

# Smooth Bayesian Kernel Machines

Rutger W. ter Borg<sup>1</sup> and Léon J.M. Rothkrantz<sup>2</sup>

<sup>1</sup> Nuon NV, Applied Research & Technology  
Spaklerweg 20, 1096 BA Amsterdam, the Netherlands  
`rutger@terborg.net`

<sup>2</sup> Delft University of Technology  
Mekelweg 4, 2628 CD Delft, the Netherlands  
`l.j.m.rothkrantz@ewi.tudelft.nl`

**Abstract.** In this paper, we consider the possibility of obtaining a kernel machine that is sparse in feature space and smooth in output space. Smooth in output space implies that the underlying function is supposed to have continuous derivatives up to some order. Smoothness is achieved by applying a roughness penalty, a concept from the area of functional data analysis. Sparseness is taken care of by automatic relevance determination. Both are combined in a Bayesian model, which has been implemented and tested. Test results are presented in the paper.

## 1 Introduction

In tasks such as time series modelling and system control engineering, representing observed data per se does not draw the primary interest, but rather how and how rapidly a system responds to certain events, i.e. the behaviour of derivatives of functions describing such a system.

Kernel machines have become a popular tool to model measured data with. Emerged by the combination of several disciplines, they have in common that they combine the kernel trick [1] and the principle of parsimony [2,3]. The latter is brought forth by obtaining a sparse model that utilises a small subset of the data to represent a function with. However, as yet no special attention has been paid to the smoothness of that function<sup>3</sup>, i.e., it should have continuous derivatives up to some order. In case of regression, we want the resulting function to be smooth, and in case of a classification problem, the resulting decision boundary should be smooth.

The remainder of this paper is organised as follows. In section 2, we introduce derivative kernels, and kernel roughness penalties. We note that through penalised regularisation, roughness penalties do not lead to sparseness. In section 3, we introduce a novel Bayesian prior, its related model, and update equations. Section 4 shows experimental results, and section 5 concludes the paper.

---

<sup>3</sup> The smooth support vector machine [4] entails a reformulation of the quadratic program of the support vector machine.

## 2 Smooth Functional Representations

In supervised learning, we consider a data set  $\mathcal{D} = (\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)$  containing  $N$  input-output pairs  $(\mathbf{x}_i, y_i) \in \mathcal{X} \times \mathcal{Y}$ , with  $\mathcal{X}$  typically containing multidimensional vectors in  $\mathbb{R}^M$ , and  $\mathcal{Y}$  representing either classes in case of classification, or scalars in  $\mathbb{R}$  in case of regression. Wahba [5] shows that we can represent our data  $\mathcal{D}$  using a linear model of the form

$$y(\mathbf{x}) = w_0 + \sum_{i=1}^N w_i k(\mathbf{x}, \mathbf{x}_i) \quad (1)$$

with bias  $w_0$ , and parameters  $w_1, \dots, w_N$ , and a kernel function  $k$  which, under certain conditions, defines an inner product in feature space,  $k(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)^\top \Phi(\mathbf{x}_j)$ .

In the field of functional data analysis, smoothness is favoured explicitly by applying a roughness penalty [6], a penalty on the degree of curvature of one or more derivatives of  $y(\mathbf{x})$ . Fortunately, as  $y(\mathbf{x})$  in (1) is linear, we can get to its derivatives by establishing the derivatives of the kernel functions  $k(\mathbf{x}, \mathbf{x}_i)$ .

### 2.1 Derivative Kernels

For the  $n$ -th power of a vector  $\mathbf{x}$ , we define  $\mathbf{x}^n = (\mathbf{x}^\top \mathbf{x})^{n/2}$  for  $n$  even, and  $\mathbf{x}^n = \mathbf{x}(\mathbf{x}^\top \mathbf{x})^{(n-1)/2}$  for  $n$  odd. Using this definition, the dimension of the  $n$ -th order derivative of a kernel  $D^n k(\mathbf{x}, \mathbf{x}_i) = \partial^n k(\mathbf{x}, \mathbf{x}_i) / \partial \mathbf{x}^n$  is either 1 for  $n$  is even, or  $M$  for  $n$  being an odd number, where  $M$  is the dimension of underlying vectors  $\mathbf{x}$  and  $\mathbf{x}_i$ . Because a kernel is a mapping  $\mathbb{R}^M \times \mathbb{R}^M \rightarrow \mathbb{R}$ , derivative kernels exist for  $n$  even or for  $M = 1$  (or both). We discuss derivatives of the commonly used Gaussian kernel and derivatives of the polynomial kernel.

