Water Quantity Prediction Using Least Squares Support Vector Machines (LS-SVM) Method

Nian ZHANG, Charles WILLIAMS
Department of Electrical and Computer Engineering, University of the District of Columbia
4200 Connecticut Ave. NW, Washington, DC, 20008, USA
nzhang@udc.edu, charles.williams4@udc.edu

and

Pradeep BEHERA
Department of Department of Engineering, Architecture and Aerospace Technology, University of the District of Columbia
4200 Connecticut Ave. NW, Washington, DC, 20008, USA
pbehera@udc.edu

ABSTRACT

The impact of reliable estimation of stream flows at highly urbanized areas and the associated receiving waters is very important for water resources analysis and design. We used the least squares support vector machine (LS-SVM) based algorithm to forecast the future streamflow discharge. A Gaussian Radial Basis Function (RBF) kernel framework was built on the data set to optimize the tuning parameters and to obtain the moderated output. The training process of LS-SVM was designed to select both kernel parameters and regularization constants. The USGS real-time water data were used as time series input. 50% of the data were used for training, and 50% were used for testing. The experimental results showed that the LS-SVM algorithm is a reliable and efficient method for streamflow prediction, which has an important impact to the water resource management field.

Keywords: Water Quantity Prediction, Least Squares Support vector Machine.

1. INTRODUCTION

The Potomac River plays an important role in watershed and river system health, and the physical, chemical, and biological viability of the river system [1]. Development, when not done in a sustainable fashion, causes many of the diseases that face the Potomac watershed. Stormwater picks up nutrients, sediment and chemical contaminants as it flows across roads, yards, farms, golf courses, parking lots and construction sites. This polluted runoff travels into storm drains and local waterways that eventually drain into the Chesapeake Bay. Development activities like clearing vegetation, mass grading, removing and compacting soil, and adding impervious surfaces have increased stormwater runoff in the Chesapeake Bay watershed.

It has been recognized that urban stormwater pollution can be a large contributor to the water quality problems of many receiving waters. Depending upon the type of sewer system the stormwater runoff transports a wide spectrum of pollutants to local receiving waters through combined sewer overflows (CSOs) and/or stormwater discharges. Stormwater pollution is one of most important issues the District of Columbia faces. The downtown core of the District is serviced by combined sewer system. The development of the District over the years has increased its impervious area significantly which combines with inadequate drainage capacity of the sewer system results in CSOs and stormwater discharges to the Anacostia River, Potomac River and Rock Creek.

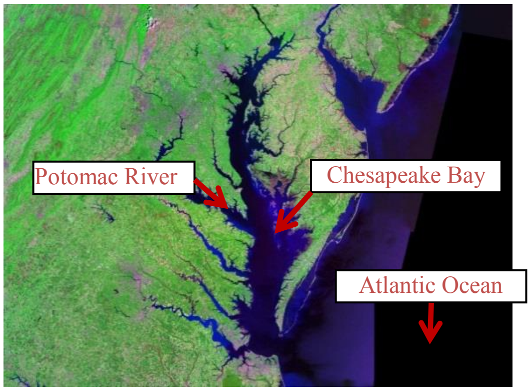


Fig. 1. Satellite landsat photo of Potomac River watershed. The Potomac river is a key entry point to the Chesapeake Bay for millions living in or visiting metropolitan Washington.

The study area will focus on the Four Mile Run at Alexandria, VA. The Four Mile Run is 9.2 miles long, and is a direct tributary of the Potomac River, which ultimately carries the water flowing from Four Mile Run to the Chesapeake Bay, as shown in Fig. 2. The stream passes from the Piedmont through the fall line to the Atlantic Coastal Plain, and eventually empties out into the Potomac River. Potomac River was determined to be one of the most polluted water bodies in the nation mainly due to the CSOs and stormwater discharges and wastewater treatment plant discharges. In addition, because of the highly urbanized nature of the Four Mile Run watershed, the neighborhoods and businesses adjacent to this portion of the run were subjected to repeated flooding, beginning in the 1940s. Therefore, the flood-control solutions are the major concern. Runoff prediction would provide a promising solution for flood-control.

