The Generalized LASSO

Volker Roth

***Abstract*—In the last few years, the support vector machine (SVM) method has motivated new interest in kernel regression techniques. Although the SVM has been shown to exhibit excellent generalization properties in many experiments, it suffers from several drawbacks, both of a theoretical and a technical nature: the absence of probabilistic outputs, the restriction to Mercer kernels, and the steep growth of the number of support vectors with increasing size of the training set. In this paper, we present a different class of kernel regressors that effectively overcome the above problems. We call this approach *generalized LASSO* regression. It has a clear probabilistic interpretation, can handle learning sets that are corrupted by outliers, produces extremely sparse solutions, and is capable of dealing with large-scale problems. For regression functionals which can be modeled as iteratively reweighted least-squares (IRLS) problems, we present a highly efficient algorithm with guaranteed global convergence. This defies a unique framework for sparse regression models in the very rich class of IRLS models, including various types of robust regression models and logistic regression. Performance studies for many standard benchmark datasets effectively demonstrate the advantages of this model over related approaches.**

***Index Terms*—Kernel regression, probabilistic interpretation, robust loss functions, sparsity, support vector machines (SVMs).**

I. Introduction

THE PROBLEM of *regression analysis* is one of the fundamental problems within the field of supervised machine learning. It can be stated as estimating a real valued function, given a sample of noisy observations. In the usual setting of supervised learning, the data is obtained as independently identically distributed (i.i.d.) pairs of feature vectors $\{\boldsymbol{x}_i\}_{i=1}^N$ and corresponding targets $\{y_i\}_{i=1}^N$. For regression problems in particular, the real-valued targets are considered corrupted versions of a set of unobserved values $\{\tilde{y}_i\}_{i=1}^N$ under an additive noise model: $y_i = \tilde{y}_i + \xi_i$, where the noise random variables ξ_i are distributed according to some density function $p(\xi)$. Prior knowledge about this noise distribution is crucial for quantifying the approximation accuracy, i.e., for choosing an adequate *loss function*.

Viewed as a function in $\boldsymbol{x}$, the conditional expectation of y given $\boldsymbol{x}$ is called a *regression function*. Estimation of this function is referred to as the central problem of regression analysis.

A very successful approach to the regression problem is the support vector machine (SVM).[1] It models the regression function by the following expansion, employing a *Mercer kernel* $k(\cdot, \cdot)$

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} k(\boldsymbol{x}, \boldsymbol{x}_i)\alpha_i + \alpha_0. \tag{1}$$

In the SVM regression setting, the approximation quality is measured in terms of an ϵ-insensitive loss function. Deviations smaller than a predefined value ϵ produce no costs at all, while larger deviations are penalized linearly

$$L_\epsilon(\xi) = \begin{cases} |\xi| - \epsilon & \text{for } |\xi| > \epsilon \\ 0 & \text{for } |\xi| \leq \epsilon. \end{cases} \tag{2}$$

Implicit with this loss function is a model of noise in which the data are corrupted by uniform noise on the interval $[-\epsilon, +\epsilon]$, cf. the discussion in Section III-C. The model complexity is controlled by: 1) choosing an adequate kernel function, i.e., an adequate mapping from the input space into some feature space

$$\phi\colon \mathbb{R}^d \mapsto \mathbb{F}, \qquad k(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{\phi}(\boldsymbol{x}_i) \cdot \boldsymbol{\phi}(\boldsymbol{x}_j))$$

and 2) by introducing a spherical Gaussian prior over the weights. Viewed in a regularization context, this allows us to interpret SVM regression as a *ridge regression* model [2]. However, these special assumptions about the noise and the prior over the weights bear some disadvantages.

- Due to the unsmooth shape of the ϵ-insensitive loss function, it is difficult to analytically derive *error bars* for the predictions, see [3] for a related discussion.
- The solutions are usually not very sparse, i.e., the percentage of support vectors usually is relatively high. Even worse, the number of support vectors is strongly correlated with the sample size.
- The use of kernel functions is restricted to those that satisfy Mercer's condition. This restriction may be viewed of minor interest, but for distance measures resulting from matching algorithms, we often derive "quasi" kernels with some negative eigenvalues. For instance, alignment-scores in string matching are an example of this kind.

In this paper, an alternative approach to sparse kernel regression is presented which overcomes these drawbacks. We refer to this method as generalized least absolute shrinkage and selection operator (LASSO) regression.[2] In regression problems, it allows us to employ a large class of loss functions in accordance with our prior knowledge about the noise. Of particular interest among these loss functions are those which correspond to a *robust density* according to Huber's theorem, [5]. Under

Manuscript received August 27, 2001; revised May 15, 2002. This work was supported by the German Research Council (DFG).

The author is with the Department of Computer Science III, University of Bonn, D-53117 Bonn, Germany (e-mail: roth@cs.uni-bonn.de).

Digital Object Identifier 10.1109/TNN.2003.809398

[1]For a recent overview on the SVM and related techniques, see [1].

[2]The acronym LASSO was introduced in [4].

very general conditions, such robust loss functions guarantee the smallest loss in a worst case scenario.

In order to make the generalized LASSO applicable to large-scale problems, we suitably adapt an efficient LASSO algorithm that has been proposed in [6]. The main contribution here is that we show the global convergence of this algorithm for all iteratively reweighted least squares (IRLS) problems. A second contribution of this paper concerns the problem of estimating the prediction variance for robust loss functions. By a theoretical analysis of the relationship between the LASSO and adaptive ridge regression, we present a method for analytically computing Lagrange parameters associated with the optimization problem. These parameters are the main ingredient for deriving error bars within a probabilistic framework.[3]

Since the class of IRLS problems also includes generalized regression methods like logistic regression (LOGREG), the proposed algorithm allows us to naturally extend the regression framework to classification problems.

One of the most outstanding features of the generalized LASSO estimates is their extreme sparsity. Usually, the largest fraction of the expansion coefficients α_i in (1) becomes zero. This sparsity has the following main advantages.

- Concerning the learning phase, it can be exploited to develop highly efficient training algorithms that successively build the final kernel expansion without solving the full N-dimensional problem.
- Once we have successfully trained a regression function, we can make predictions for new observations very efficiently, since only a few kernels $k(\boldsymbol{x}, \boldsymbol{x}_i)$ with $\alpha_i \neq 0$ must be evaluated.
- New observations must be compared with only a small subset of input vectors via the kernel function.

We conclude this paper with performance studies for both synthetic and real-word benchmark datasets. The results effectively demonstrate that the generalized LASSO model combines several advantages: on a conceptual level, it allows us to quantify a confidence interval around the predicted values in a probabilistic way. Moreover, it is capable of dealing with situations in which the learning set is corrupted by a large amount of outliers. On a more technical level, the generalized LASSO can handle large-scale problems in a highly efficient way. Compared to the SVM, the solutions are usually sparser by one or two orders of magnitude, which allows us to make predictions in extremely short time.

II. Related Work

A related approach to Bayesian kernel regression has been presented in [7]. This model is referred to as the relevance vector machine (RVM). From a technical point of view, the RVM uses the identical kernel expansion as the SVM (1), but in this case the kernels themselves are interpreted as entries of feature vectors within a generalized linear model

$$\boldsymbol{\phi}(\boldsymbol{x}) = (k(\boldsymbol{x}, \boldsymbol{x}_1), \ldots, k(\boldsymbol{x}, \boldsymbol{x}_N), 1)^T . \tag{3}$$

The additional last entry plays the role of the intercept α_0 in (1).

The key concept of the RVM is the combination of a quadratic loss function with automatic relevance determination (ARD) priors[4] over the expansion coefficients α_j. Similar to the generalized LASSO estimator proposed in this work, the RVM produces extremely sparse solutions. The use of a quadratic loss function allows us to interpret the RVM fits in a probabilistic way. Overcoming this shortcoming of the SVM, however, has introduced some new problems, notably the high computational costs during the training phase, and the sensitivity to outliers in the training set. It should be noticed, however, that some strategies for overcoming these shortcomings have been proposed in the literature recently, [9], [10]. Furthermore, also the SVM has been generalized to squared loss functions and iteratively reweighted least-squares functionals, allowing probabilistic interpretations, see, e.g., [11]–[14].

A whole set of similar approaches to sparse regression falls under the nomenclature *adaptive ridge* regression. All models of this kind essentially perform *generalized ridge regression*, [2], with either Bayesian priors, see e.g., [15], [16], or with some constraints on the shrinkage weights. A method of the latter kind has been proposed in [17], where the sum of inverse shrinkage weights is constrained to a predefined value. Minimizing a regression functional under such constraint can be viewed as balancing the penalty on each variable, while keeping the mean penalty constant. We will discuss this model in detail in Sections III and IV-B.

A different class of sparse regressors falls into the category of ℓ_1-penalized functionals, see, e.g., the work presented in [18]. The LASSO [4] is another popular method of this kind. It naturally links the class of ℓ_1-constrained models to that of *adaptive ridge* models: in [17] it has been proven that for any *adaptive ridge* model there exists an equivalent LASSO model. From our point of view, the most interesting property of the LASSO is its "built-in" sparsifying mechanism: the regression fits are sparse and interpretable, in the sense that many variables are "pruned" from the model. The main problems, on the other hand, are the following: 1) for most LASSO algorithms, the computational costs are very high for kernel models, where the number of parameters equals the number of samples; 2) applied to IRLS problems with additional subiterations (e.g., as proposed in [19]), it is unclear if global convergence takes place; and 3) a probabilistic interpretation of the fits is difficult, in particular in the case of robust loss functions.

The main ideas of the LASSO, however, can be adopted for our purposes, i.e., for constructing sparse kernel models that can be interpreted in a probabilistic way. The overall procedure can be outlined as follows: we show that an efficient subset-algorithm for the LASSO, introduced in [6], can be extended to the class of IRLS models, while global convergence remains guaranteed. Based on the work presented in [17], the LASSO can be linked to a Bayesian regression model employing ARD priors. We propose a method for analytically deriving the Lagrange parameter that makes this link explicit. This allows us to estimate the prediction variance of the estimator within a well-defined probabilistic framework.

[3]For potential problems with these variance estimates see the discussion following (18).

[4]ARD priors are formalized in the next section. For a general introduction to the ARD principle the reader is referred to [8].