Derivatives of the Gaussian kernel  $k(\mathbf{x}, \mathbf{x}_i) = \exp(-\frac{1}{2}\sigma^{-2} \|\mathbf{x} - \mathbf{x}_i\|_2^2)$  are identified by Rodrigues' formula for Hermite polynomials

$$H_n(u) = (-1)^n \exp(u^2) D^n \exp(-u^2). \quad (2)$$

By substituting  $(\sqrt{2}\sigma)^{-1}(\mathbf{x} - \mathbf{x}_i)$  for  $u$  in (2), and keeping track of additional terms of  $(\sqrt{2}\sigma)^{-1}$ , we arrive at the convenient compact form of the derivatives of Gaussian kernels

$$D^n k(\mathbf{x}, \mathbf{x}_i) = (-\sqrt{2}\sigma)^{-n} H_n((\sqrt{2}\sigma)^{-1}(\mathbf{x} - \mathbf{x}_i)) \exp(-\frac{1}{2}\sigma^{-2} \|\mathbf{x} - \mathbf{x}_i\|_2^2).$$

Derivatives of the polynomial kernel  $k(\mathbf{x}, \mathbf{x}_i) = (\gamma \mathbf{x}^\top \mathbf{x}_i + \lambda)^d$  are found to be

$$D^n k(\mathbf{x}, \mathbf{x}_i) = \frac{d!}{(d-n)!} \gamma^n \mathbf{x}_i^n (\gamma \mathbf{x}^\top \mathbf{x}_i + \lambda)^{d-n}$$

which is valid in this instantiation as long as  $d \geq n$ .

## 2.2 Kernel Roughness Penalties

We use a kernel matrix  $\mathbf{K}$  with entries  $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$  for inputs  $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N \in \mathcal{X}$ , and a design matrix  $\mathbf{H} = [\mathbf{1} \ \mathbf{K}]$  that consists of the combination of  $[1, 1, \dots, 1]^T$  and a kernel matrix. For example, in a regularisation setting, applying a roughness penalty can be accomplished by penalising the summed curvature of the second order derivatives

$$\min \|\mathbf{y} - \mathbf{H}\mathbf{w}\|_2^2 + \lambda \|D^2\mathbf{H}\mathbf{w}\|_2^2 \quad (3)$$

with  $D^2\mathbf{H} = [\mathbf{0} \ D^2\mathbf{K}]$  being a design matrix of a second order derivative kernel. For  $\lambda = 0$ , this imposes an ordinary least squares, and for  $\lambda \rightarrow \infty$ , the result will approximate a straight line. Besides a penalty on the second order derivative as shown in (3), more generally, a linear differential operator

$$Ly(\mathbf{x}) = \sum_n c_n D^n y(\mathbf{x}) \quad (4)$$

can be defined [7], and used to form a weighted sum of penalties on derivatives of  $y(\mathbf{x})$ . If applied in penalised regularisation,  $L$  takes the shape

$$\min \|\mathbf{y} - \mathbf{H}\mathbf{w}\|_2^2 + \lambda \|\mathbf{L}\mathbf{w}\|_2^2 \quad (5)$$

which is solved by  $\mathbf{w} = (\mathbf{H}^T\mathbf{H} + \lambda\mathbf{L}^T\mathbf{L})^{-1}\mathbf{y}^T\mathbf{H}$ . Despite reasonable conditions of matrices  $\mathbf{H}^T\mathbf{H}$  and  $\mathbf{L}^T\mathbf{L}$ , in practice, this system of equations is already singular in case of duplicate inputs. In the field of functional data analysis, this is addressed by pre-processing, such as starting with the dominant frequencies [8]. Also, the selection of parameter  $\lambda$  is done manually by methods such as cross-validation.

