Robust MLP *

Tommi Kärkkäinen and Erkki Heikkola
Department of Mathematical Information Technology
University of Jyväskylä, P.O. Box 35 (Agora)
FIN-40014 University of Jyväskylä, Finland
E-mail: Tommi.Karkkainen@mit.jyu.fi,Erkki.Heikkola@mit.jyu.fi

Abstract

The connection between robust statistical procedures and nonsmooth optimization is established. Based on the resulting family of optimization problems, robust learning problem formulations with regularization-based control on the model complexity of the MLP-network are described and analyzed. Numerical experiments for simulated regression problems are conducted and new strategies for determining the regularization coefficient are proposed and evaluated.

1 Introduction

A multilayered perceptron (MLP) is the most commonly used neural network for nonlinear regression approximation. The simplest model of data in regression is to assume that the given targets are generated by

$$\mathbf{y}_i = \phi(\mathbf{x}_i) + \boldsymbol{\varepsilon}_i, \tag{1.1}$$

where $\phi(\mathbf{x})$ is the unknown stationary function and $\boldsymbol{\varepsilon}_i$'s are sampled from an underlying noise process. In [Kärkkäinen, 2002] it was proved that for a special architecture and regularization (pruning) of MLP the usual least-mean-squares learning problem formulation corresponds to the Gaussian assumption for the noise statistics. In statistics, relaxation of this assumption underlies the so-called robust procedures (e.g., [Huber, 1981, Rousseeuw and Leroy, 1987, Rao, 1988, Hettmansperger and McKean, 1998, Oja, 1999]).

In neural networks literature there have been some attempts to combine robust statistical procedures with learning problem formulations and training algorithms mainly for MLP-networks (e.g., [Kosko, 1992, Chen and Jain, 1994, Liano, 1996]). However, thorough understanding of such combinations together with a link to model complexity through regularization based pruning has, as far as we know, not been considered on a solid basis. This is the goal of the present work.

*This work was financially supported by the Academy of Finland, grant 49006, and by the InBCT-project of Tekes.

The main emphasis here is to describe, analyze, and test robust learning problem formulations for the MLP in a batch mode. For the numerical comparisons, we need to utilize black box training algorithms for solving the optimization problems which are based on these formulations. A key concept then is the convergence of an algorithm which depends on the regularity of the optimization problem [Nocedal and Wright, 1999]. Hence, rigorous treatment of robust MLP requires us to establish a link between the norms behind the robust statistics and the regularity of such problems [Clarke, 1983, Mäkelä and Neittaanmäki, 1992]. As far as we know this fundamental relation has not been explicitly established in other works.

Another basis for the present work is to treat the MLP-transformation in a layer-wise form ([Hagan and Menhaj, 1994, Kärkkäinen, 2002]). This allows us to derive the optimality system in a compact form, which can be utilized in an efficient computer implementation of the proposed techniques. Clear and explicit form of the optimality conditions enable the derivation of some consequences and interpretations concerning the final structure of a trained network, which readily explain and predict the behaviour of MLP. Together with the given new heuristics for controlling the model complexity the proposed approach allows a rigorous derivation of an MLP for real applications with different noise characteristics within the training data.

The contents of the work are the following: First, in Section 2 we establish, discuss and illustrate the connection between robust statistics and nonsmooth optimization. There, we also present the layer-wise architecture and family of learning problem formulations for training an MLP. In Section 3, we compute the optimality conditions for the network learning and derive and discuss some of their consequences. In Section 4, we report results of numerical experiments for comparing different formulations and introduce two novel techniques for determining the complexity of an MLP model. Finally, in Section 5 we briefly make some conclusions.

2 Preliminaries

Throughout the paper, we denote by $(\mathbf{v})_i$ the ith component of a vector $\mathbf{v} \in \mathbb{R}^n$. Without parenthesis, $\mathbf{v}_i$ represents one element in the set of vectors $\{\mathbf{v}_i\}$. The l_q-norm of a vector $\mathbf{v}$ is given by

$$\|\mathbf{v}\|_q = \left(\sum_{i=1}^{n} |(\mathbf{v})_i|^q \right)^{1/q}, \quad q < \infty. \tag{2.1}$$

2.1 Nonsmooth optimization and robust statistics

In this section, we establish the connection between nonsmooth optimization and robust statistics. More details and further references on nonsmooth optimization can be found, e.g., in [Clarke, 1983, Mäkelä and Neittaanmäki, 1992], while robust statistics is treated in [Huber, 1981, Rousseeuw and Leroy, 1987, Rao, 1988, Hettmansperger and McKean, 1998, Oja, 1999].

