Estimation of Cyclostationary Codebooks for Kernel Adaptive Filtering

S. García-Vega, A.M. Álvarez-Meza, and Germán Castellanos-Domínguez

Universidad Nacional de Colombia, Sede Manizales,
Signal Processing and Recognition Group
km 7 vía al Magdalena, Colombia
{segarciave,amalvarezme,cgcastellanosd}@unal.edu.co
http://portal.manizales.unal.edu.co/gta/signal/

Abstract. A methodology based on kernel adaptive filtering termed DCKAF to support prediction tasks over one-dimensional time-series is proposed. DCKAF uses a linear combination of multiple *codebooks* to obtain the estimation from an input-output nonlinear mapping. This methodology employs a vector quantization based on statistic measures to check whether is necessary create a new *codebook*, then the nearest *codebook* to the current input sample is found. After that, *codebooks* are used to obtain the signal prediction at every instant, and evaluates if the current sample is added as a *codeword* or not as in traditional quantized kernel least mean square (QKLMS). Hence, DCKAF takes advantage of information learned on previous iterations to improve the system accuracy. The proposed methodology is tested on two one-dimensional time series and compared against QKLMS in terms of prediction accuracy. Obtained results show that DCKAF provides an effective way to predict time series improving prediction tasks.

1 Introduction

Nowadays, technological advances have allowed development of applications based on kernel adaptive filtering (KAF) in fields where multiple interleaved stationary time series are assumed as cyclostationary, e.g., weather forecasting, rotating machinery analysis, etc. Particularly, adaptive filters are designed for sequential learning, so that their free parameters must automatically adjust in response to cyclic variations within the operating environment. Generally, to perform better estimations, KAF makes use of input-output nonlinear mapping by minimizing a given instantaneous cost function, so that their adaptive ability relies on the correction of error prediction at every iteration.

The *least-mean-square* (LMS) is the baseline adaptive filtering rule that minimizes a Euclidean metric-based cost function. Some adaptive LMS-based methods had been also developed (like *Recursive Least-Squares* and *Extended Recursive Least-Squares*). Due to the knowledge of the cycles are practically not assured, accuracy on these L2-based algorithms is not enough, especially, under highly non-stationary conditions (not mentioning their high computational burden) [1]. To cope with this issue, generalized filtering methods include testing for cyclostationarity [2,3]. However, achieved the computational cost is also high and reached accuracy is still not good enough.

The KAF algorithms appear as another alternative transforming the input data into a high-dimensional feature space, but via a Reproducing Kernel Hilbert Space (RKHS).

These algorithms include different kernel LMS-based versions [1,4], kernel affine projection [5], etc. These online kernel learning methods compress the input space into a single quantization vector, or codebook, that is actualized at every time instant, so that its content permanently renews through the time. Hence, this memoryless codebook can not take advantage of learned information from cyclically interleaved processes. Furthermore, the KAF must learn each cyclic random structure as it were a new process, thereby riding of salient information and reducing performed accuracy.

Here, we propose to build cyclostationary dynamic KAF codebooks, termed *dynamic codebooks estimation for kernel adaptive filtering* (DCKAF), to support prediction tasks of one-dimensional time-series. As online kernel learning method, we select the quantized kernel least mean square (QKLMS) that has shown to perform high accuracy under nonstationary conditions, at the same time, providing a moderate computational cost [6]. However, contrary to the QKLMS, we generate multiple *codebooks*, which encode the relevant interleaved random processes, to be further linearly combined for cyclostationarity estimates. Namely, a convex combination of RKHSs is performed in a KAF framework [7]. Obtained results of DCKAF testing, carried out on simulated and real one-dimensional data, show an improved time series prediction accuracy with the benefit of better data interpretability.

