Sequential Relevance Vector Machine Learning from Time Series

Nikolay Nikolaev
Department of Computing, Goldsmiths College, University of London
London SE14 6NW, United Kingdom
E-mail: n.nikolaev@gold.ac.uk
Peter Tino
School of Computer Science, The University of Birmingham
Birmingham B15 2TT, United Kingdom
E-mail: P.Tino@cs.bham.ac.uk

Abstract— **This paper presents an approach to sequential training of the relevance vector machine suitable for Bayesian learning from time series. The key idea is to perform simultaneous incremental optimization of both the weight parameters and their prior hyperparameters using data arriving one at a time. Algorithms for efficient sequential regularized dynamic learning rate training of the weights and gradient-descent training of their priors are derived. It is shown that this fast sequential RVM can outperform similar Bayesian kernel methods, like: batch RVM, fast RVM, variational RVM, and Gaussian Processes on multistep ahead forecasting of time series.**

I. Introduction

Recent research indicates that kernel-based methods are successful in time series regression [1], [5], [6], [7], [9], [13], [15]. Such an approach that yields well generalizing models is the Relevance Vector Machine (RVM) [13], [14]. The RVM offers three essential advantages [13]: 1) it allows the liberal use of arbitrary kernels; 2) it performs reliable inference with training formula obtained using proper probabilistic treatment of the inductive process, including the data and the noise; and 3) it does not require to determine the error/margin parameter in advance. Most of the work on the RVM for regression and time series forecasting however has been conducted in offline setting [1], [2], [4], [13], [14], [16]. The design of good incremental algorithms for the RVM is still a challenge which can make it useful for addressing real-world applications.

An attractive characteristic of the RVM is that it produces parsimonious probabilistic models. It is a Bayesian framework that estimates the weight posterior distribution through maximization of the marginal likelihood of their hyperparameters. The marginalization leads to automatic identification of the relevant kernels, and thus sparsifies the model. The achieved property sparseness implies capacity to realize accurate predictions. A serious drawback of the original RVM [13] is that it operates in offline mode and thus it is limited to applications where all data are available and are processed as a batch.

The RVM is problematic to use for real time series modeling where the data arrive one at a time because of the following three reasons. First, it is computationally demanding as it involves inversion of a large square matrix of size equal to the number of the data. Second, the analytical batch training formulae sometimes can not be evaluated safely due to instabilities in their numerical computation. Third, the batch strategy is intrinsically incapable of capturing the dynamics of time series in the sense of dependency on the past history. Moreover in very noisy, drifting and non-stationary environments the offline algorithm fails to generate satisfactory results. These problems may be alleviated using incremental strategeies for computing the posterior distributions using one data at a time, and update the model instantaneously.

There have been suggested already several incremental approaches to relevance vector regression. The fast algorithm of Tipping and Faul [14] is an improvement over the RVM in that it stabilizes the inversion of covariance matrix, and also accelerates considerably its speed. The fast RVM [14] it carries out a greedy search by learning and unlearning of kernels till the marginal likelihood is maximized, which however can stuck at suboptimal solutions. The re-estimation formulae for the hyperparameters sometimes produce unuseful negative values which prevent from careful tuning the model. A different incremental procedure for pruning redundant kernels commencing from the complete model is offered by the Backfitting RVM [4]. It adjusts the kernels by passing continuously through the data until the desired accuracy is reached. Although the Backfitting RVM is also a probabilistic method performing expectation maximization, it uses too complex formulae with many meta-parameters that are extremely difficult to tune and synchronize as they are in interplay.

This paper develops a sequential approach to relevance vector regression suitable for Bayesian learning from time series. The key idea is to organize adaptive model selection through simultaneous incremental optimization of both the weight parameters and their prior hyperparameters. We derive and integrate two algorithms for efficient gradient descent training of the RVM: 1) a regularized dynamic learning rate (DLR) training algorithm resembling an approximated Bayesian Kalman filter, and 2) a least mean squares (LMS) training algorithm for hyperparameter adaptation. The formulae are especially derived so as to preserve the spirit of sparse Bayesian learning.