Nonsmooth optimization is a field of mathematics concentrating on functionals and optimization problems, which can not be described by using the classical (C^1) differential calculus. We consider the following unconstrained optimization problem

$$\min_{\mathbf{u}\in\mathbb{R}^n} \mathcal{J}(\mathbf{u}), \tag{2.2}$$

where $\mathcal{J} : \mathbb{R}^n \to \mathbb{R}$ is a given cost function.

Definition 2.1. A function $\mathcal{J} : \mathbb{R}^n \to \mathbb{R}$ is locally Lipschitz continuous at $\mathbf{u}^* \in \mathbb{R}^n$, if there exist $K > 0$ and $\delta > 0$ such that

$$|\mathcal{J}(\mathbf{u}) - \mathcal{J}(\mathbf{v})| \leq K \|\mathbf{u} - \mathbf{v}\|_2 \quad \text{for all } \mathbf{u}, \mathbf{v} \in B(\mathbf{u}^*, \delta), \tag{2.3}$$

where $B(\mathbf{u}, \delta) = \{\mathbf{v} \in \mathbb{R}^n : \|\mathbf{v} - \mathbf{u}\|_2 < \delta\}$.

Definition 2.2. Let $\mathcal{J}$ be locally Lipschitz continuous. The subdifferential $\partial\mathcal{J}$ (according to [Clarke, 1983]) of $\mathcal{J}$ at $\mathbf{u} \in \mathbb{R}^n$ is defined by

$$\partial\mathcal{J}(\mathbf{u}) = \{\boldsymbol{\xi} \in \mathbb{R}^n \mid \mathcal{J}^0(\mathbf{u}; \mathbf{d}) \geq \boldsymbol{\xi}^T\mathbf{d} \quad \forall \mathbf{d} \in \mathbb{R}^n\}, \tag{2.4}$$

where $\mathcal{J}^0(\mathbf{u}; \mathbf{d})$ is the generalized directional derivative

$$\mathcal{J}^0(\mathbf{u}; \mathbf{d}) = \limsup_{\substack{\mathbf{v}\to\mathbf{u} \\ \mathbf{t}\searrow 0}} \frac{\mathcal{J}(\mathbf{v} + t\mathbf{d})}{t}, \tag{2.5}$$

which coincides with the usual directional derivative $\mathcal{J}'(\mathbf{u}; \mathbf{d})$ when it exists. An equivalent characterization of the nonempty, convex, and compact set $\partial\mathcal{J}(\mathbf{u})$ is given by

$$\begin{aligned} \partial\mathcal{J}(\mathbf{u}) = \operatorname{conv}\{\boldsymbol{\xi} \in \mathbb{R}^n \mid \forall\{\mathbf{u}_i\} \subset \mathbb{R}^n \text{ such that } \mathbf{u}_i \to \mathbf{u} \\ \text{and } \exists\, \nabla\mathcal{J}(\mathbf{u}_i), \nabla\mathcal{J}(\mathbf{u}_i) \to \boldsymbol{\xi}\}, \end{aligned} \tag{2.6}$$

where the convex hull of set S, $\operatorname{conv}(S)$, is the smallest convex set containing the set S. Element $\boldsymbol{\xi} \in \partial\mathcal{J}(\mathbf{u})$ is called a subgradient.

Definition 2.3. Let $\mathcal{J}$ be locally Lipschitz continuous. Point $\mathbf{u}^* \in \mathbb{R}^n$ is called a substationary point of the minimization problem (2.2) if

$$\mathbf{0} \in \partial\mathcal{J}(\mathbf{u}^*). \tag{2.7}$$

Theorem 2.1. Let $\mathcal{J}$ be locally Lipschitz continuous. Every local minimizer $\mathbf{u}^* \in \mathbb{R}^n$ for problem (2.2) is substationary.

Theorem 2.2. If $\mathcal{J}$ is convex, then the necessary optimality condition in Theorem 2.1 is also sufficient.

