R}$-conjugated search directions. β guarantees the $\mathbf{R}$-conjugation. Several methods can be used to obtain β. This method (33) is known as Fleetcher-Reeves. The gradients can be obtained as $\mathbf{g} = \nabla J(\mathbf{w})$ and $\mathbf{p} = \nabla J(\mathbf{w} - \mathbf{g})$.

The *memoryless* LS methods in Table 3 use the instantaneous squared error cost function $J(\mathbf{w}) = e^2(n)$. The descent direction for all is a gradient $\mathbf{g}(n) = \mathbf{x}(n)e^*(n)$. The LMS algorithm is a stochastic version of the SD optimization method. NLMS frees the convergence speed of the algorithm with the power signal. FNLMS filters the signal power estimation; $0 < \beta < 1$ is a weighting factor. PNLMS adaptively controls the size of each weight.

Method	Definition	Comments
LMS	$\mu(n) = \alpha$	Least Means Squares
NLMS	$\mu(n) = \frac{\alpha}{\|\mathbf{x}(n)\|^2 + \delta}$	Normalized LMS
FNLMS	$\mu(n) = \frac{\alpha}{\mathbf{p}(n)}$	Filtered NLMS
PNLMS	$\mu(n) = \frac{\alpha \mathbf{Q}}{\mathbf{x}^H(n)\mathbf{Q}\mathbf{x}(n) + \delta}$	Proportionate NLMS

Table 3. Memoryless Least-Squares (LS) methods.

Method	Definition	Comments
RLS	$\mu(n)=\mathbf{R}^{-1}(n)$ $\mathbf{g}(n)=\mathbf{x}(n)e^{*}(n)$	Recursive Least-Squares
LMS-SW	$\mu(n)=\frac{\|\mathbf{g}(n)\|^{2}}{\mathbf{g}^{H}(n)\mathbf{X}(n)\mathbf{X}^{H}(n)\mathbf{g}(n)+\delta}$ $\mathbf{g}(n)=\mathbf{X}(n)e^{*}(n)$	Sliding-Window LMS
APA	$\mu(n)=\frac{\alpha}{\mathbf{X}(n)\mathbf{X}^{H}(n)+\delta\mathbf{I}}$ $\mathbf{g}(n)=\mathbf{X}(n)e^{*}(n)$	Affine Projection Algorithm
PRA	$\mathbf{w}(n+1)=\mathbf{w}(n-N+1)+\mu(n)\mathbf{g}(n)$ $\mu(n)=\frac{\alpha}{\mathbf{X}(n)\mathbf{X}^{H}(n)+\delta\mathbf{I}}$ $\mathbf{g}(n)=\mathbf{X}(n)e^{*}(n)$	Partial Rank Algorithm
DLMS	$\mu(n)=\frac{1}{\langle\mathbf{x}(n),\mathbf{z}(n)\rangle}$ $\mathbf{g}(n)=\mathbf{z}(n)e^{*}(n)$ $\mathbf{z}(n)=\mathbf{x}(n)+\frac{\langle\mathbf{x}(n),\mathbf{x}(n-1)\rangle}{\|\mathbf{x}(n-1)\|^{2}}\mathbf{x}(n-1)$	Decorrelated LMS
TDLMS	$\mu(n)=\frac{\alpha\mathbf{Q}}{\|\mathbf{x}(n)\|^{2}}$, $\mathbf{Q}\,\mathbf{Q}^{H}=1$ $\mathbf{g}(n)=\mathbf{x}(n)e^{*}(n)$	Transform-Domain DLMS

Table 4. Least-Squares with memory methods.

$\mathbf{Q}$ is a diagonal matrix that weights the individual coefficients of the filters, α is a *relaxation constant* and δ guarantees that the denominator never becomes zero. These algorithms are very cheap computationally but their convergence speed depends strongly on the *spectral condition number* of the autocorrelation matrix $\mathbf{R}$ (that relate the extreme eigenvalues) and can get to be unacceptable as the correlation between the P channels increases.

The *projection algorithms* in Table 4 modify the filters coefficients in the input vector direction and on the subspace spanned by the $N-1$ redecessors. RLS is a recursive solution to the normal equation that uses MSE as cost function. There is an alternative fast version FRLS. LMS-SW is a variant of SD that considers a data window. The step can be obtained by a linear search. APA is a generalization of RLS and NLMS. APA is obtained by projecting the adaptive coefficients vector $\mathbf{w}$ in the *affine subspace*. The affine subspace is obtained by means of a translation from the orthogonal origin to the subspace where the vector $\mathbf{w}$ is projected. PRA is a strategy to reduce the computational complexity of APA by updating the coefficients every N samples. DLMS replaces the system input by an orthogonal component to the last input (order 2). These changes the updating vector direction of the correlated input signals so that these ones correspond to uncorrelated input signals. TDLMS decorrelates into transform domain by means of a $\mathbf{Q}$ matrix.

The adaptation of the transversal section of the joint-process estimator in the lattice-ladder structure depends on the gradient $\mathbf{g}(n)$ and, indirectly, on the reflection coefficients, through the backward predictor, $\mathbf{g}(n)=\mathbf{b}(n)$. However, the reflection coefficient adaptation depends on the gradient of $y(n)$ with respect to them

$$\nabla J(\mathbf{K})=\left[\frac{\partial J(\mathbf{K})}{\partial \mathbf{K}_1}\quad \frac{\partial J(\mathbf{K})}{\partial \mathbf{K}_2}\quad \cdots\quad \frac{\partial J(\mathbf{K})}{\partial \mathbf{K}_L}\right]^T, \tag{35}$$

$$\nabla^2 J(\mathbf{K})=\begin{bmatrix} \frac{\partial^2 J(\mathbf{K})}{\partial \mathbf{K}_1\partial \mathbf{K}_1} & \frac{\partial^2 J(\mathbf{K})}{\partial \mathbf{K}_1\partial \mathbf{K}_2} & \cdots & \frac{\partial^2 J(\mathbf{K})}{\partial \mathbf{K}_1\partial \mathbf{K}_L} \\ \frac{\partial^2 J(\mathbf{K})}{\partial \mathbf{K}_2\partial \mathbf{K}_1} & \frac{\partial^2 J(\mathbf{K})}{\partial \mathbf{K}_2\partial \mathbf{K}_2} & \cdots & \frac{\partial^2 J(\mathbf{K})}{\partial \mathbf{K}_2\partial \mathbf{K}_L} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 J(\mathbf{K})}{\partial \mathbf{K}_L\partial \mathbf{K}_1} & \frac{\partial^2 J(\mathbf{K})}{\partial \mathbf{K}_L\partial \mathbf{K}_2} & \cdots & \frac{\partial^2 J(\mathbf{K})}{\partial \mathbf{K}_L\partial \mathbf{K}_L} \end{bmatrix}. \tag{36}$$

In a more general case, concerning to a multichannel case, the gradient matrix can be obtained as $\mathbf{G}=\nabla J(\mathbf{K})$. Two recursive updatings are necessary

$$\mathbf{w}_l(n+1)=\mathbf{w}_l(n)+\mu_l(n)\mathbf{g}_l(n), \tag{37}$$

$$\mathbf{K}_l(n+1)=\mathbf{K}_l(n)+\lambda_l(n)\mathbf{G}_l(n) \tag{38}$$

Table 5 resumes the least-squares for lattice.

GAL is a NLMS extension for a lattice structure that uses two cost functions: instantaneous squared error for the tranversal part and prediction MSE for the lattice-ladder part, $\mathbf{B}_l(n)=\beta\mathbf{B}_l(n-1)+(1-\beta)\left(\left|\mathbf{f}_l(n)\right|^2+\left|\mathbf{b}_l(n-1)\right|^2\right)$, where α and σ are relaxation factors.

Method	Definition	Comments
GAL	$\mu_l(n)=\frac{\alpha}{\left\|\mathbf{b}_l(n)\right\|^2}$ $\mathbf{g}_l(n)=\mathbf{b}_l(n)e^*(n)$ $\lambda_l(n)=\frac{\sigma}{\mathbf{B}_{l-1}(n)}$ $\mathbf{G}_l(n)=\mathbf{b}_{l-1}(n-1)\mathbf{f}_l^H(n)+\mathbf{b}_l(n)\mathbf{f}_{l-1}^H(n-1)$	Gradient Adaptive Lattice
CGAL	(See below)	CG Adaptive Lattice

Table 5. Least-Squares for lattice.

