Quantized kernel least mean square with desired signal smoothing

Xiguang Xu, Hua Qu, Jihong Zhao, Xiaohan Yang, and Badong Chen

Quantized kernel least mean square (QKLMS) is a simple yet efficient online learning algorithm, which reduces the computational cost significantly by quantizing the input space to constrain the growth of network size. The QKLMS considers only the input space compression and assumes that the desired outputs of the quantized data are equal to those of the closest centres. In many cases, however, the outputs in a neighbourhood may have big differences, especially when underlying system is disturbed by impulsive noises. Such fluctuation in desired outputs may seriously deteriorate the learning performance. To address this issue, we propose in this work a simple online method to smooth the desired signal within a neighbourhood corresponding to a quantization region. The resulting algorithm is referred to as the QKLMS with desired signal smoothing (or briefly, QKLMS-S). The desirable performance of the new algorithm is confirmed by Monte Carlo simulations.

Introduction: Recently, the kernel adaptive filtering (KAF) has become an emerging and promising subfield of the online kernel learning [1]. The KAF algorithms extend naturally the classical linear adaptive filters [2] into reproducing kernel Hilbert spaces (RKHS), which are very appealing for nonlinear adaptive signal processing. Among the KAF algorithms the kernel least mean square (KLMS) is the simplest, which is easy to implement, yet effective for learning complex systems [3]. When the kernel is Gaussian, the KLMS naturally creates a growing radial-basis function (RBF) network, with weights directly related to the errors at each sample.

The main bottleneck of KAF algorithms is their growing network structure with each new sample. As a result, the computational complexity and memory requirement will grow linearly with the number of training data, which poses a big challenge for large data sets. To address this problem, a variety of sparsification methods have been proposed to curb the network growth. In [4], a novel quantization approach was proposed and a quantized kernel least mean square (QKLMS) algorithm was developed. The basic idea of QKLMS is to compress the input space to constrain the network growth by using the idea of quantization [4-6]. Unlike most of the other sparsification methods, the quantization approach utilizes the "redundant" data to update the coefficient of the closest centre instead of purely discarding them, which may yield better accuracy and a more compact network. The QKLMS algorithm considers only the input space compression and assumes that the desired outputs of the quantized data are equal to those of the closest centres. In many cases, however, the outputs in a neighbourhood may have big differences, especially when underlying system is disturbed by impulsive noises. Such fluctuation in desired outputs may seriously deteriorate the learning performance.

In order to deal with this issue, we propose in this work a new algorithm, called the QKLMS with desired signal smoothing (QKLMS-S), to improve the performance of QKLMS. A simple online smoothing (averaging) method is proposed, which averages the desired signals within a neighbourhood corresponding to a quantization region in an iteration manner. The averaged value in the corresponding quantization region is then used as the desired output when a sample is quantized. The QKLMS-S is expected to perform significantly better than QKLMS.

*QKLMS:*In KAF, the input data are transformed into a high dimensional feature space by using a certain nonlinear mapping, and then a linear filter is applied in the feature space. Let $\boldsymbol{u}(i) \in \mathbb{R}^m$ be an m-dimensional input vector at time instant i and $d(i) \in \mathbb{R}$ the desired response. The goal of a KAF algorithm is to find an estimator of the underlying mapping $d = f(\boldsymbol{u})$ sequentially based on the data $\{\boldsymbol{u}(i), d(i)\}, i = 1, 2, \ldots$.

In the feature space, an inner product can be calculated by using a positive definite kernel function satisfying Mercer's condition [7]: $\kappa(\boldsymbol{u}, \boldsymbol{u}') = \varphi(\boldsymbol{u})^T \varphi(\boldsymbol{u}')$, where φ is a nonlinear mapping. A widely used kernel is the Gaussian kernel:

$$\kappa(\boldsymbol{u}, \boldsymbol{u}') = \exp\left(-\frac{\| \boldsymbol{u} - \boldsymbol{u}' \|^2}{2\sigma^2}\right) \tag{1}$$

where $\sigma > 0$ is the kernel size (or kernel bandwidth).

