Kernel Adaptive Filtering Algorithms for Time Series Prediction

Jackson Cornell
Department of Electrical and Computer Engineering
University of Florida
jcornell@ufl.edu

Michael Kim
Department of Electrical and Computer Engineering
University of Florida
michaelkim@ufl.edu

Abstract— In this report we present a time series prediction method based on kernel adaptive filters. We implement the quantized kernel least mean squares (QKLMS) and quantized kernel recursive least squares (QKRLS) algorithms with both the MSE and MCC cost functions to predict the next sample of the sunspot data. After obtaining metrics such as the optimal hyperparameters and error power, the sunspot time series is generated using a recurrent system with a first in last out buffer (FILO), and the results are evaluated.

Keywords— *Kernel adaptive filter, reproducing kernel Hilbert space (RKHS), quantized kernel least mean squares (QKLMS), quantized kernel recursive least squares (QKRLS), mean squared error (MSE), maximum correntropy criterion (MCC)*

I. Introduction

Kernel adaptive filters are a type of nonlinear adaptive filter. They implement a nonlinear transfer function using kernel methods, which can be effectively used to analyze nonlinear and nonstationary signals. This can serve many applications like system identification, noise cancellation, and time series prediction. This paper will focus on the latter, using recordings of sunspots retrieved monthly over two centuries as training data. The data will be used to train five separate adaptive filter models to predict sunspot behavior over a period of 2,820 months.

The two adaptive filter models that will be tested are the QKLMS and the QKRLS filter. Both models work by mapping the input data to a Reproducing Kernel Hilbert Space (RKHS) and finding the set of weights that reduces an error measurement. Using this method, non-linear data can be effectively modelled. Because the network of weights grows linearly with Kernel methods, the number of vectors in the RKHS will be reduced via the Vector Quantization (VQ) operator.

II. Methods

A. Data

The given dataset consists of a recording of a nonstationary time series of monthly sunspot activity from 1749 to 1984, as seen in Figure 1. The input signal is delayed by one sample and the desired response is the current sample. The first 200 samples are used to train the model, and the remaining samples are used as the testing dataset to compare the performances of the algorithms. The sunspot time series is recorded monthly and averaged annually, so there is a high linear relationship between the neighboring samples.

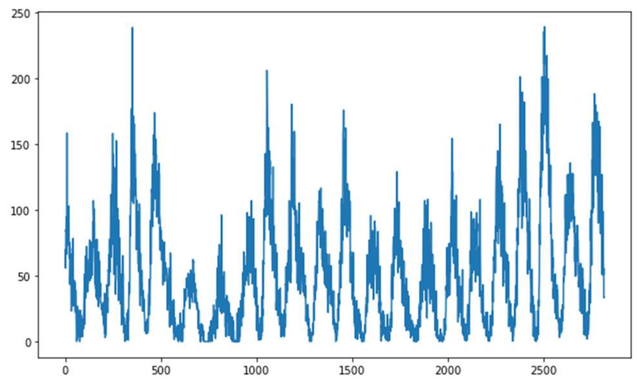

Figure 1: Monthly Sunspot Data over the period 1749-1984

B. Kernel Least Means Square (KLMS)

The KLMS filter uses stochastic gradient descent to minimize the least-squares cost function. The least-squares cost function finds a set of parameters that minimizes the sum of the squared deviations between the observed response and a function of the parameters and input response [1]. These parameters are found iteratively using stochastic gradient descent, a process where the algorithm descends the cost function and finds its minimum.

QKLMS can be obtained by quantizing the feature vector $\varphi(i)$ in the weight-update equation $\varphi(i) = \varphi(i-1) +$

ηe(i) [2]. One disadvantage of sparsification methods like KLMS is that redundant input data are discarded, which is not useful in updating the coefficients of the current network. However, QKLMS uses the redundant data to locally update the coefficient of the closest center. By limiting the expansion's size, QKLMS results in a more compact network with better accuracy.