For CGAL, the same algorithm described in (31-34) is used but it is necessary to rearrange the gradient matrices of the lattice system in a column vector. It is possible to arrange the gradients of all lattice structures in matrices. $\mathbf{U}(n)=\left[\mathbf{g}_1^T(n)\quad \mathbf{g}_2^T(n)\quad \cdots\quad \mathbf{g}_P^T(n)\right]^T$ is the $P\times L$ gradient matrix with respect to the transversal coefficients, $\mathbf{g}_p(n)=\left[g_{p1}\quad g_{p2}\quad \cdots\quad g_{pL}\right]^T$, $p=1\ldots P$. $\mathbf{V}(n)=\left[\mathbf{G}_1(n)\quad \mathbf{G}_2(n)\quad \cdots\quad \mathbf{G}_P(n)\right]^T$ is a $P\times(L-1)P$ gradient matrix with respect to the reflection coefficients; and rearranging these matrices in one single column vector, $\left[\mathbf{u}^T\quad \mathbf{v}^T\right]^T$ is obtained with

$$\mathbf{u}=\left[g_{11}\quad \cdots\quad g_{1L}\quad g_{21}\quad \cdots\quad g_{2L}\quad \cdots\quad g_{P1}\quad \cdots\quad g_{PL}\right]^T,$$
$$\mathbf{v}=\left[G_{111}\quad \cdots\quad G_{1P1}\quad \cdots\quad G_{P11}\quad \cdots\quad G_{PP1}\quad G_{112}\quad \cdots\quad G_{PP(L-1)}\right]^T.$$

$$\mathbf{q}_l=\begin{cases}-\mathbf{g}_l, & l=1\\ -\mathbf{g}_l+\beta_l\mathbf{q}_{l-1}, & l>1\end{cases} \tag{39}$$

$$\mathbf{g}_l=\begin{cases}\left[\mathbf{u}^{\mathrm{T}}\quad \mathbf{v}^T\right]^{\mathrm{T}}, & l=1\\ \alpha\mathbf{g}_{l-1}+(1-\alpha)\left[\mathbf{u}^{\mathrm{T}}\quad \mathbf{v}^T\right]^{\mathrm{T}}, & l>1\end{cases} \tag{40}$$

$$\beta_l=\frac{\left\|\mathbf{g}_l\right\|^2}{\left\|\mathbf{g}_{l-1}\right\|^2}, \tag{41}$$

$$\mathbf{w}_{l+1}=\mathbf{w}_l+\mu\,\mathbf{u}_l, \tag{42}$$

$$\mathbf{K}_{l+1}=\mathbf{K}_l+\lambda_l\mathbf{V}_l. \tag{43}$$

The time index n has been removed by simplicity. $0<\alpha<1$ is a forgetfulness factor which weights the innovation importance specified in a low-pass filtering in (40). The gradient selection is very important. A mean value that uses more recent coefficients is needed for gradient estimation and to generate a vector with more than one conjugate direction (40).

3. Multirate Adaptive Filtering

The adaptive filters used for speech enhancement are probably very large (due to the AIRs). Multirate adaptive filtering works at a lower sampling rate that allows reducing the complexity (Shynk, 1992). Depending on how the data and filters are organized, these approaches may upgrade in performance and avoid end-to-end delay. Multirate schemes adapt the filters in smaller sections at lower computational cost. This is only necessary for real-time implementations. Two approaches are considered. The *subband adaptive filtering* approach splits the spectra of the signal in a number of subbands that can be adapted independently and afterwards the filtering can be carried out in a fullband. The *frequency-domain adaptive filtering* partitions the signal in time-domain and projects it into a transformed domain (i.e. frequency) using better properties for adaptive processing. In both cases the input signals are transformed into a more desirable form before adaptive processing and the adaptive algorithms operate in transformed domains, whose basis functions orthogonalize the input signal, speeding up the convergence. The *partitioned convolution* is necessary for fullband delayless convolution and can be seen as an efficient frequency-domain convolution.

3.1 Subband Adaptive Filtering

The fundamental structure for subband adaptive filtering is obtained using band-pass filters as basis functions and replacing the fixed gains for adaptive filters. Several implementations are possible. A typical configuration uses an *analysis filter bank*, a processing stage and a *synthesis filter bank*. Unfortunately, this approach introduces an end-to-end delay due to the synthesis filter bank. Figure 5 shows an alternative structure which adapts in subbands and filters in full-band to remove this delay (Reilly et al., 2002).

K is the *decimation ratio*, M is the number of bands and N is the prototype filter length. k is the low rate time index. The sample rate in subbands is reduced to F_s/K. The input signal per channel is represented by a vector $\mathbf{x}_p(n)=\begin{bmatrix} x(n) & x(n-1) & \cdots & x(n-L+1)\end{bmatrix}^T$, $p=1\ldots P$. The adaptive filter in full-band per channel $\mathbf{w}_p=\begin{bmatrix} w_{p1} & w_{p2} & \cdots & w_{PL}\end{bmatrix}^T$ is obtained by means of the $\mathbf{T}$ operator as

$$\mathbf{w}_p = \Re\left\{\sum_{m=1}^{M/2}\left(\mathbf{h}_{m_{\downarrow K}} * \mathbf{w}_{pm}\right)_{\uparrow K} * \mathbf{g}_m\right\}, \tag{44}$$

from the subband adaptive filters per each channel $\mathbf{w}_{pm}$, $p=1\ldots P$, $m=1\ldots M/2$ (Reilly et al., 2002). The subband filters are very short, of length $C=\left\lceil\frac{L+N-1}{K}\right\rceil-\left\lceil\frac{N}{K}\right\rceil+1$, which

allows to use much more complex algorithms. Although the input signal vector per channel $\mathbf{x}_p(n)$ has size $L \times 1$, it acts as a delay line which, for each iteration k, updates K samples. $\downarrow K$ is an operator that means downsampling for a K factor and $\uparrow K$ upsampling for a K factor. $\mathbf{g}_m$ is a synthesis filter in subband m obtained by modulating a prototype filter. $\mathbf{H}$ is a *polyphase matrix* of a *generalized discrete Fourier transform* (GDFT) of an oversampled ($K < M$) analysis filter bank (Crochiere & Rabiner, 1983). This is an efficient implementation of a uniform complex modulated analysis filter bank. This way, only a *prototype filter* $\mathbf{p}$ is necessary; the prototype filter is a low-pass filter.

The band-pass filters are obtained modulating a prototype filter. It is possible to select different adaptive algorithms or parameter sets for each subband. For delayless implementation, the full-band convolution may be made by a *partitioned convolution*.