Empirical studies on time series regression demonstrate that the proposed Sequential Relevance Vector Machine (SRVM) can infer models with excellent generalization. Two time series are considered: one artificially generated series with a varying mean, and a large hard series with Electricity load measurements. We report that the SRVM outperforms the batch RVM [13], the fast RVM [14], the variational RVM [2], and a Gaussian Process model [15] on multistep ahead forecasting of the selected series.

This paper outlines the RVM in section two. Section three offers the mechanisms for sequential Bayesian learning, including the algorithms for weight and hyperparameter optimization. Section three gives experimental results on time series modelling. Finally, a discussion is given and a conclusion is derived.

II. Relevance Vector Learning

The inductive task is to find a mapping that accurately describes the relationship between a series of observables sampled at discrete time intervals. The series are viewed as a data set $D = \{(\mathbf{x}_n, y_n)\}_{n=1}^{N}$ of size N where $\mathbf{x}_n = (x_{n-(d-1)\tau}, x_{n-(d-2)\tau}, ..., x_n) \in \mathcal{R}^d$ (d is the embedding dimension and τ is the delay) and $y_n \in \mathcal{R}$. Here we use mappings that are linear superpositions of kernels [13]:

$$f(\mathbf{x}) = \sum_{n=1}^{M} w_n K(\mathbf{x}, \mathbf{x}_n) + \varepsilon = \mathbf{w}^T \mathbf{k}(\mathbf{x}) + \varepsilon \tag{1}$$

where w_n are the weight parameters forming the vector $\mathbf{w} = [w_1, w_2, ..., w_M]^T$, $K(\mathbf{x}, \mathbf{x}_n)$ are the basis functions or kernels, $\mathbf{k}(\mathbf{x})$ are the kernel vectors $\mathbf{k}(\mathbf{x}) = [K(\mathbf{x}, \mathbf{x}_1), K(\mathbf{x}, \mathbf{x}_2), ..., K(\mathbf{x}, \mathbf{x}_M)]^T$, and and ε is independent zero-mean Gaussian noise with unknown variance σ^2. Usually the Gaussian kernel is preferred $K(\mathbf{x}, \mathbf{x}_n) = exp[-(\mathbf{x} - \mathbf{x}_n)^T(\mathbf{x} - \mathbf{x}_n)/(2s^2)]$, where s^2 is the width.

The RVM of Tipping [13] provides a procedure for Bayesian learning of sparse kernel models. It estimates the weight posterior according to the Bayes' rule using the data likelihood $p(\mathbf{y}|\mathbf{x}, \mathbf{w}(\boldsymbol{\alpha}), \beta^{-1})$, the weight prior $p(\mathbf{w}|\boldsymbol{\alpha})$, and the normalising term $p(\mathbf{y}|\boldsymbol{\alpha}, \beta^{-1})$:

$$p(\mathbf{w}|\mathbf{y}, \boldsymbol{\alpha}, \beta^{-1}) = \frac{p(\mathbf{y}|\mathbf{x}, \mathbf{w}(\boldsymbol{\alpha}), \beta^{-1}) p(\mathbf{w}|\boldsymbol{\alpha})}{p(\mathbf{y}|\boldsymbol{\alpha}, \beta^{-1})} \tag{2}$$

where $\boldsymbol{\alpha} = (\alpha_1, ..., \alpha_M)$ and β are hyperparameters.

This Bayesian inference implies that the search for optimal weights is governed by their priors. The hyperparameters $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_M)$ quantify "the prior beliefs in the weights" (see below), and the hyperparameter β quantifies the output noise $\beta = \sigma^{-2}$. RVM optimizes iteratively the marginal likelihood $p(\mathbf{y}|\mathbf{x}, \boldsymbol{\alpha}, \beta^{-1})$ by re-estimation of the hyperparameters. They are computed with formulae obtained according to the evidence procedure [8] by maximization of the log likelihood $\mathcal{L}(\boldsymbol{\alpha}) = \log p(\mathbf{y}|\mathbf{x}, \boldsymbol{\alpha}, \beta^{-1})$:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\alpha}) &= \frac{1}{2}\left(\log|\boldsymbol{\Sigma}| + N \log \beta + \log |\mathbf{A}|\right) \\ &\quad -\frac{1}{2}\left(\beta||y - \mathbf{K}\mathbf{w}||^2 + \mathbf{w}^T \mathbf{A} \mathbf{w}\right)\end{aligned} \tag{3}$$

