Circuits, Systems, and Signal Processing (2021) 40:4346–4369
https://doi.org/10.1007/s00034-021-01691-z

Adaptive Sparse Quantization Kernel Least Mean Square Algorithm for Online Prediction of Chaotic Time Series

Chaochao Zhao[1] · Weijie Ren[2] · Min Han[3]

Received: 20 July 2020 / Revised: 25 February 2021 / Accepted: 1 March 2021 / Published online: 9 April 2021

Abstract

Kernel least mean square (KLMS) algorithm is a popular method for time series online prediction. It has the advantages of good robustness, low computational complexity, model simplicity and online learning ability. Unfortunately, as input data grows, the dictionary size increases and the computational complexity raises significantly. In addition, how to improve the adaptability in time-varying environments with noise is also one of the main challenges. Therefore, we propose an improved KLMS algorithm from sparse perspective in response to the above problems, called adaptive sparse quantization kernel least mean square (ASQ-KLMS) algorithm. In the new model, sequential outlier criterion for sparsification and weights adaptive adjustment are combined with coherence criterion and quantization to form ASQ-KLMS algorithm. Firstly, it makes full use of effective information and ignores the interference of abnormal information to obtain a more accurate and compact dictionary. Then, a good balance between algorithm efficiency and accuracy can be achieved by controlling the choice of parameters. In addition, it can adaptively adjust weights in time-varying environment. At last, the Lorenz chaotic time series, the ENSO chaotic time series and the Beijing $PM_{2.5}$ chaotic time series are used to prove the reliability of the ASQ-KLMS algorithm.

Keywords Kernel least mean square (KLMS) · Time series · Sparsification · Adaptive

✉ Min Han
minhan@dlut.edu.cn

Chaochao Zhao
cczhao@mail.dlut.edu.cn

Weijie Ren
renweijie90@126.com

[1] Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology, Dalian 116024, China

[2] College of Intelligent Systems Science and Engineering, Harbin Engineering University, Harbin 150001, China

[3] Key Laboratory of Intelligent Control and Optimization for Industrial Equipment of Ministry of Education, Dalian University of Technology, Dalian 116024, China

1 Introduction

Online prediction of chaotic time series has been widely applied to various fields, including medicine systems, biomedicine, environment, engineering [4,11,13,26,32], etc. In practical prediction applications, most time series are nonlinear and require online processing, such as stock trend forecasting, real-time traffic flow prediction, and real-time monitoring of air quality. This real data generally contains noise. Therefore, it is highly desirable to have an anti-noise online prediction algorithm that has fast and accurate computing ability.

In the time series online prediction field, some methods based on nonlinear models, for example, dynamic neural networks [29,39], online support vector regression (Online SVR) [14] and kernel adaptive filter (KAF) algorithms [2,18,34] have been proposed. Among these methods, dynamic neural network can improve the offline model network structure, and achieve dynamic update, but it is computationally intensive and easy to fall into local optimum. Online SVR and KAF algorithms are the classical kernel methods, which has the advantages of the kernel method and can handle time series prediction problems with nonlinear characteristics. The kernel method is an effective method to solve the nonlinear time series prediction problem. The Mercer core in reproducing kernel Hilbert space (RKHS) [1] is continuous, positive definite and reproducible which are desirable properties. Based on these properties, we cannot update the weight vector (it is also uncalculated in high-dimensional space), and use the Mercer kernel to expand the kernel function mapped to the high-dimensional space into the form of eigenvalues and eigenvectors. The output after mapping to the RKHS space is obtained, thereby realizing the combination of the kernel method and the SVR or the adaptive filter. However, it is necessary to continuously add new samples and update the model in online prediction of chaotic time series. The ever-increasing sample size and the amount of computation required for model updates pose significant challenges to the computational efficiency of online kernel methods.

As we know, Liu et al. [18] make a comprehensive introduction to the KAF algorithms. The KAF algorithms use a linear adaptive filtering algorithm in RKHS to implement nonlinear signal processing [12]. The KAF algorithms can adaptively update the model and are applied to many areas [40]. When new samples are acquired, it can be updated in recursive increments for online prediction. In addition, it solves nonlinear problems by using kernel mapping [6], which reduces computational complexity and can effectively solve time series online prediction problem with nonlinear characteristics. The KAF algorithms generally include KLMS [16] algorithm, kernel affine projection algorithm (KAPA) [17], kernel recursive least squares (KRLS) [5], and so on. The KAF algorithms are suitable for small samples, nonlinear modeling, and online recursive updating. However, it is necessary to continuously add new samples and update the model in online prediction of chaotic time series. The amount of calculation raises significantly when the sample size and the nuclear matrix dimension increase.

Among them, the KLMS algorithm is a typical kernel adaptive filter algorithm that Liu et al. proposed [16]. The KLMS algorithm has the advantages of good computational robustness and model simplicity. It has a good application prospect for online prediction of chaotic time series. But one of its main challenges is that the size of dictionary and the computational complexity increases dramatically with input data increasing [10]. If the real data contains outliers as training data, this will greatly reduce the accuracy of the prediction [21,23]. Therefore, the ability to track the time-varying characteristics of the time series and the anti-noise ability in the dynamic environment needs to be improved.

Based on the above discussions, we improve the KLMS algorithm from the sparse perspective. In this paper, first, the outlier is eliminated using sequential outlier criterion to prevent it from being accepted as a new center, and then, the coherence criterion is added to the quantized kernel least mean square (QKLMS) [3] algorithm. The quantization method establishes a standard that distinguishes whether the data is informative or redundant. Redundant data is quantified to its nearest center. It can be said that we have fully explored the informative nature of the data. It also reduces the size of the dictionary, which decreases the complexities of computation. Moreover, a good trade-off between algorithm efficiency and accuracy is achieved by controlling the choice of parameters. In addition, we consider adding weight adaptive adjustment to the algorithm to improve the noise immunity of KLMS in the time-varying environment. In this process, the weight coefficients is adaptively adjusted, which obviously improves the dynamic tracking ability. Finally, the validity of the algorithm is verified in the Lorenz, ENSO and Beijing $PM_{2.5}$ chaotic time series for online prediction.

The remainder of this paper is structured as follows. Section 2 presents the relevant research progress of KLMS and sparse methods, and points out the current problems. Section 3 briefly introduces the KLMS algorithm and quantization method. Section 4 gives the detailed introduction and explanation of the relevant parts of the algorithm and the ASQ-KLMS algorithm. Section 5 presents the simulation results for online prediction of chaotic time series to demonstrate the effectiveness of ASQ-KLMS algorithm. Section 6 provides a conclusion and points out the prospects for further research.

2 Background and Related Works

2.1 Chaotic Time Series and Applications

Chaos refers to the uncertain signal generated in the deterministic system, which is similar to the random signal to a certain extent. Chaos is not simply disorder, but has no obvious periodicity and symmetry, but it is an ordered structure with rich internal levels, and is a new form of existence in nonlinear systems. The discrete situation of chaos is often expressed as chaotic time series. In chaotic time series modeling research, phase space reconstruction is a key step in system modeling [9]. According to the Takens embedding theorem [22], the chaotic dynamic system can be reconstructed from the time series and reproduce the evolution information of the chaotic system, providing strong support for accurate prediction.

Modeling and forecasting chaotic time series with multiple variables, noise and non-stationary is a current research hotspot. For chaotic time series containing multivariate, Zhang et al. [35] used principal component analysis (PCA) to perform dimensionality reduction and feature extraction on 13 local physical factors, and used the reduced fusion features as the input of a multi-layer perceptron to achieve a wide range of short-term rainfall predictions. For noisy chaotic time series, Lu et al. [20] constructed a new robust loss function by considering the mean and variance of the sample, which increases the noise immunity of the model to a certain extent. For non-stationary signals, Sharma et al. [27] proposed a Hank matrix-based eigenvalue decomposition, which can decompose the positive and negative frequency components of complex signals separately, and its performance is better than the comparison method.

2.2 Online Sparse Methods

In the past ten years, many scholars have proposed several improved KLMS algorithms and applied them to the field of time series online prediction. In 2008, C. Richard et al. [25] first added the normalization criterion into the KLMS algorithm, effectively reducing the size of the learning dictionary. And since then, Chen et al. [3] used the quantization method to reduce the input space dimension, and proposed the QKLMS algorithm in 2012. In the same year, M. Yukawa et al. [34] studied the multi-core algorithm and added a weighted norm penalty to the original. A sparse multi-nuclear normalized least mean square algorithm based on coherent criteria was proposed. Chen et al. [37] proposed the QKLMS algorithm based on fixed memory budget (FB-QKLMS) in 2013. It adds a rejection mechanism that limits the size of the dictionary and enables tracking time-varying features. In 2014, Gao et al. [8] proposed a kernel least mean square algorithm with the norm regularization FOBOS-KLMS algorithm. Then, Van Vaerenbergh et al. [7] proposed the probability least mean square (Prob-LMS), which adds Bayesian machine learning to adaptive filtering. Zhao et al. [38] simplified the normalized variable step size and proposed a decorrelated kernel least mean square (DKLMS) algorithm in 2017. Zhang et al. [36] proposed a sequential outlier criterion to limit the interference of outliers in 2018. This method gives two conditions for judging whether the information is an abnormal value, and improves the anti-noise ability of the system. In 2018, Liu et al. [19] proposed the reduced Gaussian kernel least mean square (RGKLMS) algorithm, which avoids the continuous growth of weighted networks in non-stationary environments through implicit feature maps. In 2019, the kernel least mean square based on the Nystrom method (NysKLMS) proposed by Wang et al. [33] can significantly reduce the computational complexity.