To summarize, in nonsmooth optimization a generalization of the directional derivative is the set-valued *subdifferential* and, correspondingly, a generalization of the smooth, *local* indication of an extremum point $\nabla \mathcal{J}(\mathbf{u}^*) = \mathbf{0}$ is the existence of a *substationary point* $\mathbf{0} \in \partial \mathcal{J}(\mathbf{u}^*)$.

Let us illustrate the above definitions with an example $f(u) = |u|$ for $u \in \mathbb{R}$. The subdifferential of $f(u)$ is given by

$$\partial f(u) = \operatorname{sign}(u) = \begin{cases} -1, & \text{for } u < 0, \\ [-1, 1], & \text{for } u = 0, \\ 1, & \text{for } u > 0. \end{cases} \tag{2.8}$$

As can be seen, the subdifferential (i.e., generalized sign-function) coincides with the usual derivative in the well-defined case $u \neq 0$, and contains the whole set $[-1, 1]$ with endpoints of left/right converging directional derivatives of the sequence $f'(u^i)$ for $u^i \to 0$ according to (2.6). Moreover, $u^* = 0$ is the unique minimizer of $|u|$, because $0 \in \partial f(u)$ only for $u^* = 0$.

Next we turn our attention to robust statistics. Let $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ be a sample of a multivariate random variable $\mathbf{x} \in \mathbb{R}^n$. Consider the following family of optimization problems

$$\min_{\mathbf{u} \in \mathbb{R}^n} \mathcal{J}_q^\alpha(\mathbf{u}), \quad \text{for } \mathcal{J}_q^\alpha(\mathbf{u}) = \frac{1}{\alpha} \sum_{i=1}^N \|\mathbf{u} - \mathbf{x}_i\|_q^\alpha. \tag{2.9}$$

We restrict ourselves to the following combinations (cf. [Rao, 1988]): $q = \alpha = 2$, $q = \alpha = 1$, and $q = 2\alpha = 2$.

$q = \alpha = 2$: **average**

In this case, problem (2.9) is the quadratic least-squares problem, and the gradient of $\mathcal{J}_q^\alpha$ is given by

$$\nabla \mathcal{J}_2^2(\mathbf{u}) = \sum_{i=1}^N (\mathbf{u} - \mathbf{x}_i).$$

By enforcing the gradient to be zero, we recover the unique solution $\mathbf{a} = \mathbf{u}^*$ of (2.9) in the form

$$\mathbf{a} = \frac{1}{N} \sum_{i=1}^N \mathbf{x}_i,$$

which is the marginal mean (average) for the given sample.

$q = \alpha = 1$: **median**

This choice leads to the minimization of the sum of l_1-norms, which is a nonsmooth optimization problem. The subdifferential of $\mathcal{J}_1^1(\mathbf{u})$ reads as

$$\partial \mathcal{J}_1^1(\mathbf{u}) = \sum_{i=1}^N \boldsymbol{\xi}_i, \quad \text{where } (\boldsymbol{\xi}_i)_j = \operatorname{sign}((\mathbf{u} - \mathbf{x}_i)_j). \tag{2.10}$$

Next, we interpret the substationarity condition $\mathbf{0} \in \partial\mathcal{J}_1^1(\mathbf{u})$. We distinguish between the two cases of odd and even N, and by $\mathrm{sort}(A)$ we mean that the elements of A are sorted in ascending order.

For N odd, the solution $\mathbf{u}^*$ of (2.9) is realized by the unique vector $\mathbf{m}$ of marginal middle values, i.e. median, given by

$$(\mathbf{m})_j = \mathrm{sort}\left(\{(\mathbf{x}_i)_j\}\right)_{(N+1)/2} \quad \forall 1 \leq j \leq n. \tag{2.11}$$

This can be verified by first assuming that $\mathbf{u}^* \neq \mathbf{m}$, which implies that $(\mathbf{u}^*)_j \neq (\mathbf{m})_j$ for some j. Then, $\sum_i \mathrm{sign}((\mathbf{u}^* - \mathbf{x}_i)_j)$ can not be zero, and thus $\mathbf{0} \notin \partial\mathcal{J}(\mathbf{u}^*)$. On the other hand, if $\mathbf{u}^* = \mathbf{m}$, we see immediately that $\mathbf{0} \in \partial\mathcal{J}(\mathbf{u}^*)$. To conclude, we have for N odd

$$\mathbf{0} \in \partial\mathcal{J}(\mathbf{u}^*) \iff \mathbf{u}^* = \mathbf{m}, \tag{2.12}$$

where $\mathbf{u}^*$ is the unique minimizer of $\mathcal{J}_1^1$. This result also shows that for each feature $\{(\mathbf{x}_i)_j\}$ there is a prototype $(\mathbf{m})_j$ among the set of samples.