where $\mathbf{A}$ is the diagonal matrix $\mathbf{A} = diag(\alpha_1, \alpha_2, ..., \alpha_M)$, $\mathbf{K}$ is the design matrix $\mathbf{K} = [\mathbf{k}(\mathbf{x}_1), ..., \mathbf{k}(\mathbf{x}_M)]^T$, and $\boldsymbol{\Sigma}$ is the inverse matrix $\boldsymbol{\Sigma} = (\beta \mathbf{K}^T \mathbf{K} + \mathbf{A})^{-1}$.

The relevance vector learning apparatus uses least squares fitting to calculate the maximum aposteriori weights. It adopts a quadratic penalty function for the prior over the weights, also known as weight decay regularization. This kind of regularization shrinks the weights of irrelevant kernels leading to model sparsification. The prior distribution is a zero-mean factorized Gaussian:

$$p(\mathbf{w}|\boldsymbol{\alpha}) = \prod_{n=1}^{M} \sqrt{\frac{\alpha_n}{2\pi}} \exp\left(-\frac{1}{2}\alpha_n w_n^2\right) \tag{4}$$

assuming that the hyperparameters are independent.

The RVM begins with the full model, having basis functions centered on all given data, and adapts the weights and their hyperparameters. During the learning process some hyperparameters grow thus causing their weights to decrease toward zero. This effects essentially in pruning kernels from the model, so the approach performs automatically model selection of the relevant basis functions, called relevance vectors [13].

The RVM allows us to make probabilistic predictions, as it suggests to compute the mean y_* and variance β_*^{-1} of the predictive distribution $p(y*|\mathbf{y}, \mathbf{x}, \boldsymbol{\alpha}, \beta^{-1})$:

$$y_* = \mathbf{w}^T \mathbf{k}(\mathbf{x}_*) \tag{5}$$

$$\beta_*^{-1} = \beta^{-1} + \mathbf{k}(\mathbf{x}_*)^T \boldsymbol{\Sigma} \mathbf{k}(\mathbf{x}_*) \tag{6}$$

where $\mathbf{x}_*$ is the future input vector.

III. Sequential RVM Learning

The fast RVM method of Tipping and Faul [14] performs model selection by adding to the model and removing from it kernels until the marginal likelihood of the hyperparameters is maximised. Despite using a fast recurrent least squares fitting algorithm to estimate the weights, it has two shortcomings: 1) it carries out greedy search for kernels which is inefficient for sequential processing because when more data arrive it is difficult to balance between fitting and generalization as the kernel model naturally tends to increase in size; and 2) it uses analytical formulae for reestimation of the hyperparameters which are very sensitive to fluctuations and noise in the data. These problems are aleviated in the Backfitting RVM [4] which starts with the complete model and passes recursively through the data to determine the most relevant kernels using probabilistic expectation maximization.

The SRVM learns also by sweeping sequentially through the data. During one sweep it consideres the newly arrived data point and optimizes simultaneously both the weight parameter and the hyperparameter of the corresponding kernel. While cycling repeatedly over the data it tunes kernel by kernel, and so evolves gradually the model. Starting from the full model having all basis functions, adaptive selection results as effect from the incremental improvement of the weights and their priors.

A. Regularized Weight Training

We propose a regularized DLR algorithm for Bayesian RVM learning with proper noise treatment. It performs recursive least squares weight estimation in pursuit of reaching the minimum of the mean squared error regularized with weight decay. The algorithm as originally suggsted implements an approximated Kalman filter [12]. Instead of using the full aposteriori weight covariance matrix it uses its diagonal approximation. The difference from the previous K1 [12] is that our algorithm operates on one kernel at a time corresponding to the current training point only. More precisely we use only one entry from the diagonal corresponding to the newly arrived data point.