2 Estimation of Dynamic KAF Codebooks

Let $\boldsymbol{u}\in\mathbb{R}^m$ be a m-dimensional input vector related to the desired output signal $d\in\mathbb{R}$ through the continuous nonlinear input-output mapping $f:\mathbb{R}^m\to\mathbb{R}$. Provided a sequence of input-output pairs $\{\boldsymbol{u}_t, d_t\}$ with $t=1,\dots,n\in\mathbb{N}$, The proposed DCKAF aims to find an approximation $f_t=\sum_{p\in P} w_p f_{t-1}^p(\boldsymbol{u}_t)$, so that each preceding *codebook* is defined in the form:

$$f_{t-1}^p=\sum_{j=1}^{N_{t-1}^p}\alpha_{t-1}^{p,j} g\left(\left\|\boldsymbol{u}_t-\boldsymbol{c}_{t-1}^{p,j}\right\|;\sigma_q\right) \tag{1}$$

here, $P\in\mathbb{N}$ is the number of *codebooks* extracted from the time series, $w_p\in\mathbb{R}$ is the weight associated to the *p-th codebook* $\boldsymbol{c}_{t-1}^p\in\mathbb{R}^{N_{t-1}^p\times m}$ being $N_{t-1}^p\in\mathbb{N}$ the number of *codewords* in $\boldsymbol{c}_{t-1}^p$, and $\alpha_{t-1}^{p,j}\in\mathbb{R}$ is the weight of the *j-th codeword* $\boldsymbol{c}_{t-1}^{p,j}\in\mathbb{R}^m$ in $\boldsymbol{c}_{t-1}^p$. The function $g(\cdot,\cdot)$ is a Mercer kernel mapping from the original feature space to a Reproducing Kernel Hilbert Space (RKHS). Thus, a pairwise kernel-based similarity measure between two samples is calculated. The well-known Gaussian kernel is employed to estimate pairwise sample relationship as $g(\mathcal{D}(\cdot,\cdot);\sigma)\triangleq\exp\left(-\mathcal{D}(\cdot,\cdot)^2/(2\sigma^2)\right)$, where $\sigma\in\mathbb{R}^+$ is the kernel bandwidth and $\mathcal{D}(\cdot,\cdot)\in\mathbb{R}^+$ is a distance operator. Besides, the weight associated to each *codebook* is computed using the following equation: $w_p=g\left(\left\|\xi(\boldsymbol{u_t})-E\{\xi(\boldsymbol{c}_{t-1}^{p,j})\}\right\|;\sigma_c\right)/\sum_{p\in P} w_p$, where $\xi(\cdot)$ is the variance of its argument, and $E\{\cdot\}$ is the expected value. The general scheme of DCKAF is presented in Algorithm 1, where $ws\in\mathbb{N}$ is the window size, σ_q and σ_c are the kernel bandwidths, $\epsilon_U\geq 0$ is the quantization size, and $0<\delta<1$ is the *codebook* quantization. At the beginning, the initial *codebooks* are built directly from the input time series at initial

