

Sequential Bayesian Kernel Modelling with Non-Gaussian Noise

Nikolay Y. Nikolaev

Lilian M. de Menezes

Department of Computing

Faculty of Management

Goldsmiths College, University of London

Cass Business School

London SE14 6NW

City University

United Kingdom

London, U.K.

N.Nikolaev@gold.ac.uk

l.demenezes@city.ac.uk

Abstract

This paper presents a sequential Bayesian approach to kernel modelling of data, which contain unusual observations and outliers. The noise is heavy tailed described as a one-dimensional mixture distribution of Gaussians. The development uses a factorised variational approximation to the posterior of all unknowns, that helps to perform tractable Bayesian inference at two levels: 1) sequential estimation of the weights distribution (including its mean vector and covariance matrix); and 2) recursive updating of the noise distribution and batch evaluation of the weights prior distribution. These steps are repeated, and the free parameters of the non-Gaussian error distribution are adapted at the end of each cycle. The reported results show that this is a robust approach that can outperform standard methods in regression and time series forecasting.

1 Introduction

Kernel methods [Vapnik, 1995], [Schölkopf and Smola, 2001], [Suykens et al., 2002], [Rasmussen and Williams, 2006] have become increasingly popular in machine learning due to their ability to produce well performing linear-in-the-weights models. Such a contemporary tool is the Relevance Vector Machine (RVM) [Tipping, 2001] for probabilistic modelling with arbitrary kernels. It carries out hierarchical Bayesian inference by estimating the posterior distribution of the weight parameters after computing their priors and the output noise hyperparameters by maximization of the marginal likelihood.

The RVM has several attractive characteristics. First, it uses kernels that transform the inputs into a higher-dimensional space where they become more amenable to learning [Vapnik, 1995]. Second, it provides a learning algorithm that performs simultaneously parameter search and model selection leading to sparse models that generalise well [Tipping, 2001]. Third, this is a Bayesian method [MacKay, 1992] that yields probabilistic forecasts. The original RVM however operates in offline mode, which can be an obstacle for real-time applications because: 1) it uses offline formulae that sometimes cannot be evaluated safely due to ill-conditioning of the data matrices; 2) it requires storage and processing of large amounts of data, which is computationally demanding; and 3) it handles normally distributed data and any implications that this may have for the automatic determination of the kernels are unknown [Cawley and Talbot, 2005]. Moreover, it is unsuitable for time series modelling because time dependencies in the data are not considered.

This paper develops a kernel machine that processes data sequentially and handles outliers through a non-Gaussian noise description. Following the method of variational inference [MacKay, 1995], [Jordan et al., 1999], [Attias, 2000], [Ghahramani and Beal, 2001], [Roberts and Penny, 2002], [Beal, 2003] a factorised distribution is introduced to approximate the joint posterior of all variables in order to achieve tractable inference. Among the available variational RVM approaches [Bishop and Tipping, 2000], [Faul and Tipping, 2001], [Sato and Oba, 2002], we adopt the probabilistic framework of the offline VRVM [Tipping and Lawrence, 2005] that uses heavy-tailed noise made as a one-dimensional mixture of Gaussians [Penny and Roberts, 2000].

A Sequential Variational RVM (SVRVM) for robust Bayesian inference is elaborated at two levels. First, we derive recursive estimation formula for the mean vector and the covariance matrix as sufficient statistics of the weights distribution. Second, we obtain a noise prior hyperparameter estimation formula that allows to perform sequential optimisation of the noise distribution. Point estimates of the free parameters of the non-Gaussian noise distribution are found offline. The distribution of the weight priors is updated with the whole batch of data so as to achieve stability in the algorithm performance.

The presented research relates the SVRVM machine to several standard modelling tools: a Multilayer Perceptron (MLP) neural network with a Kalman Filter [de Freitas et al., 2000], an MLP with Student- t noise [Briegel and Tresp, 1999], a Kernel Recursive Least Squares (KRLS) algorithm [Engel et al., 2004], and a standard Box-Jenkins (SARIMA) tool. There are provided experimental results on regression and forecasting of electricity price time series data.

In the next section, we introduce robust Bayesian modelling with the kernel RVM. Section three presents the method of variational inference. The next section four describes the mechanisms of the SVRVM. Section five reports our results which indicate that the SVRVM can outperform competitive approaches in causal and time series modelling. Finally, we conclude with a brief discussion, which summarizes the key points of present study and their implication for future work.

2 Robust Bayesian Modelling

2.1 Time Series Regression

Given a series of observations: $\dots, v_t, v_{t+1}, v_{t+2}, \dots$ sampled at discrete time intervals, the task of time series modelling is to map past values into future values. The observable values v_t are usually converted into the range $[0,1]$ by scaling $x_t = \text{Normalize}(v_t)$, and the subject of the analysis is then the normalized (or standardized) series: $\dots, x_t, x_{t+1}, x_{t+2}, \dots$. The Takens theorem [Takens, 1981] justifies the use of *delayed* (lagged) vectors \mathbf{x} to reconstruct the dynamic system that has generated this series:

$$x_{t+1} = f(\mathbf{x}) = f(x_{t-(d-1)\tau}, x_{t-(d-2)\tau}, \dots, x_t) \quad (1)$$

where d is *embedding dimension*, and τ is *time delay*.

The series can be viewed as a set $D = \{(\mathbf{x}_t, y_t)\}_{t=1}^T$ of size T , where the input vectors $\mathbf{x}_t = (x_t, x_{t-1}, \dots, x_{t-d-1}) \in \mathcal{R}^d$ have dimension d . The aim is to find models f that map vectors \mathbf{x} to values $y \equiv x_{t+1}$. We consider functions $y = f(\mathbf{x})$ that are linear superpositions of kernels [Vapnik, 1995], [Smola and Schlopkof, 2001]:

$$y = \sum_{i=1}^M K(\mathbf{x}, \mathbf{x}_i)w_i + \varepsilon = \mathbf{k}(\mathbf{x})\mathbf{w} + \varepsilon \quad (2)$$

where $\mathbf{w} = [w_1, w_2, \dots, w_M]^T$ is the vector of weight parameters, $K(\mathbf{x}, \mathbf{x}_i)$ are the basis functions (also called kernels), $\mathbf{k}(\mathbf{x}) = [K(\mathbf{x}, \mathbf{x}_1), K(\mathbf{x}, \mathbf{x}_2), \dots, K(\mathbf{x}, \mathbf{x}_M)]$ are row kernel vectors, and ε is the output noise. Here M denotes the model size and $M < N$. In this paper we use the popular Gaussian kernel $K(\mathbf{x}, \mathbf{x}_i) = \exp[-(\mathbf{x} - \mathbf{x}_i)^T(\mathbf{x} - \mathbf{x}_i)/(2s^2)]$, where s^2 is the width (or spread) parameter.

2.2 Modelling with Non-Gaussian Noise

Subject of particular interest is processing real-world data which are typically contaminated with outliers and require treatment by distributions that have longer and/or thicker tails than the Normal (Gaussian) distribution. Taking however the Student- t distribution to describe *non-Gaussianity* leads to intractable marginal likelihood integrals. Recent research shows that this problem can be avoided by replacing the Student- t distribution by an equivalent one-dimensional mixture of a large number of Gaussians [Penny and Roberts, 2000]. This idea enables variational approximation to the posterior distribution that helps to produce analytical estimates not only for the parameters but also for the priors (hyperparameters) that govern the modelling precision.

The infinite mixture of zero-mean Gaussians that allows tractable inference is [Penny and Roberts, 2000]:

$$p(e_t|\mathbf{w}, c, d) = \int_0^\infty p(e_t|\beta_t)p(\beta_t|c, d)d\beta_t \quad (3)$$

where the noise terms are $p(e_t|\beta_t) = \mathcal{N}(e_t|0, \beta_t^{-1})$, and the corresponding mixture weights are $p(\beta_t|c, d) = \mathcal{G}(\beta_t|c, d) = (d^c/\Gamma(c))\beta_t^{c-1} \exp(-\beta_t d)$. Here the symbol Γ denotes the Gamma function. The constants c and d correspond to the free parameters of an equivalent Student- t distribution: degrees of freedom v , and width σ_Γ ($\sigma_\Gamma^2 = \beta_t^{-1}$). More precisely, the relationship is $v = 2c$, and $\sigma_\Gamma = \sqrt{d/c}$.

2.3 Sparse Bayesian Learning

The probabilistic inference applies the Bayes' rule to obtain the entire posterior distribution of the model parameters and also the prior hyperparameters that control the model precision. The initial knowledge in the weights and in the model uncertainties are encoded into corresponding priors. The beliefs in the weights are updated after seeing the data with their *likelihood*:

$$p(\mathbf{y}|\mathbf{w}, \boldsymbol{\beta}) = (2\pi)^{-N/2} \prod_{t=1}^T \beta_t^{1/2} \exp(-0.5\beta_t e_t^2(\mathbf{w})) \quad (4)$$

where β_t denotes the inverse noise variance at a particular moment t , which was assumed to have Gamma distribution in order to explain non-Gaussian effects, and the error is $e_t(\mathbf{w}) = y_t - \mathbf{w}^T \mathbf{k}(\mathbf{x}_t)$.

Assuming independent noise, the distribution of the *noise hyperparameters* β_t is defined by the product:

$$p(\boldsymbol{\beta}|c, d) = \prod_{t=1}^T \mathcal{G}(\beta_t|c, d) \quad (5)$$

where c and d correspond to the free parameters of the noise distribution.

Following the sparse Bayesian learning approach of the RVM [Tipping, 2001], a local weight prior is imposed on each weight $p(w|\alpha) = \mathcal{N}(w|0, \alpha^{-1})$ with a *prior hyperparameter* α^{-1} (inverse weight variance). During

training these prior hyperparameters are tuned individually and some of them become very large, thus, shrinking the corresponding weights to zero and effectively pruning them from the model. The process of sparsification, which impacts the generalisation performance, is controlled using a pruning threshold. This is described with the following multidimensional *weight prior*:

$$p(\mathbf{w}|\boldsymbol{\alpha}) = (2\pi\alpha_i^{-1})^{1/2} \prod_{m=1}^M \exp(-0.5\alpha_m w_m^2) \quad (6)$$

where $\alpha_m > 0$, $1 \leq m \leq M$, and $\boldsymbol{\alpha} = [\alpha_1, \dots, \alpha_M]$ is the vector of weight priors.