Although the resulting functions are smooth, regularisation as in (5) does not promote sparseness. We would like to have a representation that is both smooth and sparse, and in addition, an automatic choice of parameter  $\lambda$ .

## 3 Smooth Relevance Vector Machine

We combine the idea of regularisation by a roughness penalty with a Bayesian sparseness inducing prior. We presume that the outputs are corrupted by Gaussian noise  $p(\mathbf{y}|\mathbf{w}, \sigma^2) = \mathcal{N}(\mathbf{y}|\mathbf{H}\mathbf{w}, \sigma^2\mathbf{I})$ . To promote both smoothness and sparseness, we propose a zero-mean Gaussian prior over the weights, and a covariance matrix consisting of two terms

$$p(\mathbf{w}|\boldsymbol{\alpha}, \lambda) = \mathcal{N}(\mathbf{w}|\mathbf{0}, (\lambda\mathbf{L}^T\mathbf{L} + \mathbf{A})^{-1}). \quad (6)$$

The first term  $\lambda\mathbf{L}^T\mathbf{L}$  is the smoothness promoting part, formed by a roughness penalty matrix defined by a linear differential operator  $L$  as in (4). The second term is a diagonal matrix  $\mathbf{A} = \text{diag}(\alpha_0, \dots, \alpha_N)$  containing one hyperparameter  $\alpha_i$  per weight  $w_i$ , as in automatic relevance determination (ARD) [9], as applied in the relevance vector machine (RVM) [10].

By applying Bayes' rule we obtain the posterior distributions

$$\begin{aligned} p(\mathbf{w}|\mathbf{y}, \boldsymbol{\alpha}, \lambda, \sigma^2) &= \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}), \\ p(\mathbf{y}|\boldsymbol{\alpha}, \lambda, \sigma^2) &= \mathcal{N}(\mathbf{y}|\mathbf{0}, \sigma^2\mathbf{I} + \mathbf{H}\mathbf{S}\mathbf{H}^T), \end{aligned}$$

with  $\boldsymbol{\Sigma} = (\sigma^{-2}\mathbf{H}^T\mathbf{H} + \mathbf{S}^{-1})^{-1}$ ,  $\mathbf{S} = (\lambda\mathbf{L}^T\mathbf{L} + \mathbf{A})^{-1}$  and  $\boldsymbol{\mu} = \sigma^{-2}\boldsymbol{\Sigma}\mathbf{H}^T\mathbf{y}$ .

### 3.1 Update Rules

To obtain (locally) optimal values for  $\boldsymbol{\alpha}$ ,  $\lambda$  and  $\sigma^2$ , we will follow a type-II maximum likelihood approach similar to that of the RVM [10]. The update equations for the automatic relevance determination hyper-parameters  $\alpha_i$  are given by

$$\alpha_i^{t+1} = \frac{\alpha_i^t(S_{ii} - \Sigma_{ii})}{\mu_i^2}. \quad (7)$$

Note that in the case of  $\lambda = 0$ , then  $S_{ii} = (\alpha_i^{-1})^t$ , making (7) identical to the update equation of the original RVM. The roughness-penalty parameter  $\lambda$  is updated by

$$\lambda^{t+1} = \frac{(\text{Tr}(\mathbf{S}\lambda^t\mathbf{L}^T\mathbf{L}) - \text{Tr}(\boldsymbol{\Sigma}\lambda^t\mathbf{L}^T\mathbf{L}))}{\boldsymbol{\mu}^T\mathbf{L}^T\mathbf{L}\boldsymbol{\mu}}, \quad (8)$$

and the variance estimate by

$$(\sigma^2)^{t+1} = \frac{\|\mathbf{y} - \mathbf{H}\boldsymbol{\mu}\|^2}{N - \text{Tr}(\boldsymbol{\Sigma}(\sigma^{-2})^t\mathbf{H}^T\mathbf{H})}. \quad (9)$$