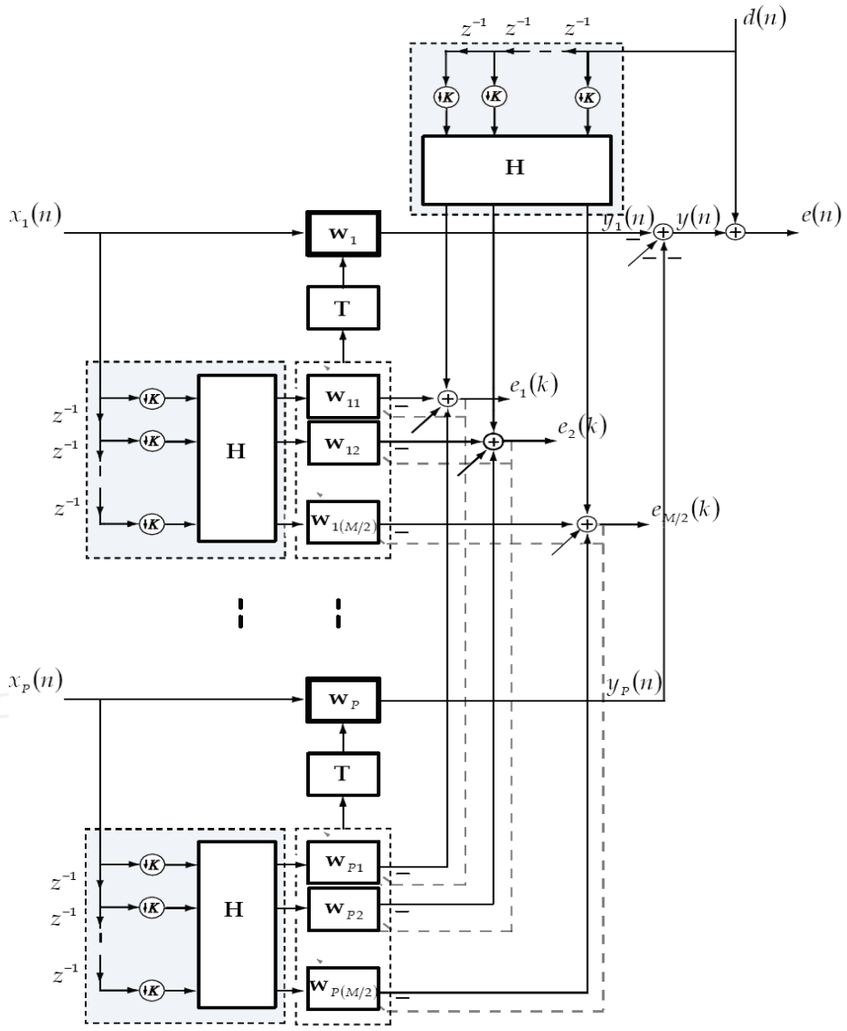

Fig. 5. Subband adaptive filtering. This configuration is known as *open-loop* because the error is in the time-domain. An alternative *closed-loop* can be used where the error is in the subband-domain. Gray boxes correspond to efficient polyphase implementations. See details in (Reilly et al., 2002).

3.2 Frequency Domain Adaptive Filtering

The basic operation in *frequency-domain adaptive filtering* (FDAF) is to transform the input signal in a "more desirable" form before the adaptation process starts (Shynk, 1992) in order to work with matrix multiplications instead of dealing with slow convolutions.

The frequency-domain transform employs one or more *discrete Fourier transforms* (DFT), **T** operator in Fig. 6, and can be seen as a pre-processing block that generates decorrelated output signals. In the more general FDAF case, the output of the filter in the time-domain (3) can be seen as the direct frequency-domain translation of the block LMS (BLMS) algorithm. That efficiency is obtained taking advantage of the equivalence between the linear convolution and the circular convolution (multiplication in the frequency-domain).

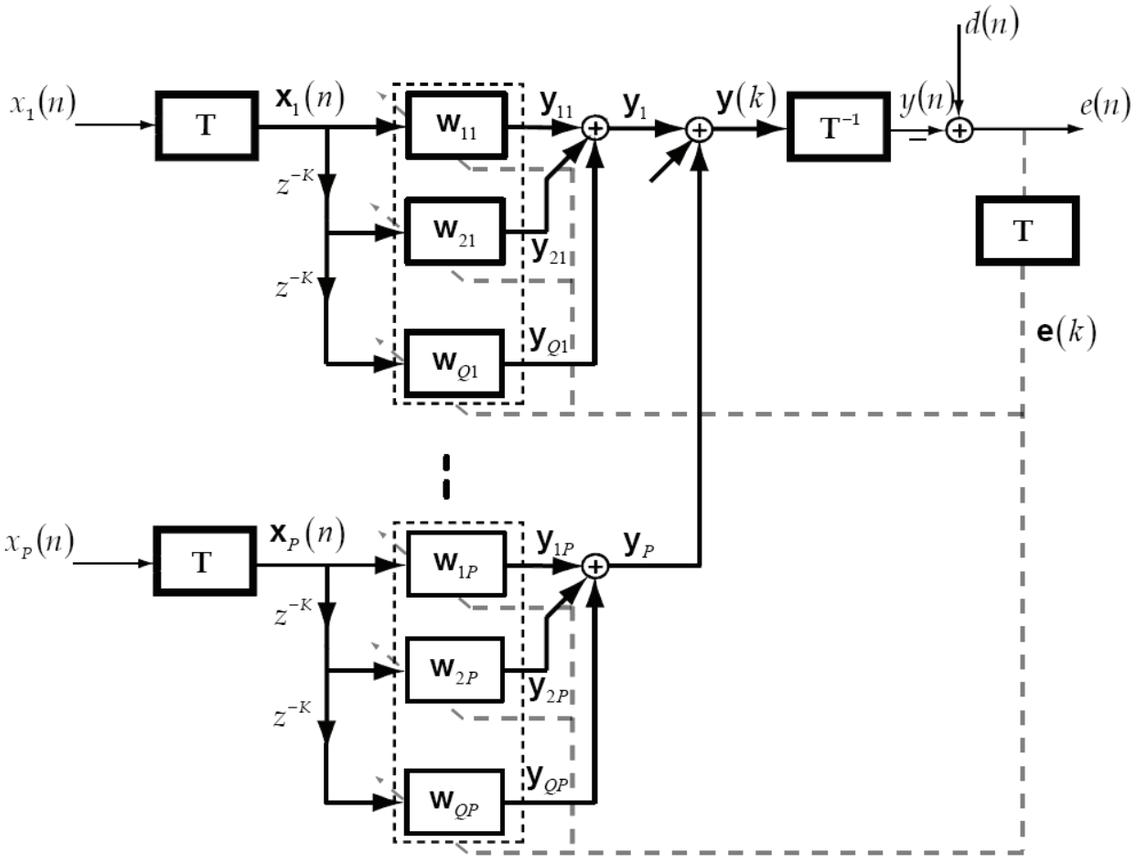

Fig. 6. Partitioned block frequency-domain adaptive filtering.

It is possible to obtain the linear convolution between a finite length sequence (filter) and an infinite length sequence (input signal) with the overlapping of certain elements of the data sequence and the retention of only a subgroup of the DFT.

The *partitioned block frequency-domain adaptive filtering* (PBFDAF) was developed to deal efficiently with such situations (Paez & Otero, 1992). The PBFDAF is a more efficient implementation of the LMS algorithm in the frequency-domain. It reduces the computational burden and bounds the user-delay. In general, the PBFDAF is widely used due to its good trade-off between speed, computational complexity and overall latency.

However, when working with long AIRs, the convergence properties provided by the algorithm may not be enough. This technique makes a sequential partition of the impulse response in the time-domain prior to a frequency-domain implementation of the filtering operation.
This time segmentation allows setting up individual coefficient updating strategies concerning different sections of the adaptive canceller, thus avoiding the need to disable the adaptation in the complete filter. In the PBFDAF case, the filter is partitioned transversally in an equivalent structure. Partitioning $\mathbf{w}_p$ in Q segments (K length) we obtain

$$y(n) = \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{m=0}^{K-1} x_p(n - qK - m) w_{p(qK+m)} , \tag{45}$$

Where the total filter length L, for each channel, is a multiple of the length of each segment $L = QK$, $K \leq L$. Thus, using the appropriate data sectioning procedure, the Q linear convolutions (per channel) of the filter can be independently carried out in the frequency-domain with a total delay of K samples instead of the QK samples needed by standard FDAF implementations. Figure 6 shows the block diagram of the algorithm using the overlap-save method. In the frequency-domain with matricial notation, (45) can be expressed as

$$\mathbf{Y} = \mathbf{X} \otimes \mathbf{W} , \tag{46}$$

where $\mathbf{X} = \mathbf{FX}$ represents a matrix of dimensions $M \times Q \times P$ which contains the Fourier transform of the Q partitions and P channels of the input signal matrix $\mathbf{X}$. $\mathbf{F}$ represents the DFT matrix defined as $\mathbf{F} = W_M^{-mn}$ of size $M \times M$ and $\mathbf{F}^{-1}$ as its inverse. Of course, in the final implementation, the DFT matrix should be substituted by much more efficient *fast Fourier transform* (FFT). Being $\mathbf{X}$, $2K \times P$ -dimensional (supposing 50% overlapping between the new block and the previous one). It should be taken into account that the algorithm adapts every K samples. $\mathbf{W}$ represents the filter coefficient matrix adapted in the frequency-domain (also $M \times Q \times P$ -dimensional) while the $\otimes$ operator multiplies each of the elements one by one; which, in (46), represents a *circular convolution*. The output vector $\mathbf{y}$ can be obtained as the double sum (rows) of the $\mathbf{Y}$ matrix. First we obtain a $M \times P$ matrix which contains the output of each channel in the frequency-domain $\mathbf{y}_P$, $p = 1 \ldots P$, and secondly, adding all the outputs we obtain the whole system output, $\mathbf{y}$. Finally, the output in the time-domain is obtained by using $\mathbf{y} = \text{last } K \text{ components of } \mathbf{F}^{-1}\mathbf{y}$. Notice that the sums are performed prior to the time-domain translation. This way we reduce $(P-1)(Q-1)$ FFTs in the complete filtering process. As in any adaptive system the error can be obtained as

$$\mathbf{e} = \mathbf{d} - \mathbf{y} \tag{47}$$

with $\mathbf{d} = \begin{bmatrix} d(mK) & d(mK+1) & \cdots & d((m+1)K-1) \end{bmatrix}^T$. The error in the frequency-domain (for the actualization of the filter coefficients) can be obtained as

$$\mathbf{e} = \mathbf{F} \begin{bmatrix} \mathbf{0}_{K \times 1} \\ \mathbf{e} \end{bmatrix}. \tag{48}$$