The KLMS is, essentially, a least mean square (LMS) algorithm in feature space [3]. The learning rule of the KLMS in original input space can be formulated as

$$\begin{cases} f_0 = \mathbf{0} \\ e(i) = d(i) - f_{i-1}(\boldsymbol{u}(i)) \\ f_i = f_{i-1} + \eta e(i) \kappa(\boldsymbol{u}(i), \cdot) \end{cases} \tag{2}$$

where $e(i)$ is the prediction error, η is the step-size parameter, and f_i denotes the learned mapping at iteration i. KLMS allocates a new kernel unit for every new example with input $\boldsymbol{u}(i)$ as a new centre and $\eta e(i)$ as the coefficient, yielding a growing RBF network.

To curb the network growth of KLMS, the QKLMS quantizes the input space using a simple online vector quantization method [4]. The learning rule of QKLMS can be simply expressed as

$$\begin{cases} f_0 = \mathbf{0} \\ e(i) = d(i) - f_{i-1}(\boldsymbol{u}(i)) \\ f_i = f_{i-1} + \eta e(i) \kappa(Q[\boldsymbol{u}(i)], \cdot) \end{cases} \tag{3}$$

where $Q[\cdot]$ denotes a quantization operator in input space:

$$Q[\boldsymbol{u}(i)] = \begin{cases} \boldsymbol{C}_{j^*}(i-1) & dis(\boldsymbol{u}(i), \boldsymbol{C}(i-1)) \le \varepsilon \\ \boldsymbol{u}(i) & dis(\boldsymbol{u}(i), \boldsymbol{C}(i-1)) > \varepsilon \end{cases} \tag{4}$$

where $\boldsymbol{C}(i)$ is the codebook (centre set) at time instant i, ε is the quantization size whose description can be found in [4,5,8], and $dis(\boldsymbol{u}(i), \boldsymbol{C}(i-1))$ denotes the distance between $\boldsymbol{u}(i)$ and $\boldsymbol{C}(i-1)$:

$$dis(\boldsymbol{u}(i), \boldsymbol{C}(i-1)) = \min_{1 \le j \le size(\boldsymbol{C}(i-1))} \| \boldsymbol{u}(i) - \boldsymbol{C}_j(i-1) \| \tag{5}$$

where $\boldsymbol{C}_j(i-1)$ denotes the j th element of the codebook $\boldsymbol{C}(i-1)$, and

$$j^* = \arg\min_{1 \le j \le size(\boldsymbol{C}(i-1))} \| \boldsymbol{u}(i) - \boldsymbol{C}_j(i-1) \| \tag{6}$$

If $dis(\boldsymbol{u}(i), \boldsymbol{C}(i-1)) \le \varepsilon$, the codebook remains unchanged: $\boldsymbol{C}(i) = \boldsymbol{C}(i-1)$, and $\boldsymbol{u}(i)$ is quantized to the closest center through updating the coefficient of that center: $\boldsymbol{\alpha}_{j^*}(i) = \boldsymbol{\alpha}_{j^*}(i-1) + \eta e(i)$, where $\boldsymbol{\alpha}_{j^*}(i)$ denotes the j^* th element of the coefficient vector $\boldsymbol{\alpha}(i)$. Otherwise, a new center and corresponding new coefficient will be added:

$$\boldsymbol{C}(i) = \{\boldsymbol{C}(i-1), \boldsymbol{u}(i)\} \tag{7}$$

$$\boldsymbol{\alpha}(i) = [\boldsymbol{\alpha}(i-1), \eta e(i)] \tag{8}$$

QKLMS-S: In QKLMS, the desired outputs of the quantized data are assumed to be equal to those of the closest centres.To address the fluctuation of the desired outputs within a quantization region, we propose a new algorithm, called the QKLMS-S, in which the desired outputs are smoothed by using an average value within a neighbourhood (i.e. quantization region). Specifically, the QKLMS-S algorithm can be expressed as:

$$\begin{cases} f_0 = \mathbf{0} \\ e(i) = \boldsymbol{D}_{j^*}(i) - f_{i-1}(\boldsymbol{u}(i)) \\ f_i = f_{i-1} + \eta e(i) \kappa(Q[\boldsymbol{u}(i)], \cdot) \end{cases} \tag{9}$$