Each weight prior hyperparameter α_m is governed by an informative *hyperprior*:

$$p(\boldsymbol{\alpha}|a, b) = \prod_{m=1}^M \mathcal{G}(\alpha_m|a, b) \quad (7)$$

where a and b are given small values to make a broad prior [Tipping and Lawrence, 2005].

The *joint posterior* of the weight parameters and hyperparameters in this model is:

$$p(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{w}, \boldsymbol{\beta})p(\boldsymbol{\beta}|c, d)p(\mathbf{w}|\boldsymbol{\alpha})p(\boldsymbol{\alpha}|a, b)}{p(\mathbf{y})} \quad (8)$$

where $p(\mathbf{y}|\mathbf{w}, \boldsymbol{\beta})$ is the likelihood, $p(\boldsymbol{\beta}|c, d)$ is the noise hyperparameter, $p(\mathbf{w}|\boldsymbol{\alpha})$ is the belief in the weights, $p(\boldsymbol{\alpha}|a, b)$ is the weight prior hyperparameter, and $p(\mathbf{y})$ is the marginal likelihood (evidence).

The RVM [Tipping, 2001] obtains analytical estimates of the weights by type-II maximum likelihood. It decomposes the joint posterior $p(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}|\mathbf{y}) = p(\mathbf{w}|\mathbf{y}, \boldsymbol{\alpha}, \boldsymbol{\beta})p(\boldsymbol{\alpha}, \boldsymbol{\beta}|\mathbf{y})$, which leads to an analytically tractable expression $p(\mathbf{w}|\mathbf{y}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = p(\mathbf{y}|\mathbf{w}, \boldsymbol{\beta})p(\mathbf{w}|\boldsymbol{\alpha})/p(\mathbf{y}|\boldsymbol{\alpha}, \boldsymbol{\beta})$. According to the evidence procedure [MacKay, 1992] the hyperparameters can be found by maximising the marginal likelihood $p(\mathbf{y}|\boldsymbol{\alpha}, \boldsymbol{\beta}) = \int p(\mathbf{y}|\mathbf{w}, \boldsymbol{\beta})p(\mathbf{w}|\boldsymbol{\alpha})d\mathbf{w}$, as in this case of Gaussian linear modelling it is a convolution of Gaussians.

3 Variational Inference

The posterior $p(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}|\mathbf{y})$ of our heavy tail, non-Gaussian noise model can not be obtained by evidence maximisation, however, because the marginal likelihood $p(\mathbf{y})$ in this case is intractable. This problem can be overcome using variational Bayesian inference [MacKay, 1995], [Jordan et al., 1999], [Attias, 2000], [Ghahramani and Beal, 2001], [Roberts and Penny, 2002], [Beal, 2003]. According to the variational Bayes realistic probability inference can be achieved through an approximating auxiliary distribution $Q(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ to the joint posterior, instead of choosing directly a tractable form for it. Analytical formula for the optimal posterior are obtained in terms of the means and variances of the ingredients of the variational distribution $Q(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})$, which is typically an ensemble from known, preselected distributions [MacKay, 1995].

The studied kernel model with non-Gaussian noise becomes amenable to inference by representing its log marginal likelihood as a sum of two terms:

$$\log p(\mathbf{y}) = \log \int p(\mathbf{y}, \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) d\mathbf{w} d\boldsymbol{\alpha} d\boldsymbol{\beta} \quad (9)$$

$$\geq \int Q(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \log \left\{ \frac{p(\mathbf{y}, \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})}{Q(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})} \right\} d\mathbf{w} d\boldsymbol{\alpha} d\boldsymbol{\beta} - \int Q(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \log \left\{ \frac{p(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta} | \mathbf{y})}{Q(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})} \right\} d\mathbf{w} d\boldsymbol{\alpha} d\boldsymbol{\beta} \quad (10)$$

$$= \mathcal{L}[Q(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})] + KL[Q(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) || p(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta} | \mathbf{y})] \quad (11)$$

which are known as the *lower bound* $\mathcal{L}[Q(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})]$, and the *Kullback-Leibler divergence* KL of the approximation $Q(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ from the true posterior. Since the optimisation of the lower bound is equivalent to decreasing the divergence, only one of these terms needs to be computable.

The learning process then seeks such a parametrisation of the ingredient approximating distributions that maximizes the lower bound of the log marginal, as a functional of these distributions:

$$\mathcal{L}[Q(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})] = \int Q(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \log \left\{ \frac{p(\mathbf{y}, \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})}{Q(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})} \right\} d\mathbf{w} d\boldsymbol{\alpha} d\boldsymbol{\beta} \quad (12)$$

where the maximum is attained when the variational distribution is equal to the joint posterior distribution $p(\mathbf{y}, \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = p(\mathbf{y} | \mathbf{w}, \boldsymbol{\beta}) p(\boldsymbol{\beta} | c, d) p(\mathbf{w} | \boldsymbol{\alpha}) p(\boldsymbol{\alpha} | a, b)$.

The lower bound $\mathcal{L}[Q(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})]$ is maximised iteratively with two alternating steps: 1) inference of the weights posterior distribution keeping the distributions of the hyperparameters fixed, and 2) inference of the distributions of the hyperparameters with the available weights posterior. Each step effectively makes the lower bound more and more tight. The convergence of this procedure to a local maximum in the function space of variational distributions is theoretically guaranteed [Beal, 2003].

3.1 Factorised Variational Approximation

The challenge in variational inference is how to select the form of the distribution $Q(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ so as to compute efficiently the lower bound. The configuration of the variational distribution should make possible tractable inference, and should be flexible enough to allow a close approximation to the true posterior.

A popular strategy is to develop separable distributions that factorise over the variables in the model [MacKay, 1995]. This strategy relies on the assumption that the variables are independent. Maximisation of the lower bound is pursued by separate optimisation over each particular factor with respect to its parameters. Having independent factor means that, after initialisation with the current estimates of the parameters, the factors can be re-estimated one-by-one by updating their parameters until some termination criteria is satisfied.

The optimisation formula for the concrete factor parameters are derived by taking the functional derivatives of the lower bound, with respect to the variational distribution over the parameters of interest, and setting to zero [Beal, 2003]. This leads to a general equation that averages the log of the joint distribution with respect to all variables not in this factor as follows [Bishop, 2006]:

$$\log Q_i(\mathbf{z}_i) = \langle \log p(\mathbf{z}_i, \mathbf{y}) \rangle_{i \neq j} + \text{const.} \quad (13)$$

where $\langle \cdot \rangle_{i \neq j}$ denotes expectation with respect to a distribution over all variables \mathbf{z}_i for $i \neq j$, and the corresponding variable in our model \mathbf{z}_i can be either \mathbf{w} , or $\boldsymbol{\alpha}$, or $\boldsymbol{\beta}$. Applied to the conjugate-exponential family of models, to which the developed kernel machine belongs, this leads to standard distributions.

3.2 The Variational Kernel Machine

The VRVM [Tipping and Lawrence, 2005] exploits variational inference using the following *factorized approximating distribution*:

$$Q(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = Q_{\mathbf{w}}(\mathbf{w})Q_{\boldsymbol{\alpha}}(\boldsymbol{\alpha})Q_{\boldsymbol{\beta}}(\boldsymbol{\beta}) \quad (14)$$

which allows us to treat the weights and the hyperparameters independently.

Solving for the factor $Q_{\boldsymbol{\alpha}}(\boldsymbol{\alpha})$ is accomplished using the log of the terms from the joint probability $p(\mathbf{y}, \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ that depend on $\boldsymbol{\alpha}$ and taking the expectation with respect to the remaining variable \mathbf{w} . Written for an individual hyperparameter α the optimisation equation is: $\log Q_m(\alpha_m) = \log p(\alpha_m | a, b) + \langle \log p(w_m | \alpha_m) \rangle_{\mathbf{w}} + \text{const.}$, which after performing the operations (Appendix B) yields a quadratic form that indicates the Gamma distribution. The whole factor distribution then becomes:

$$Q_{\boldsymbol{\alpha}}(\boldsymbol{\alpha}) = \prod_{m=1}^M Q_m(\alpha_m) = \prod_{m=1}^M \mathcal{G}(\alpha_m | \tilde{a}, \tilde{b}_m) \quad (15)$$

where the parameters are $\tilde{a} = a + 0.5$, and $\tilde{b}_m = b + 0.5 \langle w_m^2 \rangle$ using a and b from (7).

Similar manipulations are carried out for the factor $Q_{\boldsymbol{\beta}}(\boldsymbol{\beta})$ with different two terms from the joint distribution $p(\mathbf{y}, \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ that involve the noise hyperparameter β_t : $\log Q_t(\beta_t) = \log p(\beta_t | c, d) + \langle \log p(y_t | \mathbf{w}, \beta_t) \rangle_{\mathbf{w}} + \text{const.}$, where the outside variable in this factor is also the weight vector \mathbf{w} . Taking the expectation and making simplifications gives a Gamma distribution (Appendix B). The factorised noise distribution is:

$$Q_{\boldsymbol{\beta}}(\boldsymbol{\beta}) = \prod_{t=1}^T Q_t(\beta_t) = \prod_{t=1}^T \mathcal{G}(\beta_t | \tilde{c}, \tilde{d}_t) \quad (16)$$

where \tilde{c} and \tilde{d}_t are the parameters of the Gamma distribution (section 4.3).

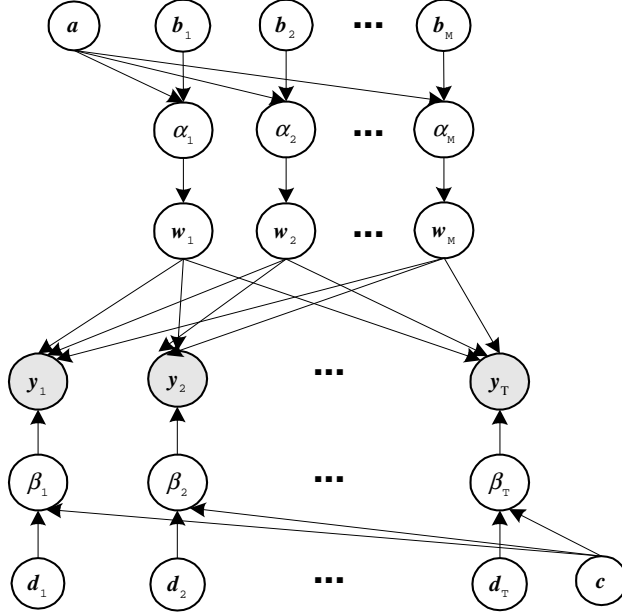


Figure 1. Graphical model representing the probabilistic hierarchical structure of the variational kernel machine.