Algorithm 1. – DCKAF

```
Inputs: ${\boldsymbol{u}_t, d_t}, w_s, \sigma_q, \sigma_c, \eta, \epsilon_U, \delta, \boldsymbol{C}_0 = \{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_{P_0}\}, \boldsymbol{\alpha}_0 = \{\boldsymbol{\alpha}_1 \ldots, \boldsymbol{\alpha}_{P_0}\}$
Computation:
  while $\{\boldsymbol{u}_t, d_t\}$ available do
    1) Create a new codebook if necessary
      if $\max_p g\left(\left\|\xi(\boldsymbol{u}_t) - E\{\xi(\boldsymbol{c}_{t-1}^{p,j})\right\|; \sigma_c\right) \geq \delta$
        $\boldsymbol{C}_t = \boldsymbol{C}_t; \boldsymbol{\alpha}_t = \boldsymbol{\alpha}_t$
      else
        $\boldsymbol{C}_t = \{\boldsymbol{C}_t, \boldsymbol{u}_t\}; \boldsymbol{\alpha}_t = \{\boldsymbol{\alpha}_t, \eta e_{t-1}\}$
      end
    2) Compute the weights of each codebook $(w_p)$
    3) Compute the adaptive filter output using Eq. (1)
    4) Compute the prediction error: $e_t = d_t - f_t$
    5) Find the nearest codebook $\boldsymbol{c}_t^{p^*}$ to the current input sample $\boldsymbol{u}_t$
      $p^* = \arg\max_p g\left(\left\|\xi(\boldsymbol{u}_t) - E\{\xi(\boldsymbol{c}_{t-1}^{p,j})\right\|; \sigma_c\right)$
    6) Updating codebooks as in traditional QKLMS
      $\kappa\left(\boldsymbol{u}_t, \boldsymbol{c}_t^{p^*}\right) = \max_{1 \leq j \leq N_{t-1}^{p^*}} g\left(\left\|\boldsymbol{u}_t - \boldsymbol{c}_t^{p^*,j}\right\|; \sigma_q\right)$
      if $\kappa\left(\boldsymbol{u}_t, \boldsymbol{c}_t^{p^*}\right) \geq \epsilon_U$
        $\boldsymbol{c}_t^{p^*} = \boldsymbol{c}_t^{p^*}; \alpha_t^{p^*,j^*} = \alpha_t^{p^*,j^*} + \eta e_t$
        where $j^* = \arg\max_{1 \leq j \leq N_{t-1}^{p^*}} g\left(\left\|\boldsymbol{u}_t - \boldsymbol{c}^{p^*,j}\right\|; \sigma_q\right)$
      else
        $\boldsymbol{c}_t^{p^*} = \{\boldsymbol{c}_t^{p^*}, \boldsymbol{u}_t\}; \boldsymbol{\alpha}_t^{p^*} = [\boldsymbol{\alpha}_t^{p^*}, \eta e_t]$
      end if
  end while
```

time $t_m \in \mathbb{N}$ using a sliding window of size $w_s \times o_l$, where $0 \leq o_l < 1$ is an overlapping interval expressed in percentage. Thus, we get a data representation matrix $\boldsymbol{d}_0 \in \mathbb{R}^{\frac{tm}{ws(1-ol)} \times ws}$. Then, the formulation for multiway spectral clustering is applied over matrix $\boldsymbol{d}_0$. This spectral clustering technique is a weighted kernel principal component analysis (WKPCA) approach based on primal-dual least-squares support vector machine (LS-SVM) [8]. Afterwards, the initial *codebooks* $\boldsymbol{C}_0$ with their corresponding coefficients $\boldsymbol{\alpha}_0$ are obtained. Thus, the *computation* stage evaluates if a new *codebook* is created with the current sample $\boldsymbol{u}_t$ or if this sample is added to an existing *codebook*. Then the weights associated to each *codebook* (w_p) are computed to obtain the output of adaptive filter as a linear combination of *codebooks*. Thereafter, the nearest codebook $\boldsymbol{c}_t^{p^*}$ to the current input sample $\boldsymbol{u}_t$ is found, and evaluates if the current sample is added as a *codeword* or not as in traditional QKLMS [6].

3 Experimental Set-up and Results

DCKAF is tested as a suitable method to support prediction tasks under cyclostationary conditions in KAF. To this end, our methodology stored the previously learned results and take advantage of this information to improve the prediction accuracy. So, a prediction task based on DCKAF can be summarized as follow: *i)* **initialization**, here the free parameters are fixed and the initial *codebooks* are computed from the initial time (t_m), *ii)* **model selection**, the *codebook* nearest to the current sample is chosen as the *best codebook*, then the current sample is assigned or discarded on this *codebook*, *iii)* **output of the adaptive filter**, on this stage, the prediction is carried out taking into account the information available in all *codebooks*. For concrete testing, system performance is validated in terms of system accuracy.

