ADAPTIVE METRIC KERNEL REGRESSION

Cyril Goutte and Jan Larsen
Department of Mathematical Modeling - Building 321
Technical University of Denmark, DK-2800 Lyngby, Denmark
Phone: +45 4525 3921,3923
Fax: +45 4587 2599
E-mail: cg,jl@imm.dtu.dk

Abstract. Kernel smoothing is a widely used non-parametric pattern recognition technique. By nature, it suffers from the curse of dimensionality and is usually difficult to apply to high input dimensions. In this contribution, we propose an algorithm that adapts the input metric used in multivariate regression by minimising a cross-validation estimate of the generalisation error. This allows to automatically adjust the importance of different dimensions. The improvement in terms of modelling performance is illustrated on a variable selection task where the adaptive metric kernel clearly outperforms the standard approach.

OVERVIEW

Neural Networks are often referred to as a non-parametric model. Their popularity is probably in part linked to the fact that there are many cases in which one tries to model some input-output relationship on the basis of empirical data, without any parametric model of the underlying phenomenon. Kernel methods are the archetypal non-parametric method [7], a well-known tool for eg pattern recognition [10]. They have been "re-invented" in the neural networks literature on several occasions [11, 12], for density estimation, classification and regression purposes. However, it has been consistently noted that they suffer badly from the "curse-of-dimensionality", ie produce poor estimators when the input dimension increases.

In this contribution, we will address a possible improvement on the traditional multivariate kernel method. We will be mainly concerned with regression estimation, but the method presented below applies to classification tasks in a straightforward manner. We will first present some general result on the uni- and multivariate kernel regression estimation. We then introduce our method, based on the adaptive estimation of the feature space metric used by the kernels. We perform a number of experiments on a regression task where a number of irrelevant dimensions are added to the input space. The superiority of the adaptive metric scheme is illustrated by the fact that its performance on the full input space is better than a standard kernel method using only the relevant input information.

KERNEL REGRESSION

Let us consider a standard regression problem: from a number N of input-output pairs $\left(\mathbf{x}^{(k)}, y^{(k)}\right)$ sampled from an unknown input-output joint distribution, we wish to learn the relation which, for a new input $\mathbf{x}$, maps an estimated output $\widehat{y}$. Kernel regression is a widely used non-parametric regression method [7]. Probably the best-known univariate kernel smoother derives from the Parzen window density estimator. It takes the form of the Nadaraya-Watson estimator, which estimates the expected value of the output y given a one-dimensional input x:

$$\widehat{y} = f_\sigma(x) = \frac{\sum_{k=1}^{N} K_\sigma\left(x - x^{(k)}\right) y^{(k)}}{\sum_{k=1}^{N} K_\sigma\left(x - x^{(k)}\right)} \tag{1}$$

where $K_\sigma(x) = K(x/\sigma)/\sigma$ is the scaled kernel with *bandwidth* σ. $K(x)$ is the kernel shape, usually a continuous, bounded and integrable function. Figure 1 displays several kernel shapes which have been scaled to integrate to 1 for easy visual comparison. The actual scaling of the kernel is irrelevant when evaluating equation (1) as the scaling terms in the denominator and the numerator cancel out. The Nadaraya-Watson estimator has been rediscovered several times in the Neural Networks literature, most recently in [11], and usually in limited forms (eg Gaussian kernel shapes only). Kernels with compact supports are uncommon in a neural networks context, though they are usually considered to have an edge in terms of computational cost [7].

The extension of (1) to the multivariate case is straightforward with the definition of a multivariate kernel. We will here consider only the *spherically symmetric* kernel. For a P dimensional input vector $\mathbf{x}$ and a $P \times P$ symmetric, positive definite *bandwidth matrix* $\mathbf{\Sigma}$, the spherically symmetric kernel is defined from the univariate kernel K as:

$$K_{\mathbf{\Sigma}}(\mathbf{x}) \propto \frac{1}{\sqrt{|\mathbf{\Sigma}|}} K\left(\sqrt{\mathbf{x}^\top \mathbf{\Sigma}^{-1}\mathbf{x}}\right) \tag{2}$$