Each covariance entry is actually learned in online manner from the data. This is done with a gradient-descent training rule which operates on individual, local learning meta-parameters. Working in a Bayesian setting this requires to take the derivative of the regularized log-likelihood taken at the posterior mean weights $E(t) = 0.5\left(\beta||y - \mathbf{K}\mathbf{w}||^2 + \mathbf{w}^T\mathbf{A}\mathbf{w}\right)$ with respect to the meta-parameters $\partial E/\partial p_i$. Thus we obtain the following gradient-descent rule:

$$p_i(t+1) = p_i(t) + \nu\left[\beta e(t)k_i(t) - \alpha_i(t)w_i(t)\right] \qquad (7)$$

where ν is a positive rate change constant, $e(t)$ is the output error $e(t) = y(t) - f(\mathbf{x}(t))$, and $k_i(t) = K(\mathbf{x}_i, \mathbf{x}_i)$. This notation uses indices i runing sequentially over the time series points, while the particular steps of the incremental learning algorithm are indexed by t.

There are two distinctive features of this meta-parameters training rule (7): 1) it is a Bayesian rule as it depends on the output noise β and the weight prior hyperparameters α_i; and 2) it depends only on the instantaneous error gradient and not on long term effects from the past as originally conceived.

Since the covariance should be positive its entries $s_{ii}(t)$ are defined with the transformation:

$$s_{ii}(t) = c\exp(p_i(t+1)) \qquad (8)$$

where c is a constant that stabilizes the convergence.

The gradual adaptation of the covariance elements is advantageous because it enables to tune the weights by varying step sizes in proportion to their impact on the current error. The different weights undergo different updates at a certain incremental step, and this changes from iteration to iteration. Such effects can be achieved with the following regularized DLR version of the approximated Kalman algorithm for sequential training of the RVM:

$$\begin{aligned} G_i(t) &= \frac{s_{ii}(t)k_i(t)}{\beta^{-1} + k_i^2(t)s_{ii}(t)} && (9) \\ w_i(t+1) &= [1 - s_{ii}(t)\alpha_i]w_i(t) + G_i(t)e(t) && (10) \end{aligned}$$

where $G_i(t)$ is known as Kalman gain. Here the factor $s_{ii}(t)\alpha_i$ in the brackets in equation (10) implements the weight decay regularization.

The regularization hyperparameters α_i and the noise hyperparameter β in the above weight update rule (10) are determined within an hierarchical Bayesian setting. The recursive formula (9) and (10) differ in two ways from the similar previous approach to hierarchical Bayesian-Kalman filtering [3]: 1) here the regularization hyperparameters α_i are tuned using a gradient-descent technique (11), while the output noise β is kept fixed so as to avoid search difficulties [3]; and 2) here the covariance diagonal is approximated by a learning rate meta-parameter (8) which is adapted by gradient-descent (7) and it is also regularized so as to depend on the weight prior hyperparameters.

The overall effect from the two integrated formula for the covariance entries (9) and for the weights (10) allows us to achieve better convergence properties, in the sense that when the weight changes continue in the same direction the convergence is accelerated. Being a sequential algorithm the DLR is however sensitive to the initial values of the weights. The initial weights may be computed as follows [14]: $w_i(0) = \mathbf{k}(\mathbf{x}(i))^T\mathbf{y}/\mathbf{k}(\mathbf{x}(i))^T\mathbf{k}(\mathbf{x}(i))$, which is the normalized projection of the i-th kernel vector $\mathbf{k}(\mathbf{x}(i))$ on the given output vector $\mathbf{y}$.

B. Hyperparameter Training

The simultaneous training of the parameters and hyperparameters is convenient to implement with different techniques in order to avoid eventual degradation of any of the two search processes. Even if one of the two learning processes stagnates when the other is technically different, one can expect from it to push the first one on its search landscape and to improve it. This is important for the RVM as its two training processes are mutually dependant. In order to achieve stable convergence the noise hyperparameter β is kept fixed [3].