III. Sparse Bayesian Kernel Regression

Applying a Bayesian method to the regression problem requires us to specify a set of probabilistic models of the data, see, e.g., [20]. A member of this set is called a hypothesis $\mathcal{H}$ which will have a prior probability $p(\mathcal{H})$. The likelihood of hypothesis $\mathcal{H}$ is $p(\mathcal{D}|\ \mathcal{H})$, where $\mathcal{D}$ represents the data given as input-output pairs $\{(\boldsymbol{x}_i,\, y_i)\}_{i=1}^N$. For regression problems, each $\mathcal{H}$ corresponds to a regression function $f(\boldsymbol{x})$. Under the assumption that the targets y are generated by corrupting the values of f by additive Gaussian noise of variance σ, the likelihood of model $\mathcal{H}$ is given by

$$\prod_i p(y_i|\,\boldsymbol{x}_i,\,\mathcal{H}) = \prod_i \frac{1}{\sqrt{2\pi\sigma^2}}\,\exp\left\{-(y_i - f(\boldsymbol{x}_i))^2/(2\sigma^2)\right\}. \quad (4)$$

A preference for smooth regression functions is encoded by introducing *priors* over the coefficients $\boldsymbol{\alpha} = (\alpha_1,\,\ldots,\,\alpha_{N+1})^T$. In a regularization context, such prior information can be interpreted as *biasing* maximum likelihood parameter estimates in order to reduce the estimator's variance. The common choice of a *spherical Gaussian prior distribution* with covariance $\Sigma_\alpha \propto \lambda^{-1}I$ leads to the well-known class of *ridge regression* models, [2]. For instance, the standard formulation of support vector regression can be viewed as a method of this kind.

Contrary to ridge regression, the key concept of both the generalized LASSO and the RVM is the use of automatic relevance determination (ARD) priors of the following form

$$p(\boldsymbol{\alpha}|\,\boldsymbol{\vartheta}') = \mathcal{N}(\mathbf{0},\,\Sigma_\alpha) = \prod_i \mathcal{N}(0,\,\vartheta'^{-1}_i) \propto \exp\left\{-\sum_i \vartheta'_i\alpha_i^2\right\}. \quad (5)$$

In this case, each expansion coefficient has its own prior variance ϑ'^{-1}_i. Denoting with K the augmented kernel "design" matrix, i.e., $K_{i,\,j} = k(\boldsymbol{x}_i,\,\boldsymbol{x}_j)\,\forall\ 1 \le i,\,j \le N,\ K_{i,\,N+1} = 1$, this prior model leads us to a posterior of the form

$$p(\boldsymbol{\alpha}|\boldsymbol{y},\,\boldsymbol{\vartheta}',\,\sigma^2) = (2\pi)^{-(N+1)/2}\,|A|^{1/2} \cdot \exp\left\{-\tfrac{1}{2}(\boldsymbol{\alpha}-\overline{\boldsymbol{\alpha}})^T A(\boldsymbol{\alpha}-\overline{\boldsymbol{\alpha}})\right\} \quad (6)$$

with (inverse) covariance matrix $A = \Sigma_\alpha^{-1} + (1/\sigma^2)K^TK$ and mean vector $\overline{\boldsymbol{\alpha}} = (1/\sigma^2)A^{-1}K^T\boldsymbol{y}$. From the form of the likelihood (4) and the prior (5) it is clear that $\overline{\boldsymbol{\alpha}}$ minimizes the quadratic form

$$M(\boldsymbol{\alpha}) = \|\boldsymbol{y} - K\boldsymbol{\alpha}\|^2 + \sigma^2\boldsymbol{\alpha}^T\Sigma_\alpha^{-1}\boldsymbol{\alpha} = \|\boldsymbol{y} - K\boldsymbol{\alpha}\|^2 + \sum_i \vartheta_i\alpha_i^2 \quad (7)$$

where we have defined $\boldsymbol{\vartheta} := \sigma^2\boldsymbol{\vartheta}'$ for the sake of simplicity.

The *mean prediction* for a new input $\boldsymbol{x}_*$ is given by

$$\overline{f}(\boldsymbol{x}_*) = \boldsymbol{\phi}^T(\boldsymbol{x}_*)\overline{\boldsymbol{\alpha}} = \frac{1}{\sigma^2}\,\boldsymbol{\phi}^T(\boldsymbol{x}_*)A^{-1}K^T\boldsymbol{y}. \quad (8)$$

The uncertainty of the prediction is measured by the variance about the posterior mean

$$\mathrm{var}[f(\boldsymbol{x}_*)] = E_{\alpha|\mathcal{D}}\left[\left(f(\boldsymbol{x}_*) - \overline{f}(\boldsymbol{x}_*)\right)^2\right] = \boldsymbol{\phi}^T(\boldsymbol{x}_*)A^{-1}\boldsymbol{\phi}(\boldsymbol{x}_*). \quad (9)$$

The total predictive variance is the sum of the noise variance and the above variance about the mean, since both sources of variation are uncorrelated by assumption.

A. Inferring the Prior Parameters

Given the above class of ARD models, there are now different inference strategies for the prior parameters:

- In [7] the RVM was proposed as a (partially) Bayesian strategy: integrating out the expansion coefficients in the posterior distribution, one obtains an analytical expression for the marginal likelihood $p(\boldsymbol{y}|\,\boldsymbol{\vartheta}',\,\sigma^2)$, or *evidence*, for the hyperparameters. For ideal Bayesian inference one should define hyperpriors over $\boldsymbol{\vartheta}$ and σ, and integrate out these parameters. Since closed-form solution for this marginalization in most cases do not exist, however, it is common to use a Gaussian approximation of the posterior mode. The *most probable* parameters $\boldsymbol{\vartheta}'_{MP}$ are chosen by maximizing $p(\boldsymbol{y}|\,\boldsymbol{\vartheta}',\,\sigma^2)$. Given a current estimate for the $\boldsymbol{\alpha}$ vector, the parameters ϑ'_k are derived as
$$(\vartheta'_k)^{\mathrm{new}} = (1 - \vartheta'_k(A^{-1})_{kk})/\alpha_k^2. \quad (10)$$
These values are then substituted into the posterior (6) in order to get a new estimate for the expansion coefficients
$$(\boldsymbol{\alpha})^{\mathrm{new}} = \left(K^TK + \sigma^2\mathrm{diag}\left\{\boldsymbol{\vartheta}'\right\}\right)^{-1}K^T\boldsymbol{y}. \quad (11)$$
During iterated application of (10) and (11), it turns out that some parameters ϑ'_j approach infinity, which means that the variance of the corresponding priors $p(\alpha_j|\vartheta'_j)$ becomes zero, and in turn the posterior $p(\alpha_j|\,\boldsymbol{y},\,\boldsymbol{\vartheta}',\,\sigma^2)$ becomes infinitely peaked at zero. As a consequence, the coefficients α_j are shrinked to zero, and the corresponding variables (the columns of the kernel matrix K) are removed from the model.
- Adaptive Ridge (AdR) regression, [17], constitutes a different type of parameter updates. The key concept of AdR is to select the parameters ϑ_j by minimizing the functional (7) under a constraint of the form
$$\frac{1}{N+1}\sum_{i=1}^{N+1}\frac{1}{\vartheta_i} = \frac{1}{\lambda}, \qquad \vartheta_i > 0 \quad (12)$$
where λ is a predefined value. This constraint connects the individual variances of the ARD prior by requiring that their *mean variance* is proportional to $1/\lambda$. The conceptual idea behind (12) is to start with an ridge-type estimate ($\vartheta_i = \lambda\,\forall\ i$) and then introduce a method of *automatically balancing* the penalization on each variable, while keeping the average penalty constant.

  Using the standard Lagrangian method, one derives the following update rules.
$$(\overline{\boldsymbol{\alpha}})^{\mathrm{new}} = \left(K^TK + \mathrm{diag}\left\{\boldsymbol{\vartheta}\right\}\right)^{-1}K^T\boldsymbol{y} \quad (13)$$
$$(\vartheta_k)^{\mathrm{new}} = \frac{\lambda}{N+1}\,\frac{\sum_{i=1}^{N+1}|\alpha_i|}{|\alpha_k|}. \quad (14)$$
Similar to the RVM model, during iterated application of these update rules many coefficients α_i are shrinked to

zero, so that the corresponding variables are removed from the model.

In order to interpret AdR in a Bayesian way, it is useful to exploit its algebraic equivalence to the LASSO (see [4] and [17] and the discussion in Section IV-B). The LASSO is simply an ℓ_1-constrained least-squares problem in which the hyperparameters ϑ no longer appear

$$\text{minimize } \|\boldsymbol{y} - K\boldsymbol{\alpha}\|_2^2 \quad \text{s.t. } \|\boldsymbol{\alpha}\|_1 < \kappa. \tag{15}$$

The constraint value κ determines the approximation accuracy and can thus be viewed as a model-selection parameter. Returning to our Bayesian viewpoint of regression, we can interpret the equivalence between AdR and LASSO by way of the following marginalization procedure (cf. [16] and [21]): given exponential hyperpriors

$$p(\vartheta_i) = \frac{\gamma}{2} \exp\left\{-\frac{\gamma\vartheta_i}{2}\right\}$$

we can analytically integrate out the hyperparameters from the prior distribution over the weights α_i

$$p(\alpha_i) = \int_0^\infty p(\alpha_i|\vartheta_i)p(\vartheta_i)\,d\vartheta_i = \frac{\gamma}{2}\exp\{-\sqrt{\gamma}|\alpha_i|\}. \tag{16}$$

Together with the Gaussian likelihood (4), this marginalization leads us to the desired ℓ_1 constrained LASSO-type functional in log-space

$$M^{\text{LASSO}}(\boldsymbol{\alpha}) = \|\boldsymbol{y} - K\boldsymbol{\alpha}\|_2^2 + \tilde{\lambda}\,\|\boldsymbol{\alpha}\|_1 \tag{17}$$

where we have defined the Lagrange parameter $\tilde{\lambda} := \sqrt{\gamma}$. From the above, we conclude that both the RVM and the LASSO employ some Bayesian marginalization steps: in the case of the RVM, one marginalizes over the weights α_i, and updates the hyperparameters ϑ_i by maximizing the likelihood. For the LASSO, one defines exponential hyperpriors and integrates out the hyperparameters. The model complexity is then controlled by the Lagrange parameter $\tilde{\lambda}$ (or by the constraint value κ), which may be chosen adaptively by some model selection criterion. It should be noticed, however, that without taking into account the above marginalization procedure, the LASSO itself may be more considered a penalized likelihood model, rather than a true Bayesian technique.