The analysis of the convergence of the multivariate kernel estimator for general matrices $\mathbf{\Sigma}$ is complicated. For the simpler case where $\mathbf{\Sigma} = \sigma^2\mathbf{I}$, $\sigma^2 \neq 0$, it is possible to show that the asymptotic rate of estimation error convergence is $N^{-4/(P+4)}$, assuming additive noise (eg [15] section 4.3). As a comparison, the rate of convergence for parametric methods will typically be $N^{-s/P}$ (eg [14] section 4.3), where s measures the smoothness of the underlying function. These theoretical results illustrate the *curse-of-dimensionality*: for high dimensional inputs (large P), the rate of convergence becomes very slow. Furthermore, for reasonably smooth function, the parametric approach will converge faster than the non-parametric estimator.

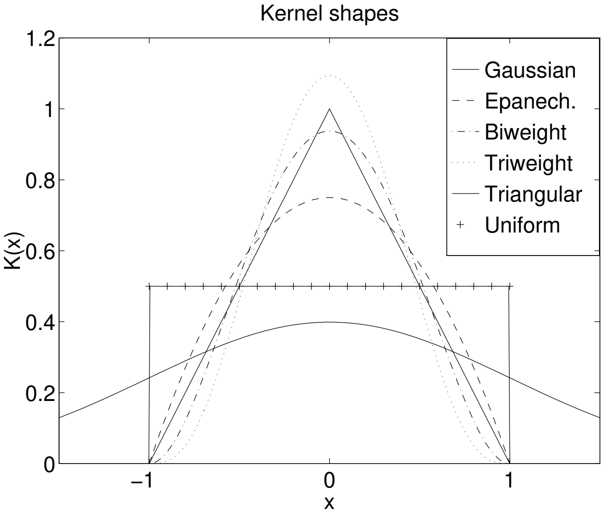

Figure 1: Kernel shapes: 'Epanech.' stands for Epanechnikov (quadratic) kernel.

A number of methods have been devised to automatically set the bandwidth parameter in a univariate setting (eg [15] chapter 3). The multivariate case is usually considered an extension of the univariate case. Likewise, a typical extension is to use a single bandwidth parameter for multivariate regression. We will now present a scheme that is specific to the multivariate nature of the regression.

ADAPTIVE METRIC

Let us now rewrite (1) for multivariate inputs using the squared distance $d^2_{\mathbf{H}}(\mathbf{u},\mathbf{v}) = (\mathbf{u}-\mathbf{v})^\top \mathbf{H}(\mathbf{u}-\mathbf{v})$, parameterised by the positive definite, symmetric, square matrix $\mathbf{H}$, yielding:

$$\widehat{y} = f_{\mathbf{H}}(\mathbf{x}) = \frac{\sum_{k=1}^{N} K\left(d_{\mathbf{H}}\left(\mathbf{x},\mathbf{x}^{(k)}\right)\right) y^{(k)}}{\sum_{k=1}^{N} K\left(d_{\mathbf{H}}\left(\mathbf{x},\mathbf{x}^{(k)}\right)\right)} \tag{3}$$

$\mathbf{H}$ defines the metric in the input space, and is equivalent to the inverse bandwidth matrix in equation (2), $\mathbf{H} \equiv \mathbf{\Sigma}^{-1}$. We will here limit ourselves to a diagonal matrix $\mathbf{H}$ with positive elements parameterised as follows:

$$\mathbf{H} = \mathrm{diag}\left(h_1^2, h_2^2, \dots, h_P^2\right) = \begin{bmatrix} h_1^2 & 0 & \dots & 0 \\ 0 & h_2^2 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \dots & h_P^2 \end{bmatrix} \tag{4}$$

The diagonal matrix introduces more flexibility than the choice of a single-parameter spherical bandwidth $\mathbf{H} = h^2\mathbf{I}$. However, it is less flexible than a full matrix. The parameterisation of a general positive definite matrix (eg by Cholesky decomposition) is more complex, but the derivations presented below can be extended in a straightforward manner. Equation (4) ensures that the diagonal elements stay positive for all h_p. Other parameterisation are possible, among others as $\exp(h)$. Note that the h_p play a role that is inverse to that of the bandwidth in (1). A small h_p corresponds to an irrelevant dimension, ie a broad kernel, while a large h_p reflects an important contribution from the corresponding dimension, akin to a small bandwidth in the traditional approach. In the following, h_p will be referred to as *smoothing* parameter, and h_p^2 as *metric* parameter.

