AN APPLICATION OF LEAST SQUARES SUPPORT VECTOR REGRESSION WITH REGROUPING PARTICLE SWARM OPTIMZATION

By

CHAOHUI SUN

Bachelor of Science in Computer science

University of Tulsa

Tulsa, Oklahoma

1996

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
July, 2011

AN APPLICATION OF LEAST SQUARES SUPPORT

VECTOR REGRESSION WITH REGROUPING

PARTICLE SWARM OPTIMZATION

Thesis Approved:

Dr. Douglas R. Heisterkamp

Thesis Adviser

Dr. Debao Chen

Dr. Johnson Thomas

Dr. Mark E. Payton

Dean of the Graduate College

TABLE OF CONTENTS

Chapter Page

I. INTRODUCTION 1

Objective 2

II. BACKGROUND 3

Least squares support vector regression 4
Regrouping particle swarm optimization 6

III. METHODOLOGY AND PROPOSAL 10

IV. EXPERIMENTAL FINDINGS 12

Comparison of results 16

V. CONCLUSION 20

REFERENCES 21

APPENDICES 24

LIST OF TABLES

Table Page

I. results of errors on each model……………………………………………………15

II. Model parameter settings ………………………………………………………….24

III. Hyper-parameters and lag value obtained for each model …………………25

LIST OF FIGURES

Figure Page

1. RegPSO + LSSVR model ……………………………………………………11
2. Selecting validation set from the left…………………………………………13
3. Selecting validation set from the right ………………………………………..14
4. RMSE w.r.t. each model for sunspot dataset 17
5. MAE w.r.t. each model for sunspot dataset 17
6. Maximum errors w.r.t. each model for sunspot dataset 18
7. RMSE w.r.t. each model for sulfuric acid production dataset 18
8. Maximum errors w.r.t. each model for sulfuric acid production dataset 19
9. MAE w.r.t. each model for sulfuric acid production dataset 19

I. Next 12 month sulfuric acid production on testing dataset................................25
II. Plot of all testing points for sulfuric acid production dataset 26
III. Plot of all training points for sulfuric acid production dataset...........................26
IV. Next 12 years of sunspot forecasting on testing dataset 27
V. Plot of all testing points for sunspots dataset..27
VI. Plot of all training points for sunspots dataset 28

CHAPTER I

INTRODUCTION

Mankind has learned to observe and record information around us in minute details overtime, and the enormity of data we have in any specific field today that it can even overwhelm experts. In order to learn and generalize information from these data, computer science has ventured into the realm of experts without the prerequisite expertise on specific subjects thanks to the help of machine learning. In the realm of short term forecasting, popular linear models such as the Box and Jenkins' ARIMA [1] (Autoregressive Integrated Moving Average) and Engle's ARCH [2] (Autoregressive Conditional Hetroskedasticity) have been adopted by many including the US Census Bureau. As we are living in a highly integrated and globalized world, the "butterfly effect" is no longer limited to describing our weather system; economic and social changes in one part of the world would have inevitable effect on all the rest. These complicated relationships make nonlinear methods such as varieties of artificial neural networks an attractive alternative. Furthermore, the proposal of the Support Vector Regression (SVR) [3], SVR has also been studied and applied to short term forecasting with success.

Empirical studies have shown that Back Propagation Neural Networks (BPNN) can achieve better results than ARIMA in forecasting [4], and SVR can give better results than BPNN [5]. However, as learning and generalization performance of SVR for time series data is greatly affected by the hyper parameters it used and the proper formation of the time series into relationship matrixes, it became important to select a set of optimal parameters and to properly transform the time series.

Objective

The objective of this study is to obtain good performance on short term forecasting with time series using Least Squares Support Vector Regression (LSSVR) [11]. In order to do so, one will need to select an optimal input data set for the SVR and optimal kernel parameters /hyper-parameters for SVR. As there are no known methods that can calculate these values, a novel method is proposed here to optimize both input data set and hyper-parameters for SVR at the same time with a hyper version of PSO -- Regrouping Particle Swarm Optimization (RegPSO) [6]. Real world data will be used to determine the performance of the proposed method versus that of a known model that uses LSSVR with standard Particle Swarm Optimization (PSO) [12] for LSSVM hyper-parameters and Average Mutual information (AMI) [9] for lag selection. A third model that uses AMI for lag selection, and grid search for hyper-parameters selection is also included for the purpose of establishing a baseline.

CHAPTER II

BACKGROUND

One can treat the hyper-parameters' fine tuning of support vector like a constrained optimizing problem. There have been many different approaches in resolving this problem; they range from grid search or random walks to gradient search or population base search algorithms like genetic algorithm [7], and in this case particle swarm optimization [8]. Among all these methods, PSO has been found to be more accurate and less computationally intensive [10]. However, Standard PSO does have a drawback as premature coverage on local minimum, and various versions of PSO have been proposed to resolve this problem. Among those variants, Regrouping PSO (Reg-PSO) has been shown to have better performance over others with synthetic data [6]. Taking this advancement into consideration, this study hopes to investigate the applicability of combing of REG-PSO with LSSVR method on real world data.

The principal methodologies that are employed in this paper are Regrouping Particle Swarm Optimization (RegPSO) and Least Squares Support Vector Regression (LSSVR), both of which will be explained briefly in this chapter.

Least Squares Support Vector Regression

LSSVR is a least square variant of the standard support vector regression (SVR) [3], and it was credited to Suykens [11]. LSSVR introduces an equality constraint to reduce the computational complexity and enhance the generalization performance over SVR for large databases. Detailed theory and proof of these algorithms are listed in reference [6] and [11].

Given a training set of N data points

$$s = \{(x_i, y_i) \mid x_i \in R^m, y_i \in R\}_{i=1}^{N}$$

Then one will need to construct the best regression of the following form:

$$f(x, \omega) = \omega^T \varphi(x) + b \qquad (1)$$

Taking the structural risk under consideration, LSSVR uses the squared loss function, and then the original problem can be reformulated as optimizing the following function:

$$\min_{\boldsymbol{\omega}, \boldsymbol{b}, \boldsymbol{\varepsilon}} \boldsymbol{J}(\boldsymbol{\omega}, \boldsymbol{\varepsilon}) = \frac{1}{2} \boldsymbol{\omega}^{\boldsymbol{T}} \boldsymbol{\omega} + \frac{1}{2} \gamma \sum_{i=1}^{N} \varepsilon_i^2 \qquad (2)$$

Subject to: $y_i = \omega^T \varphi(x_i) + b + \varepsilon_i, \mathrm{i} = 1, \dots, \mathrm{N}$ (3)

where γ is a positive constant. One can then obtain a corresponding Lagrange function as:

$$L(\omega, b, \varepsilon, \alpha) = \frac{1}{2} \omega^T \omega + \gamma \sum_{i=1}^{N} \varepsilon_i^2 - \sum_{i=1}^{N} \alpha_i \left(\omega^T \varphi(x_i) + b + \varepsilon_i - y_i\right) \qquad (4)$$

where α_i are the Lagrange multipliers; the optimal conditions per Karush-Kuhn-Tucker (KKT) are defined as:

$$\begin{cases} \dfrac{\partial L}{\partial \omega} = 0 \to \omega = \sum_{i=1}^{N} \alpha_i \varphi(x_i) \\ \dfrac{\partial L}{\partial b} = 0 \to \sum_{i=1}^{N} \alpha_i = 0 \\ \dfrac{\partial L}{\partial \varepsilon_i} = 0 \to a_i = \gamma\, \varepsilon_i, i = 1,\ldots,N, \\ \dfrac{\partial L}{\partial \alpha_i} = 0 \to \omega^T \omega\varphi(x_i) + b + \varepsilon_i - y_i, i = 1,\ldots,N \end{cases} \tag{5}$$

After eliminating □ and ω from (5), and applying Mercer's condition of

$\Omega_{ij} = K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$,

the solution is given by the following linear equations:

$$\begin{bmatrix} 0 & \vec{1}^T \\ \vec{1} & \Omega + \gamma^{-1} I \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ y \end{bmatrix} \tag{6}$$

where $y = [y_1, \ldots, y_N]^T$, $\alpha = [\alpha_1, \ldots, \alpha_N]^T$ and $\vec{1} = [1, \ldots, 1]$.