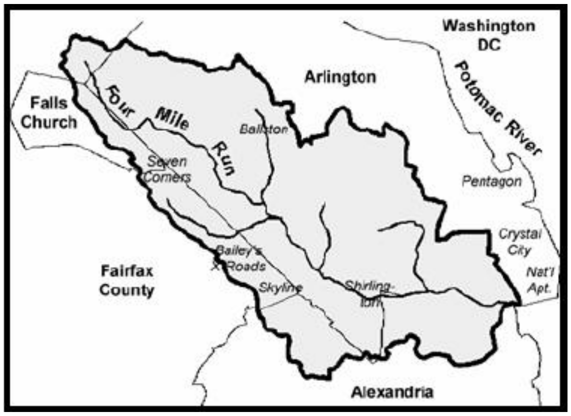


Fig. 2. Four Mile Run at Alexandria, VA is a nine-mile long stream located in a highly urbanized area in Northern Virginia. It is a direct tributary of the Potomac River, which ultimately carries the water flowing from Four Mile Run to the Chesapeake Bay.

Given the existing flow conditions of Potomac River, there is need to analyze the flow conditions at specific locations for future flow, specifically streamflow rate, and a reliable estimate under changing climactic conditions.

To resolve the above problems, it is extremely important to investigate state-of-the-art computational intelligence with the potential for higher rates for urban runoff forecast. Based on the fact that support vector machine has very successfully applications on the time series prediction problems [17], and because time series prediction is a generalized form of runoff quantity prediction, we expect this method will also work the best for the runoff prediction problem.

This paper is organized as follows. In Section II, the principle scheme and the method of Least Squares Support Vector Machines (LS-SVM) are illustrated. In Section III, the initial test on the function estimation is implemented. The USGS time series data are briefly introduced. The water data are briefly introduced. The experimental results of LS-SVM predictions on the water data are demonstrated. In Section IV, the conclusions are given.

2. METHOD

Support Vector Machines (SVMs) are a powerful kernel based statistical learning methodology for the solving problems of nonlinear classification, pattern recognition and function estimation [3]. Least Squares Support Vector Machines (LS-SVM) are an advanced version of the standard SVMs which incorporates unsupervised learning and recurrent networks. Recent developments of LS-SVM are especially relevant to the fields of time series prediction, kernel spectral clustering, and data visualization [4]-[13]. The preliminary results show that the LS-SVM modeling method is promising for time series prediction, thus we want to study the present a current LS-SVM toolbox run through Matlab to implement a number of LS-SVM algorithms.

Support Vector Machines are a new and potential data classification and regression instrument. The basic idea of SVM is based on Mercer core expansion theorem which maps sample space to a high dimension or even unlimited dimension feature space (Hilbert space) via nonlinear mapping φ. And it will boil algorithm which searches for optimal linear regression hyper plane down to a convex programming problem of solution of a convex restriction condition. And it will also obtain overall situation optimum solution so as to use the method of linear learning machine in feature space to solve the problem of high-degree nonlinear regression in sample space [14].

The principles of SVM can be summarized by Fig. 3 as follows:

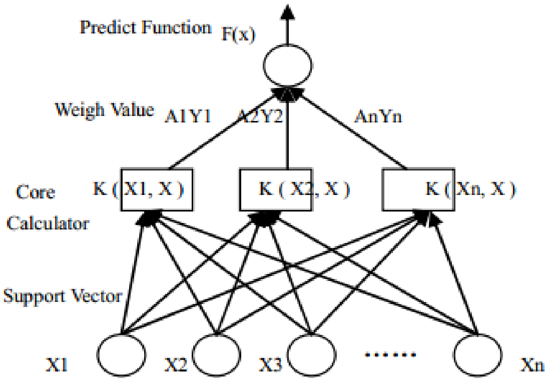


Fig 3. Principle scheme of Support Vector Machine.