In addition, some online sparse methods have been proposed to suppress the dictionary growth in the learning process, for example, an algorithm with a fixed dictionary size is studied in [28,30]. The novelty criterion (NC) proposed in [24] proposes two constraints to suppress the growth of the dictionary. An approximate linear dependency (ALD) is proposed in [5] to verify the linear dependency between the new data and the elements in the dictionary, and if the approximation is irrelevant, the input is added to the dictionary. The coherence criterion (CC), which limits the size of the dictionary based on the similarity of the input, has been described in [25]. In the online sparse method based on the surprising standard (SC) [15] of Gaussian process, the information content of samples is quantified as a surprise, which can prevent outliers from being used as new centers, thereby a more reliable network will be obtained. Chen et al. [3] used quantization conditions to determine and quantify redundant data

Table 1 The explanation of the parameters

Parameter	Explanations
μ_0	Threshold for coherence criterion
ε	Quantization threshold
η	The step size
ζ	The parameters dynamic adjustment
λ_d	The distance region coefficient
λ_v	The vector region coefficient

for network coefficient updates, thus taking advantage of more efficient information. Sequential outlier criterion (SOC) established the measurement of sample information based on the historical data sequence information, and then, the learning system processes the information measurement samples according to the sample information, redundancy or exception.

However, the form of the above sparse methods is very simple, only one aspect is considered, and the characteristics of the nonlinear system cannot be described comprehensively. Therefore, there is still room for improvement in these algorithms. In addition, KLMS and the extended algorithm based on sparse method are very sensitive to time-varying environments with noise, so it needs to improve the anti-noise ability of the model.

In the above-mentioned sparse methods, we find that the SOC criterion showed good ability to exclude outliers. KLMS-CC [25] and QKLMS show excellent predictive power, efficiency and nonlinear fitting ability. Combining the advantages of the above sparse methods, the ASQ-KLMS algorithm is proposed for online prediction of chaotic time series.

Table 1 shows an explanation of the parameters used in the ASQ-KLMS algorithm.

3 Preliminary

3.1 Kernel Least Mean Square

The KLMS algorithm is suitable for online learning. It is a generalized and evolved form of the traditional least-mean-square (LMS) method in kernel space. The KLMS algorithm maps the input into a high-dimensional feature space by introducing a kernel function based on this linear method, retains the nonlinear characteristics of the data, and then uses a simple linear LMS method for online prediction in the kernel space.

$\mathbf{X}$ is the input data, then there is a nonlinear mapping $\Omega : \mathbf{X} \rightarrow \mathbf{F}$, the original data is mapped to the feature space and then the LMS algorithm linearly filters the converted data $\Omega(\mathbf{x})$, $\mathbf{x} \in \mathbf{X}$. When the time step is n, the system output is expressed as

$$\mathbf{y}(n) = \boldsymbol{\alpha}_{n-1}{}^{T}\Omega(\mathbf{x}(n)) \tag{1}$$

where $\mathbf{x}(n) \in \mathbf{R}^M$ is the input vector, $\mathbf{y}(n) \in \mathbf{R}$ is the output of the system, $\boldsymbol{\alpha}_n$ is the weight vector.

In the LMS method, the update of the weight vector can be calculated by

$$\boldsymbol{\alpha}_n = \boldsymbol{\alpha}_{n-1} + \eta \cdot \mathbf{e}\,(n)\,\Omega\,(\mathbf{x}\,(n)) \tag{2}$$

where $\mathbf{e}$ means the prediction error at the n-th iteration, η represents the step size, and $\Omega\,(\mathbf{x}(n))$ represents the conversion of the input $\mathbf{X}$ to the feature space $\mathbf{F}$. The weight vector $\boldsymbol{\alpha}_n$ can be represented by the following equation:

$$\boldsymbol{\alpha}_n = \eta \sum_{i=1}^{n} \mathbf{e}\,(i)\Omega\,(\mathbf{x}\,(i)) \tag{3}$$

Nonlinear mapping can be implemented by means of kernel functions.

$$\boldsymbol{\kappa}\,(\mathbf{x}\,(i)\,,\mathbf{x}\,(j)) = \Omega(\mathbf{x}\,(i))^T\,\Omega\,(\mathbf{x}\,(j)) \tag{4}$$

Therefore, the filter output of KLMS is given by

$$\mathbf{y}\,(n) = \eta \sum_{i=1}^{n-1} \mathbf{e}\,(i)\,\boldsymbol{\kappa}\,(\mathbf{x}\,(i)\,,\mathbf{x}\,(n)) \tag{5}$$

Kernel functions [31] provide a convenient way to solve complex nonlinear regression problems. We choose the Gaussian kernel as the kernel function to complete the output calculation. It is expressed as

$$\boldsymbol{\kappa}\,(\mathbf{x}\,(i)\,,\mathbf{x}\,(j)) = exp\left(-\frac{\|\mathbf{x}\,(i) - \mathbf{x}\,(j)\|^2}{2\sigma^2}\right) \tag{6}$$

where σ denotes the kernel width.

3.2 Quantization

The quantization method represents the original input data using a smaller body. It does not directly discard the judged redundant input by the sparse correlation condition, but uses the quantization condition to judge, and quantizes the redundant variable for the network coefficient update, thereby utilizing more valid information. The objective function of QKLMS is defined as

$$\min \sum_{i=1}^{n} \left|\mathbf{y}\,(i) - \boldsymbol{\kappa}^T\,(\mathbf{B}\,(i-1)\,,\mathbf{x}\,(i)) \cdot \boldsymbol{\alpha}_i\right| \tag{7}$$

where $\boldsymbol{B}\,(\cdot)$ is the model dictionary. At the n-th iteration,$\boldsymbol{B}\,(n) = \mathrm{O}\,[\mathbf{x}\,(n)]$,$O\,[\cdot]$ is a quantization operator.

Chen et al. [3] proposed a simple online vector quantization method that satisfies the requirements of online prediction. They calculate the distance between $\mathbf{x}(n)$ and the dictionary $\mathbf{B}(n-1)$:$dis(\mathbf{x}(n), \mathbf{B}(n-1)) = \min_{1 \le j \le size(\mathbf{B}(n-1))} \left\| \mathbf{x}(n) - \mathbf{B}_j(n-1) \right\|$, where j means the j-th member of the dictionary $\mathbf{B}(n-1)$, $\|\cdot\|$ denotes the Euclidean norm. If the calculated Euclidean distance is less than the set quantization parameter, the dictionary is not updated, $\mathbf{B}(n) = \mathbf{B}(n-1)$, and the input data will be quantized to the $\mathbf{B}_j(n-1)$. Otherwise, we will update the dictionary: $\mathbf{B}(n) = \mathbf{B}(n-1) \cup \mathbf{x}(n)$, and it will become a new center in the existing dictionary.

Finally, the operation process of the QKLMS algorithm is described in detail in Algorithm 1.

Algorithm 1: QKLMS

```
while $n = 1, 2, \cdots$ do
    $\{\mathbf{x}(n), \mathbf{y}(n)\}$ is available,
    calculate the distance between $\mathbf{x}$ and the dictionary $\mathbf{B}(n-1)$: $dis(\mathbf{x}(n), \mathbf{B}(n-1))$;
    if $dis(\mathbf{x}(n), \mathbf{B}(n-1)) > \varepsilon$ then
        $\mathbf{x}(n)$ becomes a new center:
        $\mathbf{B}(n) = \mathbf{B}(n-1) \cup \mathbf{x}(n)$,
        and the coefficient will be updated as: $\boldsymbol{\alpha}_n = \left[\boldsymbol{\alpha}_{n-1}, \eta \cdot \mathbf{e}(n)\right]$.
    else
        dictionary does not change:
        $\mathbf{B}(n) = \mathbf{B}(n-1)$,
        and quantize $\mathbf{x}(n)$ to the j-th center: $\mathbf{B}_j(n) = \mathrm{O}[\mathbf{x}(n)]$,
        and the coefficient will be updated as: $_j = _j + \eta\left(\mathbf{y}(n) - \check{}^T_{n-1}{}_{n-1}\right)$.
    end
    Output: $\mathbf{y}(n+1)$
end
```

4 Adaptive Sparse Quantization Kernel Least Mean Square Algorithm

In this section, we will introduce the reserve knowledge about our improved algorithm, and the ASQ-KLMS algorithm from the sparse perspective will be described in detail at the end.

4.1 Sequential Outlier Criterion for Sparsification

Generally, there are often some outliers in the actual data. When these outliers are used to update the network coefficients, the prediction accuracy will be seriously affected.

Therefore, Zhang et al. [36] proposed an online sparsification approach with sequential outlier criterion. Its major contribution is to propose conditions for checking whether new arriving data is an outlier. Based on historical data, there will be a range of expectations for new arriving datum. If this range is exceeded, we think the data is outlier.

The expected neighbor distance (END) and the expected neighbor direction vector (ENDV) are measures, which are used for determining whether data is an outlier or not. Given the information of all existing data, the END infers the reasonable distance of the neighbor data, and the ENDV infers the possible direction of the oriented data.