3.1 Databases

Two time series are used to test the proposed DCKAF methodology in terms of prediction accuracy. That is, *Lorenz* system and *Santa Fe* time-series. These two time series were chosen because both consist of a non-repeating pattern complex where the prediction accuracy can be affected by sudden changes on the signal. Firstly, we select the Lorenz System that is a dynamical system of a chaotic flow, noted for its butterfly shape, described by the following set of differential equations: $\dot{x}{=}\sigma(y-x),\ \dot{y}{=}{-}xz+\gamma x-y,\ \dot{z}{=}xy-Bz$. Two set of parameters are considered, and using these two parameter sets we generate two time series $H1$ and $H2$, which we concatenate to create a non-stationary time series with rapid transition as in [9]. Secondly, we use a data set obtained from the Santa Fe time-series competition[1]. Specifically, the *Data Set A* is used, which is an univariate time record of a single observed quantity, measured in a physics laboratory experiment. This data set is predictable on the shortest time, but has global events that can be harder to predict (sudden decay of the oscillations).

Lorenz system prediction: Here the DCKAF methodology is used to predict a *Lorenz* time series. In this sense, as we explained previously, two set of parameters are considered, i.e., one with $\sigma{=}10$, $\gamma{=}28$, $B{=}8/3$, another with $\sigma{=}16$, $\gamma{=}45.62$, $B{=}4$, these parameters were chosen in order to make our methodology comparable with the publications available in state of art [9]. To evaluate the versatility of DCKAF, 10 different *Lorenz* time series were generated using the same two sets of parameters, and Gaussian noise is added to the whole sequence using 10 different signal-to-noise ratio (SNR) to each one of them.

The problem setting for *Lorenz* prediction is as follow: the previous five points $\boldsymbol{u}_t{=}[d(t-5),\dots,d(t-1)]$ are used as the input vector (window size $w_s{=}5$) in QKLMS algorithm, while the previous 25 points $\boldsymbol{u}_t{=}[d(t-25),\dots,d(t-1)]$ are used as the input vector (window size $w_s{=}25$) in DCKAF to predict the current value d_t which is the desired response. In the simulations, the step size is choose as $\eta=0.81$, the quantization size $\epsilon_U{=}0.9$, and the overlapping percentage $o_l{=}20\%$, these values are the same for both methodologies. Also, in case of DCKAF the initial time to compute initial *codebooks* is fixed as $t_m{=}100$, and the *codebook* quantization size is $\delta{=}0.7$. All these parameters were fixed empirically. In addition, as proposed in [10], both required kernel band-width values, σ_q and σ_c in section 2, are automatically computed by maximizing the variance of considered Gaussian kernel matrix as:

$$\sigma^* = \arg\max_{\sigma} \left\{ \xi \left(g(\cdot;\sigma) \right) \right\}. \tag{2}$$

Fig. 1 shows the system accuracy for a representative example using QKLMS and DCKAF. On this particular example, our aim is to predict the Lorenz time-series shown in Fig. 1(a). However, to make the prediction task more challenging, we added Gaussian noise with $SNR{=}6$ to whole signal. The validation is carried out on the last 30% of the signal, and the obtained signal using DCKAF is shown in Fig. 1(c), on this figure there are three different colors (red, green, and blue) that represent each one of the three