The derivation of the probabilistic form of the weight factor $Q_{\mathbf{w}}(\mathbf{w})$ is more involved as it requires to take expectations from two terms: $\log Q_{\mathbf{w}}(\mathbf{w}) = \langle \log p(\mathbf{y}|\mathbf{w}, \boldsymbol{\beta}) \rangle_{\boldsymbol{\beta}} + \langle \log p(\mathbf{w}|\boldsymbol{\alpha}) \rangle_{\boldsymbol{\alpha}} + \text{const.}$, and the averaging is performed with respect to the non-factor variables $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ (Appendix B). This allows us to infer a quadratic form that is recognised as a Gaussian distribution:

$$Q_{\mathbf{w}}(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \quad (17)$$

with mean $\boldsymbol{\mu} = \boldsymbol{\Sigma} \mathbf{K}^T \mathbf{B} \mathbf{y}$, and covariance matrix $\boldsymbol{\Sigma} = (\mathbf{K}^T \mathbf{B} \mathbf{K} + \mathbf{A})^{-1}$, where $\mathbf{K}_t = \{K(\mathbf{x}_i, \mathbf{x}_j)\}$, $1 \leq i \leq t$, $1 \leq j \leq M$. The matrices are: $\mathbf{A} = \text{diag}(\langle \alpha_1 \rangle, \langle \alpha_2 \rangle, \dots, \langle \alpha_M \rangle)$, and $\mathbf{B} = \text{diag}(\langle \beta_1 \rangle, \langle \beta_2 \rangle, \dots, \langle \beta_T \rangle)$, having averaged elements computable as follows: $\langle \alpha_m \rangle = \tilde{a}/\tilde{b}_m$ and $\langle \beta_t \rangle = \tilde{c}/\tilde{d}_t$.

The probabilistic structure of the variational kernel machine, adopted by the SVRVM, is given in Figure 1.

4 Sequential Variational Learning

The original VRVM [Tipping and Lawrence, 2005] operates in batch mode, which could be an obstacle for genuine real-time applications. A reasonable alternative approach is to perform incremental learning using one input at a time. Sequential variational inference can be performed by recursive optimisation of the weight factor $Q_{\mathbf{w}}(\mathbf{w})$ and the noise factor $Q_{\boldsymbol{\beta}}(\boldsymbol{\beta})$ with the arrival of the data. The aim is to achieve sequential maximization of the current lower bound $\mathcal{L}[q(\mathbf{w}_t, \boldsymbol{\alpha}, \boldsymbol{\beta}_t)]$ through the recursive optimisation of the weight distribution $Q_{\mathbf{w}}(\mathbf{w}_t)$,

along with computation of the corresponding mixture component of the noise $Q_t(\beta_t)$, with respect to all data up to the current moment. Recursive estimation formula for these factor distributions can be inferred by solving separately the corresponding equations:

$$\log Q_{\mathbf{w}}(\mathbf{w}_t) = \langle \log p(\mathbf{y}_t | \mathbf{w}_t, \boldsymbol{\beta}) \rangle_{\boldsymbol{\beta}} + \langle \log p(\mathbf{w}_t | \boldsymbol{\alpha}) \rangle_{\boldsymbol{\alpha}} + \text{const.} \quad (18)$$

$$\log Q_t(\beta_t) = \log p(\beta_t | c, d) + \langle \log p(y_t | \mathbf{w}_t, \beta_t) \rangle_{\mathbf{w}_t} + \text{const.} \quad (19)$$

where weights factor depends on all inputs up to and including the current time t .

This treatment of the noise makes the algorithm attractive for time-series modelling, because it mitigates the effects of outliers, and so helps to achieve stable evolution of the learning process.

4.1 Recursive Weight Estimation

The factor $Q_{\mathbf{w}}(\mathbf{w}_t)$ can be optimised successively by temporal adaptation and propagation of its instantaneous mean, which maximises the likelihood with respect to all data arrived till the moment. We derive the following *probabilistic recursive weighted least squares* formula (Appendix A):

$$\boldsymbol{\mu}_t = \boldsymbol{\mu}_{t-1} + \beta_t e_t \boldsymbol{\Sigma}_t \mathbf{k}_t - \alpha_i \boldsymbol{\Sigma}_t \mathbf{R}_{t-1} \boldsymbol{\mu}_{t-1} \quad (20)$$

where $\boldsymbol{\mu}_t$ is the current mean weigh vector, $\boldsymbol{\Sigma}_t$ is the covariance matrix computed recursively from $\boldsymbol{\Sigma}_{t-1}$, e_t is the error at time t , and α_i is the prior hyperparameter with index $i = t \bmod M$.

The error e_t in the second term $\beta_t e_t \boldsymbol{\Sigma}_t \mathbf{k}_t$, in (18), determines the local search direction and the noise hyperparameter β_t impacts its magnitude. The third term $\alpha_i \boldsymbol{\Sigma}_t \mathbf{R}_{t-1} \boldsymbol{\mu}_{t-1}$ in (18) accounts for the regularization that is applied partially until the number of training examples reaches the boundary $M * (N \text{ div } M)$. The index of the next hyperparameter α_i , $1 \leq i \leq M$, is such that $i = t \bmod M$ and the value of the hyperparameter α_i is scaled by a special matrix \mathbf{R}_t ($M \times M$). This matrix has zero elements, except in the diagonal at position i , which is $[\mathbf{R}_t]_{ii} = 1.0/z_i$, $i = t \bmod M$. The denominator is a constant $z_i = (N \text{ div } M)$ if $t < M * (N \text{ div } M)$ or a very large number, e.g. $z_i = 1.0e10$ [Nikolaev and Iba, 2006,p.230].

4.2 Recursive Covariance Estimation

A distinguishing feature of our the formula for the computation of the inverted matrix $\boldsymbol{\Sigma}_t = (\mathbf{K}_t^T \mathbf{B}_t \mathbf{K}_t + \mathbf{A}_t)^{-1}$ from $\boldsymbol{\Sigma}_{t-1}$ is that it incorporates the local weight prior hyperparameters. It implements exactly the RVM approach to Bayesian inference, because the regularization effect of each local hyperparameter is added in fractions, when re-estimating the covariance matrix [Ljung and Söderström, 1983].

The sequential modification of the inverted covariance matrix: $\Sigma_t^{-1} = \Sigma_{t-1}^{-1} + \mathbf{k}_t \beta_t \mathbf{k}_t^T + \alpha_i \mathbf{R}_t$, including factors $\alpha_i \mathbf{R}_t$ from partial individual hyperparameters, is obtained using the matrix inversion lemma. The *recursive covariance matrix update* formula is given by [Nikolaev and Iba, 2006,p.231]:

$$\Sigma_t = \Sigma_{t-1} - \beta_t \Sigma_{t-1} \mathbf{k}_t^* (\beta_t \mathbf{k}_t^{*T} \Sigma_{t-1} \mathbf{k}_t^* + \Lambda_t)^{-1} \mathbf{k}_t^{*T} \Sigma_{t-1} \quad (21)$$

where Λ_t is the inverted version $(\Lambda_t^{-1})^{-1}$ of the regularization matrix $\Lambda_t^{-1} = \begin{bmatrix} 1 & 0; & 0 & \alpha_i [\mathbf{R}_t]_{ii} \end{bmatrix}$ (2×2), \mathbf{k}_t^* is the kernel matrix: $\mathbf{k}_t^* = \begin{bmatrix} \mathbf{k}_t^T; & 0 \dots 0 & 1_i & \dots 0 \end{bmatrix}^T$ ($M \times 2$) formed from the kernel column vector \mathbf{k}_t^T and another column vector of zeroes, except for the i^{th} position that is equal to 1. Accordingly, the scaling constant (z_i) in the matrix Λ_t allows increments to the regularization effect by fractions $(\alpha_i / (N \text{ div } M))^{-1}$.

4.3 Noise Hyperparameter Estimation

Having a different noise level at each data point allows us to mitigate the effects from outliers, and, thus, to perform robust weights estimation. That is why, we suggest to modify sequentially the noise factor $Q_\beta(\beta)$, so as to reflect changes in the variance that are due to the arrival of new input data at time instant t , by optimising each time component $Q_t(\beta_t)$. Solving the logarithm of this factor $\log Q_t(\beta_t)$ (17) by taking the logarithm of the term $p(\beta_t | c, d)$, summing with the expectation of $\log p(y_t | \mathbf{w}_t, \beta_t)$ with respect to the weights vector and setting it to zero, yields the following *output noise hyperparameter update* formula (Appendix B):

$$\beta_t = \frac{\tilde{c}}{\tilde{d}_t} = \frac{c + 0.5}{d + 0.5 \langle y_t^2 - 2y_t \mathbf{k}_t \langle \mathbf{w}_t \rangle + \mathbf{k}_t^T \langle \mathbf{w}_t \mathbf{w}_t^T \rangle \mathbf{k}_t \rangle} \quad (22)$$

where $\tilde{c} = c + 0.5$, and $\tilde{d}_t = d + 0.5 \langle y_t^2 - 2y_t \mathbf{k}_t \langle \mathbf{w}_t \rangle + \mathbf{k}_t^T \langle \mathbf{w}_t \mathbf{w}_t^T \rangle \mathbf{k}_t \rangle$. Here the averaged estimates are computed as follows $\langle \mathbf{w}_t \rangle = \boldsymbol{\mu}_t$, and $\langle \mathbf{w}_t \mathbf{w}_t^T \rangle = \boldsymbol{\mu}_t \boldsymbol{\mu}_t^T + \Sigma_t$.