The sequential RVM training aims at marginal likelihood maximization which requires to take the derivatives of the log likelihood $\mathcal{L}(\boldsymbol{\alpha})$ (3) with respect to the hyperparameters $\partial\mathcal{L}(\boldsymbol{\alpha})/\partial\alpha_i$ [13]. Using it we develop the following gradient-descent update rule:

$$\alpha_i(t+1) = \alpha_i(t) + \upsilon\left[\alpha_i^{-1}(t) - [\mathbf{\Sigma}]_{ii} - w_i^2(t)\right] \qquad (11)$$

where υ is a positive constant, and $[\mathbf{\Sigma}]_{ii}$ denotes the i-th diagonal entry of the matrix $\mathbf{\Sigma}$. The learning rate constant υ is common for all priors.

The adaptation of the hyperparameters α_i (11) is performed after each next training example following the adjustment of the weights (10). During this sequential relevance vector learning process some hyperparameters increase while other decrease in magnitude. Thus, it enables adaptive model selection in the sense that the examination of the hyperparameters after each next example shows which kernels to eliminate from the model.

IV. Empirical Investigations

The goal of the presented research work is to develop a good tool for time series regression. There are many computational tools for time series regression which can predict well regular series, but the opened problem is how to design tools that have ability to model well and fast also drifting and non-stationary data.

Following this research goal we implemented several algorithms for Bayesian kernel regression: the SRVM, the closely related batch RVM [13], the fast FRVM [14], the variational VRVM [2], and a Gaussian Process model [15]. The presented empirical investigations are conducted using two time series: a data set with a varying mean produced with an artificially devised formula, and a hard benchmark data set with Electricity load data.

The settings in the reported experiments were as follows. The width of the Gaussian kernels in all algorithms was set to $s^2 = 1.0$ [10]. All algorithms were initialized with noise $\beta = 1.0/(N{*}stdy)$ [14], where $stdy$ is the standard dviation of the data in the given series. The pruning threshold in all algorithms was the same $\alpha > \alpha_{MAX} = 1.0$. The SRVM was made to iterate 50 times over the data. The prior hyperparameters were initialized with $\alpha_i(0) = 1.0e^{-5}/w_i^2(0)$, and their rates were $\upsilon = 1.0e^{-5}$. The initial value of the learning rate meta-parameters was $p_i(0) = -1.0$, their rate constant was set to $\nu = 0.25$, and $c = 0.025$. The RVM and VRVM were both reiterated 5 times over the data, starting with $\alpha(0) = 1.0e^{-5}$.

The FRVM was made with the same starting and pruning values but it did only two sequential passes over the data. It is important to point out that we implemented the FRVM with the same recursive least squares fitting algorithm as Tipping and Faul [14] to find the new weight, after adding kernels to the model in greedy manner, and to re-evaluate all previous weights in the current model.

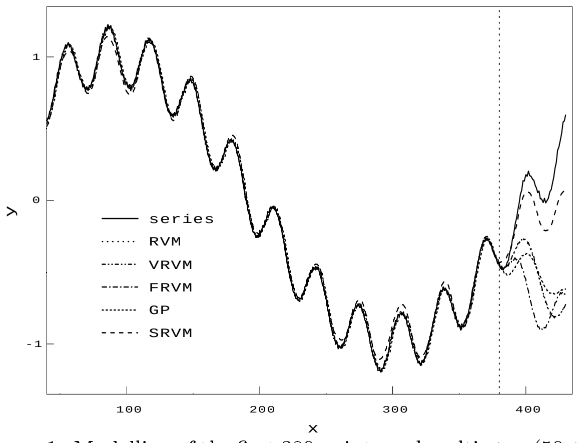

Figure 1. Modelling of the first 380 points and multi-step (50steps) ahead forecasting of the artificially generated time series with varying mean by the five Bayesian kernel algorithms.

Error	training	forecasting	RVs
	NMSE	NMSE	number
GP	0.00014	0.06155	
RVM	0.00021	0.09863	162
VRVM	0.00022	0.09719	184
FRVM	0.00039	0.11378	158
SRVM	0.00382	0.00446	292

Table 1. Training and testing errors committed by the Bayesian kernel algorithms on the time series with varying mean.

