# Compactly supported RBF kernels for sparsifying the Gram matrix in LS-SVM regression models

B. Hamers, J.A.K. Suykens, B. De Moor

K.U.Leuven, ESAT-SCD/SISTA, Kasteelpark Arenberg 10, B-3001 Leuven, Belgium
{bart.hamers,johan.suykens}@esat.kuleuven.ac.be

**Abstract.** In this paper we investigate the use of compactly supported RBF kernels for nonlinear function estimation with LS-SVMs. The choice of compact kernels recently proposed by Genton may lead to computational improvements and memory reduction. Examples however illustrate that compactly supported RBF kernels may lead to severe loss in generalization performance for some applications, e.g. in chaotic time-series prediction. As a result, the usefulness of such kernels may be much more application dependent than the use of the RBF kernel.

**Keywords.** Support vector machines, nonlinear function estimation, compactly supported kernels, direct and iterative methods.

## 1 Introduction

Recently kernel methods for pattern recognition and nonlinear function estimation have received a lot of attention. Although the performances of these methods is often excellent, one of the disadvantages is the upscaling to larger data sets. This is caused by the fact that many optimization methods demand the storage of a dense positive definite Gram matrix. Genton [1] recently showed an efficient method for constructing kernels with compact support without destroying the positive definiteness of the Gram matrix. In this paper we study the result of the use of compactly supported RBF kernels. RBF kernels are frequently used in nonlinear function estimation problems [2]. A compactified version of this kernel could be computationally attractive. In this paper we apply this kernel to a number of toy problems and real life data sets. As a result we observe that on certain problems such as chaotic time series prediction the use of compactly supported RBF kernels leads to loss in generalization performance, while for other problems (e.g. in lower dimensional problems) the quality of the results is comparable.

This paper is organised as follows. In section 2 we discuss the compactly supported RBF kernel. In section 3 we discuss methods for solving LS-SVM systems and how to exploit sparseness in the Gram matrix. In section 4 illustrations on artificial and real life data sets are given.

## 2 Kernel matrix and compactly supported kernels

The kernel functions that are used in the support vector literature [1] are functions $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R} : (x, z) \mapsto K(x, z)$. One works with positive definite kernels that satisfy the Mercer condition. Given a training data set $\{x_i, y_i\}_{i=1}^N$ with inputs $x_i \in \mathbb{R}^d$ and outputs $y_i \in \mathbb{R}$ this results in a kernel matrix or Gram matrix $\Omega \in \mathbb{R}^{N\times N}$ that is positive definite, where $\Omega_{ij} = K(x_i, x_j)$.

In nonlinear function estimation frequently used kernels are the radial basis function (RBF) kernel $K(x,z) = \exp(-\|x-z\|^2/\sigma^2)$. where $\sigma \in \mathbb{R}$ is a tuning parameter of the model. These Gaussian kernels are special cases of the class of Matérn type kernels [3]. An important property of this class of kernels is that they can easily be transformed into *compactly supported kernels.* This means that the kernel will be zero if $\|x-z\|$ is larger than a cut-off distance $\theta'$. As explained in Genton [3] one can multiply the kernel by $\max\{0, (1-\|x-z\|/\theta')^{\nu'}\}$ where $\theta' > 0$ and $\nu' \geq (d+1)/2$ to ensure positive definiteness. The danger of cutting off a kernel in another way is that one will loose positive definiteness. In this paper we investigate the use of *compactly supported Gaussian RBF kernel* (CS-RBF)

$$K(x,z) = \max\left\{0, \left(1 - \frac{\|x-z\|}{3\sigma}\right)^{\nu'}\right\} \exp\left(-\frac{\|x-z\|^2}{\sigma^2}\right). \tag{1}$$

In order to avoid having too many extra parameters we decided to take the *cut-off point* $\theta' = 3\sigma$, where $\sigma$ denotes the bandwidth of the Gaussian RBF kernel. $\nu'$ is chosen to be equal to the dimension of the input variables for the odd cases; when the dimension is even, it is augmented by one.