[1] `http://www-psych.stanford.edu/~andreas/Time-Series/SantaFe.html`

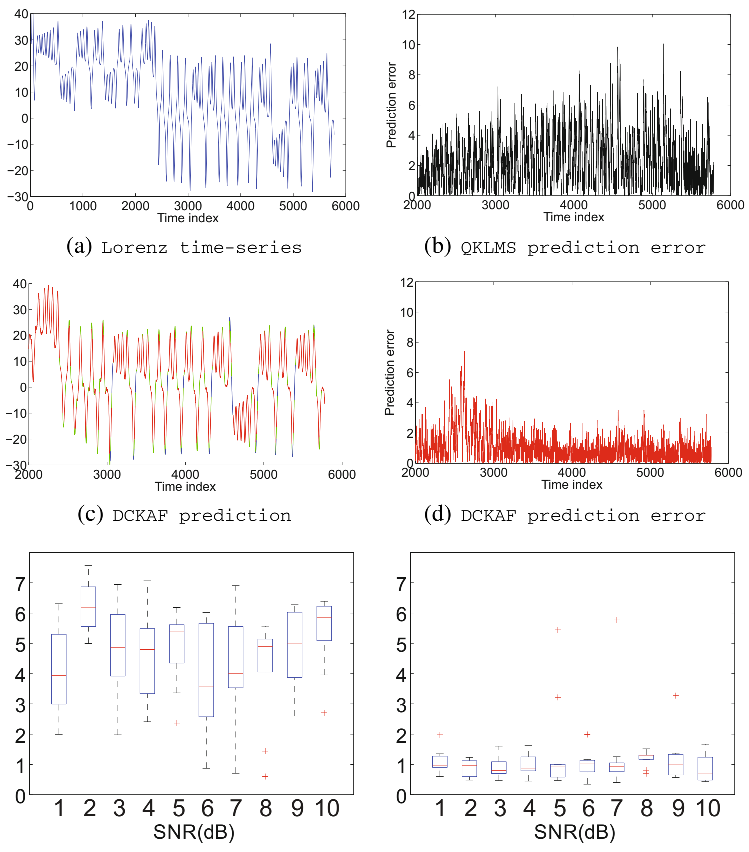
found *codebooks*, also, note that each color is associated to a particular signal behavior. The prediction errors along time for QKLMS and DCKAF are shown in Fig. 1(b) and Fig. 1(d), respectively.

Attained results in Fig. 1(d) shows that for this example the accuracy performance is higher in DCKAF than in QKLMS, i.e., note that $H2$ (second lorenz time-series) begin at iteration 3000 (see Fig. 1(a)) and the error in DCKAF decrease significantly along time, however, in case of QKLMS (Fig. 1(b)), the error increase in a continuous way. The reduction in prediction error for DCKAF (see Fig. 1(d)) can be attributed to the linear combination of *codebooks*, since take advantage from previous learning results, in contrast, QKLMS uses only one *codebook* to predict the signal, and this could not be proper under non-stationary conditions. Also, it is important to highly that in case of DCKAF the size on each *codebook* is adjusted adaptively, particularly in non-stationary conditions while the accuracy performance is acceptable, indeed, DCKAF adds *codewords* gradually to the *codebooks* on each step of the construction process and additionally considers the inter-*codebooks* relationships. So, DCKAF can be considered as a kernel adaptive filter that takes advantage of the information stored in all *codebooks*. Finally, it is important to highly that all *codewords* available into a specific *codebook* has identical statistical properties, i.e., each *codebook* encodes an unique dynamic.

The above results are a particular case that we choose to illustrate in a detailed manner the performance of DCKAF in terms of prediction accuracy, however, 10 different levels of SNR each one them with 10 different *Lorenz* time series were generated to evaluate the general performance of our algorithm under noisy and non-stationary conditions (as mentioned above). Fig. 1(e) and Fig. 1(f) shown the mean of relative errors over different noisy conditions applied to the *Lorenz* time-series. The relative errors are computed taking into account the last 30% of signal using the desired signal d with its corresponding obtained signal f on this interval. Note that, on these figures, the red line inside each box represents the median among the 10 relative errors obtained from the prediction for each SNR level, the upper and lower limit on each box represents the 25th and 75th percentiles respectively, and the operators + are the outliers. The *Lorenz* time-series employed on QKLMS are the same to the used in DCKAF.

The achieved results in Fig. 1(f), shows that DCKAF performance in terms of prediction accuracy is higher than the obtained results in QKLMS (Fig. 1(e)). In general terms, the obtained results are similar to the previous example, this fact proof that our methodology is robust under noisy conditions, i.e., the performance of prediction accuracy under different noisy conditions is very high (errors are under 2%), while the prediction accuracy in QKLMS is significantly affected (errors are upper 3%). There are some outliers in Fig. 1(f) (operator +), however, the prediction accuracy of these *Lorenz* time-series are in the corresponding range of QKLMS, which means that the prediction is still acceptable in comparison to QKLMS results.