C. Kernel Recursive Least Squares (KRLS)

Recursive Least Squares works by minimizing a regularized least squares regression equation recursively. In RKHS, the following equation is used:

$$\min_{f\in H}\sum_{j=1}^{i-1}(y(j)-f(u(j)))^2+\gamma\|f\|_H^2$$

Where $f(\cdot)=\sum_{j=1}^{i-1}\alpha K(u(j),\ \cdot)$. The Gaussian kernel $K(x,x')=\exp\left(-\frac{(x-x')^2}{2\sigma^2}\right)$ is used to map the input space to RKHS. Because of the computational complexity of computing the KRLS in an on-line environment, it is necessary to quantize the feature vectors using the Vector Quantization (VQ) operator. [3]

D. Maximum Correntropy Criterion (MCC)

QKLMS and QKRLS are computed and compared using two separate cost functions: the Mean Square Error (MSE) and Maximum Correntropy Criterion (MCC). MSE is the standard cost function in adaptive filtering and has proved useful in dealing with data that relies on its first two orders of statistical moments. When the data's distribution function has longer tails i.e. experiences more outliers, it becomes necessary to use a cost function that incorporates higher order statistical moments. MCC works by maximizing the correntropy of the transformed input data and the desired data.

Correntropy is a generalized similarity measure between two random variables using an Information Theory framework. Correntropy can be estimated using the following equation:

$$V(X,Y)=\frac{1}{N}\sum_{i=1}^{N}K_\sigma(e(i))$$

Where e(i) = x(i)− y(i) and $K_\sigma(\cdot)$ is a Mercer kernel. The following Gaussian kernel will be used:

$$K_\sigma(e)=\exp\left(-\frac{e^2}{2\sigma^2}\right)$$

This can be further expanded upon to derive the MCC cost function by using the desired signal as $x(i)$ and $y(i)$ is the weighted transformation of the input signal. By maximizing the correntropy function, a set of weights can be found that results in a maximum similarity between the transformed input signal and desired signal that uses all even order statistical moments [4].

E. Hyperparameters

To properly evaluate the performance our algorithms, we had to determine the optimal hyperparameters for our models: namely, the Gaussian kernel size, quantization size, and learning rate/forgetting factor. For MCC, an additional entropy kernel size controls the robustness of the filter output by attenuating the outliers.

There exists a tradeoff between the complexity and accuracy of any model [5]. For our purposes, the quantization size is set at 0.001 for each algorithm. For QKLMS, the step size is set at 0.2; the kernel size is 1; the forgetting factor is 0.98. For QKRLS, the regularization factor is 0.001 and the entropy kernel size is 34. All hyperparameters were found by iterating through several values until one that results in a best performance was found. Below, the MCC kernel size was found by only considering values that projected the entire data set without deviating from a third of the standard deviation. From there, the value with the peaks closest to the desired signal's peaks were found.

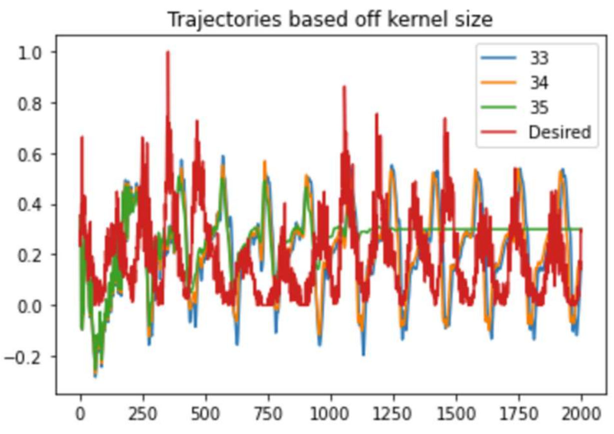

Figure 2: Testing of MCC kernel size

III. Results

A. Conventional Prediction

The adaptive filtering algorithms were applied to the dataset after normalizing it to be between 0 and 1. Figures 4-7 show the predicted output alongside the original expected value.

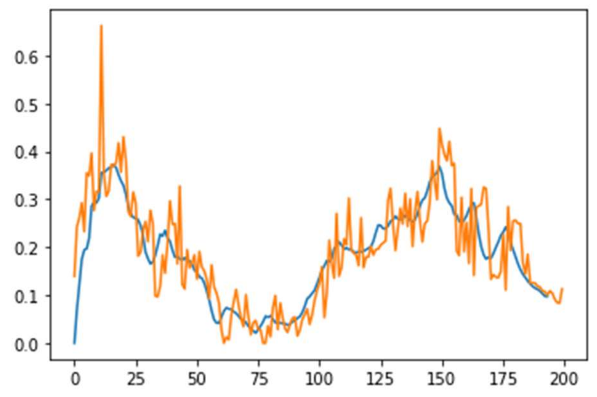

Figure 3: Actual sunspot value (orange) and predicted value using QKLMS-MSE (blue)