In the course of sequential learning, the END infers the expected distance from adjacent data. When an outlier occurs, the distance between the outlier and the previous data must be very large. Therefore, we use the END as one of the criteria to judge whether the data is an outlier or not. It is defined as

$$ND_e(n) = \sum_{i=1}^{n-2} \frac{2i}{(n-2)(n-1)} \cdot d\langle \mathbf{x}(i+1), \mathbf{x}(i)\rangle \tag{8}$$

where $d\langle \mathbf{x}(i+1), \mathbf{x}(i)\rangle = \|\mathbf{x}(i+1) - \mathbf{x}(i)\|$, and the main function of the coefficient $2i/(n-2)(n-1)$ is to give greater discourse power to the distance closer to the new data. The upper and lower bounds of the END for the n-th iteration are represented as

$$\begin{cases} B\_D_{\text{upper}}(n) = ND_e(n) + \lambda_d \sigma_d(n) \\ B\_D_{\text{lower}}(n) = ND_e(n) - \lambda_d \sigma_d(n) \end{cases} \tag{9}$$

where λ_d has different sizes according to different data sets. Its size determines the severity of the outlier determination conditions. $\sigma_d(n)$ is the standard deviation of $\{d\langle \boldsymbol{x}(i+1), \boldsymbol{x}(i)\rangle, i = 1, 2, ..., i-2\}$.

The ENDV establishes a historical vector trend based on the already existing data. When an outlier occurs, the difference between adjacent data vectors must be large. It is another criteria for judging whether the data is an outlier or not. The ENDV is described as

$$\mathbf{NV}_e(n) = \sum_{i=1}^{n-2} \frac{2i}{(n-2)(n-1)} \cdot \frac{[\mathbf{x}(i+1) - \mathbf{x}(i)]}{d\langle \mathbf{x}(i+1), \mathbf{x}(i)\rangle} \tag{10}$$

where the coefficient $2i/(n-2)(n-1)$ is to give greater discourse power to the vector closer to the new data. The upper and lower bounds of the ENDV for the n-th iteration are represented as

$$\begin{cases} \mathbf{B\_V}_{\text{upper}} = \mathbf{NV}_e(n) + \lambda_v \boldsymbol{\sigma}_v(n) \\ \mathbf{B\_V}_{\text{lower}} = \mathbf{NV}_e(n) - \lambda_v \boldsymbol{\sigma}_v(n) \end{cases} \tag{11}$$

where λ_v has different sizes according to different data sets. Its size determines the severity of the outlier determination conditions. $\boldsymbol{\sigma}_v(n)$ is the standard deviation of $\left\{[\mathbf{x}(i+1) - \mathbf{x}(i)] \big/ d\langle \mathbf{x}(i+1), \mathbf{x}(i)\rangle, i = 1, 2, ..., n-2\right\}$.

If the n-th data exceeds the specified range of END and ENDV, it will be considered as an outlier.

4.2 Coherence-sparsification Criterion

Correlation is a fundamental and effective indicator for detecting linear sparse approximations. In the past research, the dimensionality reduction of the kernel matrix by ALD conditions will be misjudged due to the inappropriate selection of thresholds, thus affecting the prediction effect. After that, the coherence coefficient was proposed as a standard by Richard et al. [25] in 2008. The coherence coefficient is written as

$$\mu = \max_{i=1,2,\ldots,n} \left|\boldsymbol{\kappa}\left(\mathbf{x}\left(n\right), \mathbf{x}\left(i\right)\right)\right| \leq \mu_0 \tag{12}$$

where μ_0 determines the size and sparseness of the sample dictionary, $\mu_0 \in [0, 1]$. The calculation cost of the coherent criterion is not affected by the sample data, only depends on the size of the final dictionary. The calculation cost is small, and it can better measure the relationship between the input sample and the original sample dictionary; thus, the judgment of the input sample is more accurate.

4.3 Weight Adaptive Adjustment

If the input contains noise or sudden changes, or other factors that cause κ_{n-1} to be very small, after updating α_n, the effect will be amplified, even catastrophic. Therefore, we use the normalization strategy of Euclidean norm to improve the system adaptive ability. The weight can be updated:

$$\boldsymbol{\alpha}_n = \boldsymbol{\alpha}_{n-1} + \frac{\eta \cdot \left(\mathbf{y}\left(n\right) - \boldsymbol{\kappa}_{n-1}^T \boldsymbol{\alpha}_{n-1}\right) \cdot \boldsymbol{\kappa}_{n-1}}{\zeta + \left\|\boldsymbol{\kappa}_{n-1}^T\right\|^2} \tag{13}$$

where $\boldsymbol{\kappa}_{n-1} = \boldsymbol{\kappa}\left(\mathbf{B}\left(n-1\right), \mathbf{x}\left(n\right)\right)$ and ζ is a user-defined parameter whose purpose is to avoid the denominator being zero. This approach can significantly improve the ability of sequential learning systems to adapt to different data sets, especially those that contain noise and mutations.

4.4 ASQ-KLMS Algorithm

In order to achieve good performance, we combine the sequential outlier criterion, coherence-sparsification criterion and dynamic adjustment of weights with the quantization to form the ASQ-KLMS algorithm.

First, we use the sequential outlier criterion to determine whether the input is an outlier or not. If the current input data does not meet our given expected range, it will be discarded. Next, we combine quantization method with coherence criterion to determine whether input data is cached or not. The input data is cached when the current correlation coefficient does not exceed the threshold and the quantization condition is met. This greatly reduces the size of the dictionary, which helps to reduce the amount of calculations and speed up the calculation.

As we know, redundant data is fully utilized with quantization method, so that it still participates in the update of the weight coefficient. In addition, weight adaptive adjustment can improve the ability of KLMS to track time-varying features. This method can adaptively adjust coefficients according to input data, and improve the prediction accuracy in chaotic time online prediction.

The ASQ-KLMS algorithm is obtained from the sparse perspective, which can reduce computational complexity by limiting matrix dimensions and make full use of redundant information, so it can improve the calculation efficiency and prediction accuracy.

The specific steps of the ASQ-KLMS algorithm are described as follows:

First, calculate Eqs. (9) and (11). Then, we will make the following judgment.

Case 1: If the neighbor distance and direction vector are outside the corresponding range, we will assume that the data is an outlier, then it will be discarded and we will continue to detect the next data.

Case 2: If $dis\left(\mathbf{x}\left(n\right),\mathbf{B}\right) > \varepsilon$ and the coherence coefficient is less than threshold μ_0, then it will become a member of the dictionary and become a new center. And update weight coefficient $\boldsymbol{\alpha}_n$ using Eq. (14).

Case 3: If $dis\left(\mathbf{x}\left(n\right),\mathbf{B}\right)$ is no more than ε, we suppose that this data is redundant, then the dictionary will not change: $\mathbf{B}\left(n\right) = \mathbf{B}\left(n-1\right)$. And $\mathbf{x}\left(n\right)$ will be quantized to the j-th center: $\mathbf{B}_j\left(n\right) = \mathrm{O}\left[\mathbf{x}\left(n\right)\right]$. And $\boldsymbol{\alpha}_n$ is updated as (15).

Case 4: If none of the above three conditions are met, it means that the input data is linearly related to the data in the current dictionary. So, it is considered as redundant data and discarded, and keep the existing dictionary unchanged. In this case, the weight coefficient will be dynamically adjusted and it will be calculated as shown in (16).

Finally, the output is calculated using Eq. (17).

In summary, the operation process of the ASQ-KLMS algorithm is described in detail in Algorithm 2.

4.5 Computational Complexity

Table 2 summarizes the computational complexity analysis of KLMS, QKLMS, KLMS-CSL1, KLMS-SOC and ASQ-KLMS algorithms, including the time complexity of updating $\boldsymbol{\alpha}_n$ during training and computing $\mathbf{e}\left(n\right)$ during testing. For KLMS, QKLMS and KLMS-CSL1, the time complexity of updating $\boldsymbol{\alpha}_n$ during training data is $O\left(n\right)$, and the time complexity of calculating $\mathbf{e}\left(n\right)$ during testing is $O\left(n\right)$, $O\left(m\right)$, $O\left(l\right)$, respectively, where n is the size of the dictionary when the model is not sparsified. m and l are close, but both are smaller than n. For KLMS-SOC and ASQ-KLMS, the time complexity of updating $\boldsymbol{\alpha}_n$ during training data is $O\left(n^2\right)$, and the time complexity of calculating B during testing is $O\left(k\right)$ and $O\left(k\right)$, respectively, where k is close to m and l, but k is much smaller than n.