In Fig. 3, n input support vectors are in the first layer and the second layer is nonlinear operation of N support vectors, that is, the core operation. For nonlinear problems, assume sample to be n-dimension vector, then in one certain domain, N samples and their values can be expressed as:

$(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N) \in R^n \times R$ (1)

Firstly, a nonlinear mapping ψ(·) is used to map samples from former space R^n to feature space:

$\psi(x) = (\emptyset(x_1), \emptyset(x_2), \dots \emptyset(x_N)$ (2)

Then, in this high-dimension feature space, optimal decision function:

$y(x) = w\emptyset(x) + b$ (3)

is established. In this function, *w* is a weighed value vector and *b* is a threshold value. In this way, nonlinear prediction function is transformed to linear prediction function in high-dimension feature space. As development and improvement of classical SVM, Least Squares Support Vector Machine (LSSVM) defines a cost function which is different from classical SVM and changes its inequation restriction to equation restriction. As a result, the solution process becomes a solution of a group of equations which greatly accelerates the solution speed [15]. In Least Squares Support Vector Machines, the problem of optimization is described as follows:

$$\min_{w,b,\varepsilon} L\,(w, b, \varepsilon) = \tfrac{1}{2}\|w\|^2 + \tfrac{c}{2}\textstyle\sum_{i=1}^{l} \varepsilon_i^2 \quad (4)$$

(4)

Such that: $y_i = w^t\emptyset(x_i) + b + \varepsilon_i$(i=1,2,…,l)

The extreme point of *Q* is a saddle point, and differentiating *Q* can provide the formulas as follows, using Lagrangian multiplier method to solve the formulas:

$$\frac{\partial Q}{\partial w} = w - \textstyle\sum_{i=1}^{l} \propto_i \emptyset(x_i) = 0 \qquad (5)$$

$$\frac{\partial Q}{\partial b} = -\sum_{i=1}^{l} \propto_i = 0$$

$$\frac{\partial Q}{\partial \propto} = w^T - \emptyset(\mathrm{x_i}) + \mathrm{b} + \varepsilon_i - y_i = 0$$

$$\frac{\partial Q}{\partial \varepsilon_i} = C\varepsilon_i - \propto_i = 0$$

From formulas above:

$$\frac{1}{2}\sum_{i=1}^{l} \propto_i \emptyset(x_i)\sum_{j=1}^{l} \propto_j \emptyset(x_j) + \frac{1}{2C}\sum_{i=1}^{l} \propto_i^2 + b\sum_{i=1}^{l} \propto_i = \sum_{i=1}^{l} \propto_i y_i \qquad (6)$$

The formula above can be expressed in matrix form:

$$\begin{bmatrix} 0 & e^T \\ e & \Omega + C^{-1}I \end{bmatrix} (l+1)(l+1) \begin{bmatrix} b \\ \propto \end{bmatrix} = \begin{bmatrix} 0 \\ Y \end{bmatrix} \qquad (7)$$

In this equation,

$$e = [1, \dots, 1]_x^T$$
$$\Omega_{ij} = K(x_i, x_j) = \emptyset(x_i)^T \emptyset(x_j) \qquad (8)$$

Formula (7) is a linear equation set corresponding to the optimization problem and can provide us with α and b. Thus, the prediction output decision function is:

$$\bar{y}(x) = \sum_{i=1}^{l} \propto_i K(x_i x) + b \qquad (9)$$

where $K(x_i, x)$ is the core function.

We are ultimately using the LS-SVM method to calculate and predict the USGS water data, specifically using time-series data prediction. After loading the data into Matlab, we first build the training and testing sets from the data. Next we cross-validate based upon a feed-forward simulation on the validation set using a feed-forwardly trained model. This will supply us with the tuning parameters: γ (gamma) which is the regularization parameter and σ2 (sigma squared) or the squared bandwidth. The tuning parameters were found by using a combination of coupled simulated annealing (CSA) and a standard simplex method. The CSA finds good starting values and these values were passed to the simplex method in order to fine tune the result. One of the parameters, γ is the regularization parameter, determining the trade-off between the training error minimization and smoothness. The other parameter, σ represents the squared bandwidth. Once the parameters are calculated, we are able to plot the function estimation or use the predict function to predict future values of the data. By using only a subset of the total data available, we can compare the predictions against real values to see how accurate the prediction is.