4.4 Free Parameters of the Noise

Since there are no analytical formula for direct estimation of the free parameters of the noise process c and d , one can use a search technique to learn them from the training data. Following the VRVM framework, we consider the scaled conjugate gradients method [Möller, 1993] to estimate offline the values of c and d , by using the *derivatives of the free parameters* [Tipping and Lawrence, 2005]:

$$\frac{\partial \langle \log p(\mathbf{y} | \mathbf{x}, \mathbf{w}, \boldsymbol{\beta}) \rangle}{\partial c} = N \log d - N \psi(c) + \sum_{t=1}^N \log p(\beta_t) \quad (23)$$

$$\frac{\partial \langle \log p(\mathbf{y} | \mathbf{x}, \mathbf{w}, \boldsymbol{\beta}) \rangle}{\partial d} = \frac{Nc}{d} - \sum_{t=1}^N \beta_t \quad (24)$$

where $\psi(\cdot)$ denotes the digamma function $\psi(c) = \partial / \partial c [\log \Gamma(x)]$.

5 Relative Performance of the SVRVM

The performance of the SVRVM is compared here with other contemporary modelling tools. The studies begin with a regression problem, where a benchmark data set has been contaminated with extreme values so that we can assess the effectiveness on noisy data. Next, the problem of time series modelling is addressed in two studies on forecasting daily electricity spot prices, which are known to be mean reverting processes with spikes. Considered in modelling context, the SVRVM is related to neural networks as well as kernel tools when investigating its performance. It should be noted also that the SVRVM may be envisioned as equivalent to a Gaussian Process, which is a kind of a neural network [Rasmussen and Williams, 2006].

Two Multilayer Perceptron (MLP) neural networks were developed, and trained with different algorithms: an Extended Kalman Filter (MLP_{Kalman}) [de Freitas et al., 2000], and an extended Kalman Filter with Student- t noise model (MLP_{Student- t}) [Briegel and Tresp, 1999]. In addition, we implemented two kernel-based methods: a Kernel Recursive Least Squares (KRLS) algorithm [Engel et al., 2004] and a RVM that uses the Hubers' M-estimator function [Huber, 1981]. We also made Box-Jenkins ARIMA models, since they are traditional benchmarks in time series forecasting exercises.

The MLP neural networks are popular nonlinear regression tools that are increasingly being used by the forecasting community [De Gooijer and Hyndman, 2006]). The first algorithm that we use, MLP_{Kalman}, is an Extended Kalman Filter in which the highly nonlinear-in-the-weights network model is linearized using the output derivatives with respect to the network weights [Haykin, 2001]. The EKF is known to be superior to the standard and the incremental backpropagation approaches, as it uses second-order information which speeds up its convergence. We implemented a probabilistic version with dynamic weight noise and output noise, which are re-estimated after each training input [de Freitas et al., 2000].

The second network filter, MLP_{Student- t} , is a Fisher scoring algorithm that handles outliers using a Student- t noise model [Briegel and Tresp, 1999]. This is an online training algorithm that involves a forward and a backward smoothing passes through the data to obtain the mode of the posterior with respect to the weights. We exploited the probabilistic nature of this algorithm by doing maximum likelihood estimation of the noise scale hyperparameter ($\tilde{d} - d$) in Expectation Maximisation (EM) style as recommended by the authors [Briegel and Tresp, 1999] (while the free parameters of the noise distribution c and d are kept fixed as in SVRVM). In addition to this we also performed state noise estimation of a common hyperparameter α . The EM cycle was repeated 10 times to facilitate comparisons with the SVRVM (whose cycle was also repeated 10 times).

The two neural networks MLP_{Kalman} and MLP_{Student- t} were designed with the same architecture: 3 hidden

units with tangential activation functions in the first two studies, and 4 hidden neurons in the last study. The initial weights were set to small values (10^{-4}). In order to be consistent with SVRVM, the covariance matrix was initialised with diagonal elements $[\Sigma_0]_{ii} = 200$. The MLP_{Student-t} used learning rate $\eta = 0.1$ and noise matrix with diagonal elements set to $[\mathbf{Q}_s]_{ii} = 10^{-5}$. We found that these two parameters have critical impact on the performance of the Briegel and Tresp’s algorithm, as they influence its convergence and its numerical stability when inverting the covariance matrix. Matrix inversion was made using Cholesky decomposition .

The Kernel Recursive Least Squares (KRLS) algorithm [Engel et al., 2004] was designed especially for sequential training of kernel models. We implemented it using Gaussian kernels with spread $s^2 = 1$ in order to be comparable to the SVRVM. This is a greedy algorithm that adds one kernel at a time to the model when this kernel, corresponding to the next arriving input, is below a predefined threshold. Thus, it performs online constructive model sparsification along with the estimation of the weights. The KRLS, however, is not probabilistic because it does not produce estimates of the entire weight distribution. Rather it yields point estimates of the weights with a standard regularisation technique. We used regularisation $\lambda = 10^{-5}$ and pruning threshold $v = 0.01$ as originally suggested by [Engel et al., 2004].

The SVRVM was initialized with the following parameters: Gaussian kernel, $s^2 = 1$ (because the data has been previously standardized), initial weight prior hyperparameter $\alpha_0 = 10^{-5}$, initial output noise hyperparameter $\beta_0 = 1$, pruning threshold $\alpha_{\max} = 4 \times 10^{-3}$, and covariance matrix with initial diagonal elements $[\Sigma_0]_{ii} = 200$. The cycle for learning the parameters and the hyperparameters was repeated 10 times.

In order to be informative about the learning potential of the SVRVM we also developed another alternative version RVM_{Huber} which is essentially the same kernel machine that handles the outliers using the Hubers’ M-estimator function: $\rho(e) = 0.5e^2$ for $|e| \leq \tau$, or $\rho(e) = \tau|e| - 0.5\tau^2$ otherwise [Huber, 1981]. This is a popular function for suppressing the effects of the outliers in the statistical community which we found suitable also for our kernel models using the recommended threshold $\tau = 1.435\sigma$ [Huber, 1981]. Since the results of RVM_{Huber} are very close to these from the SVRVM, we only report empirical results and do not provide plots in the figures.

In our two empirical studies with time series data, we identified and estimated competing models using the traditional Box-Jenkins procedure and used the BIC criterion for model selection. All nonlinear models used standardized values and the observations from the previous two weeks (x_{t-1}, \dots, x_{t-14}) as inputs. By contrast, we used the raw data in order to identify and estimate the Box-Jenkins Arima models. In doing so, we have not attempted to smooth the time series by replacing outliers, as it is often done when using standard linear models for time series forecasting.

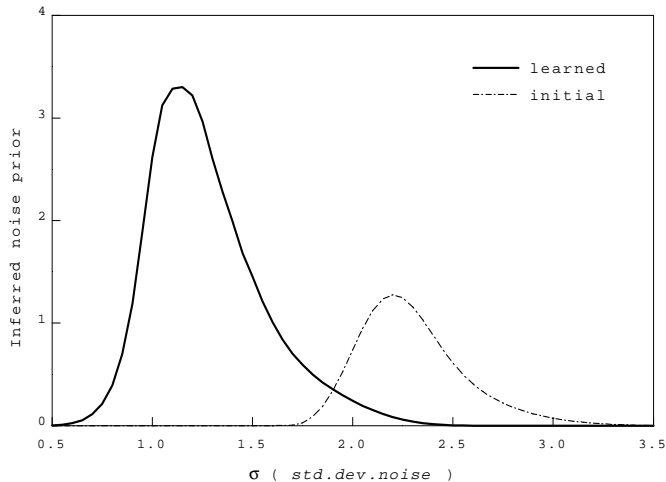


Figure 2. Error distribution learned from the artificial noisy data set.

5.1 Empirical Study 1: A Regression

The modelling potential of the proposed sequential Bayesian approach using a Student- t residual model was examined on a regression problem with outliers. A benchmark synthetic data set of 100 observations was taken which contains deliberately added noise [Neal, 2006]. The targets are zero-mean and normally distributed $\mathcal{N}(0, 0.1)$. Each target however have been contaminated with probability 0.05 by noise of standard deviation 1.0. The original targets were produced using the function:

$$y = 0.3 + 0.4x + 0.5 \sin(2.7x) + 1.1/(1 + x^2) \quad (25)$$

where x are the one-dimensional real value inputs.

Using the SVRVM, the free parameters c and d of the Student- t error distribution were predetermined using the scaled conjugate-gradients algorithm [Möller, 1993]. Starting initially from $c = 3.0$ and $d = 0.5$, it was found that the optimal values are: $c = 1.75$ ($v = 2c$), and $d = 1.24$ ($\sigma_\Gamma = \sqrt{d/c}$). The shape of the inferred distribution is shown in Figure 2, we observe a heavier than Normal tail, but also some skewness. The latter may have some implication for the robustness of our results. These optimal values for c and d were next plugged into the $\text{MLP}_{\text{Student-}t}$ online neural network training algorithm.

The regression curves are shown in Figure 3. We observe that the SVRVM consistently maps the data and seems to be the least affected by outliers. The $\text{MLP}_{\text{Kalman}}$ is worse at the beginning and at the end of the interval, where it is unable to capture the curvature in the data. The KRLS algorithm appears to have been misled by the outliers, it shows very large deviations especially in the first half of the interval.

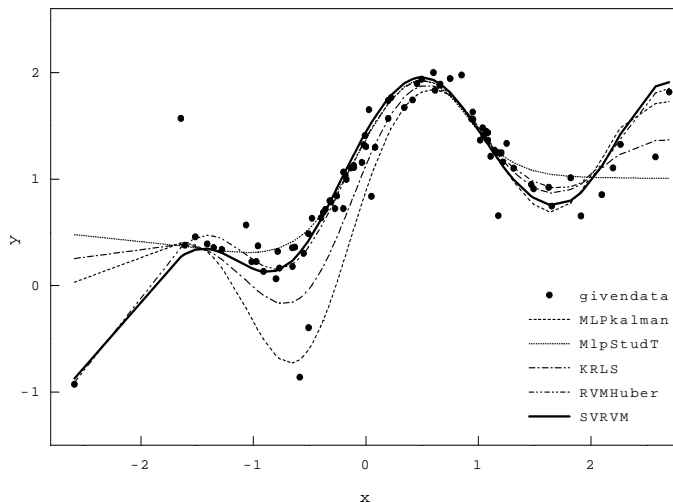


Figure 3. Regression curves obtained with different models.