As we can see, a block of K zeros is added to ensure a correct linear convolution implementation. In the same way, for the block gradient estimation, it is necessary to employ the same error vector in the frequency-domain for each partition q and channel p. This can be achieved by generating an error matrix $\mathbf{E}$ with dimensions $M \times Q \times P$ which contains replicas of the error vector, defined in (48), of dimensions P and Q ($\mathbf{E} \leftarrow \mathbf{e}$ in the notation). The actualization of the weights is performed as

$$\mathbf{W}(m+1) = \mathbf{W}(m) + \mu(m)\mathbf{G}(m). \tag{49}$$

The instantaneous gradient is estimated as

$$\mathbf{G} = -\mathbf{X}^* \otimes \mathbf{E}. \tag{50}$$

This is the unconstrained version of the algorithm which saves two FFTs from the computational burden at the cost of decreasing the convergence speed. The constrained version basically makes a gradient projection. The gradient matrix is transformed into the time-domain and is transformed back into the frequency-domain using only the first K elements of $\mathbf{G}$ as

$$\mathbf{G} = \mathbf{F} \begin{bmatrix} \mathbf{G} \\ \mathbf{0}_{K \times Q \times P} \end{bmatrix}. \tag{51}$$

A conjugate gradient version of PBFDAF is possible by transforming the gradient matrix to vectors and reverse (García, 2006). The vectors $\mathbf{g}$ and $\mathbf{p}$ in (31,32) should be changed by $\mathbf{g}_l \leftarrow \bar{\mathbf{G}}_l$, $\bar{\mathbf{G}}_l = \nabla J(\mathbf{W}_l)$ and $\mathbf{p}_l \leftarrow \bar{\mathbf{P}}_l$, $\bar{\mathbf{P}}_l = \nabla J\left(\mathbf{W}_l - \bar{\mathbf{G}}_l\right)$, with gradient estimation obtained by averaging the instantaneous gradient estimates over N past values $\bar{\mathbf{G}}_l = \nabla J(\mathbf{W}_l) = \frac{2}{N} \sum_{k=1}^{N} \mathbf{G}_{l-k} \Big|_{\mathbf{W}_l, \mathbf{X}_{l-k}, \mathbf{d}_{l-k}}$.

3.3 Partitioned Convolution

For each input i, the AIR matrix, $\mathbf{V}$, is reorganized in a column vector $\mathbf{v} = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_P \end{bmatrix}^T$ of size $N = LP \times 1$ and initially partitioned in a reasonable number Q of equally-sized blocks $\mathbf{v}_q$, $q = 1 \ldots Q$, of length K. Each of these blocks is treated as a

separate impulse response, and convolved by a standard overlap-and-save process, using $\mathbf{T}$ operator (FFT windows of length L). All input data are processed in overlapped blocks of L samples (each block at $L-K$ samples to the last). Each block is zero-padded to length L (typically equal to $2K$), and transformed with FFT so that a collection of Q frequency-domain filters $\mathbf{v}_q$ is obtained. The results of the multiplications of these Q filters $\mathbf{v}_q$ with the FFTs of the Q input blocks are summed, producing the same result as the unpartitioned convolution, by means of proper delays applied to the blocks of convolved data. Finally an $\mathbf{T}^{-1}$ operator (IFFT) of the first acummulator is made to submmit an output data block (obviosly only the last $L-K$ block samples). Each block of input data needs to be FFT transformed just once, and thus the number of forward FFTs is minimized (Armelloni et al., 2003). The main advantage compared to unpartitioned convolution is that the latency of the whole filtering processing is just M points instead of $2N$, and thus the I/O delay is kept to a low value, provided that the impulse response is partitioned in a sensible number of chunks (8-32). Figure 7 outlines the whole process.

Suposse that $A=\left(A_1,A_2,\ldots,A_Q\right)$ is a set of multiplications of the first data block and $B=\left(B_1,B_2,\ldots,B_Q\right)$ the second, then for time-index 1 it is only necessary to consider A_1. At the next index-time, corresponding to $K+1$ samples, the sum is formed with $\left(B_Q,B_1+A_2,B_2+A_3,\ldots,B_{Q-1}+A_Q\right)$. If $C=\left(C_1,C_2,\ldots,C_Q\right)$ corresponds to the third block the s formed with $\left(C_Q,C_1+B_2+A_3,C_2+B_3+A_4,\ldots,C_{Q-1}+B_Q\right)$. An efficient implementation f this sum can be implemented using a double buffering technique (Armelloni et al., 2003).

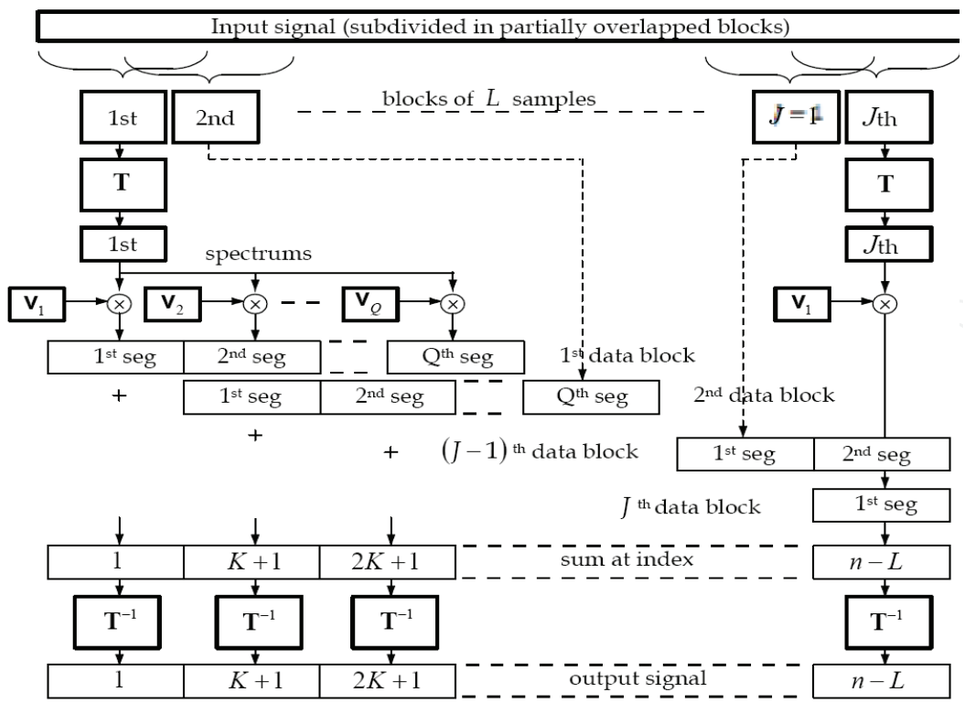

Fig. 7. Partitioned convolution. Each output signal block is produced taking only the $L-K$ last samples of the block.

3.4 Delayless Approach for Real-Time Applications

The filtering operation can be made *delayless* by operating the first block in the time-domain (direct convolution) while the rest of the blocks continue to operate in the frequency domain (Morgan & Thi, 1995). The fast convolution starts after the samples have been processed for direct convolution. The direct convolution allows giving samples to the output while data is incomming. This approach is applicable to the multirate frameworks described.

4. Applications

Once the theoretical foundations of the adaptive filtering have been reviewed, the most important techniques that can be applied to speech enhancement are introduced.

