KERNEL LEAST MEAN SQUARE BASED ON CONJUGATE GRADIENT

Siyuan Peng[1]*, Zongze Wu*[2]*, Wentao Ma*[3]*, Badong Chen*[4]

[1]Nanyang Technological University, School of Electrical and Electronic Engineering, Singapore
[2]Guangdong University of Technology, School of Electrical and Information Engineering, Guangzhou, China
[3]Xi'an University of Technology, School of Automation and Information Engineering, Xi'an, China
[4]Xi'an Jiaotong University, Institute of Artificial Intelligence and Robotics, Xi'an China

ABSTRACT

Kernel least mean square (KLMS) algorithm has been successfully applied in fields of adaptive filtering and online learning due to their ability to solve sequentially nonlinear problems by implicitly mapping the input signal to a high-dimensional reproducing kernel Hilbert space (RKHS). In this paper, we propose a novel adaptive algorithm called KLMS based on conjugate gradient (KLMS-CG), which uses the orthogonal search directions, instead of using the traditional steepest descent approach, to improve the convergence speed. Further, the quantized KLMS based on conjugate gradient (QKLMS-CG) is proposed to curb the growth of network. Simulation results indicate that the new algorithm can converge faster than the original KLMS while maintaining excellent accuracy.

Index Terms— Conjugate gradient, KLMS, QKLMS

1. INTRODUCTION

Kernel adaptive filtering (KAF) [1] has become an emerging and promising subfield of online kernel learning. The basic idea of kernel adaptive filtering is to map the input into a high-dimensional reproducing kernel Hilbert space (RKHS) [2], and then implement well-established linear adaptive algorithms using the linear structure of this space. Typical KAF algorithms include the kernel least mean square (KLMS) [3], kernel affine projection algorithms (KAPA) [4], kernel recursive least squares (KRLS) [5], and extend kernel recursive least squares (EX-KRLS) [6]. When the kernel is Gaussian, they are essentially growing RBF networks, where the weights are directly related to the errors at each sample.

The KLMS is the simplest, yet most effective KAF algorithm for online learning. However, its convergence speed is usually slow, because it is just a simple stochastic gradient based algorithm. To overcome this issue, in this work we propose a new KLMS algorithm, called KLMS based on conjugate gradient (KLMS-CG), which is derived by using a conjugate gradient (CG) to search the optimal solution. In general, the CG algorithms (by employing an orthogonal search direction) converge much faster than the simple steepest descent algorithm [7], and also have lower computational complexity when compared with the Newtons methods [8], which need to compute a matrix inversion. Therefore, the proposed algorithm will converge faster than the original KLMS while still keeping low computational complexity. In the literature, there are many variants of CG methods, such as Fletcher and Reeves (FR) [9], Hestenes and Steifel (HS) [10], Polak and Ribiere (PR) [11], Liu and Storey (LS) [12] and Dai and Yuan (DY) [13]. In this work, we adopt the FR conjugate gradient method to enhance the convergence speed of KLMS. Powell [14] showed that the FR conjugate gradient method with exact line searches is a powerful approach on general functions. Al-Baali [15] extended this result to inexact line searches with a strong Wolf condition.

There is a computational bottleneck for KLMS, namely, its growing structure with each sample, which results in increasing computational costs and memory requirements especially in continuous adaptation scenarios. In order to curb the network growth and to obtain a compact representation, recently, a novel online quantization approach has been successfully applied in kernel adaptive filtering to reduce the increasing of the networks [16, 17, 18, 19, 20, 21]. In the present work, the quantization method will be applied to the KLMS-CG, and the resulting algorithm is called quantized KLMS-CG (QKLMS-CG). Simulation results show that QKLMS-CG can achieve desirable performance (say faster convergence speed) while maintaining a smaller network scale.

The organization of this paper is as follows. In Section II, after briefly describing the KLMS, we develop the KLMS-CG and QKLMS-CG. Simulation results are then presented in Section III. Finally, section IV gives the conclusion.