The regression function for LSSVR model will take the form as follows:

$$f(x) = \sum_{i=1}^{N} \alpha_i K(x, x_i) + b \tag{7}$$

Let $A = \Omega + \gamma^{-1} I$, then α_i and b can be obtained with the following equations:

$$b = \frac{\vec{1}^T A^{-1} y}{\vec{1}^T A^{-1} \vec{1}} \ ,$$

$$\alpha = A^{-1}(y - b\overrightarrow{1)}\ ,$$

$K(x, x_i)$ represents the kernel function that maps the input space into high-dimensional feature space. Since Radial Basis Function (RBF) is adopted as kernel function for this study, then it will be represented as:

$$K(x, x_i) = \exp\left\{-\frac{\|x - x_i\|^2}{2\sigma^2}\right\} \qquad (8)$$

Regrouping Particle Swarm Optimization

RegPSO is an improved version of the original PSO, which was credited to Kennedy, Eberhart and Shi back in 1995 [12] [13]. Owing to its origin in simulation of social behaviors, PSO is a population based algorithm just like other evolutionary algorithms. However, the initial populations in PSO are constituent particles that not only represent the initial population in n-dimensional search space, but each particle is also representing a candidate solution to the n-dimensional problem. Each particle flies/searches through the n-dimensional space in search of an optimal solution to the problem, while sharing their current best known solution among the constituents; after each iteration, each particle will attempt to update their internal velocity and location based on the its current position in the search space with respect to the best known solution. Unlike most genetic algorithms, PSO doesn't have genetic operations such as crossover and mutation, which makes PSO an inexpensive heuristic optimizer. However, due to the lack of interaction between particles, the algorithm does have a tendency for premature convergence. In order to overcome this problem, many methods had been exploited and adopted to improve standard PSO, RegPSO is one of the recent techniques in doing so; it is based on the standard PSO with embedded auto-regrouping mechanism to reorganize the particles into a new search space when particles are found to be prematurely converged. RegPSO not only adopted F. Van den Bergh's *maximum swarm radius convergence detection technique* [14] to address the premature convergence problem of stand PSO, but also kept the required computation to a minimum. Hence, this method is chosen for the selection of LSSVR parameters γ and σ.

Given a cost function *f(x)*, then search space for the solution vector $\vec{x} \subset R^n$ is defined by

$$\Omega = [x_1^U \quad x_1^L] \times [x_2^U \quad x_2^L] \times \ldots \times [x_n^U \quad x_n^L] \subset R \qquad (9)$$

where x_k^U x_k^L are the upper and lower limits of the search space along dimension k.

With a swarm of size ***s***, the i-th particle has a position vector of $\overrightarrow{x_i}$ and a velocity vector of $\overrightarrow{v_i}$; Let ω be the static inertia weight chosen between [0,1], $c_1$be the cognitive acceleration coefficient, c_2 be the social acceleration coefficient; $\overrightarrow{r_1}$ and $\overrightarrow{r_2}$ be the random column vector that's between [0,1]; $\overrightarrow{p_i}$ be the personal best position vector and $\vec{g}$ be the global best position vector of the swarm, ε be the user defined stagnation threshold, and λ be the velocity clamping factor between [0.1, 0.5]

Then the algorithm can be described as:

For each new group do

- For each dimension k = 1, …, n do

  $$range_k(\Omega^r) = min(range_k(\Omega^0), \rho \max_{i \in \{1,\ldots,S\}} |x_{i,k}^{r-1} g_k^{r-1}|) \qquad (10)$$

  $$v_k^{max,r} = \lambda \cdot range_k(\Omega^r) \qquad (11)$$

  where $\rho = 6/(5\varepsilon)$,

  For each particle *i* = 1, ..., S do

Initialized velocities where $v_{i,k} \in [-v_k^{max,r}, v_k^{max,r}]$

- For each particle I = 1, …, S do
    - Initialize the particle's position $\overrightarrow{x_i}$ to be within boundaries defined by Ω^r
    - Initialize the particle's personal best known position to its initial position: $\overrightarrow{p_i} = \overrightarrow{x_i}$
- If r = 0 (e.g., prior to any regrouping)

$$\vec{g}(j) = \underset{p_i(j) \in p(j)}{\arg\min} f(\vec{p}_i(j))$$

- For each iteration j = 1, …,max iteration defined by user do
    - For each particle I = 1, …, S do
        - Update velocity as

$$\overrightarrow{v_i}(j+1) = \omega\overrightarrow{v_i}(j) + c_1\overrightarrow{r_1} \circ (\overrightarrow{p_i}(j) - \overrightarrow{x_i}(j)) + c_2\overrightarrow{r_2} \circ (\vec{g}(j) - \overrightarrow{x_i}(j))$$

        - Clamp velocity if needed
        - Update positions as

$$\overrightarrow{x_i}(j+1) = \overrightarrow{x_i}(j) + \overrightarrow{v_i}(j+1)$$

        - Update particle best known position as

$$\overrightarrow{p_i}(j) = \begin{cases} \overrightarrow{x_i}(j) & \text{if } f(\overrightarrow{x_i}(j)) < f(\overrightarrow{p_i}(j-1)) \\ \overrightarrow{p_i}(j-1) & \text{if } f(\overrightarrow{x_i}(j)) \geq f(\overrightarrow{p_i}(j-1)) \end{cases}$$

    - Update best known position for swarm as

$$\vec{g}(j) = \underset{p_i(j) \in p(j)}{\arg\min} f(\vec{p}_i(j))$$

    - Find the swarm radius as

$$\delta(j) = \max_{i \in \{1,\dots,s\}} \| \vec{x_i}(j) - \vec{g}(j) \|$$

where ||.|| is the Euclidean norm.

- o If user-defined number of function evaluation is reached or $\frac{\delta(j)}{\|\overrightarrow{range}(\Omega)\|} < \varepsilon$ (premature convergence is found)
  - ▪ regroup the swarm by updating
    - range of the search space

$$\text{range}_k(\Omega^r) = \min\left(\text{range}_k(\Omega^0), \rho \max_{i \in \{1,\dots,s\}} \left| x_{i,k}^{r-1} - g_k^{r-1} \right| \right)$$

$$\overrightarrow{range}(\Omega^r) = [range_1(\Omega^r), range_2(\Omega^r), \dots, range_m(\Omega^r)]$$

    - re-initialize the particle positions around the global best

$$\vec{x_i}(j) = \vec{g}^{r-1} + \overrightarrow{rand} \circ range(\Omega^r) - \frac{1}{2}\overrightarrow{range}(\Omega^r)$$

where $\overrightarrow{rand}$ is a random vector

    - maximum velocity for the new group is updated as

$$v_k^{max,r} = \lambda \cdot range_k(\Omega^r)$$

Terminate if maximum function evaluation for all groups is reached or the solution for the function is found.