	<i>Parameters</i>	<i>Testing</i>
		<i>MAPE</i>
MLP_{Kalman}	49	2.02
$MLP_{Student-t}$	49	1.58
KRLS	26	1.86
RVM_{Huber}	19	1.40
$SVRVM$	19	1.32

Table 1. Testing results (MAPE) and sizes of the models learned from the noisy regression set.

The Mean Absolute Percentage Errors (MAPE) on another testing data set of 100 different input points are given in Table 1. The errors in Table 1 confirm the observations made after looking at the regression curves in Figure 3. The MLP_{Kalman} network model has the largest error, which is not surprising as it assumes normally distributed residuals. The $MLP_{Student-t}$ is really able to handle the noise with a Student- t residual model better than a probabilistic filter like the MLP_{Kalman} , and it is slightly better than the KRLS kernel algorithm. In spite of the powerful approximation abilities of kernel models, due to modelling the data dependencies in a higher-dimensional feature space, the KRLS algorithm has not fulfilled the expectations as it could not capture the true character of the data and seems disturbed by the outliers. All in all, the SVRVM and the RVM_{Huber} were more robust to the presence of outliers and thus appear to be potentially more suitable to deal with unusual observations in the data.

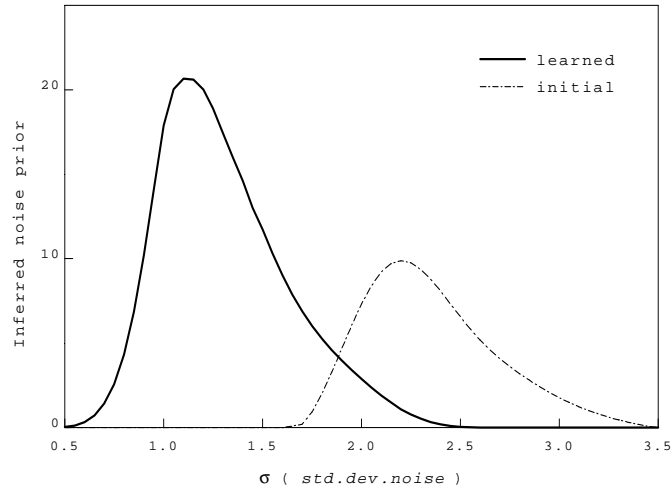


Figure 4. Error distribution learned from the one-year daily Electricity price series.

5.2 Time Series Modelling

Two studies on Nordpool average daily electricity spot prices were carried out. The prices are measured in Euro/MegaWatt hour and are available at <http://www.nordpool.no>. The first series corresponds to one year of observations and the training data is from 1/1/2004 to 31/12/2004. The second encompasses the first and includes two years of daily averages and thus the training data is from 1/1/2004 to 31/12/2005. In both cases forecasts for the month of January are produced. We note that the chosen testing period is within the Winter, when electricity demand is higher and more volatile in northern regions. Furthermore, this period starts within the Christmas holidays section reports results achieved by the sequential Bayesian SVRVM and the other algorithms on forecasting.

5.2.1 Empirical Study 2: One Year of Daily Electricity Prices

The aim of this study was to forecast the average daily prices for the day-ahead during the next month (January 2005). A seasonal ARIMA (SARIMA) model of the form $ARIMA(0, 0, 1) \times (0, 0, 2)_7$ was estimated following the standard Box-Jenkins procedure and Minitab 14. The final estimates for this model and its standard errors were: $\phi_1 = 0.7915(0.0327)$, $\Phi_1 = 0.3962(0.0521)$, $\Phi_2 = 0.2196(0.0521)$, constant = 2.29471(0.06717) and Mean = 28.6402(0.8383). As earlier explained, the nonlinear models were estimated based on the standardized data, and thus the previous two weeks (x_{t-1}, \dots, x_{t-14}) were used as inputs. All nonlinear models are of comparable size, but significantly larger than our SARIMA model with only 5 parameters.

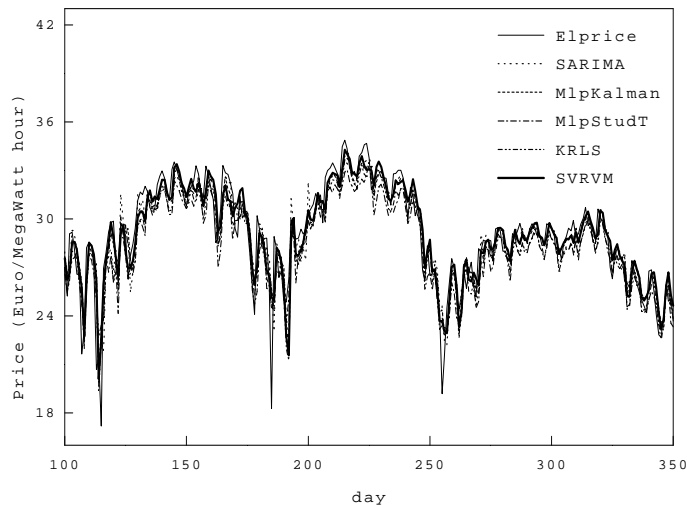


Figure 5. Approximation of the training electricity prices 2004.

	<i>Parameters</i>	<i>Training</i>	<i>Prediction</i>
		<i>15/01/04 – 31/12/04</i>	<i>1 – 31/01/05</i>
		<i>MAPE</i>	<i>MAPE</i>
MLP _{Kalman}	49	2.40	4.62
MLP _{Student-t}	49	3.61	4.21
KRLS	61	2.85	4.16
RVM _{Huber}	58	3.52	4.13
SVRVM	50	3.02	3.41
SARIMA	5	4.26	3.57

Table 2. Training performance and one-step ahead forecasts of the one-year Electricity prices.

Using SVRVM (starting from $c = 3.0$ and $d = 0.5$), there were found two distribution parameters $c = 2.73$ ($v = 2c$), and $d = 1.39$ ($\sigma_{\Gamma} = \sqrt{d/c}$). Figure 4 displays the shape of the learned error distribution for these data, which is heavy-tailed, though slightly skewed. It is thus reasonable to attempt to use the non-Gaussian distribution to estimate the error process, but we are aware that results may be affected by skewness.

The results in Table 2 show that SVRVM achieves the lowest forecasting MAPE. SVRVM and RVM_{Huber} outperform the other KRLS kernel algorithm. We are inclined to think that the MLP_{Kalman} tends to overfit the training data, and, that is why, it shows a high forecasting error. The MLP_{Student-t} model performed better than the MLP_{Kalman} filter on prediction, and it was only slightly worse than the kernel algorithms.

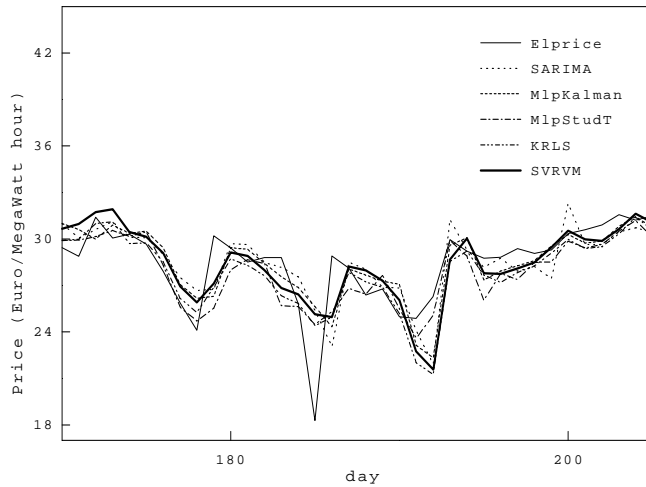


Figure 6. Segment from the approximated electricity prices 2004.

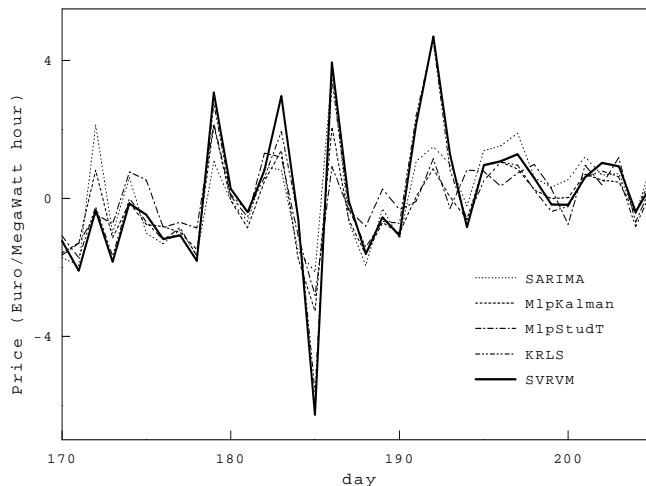


Figure 7. Residuals on the training electricity prices 2004.

We can observe that the seeming outliers (on the 115, 185 and 255 day) are discarded by all algorithms, which is an indication of their learning potential. Figure 5, however, does not help to analyze the critical spiky regions and thus we provide a close look at a particular segment, which is shown in Figure 6 and the corresponding residuals in Figure 7.

Figure 6 shows that on the 185th day the algorithms do not seem to overfit the given training data. This is confirmed by the plots in the next Figure 7, where the SVRVM and RVM_{Huber} show large absolute residuals in the neighborhood of the 185th day, more precisely, on the 179th, 186th, and 192th days. The largest training errors of SVRVM, which are shown in Figure 7, indicate that this learning algorithm achieves a smooth fit without attempting to track the noise in the data. The SVRVM has potential to avoid overfitting, which is confirmed by its one-step ahead forecasts in Figure 8.

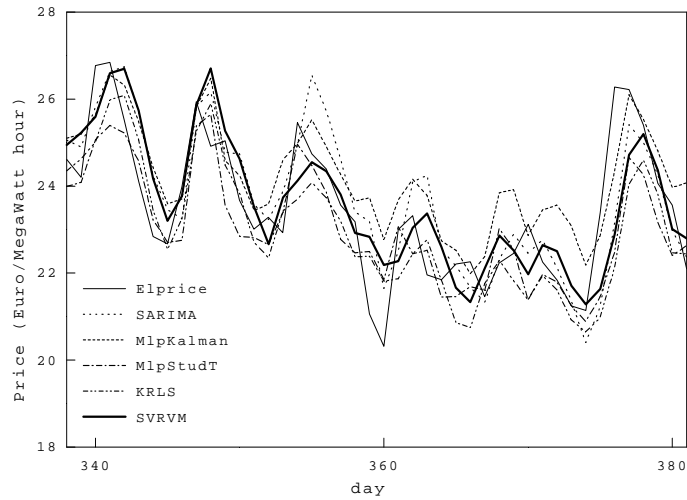


Figure 8. One-step ahead forecasts of electricity prices (January, 2005).