However we did not use their hyperparameter estimation formulae which often yield unuseful negative values, rather we used the EM formulae [13]. The GP model used Gaussian covariance functions, and conjugate gradients training iterated 10 times using initial regularization $\lambda = 0.001$ for evaluating the covariance parameters.

Artificial series. A time series of 445 points ($1 \leq t \leq 445$) was generated using the equation:

$$y(t) = \sin(0.0125t) + 0.2\sin(0.2t) \qquad (12)$$

which is a sinusoidal wave with added sinusoidal noise. In addition the series was contaminated with random noise having variance 0.01. The first 380 points were used for training (lag $d = 15$) while the remaining 50 were used only for testing. The results were measured using the normalized mean squared error (NMSE), which is the MSE divided by the series variance (Table 1).

Figure 1 shows that the SRVM outperforms on multi-step prediction the remaining studied Bayesian approaches. It seems that it learns models with really good generalization. One of the reasons for this seems to be the sequential nature of the synchronous optimization of the weights and their priors. The incremental learning offers potential to capture not only the data characteristics but also implicitly the time of their arrival.

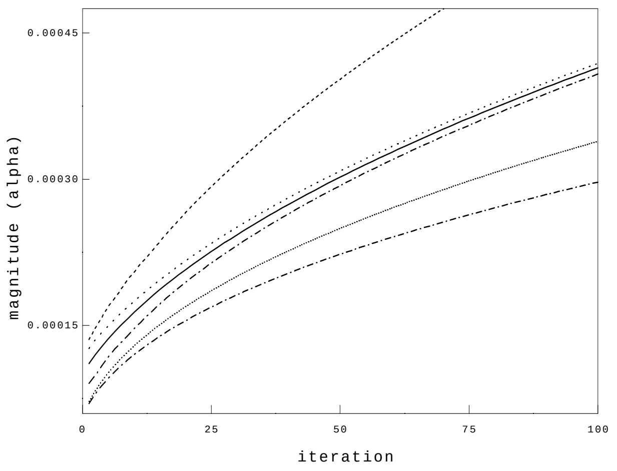

Figure 2. Evolution of six randomly selected regularization parameters during a single run of IRVM using the artificially generated series, each starting from $\alpha_i(0) = 1.0e^{-5}/w_i^2(0)$.

It is interesting to observe however that the SRVM does not learn very sparse models, it seems that it only slightly prunes the complete model. The SRVM model is not very sparse but this does not affect in a negative way the generalization performance. One can see in Table 1 that the batch RVM, the variational VRVM and even the fast FRVM algorithms produce much less complex kernel models. The further experiments with other data indicated that on average the complexity of the discovered models is close to this attained by the FRVM (Table 2).

Next we inquire how the hyperparameter gradient-descent optimimization algorithm modifies the regularizers α_i, in orther to find out whether it can prune weights from the model. When there are regularization hyperparameters that grow indefinitely this is an indication that the corresponding weight will be removed. If there are regularization hyperparameters whose magnitude saturates this means that they will remain in the model. Figure 2 depicts the evolution curves of randomly selected hyperparameters recorded during a single run carried out using the same generated series. This figure was made after picking some informative curves among the available more than two hundred curves of the remaining alphas.

It can be observed in Figure 2 that really the magnitudes of some hyperparameters increase rapidly because the top leftmost curve goes fast toward the threshold. This curve was taken by identifying one particular kernel that was removed from the model. The remaining curves in the middle show typical changes of the regularizers of kernels that were kept in the model. A comment can be made here that these curves seem smooth but this can be attributed to the fact that the generated time series was not extremely difficult to learn for the incremental algorithm.

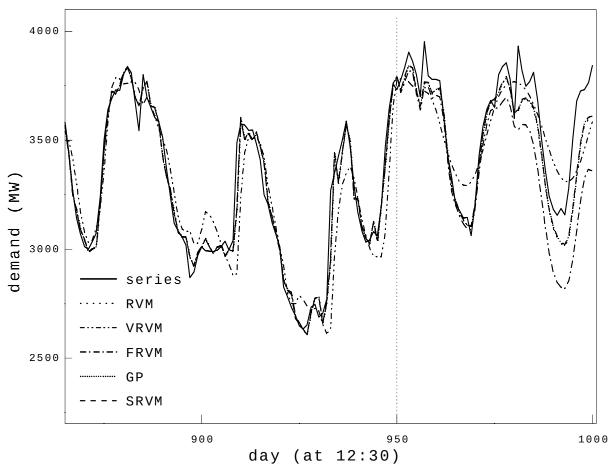

Figure 3. Modelling of the first 950 data points and multi-step (50steps) ahead forecasting of the Electricity load series with by the Bayesian kernel algorithms.