## 3 Nonlinear function estimation using LS-SVMs

We test the CS-RBF kernel in the context of LS-SVMs for nonlinear function estimation. This method is closely related to regularization networks, Gaussian processes and kernel ridge regression [1, 2]. The emphasis in the LS-SVM formulation is on primal-dual interpretations as in standard SVM, but simplified to a ridge regression formulation in the primal weight space which can be infinite dimensional. In the primal weight space one has the model $y_i = w^T\varphi(x_i) + b + e_i$ with $\varphi(\cdot)$ the mapping to a high dimensional feature space as in standard SVMs. $e_i$ denotes the error for the $i$-th training data point. One minimizes $\min_{w,b,e}(1/2)w^Tw + \gamma\sum_{i=1}^N e_i^2$ s.t. $y_i = w^T\varphi(x_i) + b + e_i$ for $i = 1,..,N$. For this constrained optimization problem one constructs a Lagrangian. The dual problem gives the KKT system

$$\left[\begin{array}{c|c} 0 & 1_v^T \\ \hline 1_v & \Omega + I_N/\gamma \end{array}\right] \left[\begin{array}{c} b \\ \hline \alpha \end{array}\right] = \left[\begin{array}{c} 0 \\ \hline y \end{array}\right] \tag{2}$$

with $\alpha = [\alpha_1; ...; \alpha_N]$, $y = [y_1; ...; y_N]$, $1_v = [1; ...; 1]$. The resulting model $\hat{f}(x) = \sum_{i=1}^N \alpha_i K(x, x_i) + b$ with application of the kernel trick $K(x_i, x_j) = \varphi(x_i)^T\varphi(x_j)$. These models can be robustified and sparsified as explained in [7]. Many algorithms for solving the linear system require a positive definite matrix which is not the case here. Therefore one can transform this system into $H\eta = 1_v$ and $H\nu = y$ with $H = \Omega + I_N/\gamma$ positive definite. From this we find that $b = \eta^T 1_v/s$ and $\alpha = \nu - b\eta$ where $s = \eta^T 1_v$.

One of the standard numerical methods for solving the linear systems with matrix $H$ is the *Cholesky factorization* [5]. An important disadvantage is that the matrix has to be completely stored in memory. As a result of the CS-RBF kernel one gets a sparse matrix. The memory requirements become proportional to the number of non-zero elements $n_z$. The computational cost is reduced by making efficient use of the zero elements in the matrix. There exist different permutation algorithms (column count permutation, symmetric minimum degree, reverse Cuthill-McKee,...)[4] on the elements of the sparse matrix that give a higher degree of sparseness in the Cholesky factor.

A second important class of methods to solve linear systems are Krylov methods. Such iterative methods are suitable for solving large scale problems. The *conjugate gradient* (CG) method can only be applied to positive definite matrices [6],[5]. The most demanding part in this algorithm is the matrix-vector product between $H$ and the conjugate directions. This can also be reduced by a CS-RBF kernel. The number $n_z$ can be exploited at this point. In the CG method the $\kappa(H)$ determines the convergence (note that this also depends on the regularization constant $\gamma$ and $\sigma$).

## 4 Examples

We investigate here the use of the CS-RBF kernel on a number of artificial and real-life data sets.

### 4.1 Sinc toy problem

Here we compare CS-RBF and RBF kernels for a noisy sinc function $f(x) = \sin(x)/x$ estimated by LS-SVMs. The tuning parameters are selected as $\gamma = 1.5$ and $\sigma = 3.7$. The inputs were take between -20 and 20 with an interspacing of 0.03. We added Gaussian noise to the inputs with zero mean and standard deviation 0.1. Fig.1 shows that the performance of regression with the RBF and CS-RBF are almost the same. CS-RBF gives a slightly larger bias and less smooth results. The pointwise variance of $\hat{f}(x)$ is larger for the CS-RBF kernel.

An advantage of the CS-RBF kernel is the sparse kernel matrix. For large data sets this results in a memory reduction. In the example of the sinc-function with 1334 training points, the number of non-zero elements decreases from $1334^2$=1779556 to $n_z$ =850160. In this one-dimensional problem the Gram matrix also has a very clear band structure. Notice that the $H$ matrix is independent of the $y_i$ values of the training set. This means that for each regression problem with the same $x_i$ values for the training set and hyperparameter set $(\gamma, \sigma)$ the $H$ matrix has this sparse band structure. This band structure, in combination with the sparseness in the matrix, makes that there is a speed-up in the training procedure. Depending on the used methods (Cholesky or Conjugate Gradient) the time needed to solve the two systems is the following: the Cholesky factorization needs 9.7450 sec. cpu-time to solve the two linear systems for the Gaussian RBF and 4.3270 sec for the compactly supported RBF. The conjugate gradient method needs respectively 4.1470 sec and 2.6540 sec. Hence, we typically observe that the compactly supported kernel results in a memory reduction and a speed-up of about 50%.