where $\boldsymbol{D}_{j^*}(i)$ denotes the smoothed desired output corresponding to the j^* th quantization region at time instant i. If $\boldsymbol{u}(i)$ satisfies $dis(\boldsymbol{u}(i), \boldsymbol{C}(i-1)) = \left\| \boldsymbol{u}(i) - \boldsymbol{C}_{j^*}(i-1) \right\| \le \varepsilon$, then $\boldsymbol{N}(i)$ and $\boldsymbol{D}(i)$ are updated as follows:

$$\boldsymbol{N}_{j^*}(i) = \boldsymbol{N}_{j^*}(i-1) + 1 \tag{10}$$

where $\boldsymbol{N}_{j^*}(i)$ denotes the number of the data falling into the j^* th quantization region, and

$$\boldsymbol{S}_{j^*}(i)=\left[\boldsymbol{S}_{j^*}(i-1),\ \ d(i)\right] \tag{11}$$

where $\boldsymbol{S}_{j^*}(i)$ denotes the desired output set corresponding to the j^* th quantization region at time instant i . Then the smoothed desired output $\boldsymbol{D}_{j^*}(i)$ can be calculated as:

$$\begin{aligned}\boldsymbol{D}_{j^*}(i)&=\frac{\sum_{k=1}^{\boldsymbol{N}_{j^*}(i)}\boldsymbol{S}_{j^*,k}(i)}{\boldsymbol{N}_{j^*}(i)}\\&=\frac{\sum_{k=1}^{\boldsymbol{N}_{j^*}(i-1)}\boldsymbol{S}_{j^*,k}(i-1)+d(i)}{\boldsymbol{N}_{j^*}(i)}\\&=\boldsymbol{D}_{j^*}(i-1)\bullet\frac{\boldsymbol{N}_{j^*}(i-1)}{\boldsymbol{N}_{j^*}(i)}+\frac{d(i)}{\boldsymbol{N}_{j^*}(i)}\\&=\frac{\boldsymbol{N}_{j^*}(i-1)\boldsymbol{D}_{j^*}(i-1)+d(i)}{\boldsymbol{N}_{j^*}(i)}\end{aligned} \tag{12}$$

where $\boldsymbol{S}_{j^*,k}(i)$ is the k th entry of $\boldsymbol{S}_{j^*}(i)$. Otherwise, a new centre and the corresponding coefficient will be added as (7) and (8), and

$$\boldsymbol{D}(i)=\left[\boldsymbol{D}(i-1),\ d(i)\right] \tag{13}$$

$$\boldsymbol{N}(i)=\left[\boldsymbol{N}(i-1),\ 1\right] \tag{14}$$

The learned mapping after the iteration i is:

$$f_i=\sum_{j=1}^{size(\boldsymbol{C}(i))}\boldsymbol{\alpha}_j(i)\kappa\left(\boldsymbol{C}_j(i),\cdot\right) \tag{15}$$

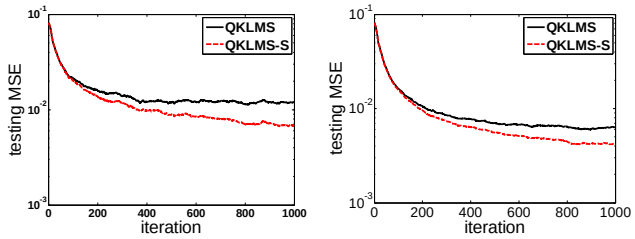
Simulation Results: Consider the Mackey-Glass (MG) chaotic time series prediction. The time series is generated from the following time-delay ordinary differential equation [1]:

$$\frac{dx(t)}{dt}=-bx(t)+\frac{ax(t-\tau)}{1+x(t-\tau)^{10}} \tag{16}$$

with $b=0.1, a=0.2, \tau=30$ .The signal is discretized at a sampling periodof 6 seconds. The goal is to predict the present value using the previous points. A segment of 1000 samples is used as the training data and another 100 as the test data. The time-embedding dimension number is chosen as 7, i.e., $\boldsymbol{u}(i)=\left[x(i-1),x(i-2),...,x(i-7)\right]^T$ is used as an input vector to predict $x(i)$, the desired response in the time-series prediction. In the simulation, the training data are disturbed by additive noises. In particular, we consider two noises: a) Gaussian noise with zero-mean and variance 0.04; b) Mix-Gaussiannoise with density

$$0.95\times N(0,0.04)+0.05\times N(0,1) \tag{18}$$

where $N(\mu,\sigma^2)$ denotes Gaussian density with mean μ and variance σ^2 .