2. KLMS-CG

Consider the learning of a continuous nonlinear input-output mapping $f : U \to R$

$$y = f(u), u \in U \subset R^m, y \in R \tag{1}$$

This work was supported by 973 Program (no. 2015CB351703) and National Natural Science Foundation of China (no. 61372152, no. 61271210).

where u is an m-dimensional input vector, $U \subset R^m$ is the input domain, and y is the output signal. If a sequence of input-output pairs $\{u(i), d(i), i = 1, 2 \ldots\}$ is received, our goal is to find an approximation $\hat{f}$ of the mapping f based on the available data. A kernel adaptive filter is an online learning machine, which finds an estimate f of such that f_i (the estimate at the i-th iteration) is updated on the basis of the last estimate f_{i-1} and the current example $\{u(i), d(i)\}$.

The kernel methods are powerful in learning a nonlinear mapping. A Mercer kernel is a continuous, symmetric and positive-definite function $\kappa : U \times U \subset R$. In practical, Gaussian kernel is a commonly used kernel, defined by

$$\kappa(u, u^{'}) = \exp(-\|u - u^{'}\|^2/2h^2) \tag{2}$$

where $h > 0$ is the Gaussian kernel bandwidth. According to Mercers theorem, any Mercer kernel $\kappa(u, u^{'})$ induces a nonlinear mapping φ from the input space U to a high-dimensional RKHS (or feature space) such that the following relationship (the so-called "*kernel trick*") holds [1]:

$$\varphi(u)^T \varphi(u^{'}) = \kappa(u, u^{'}) \tag{3}$$

where T denotes the transpose operator ($\varphi(u)$ is regarded as a vector).

2.1. KLMS

The KLMS is, essentially, a least mean square (LMS) algorithm in feature space. First, the kernel-induced mapping φ is employed to transform the input $u(i)$ into the feature space as $\varphi(i) = \varphi(u(i))$. Then using the stochastic steepest descent method on the transformed example sequence $\{\varphi(i), d(i)\}$ yields the following KLMS algorithm:

$$\begin{cases} \omega(0) = 0 \\ e(i) = d(i) - \omega(i-1)^T \varphi(i) \\ \omega(i) = \omega(i-1) + \eta e(i) \varphi(i) \end{cases} \tag{4}$$

where $\omega(i)$ denotes an estimate of the weight vector in feature space, $e(i)$ is the prediction error at iteration i , and η is the step-size. From (4), it follows easily that

$$\begin{cases} \omega(i) = \sum_{j=1}^{i} \alpha(j) \varphi(j) \\ \alpha(i) = \eta e(i) \end{cases} \tag{5}$$

where $\alpha(i)$ stands for the coefficients of the KLMS filter. If identifying $\varphi(i) = \kappa(u, .)$, we obtain

$$f_i(u) = \omega(i)^T \varphi(u) = \sum_{j=1}^{i} \alpha(j) \kappa(u(j), u) \tag{6}$$

The KLMS produces a growing RBF-type network which allocates a new hidden node for every new example by setting the input $u(i)$ as the center. Let $\boldsymbol{\alpha}(i) = [\alpha(1), \cdots, \alpha(i)]^T$, f_i can also be expressed as $f_i = \boldsymbol{\alpha}(i)^T K(i)$, where $K(i) = [\kappa(u(1), .), \cdots, \kappa(u(i), .)]$.

2.2. KLMS-CG

Instead of using the steepest descent approach, the KLMS-CG applies the orthogonal search directions to improve the convergence speed, which, with the FR conjugate gradient [9], takes the following form:

$$\begin{cases} \omega(0) = 0 \\ e(i) = d(i) - \omega(i-1)^T \varphi(i) \\ \omega(i) = \omega(i-1) + \eta \rho(i) \\ \rho(i) = -g(i) + \beta(i) \rho(i-1) \\ g(i) = -e(i) \varphi(i) \\ \beta(i) = \frac{\kappa(g(i), g(i))}{\kappa(g(i-1), g(i-1))} \end{cases} \tag{7}$$

where $\rho(i)$ is a new search direction, $g(i)$ is the stochastic gradient direction, $\beta(i)$ is the conjugate coefficient, and $\kappa(., .)$ is a kernel function which can be different from the kernel for inducing the RKHS.