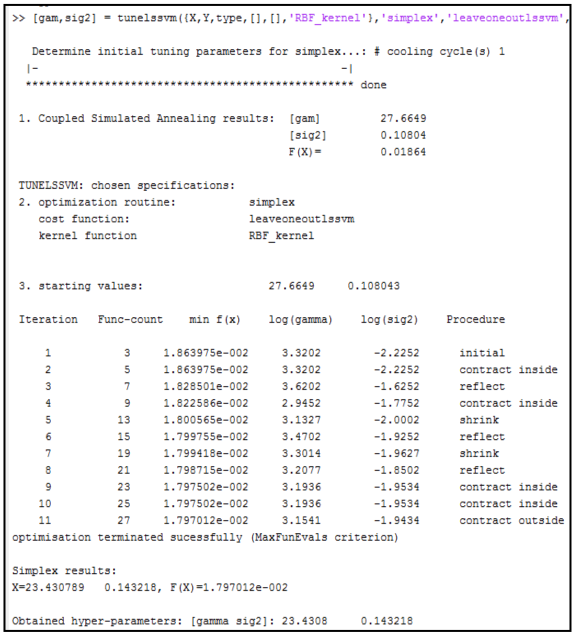
3. EXPERIMENTAL RESULTS

Initial Test on the Function Estimation

Because LS-SVM time series prediction uses function estimation in the parameter tuning, training procedures and prediction, it is important to test the function estimation ability. For this purpose, we set up a simple nonlinear system model to estimate the target function, Y, which can be implemented with only a few lines of code [16].

```
X = linspace(-1,1,50)';
Y = (15*(X.^2-1).^2 .*X.^4).*exp(-X)+
        normrnd(0,0.1,length(X),1);
type = 'function estimation';
[gam,sig2] = tunelssvm({X,Y,type,[],[],'RBF_kernel'},
        'simplex','leaveoneoutlssvm',{'mse'});
[alpha,b] = trainlssvm({X,Y,type,gam,sig2,'RBF_kernel'});
plotlssvm({X,Y,type,gam,sig2,'RBF_kernel'},{alpha,b});
```

In this case, we generate sample data sets X and Y, with X being a linearly spaced set of fifty values from -1 to 1, and Y being a quasi-random exponential function. After setting LS-SVM processing type to 'function estimation,' we then tune the coefficients using the tunelssvm command, passing it the values for X, Y, type, and other settings for the computation. The tunelssvm function outputs the two tuning hyper-parameters, gamma and sig2. After the algorithm is done tuning, the program generates the following output and specifically values for the hyper-parameters, as shown in Fig. 4.



```
>> [gam,sig2] = tunelssvm({X,Y,type,[],[],'RBF_kernel'},'simplex','leaveoneoutlssvm',

  Determine initial tuning parameters for simplex...: # cooling cycle(s) 1
 |-                                               -|
 ************************************************** done

 1. Coupled Simulated Annealing results:  [gam]         27.6649
                                          [sig2]        0.10804
                                          F(X)=         0.01864

 TUNELSSVM: chosen specifications:
 2. optimization routine:           simplex
    cost function:                  leaveoneoutlssvm
    kernel function                 RBF_kernel


 3. starting values:                   27.6649     0.108043

 Iteration   Func-count    min f(x)    log(gamma)    log(sig2)    Procedure

     1           3      1.863975e-002    3.3202        -2.2252       initial
     2           5      1.863975e-002    3.3202        -2.2252       contract inside
     3           7      1.828501e-002    3.6202        -1.6252       reflect
     4           9      1.822586e-002    2.9452        -1.7752       contract inside
     5          13      1.800565e-002    3.1327        -2.0002       shrink
     6          15      1.799755e-002    3.4702        -1.9252       reflect
     7          19      1.799418e-002    3.3014        -1.9627       shrink
     8          21      1.798715e-002    3.2077        -1.8502       reflect
     9          23      1.797502e-002    3.1936        -1.9534       contract inside
    10          25      1.797502e-002    3.1936        -1.9534       contract inside
    11          27      1.797012e-002    3.1541        -1.9434       contract outside
optimisation terminated sucessfully (MaxFunEvals criterion)

Simplex results:
X=23.430789   0.143218, F(X)=1.797012e-002

Obtained hyper-parameters: [gamma sig2]: 23.4308     0.143218
```

Fig. 4 Output generated from tunelssvm function.