4.1 Spectral Equalization

The adaptive spectral equalization is widely used for noise suppression and corresponds to the single-input and single-output (SISO) *estimator* application (class a, Fig. 2); a single microphone, $P=1$, is employed. This approach estimates a *noiseprint spectra* and subtracts it from the whole signal in the frequency-domain. The Wiener filter estimator is the result of estimating $y(n)$ from $s(n)$ that minimizes the MSE $\left\|y(n)-s(n)\right\|^2$ given by $\mathbf{y}=\mathbf{Qx}$, $\mathbf{x}=\mathbf{s}+\mathbf{r}$, and that results.

$$\mathbf{q} \cong \frac{|\mathbf{x}|^2 - |\mathbf{d}|^2}{|\mathbf{x}|^2}, \tag{52}$$

$\mathbf{Q}=\mathrm{diag}\{[\mathbf{q}_1 \quad \mathbf{q}_2 \quad \cdots \quad \mathbf{q}_M]\}$ is a diagonal matrix which contains the spectral gain in the frequency-domain; normally $\mathbf{T}$ is a short-time Fourier transform (STFT), suitable for not stationary signals, and $\mathbf{T}^{-1}$ its inverse. In this case this algorithm is known as short-time spectral attenuation (STSA). The $M\times 1$ vector $\mathbf{q}$ contains the main diagonal components of $\mathbf{Q}$. $\mathbf{d}$ is the noise spectrum (normaly unknown). In this case an estimation of the noiseprint spectra $\mathbf{d}=\hat{\mathbf{r}}$ from the mixture $\mathbf{x}$ (noisy signal) is necessary (in intervals when the speech is absent and only the noise is present). The STFT is defined as $\mathbf{X}_k=\sum_{n=1}^{N} h(n)x(m-n)W_M^{-mk}$, $m=0\ldots M-1$, where k is the time index about which the short-time spectrum is computed, m is the discrete frequency index, $h(n)$ is an analysis window, N dictates the duration over which the transform is computed, and M is the number of frequency bins at which the STFT is computed.

For stationary signals the squared-magnitude of the STFT provides a sample estimate of the power spectrum of the underlying random process. This form (53) is basic to nearly all the noise reduction methods investigated over last forty years (Gay & Benesty, 2000). The specific form to obtain $\mathbf{Q}$ is known as the *suppresion rule*.

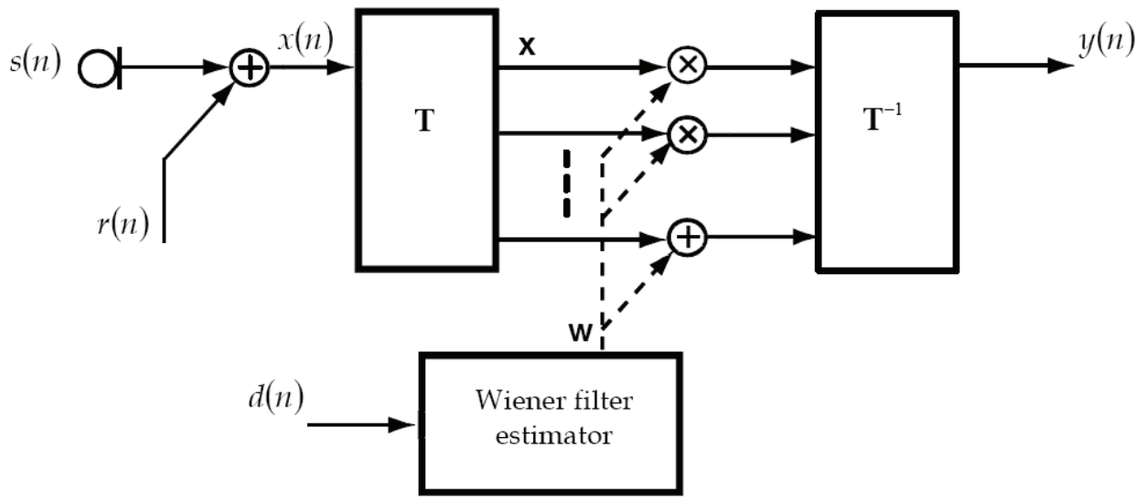

Fig. 8. Spectral equalization.

Power Subtraction

An alternative estimate from Wiener's theory is achieved assuming that $\mathbf{s}$ can be estimated if its magnitud is estimated as

$$\hat{\mathbf{s}} = \sqrt[\alpha]{|\mathbf{x}|^{\alpha} - \beta\,|\mathbf{d}|^{\alpha}}\ , \tag{53}$$

and the phase of the noisy signal $\mathbf{x}$ can be used, if its SNR is reasonably high, in place of the phase of $\mathbf{s}$. α is an exponent and β is a parameter introduced to control the amount of noise to be subtracted ($\beta = 1$ for full subtraction and $\beta > 1$ for over subtraction). A paramount issue in spectral subtraction is to obtain a good noise estimate; its accuracy greatly affects the noise reduction performance (Benesty & Huang, 2003).

4.2 Linear Prediction

The *adaptive linear prediction* (ALP) is employed in an attempt to separate the deterministic $y(n) \approx \hat{s}(n)$ and stochastic part $e(n) \approx \hat{r}(n)$ assuming that the noise and interference signal has a broadband spectra. ALP corresponds to single-input and single-ouput (SISO) *predictor* application (class b, Fig. 2) with a single microphone, $P = 1$.

Most signals, such as speech and music, are partially predictable and partially random. The random input models the unpredictable part of the signal, whereas the filter models the predictable structure of the signal. The aim of linear prediction is to model the mechanism that introduces the correlation in a signal (Vaseghi, 1996). The solution to this system corresponds to a Wiener solution (21) with the cross-correlation vector, $\mathbf{r}$, slighty modified. The delay z^{-D} in the ALP filter should be selected in such a way that $d(n) = x(n) + r(n)$ and $d(n-D)$ are still correlated. If D is too long, the correlation in $d(n)$ and $d(n-D)$ is weak and unpredictable for the ALP filter; for that reason it cannot be canceled suitably. If D is

too short, the deterministic part of signal in $d(n)$ and $d(n-D)$ remains correlated after D; for that reason it can be predicted and cancelled by the ALP filter. $D=1$ causes that the voice in $d(n)$ and $d(n-D)$ is strongly correlated. A cascade of ALP filters of lower order independently adapted improves the modeling of the general ALP filter. In this case, the prediction is performed in successive refinements, the adaptation steps μ can be greater, and thus each stage is less affected by the disparity of *eigenvalues* which results in a faster convergence.

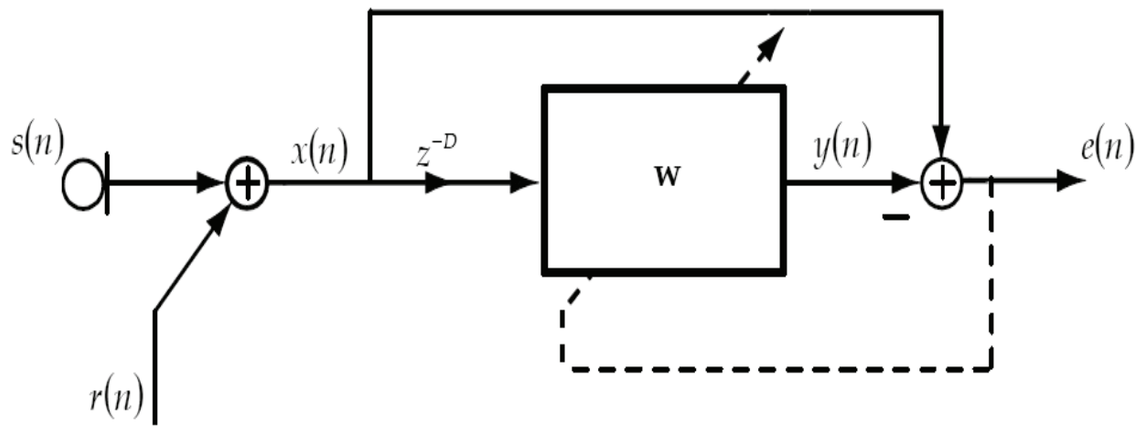

Fig. 9. Adaptive linear predictor.

4.3 Noise Cancellation

The *adaptive noise cancellation* (ANC) cancels the primary unwanted noise $r(n)$ by introducing a canceling antinoise of equal amplitude but opposite phase using a reference signal. This reference signal is derived from one or more sensors located at points near the noise and interference sources where the interest signal is weak or undetectable.