Among the many possible methods for automatically setting the amount of smoothing, we will focus on a traditional approach in statistical learning, namely minimising an estimator of the (average) generalisation error (eg [4], chapter 3). The V-fold cross-validation (XV) estimator [13] is calculated by randomly splitting the N data into V disjoint sets $S_1, \dots S_V$ of equal sizes[1], such that $\bigcup_{i=1}^{V} S_i = \{1, \dots N\}$, and averaging the error observed on one set for the estimator obtained on the rest on the data:

$$\widehat{G}_{XV} = \frac{1}{N} \sum_{j=1}^{V} \left[\sum_{i \in S_j} \left(y^{(i)} - f_{\mathbf{H}}^{(j)} \left(\mathbf{x}^{(i)} \right) \right)^2 \right] \tag{5}$$

where $f_{\mathbf{H}}^{(j)}$ is the kernel estimator excluding set S_j:

$$f_{\mathbf{H}}^{(j)}(\mathbf{x}) = \frac{\sum_{k \notin S_j} K\left(d_{\mathbf{H}} \left(\mathbf{x}, \mathbf{x}^{(k)} \right) \right) y^{(k)}}{\sum_{j \notin S_j} K\left(d_{\mathbf{H}} \left(\mathbf{x}, \mathbf{x}^{(k)} \right) \right)}$$

Equation (5) uses a squared loss function, a common choice that is consistent with the estimation of the expected value of the output y given $\mathbf{x}$. The use of other losses is straightforward. The derivatives of the cross-validation estimator $\widehat{G}_{XV}$ w.r.t. the smoothing parameters h_p turn out to have an analytical expression:

$$\frac{\partial \widehat{G}_{XV}}{\partial h_p} = -\frac{2}{N} \sum_{j=1}^{V} \sum_{i \in S_j} \left(y^{(i)} - f_{\mathbf{H}}^{(j)} \left(\mathbf{x}^{(i)} \right) \right) \frac{\displaystyle\sum_{k \notin S_j} \left(y^{(k)} - f_{\mathbf{H}}^{(j)} \left(\mathbf{x}^{(i)} \right) \right) \frac{\partial K_{ik}}{\partial h_p}}{\displaystyle\sum_{k \notin S_j} K_{ik}} \tag{6}$$

where $K_{ik} = K\left(d_{\mathbf{H}} \left(\mathbf{x}^{(i)}, \mathbf{x}^{(k)} \right) \right)$. Please note that equation (6) is an *exact* expression of the gradient of the generalisation estimator. In the case of neural networks, a similar expression, namely that of the cross-validation derivatives w.r.t. regularisation parameters, can also be derived [8]. Equation (6) depends on the derivative of the kernel weights $K\left(d_{\mathbf{H}} \left(\mathbf{x}^{(i)}, \mathbf{x}^{(k)} \right) \right)$ w.r.t. the smoothing coefficients h_p. This derivative depends on the kernel shapes, and the expressions for the most common kernel shapes are reported in table 1.

[1] The scheme is easily adapted when V is not a divisor in N.