After the tuning shown above, the trainlssvm command is called to generate alpha and b, which are then both input into the plotlssvm function to generate the following plot representing the LS-SVM estimation of the data set. The estimated function, Y in terms of the inputs is shown in Fig. 5.

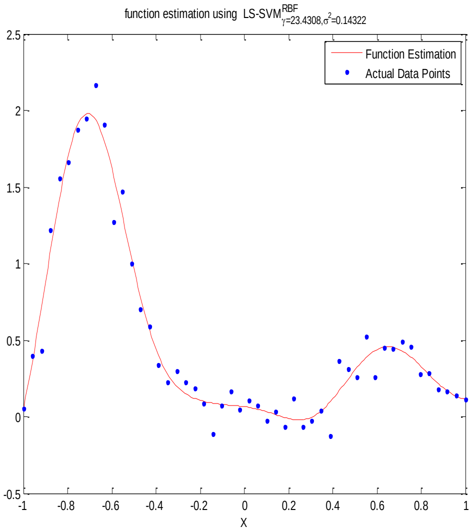


Fig. 5 Plot of Y vs. X, where blue dots represent actual data points and the red line is the prediction.

As shown in Fig. 5, the LS-SVM model is performing very well on estimating a function to a random set of data. This proves that the function estimation works effectively.

Next we used the LS-SVM command to calculate error bars based on a sample training data set. After loading a randomized *sinc* function into Y, and a linearly spaced vector into X, we implement the following code to both calculate the LS-SVM regression and error bars, a representation in the error of each predicted value. The command 'bay_error' bar is sent the variables involved after LS-SVM processing and returns a figure that graphically depicts the accuracy of the prediction:

```
[Yp,alpha,b,gam,sig2] = lssvm(X,Y,type);
sig2e = bay_errorbar({X,Y,type, gam, sig2},'figure');
```

After executing the above code, two plots are generated. The first is the plot of the actual data points (blue dots) compared to the function estimated to predict those points (red line), as shown in Fig. 6.

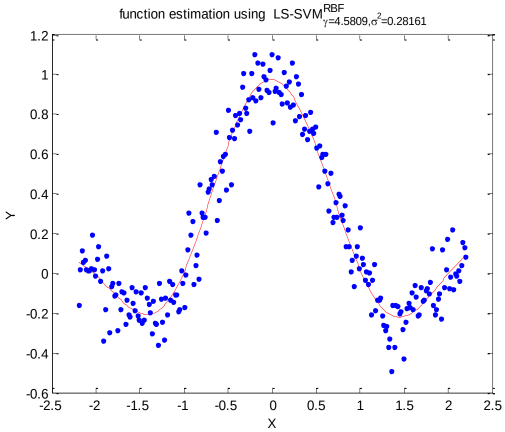


Fig. 6 LS-SVM function estimation for the randomized *sinc* function.

The second shows the error bars, or the algorithm's confidence in its own predictions in Fig. 7. The black line covered by black '+' signs is the predicted function and data, while the red dotted lines represent the 95% error bar; basically the area in which 95% of the data resides.

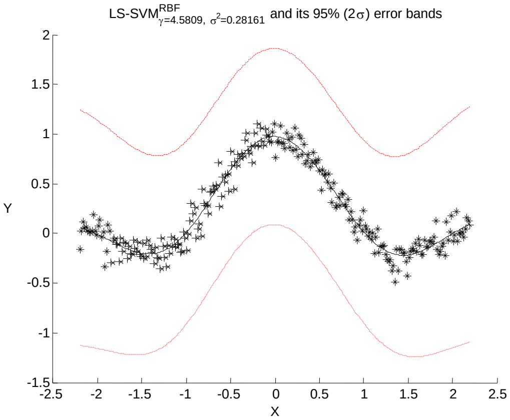


Fig. 7 Data points and prediction (black) with 95% error bars (red).