	<i>Parameters</i>	<i>Prediction</i>
		<i>1 – 07/01/05</i>
		<i>MAPE</i>
SVRVM	50	1.42
SARIMA	5	4.32

Table 3. Multistep forecasting- first week of January, 2005.

The next question that we addressed is whether SVRVM can outperform standard models in multistep-ahead predictions. We thus compared the performance of the SVRVM and SARIMA models in forecasting the first week of January 2005, which is summarized in Table 3. These results illustrate the potential of this new algorithm for similar practical applications.

5.2.2 Empirical Study 3: Two-years of Daily Electricity Prices

In the last two years, energy markets worldwide have become more volatile and this can be observed in many time series of electricity and gas prices. Our third empirical study, extends the time series used in the previous investigation, so as to address the more challenging task of forecasting average daily prices in January 2006.

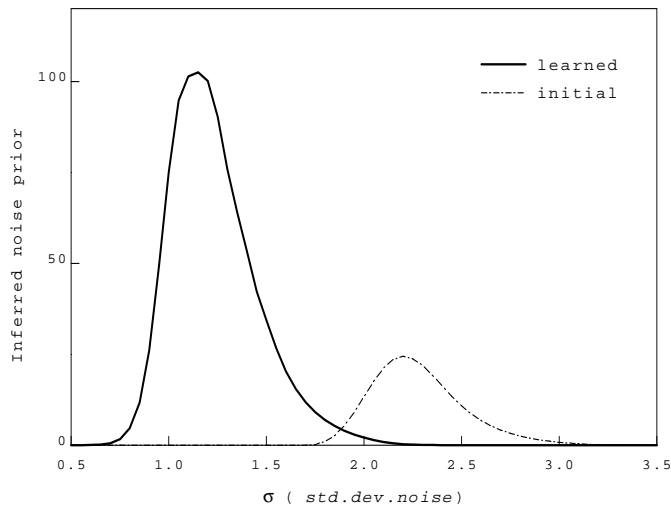


Figure 9. Error distribution learned from 2004 and 2005 daily prices.

We use the period from 1 January 2004 to 31 December 2005 to train the studied here models and to estimate a Box-Jenkins (ARIMA) model. The latter became a considerable task, because several models that fitted the data well were not invertible and thus we needed to consider many alternatives. Finally, using Eviews 4, we obtained the following parsimonious model, whose performance we report here: $0.2206x_{t-1} + 0.3258x_{t-2} + 0.06845x_{t-5} + 0.3852x_{t-7} + 0.562e_{t-1} + e_t$ and the estimated standard errors for the parameter were respectively equal to: 0.06462, 0.06049, 0.03297, 0.033412 and 0.070055.

Following the procedure that was adopted in the previous example, the optimal free parameters of the error distribution were : $c = 2.86$ ($v = 2c$), and $d = 1.54$ ($\sigma_\Gamma = \sqrt{d/c}$). Figure 9 displays the shape of the learned distribution from this series, where we observe that the initial error model (shown with dash-dotted line) has a heavy tail. After the optimization algorithm the error distribution becomes highly peaked and skewed, but kurtosis is still significant and thus the non-Gaussian distribution should lead to a better approximation of the error term than the usual normal distribution. As before, the same hyperparameters, $c = 2.86$ and $d = 1.54$, were used to run the $MLP_{Student-t}$ training algorithm.

The networks, MLP_{Kalman} and $MLP_{Student-t}$, were designed with 4 hidden units and tangential activation functions. The increase in network size (65 weights), more precisely the extension of the hidden layer by one neuron, was due to the fact that preliminary runs with this time series indicated higher nonlinearity and therefore a need for larger models. All remaining parameters in the training algorithms were kept the same.

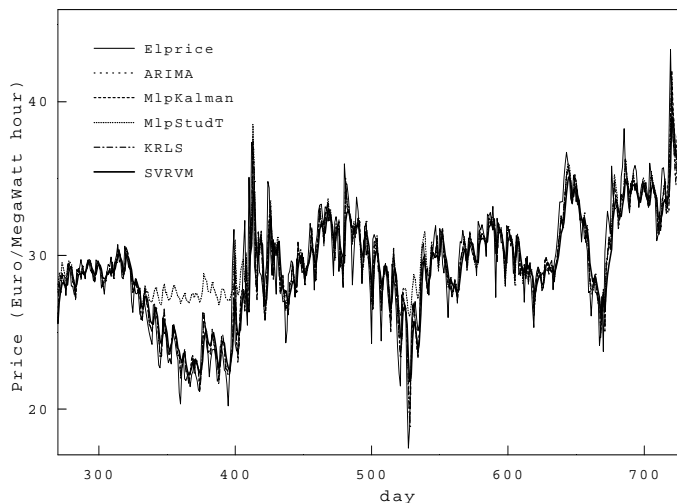


Figure 10. Approximation of the training two-years Electricity prices.

	<i>Parameters</i>	<i>Training</i>	<i>Prediction</i>
		<i>1/01/04 – 31/12/05</i>	<i>1/01/06 – 31/05/06</i>
		<i>MAPE</i>	<i>MAPE</i>
<i>MLP_{Kalman}</i>	65	5.11	9.08
<i>MLP_{Student-t}</i>	65	4.26	8.13
KRLS	56	4.61	8.45
<i>RVM_{Huber}</i>	60	3.92	7.14
<i>SVRVM</i>	59	4.25	6.78
ARIMA	5	7.39	8.74

Table 4. Training performance and one-step ahead forecasts of the two-years Electricity prices.

The width parameter for all kernel models was found to be $s^2 = 0.4$ using cross-validation. Cross-validation was performed using the first two-thirds from the training series, while the remaining one-third was used for validation. This spread selection is necessary, because the kernel width is dependent on the average distance between the training points [Schölkopf and Smola, 2001]. The pruning thresholds for the kernel models needed to be decreased to $v = 0.001$ in KRLS [Engel et al., 2004] and to $\alpha_{\max} = 1 \times 10^{-3}$ in the SVRVM and *RVM_{Huber}* algorithms. The remaining parameters were kept the same, namely: $\alpha_0 = 10^{-5}$, $\beta_0 = 1$, and $[\Sigma_0]_{ii} = 200$.

A segment from the approximated two-years Electricity price series by the studied kernel learning, neural network algorithms and ARIMA models is shown in Figure 10.

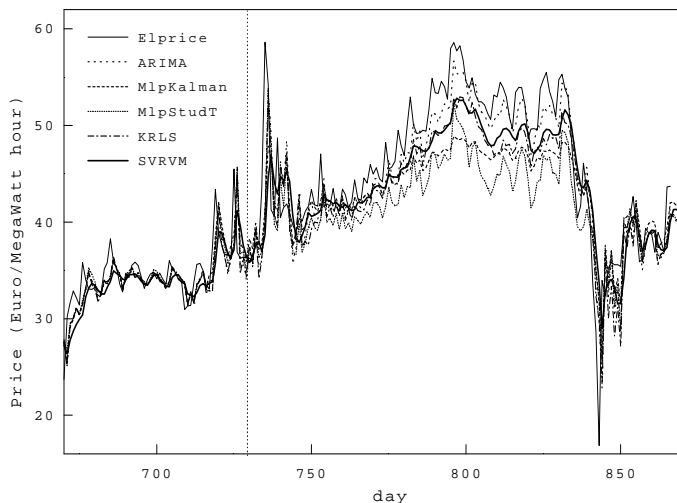


Figure 11. One-step ahead forecasts of two-years electricity prices.

The MLP_{Kalman} was outperformed by all other algorithms in training and prediction. The $MLP_{Student-t}$ algorithm performed quite well, it was better not only than MLP_{Kalman} but also better than KRLS. Without explicit noise treatment, the KRLS algorithm failed again to capture the character of the unknown function. The SVRVM and RVM_{Huber} algorithms were clearly superior. We can attribute their good results not only to their ability to handle non-Gaussian noise, because $MLP_{Student-t}$ also handles such heavy tail residuals, but also to their capacity to carry out automatic model selection which $MLP_{Student-t}$ does not do.

We note that the ARIMA model performed relatively well in comparison with the kernel and neural network models. Nevertheless, it seems that the data was more amenable to learning by kernel and nonlinear network models. Following the Box-Jenkins procedure, we attempted several models of the price and of its logarithm. Model selection became a challenging task, since several of the ARIMA models that fitted the data well lacked the required properties for good generalization. The identification and estimation steps in the Box-Jenkins procedure needed to be repeated several times, so that we had candidate models, from which we selected the model with the lowest BIC (Bayesian Information Criterion).

Figure 11 provides a closer look at the one-step ahead forecasts and we observe that SVRVM forecasts are closer to the actual values in January 2006. The model does not overfit the series, even at the spiky days (735th, 736th and 843rd). Although the remaining algorithms have similar forecasting potential on these days, they fail to reach in magnitude the future movements in the series. We also compared the multi-step ahead performance of the SVRVM with the ARIMA model. Table 5 indicates that the SVRVM can automatically produce forecasts that outperform traditional models. The curvature of the multistep ahead forecasts is shown in Figure 12.

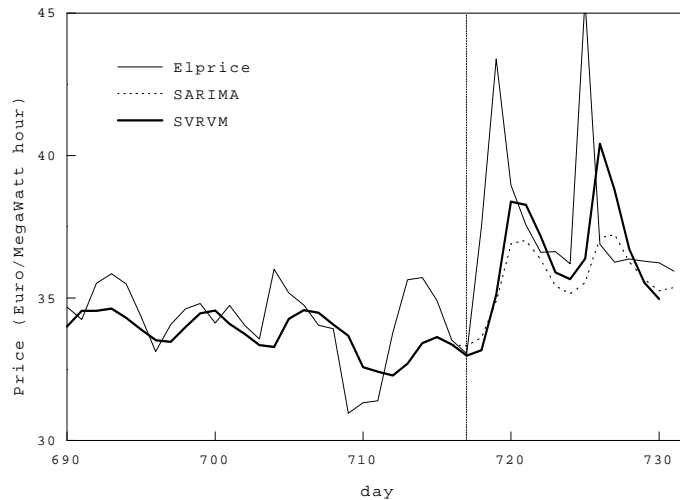


Figure 12. Multistep-step ahead (14 days ahead) forecasts of two-year elprices.