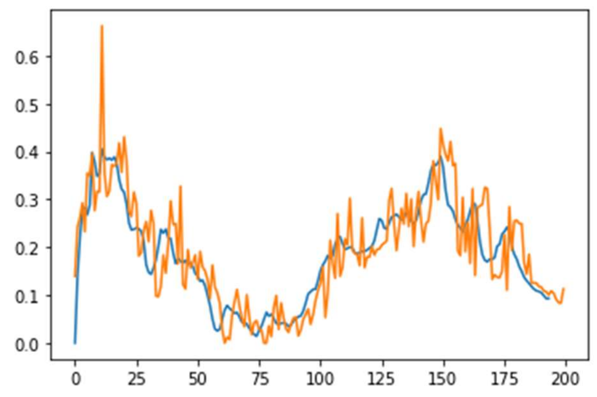

Figure 4: Actual sunspot value (orange) and predicted value using QKLMS-MCC (blue)

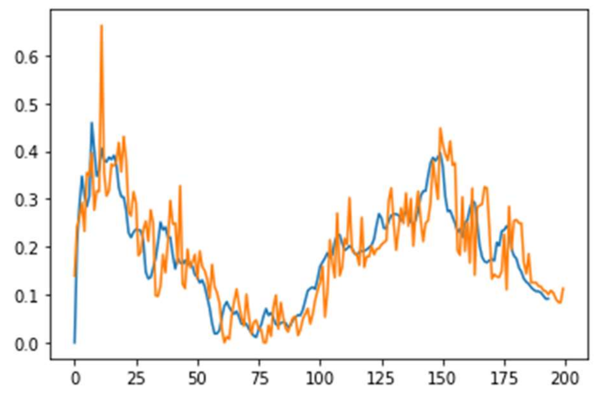

Figure 5: Actual sunspot value (orange) and predicted value using QKRLS-MSE (blue)

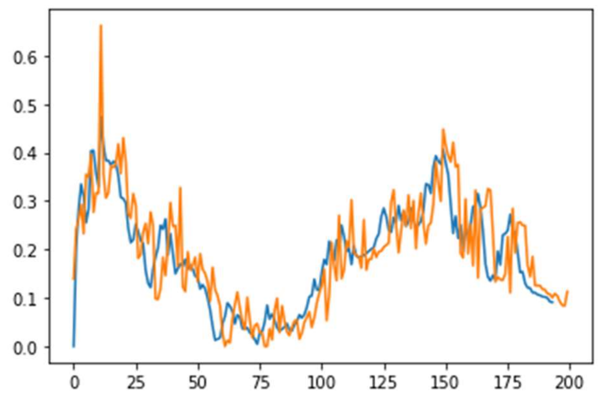

Figure 6: Actual sunspot value (orange) and predicted value using QKRLS-MCC (blue)

Figures 6 and 7 show the final error distribution in the test set for the two cost functions in the form of a histogram. From the plot we can see that for both QKLMS and QKRLS, MCC results in a smaller variance and a more accurate prediction.