In order to ensure the convergence of the algorithm, one needs to reset the CG algorithm with some approaches, otherwise the algorithm near the solution (where the problem is nearly quadratic) may get away from the solution and the advantage of conjugate directions will be lost [22]. In this work, we reset the CG algorithm every M iteration. How often the algorithm is reset will influence the convergence performance [23]. In practical applications, the value of M can be selected by scanning the performance or just set manually. In section III, we will perform simulations to demonstrate how the M value will influence the convergence behaviors of the KLMS-CG algorithm. How to select an optimal M value is, however, a big challenge and is left open in this work. In this way, the new search direction can be written as

$$\rho(i) = \begin{cases} -g(i) & i \texttt{ is } M \texttt{ multiple} \\ -g(i) + \beta(i) \rho(i-1) & \texttt{otherwise} \end{cases} \tag{8}$$

Next, we derive the update equations for $\boldsymbol{\alpha}(i)$. Let N be the sample number, and M be a positive integer. For $i = 1, 2, \cdots, N$, k denotes the remainder of i divided by M. The value of k is in the range 0 to $M - 1$. Below we consider two cases:

A: If $k = 0$, in this case

$$\begin{cases} g(i) = -e(i) \varphi(i) \\ \beta(i) = 0 \\ \rho(i) = -g(i) = e(i) \varphi(i) \end{cases} \tag{9}$$

It follows easily that

$$\begin{cases} \boldsymbol{\alpha}_j(i) = \boldsymbol{\alpha}_j(i-1) & j = 1, \cdots, i-1 \\ \boldsymbol{\alpha}_j(i) = \eta e(i) & j = i \end{cases} \tag{10}$$

where $\boldsymbol{\alpha}_j(i)$ denotes the j-th element of $\boldsymbol{\alpha}(i)$.

B: If $k \neq 0$, in this case, we have

$$\begin{cases} g(i) = -e(i)\varphi(i) \\ \beta(i) = \frac{\kappa(g(i),g(i))}{\kappa(g(i-1),g(i-1))} \\ \rho(i) = e(i)\varphi(i) + \sum_{t=i-k}^{i-1} \left(\prod_{s=t+1}^{i} \beta(s) \right) e(t)\varphi(t) \end{cases} \tag{11}$$

Then the coefficients will be updated as:

$$\begin{cases} \boldsymbol{\alpha}_j(i) = \eta e(i) & j = i \\ \boldsymbol{\alpha}_j(i) = \boldsymbol{\alpha}_j(i) + \sum_{t=j}^{i-1} \left(\prod_{s=t+1}^{i} \beta(s) \right) \eta e(t) & \\ & i-k \leq j \leq i-1 \\ \boldsymbol{\alpha}_j(i) = \boldsymbol{\alpha}_j(i-1) & j < i-k \end{cases} \tag{12}$$

Similar to the original KLMS, the KLMS-CG algorithm yields a growing RBF-type network with each sample as a hidden node. This leads to increasing computational complexity and memory requirements especially for continues adaptation scenarios. In this work, we apply a simple online vector quantization approach to curb the network growth, and the resulting algorithm is called the quantized KLMS-CG (QKLMS-CG). The same approach has been successfully used in the quantized kernel least mean square (QKLMS) algorithm [16] and the quantized kernel recursive least squares (QKRLS) algorithm [17]. The basic idea of quantization approach is to merge the new node into the closest node if the distance between the two nodes is smaller than a preset threshold, namely the quantization factor (or quantization size). When the quantization factor is set to zero, the QKLMS-CG will reduce to the KLMS-CG. The pseudo code for QKLMS-CG algorithm is described in Table 1. For more details about the quantization approach, readers are referred to [16, 17].

For KLMS, the computational complexity at the i-th iteration is $O(i)$. The computational cost of KLMS-CG is however not only determined by the value of i but also related to the value of M, which needs $O(iM)$ operations at the i-th iteration. Obviously, the computational complexity of KLMS-CG is higher than that of KLMS. With M increasing, the complexity of KLMS-CG becomes higher. But for QKLMS-CG, the computational cost is $O(SM)$, where S denotes the quantization dictionary size.