	<i>Parameters</i>	<i>Prediction</i>
		<i>1 – 14/01/06</i>
		<i>MAPE</i>
SVRVM	59	5.87
ARIMA	5	6.42

Table 5. Multistep forecasting of two-years Electricity prices.

6 Discussion and Conclusion

The Nonlinear models are powerful tools for forecasting because of their potential to capture subtle changes in time series. Our empirical studies confirm that although the architecture of MLP networks is simple to design, their training is difficult and may be inefficient because of the highly nonlinear weight dependencies. Even with Extended Kalman filters, MLPs are difficult to train recursively, because its linearization may lead to further inaccuracies [Wan and van der Merwe, 2001]. We found that sophisticated training filters, such as the Fisher scoring with Student- t noise proposed by Briegel and Tresp [1999], achieve improved performance, especially when implemented properly using EM-style hyperparameter re-estimation after each pass through the data. It should be noted that this algorithm is time consuming as it involves two matrix inversions which actually prevent its application to kernel models, like the SVRVM, as they start operating with a prohibitively large number of weights.

The main reason for the good performance of kernel models is the kernel trick. That is, although the model is linear in the weights, the dependencies between the inputs are identified in the higher-dimensional feature (Hilbert) space of their mappings, which is obtained via the nonlinear basis functions. Yet, this paper confirms that without proper Bayesian treatment, a standard kernel algorithm tends to overfit the data. Furthermore, gradually building the model by adding a new kernel at a time, with the arrival of new input point, as implemented by the KRLS algorithm, is not enough to achieve good generalization. Such greedy algorithms may easily become trapped in suboptimal solutions.

Following our empirical studies, we may conclude that the probabilistic SVRVM algorithm, which we developed in this research, has several advantages. First, the use of Bayesian regularization allows us to achieve fast results, because computationally expensive validations are not required. Second, model selection is automatically performed at the end of each cycle, i.e., by training data and discarding kernels, whose weight prior exceeds a predefined threshold. Third, the model is not simply pruned, but it is actually made smoother, and thus its generalizes better. Fourth, the approach is especially suitable for modelling time-varying systems, because it updates the parameters with the most recent information. Finally, non-Gaussian errors are applicable to any data intensive time-series modelling, for we should not forget that this distribution essentially has the same shape as the normal but with fatter tail that makes it better for dealing with outliers.

This paper contributes to the research into the development of robust sequential Bayesian inference methods for nonlinear models. The incremental nature of the SVRVM opens an avenue for future research into time series, because kernel models provide technical means for reliable modelling. Our sequential probabilistic algorithm produces as a by-product recursive residuals, which can be used for diagnosing and testing properties, e.g.: serial correlation, functional misspecification, outliers and structural changes [Harvey, 1989], [Hawkins, 1991]. The main advantage of SVRVM, in comparison to the studied neural network approaches, it is linear in the weights which guarantees convergence to the global minimum.

References

- [1] Attias, H. (2000). A Variational Bayesian Framework for Graphical Models, In: T.Leen et al. (Eds.), *Advances in Neural Inf. Proc. Systems NIPS-12*, Cambridge, MA: MIT Press, pp.209-215.
- [2] Beal, M. (2003). Variational algorithms for Approximate Bayesian Inference, PhD Thesis, Gatsby Computational Neuroscience Unit, University College London, UK.

- [3] Bishop, C.M. and Tipping, M.E. (2000). Variational Relevance Vector Machines. In: C. Boutilier and M. Goldszmidt (Eds.), *Proc. 16th Conf. on Uncertainty in Artificial Intelligence* pp.46-53. San Mateo, CA: Morgan Kaufmann.
- [4] Briegel, T. and Tresp, V. (1999). Fisher scoring and a mixture of modes approach for approximate inference and learning in nonlinear state space models, In: M.S.Kearns, S.A.Solla, and D.A.Cohn (Eds.), *Advances in Neural Inf. Proc. Systems NIPS-11*, Cambridge, MA: MIT Press, pp.403-409.
- [5] Cawley, G.C. and Talbot, N.L.C. (2005). Constructing Bayesian formulations of sparse kernel learning methods, *Neural Networks*, vol.18, N:5-6, pp.674-683.
- [6] De Gooijer, J.G. and Hyndman, R.J.(2006). 25 years of time series forecasting, *Int. Journal of Forecasting*, vol.22, N:3, pp.443-474.
- [7] de Freitas, J.F.G., Niranjan, M. and Gee, A.H. (2000). Hierarchical Bayesian-Kalman models for regularisation and ARD in sequential learning, *Neural Computation*, vol.12, N:4, pp.933-953.
- [8] de Menezes, L. and Nikolaev, N. (2006). Forecasting with genetically programmed polynomial neural networks, *Int. Journal of Forecasting*, vol.22, N:2, pp.249-265.
- [9] Engel, Y., Mannor, S. and Meir, R. (2004). The Kernel Recursive Least Squares Algorithm, *IEEE Trans. on Signal Processing*, vol.52, N:8, pp.2275-2285.
- [10] Faul, A. and Tipping, M.E. (2001). A variational approach to robust regression, In: G.Dorffner, H.Bischof, and K.Hornik (Eds.), *Proc. Int. Conf. Artificial Neural Networks ICANN'01*, Berlin: Springer, pp. 95-102.
- [11] Ghahramani, Z. and Beal, M. (2001). Propagation Algorithms for Variational Bayesian Learning, In: T.K.Leen, T.Dietterich, V.Tresp (Eds.), *Advances in Neural Inf. Proc. Systems NIPS-13*, Cambridge, MA: MIT Press, pp.507-513.
- [12] Harvey, A.C. (1989). *Forecasting, Structural Time Series Models and the Kalman Filter*, Cambridge University Press, Cambridge.
- [13] Hawkins, D.M. (1991). Diagnostics for use with regression recursive residuals. *Technometrics*, vol.33, pp.221-234.
- [14] Haykin, S. (Ed.) (2001). *Kalman Filtering and Neural Networks*, John Wiley and Sons, New York, NY.
- [15] Huber, P.J. (1981). *Robust Statistics*, John Wiley and Sons, New York, NY.

- [16] Jordan, M.I., Ghahramani, Z., Jaakkola, T. and Saul, L.K. (1999). An Introduction to Variational Methods for Graphical Models, *Machine Learning*, vol.37, N:2, pp.183-233.
- [17] Ljung, L. and Söderström, T. (1983). *Theory and Practice of Recursive Identification*, Cambridge, MA: MIT Press.
- [18] MacKay, D.J.C. (1992). Bayesian Interpolation. *Neural Computation*, vol.4, N:3, pp.415-447.
- [19] MacKay, D.J.C. (1995). Developments in Probabilistic Modelling with Neural Networks- Ensemble Learning, In: B.Kappen and S.Gielen (Eds.), *Proc. 3rd Annual Symposium on Neural Networks*, Berlin: Springer-Verlag, pp.191-198.
- [20] Mardia, K.V., Kent, J.T. and Bibby, J.M. (1979). *Multivariate Analysis*, London: Academic Press.
- [21] Möller, A.F. (1993). A Scaled Conjugate Gradient Algorithm for Fast Supervised Learning, *Neural Networks*, vol.6, N:4, pp.525-533.
- [22] Neal, R. (2006). *Software for Flexible Bayesian Modelling*,
<http://www.cs.toronto.edu/~radford/fbm.software.html>, Toronto, CA.
- [23] Nikolaev, N. and Iba, H. (2006). *Adaptive Learning of Polynomial Networks: Genetic Programming, Backpropagation and Bayesian Methods*, Springer, New York.
- [24] Penny, W.D. and Roberts, S.J. (2000). Variational Bayes for Non-Gaussian Autoregressive Models, In: B.Widrow et al. (Eds.) *Proc. IEEE Workshop on Neural Networks for Signal Processing*, IEEE Press, New York, pp.135-144.
- [25] Rasmussen, C.E. and Williams, C.K.I. (2006). *Gaussian Processes for Machine Learning*, Cambridge, MA: MIT Press.
- [26] Roberts, S.J. and Penny, W.D. (2002). Variational Bayes for Generalised Autoregressive Models, *IEEE Trans. on Signal Processing*, vol.50, N:9, pp.2245-2257.
- [27] Sato, M. and Oba, S. (2002). Incremental Sparse Kernel Machine, In: J.R.Dorronsoro (Ed.), *Proc. Int. Conf. Artificial Neural Networks ICANN-2002*, LNCS 2415, Berlin: Springer-Verlag, pp.700-706.
- [28] Schölkopf, B. and Smola, A.J. (2002). *Learning with Kernels*, Cambridge, MA: MIT Press.
- [29] Suykens, J.A.K., Van Gestel, T., De Brabanter, J., De Moor, B. and Vandewalle, J. (2002) *Least Squares Support Vector Machines*, World Scientific Publ., Singapore.

- [30] Takens, F. (1981). Detecting Strange Attractors in Turbulence. In: D.A.Rand and L.-S.Young (Eds), *Dynamical Systems and Turbulence*, Lecture Notes in Mathematics, 898, Springer-Verlag, Berlin, pp.366-381.
- [31] Tipping, M.E. (2001). Sparse Bayesian learning and the Relevance Vector Machine. *Journal of Machine Learning Research*, vol.1, pp.211-244.
- [32] Tipping, M.E. and Lawrence, N.D. (2005). Variational inference for Student-t models: Robust Bayesian interpolation and generalised component analysis, *Neurocomputing*, vol.69, pp.123-141.
- [33] Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. Berlin: Springer-Verlag.
- [34] Wan, E.A. and van der Merwe, R. (2001). The Unscented Kalman Filter, In: S.Haykin (Ed.), *Kalman Filtering and Neural Networks*, chapter 7, Wiley Publ., New York, pp.221– 280.