Time Series Prediction

We used a sample set of about 35,000 data points, all taken at a regular time intervals. We examined both gage height and discharge. The discharge is the volume of water flowing past a certain point in a water-flow. For example, the amount of cubic feet passing through a drain per second is a measure of discharge. Gage height is simply the height of water at a certain point, like the level of the Potomac River measured at Key Bridge. Initially we are only looking to input one of these variables into the LS-SVM algorithm, but in the future it would probably prove to increase prediction accuracy to include the use of both variables at once, the more data input into the system often translates into better results. Fig. 8 is a plot of the discharge vs. time.

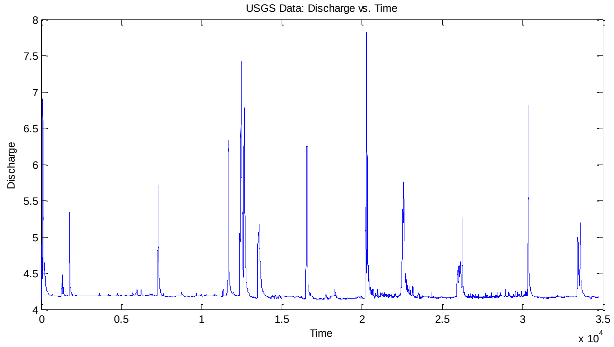


Fig. 8 Plot of entire discharge data set vs. time.

These discharge values vary significantly over time- the baseline is at around 4 on the Y-axis, with peaks reaching 8, with very little repetition to the pattern, making it more difficult to predict future values.

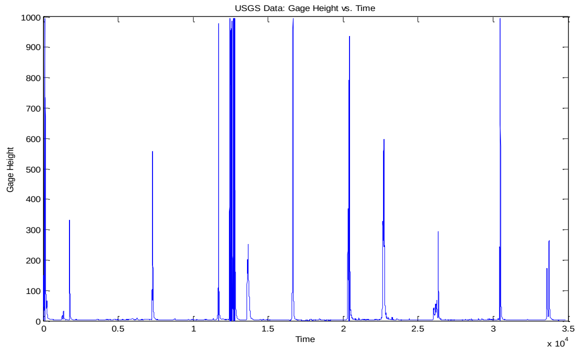


Fig. 9 Plot of entire gage height data set vs. time.

The gage height plot contains peaks in similar timeframes as the discharge plot, likely due to large rainfall events or local flooding, as shown in Fig. 9.

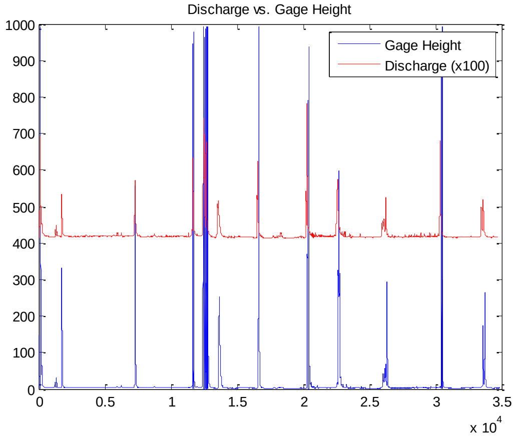


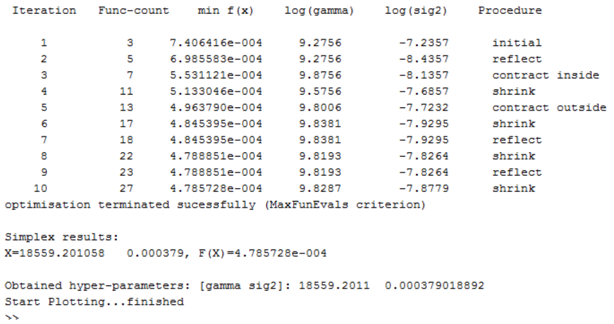
Fig. 10 USGS gage height vs. discharge (scaled) for side-by-side comparison.