Table 1. QKLMS-CG Algorithm

Input: $\{u(i) \in U, d(i)\}, i = 1, 2, \cdots$

Initialization: $\eta > 0$, $h > 0$, $\boldsymbol{\alpha}(1) = \eta d(1)$, $\beta(1) = 0$, $g(1) = -d(1)u(1)$, quantization size $\varepsilon > 0$, and the center set $C(1) = \{u(1)\}$

Computation:

while $\{u(i), d(i)\}(i > 1)$ available **do**

1) Compute the output of the adaptive filter:
$f_i(u) = \sum_{j=1}^{size(C(i-1))} \boldsymbol{\alpha}_j(i-1)\kappa(C_j(i-1), u(i))$

2) Compute the error: $e(i) = d(i) - f_i(u)$

3) Compute the distance between $u(i)$ and $C(i-1)$:
$dis(u(i), C(i-1)) = \min_{1 \leq j \leq size(C(i-1))} \|u(i) - C_j(i-1)\|$

4) If $dis(u(i), C(i-1)) \leq \varepsilon$, then $C(i) = C(i-1)$, and $\boldsymbol{\alpha}_j(i) = \boldsymbol{\alpha}_j(i) + \eta e(i)$,
where $j = \arg \min_{1 \leq j \leq size(C(i-1))} \|u(i) - C_j(i-1)\|$

Go to step **1)**

5) Update $n = n + 1$ and compute $k = mod(n-1, M)$

6) If k is equal to zero, then $\beta(i) = 0$

7) Otherwise: $\beta(n) = \frac{\kappa(g(n),g(n))}{\kappa(g(n-1),g(n-1))}$

8) Store a new center $C(i) = C(i-1), u(n)$,
and update $\boldsymbol{\alpha}(i)$ using (10) or (12).

End while

3. SIMULATION RESULTS

3.1. Mackey-Glass Time Series Prediction

In the first example, we show the performance of KLMS-CG and QKLMS-CG in short-term chaotic time series prediction. Consider the Mackey-Glass (MG) time series generated by the following time-delay ordinary differential equation:

$$\frac{dx(t)}{dt} = -bx(t) + \frac{ax(t-\tau)}{1 + x(t-\tau)^{10}} \tag{13}$$

with $b = 0.1, a = 0.2, \tau = 30$. The above equation displays the characteristics of the periodic and chaotic dynamics, which is a benchmark problem for nonlinear learning. Our goal is to predict the current value of the time series using the past 10 samples. The input vector of the adaptive filter is thus $u(k) = [x(k-1), \ldots, x(k-10)]^T$. Simulation results in this example are averaged over 100 independent Monte Carlo runs, and in each Monte Carlo run, a segment of 1500 samples is used as the training data and another 100 as the test data. The training data are corrupted by a Gaussian noise with zero mean and variance 0.02. The Gaussian kernel is used and the kernel width is set at $h = 1.0$.

Fig.1 shows the convergence curves of the KLMS-CG with different M values. As one can see clearly, if M is too small (say, $M = 2$) or too large (say, $M = 12$), the convergence performance will become worse. In this example, the algorithm achieves the best balance between convergence speed and steady-state accuracy when $M = 10$. Without mentioned otherwise, this value will be used in the rest of simulations. Fig.2 illustrates the performance comparison between KLMS, KLMS-CG, QKLMS and QKLMS-CG.

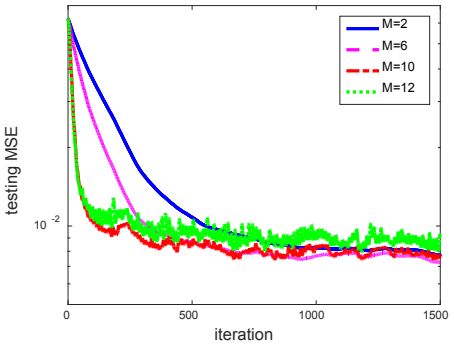

Fig. 1. Convergence curves of KLMS-CG with different M values.