Error	training	forecasting	RVs
	NMSE	NMSE	number
GP	0.66282	0.26742	
RVM	0.74921	0.29562	547
VRVM	0.95554	0.38865	541
FRVM	0.79278	0.41619	628
SRVM	0.81562	0.26733	732

Table 2. Training and testing errors committed by the Bayesian kernel algorithms on the Electricity load series.

In practice one can expect more fluctuating curves of the evolution of the hyperparameters and this depends, first, on the training data, and, second, on the tuning of the other meta-parameters. The performance of the regularized DLR training algorithm depends strongly on the accurate tuning of its parameters which are in interplay so special care has to be taken for its proper initialization. The plots in this Figure 2 can be used as a pattern to find proper algorithm settings.

Electricity load series. An important real-world time series problem is forecasting the electricity demand. This problem has serious economic implications because having capacity to predict well the electricity demand may help to make huge savings. We considered a series of 1015 electricity load measurements (in megawatts) recorded every day at 12 : 30 in a certain utility area. Here for simplicity we assume the series directly and do not smooth for unusual observations. The initial 950 points were taken for training assuming input dimension $d = 15$, and the remaining 50 points were used only for testing. In order to facilitate comparisons, and to convince in the ease of tuning the SRVM the same parameter settings were used as in the previous experiments are given above. The results from the algorithms using the Electricity load series are shown in Table 2.

One can see in Table 2 that SRVM demonstrates lower forecasting error than the other algorithms on multi-step ahead load prediction. What is interesting to note in Table 2 is that the batch RVM and GP produced models with quite close testing performance, although the Gaussian Process model was best on the training data. One notes again that the best forecast was obtained with the SRVM which is better than the GP model, and much better than the results attained by the other RVM approaches. The variational VRVM seem to oversmooth the model, and we think that it is also sensitive to the initialization of the parparameters of the t-student noise distribution. Since the error on the training series made by the SRVM is large, we are inclined to conjecture that the SRVM does not seem to overfit the training data and that is why it shows very good generalization.

V. Discussion

The realisation of the proposed approach to sequential Bayesian learning depends strongly on the behaviour of the regularized DLR training algorithm. First, it has meta-parameters which can be arduous to set so they require making reasonable efforts. Second, the developed Bayesian version of the DLR does not have a strong inductive bias toward less complex models as it does not change the weights rapidly, that is it does not impose a strong pressure toward keeping less kernels. Despite this, we found that it still tends to distinguish relevant from irrelevant weights with respect to the cost function and achieves good generalization on unseen data. Third, the use of the diagonal approximation of the covariance matrix could be a limitation as it discards the cross-information, however the proper probabilistic treatment of the regularization and output noise hyperparameters help to overcome this shortcoming to a great degree as it can be seen from the presented empirical results.

The regularized DLR algorithm is a successor of K1 [13] for approximate Kalman filtering. These are both mechanisms for training linear in the parameters models which are unfortunately quite complex, and extremely difficult to analyse so as to determine their mistake bounds and convergence rates. The main reason for the difficulty to perform error analysis is the complex interaction between the weights and their learning rate meta-parameters.

VI. Conclusion

This paper offered a sequential kernel-based approach suitable for Bayesian learning of large scale times series models. The reported empirical results allow us to think that it is promising for modeling difficult and nonstationary time series. Although this sequential approach has been designed for time series modelling, it is a general purpose method that can be used for doing any kind of regression. It should be clarified that the SRVM is not perfect on the training data but it tends to show very good generalization performance on unseen data as it was demonstrated using time series regression tasks with different characteristics. In this sense it may be considered as altenative to the available Bayesian learners from the RVM family. The SRVM is extremely fast and it makes Bayesian kernel learning practical. Future research will be done to develop variational sequential RVM which can deal properly with various forms of noise in the data.