As stated earlier, both the RVM and the LASSO fits are usually very sparse, in the sense that most of the parameters ϑ_i will tend to infinity. To deal with these infinite parameters when computing variance estimates according to (9), it is useful to introduce the permutation matrix P that collects the nonzero entries of $\boldsymbol{\alpha}$ (or the finite entries of $\boldsymbol{\vartheta}$, respectively) in the first $|\tau|$ components, i.e., $\boldsymbol{\alpha} = P^T\begin{pmatrix}\boldsymbol{\alpha}_\tau \\ \boldsymbol{0}\end{pmatrix}$. We use the same permutation to collect the corresponding features in the vectors $\boldsymbol{\phi}_\tau$, which are the rows of the rearranged $(N \times |\tau|)$ design matrix K_τ. With this notation, the predicted function value of a new observation reads $\overline{f}(\boldsymbol{x}_*) = \boldsymbol{\phi}_\tau^T(\boldsymbol{x}_*)\overline{\boldsymbol{\alpha}}_\tau$, and we can measure the uncertainty of the prediction as

$$\text{var}[f(\boldsymbol{x}_*)] = \sigma^2\boldsymbol{\phi}_\tau^T(\boldsymbol{x}_*)\left[K_\tau^T K_\tau + \text{diag}\{\boldsymbol{\vartheta}_\tau\}\right]^{-1}\boldsymbol{\phi}_\tau(\boldsymbol{x}_*) \tag{18}$$

where σ^2 is an estimate for the variance of the Gaussian noise in the least-squares problem (7). While (18) is the "correct" probabilistic variance estimate, a potential problem with this formula is that it gives an estimated variance of zero for variables with zero α_i (cf. [4]). Thus, kernels with $\alpha_i = 0$ do not contribute to increase the estimate of variance although these coefficients might be nonzero for other i.i.d. observations. The variance estimates presented in this paper should thus be considered as an approximative version of the "true" estimates. It should be noticed, that in [6] a different strategy for estimating the variance has been proposed which overcomes this problem. For kernel models, however, the latter approach is inapplicable, since it requires a full rank design matrix, whereas the set of augmented feature vectors defined in (3) always leads to rank-deficient design matrices.

B. Generalized LASSO as an IRLS Problem

A central assumption so far was the Gaussian noise model that directly lead us to problems of minimizing least-squares functionals. If the true noise density deviates from the Gaussian model, however, the use of such quadratic loss functions may be clearly suboptimal. Since strict prior knowledge about normality of the underlying noise process is hardly available in practice, the more general situation where we have only partial prior information was investigated in [5]. As a key result, it has been proven that the famous Huber's robust loss function

$$L(\xi) = \begin{cases} c|\xi| - \dfrac{c^2}{2} & \text{for } |\xi| > c \\ \dfrac{\xi^2}{2} & \text{for } |\xi| \le c \end{cases} \tag{19}$$

is optimal (in the sense that it guarantees the smallest loss in a worst case scenario), if the true noise density is a mixture of two components, one of which is known to be Gaussian distributed and the other one is an arbitrary density. Huber's loss function penalizes large deviations only linearly. Thus, it is superior to its quadratic counterpart in situations where the data contains outliers which are generated by an unknown and possibly highly fluctuating noise source.

In spite of the fact, that for Huber's loss function the posterior distribution of the coefficients α_i is no longer Gaussian, we can still approximate error bars for the predictions. The key concept here is that the minimization problem can be reformulated as an IRLS problem, which allows us to estimate the prediction variance based on this transformed least-squares problem. Suppose we are given an estimate of the parameters ϑ_k. Many of them usually will tend to infinity, so that the coefficients α_k must vanish and the corresponding variables (the columns of K) are removed from the model. With the notation of the permutation matrix τ, the robust analog of the least-squares functional (7) reads

$$M_{\text{rob}} = \sum_{i=1}^{N} L\left(y_i - (K_\tau\boldsymbol{\alpha}_\tau)_i\right) + \boldsymbol{\alpha}_\tau^T\,\text{diag}\{\boldsymbol{\vartheta}_\tau\}\boldsymbol{\alpha}_\tau. \tag{20}$$

For a wide class of loss functions, the stationary weight vector $\overline{\boldsymbol{\alpha}}$ that minimizes (20) can be found by an iterative algorithm, originally introduced by HUBER, see [5]. A global convergence

proof for a wide class of differentiable loss functions is given in [22]. This algorithm is outlined in the following: computing the partial derivatives of (20), we obtain

$$\nabla_\alpha M_{\rm rob} = K_\tau^T \Omega(\boldsymbol{\alpha}_\tau)\boldsymbol{y} - K_\tau^T \Omega(\boldsymbol{\alpha}_\tau) K_\tau \boldsymbol{\alpha}_\tau + 2\,\mathrm{diag}\{\boldsymbol{\vartheta}_\tau\}\boldsymbol{\alpha}_\tau \tag{21}$$

where Ω denotes the diagonal matrix $\Omega(\boldsymbol{\alpha}_\tau) = \mathrm{diag}\,\{\omega([\boldsymbol{y} - K_\tau \boldsymbol{\alpha}_\tau]_i)\}$, with the auxiliary function $\omega(u) = \partial L(u)/(u\,\partial u)$. For the optimal solution the gradient must vanish, which suggests the following natural successive-substitutions iteration: at any iteration level, the new estimate for the $\boldsymbol{\alpha}$ vector is derived as

$$\boldsymbol{\alpha}_\tau^{\rm new} = \left[K_\tau^T \Omega(\boldsymbol{\alpha}_\tau) K_\tau + 2\,\mathrm{diag}\{\boldsymbol{\vartheta}_\tau\}\right]^{-1} K_\tau^T \Omega(\boldsymbol{\alpha}_\tau)\boldsymbol{y}. \tag{22}$$

A necessary condition for convergence of this procedure is

$$0 < \omega(u) \le \omega(0) < \infty, \qquad \forall\, u. \tag{23}$$

Formally, (22) defines normal equations of a least-squares problem with design matrix $\tilde{K}_\tau = \Omega^{1/2} K_\tau$ and dependent variables $\tilde{\boldsymbol{y}} = \Omega^{1/2}\boldsymbol{y}$. Once the iterations have converged, i.e., once the optimal $\overline{\boldsymbol{\alpha}}$ fixed, we can predict the function value of a new observation with feature vector $\boldsymbol{\phi}_\tau(\boldsymbol{x}_*)$ as $\overline{f}(\boldsymbol{x}_*) = \boldsymbol{\phi}_\tau^T(\boldsymbol{x}_*)\overline{\boldsymbol{\alpha}}_\tau$. Furthermore, in analogy to (9) we can measure the uncertainty of the prediction based on the variance of the expansion coefficients in the transformed least-squares problem:

$$\mathrm{var}[f(\boldsymbol{x}_*)] = \sigma^2 \boldsymbol{\phi}_\tau^T(\boldsymbol{x}_*)\left[\tilde{K}_\tau^T \tilde{K}_\tau + 2\,\mathrm{diag}\{\boldsymbol{\vartheta}_\tau\}\right]^{-1} \boldsymbol{\phi}_\tau(\boldsymbol{x}_*) \tag{24}$$

where σ^2 is an estimate for the variance of the Gaussian noise in the transformed problem.

The IRLS framework, however, also allows us to extend the LASSO principle to classification problems: it is well known in the literature, that LOGREG can be reformulated as a IRLS problem, cf. [23]. In LOGREG, we want to approximate the binary targets for a *Bernoulli* error model

$$p(y|f(\boldsymbol{x})) = (\pi(\boldsymbol{x}))^y (1-\pi(\boldsymbol{x}))^{1-y} = \exp\{\eta y - \log(1+e^\eta)\}$$

where $\pi(\boldsymbol{x}) = p(y=1|\boldsymbol{x})$ denotes the "success" probability, and $\eta = \boldsymbol{\alpha}^T\boldsymbol{\phi}(\boldsymbol{x}) = \log(\pi(\boldsymbol{x})/(1-\pi(\boldsymbol{x})))$. We can also interpret this as approximating the targets with a linear function $f(\boldsymbol{x}) = \eta$ and a cost model according to

$$L_{\rm LOGREG}(y,\, f(\boldsymbol{x})) = -\log\{p(y|f(\boldsymbol{x}))\} = -\eta y + \log(1+e^\eta).$$

The gradient of L can be written as $\nabla_{\boldsymbol{\alpha}} L(\boldsymbol{\alpha}) = K^T W \boldsymbol{e}$, where $\boldsymbol{e}$ is a vector with entries $e_j = (\pi_j - y_j)/W_{jj}$, and $W = W(\boldsymbol{\alpha})$ is a diagonal matrix $W(\boldsymbol{\alpha}) = \mathrm{diag}\{\pi_1(1-\pi_1), \ldots, \pi_N(1-\pi_N)\}$. Forming a variable $\boldsymbol{q} = K\boldsymbol{\alpha} + \boldsymbol{e}$, we can iteratively optimize a LOGREG model with ARD prior by the update equations

$$\boldsymbol{\alpha}_\tau^{\rm new} = [K_\tau^T W(\boldsymbol{\alpha}_\tau) K_\tau + 2\,\mathrm{diag}\{\boldsymbol{\vartheta}_\tau\}]^{-1} K_\tau^T W(\boldsymbol{\alpha}_\tau)\boldsymbol{q}. \tag{25}$$

These, however, are the normal form equations of a regularized least squares problem with input matrix $W^{1/2}K_\tau$ and dependent variables $W^{1/2}\boldsymbol{q}$. Exploiting the identity between AdR models and the LASSO, we can rewrite the above two types of generalized regression problems as ℓ_1-constrained IRLS problems

$$\begin{aligned} &\text{minimize} \|\tilde{\boldsymbol{y}} - \tilde{K}_\tau \boldsymbol{\alpha}_\tau\|_2^2 \qquad \text{s.t.}\ \|\boldsymbol{\alpha}_\tau\|_1 < \kappa \\ &(\tilde{y},\, \tilde{K}_\tau) = \begin{cases} (\Omega^{1/2}\boldsymbol{y},\, \Omega^{1/2}K_\tau) & \text{for robust regression} \\ (W^{1/2}\boldsymbol{q},\, W^{1/2}K_\tau) & \text{for LOGREG.} \end{cases} \end{aligned} \tag{26}$$

C. Relations to SVM Regression

Both the generalized LASSO estimates and the SVM model lead to regression functions that are sparse insofar, as they only depend on a possibly small subset of kernel functions. Thus, (1) usually consists of many zero coefficients. But it is worth noticing that the mechanism that forces some coefficients to vanish is quite different in both models. The sparsity for LASSO estimates with smooth loss functions results from an automated selection of regression variables for a given problem. From a Bayesian viewpoint, this determination of important variables depends only on the ARD prior, whereas the form of the likelihood is readily determined by the noise model. Given such noise model, Huber's theorem, allows us to choose an asymptotically optimal loss function, cf. [5],[24].