CHAPTER III

METHODOLOGY AND PROPOSAL

In this study, the proposed model adopts RegPSO for parameter selection of the support vector – specifically, the Least Squares Support Vector. The parameters γ and σ of the LSSVR will become the first and second dimensions of the RegPSO model. Since the time series only contain observed values, the series must be reformatted into a matrix of features that contain enough resolution to infer the series while generating minimum amount of interference. In this paper, the number of feature selections of series is known as number of lags. While there are no known methods that can be applied to all series in selecting the optimal time lag value, many opted for a simple trial and error method [15]. Others employed average mutual information (AMI) [9]. For this study, the time series will be transformed according to

$$f_t(x) = f\left(f_t(x), f_{t-1}(x), f_{t-2}(x) \ldots, f_{t-n}(x)\right)$$

where n is the lag size of the series. Instead of looking for n with trial and error or AMI, it will become the last dimension of the RegPSO model. Hence each particle of the swarm will be represented by a three dimensional vector [γ, σ, lag], and the cost function for RegPSO will be the root mean squared error(RMSE) of the LSSVM obtained under cross-validation. As RegPSO has been proven to outperform other PSO methods with simulated data [6], it's reasonable to expect the proposed model to perform well even with real world data. The following figure 1 shows the flow chart of the proposed model in detail.

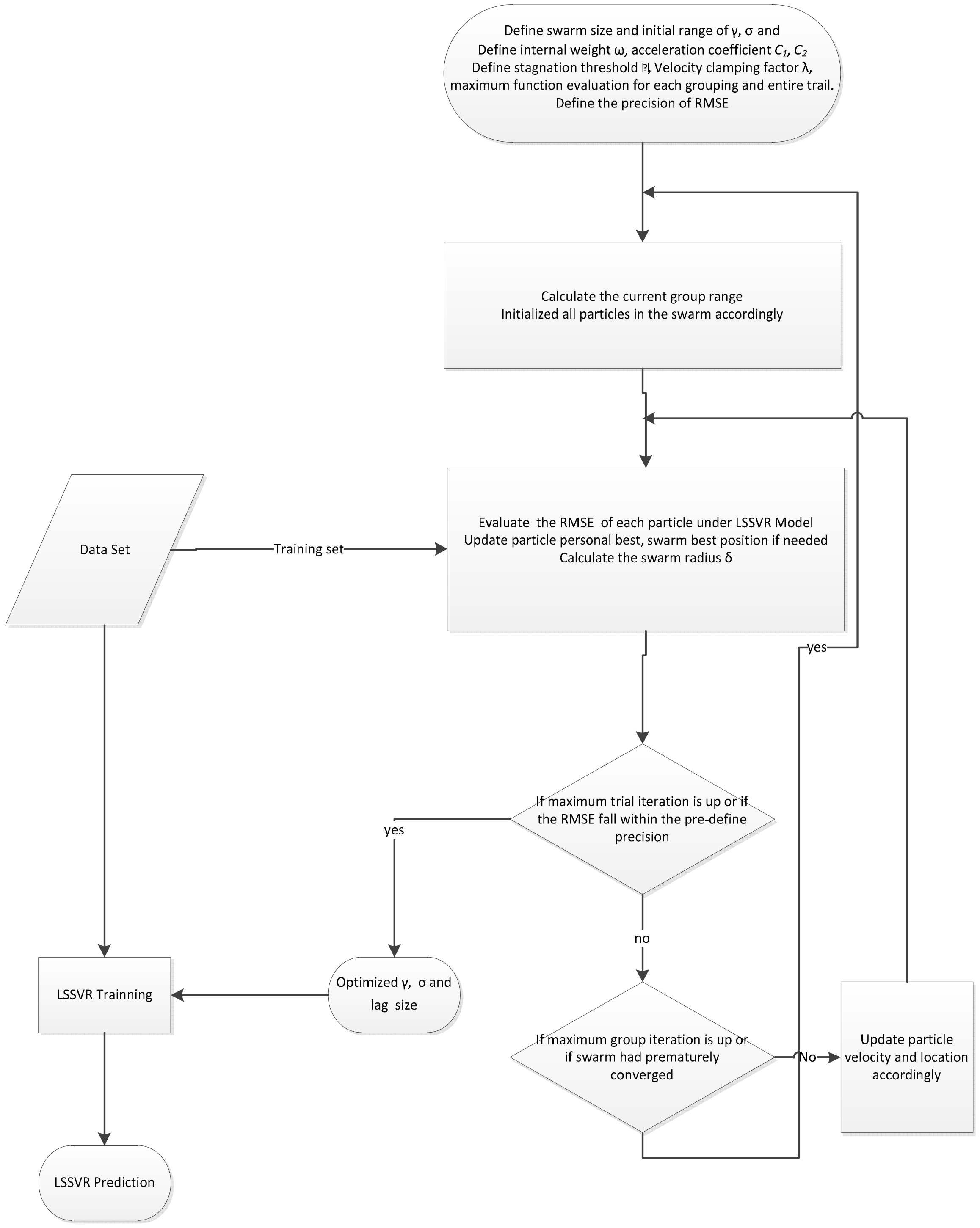
Define swarm size and initial range of γ, σ and
Define internal weight ω, acceleration coefficient C1, C2
Define stagnation threshold , Velocity clamping factor λ,
maximum function evaluation for each grouping and entire trail.
Define the precision of RMSE
Calculate the current group range
Initialized all particles in the swarm accordingly
Data Set
Training set
Evaluate the RMSE of each particle under LSSVR Model
Update particle personal best, swarm best position if needed
Calculate the swarm radius δ
yes
If maximum trial iteration is up or if the RMSE fall within the pre-define precision
yes
no
LSSVR Trainning
Optimized γ, σ and lag size
If maximum group iteration is up or if swarm had prematurely converged
No
Update particle velocity and location accordingly
LSSVR Prediction

Figure 1: RegPSO+LSSVR model

CHAPTER IV

EXPERIMENTAL FINDINGS

In order to evaluate the proposed RegPSO+LSSVR model, two other models are also constructed for comparison purposes. The first model is LSSVR with AMI for lags selection and grid search algorithm for hyper-parameters selection (AMI+GRID+LSSVR). The second model is as follows; LSSVR uses AMI for lags selection and uses standard PSO to find hyper-parameters (AMI+PSO+LSSVR). AMI+GRID+LSSVR is constructed mainly using LSSVMLAB 1.7 [11], AMI+PSO+LSSVR and RegPSO+LSSVR are constructed using the combination of LSSVMLAB 1.7 [11] and G. Evers' MATLAB PSO Research Toolbox [6]. The experiments were run under a PC with AMD Phenom II 2.8 GHZ as processors and 8 GB of RAM. The Operating system is Windows 7, and the development platform is MATLAB 7.11.0. The detail parameters setting and the results of each model are listed in appendix.

Two real world datasets were used in this study. The first dataset was the monthly production of sulfuric acid in Australia from January 1956 to July 1994 [16]; out of the 462 samples, the first 323 were used as training samples, and the testing samples are the remaining 139; their values ranged from 42 to 228 in thousands of tons. The second dataset was the annual sunspot numbers from the Royal Observatory of Belgium from

1700 to 2011 [17]; it contains 311 samples; the first 233 are treated as training samples, and the remaining 78 samples were used for testing purposes; the sample value ranged from 0 to 190.2.

In this paper, time series were pretreated by copying 'lags' number of next data points into a matrix, and the traditional K-fold cross-validation method that randomly partitions the data into K complementary subsets, will cause the some of the validating data being used as part of the training data. In order to segregate the training from validating data sets, an adaptation of Monte Carlo cross-validation method is used in this paper. For example, during one round of a 10% cross-validation, the size of the validation block will be $10\% * size\ of\ training\ set + 2 * lag$; and the validation block will be randomly selected from the training set as a whole.