Algorithm 2: ASQ-KLMS

```
parameters: $\eta, \varepsilon, \mu_0, \lambda_d, \lambda_v$,
initialize: $\boldsymbol{\alpha}_1 = \eta \cdot \mathbf{y}(1)$;
while $n = 1, 2, \cdots$ do
    $\{\mathbf{x}(n), \mathbf{y}(n)\}$ is available
    Compute the standard deviation of $d\langle \mathbf{x}(i), \mathbf{x}(i+1)\rangle$ and
    $[\mathbf{x}(i+1) - \mathbf{x}(i)]$,$i \in \{1, 2, ..., n-2\}$, $\sigma_d(n)$ and $\boldsymbol{\sigma}_v(n)$
    Compute the expected neighbor distance $ND_e(n)$ using (8)and the expected neighbor direction
    vector $\mathbf{NV}_e(n)$ using (10) and their corresponding ranges:
    $\left[ND_e(n) - \lambda_d\sigma_d(n), ND_e(n) + \lambda_d\sigma_d(n)\right]$
    $[\mathbf{NV_e}(n) - \lambda_v\boldsymbol{\sigma}_v(n), \mathbf{NV_e}(n) + \lambda_v\boldsymbol{\sigma}_v(n)]$
    Compute the neighbor distance $d\langle \mathbf{x}(n), \mathbf{x}(n-1)\rangle$ and the neighbor direction vector
    $[\mathbf{x}(n+1) - \mathbf{x}(n)]/d\langle \mathbf{x}(n+1), \mathbf{x}(n)\rangle$ for the new arriving datum.
    if $d\langle \mathbf{x}(n), \mathbf{x}(n-1)\rangle \notin \left[ND_e(n) - \lambda_d\sigma_d(n), ND_e(n) + \lambda_d\sigma_d(n)\right]$ and
    $[\mathbf{x}(n+1) - \mathbf{x}(n)]/d\langle \mathbf{x}(n+1), \mathbf{x}(n)\rangle \notin [\mathbf{NV_e}(n) - \lambda_v\boldsymbol{\sigma}_v(n), \mathbf{NV_e}(n) + \lambda_v\boldsymbol{\sigma}_v(n)]$ then
        $\mathbf{B}(n) = \mathbf{B}(n-1)$, $\mathbf{x}(n)$ is discarded.
    else if
        $$\max_{i=1,2,...,n} |\boldsymbol{\kappa}(\mathbf{x}(n), \mathbf{x}(i))| \le \mu_0$$
    and $dis(\mathbf{x}(n), \mathbf{B}) > \varepsilon$ then
        $\mathbf{B}(n) = \mathbf{B}(n-1) \cup \mathbf{x}(n)$,
        $$\boldsymbol{\alpha}_n = \begin{bmatrix} \boldsymbol{\alpha}_{n-1} \\ \eta \cdot \left(\mathbf{y}(n) - \boldsymbol{\kappa}_{n-1}^T \boldsymbol{\alpha}_{n-1}\right) \end{bmatrix} \quad (14)$$
    else if $dis(\mathbf{x}(n), \mathbf{B}) \le \varepsilon$ then
    $\mathbf{B}(n) = \mathbf{B}(n-1)$,
        $$\boldsymbol{\alpha}_j = \boldsymbol{\alpha}_j + \eta\left(\mathbf{y}(n) - \boldsymbol{\kappa}_{n-1}^T \boldsymbol{\alpha}_{n-1}\right) \quad (15)$$
    else
    $\mathbf{B}(n) = \mathbf{B}(n-1)$,
        $$\boldsymbol{\alpha}_n = \boldsymbol{\alpha}_{n-1} + \frac{\eta \cdot \left(\mathbf{y}(n) - \boldsymbol{\kappa}_{n-1}^T \boldsymbol{\alpha}_{n-1}\right) \cdot \boldsymbol{\kappa}_{n-1}}{\zeta + \left\|\boldsymbol{\kappa}_{n-1}^T\right\|^2} \quad (16)$$
    calculate
        $$\mathbf{y}(n+1) = \boldsymbol{\kappa}_n^T \cdot \boldsymbol{\alpha}_n \quad (17)$$
    Output: $\mathbf{y}(n+1)$
end
```

5 Experimental Results

In this section, experiments will be used to illustrate the good performance of our proposed algorithm regarding chaotic time series prediction. Three time series will be used for the simulation experiment. The first group is the Lorenz time series, which is a set of benchmark data. The second group and the third group are actual time series, which are the ENSO chaotic time series and the Beijing $PM_{2.5}$ chaotic time series, respec-

Table 2 Comparing the computational complexity of each iteration

Algorithm	Update $\boldsymbol{\alpha}_n$ during training	Calculate $\mathbf{e}(n)$ during testing
KLMS	$O(n)$	$O(n)$
QKLMS	$O(n)$	$O(m)$
KLMS-CSL1	$O(n)$	$O(l)$
KLMS-SOC	$O\left(n^2\right)$	$O(k)$
ASQ-KLMS	$O\left(n^2\right)$	$O(j)$

tively. At the same time, our proposed algorithm is compared with KLMS, QKLMS, KLMS-CSL1, KLMS-SOC algorithms. In order to prevent accidental situations, we performed ten simulation experiments on the three sets of data, and calculated the average value as the final result.

We will use the root mean square error (RMSE) and R^2 as the prediction accuracy evaluation indicators. The RMSE can be defined as

$$\text{RMSE} = \sqrt{\frac{\sum_{n=1}^{N}\left(\overset{\wedge}{\mathbf{y}}(n) - \mathbf{y}(n)\right)^2}{N}} \tag{18}$$

where n means the time step, and N is the number of samples calculated. If a method performs time series prediction, the smaller the RMSE of the calculated result, the higher the prediction accuracy of the method in time series prediction.

The R^2 can be calculated as

$$R^2 = 1 - \frac{\sum_{n=1}^{N}\left(\overset{\wedge}{\mathbf{y}}(n) - \mathbf{y}(n)\right)^2}{\sum_{n=1}^{N}\left(\mathbf{y}(n) - \overset{-}{\mathbf{y}}\right)^2} \tag{19}$$

where $\overset{-}{\mathbf{y}}$ denotes average number of predicted targets, and N is the number of samples calculated. The larger the R^2 of the calculated result, the higher the prediction accuracy of the method in time series prediction.

In order to better describe the prediction results, training time and testing time are used as indicators for calculating time.

Parameter optimization methods mainly include cross-validation and grid search methods. The algorithm parameter tuning in this paper uses grid search method. The main idea is to use the current parameters that have the greatest impact on the model until the optimization; tune the parameters, and so on, until all the parameters are adjusted. The steps of our proposed algorithm tuning are to first adjust the step size η and kernel bandwidthσ ; then adjust the parameters λ_d andλ_v of the SOC method, discarding about 1%-5% outliers; finally adjust the μ_0 ,ε and ζ .

Table 3 Parameters setting of 5-step prediction of Lorenz-x chaotic time series

Algorithms	μ_0	ε	η	ζ	λ_d	λ_v	Kernel Type	Kernel bandwidth
KLMS	–	–	0.06	–	–	–	Gauss	7
QKLMS	–	3.5	0.06	–	–	–	Gauss	7
KLMS-CSL1	0.5	3.5	0.06	–	–	–	Gauss	7
KLMS-SOC	–	3.5	0.06	–	2	3	Gauss	7
ASQ-KLMS	0.5	3.5	0.06	0.01	2	3	Gauss	7

5.1 Lorenz Chaotic Time Series

The Lorenz data set is a set of classical time series with chaotic characteristics. In this part, the validity of the proposed algorithm is verified. The simulation results of 2501 Lorenz three-dimensional data are used:

$$\begin{cases} \frac{dx}{dt} = e(-x + y) \\ \frac{dy}{dt} = fx - y - xz \\ \frac{dz}{dt} = xy - gz \end{cases} \tag{20}$$

where $e = 10$, $f = 28$, $g = 8/3$, and this time series is deteriorated by adding white Gaussian noise.

In the experiment, we use the C-C method to calculate and select the appropriate phase space reconstruction parameters. The delay time of the three-dimensional variables x, y, and z is selected as 8, 7, 8, and the embedding dimension is 6, 6, 6. After that, the first 80% of the samples are used for training, and the last 20% are used for testing. Table 3 shows the parameter settings for the 5-step prediction of the Lorenz-x time series.

Figure 1 shows the training MSE of ASQ-KLMS algorithm in Lorenz-x chaotic time series. Figure 2 shows the 5-step prediction curve of Lorenz-x chaotic time series using ASQ-KLMS algorithm, and Table 4 shows 5-step prediction results of Lorenz-x time series by KLMS, QKLMS, KLMS-CSL1, KLMS-SOC and ASQ-KLMS algorithm.

The RMSE and R^2 are important evaluation indicators, and the training time and testing time are efficiency assessment indicators. From Fig. 1, we can see that the model is well trained and has good convergence. From Table 4, we can see that the ASQ-KLMS algorithm simulation results are satisfactory on these four indicators. The shortest testing time means that the algorithm constitutes the most accurate and compact dictionary, which greatly reduces the computational complexity. Therefore, ASQ-KLMS algorithm has high efficiency online prediction capability. Because the SOC criterion is adopted, the training time is sacrificed, but the model has good anti-noise ability. From prediction accuracy, the ASQ-KLMS algorithm performs better than the KLMS-SOC algorithm, which means that the ASQ-KLMS algorithm improves the KLMS-SOC algorithm. At the same time, in Fig. 2, the blue line represents the original data curve and the orange line represents the predicted value curve from the upper subgraph, and the prediction curve can follow the original curve well.