In practice, we obtain traces of  $\mathbf{S}\mathbf{A}$  and  $\boldsymbol{\Sigma}\mathbf{A}$  by evaluating (7). We fully compute  $\text{Tr}(\boldsymbol{\Sigma}\lambda^t\mathbf{L}^T\mathbf{L})$ , and obtain the other trace in (8) by  $\text{Tr}(\mathbf{S}\lambda^t\mathbf{L}^T\mathbf{L}) = N - \text{Tr}(\mathbf{S}\mathbf{A})$ , and that of (9) by using  $\text{Tr}(\boldsymbol{\Sigma}(\sigma^{-2})^t\mathbf{H}^T\mathbf{H}) = N - \text{Tr}(\boldsymbol{\Sigma}\mathbf{A}) - \text{Tr}(\boldsymbol{\Sigma}\lambda^t\mathbf{L}^T\mathbf{L})$ .

## 4 Experimental Results

To measure the effectiveness of the proposed Bayesian model, we have performed a series of experiments. Because we need availability of the derivatives of the underlying function in order to be able to measure and compare performance, we have chosen the well-known sinc benchmark.

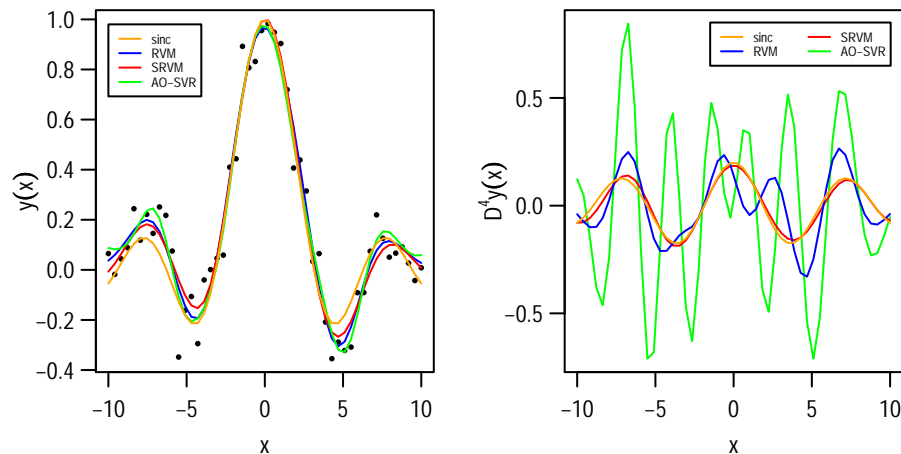
The data for one experiment consist of 50 points, with  $x_i$  uniformly distributed over  $[-10, 10]$ , and  $y_i \sim \mathcal{N}(\text{sinc}(x_i), 0.1)$ . We trained a series of kernel machines using these data with our implementation<sup>4</sup>. For our method, we have chosen to penalise the 10th order derivative only, which should effectuate smoothness up to the 8th order [8]. A Gaussian kernel is used by all methods, parametrised by  $\sigma = 1.6$ . We repeated this experiment 1000 times. Table 4 shows the mean and standard deviation of the results, subsequently of the number of basis vectors, and of the root-mean-square of errors of zeroth, second and fourth order derivatives.

<sup>4</sup> The Kernel-Machine Library is available at <http://www.terborg.net/research/kml/>.

**Table 1.** Comparative performance.

Method	BVs	$D^0$	$D^2$	$D^4$
RVM (old) [10]	$6.5 \pm 1.0$	$0.047 \pm 0.010$	$0.054 \pm 0.014$	$0.106 \pm 0.030$
RVM (new) [11]	$6.0 \pm 1.1$	$0.049 \pm 0.010$	$0.056 \pm 0.013$	$0.108 \pm 0.026$
Figueiredo [12]	$6.3 \pm 1.3$	$0.053 \pm 0.012$	$0.067 \pm 0.022$	$0.136 \pm 0.051$
KRLS [13]	$17.0 \pm 0.0$	$0.054 \pm 0.012$	$0.077 \pm 0.017$	$0.242 \pm 0.066$
SOG-SVR [14]	$17.0 \pm 0.0$	$0.062 \pm 0.010$	$0.050 \pm 0.008$	$0.062 \pm 0.012$
AO-SVR [15]	$20.7 \pm 3.2$	$0.057 \pm 0.011$	$0.112 \pm 0.029$	$0.374 \pm 0.115$
SRVM [this paper]	$16.3 \pm 3.4$	$0.039 \pm 0.009$	$0.030 \pm 0.009$	$0.033 \pm 0.013$