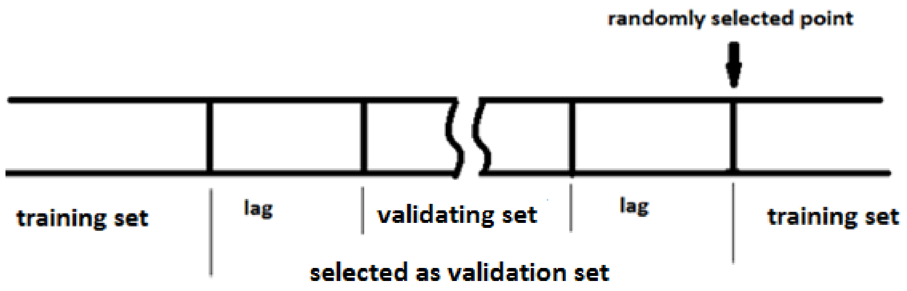

Figure 2: selecting validation set from the left (when the size of validation set is less than the selected index)

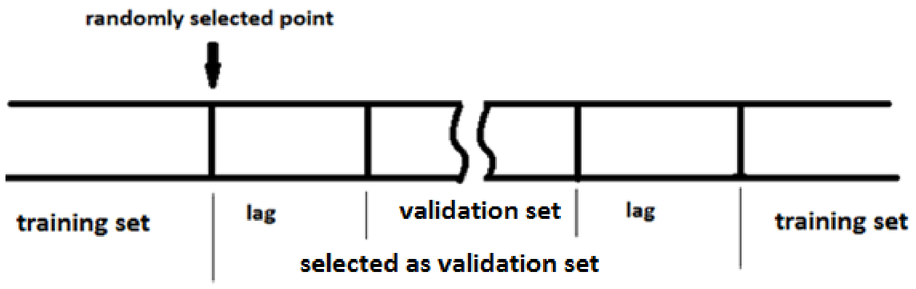

Figure 3: selection of validation set from right (when the size of validation set is greater than the selected index).

As illustrated above, there were 'lag' number of extra points selected before and after the actual validation set. These extra points were excluded during the comparison for test results. In order to measure the errors on an even scale, the entire training set were standardized by zero mean and unit variant before given to LSSVR for training and cross-validation. Three types of errors were measured for each model; namely, mean absolute errors (MAE), Maximum errors (MAX), the root mean squared errors (RMSE). They are defined as follows:

$$MAE = \frac{\sum_{i=1}^{N} |f(x_i) - Y_i|}{N}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (f(x_i) - Y_i)^2}{N}}$$

$$MAX = sup_i |f(x_i) - Y_i|$$

where $f(x_i)$ is the standardized value obtained from the current model, and Y_i is the standardized observed value. The search criteria for all models were based on RMSE obtained under cross-validation of the training sets. Three sets of errors were measured across the models based on one-step look-ahead prediction on training sets; one-step look-ahead prediction for testing sets; and lastly recursive prediction on 1st 12 steps of the testing set after the solutions had been found.

Results of errors on each model

Dataset	Error type	Dataset	RegPSO + LSSVM	AMI + PSO + LSSVM	AMI + Grid Search + LSSVM
Training (one step ahead prediction)	MAX	Sulfuric acid	**1.1745**	1.5131	1.5229
		Sun spots	**1.2916**	1.6298	1.7992
	RMSE	Sulfuric acid	**0.331**	0.4252	0.4296
		Sun spots	**0.3087**	0.3611	0.3885
	MAE	Sulfuric acid	**0.2511**	0.3165	0.3191
		Sun spots	**0.2281**	0.2736	0.2929
Testing (One step ahead prediction)	MAX	Sulfuric acid	**1.372**	1.5545	1.5554
		Sun spots	2.708	**2.188**	2.1998
	RMSE	Sulfuric acid	0.5201	**0.4665**	0.4691
		Sun spots	0.7994	0.7237	**0.6271**
	MAE	Sulfuric acid	0.4232	**0.3638**	0.3665
		Sun spots	0.5926	0.5317	**0.4675**
Testing (recursive prediction 1st 12 steps)	MAX	Sulfuric acid	**0.8084**	1.3628	1.3895
		Sun spots	**0.5854**	1.1973	1.3897
	RMSE	Sulfuric acid	**0.2886**	0.6026	0.6144
		Sun spots	**0.4112**	0.6969	0.707
	MAE	Sulfuric acid	**0.2081**	0.4901	0.499

		Sun spots	**0.3657**	0.6022	0.5702

Table I: errors collected for each model with respect to training and testing sets

Comparison of results

From the above errors table, all models perform reasonably well under 1-step ahead prediction. The proposed model obtained smaller errors than the other two models on training data with one-step ahead prediction; it also obtained better results on recursive short term prediction (first 12 steps) for testing data as well. Figure 2 -9 plots the errors in table I for illustration purposes. The plotted short-term testing results (figure I and figure IV from the appendix) confirmed the views drawn from the training error table.