When looking at both gage height and discharge on the same plot as shown in Fig. 10 (with discharge multiplied by 100 so it is visible on the same scale), you can see that they closely correlate with each other. Again this is likely due to the patterns in local weather, specifically precipitation. Sending only one of these variables to the LS-SVM function will produce good predictions, but if we can go further and implement a two input algorithm, that will analyze both discharge and gage height at the same time, this will definitely increase the accuracy of predictions.

To test LS-SVM predictions on the water data we selected a random portion of the discharge data, 500 data points from the original sample of about 35,000. The LS-SVM algorithm is known to be very resource efficient, meaning it can process large amounts of data without using too much processor or memory power. By even this algorithm would take a very long time to process more than a few thousand data points.

The following plot was processed in a similar way to the examples above, but instead of using a random function for Y, we utilize a random selection of points from the discharge dataset. In this case X is just the time axis, as in all time-series datasets, this data was taken at regular intervals so the values used for X increment one by one.

After loading the data, we tune the hyper-parameters gamma and sigma squared with the tunelssvm command, which generates the following values after 10 iterations, as shown in Fig. 11.



```
 Iteration   Func-count    min f(x)     log(gamma)     log(sig2)     Procedure

     1            3      7.406416e-004     9.2756        -7.2357      initial
     2            5      6.985583e-004     9.2756        -8.4357      reflect
     3            7      5.531121e-004     9.8756        -8.1357      contract inside
     4           11      5.133046e-004     9.5756        -7.6857      shrink
     5           13      4.963790e-004     9.8006        -7.7232      contract outside
     6           17      4.845395e-004     9.8381        -7.9295      shrink
     7           18      4.845395e-004     9.8381        -7.9295      reflect
     8           22      4.788851e-004     9.8193        -7.8264      shrink
     9           23      4.788851e-004     9.8193        -7.8264      reflect
    10           27      4.785728e-004     9.8287        -7.8779      shrink
optimisation terminated sucessfully (MaxFunEvals criterion)

Simplex results:
X=18559.201058   0.000379, F(X)=4.785728e-004

Obtained hyper-parameters: [gamma sig2]: 18559.2011  0.000379018892
Start Plotting...finished
>>
```

Fig. 11 Output generated from tunelssvm command operating on USGS water data.

Once the hyper-parameters are tuned, we just assigned alpha and b with the trainlssvm command and finally use the predict function to make a prediction for the next set of values. The final plot generated is shown in Fig. 12, with real USGS discharge datapoints shown in blue while the LS-SVM prediction is the red line.

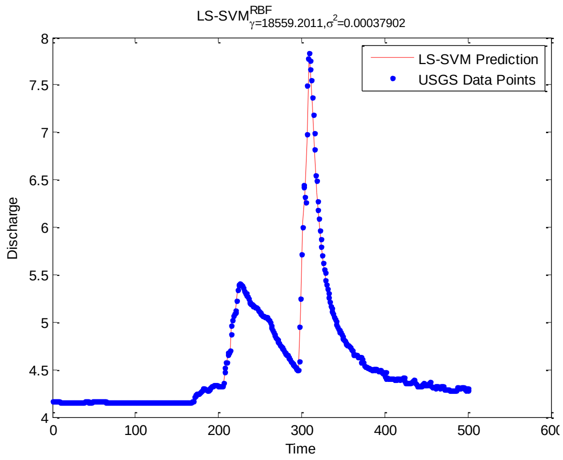


Fig. 12 USGS discharge data set (blue dots) and LSSVM prediction (red).

4. CONCLUSIONS

In this paper, the least squares support vector machine (LS-SVM) based algorithm to forecast the future streamflow discharge. A Gaussian Radial Basis Function (RBF) kernel framework was built on the data set to optimize the tuning parameters and to obtain the moderated output. The training process of LS-SVM was designed to select both kernel parameters and regularization constants. The USGS real-time water data were used as time series input. 50% of the data were used for training, and 50% were used for testing.

The experimental results demonstrated that the proposed LS-SVM based predictive model and the training algorithm ensure an accurate prediction of LS-SVM, and by association any natural measurable system. In addition, this provides an excellent prediction method for the time series data, and if correctly implemented can be an invaluable tool in predicting natural weather events. Even outside of storm-water, this algorithm could be very useful to engineers who wish to develop a resource efficient prediction model for any quantifiable data set, i.e. solar radiation, global warming, glacier melting, and more.