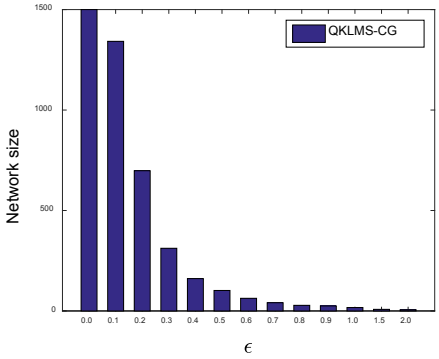

Fig. 3. Network sizes of QKLMS-CG with different quantization factors.

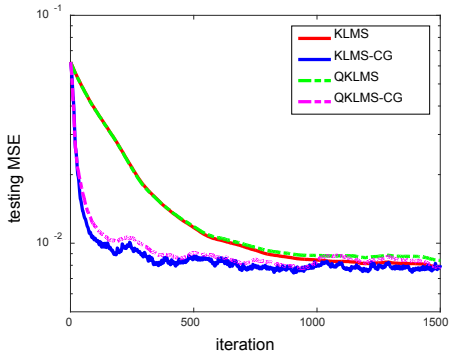

Fig. 2. Convergence curves of KLMS, KLMS-CG, QKLMS and QKLMS-CG.

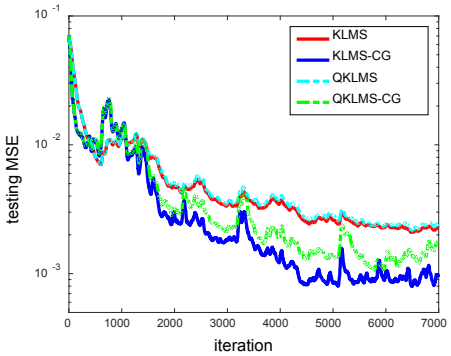

Fig. 4. Convergence curves of KLMS, KLMS-CG, QKLMS and QKLMS-CG.

The parameters are chosen such that the four algorithms yield almost the same steady-state MSE. The quantization factor is set at $\varepsilon = 0.4$. From the simulation result, one can observe that KLMS-CG and QKLMS-CG converge much faster than the original KLMS and QKLMS, while QKLMS-CG can achieve almost the same performance as that of KLMS-CG. Fig.3 shows the network sizes (at the final iteration) of the QKLMS-CG with different quantization factors. As expected, with the quantization factor increasing, the final network size will decrease significantly.

3.2. Methuselah (tree) Walk Data

In the second example, we use a real-world data set from the Time Series Data Library[1]. The methuselah walk data is over the period $-6000 \sim 1979$. The task is to predict the current sample using the past 10 samples, just like the previous example. The methuselah walk data over the period $-6000 \sim 1000$ are used as the training data and the data over the period $1001 \sim 1100$ are used as the test data. Fig.4 shows the convergence curves of KLMS, KLMS-CG, QKLMS and QKLMS-CG. The related parameters are set as: $M = 8, \varepsilon = 0.2, h = 1.0$. Again, the KLMS-CG and QKLMS-CG outperform the KLMS and QKLMS respectively.

[1]URL: https://datamarket.com/data/list/?q=provider:tsdl

4. CONCLUSION

In this paper, Kernel least mean square based on conjugate gradient (KLMS-CG), and its quantized version called QKLMS-CG, are developed to solve efficiently the nonlinear least square regression in a sequential manner. The KLMS-CG employs orthogonal search directions to find the solution instead of using the steepest descent approach as in the original KLMS, hence achieves a much faster convergence speed. Simulation results demonstrate the excellent performance of the proposed method.

There are some problems that need to be studied in the future. The most important is how to choose the frequency to reset the search direction or to terminate the CG algorithm. It is also promising to apply the conjugate gradient methods to other kernel adaptive filtering algorithms, such as the kernel affine projection algorithm (KAPA).

5. REFERENCES

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

[2] J. Xu, Paiva, A.R.C, Park, J.C. Principe, "A Reproducing Kernel Hilbert Space Framework for Information-Theoretic Learning," *IEEE Trans. on Signal Processing*, vol. 56, no. 12, pp. 5891-5902, 2008.

[3] W. Liu, P. Pokharel, J.C. Principe, "The kernel least mean square algorithm," *IEEE Trans. on Signal Processing*, vol. 56, no. 2, pp. 543-554, 2008.