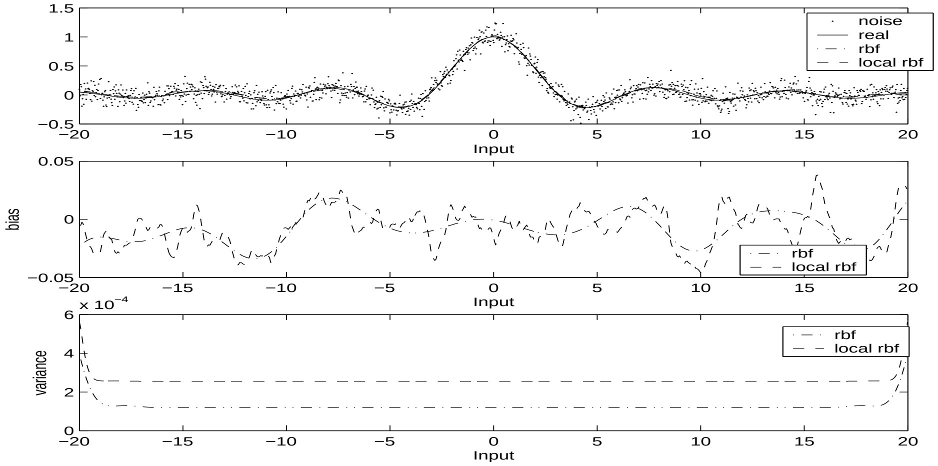


**Fig. 1.** LS-SVM results for nonlinear function estimation on the The middle and bottom part show respectively the bias and variance of both estimates for both kernels.

We also tested the influence of the localization on the condition number of the matrix $H$. Fig.2 shows that there is only a small deference in the condition number of the matrix $H$ for the different values of $(\gamma, \sigma)$ . Therefore, the speed of convergence for CG with RBF or CS-RBF kernels is comparable.

## 4.2 Boston Housing data

As a second example we tested the Boston housing data set. This data set consists of 506 cases in 14 attributes. We trained LS-SVMs on 406 randomly selected training data and used 100 points as test set. We normalized the data except the binary variables. In Table 1 we show the performances of the LS-SVM for different values of the hyperparameter $\sigma$ where $\gamma = 30$ is kept constant. The performances of RBF and CS-RBF kernels were comparable on all performed tests. We see that by decreasing $\sigma$, the Gram matrix will become more and more sparse as a result of the localization of the kernel.

| | $\sigma = 1.5$ | $\sigma = 2.0$ | $\sigma = 10$ |
|---|---|---|---|
| MSEtr | 5.8e-3 | 1.3e-2 | 1.20e-1 |
| MSEtest | 1.1e-1 | 1.0e-1 | 8.45e-2 |
| $n_z/N^2$ | 0.37 | 0.84 | 1 |

**Table 1.** Performance for different values of bandwidth $\sigma$ for CS-RBF kernels onthe Boston housing data. MSEtr and MSEtest are respectively the mean squared error on the training and test set. The ratio $n_z/N^2$ characterizes the degree of sparseness in the Gram matrix.

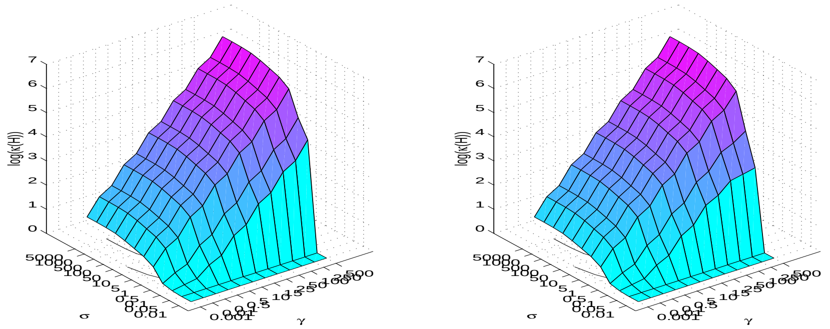
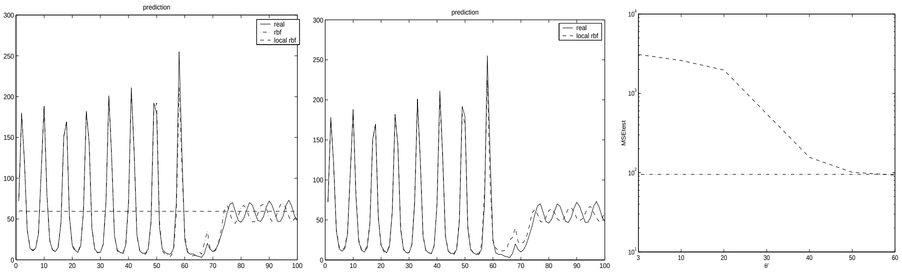


**Fig. 2.** This figure shows the logarithm of the condition number for different hyperparameters $(\gamma, \sigma)$. (Left) RBF kernel; (Right) CS-RBF kernel. Notice that the condition number is not significant larger for the CS-RBF.