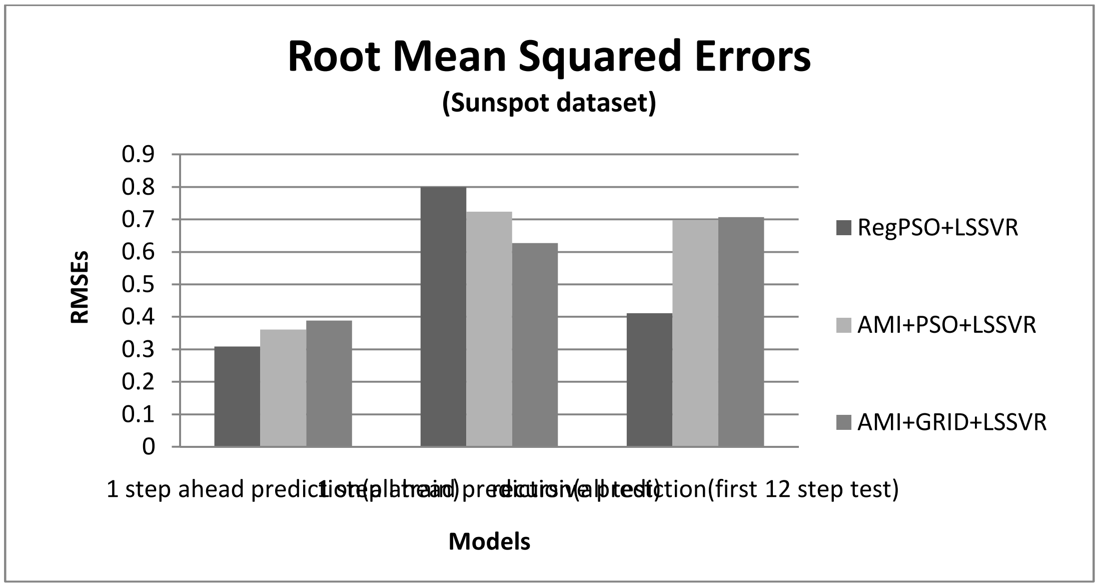

Figure 4: RMSE with respect to each model for sunspot dataset set

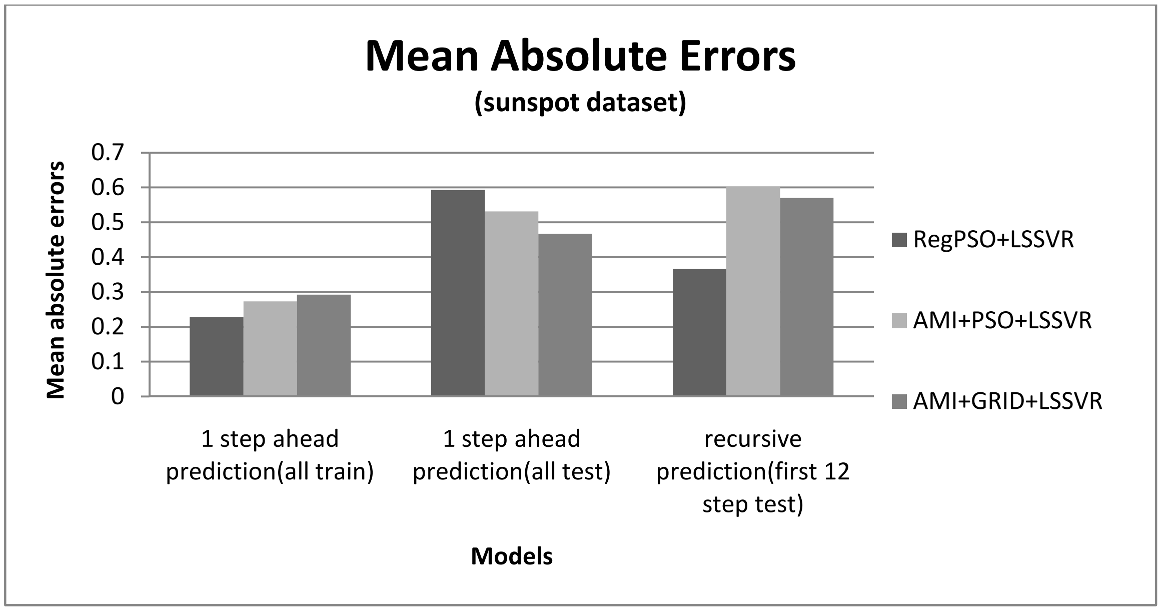

Figure 5: MAE with respect to each model for sunspots dataset

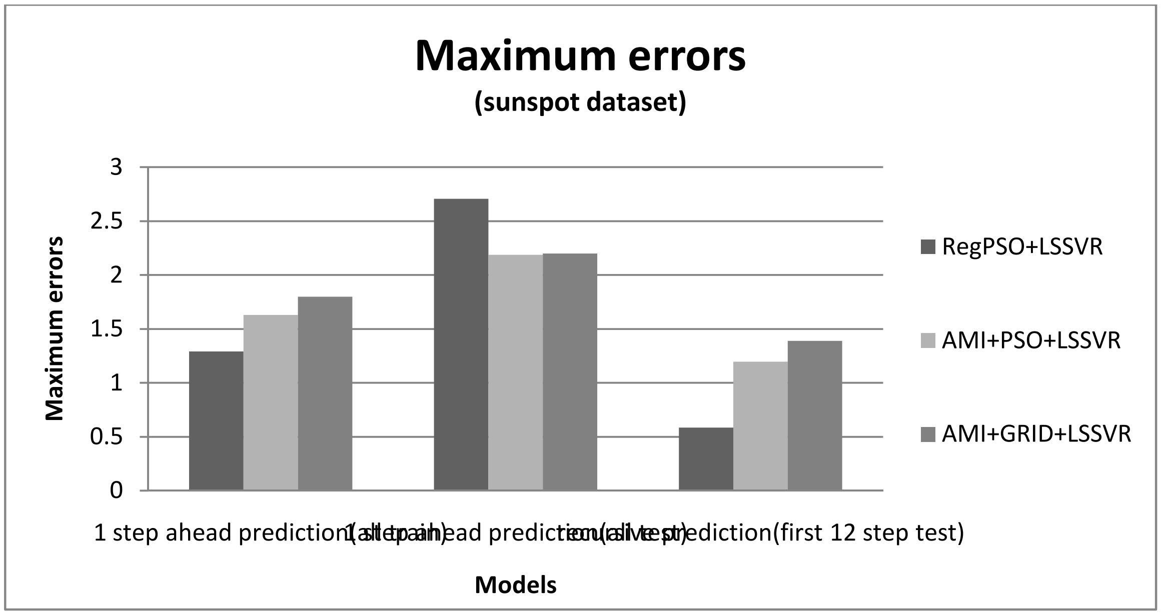

Figure 6: MAX with respect to each model for sunspot dataset

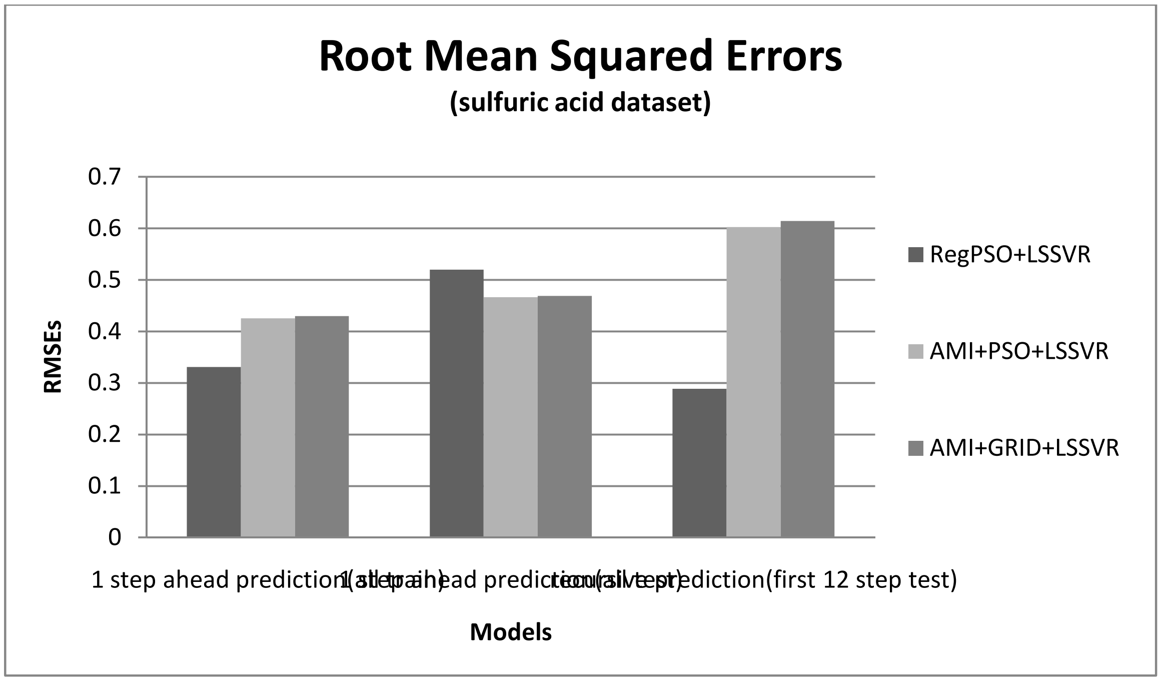

Figure 7: RMSEs with respect to. each model for sulfuric acid dataset

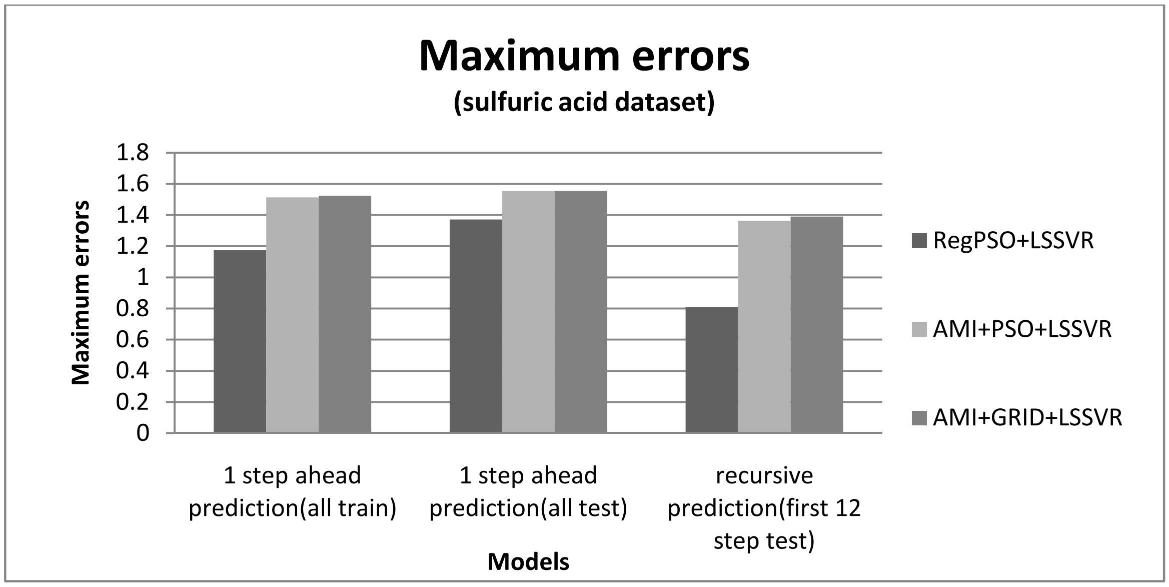

Figure 8: Maximum errors with respect to each model for sulfuric acid dataset

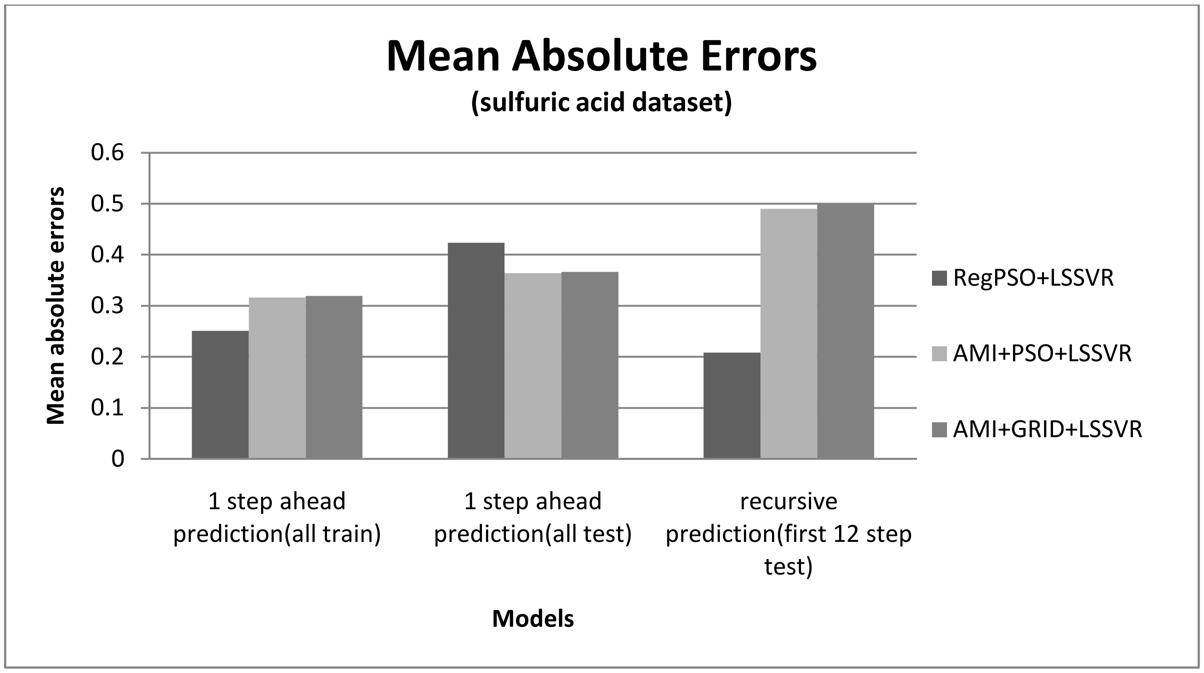

Figure 9: MAE with respect to each model for sulfuric acid data set

CHAPTER V

CONCLUSION

Based on the empirical results, the proposed model consistently performs well across both real world datasets. One can conclude that the proposed RegPSO + LSSVR model indeed can be used as an alternative approach for short term time series forecasting. Since the cost of evaluating the fitness of each particle at any location is the same as constructing and evaluating a LSSVR at that given setting, it is no doubt that a faster SVM approach would greatly speed up this type of parameter optimization approach. It would be interesting to see the effect of extending this approach to algorithms such as the fast sparse approximation for least squares support vector machine (FSALSSVM) [16].

REFERENCES

1. G.E.P. Box, G.M. Jenkins, (1978) *Time series analysis: Forecasting and control*. 3rd Edition, Holden Day, San Francisco, ISBN-10: 0130607746.

2. R. Engle (1982) "Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflations". *Econometrica*, 50:987-1007.

3. H. Drucker, C. J.C. Burges, L. Kaufman, A. Smola and V. Vapnik (1997) "Support vector regression machines". *Advances in neural information processing systems* 9:155-161

4. C.-M. Kuan, T. Liu, (1995) "Forecasting exchange rates using feed forward and recurrent neural networks". *Journal of applied econometrics*, 10:347–364.