In the SVM framework, this is somewhat different. Here, the choice of the loss function does not necessarily reflect the knowledge about the noise. It is rather selected in order to *enforce* sparse solutions independent of the expected noise distribution.[5] For SVM regression, the sparsity results directly from the shape of the ϵ-insensitive loss function. The ϵ gap determines the approximation accuracy and coincides with an ϵ tube around the regression fit. Within this tube, all data produce identically zero costs, independent of their location. Hence, the corresponding input vectors do neither influence the regression fit (all dot products with these vectors have zero weights α_k), nor influence the expansion coefficients of the remaining vectors outside the ϵ tube. The identical fit would be obtained, if they were removed from the sample.

For LASSO estimates, however, the nonzero optimal coefficients $\overline{\alpha}_i$ still depend on *all* input vectors. This can be seen e.g., in (13): if during the iterations a coefficient ϑ_j tends to infinity, only the corresponding column (and not also the row!) of the "design matrix" K will be deleted from the model. Therefore, even an input vector $\boldsymbol{\phi}(\boldsymbol{x}_j)$ with $\vartheta_j = \infty$ influences the nonzero coefficients α_i due to its appearance in the remaining row of K.

Contrary to LASSO regression, for SVM models the reduction of variables can not be interpreted as a method to decrease variance by adding some bias to maximum likelihood estimates. Deleting a variable must be payed for with excluding an input vector from the sample, which implies a complete loss of information contained in this vector. It is obvious that this can only be desirable, if this vector does not provide any further information about the problem, i.e., if its deviance from the regression function is not significant and can be explained as a pure noise event.

[5]To be exact, only the parameterized family of ϵ-insensitive loss functions is selected, and the adaptivity to the assumed noise distribution is limited to the selection of the parameter ϵ within this family.

Implicit with this interpretation is the following noise model: the data are assumed to be affected by *uniform noise* on the interval $[-\epsilon, \epsilon]$, which may be the result of quantization or clipping effects during the measurement process. Note that the ϵ-insensitive loss function has the same structure as the functions that are robust according to Huber's theorem. The associated noise model consists of two components, one of which is uniformly distributed. However, formally the ϵ-insensitive loss does not belong to the family of robust estimators, since the uniform distribution does not possess a smooth derivative, cf. [24, p. 448]. But one can define smooth approximations for which this interpretation is valid. A different interpretation of the noise model can be found in [25], where it is shown that the ϵ-insensitive loss corresponds to a model of Gaussian noise with fluctuating mean and variance. It turns out, that the noise mean is uniformly distributed on the interval $(-\epsilon, \epsilon)$. Note that both interpretations are consistent, since in the latter model errors smaller than ϵ do not count because they may be entirely due to the bias of the Gaussian noise.

If this kind of noise model is in accordance with our prior knowledge, we can combine both methods in the following way: instead of applying the usual quadratic ridge-type regularizer, we may use the LASSO method of penalizing the ℓ_1 norm of the coefficients in order to reduce the variance of the SVM estimates. This gives a model that combines the ϵ-insensitive loss function of SVM regression with the ARD mechanism of the LASSO. Algorithms for models of this kind have been studied in the SVM literature, see e.g., [26], [27], [28]. The unsmooth shape of the ϵ-insensitive loss function, however, makes it impossible to apply the optimization algorithm and the variance estimation procedure presented in this paper.

IV. Algorithms for the Generalized Lasso

In the last section we introduced two different types of sparse kernel regressors, namely the RVM and AdR models. Since we have restated generalized regression problems in an IRLS setting, in principle the update equations of either type can be solved by introducing an additional loop of IRLS subiterations. Both algorithms, however, share two main drawbacks:

i) Even for quadratic loss functions, convergence is rather slow, thus many iterations are needed. This problem becomes even worse, if robust loss functions are employed, due to the additional subiterations.

ii) If during the iterations a coefficient α_k tends to zero, the final estimate will also be zero. Thus, small coefficients have no chance to "recover," which implies that the iterations must be started from the full set of $(N+1)$ variables. Solving the update equations for the new weights α_i then means solving a system of $(N+1)$ linear equations in $(N+1)$ variables, which is very time consuming for large-scale problems.

In the case of AdR regression, the latter problem can partially be overcome by applying approximative conjugate gradient methods for computing the matrix-vector products $\left(K^T K + \operatorname{diag}\{\boldsymbol{\vartheta}\}\right)^{-1}(K^T \boldsymbol{y})$ in (13), cf. [29, p. 83]. Because of problem i), however, the over-all procedure still remains rather time consuming. For the original RVM algorithm, even this speedup is not suitable, since here the update equations of the hyperparameters (10) require us to *explicitly* invert the matrix $\left(K^T K + \operatorname{diag}\{\boldsymbol{\vartheta}\}\right)$ anyway.

While in [9] a new algorithm for more efficient RVM updates has been introduced, in this paper we present an alternative approach by optimizing the generalized LASSO functional. This of course requires efficient optimization strategies for the LASSO functional. Among several LASSO algorithms that have been proposed in the literature, in the following we will focus on the methods described in [4] and [6].

In [4] two quadratic programming algorithms for the LASSO have been proposed, one of which is stated as a problem in d variables and 2^d constraints, the other one as a problem in $2d$ variables and $(2d+1)$ constraints (d denotes the dimensionality of the feature vectors). However, both algorithms are only efficient for small and moderate sizes of d, which unfortunately usually is not the case for kernel regressors with $(d = N+1)$ input dimensions. In real-world applications with several thousands of input vectors, the first algorithm involving 2^{N+1} constraints becomes inapplicable. The second algorithm also bears no advantage over the RVM/AdR updates, since solving a quadratic program in $(2N+3)$ variables is even more difficult than solving a linear system in $(N+1)$ variables. Moreover, applied to IRLS problems with additional subiterations, it is unclear if the algorithms converge in general.

In [6] and [30] a different LASSO algorithm has been introduced that effectively overcomes these numerical problems. Unfortunately, the algorithm has been stated only for quadratic loss functions. In the following we will extend this efficient algorithm for general loss functions. The main contributions here are: 1) it is shown that the convergence proof of the algorithm in [6] can be extended to all loss functions associated with IRLS functionals and 2) an efficient procedure for estimating the variance of generalized LASSO regressors is presented.

Assuming that the loss function $L(\cdot)$ is differentiable, the Lagrangian for the general LASSO problem can be rewritten as

$$\mathcal{L}(\boldsymbol{\alpha}, \tilde{\lambda}) = \left\| \tilde{\boldsymbol{y}} - \tilde{K}\boldsymbol{\alpha} \right\|_2^2 - \tilde{\lambda}\left(\kappa - \|\boldsymbol{\alpha}\|_1\right). \tag{27}$$

According to the Kuhn–Tucker conditions, at the optimal solution the subdifferential $\partial_{\boldsymbol{\alpha}}\mathcal{L}$ must vanish, cf. [31]

$$\partial_{\boldsymbol{\alpha}}\mathcal{L} = -\tilde{K}^T\tilde{\boldsymbol{r}} + \tilde{\lambda}\boldsymbol{v} \stackrel{!}{=} \mathbf{a}, \tag{28}$$

$$\text{with } v_i = \begin{cases} \operatorname{sign}(\alpha_i) & \text{if } \alpha_i \neq 0 \\ a_i \in [-1, 1] & \text{if } \alpha_i = 0. \end{cases} \tag{29}$$

In (29), $\tilde{\boldsymbol{r}} := \tilde{\boldsymbol{y}} - \tilde{K}\boldsymbol{\alpha}$ denotes the vector of residuals. Since $L(\cdot)$ is continuous by assumption and the region of feasible vectors $\boldsymbol{\alpha}$ is compact (the ℓ_1-sphere), a solution $\overline{\boldsymbol{\alpha}}$ is guaranteed to exist. Furthermore, if the constraint is active, the solution vector lies on the boundary of the feasible region, i.e., it satisfies $\|\overline{\boldsymbol{\alpha}}\|_1 = \kappa$.

We now present an efficient subset selection algorithm for the generalized LASSO problem. For the special case of a quadratic loss function, it reduces to the algorithm given in [6]. For the derivation it is useful to introduce some notations: from the form of $\boldsymbol{v}$ in (29) it follows that $\|\boldsymbol{v}\|_\infty = 1$, which together with (28) implies

$$\tilde{\lambda} = \left\| \tilde{K}^T\tilde{\boldsymbol{r}} \right\|_\infty. \tag{30}$$

Furthermore, denote by $\boldsymbol{\theta}_\sigma = \operatorname{sign}(\boldsymbol{\alpha}_\sigma)$ a *sign vector*. The algorithm can now be outlined as follows: given the current estimate $\boldsymbol{\alpha}$, the key idea is to calculate a new search direction $\boldsymbol{h} = P^T\binom{\boldsymbol{h}_\tau}{\mathbf{0}}$ *locally* around $\boldsymbol{\alpha}$. This local problem reads

$$\underset{\boldsymbol{h}}{\text{minimize}} \quad \left\|\tilde{\boldsymbol{y}} - \tilde{K}(\boldsymbol{\alpha}+\boldsymbol{h})\right\|_2^2 \qquad \text{s.t. } \boldsymbol{\theta}_\tau^T(\boldsymbol{\alpha}_\tau + \boldsymbol{h}_\tau) \le \kappa. \tag{31}$$

For *quadratic* loss functions, this problem can be solved analytically (cf. [6]).[6] Note that $\boldsymbol{\alpha}$ in (31) is fixed, and the transormend quantities $(\tilde{\boldsymbol{y}}, \tilde{K})$ are defined with respect to the matrices $\Omega = \Omega(\boldsymbol{\alpha}+\boldsymbol{h})$ in the case of robust regression, or $W = W(\boldsymbol{\alpha}+\boldsymbol{h})$ for LOGREG, cf. (26).