If N is even, exactly the same reasoning as above yields

$$\mathbf{0} \in \partial\mathcal{J}(\mathbf{u}^*) \iff (\mathbf{u}^*)_j \in [\mathrm{sort}(\{(\mathbf{x}_i)_j\})_{N/2}, \mathrm{sort}(\{(\mathbf{x}_i)_j\})_{N/2+1}] \tag{2.13}$$

for all j, which means that the minimizer of $\mathcal{J}_1^1(\mathbf{u})$ is given by the whole interval between the two middle values in the ordering according to the jth component of the sample data. Especially, $\mathbf{u}^*$ (median) is not unique in this case, and for each feature there are two prototypes (endpoints in (2.13)) among the set of samples.

$q = 2\alpha = 2$: **spatial median**

The gradient of the convex function $f(\mathbf{u}) = \|\mathbf{u}\|_2$ is well-defined and unique for all $\mathbf{u} \neq \mathbf{0}$. By the definition $f^0(\mathbf{0}, \mathbf{d}) = f'(\mathbf{0}, \mathbf{d}) = \|\mathbf{d}\|_2$, so that, according to (2.4), $\boldsymbol{\xi} \in \partial f(\mathbf{0})$ iff $\boldsymbol{\xi}^T\mathbf{d} \leq \|\mathbf{d}\|_2$ for all $\mathbf{d} \in \mathbb{R}^n$. Hence, by choosing $\mathbf{d} = \boldsymbol{\xi}$ it follows that $\|\boldsymbol{\xi}\|_2 \leq 1$, and, on the other hand, if $\|\boldsymbol{\xi}\|_2 \leq 1$, then $\boldsymbol{\xi}^T\mathbf{d} \leq \|\mathbf{d}\|_2$. This shows that the subgradient $\boldsymbol{\xi}$ of $f(\mathbf{u})$ at zero is characterized by the condition $\|\boldsymbol{\xi}\|_2 \leq 1$. This readily yields

$$\partial\mathcal{J}_2^1(\mathbf{u}) = \sum_{i=1}^N \boldsymbol{\xi}_i, \quad \text{with} \begin{cases} (\boldsymbol{\xi}_i)_j &= \dfrac{(\mathbf{u} - \mathbf{x}_i)_j}{\|\mathbf{u} - \mathbf{x}_i\|_2}, \text{ for } \|\mathbf{u} - \mathbf{x}_i\|_2 \neq 0, \\ \|\boldsymbol{\xi}_i\|_2 &\leq 1, \text{ for } \|\mathbf{u} - \mathbf{x}_i\|_2 = 0. \end{cases} \tag{2.14}$$

Thus, in the present case solution of (2.9) is realized by the so-called *spatial median* $\mathbf{s}$, which satisfies (2.14).

Comparison of different estimators

In statistical context robustness refers to the insensitivity of estimators towards outliers, i.e. observations which do not follow the characteristic distribution of the rest of the data. Sensitivity