5. D. Li, W. Xu, H. Zhao and R. Chen, (2009) "A SVR based forecasting approach for real estate price prediction", *International conference on machine learning and cybernetics 2009,* 9:970-974

6. G.I. Evers, B.M. Ghalia, (2009) "Regrouping particle swarm optimization: A new global optimization algorithm with improved performance consistency across benchmarks." *IEEE International conference on systems, man and cybernetics,* 2009. SMC 2009. 3901 – 3908

7. E. Huerta, B. Duval and J. Hao, (2006) "A hybrid GA/SVM approach for gene selection and classification for microarray data" *EvoWorkshops*, LNCS 3907, 34–44

8. M. Pan, D. Zeng and G. Xu, (2010) “Temperature prediction of hydrogen producing reactor using SVM regression with PSO” *Journal of computers*, 5:388-393

9. T.M. Cover and J.A. Thomas, (1991) *Elements of information theory.* Wiley-Interscience, New York, ISBN: 9780471062592

10. Y. Ren, (2010) “Determination of optimal SVM parameters by using GA/PSO” *Journal of computers* 2010, 5:1160-1168

11. J.A. K. Suykens, T.V. Gestel, J. D. Brabanter, (2002) *Least squares support vector machines* World scientific, ISBN: 978-981-238-151-4

12. J. Kennedy, R. C. Eberhart, (1995) "Particle swarm optimization". *Proceedings of IEEE international conference on neural networks*. 4:1942–1948.

13. Y, Shi, R.C. Eberhart, (1998) "A modified particle swarm optimizer". *Proceedings of IEEE international conference on evolutionary computation*. 69–73.

14. F. Van den Bergh and A. P. Engelbrecht, (2002) "A new locally convergent particle swarm optimiser," *Proceedings of the IEEE conference on systems, man and cybernetics*, Hammamet, Tunisia, 96-101.

15. R. Samsudin, A. Shabri and P. Saad, (2010) “A comparison of time series forecasting using support vector machine and artificial neural network model” *Journal of applied science*, 10:950-958.

16. Monthly production of sulphuric acid in Australia: in thousand tons, Jan 1956 – Jul 1994. Source: Australian Bureau of Statistics. (http://robjhyndman.com/tsdldata/data/sulphur.dat).