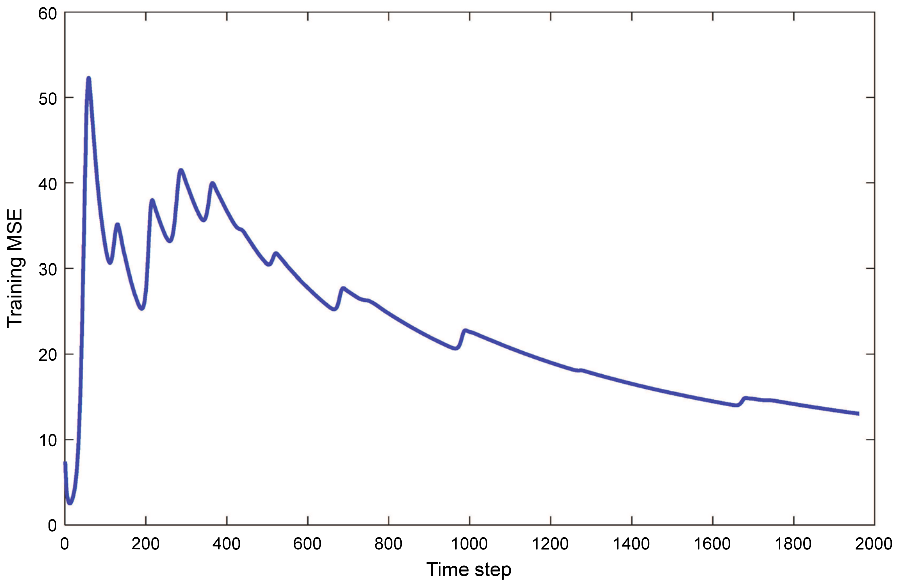


Fig. 1 The training MSE of ASQ-KLMS algorithm in Lorenz-x chaotic time series

In the next subgraph, we can see that the error curve has less fluctuation. Therefore, the ASQ-KLMS algorithm is considered to have good performance for online prediction.

5.2 ENSO Chaotic Time Series

The ENSO is the abbreviation of El Nino-Southern Oscillation. ENSO is a climatic phenomenon that exists in the equatorial sea. Among them, the Southern Oscillation refers to the reversed-pressure vibration occurring between the Southeast Pacific and the Indian Ocean and Indonesia. It is one of the most prominent and important phenomena in the interannual variation of the tropical circulation. El Nino is an abnormal phenomenon in which the sea surface temperature in the equatorial Pacific Ocean in the Middle East continues to rise abnormally. The Southern Oscillation and El Nino are interrelated. When the El Nino phenomenon occurs, the sea surface temperature rises, the Pacific water temperature rises, and the global atmospheric circulation changes, while the Southern Oscillation is the most direct change. The ENSO occurs approximately every two to seven years and lasts for several months. It is one of the main culprits for natural weather disasters such as floods. The catastrophic climate problems such as the Yangtze River floods and the southern blizzards that have occurred in China are inextricably linked to ENSO. Therefore, timely and accurate prediction of ENSO related index has important significance and practical application value. The Pacific Decadal Oscillation (PDO) phenomenon occurs every 10 years. Many scholars believe that PDO and ENSO have strong correlations. Therefore, this paper combines the PDO and ENSO indices into a set of time series prediction data for simulation experiments.

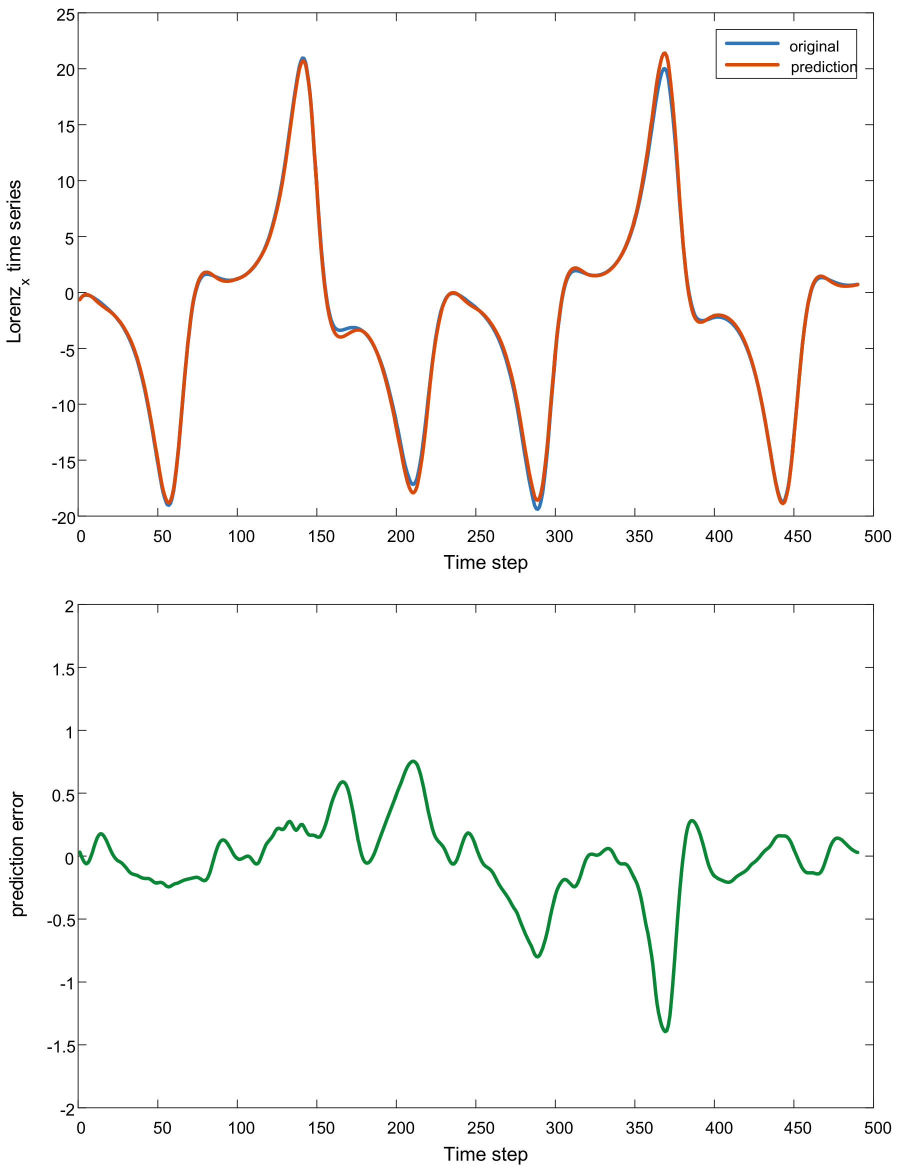


Fig. 2 Lorenz-x chaotic time series 5-step prediction curve and error curve using ASQ-KLMS algorithm

By above knowable, we use the National Oceanic and Atmospheric Administration (NOAA) website (http://www.psl.noaa.gov/gcos_wgsp/Timeseries/) to obtain the monthly Nino 1.2, Nino 3, Nino 4, Nino3.4, SOI index and PDO index from 1949 to June 2018, including 834 samples. The acquired data is in standard PSL format. The Lyapunov exponent indicates that the ENSO time series is chaotic. At the same time, we use the C-C method to calculate and select the appropriate phase space reconstruc-

Table 4 Five-step online prediction test results of Lorenz-x chaotic time series

Algorithms	Training time	Testing time	Testing RMSE	Testing R^2
KLMS	0.3684	0.0602	0.9028	0.9890
QKLMS	0.2647	0.0293	0.7562	0.9923
KLMS-CSL1	**0.1611**	**0.0110**	0.9492	0.9879
KLMS-SOC	2.9795	0.0546	1.0828	0.9842
ASQ-KLMS	0.9625	0.0130	**0.7094**	**0.9932**

Table 5 Parameter setting of five-step prediction for NINO 3.4 index

Algorithms	μ_0	ε	η	ζ	λ_d	λ_v	Kernel Type	Kernel bandwidth
KLMS	–	–	0.2	–	–	–	Gauss	3
QKLMS	–	2	0.2	-	–	–	Gauss	3
KLMS-CSL1	0.5	–	0.2	–	–	–	Gauss	3
KLMS-SOC	–	2	0.2	-	2	3	Gauss	3
ASQ-KLMS	0.5	2	0.2	0.01	2	3	Gauss	3

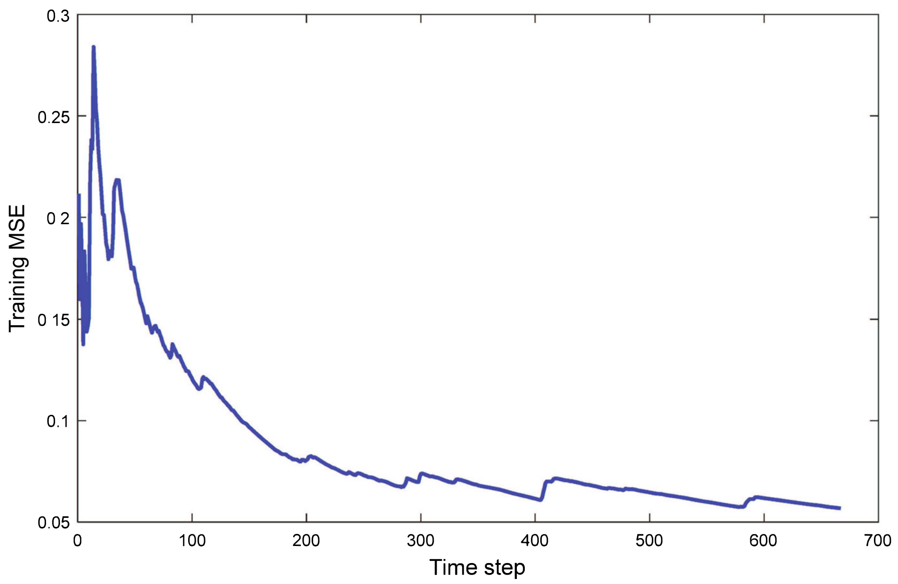


Fig. 3 The training MSE of ASQ-KLMS algorithm in Nino 3.4 chaotic time series

tion parameters. The embedding dimension is selected as 3, 3, 4, 3, 4, and 4. The delay time of the six-dimensional variables is selected as 2, 2, 3, 3, 2, and 2.