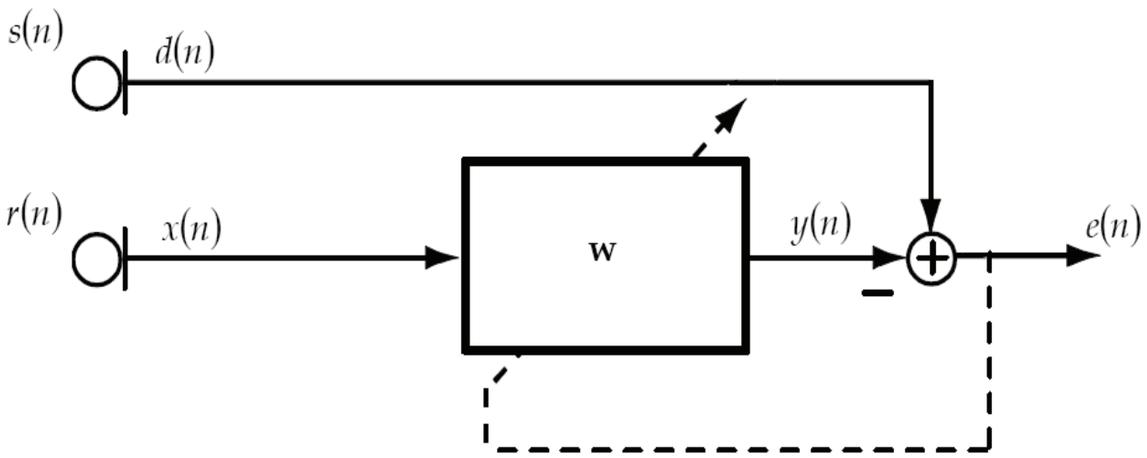

Fig. 10. Adaptive noise cancellation.

A typical ANC configuration is depicted in Fig. 10. Two microphones are used, $P=2$. The

primary input $d(n)=s(n)+r(n)$ collects the sum of unwanted noise $r(n)$ and speech signal $s(n)$, and the auxiliary or reference input measures the noise signal $x(n)=r(n)$. ANC corresponds to multiple-input and single-output (MISO) *joint-process estimator* application (class c, Fig. 2) with at least two microphones, $P=2$.

4.4 Beamforming

Beamforming is a multiple-input and single-output (MISO) application and consists of multichannel advanced multidimensional (space-time domain) filtering techniques that enhance the desired signal as well as suppress the noise signal.

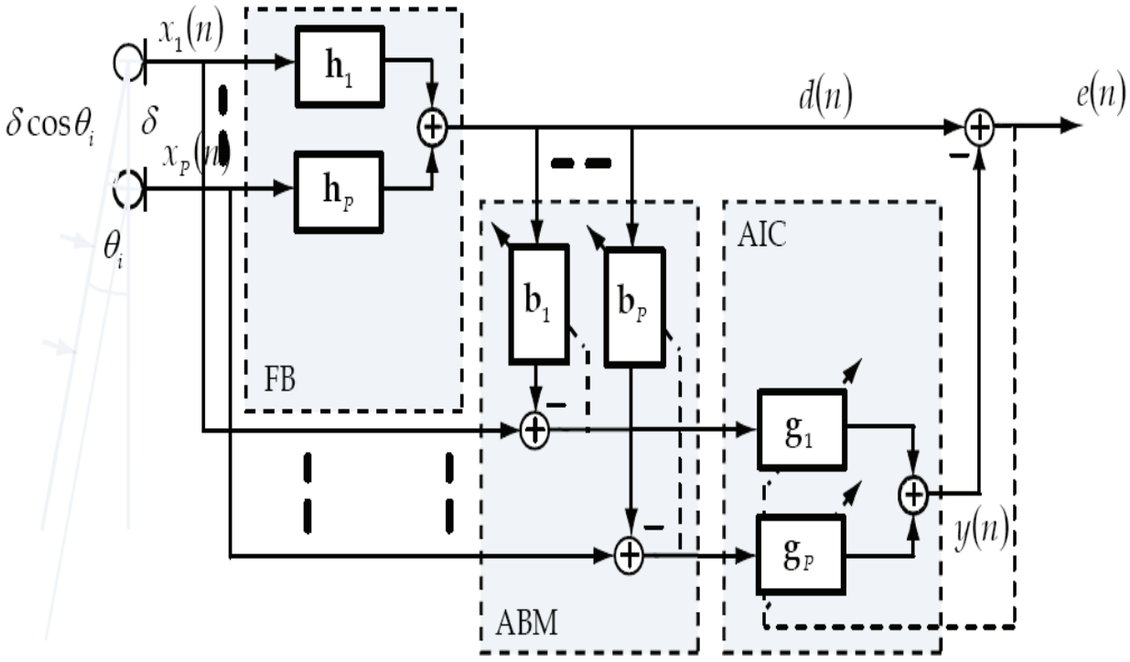

Fig. 11. Adaptive beamforming. Robust generalized sidelobe canceller (RGSC). Fixed beamforming (FB) allow conforming determined directivity pattern. The adaptive block matrix (ABM) or *blocking matrix*, with coefficient-constrained adaptive filters, prevents the target signal from leaking into the adaptive interference canceller (AIC). The AIC uses norm-constrained adaptive filters that can further improve the robustness against target signal cancellation.

In beamforming, two or more microphones are arranged in an array of some geometric shape. A *beamformer* is then used to filter the sensor outputs and amplifies or attenuates the signals depending on their *direction of arrival* (DOA), θ. The spatial response, or *beampattern*, of a beamformer generally features a combination of mainlobes that may be aimed at the target sources, and smaller sidelobes and null points aimed at the interference sources. Beampatterns are generally frequency-dependent, unless the beamformer is specifically designed to be frequency independent.

The sound sources are assumed to be in the *far-field* of the microphone array, i.e. the distance of the source from the array is much greater than the distance between the microphones (the spherical wavefronts emanating from the sources can be approximated as plane wavefronts). Each source $s_i(n)$ arrives to microphone 1 with delay $\kappa_i = \delta \cos\theta_i / v$ relative to its arrival to 2 because it has to travel an extra distance $\delta \cos\theta_i$; θ_i is the DOA of $s_i(n)$ and $v \approx 355\text{ms}^{-1}$ is a velocity of sound. $\delta \le v/F_s$ represents the spatial sampling interval of the wavefield; it has to fulfill this inequality to avoid spatial aliasing. The generalized sidelobe canceller (GSC) is an adaptive beamformer that keeps track of the characteristics of the interfering signal, leading to a high interference rejection performance. Initially, the P microphone inputs $x_p(n)$, $p = 1 \ldots P$, go through the FB that directs the beam towards the expected DOA. The beamformer output $y(n) = \langle \mathbf{h}, \mathbf{x}(n) \rangle$ contains the enhanced signal originating from the pointed direction, which is used as a reference by the ABM. The coefficient vector $\mathbf{h}$ has to fulfill both spatial and temporal constrains $\mathbf{Ch} = \mathbf{c}$, $\mathbf{h} = \mathbf{C}\left[\mathbf{C}^H\mathbf{C}\right]^{-1}\mathbf{c}$. The ABM adaptively subtracts the signal of interest, represented by the reference signal $y(n)$, from each channel input $x_p(n)$, and provides the interference signals. The columns of $\mathbf{C}$ must be pairwise orthogonal to the columns of the blocking matrix $\mathbf{B}$, $\mathbf{C}^H\mathbf{B} = 0$. The quiescent vector $\mathbf{h}$ is a component independent of data and $\mathbf{w} = \mathbf{h} - \mathbf{Bg}$ is a filter that satisfies the linear constrains $\mathbf{C}^H\mathbf{w} = \mathbf{C}^H(\mathbf{h} - \mathbf{Bg}) = \mathbf{C}^H\mathbf{h} = \mathbf{c}$. The upper signal path in Fig. 11 has to be orthogonal to the lower signal path. In order to suppress only those signals that originate from a specific tracking region, the adaptive filter coefficients are constrained within predefined boundaries (Benesty & Huang, 2003). These boundaries are specified based on the maximum allowed deviation between the expected DOA and the actual DOA. The interference signals, obtained from the ABM, are passed to the AIC, which adaptively removes the signal components that are correlated to the interference signals from the beamformer output $y(n)$. The norm of the filter coefficients in the AIC is constrained to prevent them from growing excessively large. This minimizes undesirable target signal cancellation, when the target signal leaks into the AIC, further improving the robustness of the system (Yoon et al., 2007).