17. SIDC, RWC Belgium, World Data Center for the Sunspot Index, Royal Observatory of Belgium, `311 years-of-data' (http://sidc.oma.be/DATA/yearssn.dat).

18. L. Jiao, L. Bo and L. Wang, (2007) "Fast sparse approximation for least squares support vector machine". *IEEE transactions on neural networks*, 18:685-697

APPPENDICES

Table I

Model parameter settings				
	Datasets	Standard PSO	Regrouping PSO	Grid search
Maximum number of function evaluations (total)	Sulfuric acid	4000	4000	4000
	Sun spots	4000	4000	4000
Maximum function evaluations per grouping	Sulfuric acid	N/A	400	N/A
	Sun spots	N/A	400	N/A
Population size for PSO / step division for grids	Sulfuric acid	20	20	25/25
	Sun spots	20	20	25/25
The minimum inertia weight	Sulfuric acid	0.4	0.4	N/A
	Sun spots	0.4	0.4	N/A
The maximum inertia weight	Sulfuric acid	0.9	0.9	N/A
	Sun spots	0.9	0.9	N/A
Gamma search range	Sulfuric acid	0-5000	0-5000	0-5000
	Sun spots	0-5000	0-5000	0-5000
Sig2 search range	Sulfuric acid	0-5000	0-5000	0-5000
	Sun spots	0-5000	0-5000	0-5000
Lag search range	Sulfuric acid	N/A	0-30	N/A
	Sun spots	N/A	0-30	N/A
Stagnation thresholds	Sulfuric acid	N/A	0.00011	N/A
	Sun spots	N/A	0.00011	N/A

Table II

Results obtained for each model				
Model	Dataset	RegPSO + LSSVM	AMI + PSO + LSSVM	AMI + Grid Search + LSSVM
lags	Sulfuric acid	22	6	6
	Sun spots	7	4	4
gamma	Sulfuric acid	3233.2	2133.4	4194
	Sun spots	953.6	3230.3	3028.1
Sig2	Sulfuric acid	1596.6	560.98	1101.4
	Sun spots	1313.1	144.12	953.6

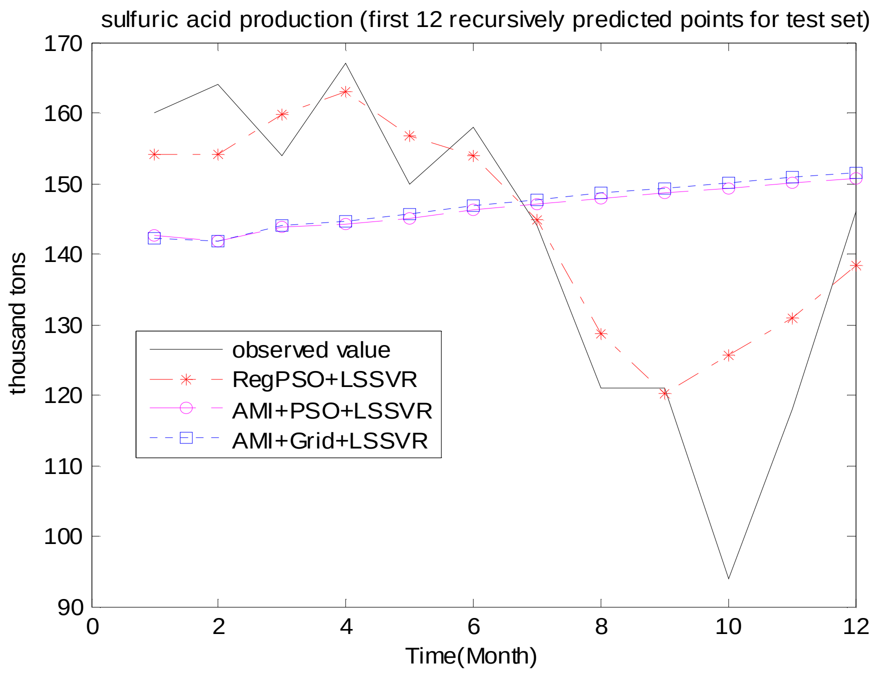

Figure I: next 12 monthly sulfuric acid production forecasting on testing dataset