Kernel	$K(\mathbf{u},\mathbf{v})$	$\frac{\partial K(\mathbf{u},\mathbf{v})}{\partial h_p}$
Gaussian	$\exp\left(-\frac{d^2_{\mathbf{H}}(\mathbf{u},\mathbf{v})}{2}\right)$	$-h_p(u_p-v_p)^2\exp\left(-\frac{d^2_{\mathbf{H}}(\mathbf{u},\mathbf{v})}{2}\right)$
Epanechnikov	$\left(1-d^2_{\mathbf{H}}(\mathbf{u},\mathbf{v})\right)1_{\mathbf{H}}$	$-2h_p(u_p-v_p)^2 1_{\mathbf{H}}$
Biweight	$\left(1-d^2_{\mathbf{H}}(\mathbf{u},\mathbf{v})\right)^2 1_{\mathbf{H}}$	$-2h_p(u_p-v_p)^2\left(1-d^2_{\mathbf{H}}(\mathbf{u},\mathbf{v})\right)1_{\mathbf{H}}$
Triweight	$\left(1-d^2_{\mathbf{H}}(\mathbf{u},\mathbf{v})\right)^3 1_{\mathbf{H}}$	$-2h_p(u_p-v_p)^2\left(1-d^2_{\mathbf{H}}(\mathbf{u},\mathbf{v})\right)^2 1_{\mathbf{H}}$
Triangular	$\left(1-\sqrt{d^2_{\mathbf{H}}(\mathbf{u},\mathbf{v})}\right)1_{\mathbf{H}}$	$-\frac{h_p(u_p-v_p)^2}{\sqrt{d^2_{\mathbf{H}}(\mathbf{u},\mathbf{v})}}1_{\mathbf{H}}$

Table 1: Kernel expressions and their derivatives w.r.t. the smoothing parameters, using the parameterisation in (4). $1_{\mathbf{H}} = 1_{\{d^2_{\mathbf{H}}<1\}}$ is the indicator function. Note that no scaling is used so that these kernels do not integrate to 1.

Note that the h_p that factors in front of each derivative is actually the (half) derivative of the metric parameter h_p^2 w.r.t. to the smoothing parameter h_p. We will adapt the expressions in table 1 for other types of parameterisation by replacing $2h_p$ with the corresponding derivative, eg $\exp(h_p)$ if the metric diagonal is parameterised with exponentials.

The tuning of the smoothing parameters now becomes a first-order multi-dimensional optimisation problem, where we are trying to minimise $\widehat{G}_{XV}$ in the P dimensional metric space. This can be performed by a number of methods such as conjugate gradient or quasi-Newton methods [2].

Implementation. The straightforward application of (5) and (6) seems complex due to the many pair distances to consider. However, the calculation of the cross-validation can be eased by considering the $N \times N$ matrix $\mathbf{K}$, the elements of which are K_{ij} if i and j belong to different sets S_k and S_ℓ, $\ell \neq k$, and 0 otherwise. Equation (5) becomes:

$$\widehat{G}_{XV} = \frac{1}{N}\left(\mathbf{Y} - \mathrm{diag}\left(\mathbf{K}\times\mathbf{1}\right)^{-1}\mathbf{K}\mathbf{Y}\right)^2$$

This is calculated in $\mathcal{O}(N^2)$ time. Note also that when N is large, $\mathbf{K}$ can be split in horizontal slabs to fit in memory. The gradient of $\widehat{G}_{XV}$ is also expressed in a simple matrix format:

$$\frac{\partial \widehat{G}_{XV}}{\partial h_p} = -\frac{2}{N}\mathrm{diag}\left(\mathbf{E}\right)^\top \mathrm{diag}\left(\mathrm{diag}\left(\mathbf{K}\times\mathbf{1}\right)^{-1}\mathbf{E}\frac{\partial \mathbf{K}}{\partial h_p}\right) \tag{7}$$

where $\mathbf{E} = \left[e_{ik} = \left(y^{(k)} - f^{(j)}_{\mathbf{H}}\left(\mathbf{x}^{(i)}\right)\right)\right]$, with j such that $i \in S_j$, is the $N \times N$ cross-validation error matrix. The diagonal operator $\mathrm{diag}(\cdot)$ returns either a vector of diagonal elements when the input is a square matrix, or a square diagonal matrix when the input is a vector. As we consider only diagonal elements, equation (7) is calculated in $\mathcal{O}(N^2)$ time. The full calculation of the cross-validation error and its derivatives is therefore $\mathcal{O}(PN^2)$. For V-fold cross-validation, only $\frac{V-1}{V}N$ data samples are available for estimating the regressions $\widehat{y}$. In parametric models, this is justified by the fact that performing successively V regressions can be computationally costly, hence the need for V to remain relatively low. In a non-parametric setting such as kernel regression, this is not an issue, as the estimation is $\mathcal{O}(PN^2)$ whatever the value of V. Finally, note that the smoothing parameter depends on the amount of data. If V-fold cross-validation provides an estimate h_V of a smoothing parameter, the value h used for estimating the regression on the basis of the entire dataset will usually be underestimated. An asymptotic argument based on the volume encompassed by the kernel suggests to correct the overall value as $h \equiv (1 - 1/V)^{-1/P}\, h_V$. When V tends to N, the correcting factor approaches 1 and is thus irrelevant.