As shown in table 4, smoothness requires notably more basis vectors (BVs) than common sparse Bayesian models. On the other hand, the basis vectors seem to be spent wisely, because the error on all derivative orders is consistently lower than that of the compared methods. Figure 4 illustrates a function that is smooth in output space. Although at first sight the fit of the function itself is not dramatically different from others (left), the gain in quality of fit is clearly visible at a higher order derivative (right).



**Fig. 1.** Input data to a single experiment, the underlying  $\text{sinc}(x)$  function, and the resulting RVM, SRVM, and AO-SVR (left). The 4th order derivative of  $\text{sinc}(x)$ , RVM, SRVM, and AO-SVR (right).

## 5 Conclusions

In this paper we introduced a concept from functional data analysis to promote smoothness in output space. We successfully established derivative kernels, kernel roughness penalties, and a Bayesian model to combine kernel rough-

ness penalties with automatic relevance determination. Experiments elicited that smoothness in output space increased the quality of fit on all measured derivative orders. The required number of basis vectors is not the lowest, but the quality of the approximation to the underlying function is empirically better.

Further studies should include the handling of an automatic selection of several  $c_n$  in (4), i.e., a penalty on more than one derivative order. An interesting open theoretical issue is the maximum derivative order on which a signal may be expected, given a data set.

## References

1. Aizerman, M., Braverman, E., Rozonoër, L.: Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control* **25** (1964) 821–837
2. Schölkopf, B., Smola, A.: *Learning with Kernels*. Adaptive Computation and Machine Learning. The MIT Press, Cambridge, Massachusetts, USA (2002)
3. Vapnik, V.: *The Nature of Statistical Learning Theory*. Statistics for Engineering and Information Science. Springer-Verlag, New York (1995)
4. Lee, Y., Mangasarian, O.: SSVM: A smooth support vector machine for classification. *Computational Optimization and Applications* **20**(1) (2001) 5–22
5. Wahba, G.: Spline models for observational data. *Journal of the Royal Statistical Society. Series B* **59** (1990) 133–150
6. Green, P., Silverman, B.: *Nonparametric Regression and Generalized Linear Models: A Roughness Penalty Approach*. Chapman & Hall (1993)
7. Ramsay, J., Silverman, B.: *Applied Functional Data Analysis*. Springer-Verlag (2002)
8. Ramsay, J., Silverman, B.: *Functional Data Analysis*. Springer Series in Statistics. Springer-Verlag, New York (1997)
9. MacKay, D.: A practical bayesian framework for backprop networks. *Neural Computation* **4**(3) (1992) 448–472
10. Tipping, M.: Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning Research* **1** (2001) 211–244
11. Tipping, M., Faul, A.: Fast marginal likelihood maximisation for sparse bayesian models. In Bishop, C., Frey, B., eds.: *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, January 3–6 2003, Key West, Florida. (2003)
12. Figueiredo, M.: Adaptive sparseness for supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **25**(9) (2003) 1150–1159
13. Engel, Y., Mannor, S., Meir, R.: The kernel recursive least squares algorithm. ICNC03 001, Interdisciplinary Center for Neural Computation, Hebrew University, Jerusalem, Israel (2003)
14. Engel, Y., Mannor, S., Meir, R.: Sparse online greedy support vector regression. In Elomaa, T., Mannila, H., Toivonen, H., eds.: *Machine Learning: ECML 2002*. Volume 2430 of *Lecture Notes in Computer Science*., Berlin, Springer-Verlag (2002)
15. Ma, J., Theiler, J., Perkins, S.: Accurate on-line support vector regression. *Neural Computation* **15**(11) (2003) 2683–2704