For more general loss functions, (31) defines a "simple" nonlinear optimization problem in $|\tau|$ variables (note that $\boldsymbol{\alpha}$ is fixed). The problem is simple for the following reasons:

1) it is usually a low-dimensional problem, $|\tau| \ll N$;
2) for a wide class of differentiable loss functions it defines a *convex* optimization problem;
3) according to the Kuhn–Tucker conditions, either the constraint is inactive, or the solution lies on the constraint boundary. In the latter case (if the unconstrained solution is not feasible), we have to handle only one simple linear equality constraint, $\theta_\tau^T \boldsymbol{h}_\tau = \kappa'$, with $\kappa' = \kappa - \boldsymbol{\theta}_\tau^T\boldsymbol{\alpha}_\tau$. Efficient solution strategies for problems of this kind can be found in standard textbooks on nonlinear optimization (see, e.g., [32]).

The main advantage of the optimization algorithm is that the iteration can be started from $\boldsymbol{\alpha} = \mathbf{0}$ by choosing an initial s to insert into τ and solving the resulting one-variable subproblem (31). Thus, the optimal $|\tau|$ is found by starting from a small variable set rather than by pruning a large set which would impose severe computational problems and even could be ill-conditioned. With the concept of *sign feasibility* (cf. [31]), the algorithm proceeds as summarized in Algorithm 1.

Check if $\boldsymbol{\alpha}^\dagger := \boldsymbol{\alpha} + \boldsymbol{h}$ is sign feasible, i.e., if $\operatorname{sign}(\boldsymbol{\alpha}_\tau^\dagger) = \boldsymbol{\theta}_\tau$. Otherwise

- A1) Move to the first new zero component in direction $\boldsymbol{h}$, i.e., find the smallest γ, $0 < \gamma < 1$ and corresponding $k \in \tau$ such that $0 = \alpha_k + \gamma h_k$ and set $\boldsymbol{\alpha} = \boldsymbol{\alpha} + \gamma\boldsymbol{h}$.
- A2) There are two possibilities.
  a) Set $\theta_k = -\theta_k$ and recompute $\boldsymbol{h}$. If $(\boldsymbol{\alpha}+\boldsymbol{h})$ is sign feasible for the revised θ_τ, set $\boldsymbol{\alpha}^\dagger = \boldsymbol{\alpha} + \boldsymbol{h}$ and proceed to the next stage of the algorithm.
  b) Otherwise update τ by deleting k, resetting α_k and θ_k accordingly, and recompute $\boldsymbol{h}$ for the revised problem.
- A3) Iterate until a sign feasible $\boldsymbol{\alpha}^\dagger$ is obtained.

Once sign feasibility is obtained, we can test optimality by verifying (28): calculate

$$\boldsymbol{v}^\dagger = \frac{\tilde{K}^T\tilde{\boldsymbol{r}}^\dagger}{\|\tilde{K}_\tau^T\tilde{\boldsymbol{r}}^\dagger\|_\infty} = P^T\begin{pmatrix}\boldsymbol{v}_1^\dagger \\ \boldsymbol{v}_2^\dagger\end{pmatrix}$$

where $\tilde{\boldsymbol{r}}^\dagger = \tilde{\boldsymbol{y}} - \tilde{K}\boldsymbol{\alpha}^\dagger$. By construction $(\boldsymbol{v}_1^\dagger)_i = \theta_i$ for $i \le |\tau|$, and if

$$-1 \le (\boldsymbol{v}_2^\dagger)_i \le 1 \qquad \text{for } 1 \le i \le N - |\tau|$$

then $\boldsymbol{\alpha}^\dagger$ is the desired solution. Otherwise, one proceeds as follows:

- B1) Determine the most violated condition, i.e., find the index s such that $(\boldsymbol{v}_2^\dagger)_s$ has maximal absolute value.
- B2) Update τ by adding s to it and update $\boldsymbol{\alpha}_\tau^\dagger$ by appending a zero as its last element and θ_τ by appending $\operatorname{sign}(\boldsymbol{v}_2^\dagger)_s$.
- B3) Set $\boldsymbol{\alpha} = \boldsymbol{\alpha}^\dagger$, compute a new direction $\boldsymbol{h}$ by solving (31) and iterate.

*Algorithm*1 : LASSO algorithm for robust loss functions.

A. Convergence of the Algorithm

In [6] a convergence proof is presented for the case of quadratic loss functions. This proof, however, can be easily extended to IRLS problems, as long as we guarantee that both the transformed design matrix $\tilde{K}_\tau$ and the transformed residuals $\tilde{\boldsymbol{r}}^\dagger$ remain finite as the optimization proceeds. Note, however, that this is always the case for all robust loss functions which satisfy the convergence condition (23) for IRLS updates. For the LOGREG model, the guarantee follows directly from the fact that the entries of the diagonal matrix W in (26) are bounded by one, since they are products of probabilities. The proof in [6] consists of two parts, the first of which concerns justification of steps A1)–A3). All arguments in [6] for this part are independent of the loss function, as long as it is differentiable. First, note that the current $\boldsymbol{\alpha}$ can be assumed suboptimal for the problem (31), otherwise this portion of the algorithm is skipped. This implies, however, that $\boldsymbol{h}$ is a descent direction, so that the objective function is reduced in every step, and no cycling is possible. Note that there are only finitely many possible configurations of τ, and the final $\boldsymbol{\alpha}$ must be sign feasible.

The second part of the proof concerns the case if the current $\tilde{\boldsymbol{\alpha}}^\dagger$ is *sign feasible*, but not optimal. Then, the augmented vector $(\boldsymbol{\alpha}, 0)^T$ is also suboptimal for the augmented problem (31) with τ updated by adding s, and $\boldsymbol{\theta}_\tau$ augmented to $(\boldsymbol{\theta}_\tau, \theta_s)^T$. Hence, the solution, say $(\boldsymbol{h}_\tau, h_s)^T$, of the augmented problem will be a descent direction for the augmented problem. In this part of proof, however, it remains to show that for robust loss functions primal feasibility is ensured during the steps B1)–B3). Otherwise, we could not conclude that $(\boldsymbol{h}_\tau, h_s)^T$ is also a descent direction for the original problem (26).

The key ingredient is that ℓ_1-constrained IRLS problems formally have the same Karush–Kuhn–Tucker conditions as least-squares LASSO problems: if the constraint in problem (31) is active, the KKT conditions read (cf. [33])

$$\begin{pmatrix}\tilde{K}_\tau^T\tilde{K}_\tau & \boldsymbol{\theta}_\tau \\ \boldsymbol{\theta}_\tau^T & 0\end{pmatrix}\begin{pmatrix}\boldsymbol{h}_\tau \\ \mu\end{pmatrix} = \begin{pmatrix}\tilde{K}_\tau^T(\tilde{\boldsymbol{y}} - \tilde{K}_\tau\boldsymbol{\alpha}_\tau) \\ \kappa - \boldsymbol{\theta}_\tau^T\boldsymbol{\alpha}_\tau\end{pmatrix}. \tag{32}$$

Equation (32) implies $\mu\boldsymbol{\theta}_\tau = \tilde{K}_\tau^T\tilde{\boldsymbol{r}}^\dagger$ and $\mu = \|\tilde{K}_\tau^T\tilde{\boldsymbol{r}}^\dagger\|_\infty$. With these two identities, we can now simply follow the further proof

[6]For the algorithm in [6], it is necessary to center the design matrix K and to exclude the intercept α_0 from the ℓ_1-penalty.

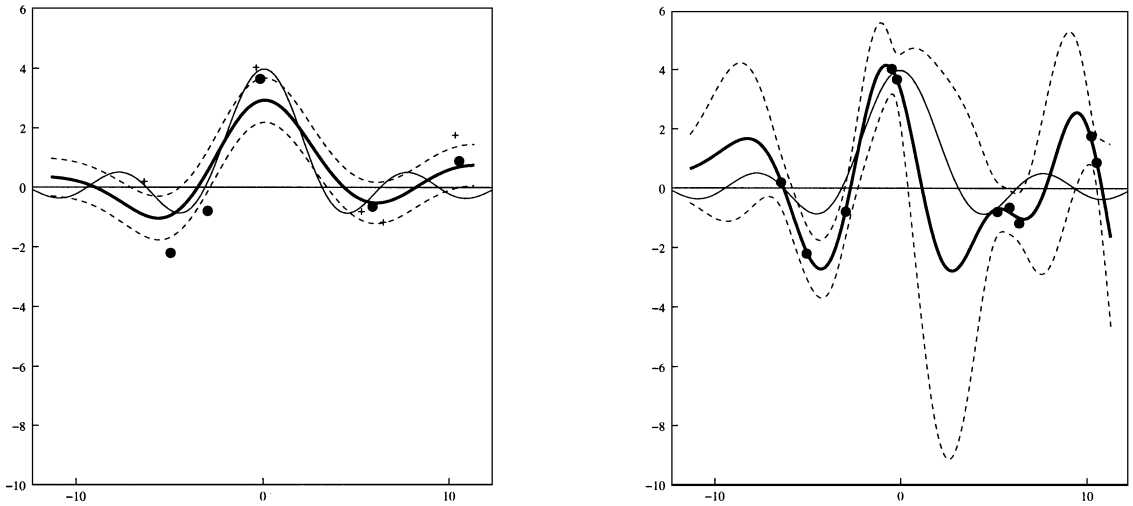


Fig. 1. Fitting the noisy sinc function. Left: best LASSO fit with quadratic loss function (bold curve). The 1-standard-deviation confidence interval is depicted by the dashed lines around the fit. Right: overfitting effects due to a too complex model.