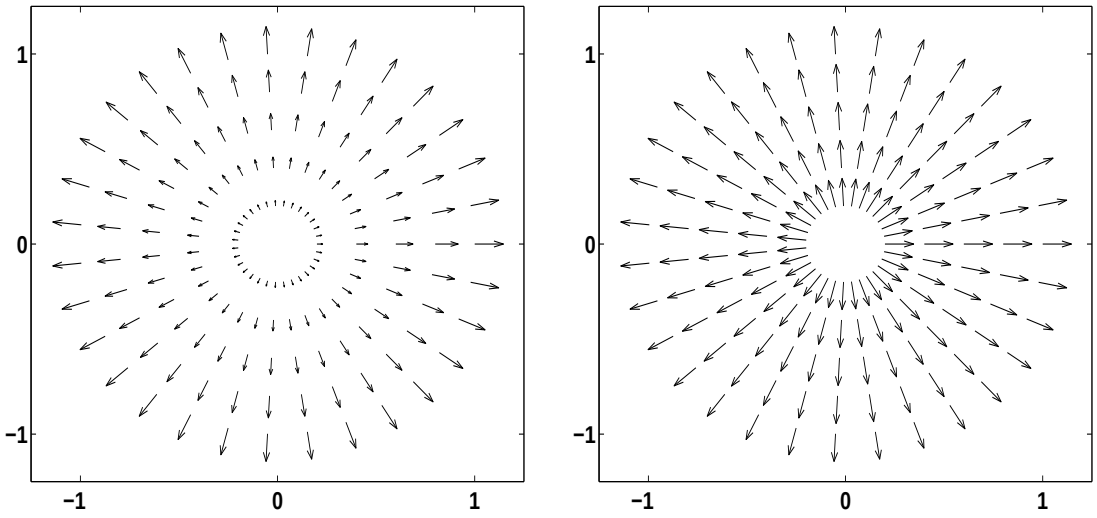

Figure 1: Scaled (scale 0.4) gradient fields of $\|\mathbf{x}\|_2^2$ (left) and $\|\mathbf{x}\|_2$ (right).

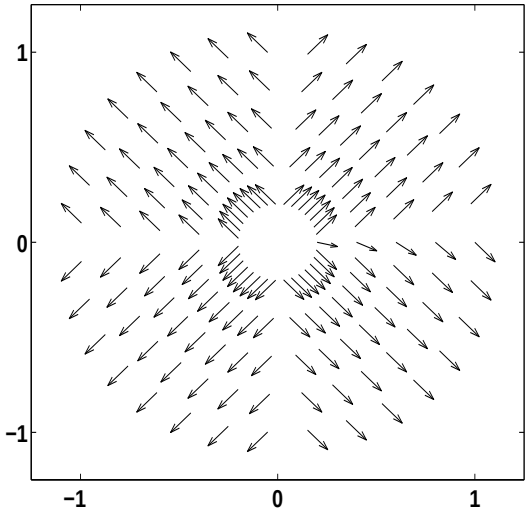

Figure 2: Scaled (scale 0.4) gradient field of $f(\mathbf{x}) = \|\mathbf{x}\|_1$.

of the average $\mathbf{a}$ towards observations lying far from the origin (representing the mean-valued estimator) is illustrated in Figure 1 (left), where the gradient field $\nabla f(\mathbf{x}) = (\mathbf{x}_1, \mathbf{x}_2)$ of 2d function $\|\mathbf{x}\|_2^2$ is given. As we can see, the size of the gradient vector increases when moving away from the origin, so that those points are weighted more heavily at the equilibrium $\nabla\|\mathbf{x}\|_2^2 = 0$. This readily explains why the (symmetric) Gaussian distribution with enough samples is the intrinsic assumption behind the least-squares estimate $\mathbf{a}$. On the other hand, an estimator with equal weight of all samples is obtained by dividing the gradient by its length, and then we precisely get the spatial median $\mathbf{s}$, which is illustrated through the gradient field of function $\|\mathbf{x}\|_2$ in Figure 1 (right). As stated, e.g., in [Hettmansperger and McKean, 1998] the corresponding estimating function $\mathcal{J}_2^1$ depends on the data only through the directions and not on the magnitudes of $\mathbf{u} - \mathbf{x}_i,\ i = 1, \ldots, N$, which significantly decreases both the sensitivity towards outliers and

requirements concerning the necessary amount of data. Finally, in Figure 2 the gradient field of a function $\|\mathbf{x}\|_1$ is depicted, where the insensitivity with respect to the distance but also the lack of rotational invariance (due to different contour lines of the unit ball in the 1-norm) are clearly visible.