Table 5 shows parameter setting of five-step prediction for NINO 3.4 index. After that, the first 80% of the samples are used for training and the last 20% are used for testing.

Table 6 Five-step online prediction test results of NINO 3.4 chaotic time series

Algorithms	Training time	Testing time	Testing RMSE	Testing R^2
KLMS	0.2893	0.0218	0.2156	0.9346
QKLMS	**0.2476**	0.0036	0.1797	0.9574
KLMS-CSL1	0.4446	0.0178	0.3528	0.8357
KLMS-SOC	0.3554	0.0062	0.2009	0.9467
ASQ-KLMS	0.3242	**0.0027**	**0.1657**	**0.9665**

Figure 3 shows the training MSE of ASQ-KLMS algorithm in Nino 3.4 chaotic time series. Figure 5 shows Nino 3.4 chaotic time series 5-step prediction curve using ASQ-KLMS algorithm. Table 6 shows 5-step prediction results of Nino 3.4 chaotic time series by KLMS, QKLMS, KLMS-CSL1, KLMS-SOC and ASQ-KLMS algorithm.

From Fig. 3, we can see that the model is well trained and has good convergence. It can be seen from the simulation results in Table 6 that the ASQ-KLMS algorithm shows excellent performance. ENSO is the actual data that contains noise, so there will be outliers in the sequence. The ASQ-KLMS algorithm has SOC criterion and weight adaptive adjustment, so it has good anti-noise ability, and the simulation results also illustrate this point. Specifically, we can see that the ASQ-KLMS algorithm has the shortest test time, which means that it constitutes the most accurate and compact

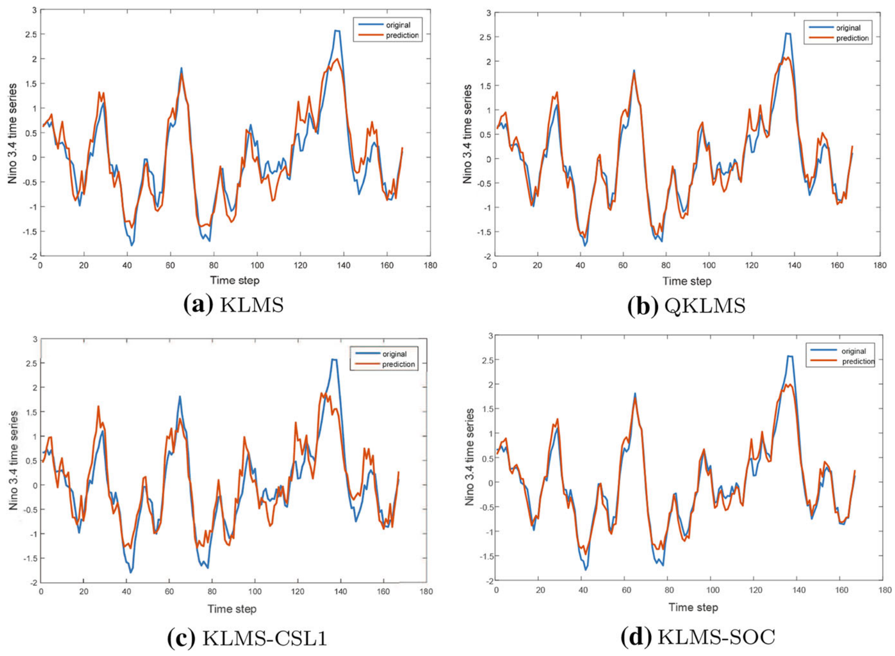
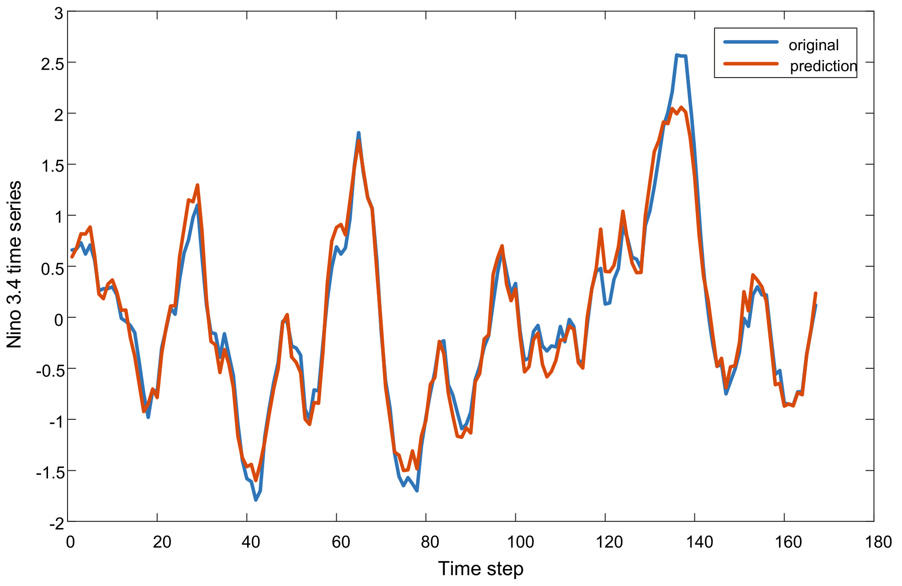


Fig. 4 Nino 3.4 chaotic time series 5-step prediction curve using KLMS, QKLMS, KLMS-CSL1, KLMS-SOC algorithm



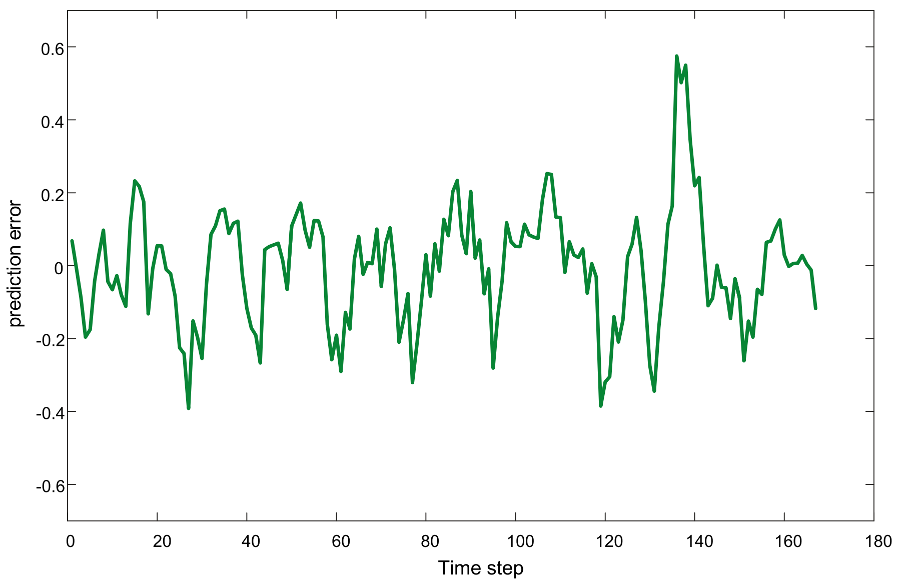


Fig. 5 Nino 3.4 chaotic time series 5-step prediction curve and error curve using ASQ-KLMS algorithm

Table 7 Parameters setting of five-step prediction of Beijing $PM_{2.5}$ chaotic time series

Algorithms	μ_0	ε	η	ζ	λ_d	λ_v	Kernel Type	Kernel bandwidth
KLMS	–	–	0.2	–	–	–	Gauss	2
QKLMS	–	0.1	0.2	–	–	–	Gauss	2
KLMS-CSL1	0.5	-	0.2	–	–	–	Gauss	2
KLMS-SOC	–	0.1	0.2	–	8	9	Gauss	2
ASQ-KLMS	0.9	0.1	0.2	0.01	8	9	Gauss	2

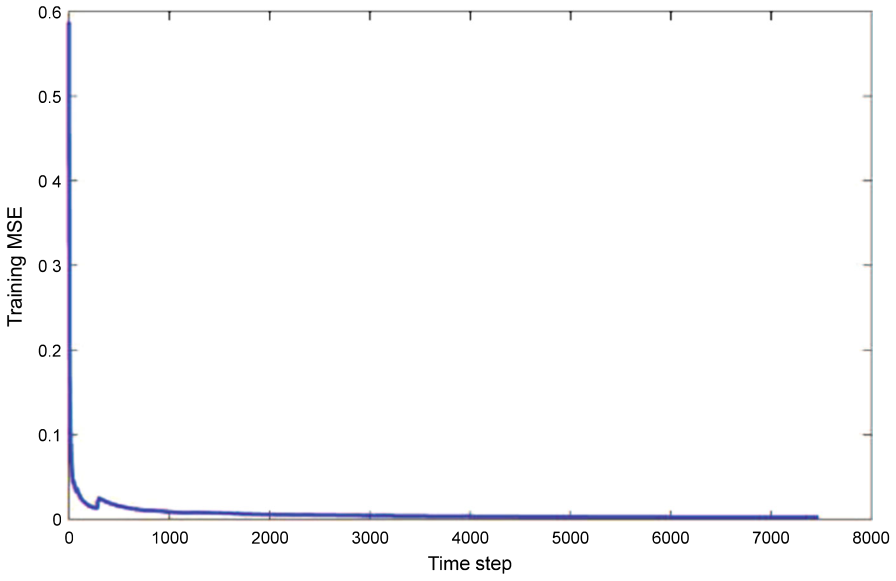


Fig. 6 The training MSE of ASQ-KLMS algorithm in Beijing $PM_{2.5}$ chaotic time series

dictionary, which greatly reduces the test time and has higher prediction efficiency. In addition, the accuracy index predicted by the ASQ-KLMS algorithm is also superior to other algorithms in Fig. 4. Therefore, the ASQ-KLMS algorithm has been proven to have a good predictive effect on the ENSO chaotic time series. Simultaneously, the blue line represents the original data curve, and the orange line represents the predicted value curve from the upper subgraph in Fig. 5, the prediction curve can follow the original curve well. In the next subgraph, error curve fluctuations are small. So we can get a conclusion that the ASQ-KLMS algorithm has excellent effectiveness for online prediction of ENSO chaotic time series.