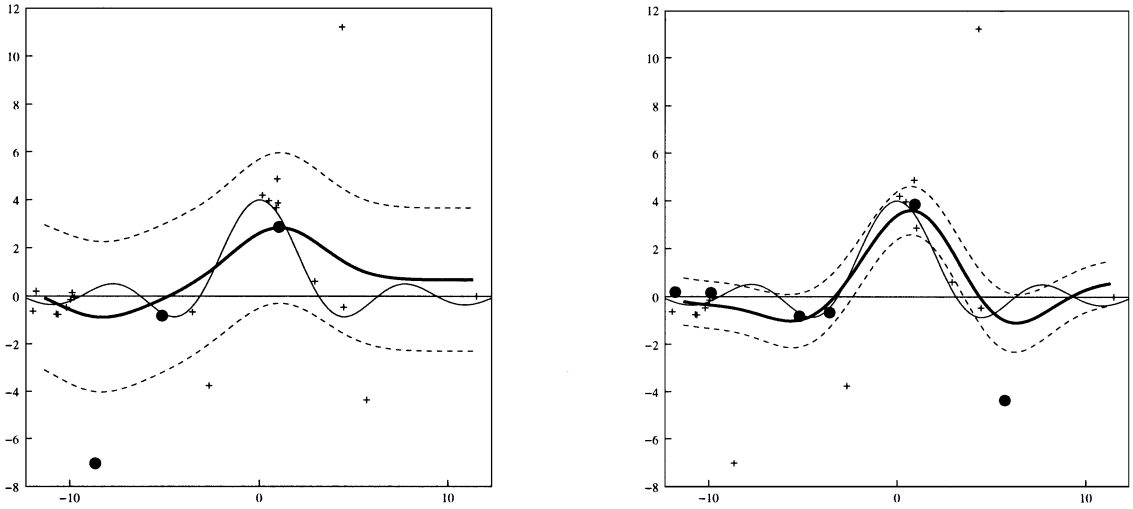


Fig. 2. Fitting the noisy sinc function (the thin solid curve) with 30% outliers. Left: best LASSO fit with quadratic loss function (the bold solid curve), and estimated error bars (the dashed curves around the fit). Right: same situation for Huber's robust loss function.

given in [6]: since $(\boldsymbol{h}_\tau,\, h_s)^T$ is a descent direction for the augmented problem, we have

$$\begin{aligned} 0 > &-(\tilde{\boldsymbol{r}}^\dagger)^T(\tilde{K}_\tau,\, \boldsymbol{\phi}_s)(\boldsymbol{h}_\tau,\, h_s)^T \\ &= -(\tilde{\boldsymbol{r}}^\dagger)^T\tilde{K}_\tau\boldsymbol{h}_\tau - (\tilde{\boldsymbol{r}}^\dagger)^T\boldsymbol{\phi}_s h_s = -\mu(\boldsymbol{\theta}_\tau^T\boldsymbol{h}_\tau + (v_2^\dagger)_s h_s). \end{aligned}$$

On the other hand, feasibility for (31) requires that $\boldsymbol{\theta}_\tau^T\boldsymbol{h}_\tau + \theta_s h_s \leq 0$. Multiplying by μ and adding the former inequality yields $0 > \mu(\theta_s - (v_2^\dagger)_s)h_s$. Now since $|(v_2^\dagger)_s| > 1$, choosing $\theta_s = \operatorname{sign}((v_2^\dagger)_s)$ yields $\operatorname{sign}(h_s) = \theta_s$. This implies that the linearized constraint for the augmented problem is equivalent to the norm constraint for small enough displacements in the direction $(\boldsymbol{h}_\tau,\, h_s)^T$. This in turn ensures that primal feasibility is maintained during the algorithm.

B. Variance Estimates

In [17] it has been shown that AdR regression and LASSO are identical in the sense that for every constraint value κ in (26) there exists a parameter λ in (12) such that the solutions of both models coincide. Concerning both technical issues of optimization and analytical variance estimates, however, there are pronounced differences: for the LASSO there exists the highly efficient subset selection algorithm presented in the last section, which to our knowledge is not applicable to AdR. Concerning analytic variance estimates, however, the situation is different: due to the non-Gaussian shape of the LASSO prior, there is no closed-form solution for the prediction variances. Having solved the AdR update (14) for the hyperparameters ϑ_i, on the other hand, we can simply use the standard formula (9). Our goal is, thus, to first solve the LASSO problem and then to find a way to identify the corresponding AdR functional that would have produced the same solution. Variance estimates can then be obtained within the standard Bayesian setting.

In order to find the corresponding AdR functional it is necessary to shed light on the exact relationship between both methods. Once the iterations (14) have converged, we can resubstitute the optimal hyperparameters ϑ_i into the functional (7):

$$M = \|\boldsymbol{y} - K\boldsymbol{\alpha}\|_2^2 + \lambda(N+1)^{-1}\|\boldsymbol{\alpha}\|_1. \tag{33}$$

Interpreting λ as a Lagrange parameter, this functional corresponds to a constrained minimization problem

$$\text{minimize } \|\boldsymbol{y} - K\boldsymbol{\alpha}\|_2^2 \qquad \text{s.t. } (N+1)^{-1}\|\boldsymbol{\alpha}\|_1 < \kappa'. \quad (34)$$

Defining $\kappa := \sqrt{\kappa'(N+1)}$ we arrive at the ℓ_1-constrained LASSO problem (15), the corresponding functional of which is given in (17). We notice that the Lagrange parameters in both problems are connected in the following way.

$$\lambda = (N+1)\tilde{\lambda}/\|\overline{\boldsymbol{\alpha}}\|_1 = (N+1)\tilde{\lambda}\,\kappa. \quad (35)$$

Note that this identity holds for any differentiable loss function. In order to compute variance estimates of LASSO predictions, (35) can be exploited as summarized in $Algorithm 2$.

1) Given the LASSO constraint κ (selected according to some model selection criterion), run the subset selection algorithm for the generalized LASSO.
2) Determine the Lagrange parameter $\tilde{\lambda}$ by (30), and in turn calculate λ of the corresponding AdR functional by (35).
3) Compute the parameters $\boldsymbol{\vartheta}$ of the ARD prior by evaluating (14).
4) Estimate the variance of the Gaussian noise in the transformed least-squares problem by the empirical deviations of the fitted values from the transformed targets.
5) Given a new observation, evaluate (24) and add the estimated noise variance in order to obtain the total predictive variance.

Algorithm 2: estimating the predictive variance.

The fact that we can analytically compute λ by (35) constitutes the main advantage over the very similar variance estimation procedure proposed in [4]: in the latter approach it was suggested to stepwise adjust λ by iteratively solving a ridge-regression problem. For IRLS problems with additional sub-iterations, in particular, this can be a very time-consuming process.

One concluding remark concerning the relation of the above procedure to variance estimates in the RVM algorithm: in the RVM framework the noise σ is estimated by maximizing the marginal likelihood, whereas step 4 here can be considered somewhat more heuristic.

V. Experiments

A. Toy Examples: Noisy Sinc-Function

In the first experiment, we consider the problem of fitting the noisy sinc-function. In Fig. 1, we have sampled 11 points from the sinc-function, and we have added Gaussian noise with fixed variance $\sigma^2 = 0.2$. In accordance with the Gaussian noise distribution, the standard LASSO model with quadratic loss function is chosen. In all experiments of this kind RBF kernels are employed. The variance is estimated as proposed in Algorithm 2: in step 1, the constraint $\kappa = 4.5$ turned out to be optimal for predicting the function values of 1000 test points, measured in terms of the mean squared error with respect to the original sinc-function. In this special case, knowing the true noise variance made it possible to skip step 4 in the variance estimation procedure, and to use the known value $\sigma^2 = 0.2$ in step 5. Taking into account that the learning set is extremely small for the given task, both the fit and the one-standard-deviation confidence interval plotted in the left graph seem to be reasonable. In the right graph it is demonstrated, how the choice of a far too complex model affects the variance of the estimates. In this severe overfitting situation ($\kappa = 500$), the uncertainty region becomes huge in areas where we have little data evidence. The predictive information becomes vanishingly small with increasing model complexity.

TABLE I
Results for Friedman's Benchmark Functions. Mean Prediction Error Averaged Over 100 Randomly Generated 240/1000 Training/Test Splits, and Number of Support/Relevance Vectors. The SVM/RVM Results are Taken From [37]

problem	SVM	RVM	LASSO (quadratic loss)
#1	2.92 / 116.6	2.80 / 59.4	2.84 / 73.5
#2	4140 / 110.3	3505 / 6.9	3808 / 14.2
#3	0.0202 / 106.5	0.0164 / 11.5	0.0192 / 16.4

In the following two experiments we concentrate on situations, in which the learning set is additionally corrupted by 30% outliers, drawn from a uniform distribution on $[-10, 10]$. The optimal LASSO fit employing a quadratic loss function is depicted in the left graph of Fig. 2 (optimality is again measured on a test set of size 1000). The quadratic shape of the loss function imposes the problem that the distant outliers gain disproportionate influence on the solution. The only possible workaround is to restrict the model complexity by highly regularizing the problem ($\kappa = 1.8$). This, however, leads to a clear *underfitting* situation with respect to the original sinc-function. The estimated function is readily expanded in terms of only three relevance vectors, i.e., all but three variables are removed from the model. Concerning the variance estimates, modeling the noise distribution as a Gaussian introduces a second problem: the empirically estimated noise variance (step 4) becomes very high, and the error bars on the predictions are so large that even the sign of the predicted function values appears unreliable.

Both problems can be overcome by employing a robust loss function of Huber's type: penalizing distant outliers only linearly allows us to choose a model of adequate complexity while simultaneously reducing the outlier's influence. Concerning the prediction accuracy, the robust LASSO fit depicted in the right graph outperforms its quadratic counterpart significantly (the mean squared error reduces from 0.31 to 0.17). Moreover, implicit with the use of a robust loss function is the interpretation of data points far away from the regression fit as "untypical" events. Compared to standard least-squares functionals, the influence of these untypical events is reduced by assigning smaller "weights" Ω_{jj} to them in the functional (26). The resulting variance estimates depicted by the dashed lines appear much more adequate for the given problem.

B. Friedman's Benchmark Functions

In this section, the main focus concerns a comparison of the *prediction performance* of LASSO regression relative to both the SVM and the RVM for Friedman's benchmark functions.