In noise reduction systems, the beamformer can be used to either reject the noise (interference) by attenuating signals from certain DOAs, or focus on the desired signal (target) by amplifying signals from the target DOA and attenuating all signals that are not from the target DOAs. For non real-time speech enhancement applications it is possible to select a set of DOAs to be tested. Therefore adaptive algorithms with directional constrains, like a RGSC, can be exploited to achieve better noise reduction performance.

4.5 Deconvolution

Both *blind signal separation* (BSS), also known as *blind source separation*, and *multichannel blind deconvolution* (MBD) problems are a type of inverse problems with similarities and subtle differences between them: in the MBD only one source is considered, and thus the system is single-input single-output (SISO), while in BSS there are always multiple independent sources and the mixing system is MIMO; the interest of MBD is to deconvolve the source from the AIRs, while the task of BSS is double: on the one hand the sources must be separated, on the other hand the sources must be deconvolved from the multiple AIRs since

each sensor collects a combination of every original source convolved by different filters (AIRs) according to (2) (Gkalelis, 2004).

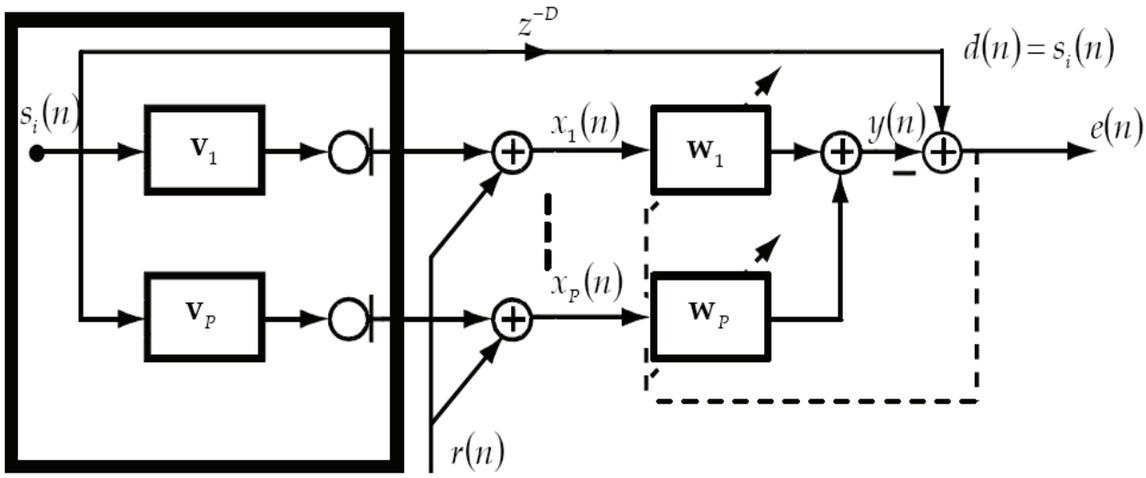

Fig. 12. Multichannel blind deconvolution.

In both cases, the *blind deconvolution* or *equalization* approach as well as the *blind separation* one, must estimate adaptively the inverse of the convolutive system that allows recovering the input signals and suppressing the noise. The goal is to adjust $\mathbf{W}$ so that $\mathbf{WV}=\mathbf{PD}$, where $\mathbf{P}$ is a permutation matrix and $\mathbf{D}$ is a diagonal matrix whose (p,p)th is $\alpha_p z^{-\kappa_p}$; α_p is a nonzero scalar weigthing, and κ_p is an integer delay.

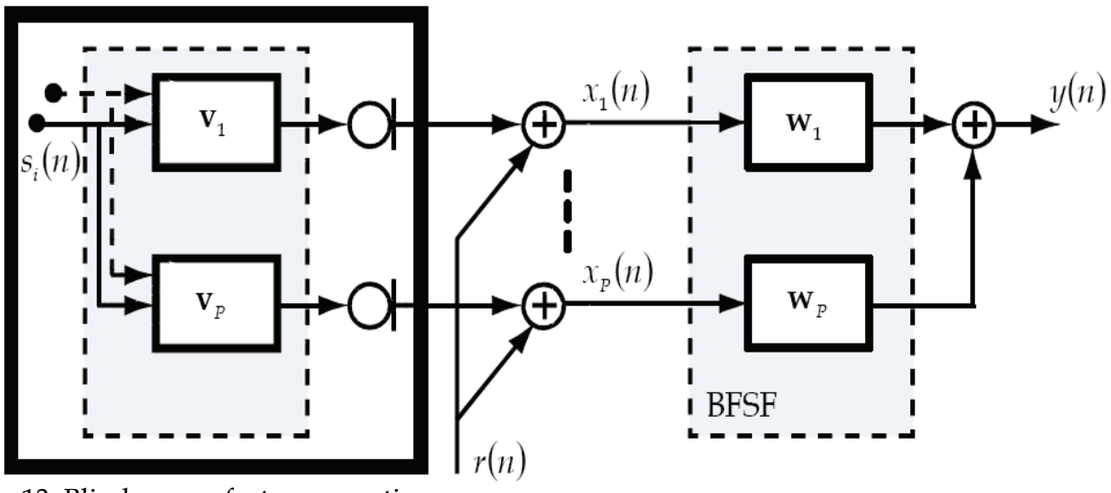

Fig. 13. Blind source factor separation.

BSS deals with the problem of separating I unknown sources by observing P microphone signals. In the *underdetermined* case ($P<I$) there are infinitely possible vectors $\mathbf{s}(n)$ that satisfy (3). There are mainly two ways to achieve the minimum norm solution. In the first,

the right generalized inverse of $\mathbf{V}$ is estimated and then applied to the set of microphone signals $\mathbf{x}(n)$. Another class of algorithms employ the sparseness of speech signal to design better inversion strategies and identify the minimum norm solution. Many techniques of convolutive BSS have been developed by extending methods originally designed for blind deconvolution of just one channel. A usual practice is to use *blind source factor separation* (BSFS) technique, where one source (factor) is separated from the mixtures, and combine it with a *deflationary* approach, where the sources are extracted one by one after deflating, i.e. removing, them from the mixed signals. The MIMO FIR filter $\mathbf{W}$ used for BSS becomes a multiple-input single-output (MISO) depicted in Fig. 13. The output $y(n)$ corresponds to (8) and the tap-stacked column vector containing all demixing filter weights defined in (7) is obtained as

$$\mathbf{u} = \mathbf{R}\mathbf{p} \tag{54}$$

$$\mathbf{w} = \frac{\mathbf{u}}{\sqrt{\mathbf{u}^H \mathbf{R}\mathbf{u}}}$$

where $\mathbf{R}$ is a block matrix where its blocks are the correlation matrices $\mathbf{R}_{pq}$ between the p-th channel and q-th channel defined in (22) and $\mathbf{p}$ is a block vector where its blocks are the cross-cumulant vector $\mathbf{p} = \text{cum}\{\mathbf{x}(\text{n}), y(n)\ldots y(n)\}$ (Gkalelis, 2004). The second step in (54) is just the normalization of the output signal $y(n)$. This is apparent left multiplying by $\mathbf{x}(\text{n})$.
The deflationary BSS algorithm for $i = 1\ldots I$ sources can be summarized as following: one source is extracted with the BSFS iterative scheme till convergence (54) and the filtering of the microphone signals with the estimated filters from the BSFS method (8) is performed; the contribution of the extracted source into the mixtures x_p, $p = 1\ldots P$, is estimated (with the LS criterion) and the contribution of the o-th source into i-th mixture is computed by using the estimated filter $\mathbf{b}$, $c(n) = \langle \mathbf{b}, \mathbf{y}(n) \rangle$ with $\mathbf{y}(n) = [y(n) \quad y(n-1) \quad \cdots \quad y(n-B+1)]$, $B << L$; deflate the contribution $c(n)$ from the p-th mixture, $x_p(n) = x_p(n) - c(n)$, $p = 1\ldots P$.
This method is very suitable for speech enhancement application where only one source should be extracted, i.e. speech.
It is possible to consider the deflationay BSFS (DBSFS) structure as a GSC. ABM exactly corresponds to the deflating filters of the deflationary approach. By comparing the different parts, i.e. the BSFS block and the fixed beamformer, it is concluded that it may be possible to construct similar algorithms to those of GSC.