EXPERIMENTS

Let us now perform some input selection experiments, in which we illustrate the effect of the adaptive metric. We use a dataset generated by a system described in [3]:

$$y = 10 \sin(\pi x_1 x_2) + 20 \left(x_3 - \frac{1}{2} \right)^2 + 10 x_4 + 5 x_5 + \epsilon \tag{8}$$

with Gaussian noise $\epsilon \sim \mathcal{N}(0, 1)$. The input vector contains 10 uniformly distributed values, $x_1 \ldots x_{10}$, $x_i \sim \mathcal{U}([0, 1])$. It is obvious from equation (8) that the last five inputs are irrelevant. However, the input space has 10 dimensions, which can be considered rather large for classical kernel regression. In order to test several small sample conditions we will consider 4 training set sizes: 50, 100, 200 and 500 samples. In order to test the generalisation abilities, we generate a large test set containing 5000 samples. It should be noted that 500 data points is still a small sample for a non-parametric estimation of this system, even if we limit ourselves to the relevant 5-dimensional space. If we wanted to estimate this function using a mesh of regularly spaced points on each dimension, we would need more than 3000 points to map the uniformly sampled interval [0;1] with 5 points on each of 5 dimension, and close to 10^7 for 10 dimensions.

Metric adaptation. In our experiments, we use Gaussian kernel shapes. The smoothing parameters are estimated by minimising the leave-one-out (LOO) cross-validation error using the conjugate gradient method with an approximate line search [2]. The comparison between different splits will be presented elsewhere [6]. Note that initialisation is not an issue here. Figure 2 presents the metric parameters h_p^2 obtained for different sample sizes. Several replications of each sample size have been used: 200 replication of 50 samples, 100 replications of 100 samples, 50 replications of 200 samples and 20 replications of 500 samples, so that the total number of generated samples is 10^4 in all cases. All average results display a sizeable attenuation of the metric parameters associated with the 5 irrelevant dimensions, numbered 6 to 10. Recall that small values of the metric parameter correspond to broad kernels, ie average over the corresponding input. Another commendable feature is

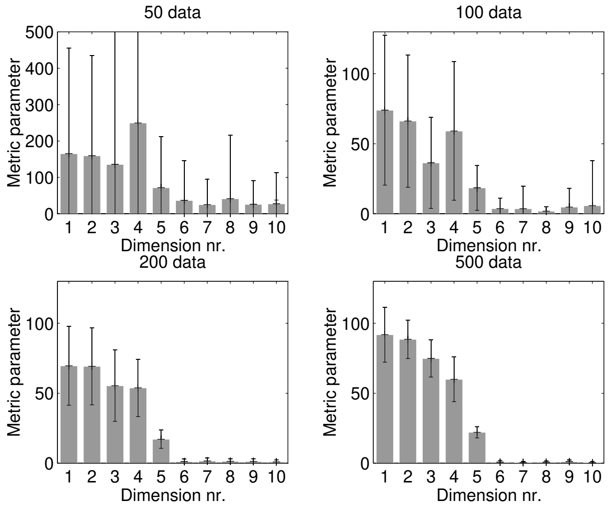

Figure 2: Metric parameters h_p^2, $p \in \{1, \dots 10\}$, for 50 to 500 samples training sets. The broad bar indicates the average and the whiskers the standard deviation, both calculated on 20 to 200 replicated experiments, so that the total amount of samples is 10000. Notice that all but the top-left plot have the same vertical scale.

observed on the first two dimensions: equivalent average metric parameters reflect the symmetric role played by x_1 and x_2.