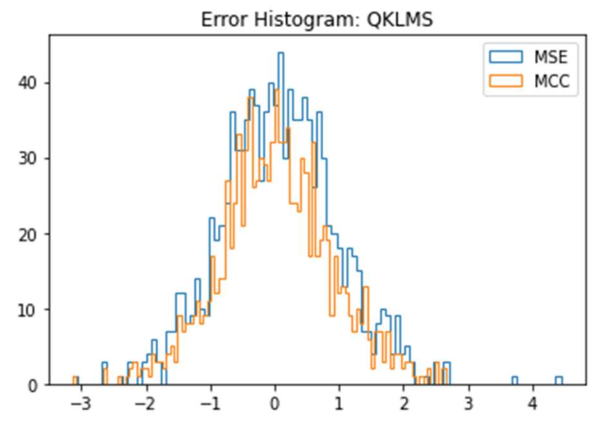

Figure 7: Histogram of Errors using QKLMS

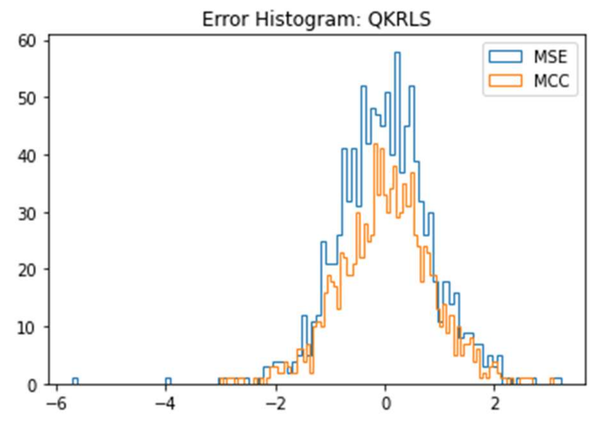

Figure 8: Histogram of Errors using QKRLS

Table 1: Comparison of Final Prediction Error Power

Algorithm	**Normalized Error Power**	**Computation Time (sec)**
LMS-MSE	0.1582	0.1083
LMS-MCC	0.1648	0.1202
QKLMS-MSE	0.0891	0.0929
QKLMS-MCC	0.0945	0.0885
QKRLS-MSE	0.0774	0.0693
QKRLS-MCC	0.0846	0.0781

Table 2: Comparison of error based on lag

Algorithm	**Error (10 samples ahead)**	**Error (20 samples ahead)**
LMS-MSE	0.3019	0.5418
LMS-MCC	0.3011	0.5402
QKLMS-MSE	0.2989	0.5035
QKLMS-MCC	0.2770	0.4948
QKRLS-MSE	0.2854	0.4781
QKRLS-MCC	0.2801	0.4802

From Table 1, we can objectively compare the different adaptive algorithms by quantifying the final prediction error power in the dataset and the tradeoff with computation time. As a baseline for the comparison, we implemented the linear LMS filter trained with the same two cost functions. We can see that QKLMS trained with the MSE cost resulted in the lowest error and execution time.

B. Tracjetory Generation

An estimate of the trajectory was made by using the previously found hyperparameters to generate a series of 20 estimates, each with different starting embedded vectors. The 20 outputs were then averaged together. The implementation for the trajectory was by using a FILO and feeding the generated outputs into the buffer.

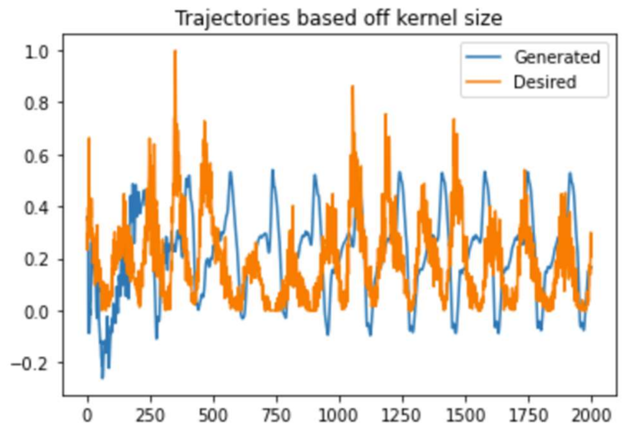

Figure 9: Generated Trajectory for QKRLS MCC

It can be seen from the above figure that the estimated signal is very close to the desired signal. The exception is, since MCC was used, outliers are mostly ignored.