### 4.3 Santa Fe chaotic laser data time series prediction

In a third example we use LS-SVM for time series prediction on the Santa Fe laser data set. The model that we use here is based on a trained *one step ahead predictor* $\hat{y}_k = f\left(y_{k-1}, y_{k-2}, ..., y_{k-n}\right)$ with $n = 50$ where $y_k$ denotes the true output at discrete time instant $k$. In Fig.3 we see that a good iterative prediction performance is obtained for the RBF kernel with hyperparameters $(\gamma, \sigma) = (70, 4)$ found by 10-fold crossvalidation. For the same hyperparameters the CS-RBF kernel has a very bad performance as can be seen in Fig.3. For almost similar performance either the cut-off point has to be increased $\theta' = 50\sigma$ or the bandwidth of CS-RBF kernel has to increase. Unfortunately, both reduce the degree of sparseness in the Gram matrix to zero.



(a) (b) (c)

**Fig. 3.** Santa Fe laser data prediction: (a) (-) real data, (- .) RBF kernel (- -) CS-RBF kernel; (b) (- -) CS-RBF with cut-off point $\theta^{'} = 50\sigma$ having no sparseness; (c) MSE on testdata with respect to cut-off point $\theta^{'}$, showing bad results for smaller $\theta^{'}$, i.e. sparse Gram matrix.

## 5 Conclusion

We have studied the use of compactly supported RBF kernels based on recent work by Genton. RBF kernels are frequently used for many applications. The use of compactly supported kernels could decrease the computational cost and memory requirements. In our study we have seen that for certain problems the generalization performance is comparable as well as the conditioning of the matrices towards iterative methods of conjugate gradient. However, on a problem of chaotic time series prediction the compactly supported RBF kernels fails to produce good results when having a sparse Gram matrix. As a result one may conclude that compactly supported RBF kernels may be useful for some specific applications but one should be careful to use it in a general context.

**Acknowledgements**
Our research is supported by grants from several funding agencies and sources: Research Council KUL: Concerted Research Action GOA-Mefisto 666 (Mathematical Engineering), IDO (IOTA Oncology, Genetic networks), several PhD/postdoc & fellow grants; Flemish Government: Fund for Scientific Research Flanders (several PhD/postdoc grants, projects G.0256.97 (subspace), G.0115.01 (bio-i and microarrays), G.0240.99 (multilinear algebra), G.0197.02 (power islands), G.0407.02 (support vector machines), research communities ICCoS, ANMMM), AWI (Bil. Int. Collaboration Hungary/Poland), IWT (Soft4s (softsensors), STWW-Genprom (gene promotor prediction), GBOU McKnow (Knowledge management algorithms), Eureka-Impact (MPC-control), Eureka-FLiTE (flutter modeling), several PhD grants); Belgian Federal Government: DWTC (IUAP IV-02 (1996-2001) and IUAP V-10-29 (2002-2006): Dynamical Systems and Control: Computation, Identification & Modelling), Program Sustainable Development PODO-II (CP-TR-18: Sustainibility effects of Traffic Management Systems); Direct contract research: Verhaert, Electrabel, Elia, Data4s, IPCOS; BDM is a full professor at K.U.Leuven Belgium, JS is a professor at K.U.Leuven Belgium and a postdoctoral researcher with FWO Flanders.

## References

1. Cristianini N., Shawe-Taylor J., *An Introduction to Support Vector Machines*, Cambridge University Press, 2000.
2. Evgeniou T., Pontil M., Poggio T., “Regularization networks and support vector machines”, *Advances in Computational Mathematics*, **13**(1), 1-50, 2000.
3. Genton M., “Classes of kernels for machine learning: a statistics perspective”, *Journal of Machine Learning Research*, **2**, 299-312, 2001.
4. Gilbert J. , Moler C., Schreiber R., “Sparse matrices in Matlab: design and implementation”, *SIAM Journal on Matrix Analysis*, **13**(1), 333-356, 1992.
5. Golub G., Van Loan C. *Matrix Computations*, Baltimore: The John Hopkins University Press, 2nd ed., 1990.
6. Greenbaum A., *Iterative Methods for Solving Linear Systems*, Philadelphia: SIAM, 1997.
7. Suykens J.A.K., De Brabanter J., Lukas L., Vandewalle J. , “Weighted least squares support vector machines: robustness and sparse approximation”, *Neurocomputing*, in press.