TABLE II
RESULTS FOR THE "HOUSE-PRICE-8L," "BANK-32-FH" AND "ABALONE" DATASETS FROM THE DELVE REPOSITORY. IN THE SECOND ROW, R STANDS FOR RVM, S FOR SVM AND L FOR LASSO. THE TIMES ARE MEASURED ON A 500 MHz PC. COLUMNS 2–4 SHOW THE MEAN SQUARED PREDICTION ERROR ON A TEST SET OF SIZE 4000 (FOR "ABALONE" SIZE 2000), THE LAST 3 COLUMNS SHOW THE TIME IN SECONDS FOR THIS PREDICTION

sample	MSE			# SV/RV			t_{learn}[s]			t_{test}[s]		
size	R	S	L	R	S	L	R	S	L	R	S	L
	($\cdot 10^3$)			**house-price-8L**								
1000	1099	1062	1075	33	597	61	$4.2 \cdot 10^3$	33	26	0.11	1.8	0.20
2000	1048	1022	1054	36	1307	63	$3.5 \cdot 10^4$	101	72	0.3	3.9	0.21
4000	-	1012	1024	-	2592	69	-	428	312	-	7.8	0.23
	($\cdot 10^{-3}$)			**bank-32-fh**								
2000	7.41	7.82	7.39	14	1638	22	$3 \cdot 10^4$	15	24	0.05	6.0	0.08
4000	-	7.75	7.49	-	3402	23	-	83	102	-	12.7	0.08
				abalone								
2000	4.35	4.37	4.35	12	972	19	$3 \cdot 10^4$	14	22	0.02	1.3	0.03
				abalone + 30% outliers								
2000	6.78	4.59	4.58	10	1335	14	$3 \cdot 10^4$	17	38	0.01	1.8	0.02

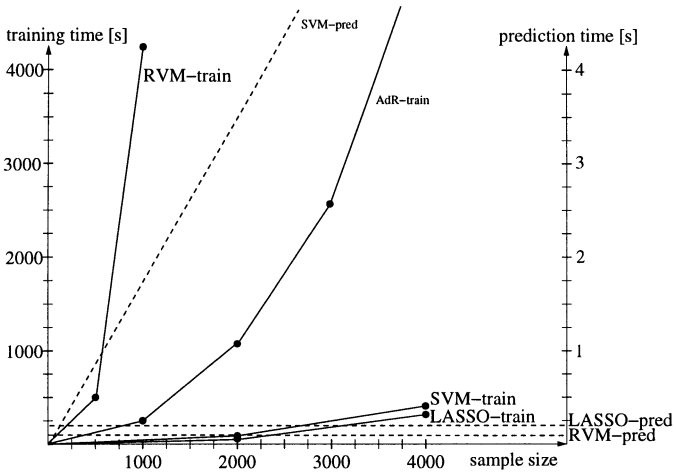


Fig. 3. Computation times for the "house-8L" dataset. Solid lines depict training times, dashed lines depict prediction times for a test set of size 4000.

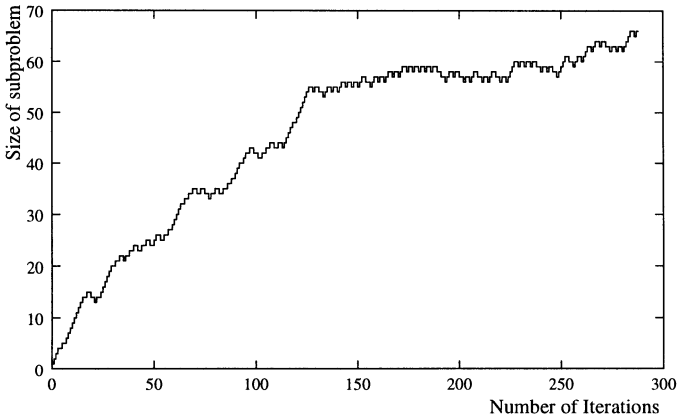


Fig. 4. Size of the LASSO-subproblems versus iteration number for the "house-8L" dataset.

They have been introduced in [34] and have become a widely used benchmark for regression models (see, e.g., [35] and [36]).

Friedman #1 is a nonlinear prediction problem with ten independent variables that are uniformly distributed in $[0, 1]$

$$\#1: y = 10\sin(\pi x_1 x_2) + 20\,(x_3 - 1/2)^2 + 10\,x_4 + 5\,x_5 + \nu$$

where $\nu \sim \mathcal{N}(0, 1)$ is normal distributed noise. The function, however, depends only on five variables, and the predictor has to distinguish the variables that have no prediction ability (x_6 to x_{10}) from the others.

Friedman #2 and #3 both have four independent variables that are uniformly distributed in the following ranges

$$0 \le x_1 \le 100, \qquad 40\pi \le x_2 \le 560\pi, \qquad 0 \le x_3 \le 1$$
$$1 \le x_4 \le 11.$$

Function #2 and #3 are defined as

$$\#2: \quad y = \sqrt{x_1^2 + [x_2 x_3 - (x_2 x_4)^{-1}]^2} + \nu$$
$$\#3: \quad y = \tan^{-1}\left(x_1^{-1}\left(x_2 x_3 - (x_2 x_4)^{-1}\right)\right) + \nu$$

where the noise is adjusted to give 3 : 1 ratio of signal power to noise power.

The prediction results for the three functions are summarized in Table I. The results are averaged over 100 randomly generated training sets of size 240 and test sets of size 1000. The performance is measured in terms of mean squared error with respect to the original functions before the noise was added. Since only relatively small learning sets are considered, we postpone a detailed analysis of *computational costs* to the real-world experiments in the next section.

TABLE III
CLASSIFICATION RESULTS FOR UCI BENCHMARK DATASETS. NUMBER OF SVS/RVS AND PREDICTION ERRORS, AVERAGED OVER 100 TRAINING/TEST SPLITS (STANDARD ERRORS IN BRACKETS)

	SVM		LOGREG-LASSO	
Dataset	# SV	error [%]	# RV	error [%]
Banana	97.2	11.8 (0.8)	21.8	10.7 (0.5)
Breast-cancer	82.7	26.0 (4.7)	17.3	26.1 (4.6)
Diabetes	238.7	24.1 (1.8)	7.6	23.5 (1.9)
Heart	62.1	16.1 (3.1)	10.7	16.0 (3.1)
Flaire-solar	493.8	33.1 (1.8)	6.4	33.3 (1.6)
German	415.3	23.7 (2.2)	22.3	23.6 (2.3)
Ringnorm	149.5	1.7 (0.1)	12.3	1.8 (0.3)
Thyroid	18.7	4.9 (2.2)	6.3	4.8 (2.3)
Titanic	70.8	22.7 (1.3)	5.0	22.9 (1.2)
Twonorm	72.4	2.9 (0.2)	8.7	2.6 (0.2)

It should be noticed that all three models attain a very similar level of accuracy. The prediction differences can be readily explained by the statistical fluctuations when randomly generating the training/test data. Distinct differences, however, occur in the number of support/relevance vectors: the models employing ARD priors produce much sparser solutions than the SVM, in accordance with our theoretical considerations and with the above results from the toy examples.

C. Real-World Regression Problems

As real-world examples, we present results for the "house-price-8L," "bank-32-fh," and "abalone" datasets from the DELVE benchmark repository.[7] We compared both the prediction accuracy and the computational costs of RVM, SVM,[8] and LASSO for different sample sizes. The results are summarized in Table II. In all experiments RBF kernels are used, and the inputs are standardized to have unit variance. The model parameters (width of RBF kernel and regularization parameter) are selected by minimizing the prediction error on a randomly chosen set of 1000 test examples. The final prediction results are reported on a test set of size 4000.

The LASSO results for the first three experiments refer to a quadratic loss function. It is worth noticing that within the *whole* DELVE archive for regression benchmarks we could not find a single problem for which a robust loss function significantly improved the accuracy. This, however, only means that for the task of discriminating between robust and nonrobust regression models, these benchmark datasets are too "well-behaved" in the sense that they obviously contain no or only very few outliers. In order to present at least one highly disturbed large-scale problem, we artificially corrupted 30% of the training patterns from the abalone dataset by uniform noise on the interval $[-200, +200]$. For this last experiment, we used a robust LASSO estimator employing a loss function of Huber's type.

From Table II we conclude, that

- For the original benchmark datasets, the prediction accuracy of all models is comparable. After adding outliers, the robust models outperform the RVM significantly.
- The ARD models are sparser than the SVM by 1–2 orders of magnitude.
- The RVM has severe computational problems for large training sets.
- The LASSO combines the advantages of efficiently handling large training sets and producing extremely sparse solutions.

Concerning the training times, the reader should notice that we are comparing the highly tuned *SVMTorch* optimization package, [38], with our straight forward LASSO implementation, which we consider to yet possess ample opportunities for further optimization.

Fig. 3 shows a schematic plot of the computation times for the "house-8L" dataset. Solid lines depict training times, dashed lines depict prediction times for a test set of size 4000. In the training phase, both SVM and LASSO are clearly superior to the RVM. The curve labeled "AdR-train" depicts the training times for a AdR model with conjugate gradient approximation, cf. Section IV. Note that the costs for the exact AdR algorithm are almost identical to those of the RVM, since in both cases a $(N+1) \times (N+1)$ matrix must be inverted, see (11) and (13).

For the purpose of efficiently predicting function values of new observation, however, the ARD models outperform the SVM significantly. Note that the prediction time solely depends on the number of nonzero expansion coefficients, which for ARD models roughly remains constant with increasing size of the training set.

In a last regression experiment, we have plotted the size of the subproblems (31) during the iterations for the "house-8L" dataset with a learning set of size 2000. The "smooth" curve in Fig. 4 shows that the optimal variable set is found without solving unnecessarily large subproblems in intermediate states of the iteration.

D. Classification Experiments

We conclude the experiments section with several benchmark classification problems. Our main focus here concerns a comparison of the LOGREG version of LASSO with the SVM. We have chosen several UCI benchmark datasets,[9] since they have been widely used for benchmarking in the literature. Table III reports prediction errors and number of support/relevance vectors, averaged over 100 training/test splits. In each experiment the model parameters (width of RBF kernel and regularization constant) are selected by minimizing the averaged test error on five randomly chosen splits. We conclude, that the LOGREG-LASSO model attains a level of prediction accuracy comparable with a "state-of-the-art" SVM, while using significantly less vectors in the kernel expansion.

[7]The datasets are available via http://www.cs.toronto.edu/~delve/delve.html.

[8]We used the *SVMTorch* V 3.07 implementation (see [38]).

[9]Available at http://www.ics.uci.edu/~mlearn/MLSummary.html (except BANANA). We used the normalized versions of these datasets from http://www.first.gmd.de/~raetsch/.

VI. Discussion

Sparsity is an important feature of kernel regression models, since it simultaneously allows us to efficiently learn a regression function and to efficiently predict function values of new examples. For the SVM, highly tuned training algorithms have been developed during the last years. However, the SVM approach still suffers from the steep growth of the number of *support vectors* with increasing size of the training set. From a more conceptual viewpoint, the difficulty of analytically deriving error bars may be considered an even more severe shortcoming.