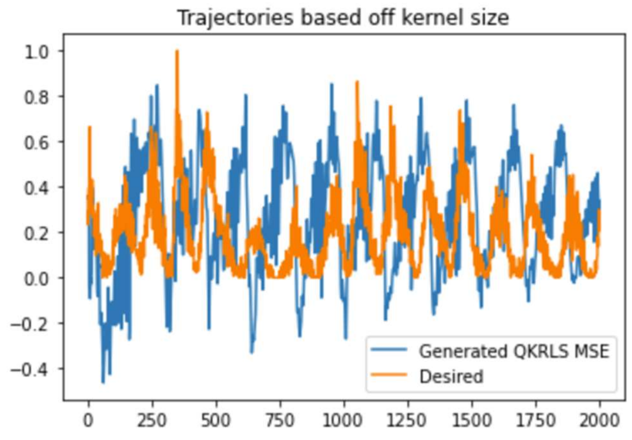

Figure 10: Generated trajectory for QKRLS MSE

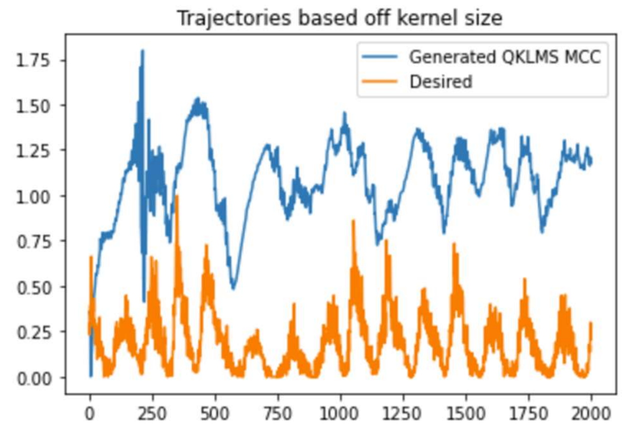

Figure 11: Generated trajectory for QKLMS MCC

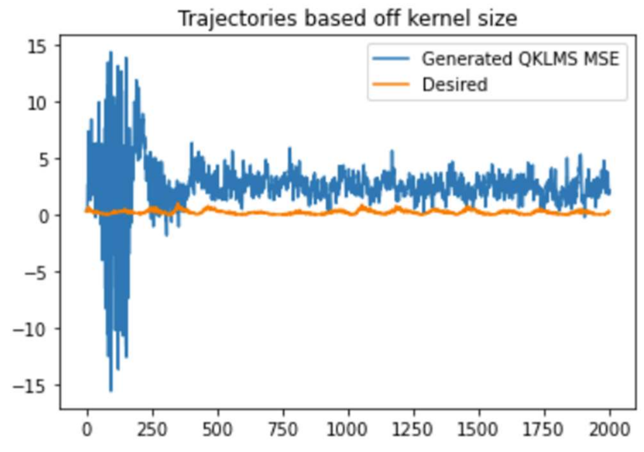

Figure 12: Generated trajectory for QKLMS MSE

From the above plots there are two main points to make:

1. MCC tended to perform better than MSE
2. QKRLS tended to perform better than QKLMS

This is due to the nature of the data. The data's distribution has long tails (i.e. many outliers) which means an entropy-based cost function will be better suited. Additionally, it is important for the weights to converge quickly given the data, and so the QKRLS is better suited than the QKLMS.

IV. Discussion

A. *Performance Analysis*

Using the quantized version of the filters it is apparent that computational complexity increases exponentially with no vector thresholding. This will, however, minimize the error. Nonetheless, it is beneficial to quantize the input vectors so that the computational speed is greatly increased, with little discernable difference in the error.

Exceeding expectations, the trajectory was able to estimate the entire data set from a single embedding vector without its error deviating by a third of the desired signal's standard deviation. This demonstrates that, even with training on the first two hundred samples of the data, mapping the input to RKHS can greatly help with estimating a non-linear function. It also shows that MCC is a powerful cost function robust against outliers due to considering statistical moments beyond the first two orders.

V. Conclusion

In this paper, we presented a solution to a nonstationary time series prediction problem using kernel based adaptive filtering algorithms. A comparative analysis was conducted between several models, including implementations of QKLMS and QKRLS trained with both the MSE and MCC cost functions. After first creating models to predict future samples, we then generated the sunspot time series with the different trained models. Based on the results obtained, we can conclude that the MCC criterion performed better than the MSE criterion due to the presence of outliers in the data set. QKRLS also performed better than QKLMS due to its lower error power and faster convergence rate. Further work may involve implementing related algorithms, such as QKAPA, Sparse QKLMS, and Sparse QKRLS, or reviewing the current literature for more advanced online VQ methods.

VI. References

[1] Engineering Statistics Handbook https://www.itl.nist.gov/div898/handbook/pmd/section4/pmd431.htm

[2] B. Chen, S. Zhao, P. Zhu and J. C. Principe, "Quantized Kernel Least Mean Square Algorithm," in IEEE Transactions on Neural Networks and Learning Systems, vol. 23, no. 1, pp. 22-32, Jan. 2012, doi: 10.1109/TNNLS.2011.2178446.

[3] B. Chen, S. Zhao, P. Zhu and J. C. Príncipe, "Quantized Kernel Recursive Least Squares Algorithm," in IEEE Transactions on Neural Networks and Learning Systems, vol. 24, no. 9, pp. 1484-1491, Sept. 2013, doi: 10.1109/TNNLS.2013.2258936.

[4] B. Chen, X. Liu, H. Zhao, J. C. Principe, "Maximum Correntropy Kalman Filter", in Automatica, vol. 76, pp. 70-77, Feb. 2017, doi: 10.1016/j.automatica.2016.10.004

[5] C. Bishop, Pattern Recognition and Machine Learning. Springer: New York, 2006.