(a) (b)

Fig.1*Average learning curves with different noises: (a)Gaussian; (b) Mix-Gaussians*

We compare the performance of QKLMS and QKLMS-S. The Gaussian kernel with width 1.0 is selected as the Mercer kernel. The step-sizes of the two algorithms are experimentally chosen so as to yield almost the same initial convergence speed. The quantization size is set at ε=0.2 .The simulation results are averaged over 100 independent Monte Carlo runs with different segments of data. The average learning curves and network growth curve are shown, respectively, in Fig. 1 and Fig. 2, and the testing MSEs at final iteration are listed in Table 2. One can see clearly that the QKLMS-S achieves much better performance than the QKLMS, no matter when the noise is Gaussian or Mix-Gaussian. In this example, the network size is reduced to around 200, which is very small compared with the original network size (which is 1000).

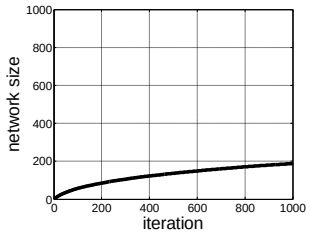


Fig.2*Network growth curve*

Table 2: Testing MSE at final iteration

	Testing MSE (Gaussian noise)	Testing MSE (Mix-Gaussians noise)
QKLMS	0.0062354 ± 9.4799e-05	0.011876 ± 0.00014243
QKLMS-S	0.0042328 ± 4.8773e-05	0.0069045 ± 0.00014989

Conclusion: A new online kernel learning algorithm, called QKLMS-S algorithm, has been proposed, which improves the learning performance of QKLMS by smoothing the desired signals. Simulation results confirmed the desirable performance of the proposed algorithm.

Acknowledgments: This work was supported by 973 Program (No. 2015CB351703) and the National Natural Science Foundation of China (no. 61372152，no. 61372092).

X. Xu, H. Qu, J. Zhao, X. Yang and B. Chen (*School of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an 710049, People's Republic of China*)

E-mail:chenbd@mail.xjtu.edu.cn

References

1.Liu, W., Pr ńcipe, J.C., and Haykin, S.:'Kernel Adaptive Filtering: A Comprehensive Introduction' (Wiley-IEEE Press,NewYork, 2010)
2. Haykin, S.:'Adaptive Filter Theory' (PrenticeHall, New Jersey, 2002, fourth edition)
3. Liu, W., Pokharel, P.P., and Pr ńcipe, J.C.:'The kernel least-mean-square algorithm',*IEEE Trans. Signal Process.*, 2008, 56, (2), pp. 543–554
4. Chen, B., Zhao, S., Zhu, P., and Pr ńcipe, J.C.: 'Quantized kernel least mean square algorithm',*IEEE Trans. Neural Netw. Learn. Syst.*, 2012, 23, (1), pp. 22-32
5. Chen, B., Zhao, S., Zhu, P., and Pr ńcipe, J.C.:'Quantized kernel recursive least squares algorithm', *IEEE Trans. Neural Netw. Learn. Syst.*, 2013, 24, (9), pp. 1484-1491
6. Zhao, S., Chen, B., and Principe, J.C.:'Fixed budget quantized kernel least mean square algorithm', *Signal Processing*, 2013, 93, (9),pp. 2759-2770
7. Vapnik, N.V.:'The Nature of Statistical Learning Theory' (Springer-Verlag New York, Inc., NY, USA,1995)
8. Platt, J.:'A resource-allocating network for function interpolation', *Neural Comput.*, 1991, 3, (2), pp. 213–225