5. Conclusion

This chapter is an advanced tutorial about multichannel adaptive filtering for speech enhancement. Different techniques have been examined in a common foundation. Several approaches of filtering techniques were presented as the number of channels increases.
The spectral equalization (power subtraction), in general, can achieve more noise reduction than an ANC and a beamformer method. However, it is based on the noise spectrum

estimator instead of the unknown noise spectra at each time, producing a distortion known as "musical noise" (because of the way it sounds). The performance of ANC depends on the coherence between the input noisy signal and the reference noise signal. Only if the coherence is very high the results are spectacular, therefore, this fact limits its application to particular cases. The amount of noise that can be canceled by a beamformer relies on the number of microphones in the array and on the SNR of the input signal. More microphones can lead to more noise reduction. However, the effectiveness of a beamformer in suppressing directional noise depends on the angular separation between signal and the noise source (Benesty & Huang, 2003). The ALP method is very simple because only second order statistics are required, but the estimation is only optimal if the residue is i.i.d. Gaussian (Solé-Casals et al., 2000).

All these techniques are narrowly connected. The linear prediction of $x(n)$ is nothing but the deconvolution of $x(n)$ (Solé-Casals et al., 2000). In (Taleb et al., 1999), the problem of Wiener system blind inversion using source separation methods is addressed. This approach can also be used for blind linear deconvolution. In (Gkalelis, 2004) the link between the deflationary approach (the extension of the single channel blind deconvolution algorithm) and the traditional GSC structure is showed. Several strategies between different approaches are also possible, i.e. in (Yi & Philipos, 2007), a Wiener filter, that uses linear prediction to estimate the signal spectrum, is presented.

The best filter to enhance a particular recording will be chosen based on experience and experimentation (Koenig et al., 2007). Nevertheless, the algorithm developer would find it useful to have a quality measure that helps to compare, in general terms, the performance of different implementations of a certain algorithm (Yi & Philipos, 2007). One substantial ingredient of this performance is the intelligibility attained after processing the recording, or even better the increase of intelligibility compared to the unprocessed sample. Therefore, one possible way to measure the performance of an enhancement algorithm, and probably the best, would be to use a panel of listeners and subjective tests. To attain significant results, different speech recordings with different types and degrees of noise and distortion should be used as inputs to the algorithm, and therefore the task would probably become unapproachable in terms of time and effort, setting aside the fact that the experiment would hardly be repeatable.

In order to properly monitor the performance of the algorithms, different types and degrees of degradations should be imposed to the test signal. The model used to deal with degradations can be as simple as an additive noise, for a mono version of the test signal corrupted by random noise or a second talker speech, or as complex as a virtual room simulator for early reflexions and a stocastic reverberation generator, for a detailed acoustic model of the recording room, where several noise sources can be placed in different places. Measured impulse responses of a real chamber is another option to obtain very realistic mono or multi-channel virtual recordings.

6. References

Armelloni, E.; Giottoli, C. & Farina, A. (2003) Implementation of Real-time Partitioned Convolution on a DSP Board, *2003 IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics*, pp. 71–74.

Benesty, J. & Huang, Y. (2003) *Adaptive Signal Processing (Applications to Real-World Problems)*. Springer, Berlin Heidelberg New York Hong Kong London Milan Paris Tokyo.

Boray, G.K. & Srinath, M.D. (1992) Conjugate Gradient Techniques for Adaptive Filtering. *IEEE Transactions on Circuits and Systems*, 39(1), 1-10.

Chau, E.Y-H. (2001) *Adaptive Noise Reduction Using A Cascaded Hybrid Neural Network*. MS Thesis, University of Guelph, Ontario.

Crochiere, R.E. & Rabiner, L.R. (1983) *Multirate Digital Signal Processing*. Prentice-Hall, London Sidney Toronto Mexico New Delhi Tokyo Singapore Rio de Janeiro.

Friedlander, B. (1982) Lattice Filters for Adaptive Processing. *Proceedings of the IEEE*, 70(8), 829–867.

García, L. (2006) *Cancelación de Ecos Multicanal*. PhD Thesis, Universidad Politécnica de Madrid, Spain.

Gay, S.L. & Benesty, J. (2000) *Acoustic Signal Processing for Telecommunication*. Kluwer Academic Publishers, Boston Dordrecht London.

Gkalelis, N. (2004) *Undetermined Blind Source Separation for Speech Signals*. MS Thesis, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany.

Glentis, G.-O.; Berberidis, K. & Theodoridis, S. (1999) Efficient least square adaptive algorithms for FIR transversal filtering: A unified view, *IEEE Signal Processing Magazine*, 16(4), 13–41.

Haykin, S. (2002) *Adaptive Filter Theory*. Prentice-Hall, Inc., New Jersey.

Honig, M.L. & Messerschmitt, D.G. (1984) *Adaptive Filters: Structures, Algorithms and Applications*. Kluwer Academic Publishers, Boston The Hague London Lancaster.

Koenig, B.E.; Lacey, D.S. & Killion, S.A. (2007) Forensic enhancement of digital audio recordings, *Journal of de Audio Engineering Society*, 55(5), 352–371.

Morgan, D.R. & Thi, J.C. (1995) A Delayless Subband Adaptive Filter Architecture, *IEEE Transactions on Signal Processing*, 43(8), 1819–1830.

Páez Borrallo, J.M. & Otero, M.G. (1992) On The Implementation of a Partitioned Block Frequency Domain Adaptive Filter (PBFDAF) For Long Acoustic Echo Cancellation, *Signal Processing*, 27(3), 301–315.

Reilly, J.P.; Wilbur, M.; Seibert, M. & Ahmadvand, N. (2002) The Complex Subband Decomposition and its Application to the Decimation of Large Adaptive Filtering Problems, *IEEE Transactions on Signal Processing*, 50(11), 2730–2743.

Shynk, J.J. (1992) Frequency-domain and Multirate Adaptive Filtering, *IEEE Signal Processing Magazine*, 9(1), 14–37.

Solé-Casals, J.; Jutten, C. & Taleb, A. (2000) Source Separation Techniques Applied to Linear Prediction, *2th International Workshop on Independent Component Analysis and Blind Source Separation Proceedings ICA2000*, pp. 193–198.

Taleb, A.; Solé-Casals, J. & Jutten, C. (1999) Blind Inversion of Wiener Systems, *IWANN 99*, pp. 655–664.

Vaseghi, S.V. (1996) *Advanced Signal Processing and Digital Noise Reduction*. John Willey & Sons Ltd. and B.G. Teubner, Chichester New York Brisbane Toronto Singapore Stuttgart Leipzig

Yi, H. & Philipos, C.L. (2007) A comparative intelligibility study of single microphone noise reduction algorithms, *The Journal of the Acoustical Society of America*, 122(3), 1777-1786.

Yoon. B-Y.; Tashev, I. & Acero, A. (2007) Robust Adaptive Beamforming Algorithm Using Instantaneous Direction Of Arrival With Enhanced Noise Suppression Capability. *IEEE International Conference on Acoustics, Speech and Signal Processing* 1:I-133–I-136.

New Developments in Robotics Automation and Control

Edited by Aleksandar Lazinica

ISBN 978-953-7619-20-6
Hard cover, 450 pages
Publisher InTech
Published online 01, October, 2008
Published in print edition October, 2008

This book represents the contributions of the top researchers in the field of robotics, automation and control and will serve as a valuable tool for professionals in these interdisciplinary fields. It consists of 25 chapter that introduce both basic research and advanced developments covering the topics such as kinematics, dynamic analysis, accuracy, optimization design, modelling , simulation and control. Without a doubt, the book covers a great deal of recent research, and as such it works as a valuable source for researchers interested in the involved subjects.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Lino Garcia and Soledad Torres-Guijarro (2008). Multichannel Speech Enhancement, New Developments in Robotics Automation and Control, Aleksandar Lazinica (Ed.), ISBN: 978-953-7619-20-6, InTech, Available from: http://www.intechopen.com/books/new_developments_in_robotics_automation_and_control/multichannel_speech_enhancement

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen