Online Kernel Canonical Correlation Analysis for Supervised Equalization of Wiener Systems

Steven Van Vaerenbergh, *Member, IEEE*, Javier Vía, *Member, IEEE*, and Ignacio Santamaría, *Member, IEEE*
Dept. of Communications Engineering, University of Cantabria, Spain
E-mail: {steven,jvia,nacho}@gtas.dicom.unican.es

***Abstract*— We consider the application of kernel canonical correlation analysis (K-CCA) to the supervised equalization of Wiener systems. Although a considerable amount of research has been carried out on identification/equalization of Wiener models, in this paper we show that K-CCA is a particularly suitable technique for the inversion of these nonlinear dynamic systems. Another contribution of this paper is the development of an online K-CCA algorithm which combines a sliding-window approach with a recently proposed reformulation of CCA as an iterative regression problem. This online algorithm permits fast equalization of time-varying Wiener systems. Simulation examples are added to illustrate the performance of the proposed method.**

I. Introduction

Wiener systems are well-known nonlinear models consisting of a possibly time-varying linear filter followed by a memoryless nonlinearity. These structures have been successfully applied in a number of areas such as chemical and biological modeling, signal processing and communications. A considerable amount of research has been carried out in the last decades on identifying and/or inverting Wiener systems. These techniques include neural network models, orthonormal functions, higher-order input-output cross-correlations and many others (see, for instance, [1], [2], [3], [4], [5]). However, most of these techniques do not exploit the specific cascade structure of Wiener systems in the identification/equalization procedure: they are black-box models. A recent exception to this rule is the work by Aschbacher and Rupp [6], which jointly identifies the inverse nonlinearity and the linear filter of the Wiener model. Inspired by this work, in this paper we propose to use kernel canonical correlation analysis (K-CCA) as a suitable technique to exploit the structure of Wiener systems in supervised identification/equalization algorithms.

Canonical correlation analysis (CCA) is a well-known technique in multivariate statistical analysis, which has been widely used in economics, meteorology and in many modern information processing fields, such as communication theory, statistical signal processing and blind source separation (BSS). The concept of CCA was first introduced by H. Hotelling [7] as a way of measuring the linear relationship between two multidimensional sets of variables and was later extended to several data sets [8].

Several extensions of CCA to account for nonlinear relationships between two data sets have recently been proposed [9], [10]. Among them, one of the most promising approaches is the application of K-CCA [11], [12]. K-CCA searches for *nonlinear* relationships between data sets, by first applying a nonlinear kernel transformation that maps the input data onto a high dimensional feature space (usually infinitely dimensional) and then performing conventional (linear) CCA. In the proposed approach for supervised equalization of Wiener systems, given a set of input-output patterns, K-CCA tries to maximize the correlation between the linearly transformed input and the nonlinearly transformed output (i.e. we use a linear kernel for the input data and a nonlinear kernel for the output). In this way, K-CCA efficiently exploits the cascade structure of Wiener systems and provides estimates of the linear filter and the inverse nonlinearity.

In their original forms, most of the kernel algorithms cannot be used to operate online because of a number of difficulties such as the growing dimensionality of the kernel matrix and the need to avoid overfitting of the problem. Recently, a kernel RLS algorithm was proposed that deals with both difficulties [13]: by applying a sparsification procedure the kernel matrix size was limited and the order of the problem was reduced. In [14] we presented a different kernel RLS approach in which conventional regularization was used to avoid overfitting and a sliding-window procedure fixed the kernel matrix size, allowing the algorithm to operate online in time-varying environments. In this paper we extend the basic sliding-window kernel RLS algorithm to K-CCA. In addition, we also extend to K-CCA a recent reformulation of CCA as a pair of coupled iterative regression problems [15], [16], which allows us to avoid any generalized eigenvalue decomposition in the computation of the K-CCA solution for each window.

The rest of this paper is organized as follows: in Section II CCA and K-CCA are briefly reviewed. In Section III the proposed procedure to identify/equalize Wiener systems based on K-CCA is described. In Section IV the online version of this algorithm is derived, followed by simulation results and comparisons in Section V. Finally, Section VI summarizes the main conclusions of this work.

II. CCA and Kernel CCA

A. Canonical Correlation Analysis

Given two full-rank data matrices $\mathbf{X}_1 \in \mathbb{R}^{N\times m_1}$ and $\mathbf{X}_2 \in \mathbb{R}^{N\times m_2}$, canonical correlation analysis (CCA) is defined as the problem of finding two canonical vectors $\mathbf{h}_1 \in \mathbb{R}^{m_1\times 1}$ and $\mathbf{h}_2 \in \mathbb{R}^{m_2\times 1}$ that maximize the correlation between the canonical variates $\mathbf{y}_1 = \mathbf{X}_1\mathbf{h}_1$ and $\mathbf{y}_2 = \mathbf{X}_2\mathbf{h}_2$, i.e.

$$\underset{\mathbf{h}_1,\mathbf{h}_2}{\operatorname{argmax}}\, \rho = \frac{\mathbf{y}_1^T\mathbf{y}_2}{\|\mathbf{y}_1\|\|\mathbf{y}_2\|} = \frac{\mathbf{h}_1^T\mathbf{R}_{12}\mathbf{h}_2}{\sqrt{\mathbf{h}_1^T\mathbf{R}_{11}\mathbf{h}_1\mathbf{h}_2^T\mathbf{R}_{22}\mathbf{h}_2}}, \qquad (1)$$

or equivalently

$$\underset{\mathbf{h}_1,\mathbf{h}_2}{\operatorname{argmax}}\, \rho = \mathbf{y}_1^T\mathbf{y}_2 \quad \text{s.t.} \quad \|\mathbf{y}_1\| = \|\mathbf{y}_2\| = 1,$$

where $\mathbf{R}_{kl} = \mathbf{X}_1^T\mathbf{X}_2$ is an estimate of the cross-correlation matrix. An alternative formulation of CCA into the framework of least squares (LS) regression has been proposed in [16], [17]. Specifically, it has been proved that CCA can be reformulated as the problem of minimizing

$$J = \frac{1}{2}\|\mathbf{X}_1\mathbf{h}_1 - \mathbf{X}_2\mathbf{h}_2\|^2 \quad \text{s.t.} \quad \frac{1}{2}\sum_{k=1}^{2}\|\mathbf{X}_k\mathbf{h}_k\|^2 = 1, \qquad (2)$$

and solving (1) or (2) by the method of Lagrange multipliers, CCA can be rewritten as the following generalized eigenvalue (GEV) problem

$$\frac{1}{2}\begin{bmatrix}\mathbf{X}_1^T\mathbf{X}_1 & \mathbf{X}_1^T\mathbf{X}_2\\ \mathbf{X}_2^T\mathbf{X}_1 & \mathbf{X}_2^T\mathbf{X}_2\end{bmatrix}\mathbf{h} = \beta\begin{bmatrix}\mathbf{X}_1^T\mathbf{X}_1 & \mathbf{0}\\ \mathbf{0} & \mathbf{X}_2^T\mathbf{X}_2\end{bmatrix}\mathbf{h}, \qquad (3)$$

where $\mathbf{h} = [\mathbf{h}_1^T\mathbf{h}_2^T]^T$, and $\beta = (\rho+1)/2$ is a parameter related to a principal component analysis (PCA) interpretation of CCA [17].

The solution of (3) can be directly obtained applying standard GEV algorithms. However, the special structure of the CCA problem has been recently exploited to obtain efficient CCA algorithms [16], [17], [18]. Specifically, denoting the pseudoinverse of $\mathbf{X}_k$ as $\mathbf{X}_k^+ = (\mathbf{X}_k^H\mathbf{X}_k)^{-1}\mathbf{X}_k^H$, the GEV problem (3) can be viewed as two coupled LS regression problems

$$\beta\mathbf{h}_k = \mathbf{X}_k^+\mathbf{y}, \quad k = 1, 2,$$

where $\mathbf{y} = (\mathbf{y}_1 + \mathbf{y}_2)/2$. This idea has been used in [16], [17] to develop an algorithm based on the solution of these regression problems iteratively: at each iteration t two LS regression problems are solved using

$$\mathbf{y}(t) = \frac{\mathbf{y}_1(t) + \mathbf{y}_2(t)}{2} = \frac{\mathbf{X}_1\mathbf{h}_1(t-1) + \mathbf{X}_2\mathbf{h}_2(t-1)}{2}$$

as desired output. Furthermore, this LS regression framework has been exploited to develop adaptive CCA algorithms based on the recursive least-squares algorithm (RLS) [16], [17].

B. Kernel Canonical Correlation Analysis

Although CCA constitutes a good technique to find linear relationships between two or several [17] data sets, it is not able to extract nonlinear relationships. In order to solve this problem, CCA has been extended to nonlinear CCA and kernel-CCA (K-CCA) [11], [12], [19]. Specifically, kernel CCA exploits the characteristics of kernel methods, consisting in the implicit nonlinear transformation of the data $\mathbf{x}_i$ from the input space to a high dimensional feature space $\tilde{\mathbf{x}}_i = \Phi(\mathbf{x}_i)$. Then, solving CCA in the feature space we are able to extract nonlinear relationships.

The key property of the kernel methods and reproducing kernel Hilbert spaces (RKHS) is that, since the scalar products in the feature space can be seen as nonlinear (kernel) functions of the data in the input space, the explicit mapping to the feature space can be avoided, and any linear technique can be performed in the feature space by solely replacing the scalar products by the kernel function in the input space. In this way, any positive definite kernel function satisfying Mercer's condition [20]: $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \langle\Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j)\rangle$ has an implicit mapping to some higher-dimensional feature space. This simple and elegant idea is known as the "kernel trick", and it is commonly applied by using a nonlinear kernel such as the Gaussian kernel

$$\kappa(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{y}\|^2}{2\sigma^2}\right),$$

which implies an infinity-dimensional feature space.

After transforming the data and canonical vectors to feature space, the GEV problem (3) can be written as

$$\frac{1}{2}\begin{bmatrix}\tilde{\mathbf{X}}_1^T\tilde{\mathbf{X}}_1 & \tilde{\mathbf{X}}_1^T\tilde{\mathbf{X}}_2\\ \tilde{\mathbf{X}}_2^T\tilde{\mathbf{X}}_1 & \tilde{\mathbf{X}}_2^T\tilde{\mathbf{X}}_2\end{bmatrix}\tilde{\mathbf{h}} = \beta\begin{bmatrix}\tilde{\mathbf{X}}_1^T\tilde{\mathbf{X}}_1 & \mathbf{0}\\ \mathbf{0} & \tilde{\mathbf{X}}_2^T\tilde{\mathbf{X}}_2\end{bmatrix}\tilde{\mathbf{h}}, \qquad (4)$$

where $\tilde{\mathbf{X}}_k \in \mathbb{R}^{N\times m_k'}$ represents the transformed data and $\tilde{\mathbf{h}} = [\tilde{\mathbf{h}}_1^T\tilde{\mathbf{h}}_2^T]^T$ contains the transformed canonical vectors of length m_1' and m_2'. Taking into account that the canonical vectors $\tilde{\mathbf{h}}_1$ and $\tilde{\mathbf{h}}_2$ belong to the subspace defined by the rows of $\tilde{\mathbf{X}}_1$ and $\tilde{\mathbf{X}}_2$ respectively we can find two vectors $\boldsymbol{\alpha}_k \in \mathbb{R}^{N\times 1}$ such that

$$\tilde{\mathbf{h}}_k = \tilde{\mathbf{X}}_k^T\boldsymbol{\alpha}_k \quad k = 1, 2,$$

and left-multiplying (4) by

$$\begin{bmatrix}\tilde{\mathbf{X}}_1 & \mathbf{0}\\ \mathbf{0} & \tilde{\mathbf{X}}_2\end{bmatrix}$$

we obtain

$$\frac{1}{2}\begin{bmatrix}\mathbf{K}_1^2 & \mathbf{K}_1\mathbf{K}_2\\ \mathbf{K}_2\mathbf{K}_1 & \mathbf{K}_2^2\end{bmatrix}\boldsymbol{\alpha} = \beta\begin{bmatrix}\mathbf{K}_1^2 & \mathbf{0}\\ \mathbf{0} & \mathbf{K}_2^2\end{bmatrix}\boldsymbol{\alpha}, \qquad (5)$$

where $\mathbf{K}_k = \tilde{\mathbf{X}}_k\tilde{\mathbf{X}}_k^T$ is the kernel matrix with elements

$$\mathbf{K}_k(i, j) = \kappa_k(\mathbf{x}_k[i], \mathbf{x}_k[j]),$$

in its i-th row and j-th column, $\kappa_k(\cdot,\cdot)$ is the kernel applied to the k-th data set, and $\mathbf{x}_k^T[i]$ denotes the i-th row of the k-th data matrix. Finally, (5) can be simplified to

$$\frac{1}{2}\begin{bmatrix}\mathbf{K}_1 & \mathbf{K}_2\\ \mathbf{K}_1 & \mathbf{K}_2\end{bmatrix}\boldsymbol{\alpha} = \beta\begin{bmatrix}\mathbf{K}_1 & \mathbf{0}\\ \mathbf{0} & \mathbf{K}_2\end{bmatrix}\boldsymbol{\alpha}, \qquad (6)$$

and the resulting GEV constitutes again two coupled kernel-LS regression problems defined now as

$$J_k(\beta\boldsymbol{\alpha}_k) = \|\mathbf{K}_k\beta\boldsymbol{\alpha}_k - \mathbf{y}\|^2 \quad k = 1, 2,$$

where $\mathbf{y}_k = \mathbf{K}_k\boldsymbol{\alpha}_k$ and $\mathbf{y} = (\mathbf{y}_1 + \mathbf{y}_2)/2$ is the desired output.

To summarize, the application of the kernel trick permits the solution of the CCA problem in the feature space without increasing the computational cost and conserving the LS regression framework.

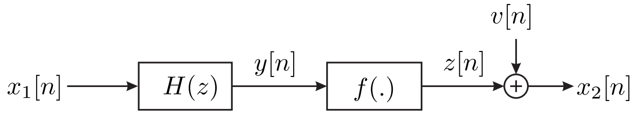

Fig. 1. A nonlinear Wiener system.

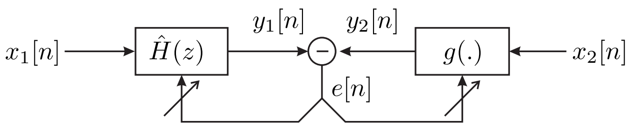

Fig. 2. The used Wiener system identification diagram.

C. Measures Against Overfitting

For most useful kernel functions, the dimension of the feature space, m_k', will be much higher than the number of available data points, N. In such cases the kernel matrices $\mathbf{K}_k$ do not have full rank and Eq. (6) could have an infinite number of solutions, representing an overfit problem. Various techniques to handle this overfitting have been presented.

1) Order Reduction: One way of dealing with overfitting is by reducing the order of the transformed data $\tilde{\mathbf{X}}_k$ through PCA or a similar technique [11], [12], [13], in which the data $\tilde{\mathbf{X}}_k \in \mathbb{R}^{N\times m_1'}$ are to be transformed to $\bar{\mathbf{X}}_k \in \mathbb{R}^{N\times m_k''}$ with $m_k'' < N$ by considering its m_k'' principal components.

2) Regularization: A second method considers regularizing the solution by adding a normalization restriction to the norm of $\tilde{\mathbf{h}}_k$ [11], [12] or $\boldsymbol{\alpha}_k$ [21]. Here, we apply the classical regularization technique in the framework of LS regression, which yields the following CCA-GEV problem

$$\frac{1}{2}\begin{bmatrix}\mathbf{X}_1^T\mathbf{X}_1 & \mathbf{X}_1^T\mathbf{X}_2\\ \mathbf{X}_2^T\mathbf{X}_1 & \mathbf{X}_2^T\mathbf{X}_2\end{bmatrix}\tilde{\mathbf{h}} = \beta\begin{bmatrix}\mathbf{X}_1^T\mathbf{X}_1 + c\mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{X}_2^T\mathbf{X}_2 + c\mathbf{I}\end{bmatrix}\tilde{\mathbf{h}},$$

or, in the case of K-CCA

$$\frac{1}{2}\begin{bmatrix}\mathbf{K}_1 & \mathbf{K}_2\\ \mathbf{K}_1 & \mathbf{K}_2\end{bmatrix}\boldsymbol{\alpha} = \beta\begin{bmatrix}\mathbf{K}_1 + c\mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{K}_2 + c\mathbf{I}\end{bmatrix}\boldsymbol{\alpha}. \qquad (7)$$

Therefore, the regularized version of CCA (or K-CCA) can be reformulated as two coupled LS (or kernel-LS) regression problems.

III. Kernel CCA for Wiener Systems

A Wiener system is a well-known and simple nonlinear system which consists of a cascade of a linear dynamic system and a memoryless nonlinearity (see Fig. 1). Such a nonlinear channel is encountered frequently, e.g. in digital satellite communications [22] or in digital magnetic recording [23]. Traditionally the problem of nonlinear equalization or identification has been tackled by considering nonlinear structures such as multilayer perceptrons (MLPs) [24], recurrent neural networks [25] or piecewise linear networks [26]. Most of the proposed techniques treated the Wiener system as a black-box, although use can be made of its simple structure.

Recently a supervised identification setup for Wiener systems was presented [6] that exploits the cascade structure by introducing joint identification of the linear filter and the inverse nonlinearity, as in Fig. 2. The estimator models for linear filter and nonlinearity are adjusted by minimizing the error between their outputs $y_1[n]$ and $y_2[n]$. By doing so, $\hat{H}(z)$ will represent an estimator of $H(z)$, while $g(.)$ will represent $f^{-1}(.)$ in the noiseless case, assuming that $f(.)$ is invertible in the output data range.

To avoid the trivial zero solution or divergence of the estimators, a restriction should be taken into account for this approach to work. The two most obvious options are imposing restrictions on

1) the norm of the estimator coefficients
2) the norm of the signals $y_1[n]$ and $y_2[n]$.

The first type of restriction was used in [6]. Interestingly, the second type is a direct application of (kernel) CCA, as can be seen from Eq. (2). K-CCA can be applied to this problem by maximizing the correlation between the linear projection $y_1[n]$ (linear kernel, $\kappa(\mathbf{x},\mathbf{y}) = \mathbf{x}^T\mathbf{y}$) of the system input $x_1[n]$ and the nonlinear projection $y_2[n]$ (gaussian or other nonlinear kernel) of the system output $x_2[n]$. To prevent overfitting, the nonlinear kernel matrix is regularized as mentioned in Section II-C. However, regularization is not needed for the linear kernel, since the feature space of a linear kernel is the original data space, with dimension $m_k < N$. Using this insight the linear kernel matrix can be replaced by an estimate of the correlation matrix, as shown below. Starting from the general form of the GEV problem (7)

$$\frac{1}{2}\begin{bmatrix}\mathbf{K}_1 & \mathbf{K}_2\\ \mathbf{K}_1 & \mathbf{K}_2\end{bmatrix}\begin{bmatrix}\boldsymbol{\alpha}_1\\ \boldsymbol{\alpha}_2\end{bmatrix} = \beta\begin{bmatrix}\mathbf{K}_1 & \mathbf{0}\\ \mathbf{0} & \mathbf{K}_2 + c\mathbf{I}\end{bmatrix}\begin{bmatrix}\boldsymbol{\alpha}_1\\ \boldsymbol{\alpha}_2\end{bmatrix}, \qquad (8)$$

and substituting $\mathbf{K}_1 = \mathbf{X}_1\mathbf{X}_1^T$ and $\mathbf{w}_1 = \mathbf{X}_1^T\boldsymbol{\alpha}_1$, (8) is equivalent to solving

$$\frac{1}{2}\begin{bmatrix}\mathbf{X}_1^T\mathbf{X}_1 & \mathbf{X}_1^T\mathbf{K}_2\\ \mathbf{K}_2\mathbf{X}_1 & \mathbf{K}_2^2\end{bmatrix}\begin{bmatrix}\mathbf{w}_1\\ \boldsymbol{\alpha}_2\end{bmatrix} = \beta\begin{bmatrix}\mathbf{X}_1^T\mathbf{X}_1 & \mathbf{0}\\ \mathbf{0} & \mathbf{K}_2(\mathbf{K}_2 + c\mathbf{I})\end{bmatrix}\begin{bmatrix}\mathbf{w}_1\\ \boldsymbol{\alpha}_2\end{bmatrix}$$

where $\mathbf{w}_1$ represents the solution to the primal problem, for the linear part, and $\boldsymbol{\alpha}_2$ represents the solution to the dual problem, for the nonlinear part. Substituting $\mathbf{y} = \frac{1}{2}(\mathbf{y}_1 + \mathbf{y}_2) = \frac{1}{2}(\mathbf{X}_1\mathbf{w}_1 + \mathbf{K}_2\boldsymbol{\alpha}_2)$, this GEV constitutes two coupled LS regression problems:

$$\left\{\begin{array}{l}\beta\mathbf{w}_1 = (\mathbf{X}_1^T\mathbf{X}_1)^{-1}\mathbf{X}_1^T\mathbf{y}\\ \beta\boldsymbol{\alpha}_2 = (\mathbf{K}_2 + c\mathbf{I})^{-1}\mathbf{y}\end{array}\right. \qquad (9)$$

IV. Online Algorithm

A. System Identification

An important characteristic of the Wiener system is that its linear filter is usually time-varying. Whereas the identification method of Section III will reliably identify the different blocks of a static system as a batch method, an online approach is required to track the time-variations of a Wiener system.

An online prediction setup assumes we are given a stream of input-output pairs $\{(\mathbf{x}_1[n], x_2[n]), (\mathbf{x}_1[n-1], x_2[n-1]), \ldots\}$, in which every $\mathbf{x}_1[n] = (x_1[n], x_1[n-$

Algorithm 1 The sliding-window K-CCA algorithm.

```
Initialize $\mathbf{K}^{(0)} = \mathbf{I}$ and $\mathbf{K}_{reg}^{(0)} = (1+c)\mathbf{I}$.
Initialize $\mathbf{w}_1^{(0)}$ and $\boldsymbol{\alpha}_2^{(0)}$ randomly.
for $n = 1, 2, \dots$ do
    Obtain $\mathbf{K}_{reg}^{(n)}$ from the data $\mathbf{x}_2^{(n)}$ as in (10).
    Calculate $\begin{cases} \mathbf{y}_1^{(n)} = \mathbf{X}_1^{(n)} \mathbf{w}_1^{(n-1)} \\ \mathbf{y}_2^{(n)} = \tilde{\mathbf{x}}_2^{(n)} (\tilde{\mathbf{x}}_2^{(n-1)})^T \boldsymbol{\alpha}_2^{(n-1)}. \end{cases}$
    Calculate $\mathbf{y}^{(n)} = \frac{1}{2}(\mathbf{y}_1^{(n)} + \mathbf{y}_2^{(n)})$.
    Calculate $(\mathbf{K}_{reg}^{(n)})^{-1}$ according to (11).
    Obtain the updated solutions $\mathbf{w}_1^{(n)}$ and $\boldsymbol{\alpha}_2^{(n)}$ as in (9).
end for
```

$1], \dots, x_1[n-L+1])^T$ is a vector representing a memory of the length of the linear filter $H(z)$. A key feature of online algorithms is that the number of computations must not increase as the number of samples increases. Since the size of a kernel matrix depends on the number of samples used to calculate it, we chose to take into account only a "sliding-window" containing the last N input-output pairs of this stream. For window n, the observation matrix $\mathbf{X}_1^{(n)} = (\mathbf{x}_1[n], \mathbf{x}_1[n-1], \dots, \mathbf{x}_1[n-N+1])^T$ and the observation vector $\mathbf{x}_2^{(n)} = (x_2[n], x_2[n-1], \dots, x_2[n-N+1])^T$ are formed and the corresponding kernel matrix $\mathbf{K}^{(n)} = \tilde{\mathbf{x}}_2^{(n)} (\tilde{\mathbf{x}}_2^{(n)})^T$ and regularized kernel matrix $\mathbf{K}_{reg}^{(n)} = \mathbf{K}^{(n)} + c\mathbf{I}$ can be calculated.

To solve (9), in each iteration the $N \times N$ inverse matrix $(\mathbf{K}_{reg}^{(n)})^{-1}$ must be calculated. This is costly both computationally and memory-wise (requiring $O(N^3)$ operations). Therefore in [14] an update algorithm was developed that can compute $(\mathbf{K}_{reg}^{(n)})^{-1}$ solely from knowledge of the data of the current observation vector $\mathbf{x}_2^{(n)}$ and the previous $(\mathbf{K}_{reg}^{(n-1)})^{-1}$.

Given the kernel matrix $\mathbf{K}_{reg}^{(n-1)}$, the new kernel matrix $\mathbf{K}_{reg}^{(n)}$ can be constructed by removing the first row and column of $\mathbf{K}_{reg}^{(n-1)}$, referred to as $\hat{\mathbf{K}}_{reg}^{(n-1)}$, and adding kernels of the new data as the last row and column:

$$\mathbf{K}_{reg}^{(n)} = \begin{bmatrix} \hat{\mathbf{K}}_{reg}^{(n-1)} & \mathbf{k}_{n-1}(\mathbf{x}_2^{(n)}) \\ \mathbf{k}_{n-1}(\mathbf{x}_2^{(n)})^T & k_{nn} + c \end{bmatrix} \qquad (10)$$

with $\mathbf{k}_{n-1}(\mathbf{x}_2^{(n)}) = [\kappa(\mathbf{x}_2^{(n-N+1)}, \mathbf{x}_2^{(n)}), \dots, \kappa(\mathbf{x}_2^{(n-1)}, \mathbf{x}_2^{(n)})]^T$ and $k_{nn} = \kappa(\mathbf{x}_2^{(n)}, \mathbf{x}_2^{(n)})$.

Calculating the inverse kernel matrix $(\mathbf{K}_{reg}^{(n)})^{-1}$ is done in two steps, using the two inversion formulas from the appendix at the end of this paper. Note that these formulas do not calculate the inverse matrices explicitly, but rather derive them from known matrices maintaining an overall time and memory complexity of $O(N^2)$ of the algorithm.

First, given $\mathbf{K}_{reg}^{(n-1)}$ and $(\mathbf{K}_{reg}^{(n-1)})^{-1}$, the inverse of the $N-1 \times N-1$ matrix $\hat{\mathbf{K}}_{reg}^{(n-1)}$ is calculated according to Eq. (12). Then $(\mathbf{K}_{reg}^{(n)})^{-1}$ can be calculated applying the matrix inversion formula from Eq. (11), based on the knowledge of $(\mathbf{K}_{reg}^{(n-1)})^{-1}$ and $\mathbf{K}_{reg}^{(n)}$.

The complete algorithm to solve (9) in an adaptive manner is summarized in Alg. (1).

B. *System Equalization*

While performing system identification of the Wiener system, an estimate is made of the inverse nonlinearity $g(.)$, which compensates for the nonlinearity $f(.)$. A linear equalizer $W(z)$ is proposed to compensate for $H(z)$, as shown in Fig. 3. A wide range of techniques are available to estimate this linear filter, among others the least mean squares algorithm (LMS), RLS, linear Wiener filter estimation, etc. We opted for the RLS algorithm with convergence speed in mind.

V. Simulation Results

A. *Static Wiener System*

Simulations are carried out to illustrate the performance of the proposed equalization algorithm. The performance is evaluated on a Wiener system consisting of the non-minimum phase linear filter $H(z) = 1 + 0.8668z^{-1} - 0.4764z^{-2} + 0.2070z^{-3}$ and the nonlinearity $f(x) = \tanh(x)$. The input signal is a white zero-mean Gaussian with unit variance. The output is affected by additive white Gaussian noise, matching an SNR of 25dB. Given $x_2[n]$, the desired output of the equalizer is a delayed version of the signal $x_1[n-d]$, with $d = 1/2(L + L_W)$, where L is the length of the linear filter $\hat{H}(z)$ and L_W is the length of the equalizer $W(z)$.

Equalization is performed by the proposed online K-CCA method, for which a Gaussian kernel with $\sigma = 0.2$ and the regularization constant $c = 0.1$ were used. The filter $W(z)$ has length $L_W = 15$ and the RLS forgetting factor is 0.99. For comparison, two other equalization methods are also included. The first one is the gradient identification method proposed in [6] to which we added the same RLS block for equalization as in the presented K-CCA method. The second one is a time-delay MLP with $L_W = 15$ inputs for its time-delay (i.e. equal to the equalizer's length), 15 neurons in its hidden layer and $\mu = 0.01$. The MLP *does not* take the system structure into account, and hence its equalization results are only included to see the advantages of the other two methods (that *do* exploit the Wiener system structure). All three methods were trained with a training data set in an adaptive manner, while at every iteration the equalizing capabilities of each method were tested using a separate test data set, generated by the same Wiener system. In Fig. 4 the mean square error (MSE) curves are compared for these three methods, averaged out over 50 Monte-Carlo simulations.

Fig. 5 shows the coefficients estimated by the K-CCA algorithm after processing 1000 samples online for the given example when $L = 10$ instead of the correct $L = 4$ coefficients for the linear filter. Fig. 6 compares the MSE curves obtained for different values of L when the correct value is $L = 4$. Note that the effect of overestimating L on the algorithm's performance is minimal.

A parameter that affects the performance of the K-CCA algorithm more is the length N of the sliding-window. Fig. 7 shows MSE curves for different window lengths, for the given setup. A longer window corresponds to a bigger kernel matrix, leading in turn to a better representation of the inverse

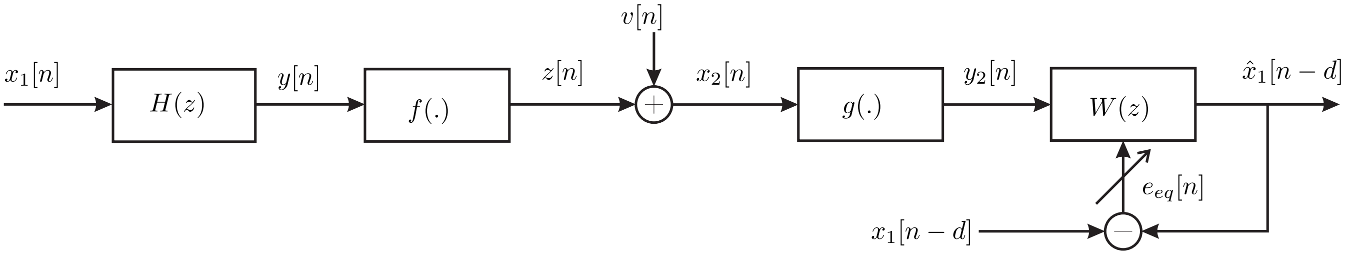

Fig. 3. Diagram for supervised equalization: Sliding-window K-CCA is applied on the input $x_1[n]$ and output $x_2[n]$ of the Wiener system. This estimates the nonlinear function $g(.)$ and its output $y_2[n]$. Using $y_2[n]$ and a time-delayed version of the system input $x_1[n-d]$, an equalizer $W(z)$ is estimated.

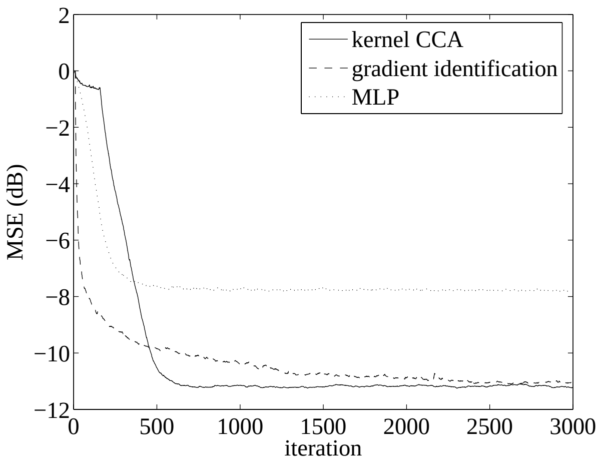

Fig. 4. Wiener system equalization MSE for the presented online K-CCA method, the gradient identification method from [6] and a time-delay MLP. The MLP does not make use of the system structure and hence achieves a worse result. The K-CCA method needs an initialization period of the length of its window ($L = 150$), after which the MSE drops fast and reaches convergence. The steepness of this slope is mainly determined by the speed of the RLS algorithm.

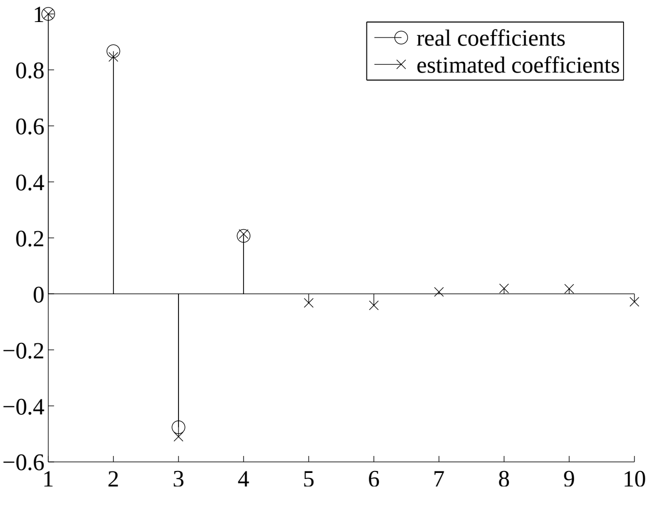

Fig. 5. The 4 real coefficients of the filter $H(z)$ versus the 10 coefficients of the estimated filter $\hat{H}(z)$. The 6 additionally estimated coefficients are very close to 0.

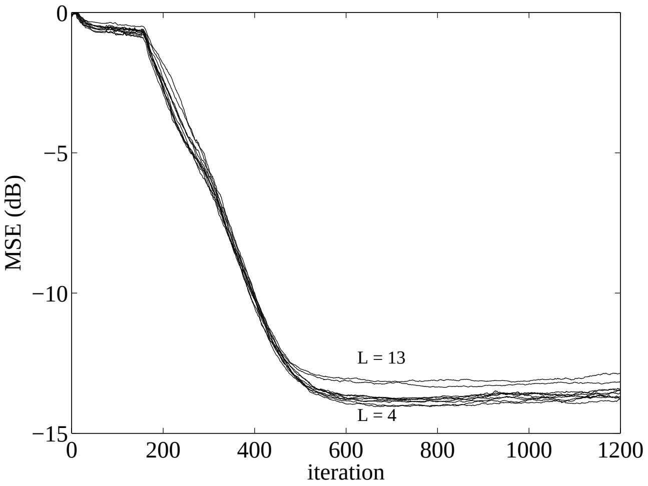

Fig. 6. MSE curves for equalization of a Wiener system with linear filter length 4, for different values of L, the length of the linear filter used in system identification. In the ideal case ($L = 4$) the filter length is known. The presented MSE curves for $L = 5$ till $L = 13$ show very similar equalizer performance for all cases. The curves were averaged out over 50 simulations.

nonlinearity $g(.)$ and hence a lower equalization error. The curves were averaged out over 50 simulations.

In a second setup, the same Wiener system is used with a BPSK input signal. After training the K-CCA algorithm online with 1000 symbols, its bit error rate (BER) was calculated on a test data set. The BER curve is shown in Fig. 8.

B. *Time-varying Wiener System*

A third setup is presented to test the tracking capability of the online K-CCA algorithm. The analyzed Wiener system has a minimum phase linear filter whose coefficients change linearly from $H(z) = 1 + 0.3551z^{-1} + 0.4587z^{-2} - 0.1708z^{-3}$ to $H(z) = 1 + 0.0563z^{-1} - 0.3677z^{-2} - 0.2046z^{-3}$ over 2000 input samples, and nonlinearity $f(x) = x + 0.1x^3$. The input signal is a white zero-mean Gaussian with unit variance and additive white Gaussian noise with zero-mean is added to the output, matching an SNR of 25dB. The online K-CCA algorithm is applied with $N = 150$. As an example we present the evolution of the third coefficient of $\hat{H}(z)$ compared to the third coefficient of $H(z)$ (see Fig. 9). After an initialization period of length N in which the

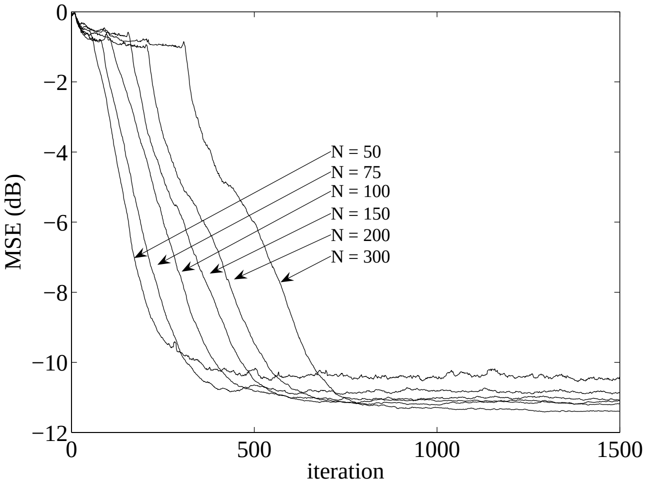

Fig. 7. Influence of the window length N on the MSE curves of the online K-CCA algorithm. Note the initialization period of length N, needed for replacing the initialization data in the kernel matrix by real data.

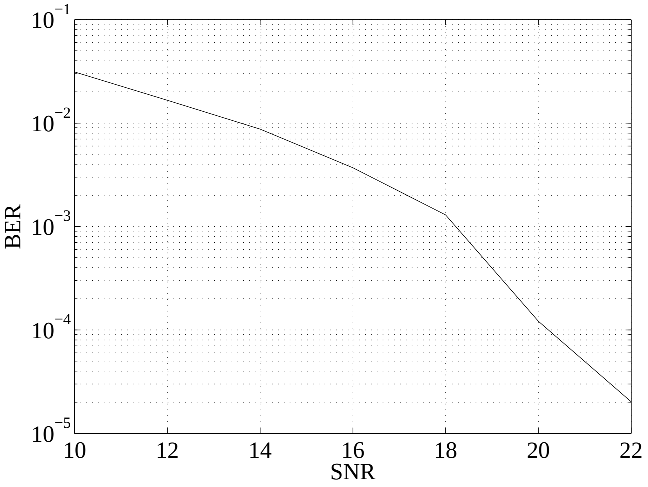

Fig. 8. BER curve for the online K-CCA algorithm using BPSK input symbols.

initialization data in the kernel matrix are replaced by real data, it can be observed that the algorithm is capable of functioning in a time-varying environment.

VI. Conclusions

We presented a novel K-CCA algorithm for the supervised equalization of nonlinear Wiener systems, exploiting the system structure. We also developed an online version of this algorithm, which combines a sliding-window approach with a reformulation of CCA as an iterative regression problem. Simulation examples show fast equalization of time-varying Wiener systems and results of the influence of the different algorithm parameters on its performance were presented. In particular, if the length of the Wiener system filter is not known and overestimated, the algorithm performance is hardly affected.

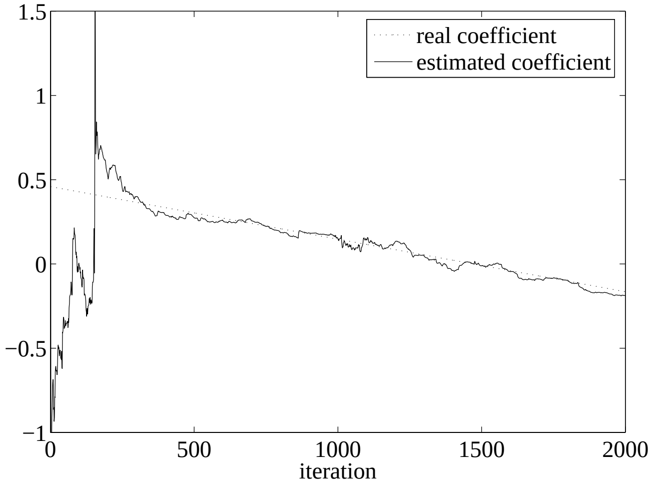

Fig. 9. Tracking capability of the online K-CCA algorithm. The dotted line represents a coefficient of the linear filter of a time-varying Wiener system. The straight line represents the estimated filter coefficient.

Appendix

Matrix inversion formulas

Adding a row and a colum: To a given non-singular matrix $\mathbf{A}$ a row and column are added as shown below, resulting in matrix $\mathbf{K}$. The inverse matrix $\mathbf{K}^{-1}$ can then be expressed in terms of the known elements and $\mathbf{A}^{-1}$ as follows:

$$\mathbf{K} = \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{b}^T & d \end{bmatrix}, \quad \mathbf{K}^{-1} = \begin{bmatrix} \mathbf{E} & \mathbf{f} \\ \mathbf{f}^T & g \end{bmatrix}$$

$$\Rightarrow \mathbf{K}^{-1} = \begin{bmatrix} \mathbf{A}^{-1}(\mathbf{I} + \mathbf{b}\mathbf{b}^T\mathbf{A}^{-1H}g) & -\mathbf{A}^{-1}\mathbf{b}g \\ -(\mathbf{A}^{-1}\mathbf{b})^T g & g \end{bmatrix} \tag{11}$$

with $g = (d - \mathbf{b}^T\mathbf{A}^{-1}\mathbf{b})^{-1}$.

Removing the first row and column: From a given non-singular matrix $\mathbf{K}$ a row and column are removed as shown below, resulting in matrix $\mathbf{D}$. The inverse matrix $\mathbf{D}^{-1}$ can then easily be expressed in terms of the known elements of $\mathbf{K}^{-1}$ as follows:

$$\mathbf{K} = \begin{bmatrix} a & \mathbf{b}^T \\ \mathbf{b} & \mathbf{D} \end{bmatrix}, \quad \mathbf{K}^{-1} = \begin{bmatrix} e & \mathbf{f}^T \\ \mathbf{f} & \mathbf{G} \end{bmatrix}$$

$$\Rightarrow \mathbf{D}^{-1} = \mathbf{G} - \mathbf{f}\mathbf{f}^T/e. \tag{12}$$

Acknowledgment

This work was supported by MEC (Ministerio de Educación y Ciencia) under grant TEC2004-06451-C05-02.

References

[1] S. Billings, “Identification of nonlinear systems: A survey,” *Proc. of IEE, Part D*, vol. 127, pp. 272–285, 1980.

[2] R. Haber and H. Unbehauen, “Structure identification of nonlinear dynamic systems: a survey on input-output approaches,” *Automatica*, vol. 26, no. 4, pp. 651–675, 1990.

[3] O. Nelles, *Nonlinear System Indetification*, Springer-Verlag, Berlin, Germany, 2000.

[4] I. W. Hunter and M. J. Korenberg, “The identification of nonlinear biological systems: Wiener and Hammerstein cascade models,” *Biol. Cybern.*, vol. 55, pp. 135–144, 1985.

[5] W. Greblicki, "Nonparametric identification of Wiener systems by orthogonal series," *IEEE Trans. on Automatic Control*, vol. 39, no. 10, pp. 2077–2086, 1994.

[6] E. Aschbacher and M. Rupp, "Robustness analysis of a gradient identification method for a nonlinear Wiener system," in *Proceedings of the 13th Statistical Signal Processing Workshop (SSP)*, Bordeaux, France, July 2005.

[7] H. Hotelling, "Relations between two sets of variates," *Biometrika*, vol. 28, pp. 321–377, 1936.

[8] R. Kettenring, "Canonical analysis of several sets of variables," *Biometrika*, vol. 58, pp. 433–451, 1971.

[9] P. L. Lai and C. Fyfe, "A neural network implementation of canonical correlation analysis," *Neural Networks*, vol. 12, no. 10, pp. 1391–1397, 1999.

[10] W.W. Hsieh, "Nonlinear canonical correlation analysis by neural networks," *Neural Networks*, vol. 13, no. 10, pp. 1095–1105, 2000.

[11] F. R. Bach and M. I. Jordan, "Kernel independent component analysis," *Journal of Machine Learning Research*, vol. 3, pp. 1–48, 2003.

[12] D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor, "Canonical correlation analysis; an overview with application to learning methods," *Neural Computation*, vol. 16, no. 12, pp. 2639–2664, Dec. 2004.

[13] Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least squares-algorithm," *IEEE Transactions on Signal Processing*, vol. 52, no. 8, Aug. 2004.

[14] S. Van Vaerenbergh, J. Vía, and I. Santamaría, "A sliding-window kernel RLS algorithm and its application to nonlinear channel identification," in *Proc. of the 2006 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Toulouse, France, May 2006.

[15] J. Vía, I. Santamaría, and J. Pérez, "Learning algorithms for adaptive canonical correlation analysis of several data sets," *Submitted to Neural Networks*, vol. 13, no. 10, pp. 1095–1105, 2006.

[16] J. Vía, I. Santamaría, and J. Pérez, "A robust RLS algorithm for adaptive canonical correlation analysis," in *Proc. of 2005 IEEE International Conference on Acoustics, Speech and Signal Processing*, Philadelphia, USA, March 2005.

[17] J. Vía, I. Santamaría, and J. Pérez, "Canonical correlation analysis (CCA) algorithms for multiple data sets: Application to blind SIMO equalization," in *Proc. of European Signal Processing Conference (EUSIPCO)*, Antalya, Turky, Sept. 2005.

[18] A. Pezeshki, L. L. Scharf, M. R. Azimi-Sadjadi, and Y. Hua, "Two-channel constrained least squares problems: solutions using power methods and connections with canonical coordinates," *IEEE Trans. Signal Processing*, vol. 53, pp. 121–135, Jan. 2005.

[19] P. L. Lai and C. Fyfe, "Kernel and nonlinear canonical correlation analysis," *International Journal of Neural Systems*, vol. 10(5), pp. 365–377, 2000.

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

[21] M. Kuss and T. Graepel, "The geometry of kernel canonical correlation analysis," Tech. Rep. TR-108, Max Planck Institute for Biological Cybernetics, Tübingen, Germany, 2003.

[22] G. Kechriotis, E. Zarvas, and E. S. Manolakos, "Using recurrent neural networks for adaptive communication channel equalization," *IEEE Trans. on Neural Networks*, vol. 5, pp. 267–278, Mar 1994.

[23] N. P. Sands and J. M. Cioffi, "Nonlinear channel models for digital magnetic recording," *IEEE Trans. Magn.*, vol. 29, pp. 3996–3998, Nov 1993.

[24] D. Erdogmus, D. Rende, J. C. Principe, and T. F. Wong, "Nonlinear channel equalization using multilayer perceptrons with information-theoric criterion," in *Proc. IEEE Workshop on Neural Networks and Signal Processing XI*, Falmouth, MA, Sept 2001, pp. 401–451.

[25] T. Adali and X. Liu, "Canonical piecewise linear network for nonlinear filtering and its application to blind equalization," *Signal Process.*, vol. 61, no. 2, pp. 145–155, Sept 1997.

[26] P. W. Holland and R. E. Welch, "Robust regression using iterative reweighted least squares," *Commun. Statist. Theory Methods*, vol. A. 6, no. 9, pp. 813–827, 1997.