The standard deviations show that the results obtained with 50 samples are extremely variable, while the high level of the average values suggest that the resulting kernel smoother over-fits the data rather badly. For 100 or more samples, the results associate consistently the first 5 inputs with non-zero metric parameters. The standard deviation on each parameter decreases with larger sample sizes, while the metric parameters tend to increase. This means that the equivalent kernel size becomes smaller as the amount of data increases, a feature that is expected intuitively and asymptotically.

Performance. Let us now compare the three following models:

- linear ridge regression on 10 inputs, obviously a bad choice for modelling (8),
- kernel regression with one smoothing parameter $\mathbf{H} = h^2\mathbf{I}$,
- adaptive metric kernel with a diagonal smoothing matrix $\mathbf{H} = \mathrm{diag}(h_1^2, h_2^2, \dots, h_{10}^2)$.

All parameters (ridge, h and h_p) are set by minimising the LOO error. In order to test how the input dimension is effectively reduced by the adaptive metric approach, we will use the last two models on the full input space (5 relevant dimensions + 5 noisy dimensions) as well as on the reduced input space (5 relevant dimensions). The results from table 2 are averages obtained on three dataset sizes containing 100, 200 and 500 samples, with 100, 50

Sample size:		100		200		500	
Model	Dim	Train	Test	Train	Test	Train	Test
Linear	10	6.21	7.74	6.33	7.38	6.92	7.19
Standard kernel	10	0.48	11.4	0.42	9.82	0.34	7.84
Standard kernel	5	0.93	6.37	0.78	4.96	0.74	3.44
Adaptive metric	10	0.24	7.30	0.38	**4.87**	0.50	**3.20**
Adaptive metric	5	0.64	6.50	0.60	**4.71**	0.65	**3.17**

Table 2: Average results of the three models for three training set sizes. 'Train' and 'Test' stand for the training and test error (calculated on 5000 samples).

and 20 replications, respectively. As expected, both kernel methods perform better than linear regression on the training set. However, the standard kernel (with 10 inputs) does consistently worse on the test set, while the adaptive metric method performs slightly worse on this particular 100 sample dataset, and clearly better on larger sample sizes. For most kernel methods, the results indicate some kind of over-fitting, while the test error stay well above the noise level of 1. However, as noted above, we have too few data for a reliable estimation of the regression on the basis of neighbouring data points. When modelling from the 10-dimensional input space, adapting the metric produces a rather impressive gain in performance for all sample sizes. A comparison between the results obtained with 10 inputs versus the 5 relevant inputs shows that the standard kernel gains a lot from the removal of the noisy dimensions. On the other hand, the adaptive metric method displays a very limited gain, reflecting the fact that these noisy inputs have been effectively neutralised. Notice that with 200 and 500 samples, the 10 dimensional adaptive metric kernel performs slightly better than the 5 dimensional standard kernel.

COMMENTS

The above experiments show that the adaptive metric kernel is an efficient alternative to standard kernel methods. It provides a straightforward non-parametric estimator, requires limited training time to estimate the smoothing parameter, and manages to effectively adapt the input space metric to provide better performance. However, we have noted that the curse-of-dimensionality, though limited by adapting the metric, is still a major concern for multivariate kernel regression. As a comparison, we published in [5] an analysis showing that on the same 200 sample dataset, the performance of a non-linear neural network model on the test set was 3.01 and even down to 2.26 after retraining.

The adaptive metric method has some remote links with the method of Gaussian processes [16], or with the expanded Gaussian kernels of [1]. A similar method was proposed in [9], with a number of key differences. First it addressed the case of classification instead of regression. The focus was also on a *local kernel* method, where the size of the kernel depends on the location in space, as with the nearest-neighbour estimation. Furthermore, only leave-one-out, not general V-fold cross-validation is considered. The last and possibly most significant difference is the parameterisation chosen. We use a single diagonal smoothing matrix, while [9] combines the metric parameters with a local kernel size σ. An undesirable effect of this scheme is that it introduces a redundancy in the parameters, hence an infinite number of equivalent solutions.