Santa Fe time-series prediction: This time-series is used to evaluate the DCKAF versatility under non-stationary conditions. The aim is to predict the figure shown in Fig. 2(a). This signal is a time series concatenated two times. The problem setting for *Santa Fe* time-series prediction is as follow: for both QKLMS and DCKAF, the previous forty points $\boldsymbol{u}_t{=}[d(t-40),\ldots,d(t-1)]$ are used as the input vector (window size $w_s{=}40$), to predict the current value d_t which is the desired response, the step size is empirically



(a) Lorenz time-series (b) QKLMS prediction error

(c) DCKAF prediction (d) DCKAF prediction error

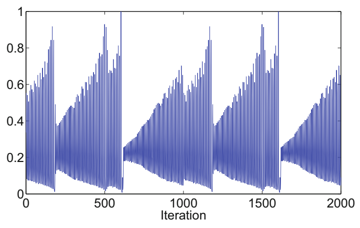
(e) QKLMS prediction performance (f) DCKAF prediction performance

Fig. 1. Lorenz time-series prediction

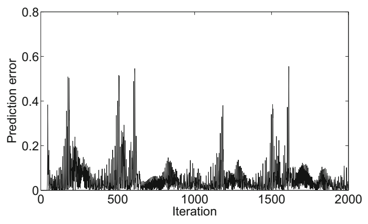
fixed as η=0.9, the quantization size ϵ_U=0.9, and the overlapping percentage o_l=20%. In case of DCKAF the initial time is t_m=170, and the *codebooks* quantization size is δ=0.7. The required kernel band-width values σ_q and σ_c are automatically computed using (2).

Fig. 2 shows the system accuracy using QKLMS and DCKAF. Similar as in Fig. 1, our aim is to predict the *Santa Fe* time-series shown in Fig. 2(a) (concatenated signal). The validation is carried out on the last 30% of the signal. The obtained results in terms of prediction error for both QKLMS and DCKAF are shown in Fig. 2(b) and Fig. 2(d), respectively. The obtained signal using DCKAF is shown in Fig. 2(c).

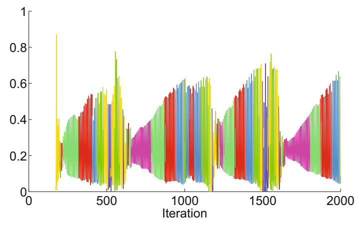
Obtained results in Fig. 2 confirm that our algorithm is robust against non-stationary conditions, i.e., prediction error in DCKAF (Fig. 2(d)) is better than QKLMS (Fig. 2(b)), in this sense, note that prediction error in DCKAF decreases along the time, which is a clear indication that the linear combination used by DCKAF improves the system accuracy. Remember that, signal in Fig. 2(a) is composed by the concatenation of