References

[1] D. Anguita and M. Gagliolo, "MDL based model selection for relevance vector regression", in *Proc. Int. Conf. on Artificial Neural Networks* (ICANN'02), J.Dorronsoro, Ed., Berlin: Springer, 2002, pp.468-473.

[2] C.M. Bishop and M.E. Tipping, "Variational relevance vector machines", in *Proc. 16th Conf. on Uncertainty in Artificial Intelligence*, C. Boutilier and M. Goldszmidt, Eds., San Mateo, CA: Morgan Kaufmann, 2000, pp.46-53.

[3] J.F.G. DeFreitas, M. Niranjan, and A.H. Gee, "Hierarchical Bayesian-Kalman Models for Regularisation and ARD in Sequential Learning", *Neural Computation*, vol.12, N:4, pp.933-953, 2000.

[4] A. D'Souza, S. Vijayakumar, and S. Schaal, "The Bayesian backfitting Relevance Vector Machine", in *Proc. 21st Int. Conf. Machine Learning* (ICML 2004), Banff, Canada, 2004.

[5] Y. Engel, S. Mannor and R. Meir, "The kernel recusrive least squares algorithm", *IEEE Tran. Signal Processing*, vol.52, N:8, pp.2275-2285, 2004.

[6] A. Girard, C.E. Rasmussen, J.Quiñonero-Candela and R. Murray-Smith, "Multiple-step ahead prediction for non linear dynamic systems– a Gaussian Process treatment with propagation of the uncertainty", in *Advances in Neural Information Processing Systems 15*, S.Becker, S.Thrun and K.Obermayer, Eds., Cambridge, MA: MIT Press, 2003, pp.529-536.

[7] J. Kivinen, A.J. Smola, and R.C. Williamson, "Online learning with kernels", *IEEE Tran. Signal Processing*, vol.52, N:8, pp.2165- 2176, 2004.

[8] D.J.C. MacKay, "Bayesian interpolation", *Neural Computation*, vol.4, N:3, pp.415-447, 1992.

[9] K.-R. Müller, A. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik, "Using support vector machines for time series prediction", in *Advances in Kernel Methods- Support Vector Learning*, B.Schölkopf, C.Burges, and A.Smola, Eds., Cambridge, MA: The MIT Press, 1998, pp.243-254.

[10] I. Nabney, *Netlab: Algorithms for Pattern Recognition*, London: Springer-Verlag, 2002.

[11] J. Quiñonero-Candela and L.K. Hansen, "Time series prediction based on the Relevance Vector Machine with adaptive kernels", in *Proc. Int. Conference on Acoustics, Speech, and Signal Processing*, Piscataway, NJ: IEEE Press, 2002, pp.985-988.

[12] R. Sutton, "Gain Adaptation Beats Least Squares?", in *Proc. Seventh Yale Workshop on Adaptive and Learning Systems*, Yale University, New Haven, CT, 1992, pp.161-166.

[13] M.E. Tipping, "Sparse Bayesian learning and the relevance vector machine", *Journal of Machine Learning Research*, vol.1, pp.211-244, 2001.

[14] M.E. Tipping. and A.C. Faul, "Fast marginal likelihood maximisation for sparse Bayesian models", in *Proc. Ninth Int. Workshop on AI and Statistics*, C.M.Bishop and B.J.Frey, Eds., Key West, FL, 2003.

[15] C.K.I. Williams and C.E. Rasmussen, "Gaussian Processes for regression", in *Advances in Neural Information Processing Systems 8*, D.S.Touretzky, M.C.Mozer, M.E.Hasselmo, Eds., Cambridge, MA: MIT Press, 1996, pp.514-520.

[16] D.P. Wipf, J.A. Palmer, and B.D. Rao, "Perspectives on sparse Bayesian learning," *Advances in Neural Information Processing Systems 16*, Cambridge, MA: MIT Press, 2004.