[4] W. Liu and J.C. Principe, "Kernel affine projection algorithm," *EURASIP J. Adv. Signal Process.*, vol. 1, pp. 1-13, 2008.

[5] Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least squares algorithm," *IEEE Trans. on Signal Process.*, vol. 52, no. 8, pp. 2275-2285, 2004.

[6] W. Liu, Park, Y. Wang, J.C. Principe, "Extended kernel recursive least squares algorithm," *IEEE Trans. on Signal Processing*, vol. 57 pp. 3801-3814, 2009.

[7] B. Widrow, J.M. Cool, "A comparison of adaptive algorithms based on the methods of steepest descent and random search," *IEEE Antennas and Propagation*, vol. 24, no. 5, pp. 615-637, 1976.

[8] R. Battiti, "First and second order methods for learning: Between steepest descent and Newtons method," *Neural Computation*, vol. 4, no. 2, pp. 141-166, 1992.

[9] R. Fletcher and C.M. Reeves, "Function minimization by conjugate gradients," *Computer Journal*, vol. 7, pp. 149-154, 1964.

[10] M.R. Hestenes, E. Steifel, "Method of conjugate gradient for solving linear equations," *J. Res. Nat. Bur. Stand.*, vol. 49, pp. 409-436, 1952.

[11] E. Polak, G. Ribiere, "Note sur la convergence de directions conjugees," *Rev. Francaise Informat. Recherche Operationelle*, vol. 16 , pp. 35-43, 1969.

[12] Y. Liu and C. Storey, "Efficient generalized conjugate gradient algorithms part 1: Theory," *Journal of Optimization Theory and Application*, vol. 69, pp. 129-137, 1992.

[13] Y.H. Dai and Y. Yuan, "A Nonlinear Conjugate Gradient Method with a Strong Global Convergence Property," *SIAM J. Optim.*, vol. 10, no. 1, pp. 177-182, 1999.

[14] M.J.D. Powell, "Convergence properties of algorithm for nonlinear optimization," *SIAM Rev.*, vol. 28, pp. 487-500, 1986.

[15] M. Al-Baali, "Descent property and global convergence of Fletcher-Reeves method with inexact line search," *IMA. J. Numer. Anal.*, vol. 5 pp. 121-124, 1985.

[16] B. Chen, S. Zhao, P. Zhu, J.C. Principe, "Quantized kernel least mean square algorithm," *IEEE Trans. on Neural Networks and Learning Systems*, vol. 23, no. 1, pp. 22-32, 2012.

[17] B. Chen, S. Zhao, P. Zhu, J.C. Principe, "Quantized kernel recursive mean square algorithm," *IEEE Trans. on Neural Networks and Learning Systems*, vol. 24, no. 9, pp. 1484-1491, 2013.

[18] X. Xu, H. Qu, J. Zhao, X. Yang, B. Chen, "Quantized kernel least mean square with desired signal smoothing," *Electronics Letters*, vol. 51, no. 18, pp. 1457-1459, 2015.

[19] S. Zhao, B. Chen, Z. Cao, P. Zhu, J.C. Principe, "Self-Organizing Kernel Adaptive Filtering," *EURASIP Journal on Advances in Signal Processing*, 2016(1), 2016: 106.

[20] S. Zhao, B. Chen, J.C. Principe, "Fixed budget quantized kernel least mean square algorithm," *Signal Processing*, vol. 93, pp. 2759-2770, 2013.

[21] S. Nan, L. Sun, B. Chen, Z. Lin, K. Toh, "Density-dependent quantized least squares support vector machine for large data sets," *IEEE Trans. on Neural Networks and Learning Systems*, vol. PP, no. 99, pp. 1-13, 2015.

[22] G.K. Boray and M.D. Srinath, "Conjugate Gradient Techniques for Adaptive Filtering," *IEEE Transactions on Circuits and Systems*, vol. 39 , no. 1, pp. 1-10, 1992.

[23] P.S. Chang, Willson, A.N. Jr. "Analysis of conjugate gradient algorithms for adaptive filtering," *IEEE Trans. on Signal Processing*, vol. 48, no. 2, pp. 409-418, 2000.