The key idea to overcome these drawbacks is to apply the Bayesian methodology to kernel regression models. In combination with the mechanism of ARD, interpretable and extremely sparse solutions can be obtained. The RVM has been introduced in [7] as a first method of this kind. For the RVM (at least in its original version), however, the probabilistic modeling of the data has introduced two new problems: since the update equation for the hyperparameters (10) requires us to explicitly invert a large matrix, the learning phase has become very costly. It is, thus, difficult to apply the RVM to large-scale real-world problems. A second shortcoming results from the prior assumptions about the noise by which the data are corrupted. In order to make probabilistic interpretations possible, a simple Gaussian noise model is assumed. Implicit with such a Gaussian noise model is measuring the approximation accuracy by a quadratic loss function. However, if the noise assumption is violated in the sense that the learning set contains a considerable amount of outliers, these outliers will exhibit a disproportionate influence on the estimated regression function. It should be noticed, however, that recent work on the RVM addresses these shortcomings (see [9] and [10]).

In this paper we present a different type of kernel regressors which overcome the above problems. Given the identical ARD prior model, one can analytically integrate out the hyperparameters by defining exponential hyperpriors. This leads us to the LASSO method which optimizes a ℓ_1-constrained regression functional. Reformulating the relevance determination problem in the LASSO framework makes it possible to use a highly efficient subset algorithm introduced in [6].

We have shown that this algorithm can be generalized to the class of IRLS models, including *robust regression models* and *logistic regression*. The use of robust loss functions which penalize distant outliers only linearly constitutes a significant advantage for handling highly corrupted learning sets. We further have proposed an efficient method for deriving variance estimates for this class of robust kernel regressors. The key idea here is that we can analytically derive the Lagrange parameter associated with the LASSO problem. This parameter allows us to use the standard Bayesian variance estimates in transformed least-squares problems. The fact that the LOGREG model is also part of the IRLS class, naturally extends the robust regression framework with ℓ_1 penalty to the case of binary targets in classification.

Experiments for both synthetic and real-world benchmark datasets have effectively demonstrated that the generalized LASSO combines several important features.

- The user is provided with *probabilistic outputs*, even if robust loss functions are employed. This makes it possible to estimate a confidence region around the fitted values.
- We can identify vectors in the training set that are considered *relevant*, in the sense that new observations must only be pairwise compared via the kernel function with these "relevance" vectors.
- Both robust regression models and LOGREG classifiers can be optimized using the same subset algorithm.
- Concerning the computational costs during the learning phase, this algorithm performs comparable to highly tuned "state-of-the-art" SVM optimization packages for several benchmark problems. Moreover, we strongly believe that implementations of this algorithm possess ample opportunities for further optimization, and might undergo a development similar to that of the SVM algorithms during the last couple of years.
- Concerning the time needed for making predictions, the LASSO outperforms the SVM drastically, due to the extreme sparsity of its solutions. This may be of particular interest for prediction tasks with real-time requirements.

Acknowledgment

The author would like to thank M. Braun, J. Buhmann, and V. Steinhage for helpful discussions.

References

[1] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf, "An introduction to kernel-based learning algorithms," *IEEE Trans. Neural Networks*, vol. 12, pp. 181–201, Mar. 2001.

[2] A. Hoerl and R. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, pp. 55–67, 1970.

[3] J. Gao, S. Gunn, C. Harris, and M. Brown, "A probabilistic framework for svm regression and error bar estimation," *Machine Learning*, vol. 46, no. 1, pp. 71–89, 2002.

[4] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Roy. Statist. Soc.*, vol. B 58, no. 1, pp. 267–288, 1996.

[5] P. Huber, *Robust Statist.*. New York: Wiley, 1981.

[6] M. Osborne, B. Presnell, and B. Turlach, "On the LASSO and its dual," *J. Comput. Graphical Statist.*, vol. 9, pp. 319–337, 2000.

[7] M. Tipping, "Sparse bayesian learning and the relevance vector machine," *J. Machine Learning Res.*, vol. 1, pp. 211–244, 2001.

[8] D. MacKay, "Bayesian nonlinear modeling for the prediction competition," in *ASHRAE Trans.*, vol. 100, Atlanta, GA, 1994, pp. 1053–1062.

[9] A. Faul and M. Tipping, "Analysis of sparse bayesian learning," in *Advances in Neural Information Processing Systems 14*, T. Dietterich, S. Becker, and Z. Ghahramani, Eds. Cambridge, MA: MIT Press, 2002, pp. 383–389.

[10] ——, "A variational approach to robust regression," in *Artificial Neural Networks—ICANN'01*, G. Dorffner, H. Bischof, and K. Hornik, Eds., 2001, pp. 95–102.

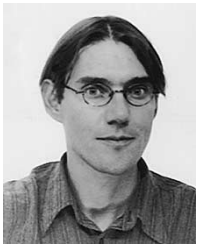
[11] A. Navia-Vázquez, F. Pérez-Cruz, A. Artés-Rodríguez, and A. Figueiras-Vidal, "Weighted least squares training of support vector classifiers leading to compact and adaptive schemes," IEEE Trans. Neural Networks, vol. 12, pp. 1047–1059, Sept. 2001, to be published.

[12] J. Suykens, J. De Brabanter, J. Lukas, and L. Vandewalle, "Weighted least squares support vector machines: Robustness and sparse approximation," *Neurocomputing: Special Issue on Fundamental and Information Processing Aspects of Neurocomputing*, vol. 48, no. 1–4, pp. 85–105, 2002.

[13] J. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle, Least Squares Support Vector Machines. Singapore: World Scientific, 2002, to be published.

[14] V. Roth and V. Steinhage, "Nonlinear discriminant analysis using kernel functions," in *Advances in Neural Information Processing Systems 12*, S. Solla, T. Leen, and K.-R. Müller, Eds. Cambridge, MA: MIT Press, 1999, pp. 568–574.

[15] D. Denison and E. George, "Bayesian prediction using adaptive ridge estimators," Dept. Math., Imperial College, London, U.K., Tech. Rep., 2000.
[16] M. Figueiredo and A. Jain, "Bayesian learning of sparse classifiers," in *Proc. Computer Vision and Pattern Recognition*, 2001, pp. 35–41.
[17] Y. Grandvalet, "Least absolute shrinkage is equivalent to quadratic penalization," in *Perspectives in Neural Computing*, L. Niklasson, M. Bodén, and T. Ziemske, Eds. New York: Springer-Verlag, 1998, pp. 201–206.
[18] S. Chen, D. Donoho, and M. Saunders, "Atomic decomposition by basis pursuit," *SIAM J. Sci. Comput.*, vol. 20, no. 1, pp. 33–61, 1999.
[19] J. Lokhorst, *The LASSO and Generalized Linear Models*. Sydney, Australia: Univ. Adelaide, 1999.
[20] C. Williams, "Prediction with Gaussian processes: From linear regression to linear prediction and beyond," in *Learning and Inference in Graphical Models*, M. Jordan, Ed. Boston, MA: Kluwer, 1998.
[21] Y. Grandvalet, "Anisotropic noise injection for input variables relevance determination," *IEEE Trans. Neural Networks*, vol. 11, pp. 1201–1212, Nov. 2000.
[22] J. Fessler, "Grouped coordinate descent algorithms for robust edge-preserving image restoration," in *Proc. Int. Soc. Opt. Eng. Image Reconstruction and Restoration II*, vol. 3170, T. Schulz, Ed., 1997, pp. 184–194.
[23] I. Nabney, "Efficient training of RBF networks for classification," Neural Computing Research Group, Aston Univ., Birmingham, U.K., Tech. Rep. NCRG/99/002, 1999.
[24] V. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.
[25] M. Pontil, S. Mukherjee, and F. Girosi, "On the noise model of support vector machine regression," Artificial Intell. Lab., Mass. Inst. Technol., Cambridge, MA, Tech. Rep. A.I. Memo 1651, 1998.
[26] A. Smola and B. Schölkopf, "A tutorial on support vector regression," Royal Holloway College, Univ. London, London, U.K., Tech. Rep. NeuroCOLT2, NC2-TR-1998-030, 1998.
[27] S. Gunn, "Supanova: A sparse, transparent modeling approach," presented at the IEEE Int. Workshop Neural Networks for Signal Processing, Madison, WI, 1999.
[28] J. Weston, A. Gammerman, M. Stitson, V. Vapnik, V. Vovk, and C. Watkins, "Support vector density estimation," in *Advances in Kernel Methods—Support Vector Learning*, B. Schölkopf, C. Burges, and A. Smola, Eds. Cambridge, MA: MIT Press, 1999.
[29] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes in C*. Cambridge, U.K.: Cambridge Univ. Press, 1992.
[30] M. Osborne, B. Presnell, and B. Turlach, "A new approach to variable selection in least squares problems," *IMA J. Numer. Anal.*, vol. 20, no. 3, pp. 389–404, July 2000.
[31] D. Clark and M. Osborne, "On linear restricted and interval least-squares problems," *IMA J. Numer. Anal.*, vol. 8, pp. 23–36, 1988.
[32] M. Bazaraa, H. Sherali, and C. Shetty, *Nonlinear Programming: Theory and Algorithms*. New York: Wiley, 1993.
[33] S. G. Nash and A. Sofer, *Linear and Nonlinear Programming*. New York: McGraw-Hill, 1996.
[34] J. Friedman, "Multivariate adaptive regression splines," *Ann. Statist.*, vol. 19, no. 1, pp. 1–82, 1991.
[35] H. Drucker, "Improving regressors using boosting techniques," in *Proc. Int. Conf. Machine Learning*, J. D. H. Fisher, Ed.. San Mateo, CA, 1997, pp. 107–113.
[36] S. Saunders, A. Gammermann, and V. Vovk, "Ridge regression learning algorithm in dual variables," Royal Holloway College, Univ. London, London, U.K., Tech. Rep., 1998.
[37] M. Tipping, "The relevance vector machine," in *Advances in Neural Information Processing Systems 12*, S. Solla, T. Leen, and K.-R. Müller, Eds. Cambridge, MA: MIT Press, 1999, pp. 652–658.
[38] R. Collobert and S. Bengio, "Support vector machines for large-scale regression problems," Institut Dalle Molle d'Intelligence Artificelle Perceptive (IDIAP), Martigny, Switzerland, Tech. Rep. IDIAP-RR-00-17, 2000.



Volker Roth was born in Klagenfurt, Austria, in 1970. He received the diploma degree in physics in 1997 and the Ph.D. degree in computer science in 2001, both from the University of Bonn, Bonn, Germany.

Currently, he is a Postdoctoral Researcher with the Computer Vision and Pattern Recognition Group, University of Bonn. His research interests include Support Vector Machines and kernel-based learning algorithms, unsupervised learning and clustering, bioinformatics, and computational biology.