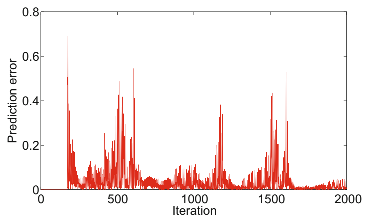
(a) Santa Fe time-series



(b) QKLMS prediction error



(c) DCKAF prediction



(d) DCKAF prediction error

Fig. 2. Santa Fe time-series prediction

one Santa Fe time-series, that is, one from iteration 1 to 1000 and another from iteration 1001 to 2000 (both Santa Fe time-series are exactly the same), therefore there are mainly 3 segments in Fig. 2(a), the first one is between 0 to 186 and 1001 to 1186, the second one is between 187 to 613 and 1187 to 1613, and the third one is between 614 to 1000 and 1614 to 2000. Finally, each color show in signal reconstruction using DCKAF (Fig. 2(c)) represents one different *codebook*. Note that DCKAF identify exactly the same *codebook* sequence on each segment, e.g., on the last segment (614 to 1000 and 1614 to 2000), when the sample number 614 arrives, the algorithm does not known this dynamic and therefore a new *codebook* is created (magenta color), after that another *codebook* is created (cyan color) because on this point a new dynamic is identified, and so on. Then when the same segment appears again at iteration 1614, the algorithm does not create a new *codebook* because this dynamic it was previously learned at iteration 614, and therefore the *codebook* represented by the magenta color is employed once again. Additionally, note that when magenta *codebook* is used by second time (iteration 1614), the error prediction is reduced significantly (Fig. 2(d)) in comparison when this *codebook* was employed by first time (iteration 614), which means that DCKAF take advantage of the information stored on previous *codebooks*, while QKLMS in non-stationary conditions forgets the previous learning results and re-learning the input-output mapping when system switch to a new state, and consequently the prediction error in QKLMS is higher than DCKAF. The attained results with DCKAF are promising, however, the computational burden in DCKAF is higher than QKLMS, because employs a cost function considering all the previous states to obtain a better prediction.

4 Conclusions and Future Work

Here, a methodology based on kernel adaptive filtering to support prediction tasks on one-dimensional time-series was presented. To this end, the proposed methodology uses a linear combination of *codebooks* to obtain the estimation from an input-output nonlinear mapping. After that, a vector quantization based on statistic measures is employed to check whether a new *codebook* must be created. Our methodology takes advantage of information learned on previous states to improve the system accuracy. The obtained results in terms of system accuracy show DCKAF is a suitable tool to predict one-dimensional time series under noisy and non-stationary conditions. Particularly, attained results in *Santa Fe* time-series exhibits how DCKAF take advantage of the information stored on previous *codebooks*. The above demonstrates that using multiple *codebooks* with particular statistical properties, improves the prediction tasks.

As future work proposed DCKAF should be tested in multi-dimensional time series to validate its accuracy and stability for prediction tasks. Moreover, other algorithms based on kernel adaptive filtering must be tested using DCKAF. Finally, a methodology to tuning free parameters should be designed.

Acknowledgements. This work is supported by the project "Sistemas de interacción cerebro - máquina (BMI) a partir de registros de actividad eléctrica cerebral", *Programa Doctoral Becas COLCIENCIAS-Formación de alto nivel para la ciencia, la tecnología y la innovación año 2014.*, and *Programa Nacional de Formación de Investigadores "Generación del Bicentenario" 2011/2012* all funded by COLCIENCIAS.

References

1. Liu, W., Pokharel, P., Principe, J.: The kernel least-mean-square algorithm. IEEE Transactions on Signal Processing 56(2), 543–554 (2008)
2. Madisetti, V.: Digital Signal Processing Fundamentals, 2nd edn. CRC Press, Inc., Boca Raton (2009)
3. Lundén, J., Koivunen, V., Huttunen, A., Poor, H.V.: Collaborative cyclostationary spectrum sensing for cognitive radio systems. IEEE Transactions on Signal Processing 57(11) (2009)
4. Liu, W., 0002, I.P., 0002, Y.W., Príncipe, J.C.: Extended kernel recursive least squares algorithm. IEEE Transactions on Signal Processing 57(10), 3801–3814
5. Liu, W., Príncipe, J.C.: Kernel affine projection algorithms. EURASIP (2008)
6. Zhao, S., Zhu, P., Jose, P.: Quantized kernel least mean square algorithm. IEEE Trans. on Neural Networks and Learning Systems 23, 22–32 (2012)
7. Pokharel, R., Seth, S., Principe, J.C.: Mixture kernel least mean square. In: IJCNN (2013)
8. Alzate, C., Suykens, J.A.K.: Multiway spectral clustering with out-of-sample extensions through weighted kernel pca. 32(2), 335–347 (2010)
9. Zhao, S.: From fixed to adaptive budget robust kernel adaptive filtering. PhD thesis, University of Florida, Gainesville, FL, USA (2012)
10. Cardenas-Pena, D., Orbes-Arteaga, M., Castro-Ospina, A., Alvarez-Meza, A., Castellanos-Dominguez, G.: A kernel-based representation to support 3d mri unsupervised clustering. In: ICPR (2014)