5.3 Beijing $PM_{2.5}$ Chaotic Time Series

As we all know, continuous smog weather has occurred in many places, seriously affecting the health of residents. Therefore, it is urgent to increase air quality research, especially to monitor and predict air pollution. $PM_{2.5}$ is one of the important indicators

Table 8 Five-step online prediction test results of Beijing $PM_{2.5}$ chaotic time series

Algorithms	Training time	Testing time	Testing RMSE	Testing R^2
KLMS	2.3283	0.1776	0.0394	0.9683
QKLMS	**0.3372**	0.0223	0.0373	0.9658
KLMS-CSL1	0.3900	0.0195	0.0632	0.9018
KLMS-SOC	3.0808	0.1628	0.0339	0.9717
ASQ-KLMS	2.5361	**0.0127**	**0.0315**	**0.9774**

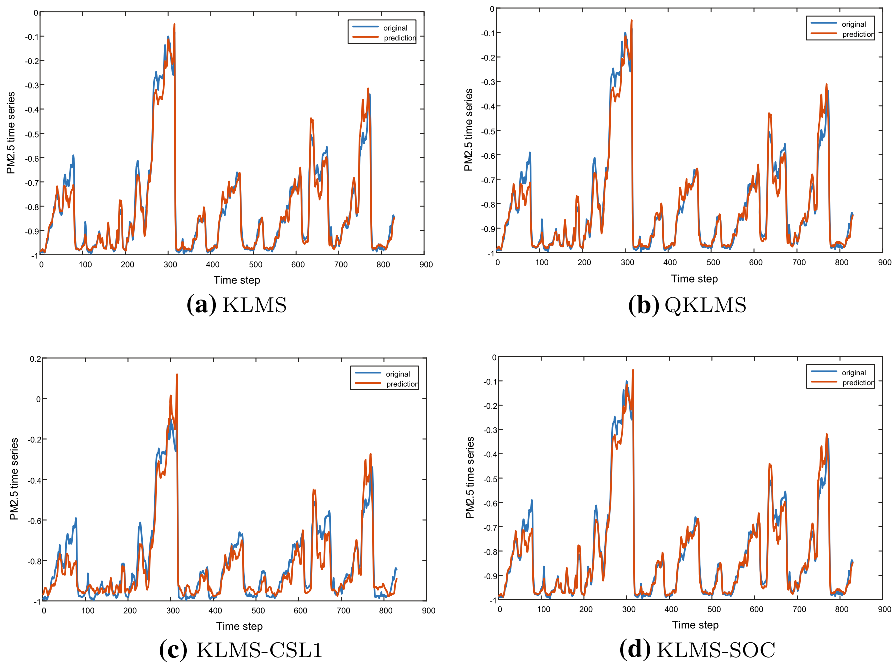


Fig. 7 Beijing $PM_{2.5}$ chaotic time series 5-step prediction curve using KLMS, QKLMS, KLMS-CSL1, KLMS-SOC algorithm

used to measure air quality. Therefore, predicting the value of $PM_{2.5}$ has important practical significance. The Lyapunov exponent indicates that the Beijing $PM_{2.5}$ time series data is chaotic.

In this section, we conduct simulations on Beijing $PM_{2.5}$ chaotic time series which consists of hourly $PM_{2.5}$, PM_{10}, CO, NO_2, O_3, SO_2 in Beijing in 2019. The original data set is confirmed by the test station. This data set comes from the website of Beijing Environmental Protection Inspection Center (http://www.bjmemc.com.cn/). After preprocessing the data, a total of 8300 samples will be used for simulation experiments, the first 90% will be used to train the model, and the remaining 10% will be used for testing. The C-C method is used to calculate and select the appropriate phase space reconstruction parameters. And the embedding dimensions are 3, 3, 4, 3,

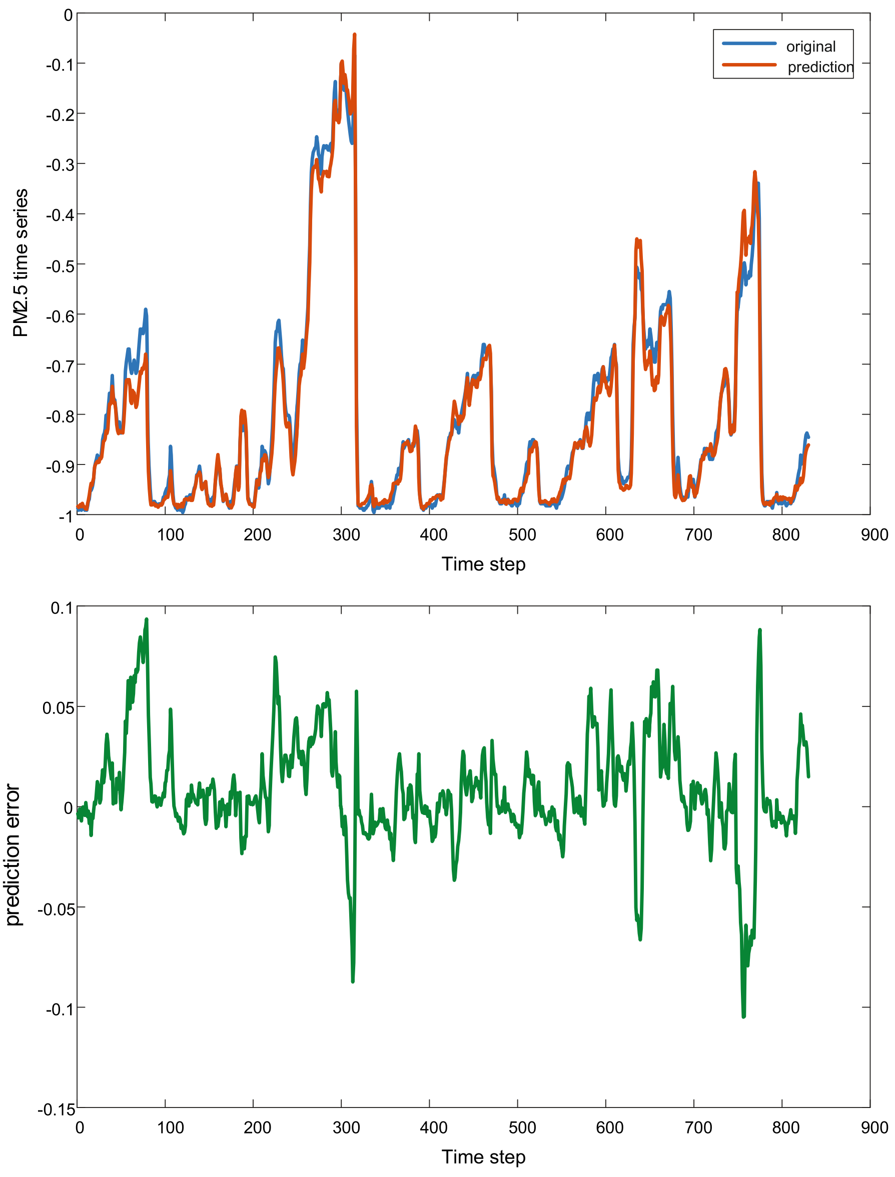


Fig. 8 Beijing $PM_{2.5}$ chaotic time series five-step prediction curve and error curve using ASQ-KLMS algorithm

4, and 3, and the delay times are 10, 10, 10, 10, 8, and 10. Table 7 shows the parameters setting of five-step prediction of Beijing $PM_{2.5}$ chaotic time series.

Figure 6 shows the training MSE of ASQ-KLMS algorithm in Lorenz-x chaotic time series. Figure 8 shows Beijing $PM_{2.5}$ chaotic time series five-step prediction curve using ASQ-KLMS algorithm. Table 8 shows five-step prediction results of Beijing

$PM_{2.5}$ chaotic time series by KLMS, QKLMS, KLMS-CSL1, KLMS-SOC and ASQ-KLMS algorithm.