Appendix A: Derivation of the Recursive Formulae

The derivations below use the following notation: the weight vector modified up to moment (arrival of data point \mathbf{x}_t) t is \mathbf{w}_t , the target vector is $\mathbf{y}_t = [y_1, y_2, \dots, y_t]^T$, the kernel vector from the t -th data point is $\mathbf{k}_t = \mathbf{k}(\mathbf{x}_t) = [K(\mathbf{x}_t, \mathbf{x}_1), K(\mathbf{x}_t, \mathbf{x}_2), \dots, K(\mathbf{x}_t, \mathbf{x}_M)]^T$, and the design matrix includes the relevant M basis functions evaluated with all arrived training points \mathbf{x}_i , that is $\mathbf{K}_t = \{K(\mathbf{x}_i, \mathbf{x}_j)\}$ where: $1 \leq i \leq t, 1 \leq j \leq M$.

The proposed approach implements Bayesian regularization when adding a new training input, which affects both: the weight estimation and the covariance matrix estimation. The recursive update equation for the mean weight vector is obtained as a modification of the batch least squares formula as follows:

$$\begin{aligned}
 \boldsymbol{\mu}_t &= \boldsymbol{\Sigma}_t \mathbf{K}_t^T \mathbf{B}_t \mathbf{y}_t \\
 &= \boldsymbol{\Sigma}_t (\mathbf{K}_{t-1}^T \mathbf{B}_{t-1} \mathbf{y}_{t-1} + \mathbf{k}_t \beta_t y_t) \\
 &= \boldsymbol{\Sigma}_t (\boldsymbol{\Sigma}_{t-1}^{-1} \boldsymbol{\mu}_{t-1} + \mathbf{k}_t \beta_t y_t) \\
 &= \boldsymbol{\Sigma}_t ((\boldsymbol{\Sigma}_t^{-1} - \mathbf{k}_t \beta_t \mathbf{k}_t^T - \alpha_i \mathbf{R}_t) \boldsymbol{\mu}_{t-1} + \mathbf{k}_t \beta_t y_t) \\
 &= \boldsymbol{\Sigma}_t \boldsymbol{\Sigma}_t^{-1} \boldsymbol{\mu}_{t-1} - \boldsymbol{\Sigma}_t \mathbf{k}_t \beta_t \mathbf{k}_t^T \boldsymbol{\mu}_{t-1} - \boldsymbol{\Sigma}_t \alpha_i \mathbf{R}_t \boldsymbol{\mu}_{t-1} + \boldsymbol{\Sigma}_t \mathbf{k}_t \beta_t y_t \\
 &= \boldsymbol{\mu}_{t-1} + \boldsymbol{\Sigma}_t \mathbf{k}_t \beta_t (y_t - \mathbf{k}_t^T \boldsymbol{\mu}_{t-1}) - \alpha_i \boldsymbol{\Sigma}_t \mathbf{R}_t \boldsymbol{\mu}_{t-1} \\
 &= \boldsymbol{\mu}_{t-1} + \boldsymbol{\Sigma}_t \mathbf{k}_t \beta_t e_t - \alpha_i \boldsymbol{\Sigma}_t \mathbf{R}_t \boldsymbol{\mu}_{t-1}
 \end{aligned} \tag{26}$$

where the second term is the incremental update, and the third is the regularizer. According to the specific definition of the matrix \mathbf{R}_t the amount of regularization is partially distributed during training until reaching the boundary condition $M * (N \text{ div } M)$.

When adding a new input to the covariance matrix Bayesian regularization is applied as follows:

$$\boldsymbol{\Sigma}_t^{-1} = \boldsymbol{\Sigma}_{t-1}^{-1} + \mathbf{k}_t \beta_t \mathbf{k}_t^T + \alpha_i \mathbf{R}_t \tag{28}$$

which using $[\mathbf{R}_t]_{ii} = 1.0/z_i, i = t \text{ mod } M, z_i = (N \text{ div } M)$ if $t < M * (N \text{ div } M)$ or a very large number $z_i = 1.0e10$ otherwise, after passing through all training data becomes the same as: $\boldsymbol{\Sigma}^{-1} = \mathbf{K}^T \mathbf{B} \mathbf{K}^T + \mathbf{A}$.

The key moment in the derivation is to transform this expression in order to apply the matrix inversion lemma as follows:

$$\boldsymbol{\Sigma}_t^{-1} = \boldsymbol{\Sigma}_{t-1}^{-1} + \beta_t \mathbf{k}_t^* \boldsymbol{\Lambda}_t^{-1} \mathbf{k}_t^{*T} \tag{29}$$

where $\boldsymbol{\Lambda}_t^{-1} = \begin{bmatrix} 1 & 0 \\ 0 & \alpha_i [\mathbf{R}_t]_{ii} \end{bmatrix}$ is the regularization matrix, and \mathbf{k}_t^* is the augmented kernel matrix.

The kernel matrix \mathbf{k}_t^* consists of the kernel column vector \mathbf{k}_t^T augmented by a column with zero elements, except for the element at position $i = t \bmod M$ as follows [Ljung and Söderström, 1983]:

$$\mathbf{k}_t^* = \begin{bmatrix} \mathbf{k}_t^T \\ 0 \dots 0 \ 1_i \ \dots 0 \end{bmatrix}^T \quad (30)$$

Applying the matrix inversion lemma: $(A + BCD)^{-1} = A^{-1} - A^{-1}B [DA^{-1}B + C^{-1}]^{-1} DA^{-1}$ [Mardia et al., 1979] to the above expression yields:

$$(\boldsymbol{\Sigma}_{t-1}^{-1} + \beta_t \mathbf{k}_t^* \boldsymbol{\Lambda}_t^{-1} \mathbf{k}_t^{*T})^{-1} = \boldsymbol{\Sigma}_{t-1} - \boldsymbol{\Sigma}_{t-1} \beta_t \mathbf{k}_t^* (\mathbf{k}_t^{*T} \boldsymbol{\Sigma}_{t-1} \beta_t \mathbf{k}_t^* + \boldsymbol{\Lambda}_t)^{-1} \mathbf{k}_t^{*T} \boldsymbol{\Sigma}_{t-1} \quad (31)$$

which is the recursive covariance formula (7): $\boldsymbol{\Sigma}_t = \boldsymbol{\Sigma}_{t-1} - \beta_t \boldsymbol{\Sigma}_{t-1} \mathbf{k}_t^* (\beta_t \mathbf{k}_t^{*T} \boldsymbol{\Sigma}_{t-1} \mathbf{k}_t^* + \boldsymbol{\Lambda}_t)^{-1} \mathbf{k}_t^{*T} \boldsymbol{\Sigma}_{t-1}$. It should be noted that the denominator of the second term $\mathbf{k}_t^{*T} \boldsymbol{\Sigma}_{t-1} \beta_t \mathbf{k}_t^* + \boldsymbol{\Lambda}_t$ is a two dimensional square matrix which is easily inverted [Mardia et al., 1979].

Appendix B: Optimisation of the Factors

The distribution of the weight prior hyperparameters $Q_m(\alpha_m)$ is identified with the following derivations:

$$\log Q_m(\alpha_m) = \log p(\alpha_m | a, b) + \langle \log p(w_m | \alpha_m) \rangle_{\mathbf{w}} + const. \quad (32)$$

$$= (a - 1) \log \alpha_m - b \alpha_m + 0.5 \log \alpha_m - 0.5 \alpha_m \langle w_m^2 \rangle + const. \quad (33)$$

$$= [(a + 0.5) - 1] \log \alpha_m - (b + 0.5 \langle w_m^2 \rangle) \alpha_m + const. \quad (34)$$

which indicates a Gamma distribution with parameters $\tilde{a} = a + 0.5$ and $\tilde{b}_m = b + 0.5 \langle w_m^2 \rangle$.

The form of the noise factor distribution $Q_t(\beta_t)$ is derived as follows:

$$\log Q_t(\beta_t) = \log p(\beta_t | c, d) + \langle \log p(y_t | \mathbf{w}, \beta_t) \rangle_{\mathbf{w}} + const. \quad (35)$$

$$= (c - 1) \log \beta_t - d \beta_t + 0.5 \log \beta_t - 0.5 \beta_t \langle y_t^2 - 2y_t \mathbf{k}_t \langle \mathbf{w}_t \rangle + \mathbf{k}_t^T \langle \mathbf{w}_t \mathbf{w}_t^T \rangle \mathbf{k}_t \rangle + const. \quad (36)$$

$$= [(c + 0.5) - 1] \log \beta_t - (d + 0.5 \langle y_t^2 - 2y_t \mathbf{k}_t \langle \mathbf{w}_t \rangle + \mathbf{k}_t^T \langle \mathbf{w}_t \mathbf{w}_t^T \rangle \mathbf{k}_t \rangle) \beta_t + const. \quad (37)$$

which is a Gamma distribution with parameters $\tilde{c} = c + 0.5$ and $\tilde{d}_t = d + 0.5 \langle y_t^2 - 2y_t \mathbf{k}_t \langle \mathbf{w}_t \rangle + \mathbf{k}_t^T \langle \mathbf{w}_t \mathbf{w}_t^T \rangle \mathbf{k}_t \rangle$.

The weight factor distribution $Q_{\mathbf{w}}(\mathbf{w})$ is obtained by taking the following two expectations:

$$\log Q_{\mathbf{w}}(\mathbf{w}) = \langle \log p(\mathbf{y} | \mathbf{w}, \boldsymbol{\beta}) \rangle_{\boldsymbol{\beta}} + \langle \log p(\mathbf{w} | \boldsymbol{\alpha}) \rangle_{\boldsymbol{\alpha}} + const. \quad (38)$$

$$= -0.5 \sum_{t=1}^T \langle \beta_t \rangle (y_t - \mathbf{w}^T \mathbf{k}(\mathbf{x}_t))^2 - 0.5 \mathbf{w}^T \mathbf{A} \mathbf{w} + const. \quad (39)$$

$$= -0.5 \mathbf{w}^T (\mathbf{K}^T \mathbf{B} \mathbf{K} + \mathbf{A}) \mathbf{w} + \mathbf{w}^T \mathbf{K}^T \mathbf{B} \mathbf{y} + const. \quad (40)$$

which is identified as Gaussian $Q_{\mathbf{w}}(\mathbf{w}) = \mathcal{N}(\mathbf{w} | \boldsymbol{\mu}, \boldsymbol{\Sigma})$.