5. ACKNOWLEDGEMENT

The authors would like to express thanks to the University of the District of Columbia STEM Center (NSF/HBCU-UP/HRD-0928444) grant and DC Water Resources Research Institute (WRRI) Grant.

6. REFERENCES

[1] Potomac Conservancy, **State of the Nation's River**, Potomac Watershed. 2007. Available: http://www.potomac.org.

[2] N.I. Sapankevych, Ravi Sankar, "Time Series Prediction Using Support Vector Machines: A Survey," **IEEE Computational Intelligence Magazine**, vol. 4, no. 2, pp. 24-38, 2009.

[3] **Support Vector Machines Toolbox**, http://www.esat.kuleuven.be/sista/lssvmlab/.

[4] Z. Liu, X. Wang, L. Cui, X. Lian, J. Xu, "Research on Water Bloom Prediction Based on Least Squares Support Vector

Machine," **2009 WRI World Congress on Computer Science and Information Engineering**, vol.5, pp. 764 - 768, 2009.

[5] Y. Xiang, L. Jiang, "Water Quality Prediction Using LS-SVM and Particle Swarm Optimization," **Second International Workshop on Knowledge Discovery and Data Mining**, 2009. pp. 900- 904, 2009.

[6] X. Wang, J Lv, D. Xie, "A hybrid approach of support vector machine with particle swarm optimization for water quality prediction," **International Conference on Computer Science and Education (ICCSE)**, pp. 1158- 1163, 2010.

[7] W. Liu, K. Chen; L. Liu, "Prediction model of water consumption using least square support vector machines optimized by hybrid intelligent algorithm," **2011 Second International Conference on Mechanic Automation and Control Engineering (MACE)**, pp. 3298- 3300, 2011.

[8] L. Yu, X. Wang, Q. Ming, H. Mu, Y. Li, "Application and comparison of several modeling methods in spectral based water quality analysis," **2011 30th Chinese Control Conference (CCC)**, pp. 5227- 5230, 2011.

[9] L. Liang, F. Xie, "Applied research on wastewater treatment based on least squares support vector machine," **2011 International Conference on Remote Sensing, Environment and Transportation Engineering (RSETE)**, pp. 4825- 4827, 2011.

[10] X. Zhang, S. Wang, Y. Zhao, "Application of support vector machine and least squares vector machine to freight volume forecast," **2011 International Conference on Remote Sensing, Environment and Transportation Engineering (RSETE)**, pp. 104- 107, 2011.

[11] R.J. Liao, J.P. Bian, L.J. Yang, S. Grzybowski, Y.Y. Wang, J. Li, "Forecasting dissolved gases content in power transformer oil based on weakening buffer operator and least square support vector machine–Markov," **Generation, Transmission & Distribution**, IET, vol. 6, no. 2, pp. 142- 151, 2012.

[12] L. Hou, Q. Yang, J. An, "An Improved LSSVM Regression Algorithm," **International Conference on Computational Intelligence and Natural Computing**, vol. 2, pp. 138- 140, 2009.

[13] X. Zhang, Y. Zhao, S. Wang, "Reliability prediction of engine systems using least square support vector machine," **2011 International Conference on Electronics, Communications and Control (ICECC)**, pp. 3856- 3859, 2011.

[14] T. A. Stolarski. **System for wear prediction in lubricated sliding contacts**. Lubrication Science, 1996, 8 (4): 315 -351.

[15] Suykens J A K,Vandewalle J . Least squares support vector machine classifiers. **Neural Processing Letter**, 1999, 9(3):293-300.

[16] De Brabanter K., Karsmakers P., Ojeda F., Alzate C., De Brabanter J., Pelckmans K., De Moor B., Vandewalle J., Suykens J.A.K., **LS-SVMlab Toolbox User's Guide Version 1.8**, Internal Report 10-146, ESAT-SISTA, K.U.Leuven (Leuven, Belgium), 2010.