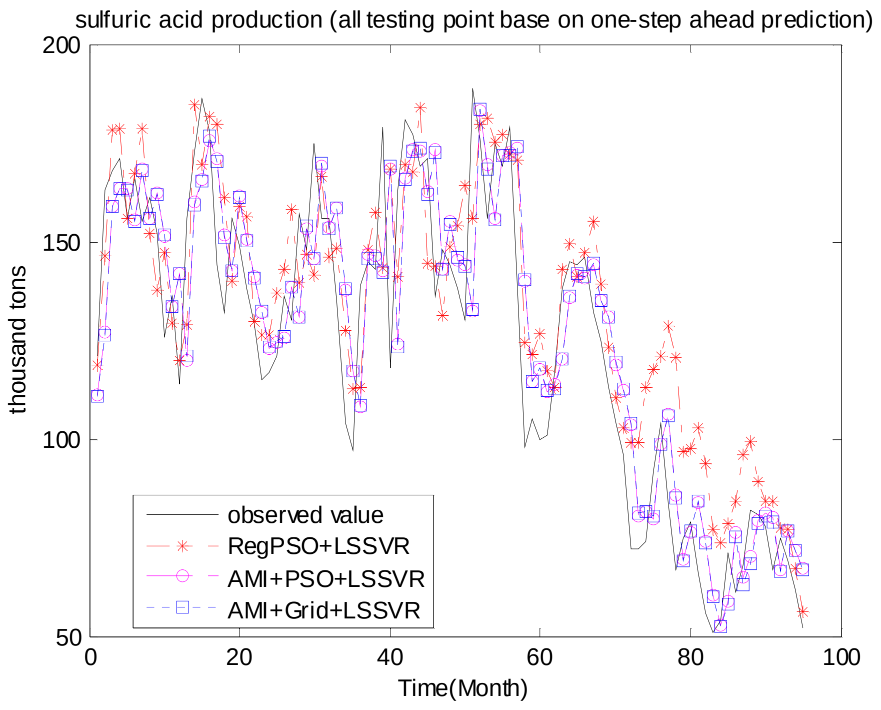

Figure II: plots of all testing points for sulfuric acid dataset

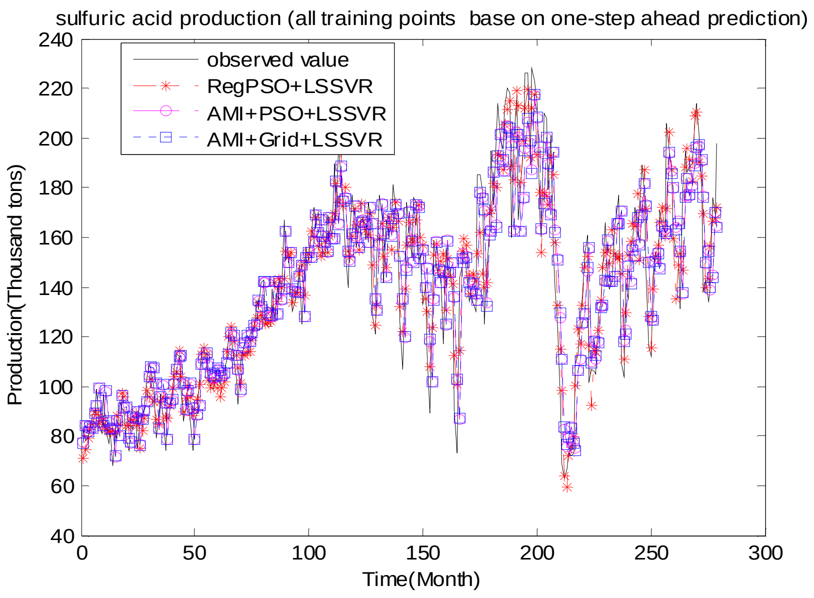

Figure III: plot of all training data for sulfuric acid production dataset

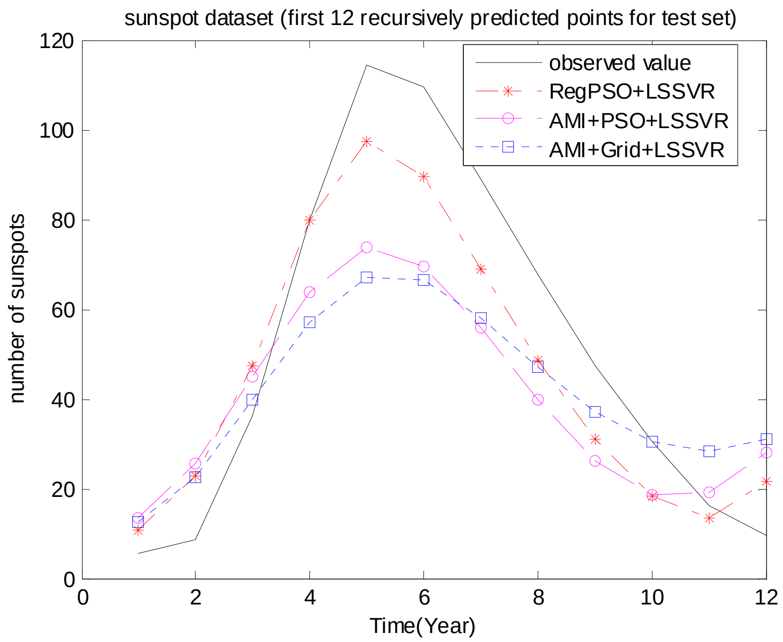

Figure IV: next 12 years of sun spots number forecasting on testing dataset.

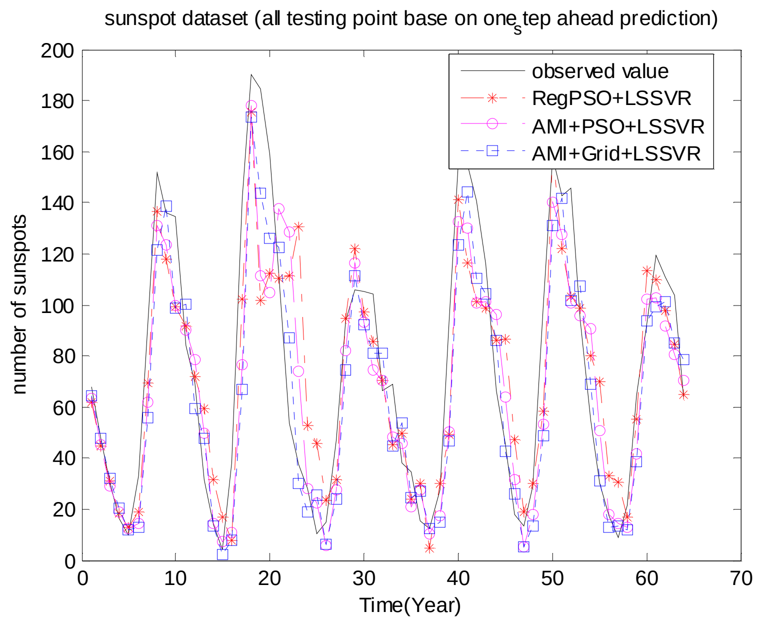

Figure V: plot of all testing points for sunspots dataset

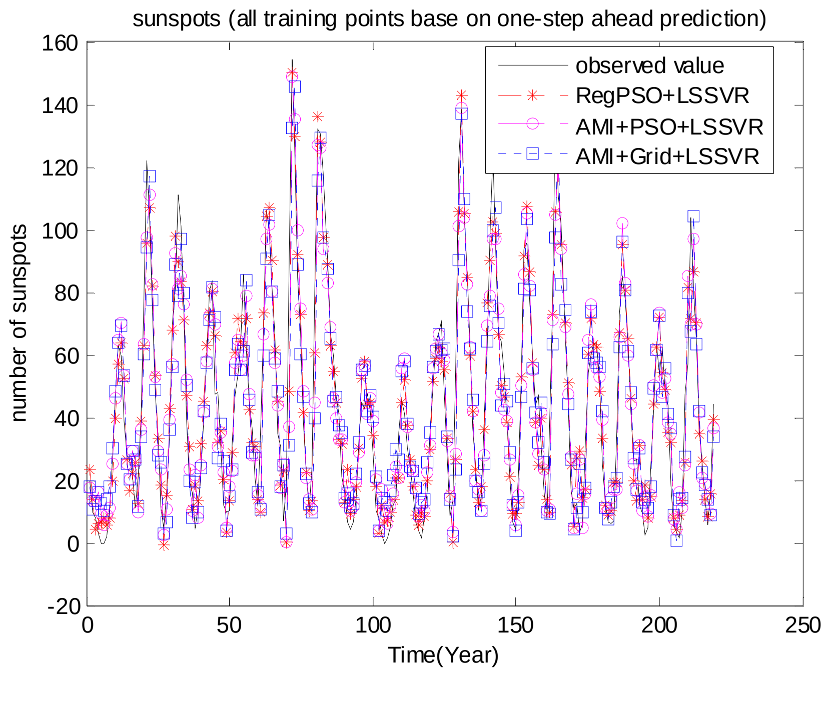

Figure VI: plot of all training points for sunspots dataset

VITA

Chaohui Sun

Candidate for the Degree of Computer Science

Master of Science

Thesis: AN APPLICATION OF LEAST SQUARES SUPPORT VECTOR

REGRESSION WITH REGROUPING PARTICLE SWARM OPTIMZATION

Major Field: Computer Science

Biographical:

Education:

Completed the requirements for the Master of Science in Computer Science at Oklahoma State University, Stillwater, Oklahoma in July, 2011.

Completed the requirements for the Bachelor of Science in Computer Science at University of Tulsa, Tulsa, Oklahoma in 1996.

Experience:

Professional Memberships:

Name: Chaohui Sun Date of Degree: July, 2011

Institution: Oklahoma State University Location: Stillwater, Oklahoma

Title of Study: AN APPLICATION OF LEAST SQUARES SUPPORT VECTOR REGRESSION WITH REGROUPING PARTICLE SWARM OPTIMZATION

Pages in Study: 28 Candidate for the Degree of Master of Science

Major Field: Computer science

Scope and Method of Study:

applying LSSVR and REGPSO in short term time series forecasting.

Findings and Conclusions:

Least Squares Support Vector Regression (LSSVR) is a powerful machine learning tool. The performance of LSSVR is not only directly linked to the proper selection of its hyper-parameters, but also to the proper feature selection of the targeted dataset. In time series forecasting, features selection can be viewed as selecting the numbers of past data points. It became important for selecting a good combination of both these parameters and features, if we want to do any meaningful short-term forecasting for time series data. The existing parameter selection methods employ many optimizing techniques that range from grid search to neural networks and particle swarm optimization, but they all left the feature selection of the series to users. A novel method is proposed here to select both LSSVR parameters and the features of the time series at the same time. The real world data used in this study demonstrate the proposed method achieves better performance in terms of recursive short-term forecasting, when compared to existing standard PSO and grid search methods that focus on hyper-parameters selection and leaves the feature selection to Average Mutual Information (AMI).

ADVISER'S APPROVAL: Dr. Douglas R. Heisterkamp