The software and the datasets used for these experiments (supporting several kernel shapes and general cross-validation split-ratio) are publicly available through the WWW at the following URL:
`ftp://eivind.imm.dtu.dk/dist/software/matlab/AMKREG`

SUMMARY

In this contribution we consider the well-known non-parametric kernel regression method. We show that in a multivariate setting, the smoothing matrix can be parameterised to reflect the metric of the input space. The cross-validation error associated with a given smoothing matrix is explicitly calculated, as well as its derivative w.r.t. the smoothing parameters. This allows to automatically adapt the metric so as to minimise the estimated average generalisation error.

The performance of this approach is compared to a standard kernel regression approach on an input selection problem. It is shown to limit the input space efficiently by selecting the proper features, and yields improved performance for moderate sample sizes. Challenging prospects for future research include the study of other methods for setting the smoothing parameters, as well as the extension of this scheme to a locally weighted regression.

Acknowledgements. This work was supported by the Technical University of Denmark, by BIOMED II grant number BMH4-CT97-2775 and by the Danish Natural Science and Technical Research Councils through the Computational Neural Network Center. CG thanks Carl Edward Rasmussen for discussions on this method and for suggesting the dataset. JL furthermore acknowledges the Radio Parts Foundation for financial support.

REFERENCES

[1] D. T. Davis and J.-N. Hwang, "Expanding gaussian kernels for multivariate conditional density estimation," **IEEE Transactions on Signal Processing**, vol. 46, no. 1, pp. 269–275, 1998.

[2] R. Fletcher, **Practical Methods of Optimization**, Wiley, 1987.

[3] J. Friedman, "Multivariate adaptive regression splines," **The Annals of Statistics**, vol. 19, no. 1, pp. 1–141, 1991.

[4] C. Goutte, **Statistical learning and regularisation in regression**, Ph.D. thesis 1997/033, Université Paris 6, Paris, july 1997, `http://eivind.imm.dtu.dk/staff/goutte/PUBLIS/thesis.html`.

[5] C. Goutte and J. Larsen, "Adaptive regularization of neural networks using conjugate gradient," in **Proceedings of ICASSP'98**, IEEE, 1998, `ftp://eivind.imm.dtu.dk/dist/1998/goutte.icassp98.ps.Z`.

[6] C. Goutte and J. Larsen, "Optimal cross-validation split ratio: Experimental investigation," in **ICANN'98**, 1998, ftp://eivind.imm.dtu.dk/dist/1998/goutte.icann98.html.

[7] W. Härdle, **Applied nonparametric regression**, no. 19 in Econometric Society Monographs, Cambridge University Press, 1990.

[8] J. Larsen, L. K. Hansen, C. Svarer and M. Ohlsson, "Design and regularization of neural networks: the optimal use of a validation set," in S. Usui, Y. Tohkura, S. Katagiri and E. Wilson, eds., **Neural Networks for Signal Processing VI – Proceedings of the 1996 IEEE Workshop**, Piscataway, New Jersey: IEEE, 1996, no. VI in NNSP, pp. 62–71, `ftp://eivind.imm.dtu.dk/dist/1996/larsen.nnsp96.ps.Z`.

[9] D. Lowe, "Similarity metric learning for a variable-kernel classifier," **Neural Computation**, vol. 7, no. 1, pp. 72–85, January 1995.

[10] B. D. Ripley, **Pattern Recognition and Neural Networks**, Cambridge University Press, 1996.

[11] H. Schiøler and U. Hartmann, "Mapping neural networks derived from the Parzen window estimator," **Neural Networks**, vol. 5, no. 6, pp. 903–909, 1992.

[12] D. F. Specht, "A general regression neural network," **IEEE Transactions on Neural Networks**, vol. 2, no. 6, pp. 568–576, November 1991.

[13] M. Stone, "Cross-validatory choice and assessment of statistical predictions," **Journal of the Royal Statistical Society B**, vol. 36, pp. 111–147, 1974, with discussion.

[14] V. N. Vapnik, **The Nature of Statistical Learning Theory**, Springer, 1995.

[15] M. Wand and M. Jones, **Kernel Smoothing**, no. 60 in Monographs on Statistics and Applied Probability, London: Chapman & Hall, 1995.

[16] C. K. I. Williams, "Prediction with Gaussian processes: from linear regression to linear prediction and beyond," Technical Report NCRG/97/012, Neural Computing Research Group, Aston University, UK, 1997.