From Fig. 6, we can see that the model is well trained and has good convergence. The simulation results in Table 8 show that the ASQ-KLMS algorithm performs well in both prediction accuracy and efficiency. The Beijing $PM_{2.5}$ chaotic time series is the actual data containing noise, so there will be abnormal values in the sequence. The ASQ-KLMS algorithm has SOC criterion and weight adaptive adjustment, so it has good anti-noise ability, and the simulation results also illustrate this point. Although we added a SOC criterion, we sacrificed a little training time. Fortunately, we have formed an accurate and compact dictionary, which greatly reduces the test time and has higher prediction efficiency. Therefore, the ASQ-KLMS algorithm has been proven to have a good predictive effect on the Beijing $PM_{2.5}$ chaotic time series in Fig. 7. In Fig. 8, the blue line represents the original data curve and the orange line represents the predicted value curve; the prediction curve can follow the original curve well. In the next subgraph, the prediction error curve has a small fluctuation range. Therefore, the ASQ-KLMS algorithm is proved to be effective in completing the online prediction of Beijing $PM_{2.5}$ chaotic time series.

6 Conclusion

In this paper, the coherence criterion and the weight adaptive adjustment integrated with the quantization method form our proposed ASQ-KLMS algorithm. Compared with the other four algorithms, the ASQ-KLMS algorithm forms a more accurate and compact dictionary, and the prediction efficiency and prediction accuracy are very good. In addition, it also has good tracking time-varying characteristics and has certain adaptive ability to noise. The experiments of the Lorenz, the ENSO and the Beijing $PM_{2.5}$ chaotic time series prove the effectiveness of the ASQ-KLMS algorithm.

In order to improve the ability to track time-varying characteristics, we consider combining KAF and recurrent neural networks, drawing on the advantages of each other, and adding appropriate sparse methods to improve prediction accuracy in the future.

Acknowledgements This work is supported by the National Natural Science Foundation of China (61773087) and the Fundamental Research Funds for the Central Universities (DUT20LAB114, DUT2018TB06).

Data Availability The data used to support the findings of this study are available from the corresponding author on reasonable request.

References

1. N. Aronszajn, Theory of reproducing kernels. Trans. Am. Math. Soc. **68**(3), 337–404 (1950)
2. B. Chen, W. Liu, J.C. Principe, Theoretical methods in machine learning, in Springer Handbook of Computational Intelligence, (Springer, 2015), pp. 523–543
3. B. Chen, S. Zhao, P. Zhu, J.C. Príncipe, Quantized kernel least mean square algorithm. IEEE Trans. Neural Netw. Learn. Syst. **23**(1), 22–32 (2011)

4. J. Chen, K. Li, H. Rong, K. Bilal, K. Li, S.Y. Philip, A periodicity-based parallel time series prediction algorithm in cloud computing environments. Inf. Sci. **496**, 506–537 (2019)
5. Y. Engel, S. Mannor, R. Meir, The kernel recursive least-squares algorithm. IEEE Trans. Signal Process. **52**(8), 2275–2285 (2004)
6. H. Fan, Q. Song, A sparse kernel algorithm for online time series data prediction. Expert Syst. Appl. **40**(6), 2174–2181 (2013)
7. J. Fernandez-Bes, V. Elvira, S. Van Vaerenbergh, A probabilistic least-mean-squares filter, in 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), (IEEE, 2015), pp 2199–2203
8. W. Gao, J. Chen, C. Richard, J. Huang, Online dictionary learning for kernel lms. IEEE Trans. Signal Process. **62**(11), 2765–2777 (2014)
9. M. Han, W. Ren, M. Xu, T. Qiu, Nonuniform state space reconstruction for multivariate chaotic time series. IEEE Transa. Cybern. **49**(5), 1885–1895 (2018)
10. M. Han, S. Zhang, M. Xu, T. Qiu, N. Wang, Multivariate chaotic time series online prediction based on improved kernel recursive least squares algorithm. IEEE Trans. Cybern. **49**(4), 1160–1172 (2019)
11. D. Li, M. Han, J. Wang, Chaotic time series prediction based on a novel robust echo state network. IEEE Trans. Neural Netw. Learn. Syst. **23**(5), 787–799 (2012)
12. K. Li, J.C. Príncipe, The kernel adaptive autoregressive-moving-average algorithm. IEEE Trans. Neural Netw. Learn. Syst. **27**(2), 334–346 (2016)
13. Y. Li, Z. Zhu, D. Kong, H. Han, Y. Zhao, Ea-lstm: Evolutionary attention-based LSTM for time series prediction. Knowledge-Based Systems **181**, 104785 (2019)
14. J. Liu, E. Zio, An adaptive online learning approach for support vector regression: Online-SVR-FID. Mech. Syst. Signal Process. **76**, 796–809 (2016)
15. W. Liu, I. Park, J.C. Principe, An information theoretic approach of designing sparse kernel adaptive filters. IEEE Trans. Neural Netw. **20**(12), 1950–1961 (2009)
16. W. Liu, P.P. Pokharel, J.C. Principe, The kernel least-mean-square algorithm. IEEE Trans. Signal Process. **56**(2), 543–554 (2008)
17. W. Liu, J.C. Principe, Kernel affine projection algorithms. EURASIP J. Adv. Signal Process. **2008**, 1–12 (2008)
18. W. Liu, J.C. Principe, S. Haykin, Kernel adaptive filtering: A comprehensive introduction. IEEE Comput. Intell. Mag. **5**(3), 52–55 (2010)
19. Y. Liu, C. Sun, S. Jiang, A reduced gaussian kernel least-mean-square algorithm for nonlinear adaptive signal processing. Circuits Syst. Signal Process. **38**(1), 371–394 (2019)
20. X. Lu, L. Ming, W. Liu, H. Li, Probabilistic regularized extreme learning machine for robust modeling of noise data. IEEE Trans. Cybern. **48**(8), 2368–2377 (2018)
21. W. Ma, J. Duan, W. Man, H. Zhao, B. Chen, Robust kernel adaptive filters based on mean p-power error for noisy chaotic time series prediction. Eng. Appl. Artif. Intell. **58**, 101–110 (2017)
22. L. Noakes, The takens embedding theorem. Int. J. Bifurc. Chaos **1**(04), 867–872 (1991)
23. K. Pelekanakis, M. Chitre, Adaptive sparse channel estimation under symmetric alpha-stable noise. IEEE Trans. Wirel. Commun. **13**(6), 3183–3195 (2014)
24. J. Platt, *A Resource-Allocating Network for Function Interpolation* (MIT Press, Cambridge, 1991)
25. C. Richard, J.C.M. Bermudez, P. Honeine, Online prediction of time series data with kernels. IEEE Trans. Signal Process. **57**(3), 1058–1067 (2008)
26. R.R. Sharma, R.B. Pachori, Baseline wander and power line interference removal from ecg signals using eigenvalue decomposition. Biomed. Signal Process. Control **45**, 33–49 (2018)
27. R.R. Sharma, R.B. Pachori, Eigenvalue decomposition of hankel matrix-based time-frequency representation for complex signals. Circuits Syst. Signal Process. **37**(8), 3313–3329 (2018)
28. F. Sheikholeslami, D. Berberidis, G.B. Giannakis, Kernel-based low-rank feature extraction on a budget for big data streams, in 2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP), (2015), pp 928–932
29. C. Vairappan, H. Tamura, S. Gao, Z. Tang, Batch type local search-based adaptive neuro-fuzzy inference system (ANFIS) with self-feedbacks for time-series prediction. Neurocomputing **72**(7–9), 1870–1877 (2009)
30. S. Van Vaerenbergh, I. Santamaría, W. Liu, J.C. Príncipe, Fixed-budget kernel recursive least-squares, in 2010 IEEE International Conference on Acoustics, Speech and Signal Processing, (IEEE 2010) pp 1882–1885

31. V. Vapnik, *The Nature of Statistical Learning Theory* (Springer science & business media, Berlin, 2013)
32. L. Wang, R. Liu, Human activity recognition based on wearable sensor using hierarchical deep LSTM networks. Circuits Syst. Signal Process. **39**(2), 837–856 (2020)
33. S. Wang, W. Wang, L. Dang, J. Yunxiang, Kernel least mean square based on the nystrom method. Circuits Syst. Signal Process. **38**(7), 3133–3151 (2019)
34. M. Yukawa, Multikernel adaptive filtering. IEEE Trans. Signal Process. **60**(9), 4672–4682 (2012)
35. P. Zhang, Y. Jia, J. Gao, W. Song, H. Leung, Short-term rainfall forecasting using multi-layer perceptron. IEEE Trans. Big Data **6**(1), 93–106 (2020)
36. S. Zhang, H. Cao, S. Yang, Y. Zhang, X. Hei, Sequential outlier criterion for sparsification of online adaptive filtering. IEEE Trans. Neural Netw. Learn. Syst. **29**(11), 5277–5291 (2018)
37. S. Zhao, B. Chen, P. Zhu, J.C. Príncipe, Fixed budget quantized kernel least-mean-square algorithm. Signal Process. **93**(9), 2759–2770 (2013)
38. Z. Zhao, M. Jin, The decorrelated kernel least-mean-square algorithm, in 2016 IEEE 13th International Conference on Signal Processing (ICSP), (IEEE 2016), pp 367–371
39. H. Zhou, J. Huang, L. Feng, J. Thiyagalingam, T. Kirubarajan, Echo state kernel recursive least squares algorithm for machine condition prediction. Mech. Syst. Signal Process. **111**, 68–86 (2018)
40. H. Zhu, G. Leus, G.B. Giannakis, Sparsity-cognizant total least-squares for perturbed compressive sampling. IEEE Trans. Signal Process. **59**(5), 2002–2016 (2011)