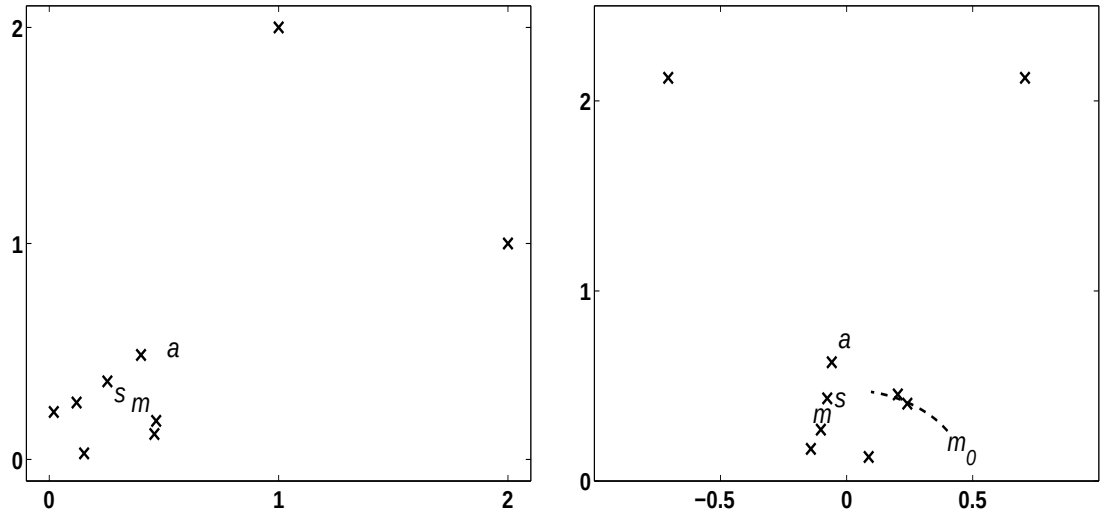

Figure 3: Left: Bivariate example of a uniform sample with two outliers illustrating the average $\mathbf{a}$, median $\mathbf{m}$, and spatial median $\mathbf{s}$. Right: Same plot for the 45 degrees rotated data including the original median $\mathbf{m}_0$ and its rotation path showing the lack of rotational invariance.

We further illustrate the behaviour of the three estimators $\mathbf{a}$, $\mathbf{m}$, and $\mathbf{s}$ in Figure 3 with a sample data. This example also depicts both the sensitivity of $\mathbf{a}$ towards outliers and lack of rotational invariance of $\mathbf{m}$.

2.2 MLP in a layer-wise form

A compact description for the action of the multilayered perceptron neural network is given by ([Hagan and Menhaj, 1994, Kärkkäinen, 2002])

$$\mathbf{o}^0 = \mathbf{x}, \quad \mathbf{o}^l = \mathcal{F}^l(\mathbf{W}^l \hat{\mathbf{o}}^{(l-1)}) \quad \text{for} \quad l = 1, \ldots, L. \tag{2.15}$$

Here the superscript l corresponds to the layer number (starting from zero for the input) and by the circumflex we indicate the normal extension of a vector by unity. $\mathcal{F}^l(\cdot)$ denotes the usual componentwise activation on the lth level, which can be represented by using a *diagonal function-matrix* $\mathcal{F} = \mathcal{F}(\cdot) = \mathrm{Diag}\{f_i(\cdot)\}_{i=1}^m$ supplied with the natural definition of the matrix-vector product $\mathbf{y} = \mathcal{F}(\mathbf{v}) \equiv (\mathbf{y})_i = f_i((\mathbf{v})_i)$. Notice though that the following analysis generalizes straightforwardly to the case of an activation with nondiagonal function matrix [Kärkkäinen, 2002]. The dimensions of the weight-matrices are given by $\dim(\mathbf{W}^l) = n_l \times (n_{l-1} + 1),\ l = 1, \ldots, L$, where n_0 is the length of an input-vector $\mathbf{x}$, n_L the length of

the output-vector $\mathbf{o}^L$, and $n_l, 0 < l < L$, determine the sizes (number of neurons) of the hidden layers. Due to the special bias weights in the first column, the column numbering for each weight-matrix starts from zero.

Instead of precisely (2.15) we consider an architecture of MLP containing only a linear transformation in the final layer as $\mathbf{o}_i^L = \mathcal{N}(\{\mathbf{W}^l\})(\mathbf{x}_i) = \mathbf{W}^L \hat{\mathbf{o}}_i^{(L-1)}$. With a given training data $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N$, $\mathbf{x}_i \in \mathbb{R}^{n_0}$ and $\mathbf{y}_i \in \mathbb{R}^{n_L}$, the unknown weight matrices $\{\mathbf{W}^l\}_{l=1}^L$ are determined as a solution of the optimization problem

$$\min_{\{\mathbf{W}^l\}_{l=1}^L} \mathcal{L}_{q,\beta}^{\alpha}(\{\mathbf{W}^l\}), \tag{2.16}$$

where the cost functional is of the general form

$$\mathcal{L}_{q,\beta}^{\alpha}(\{\mathbf{W}^l\}) = \frac{1}{\alpha N} \sum_{i=1}^{N} \left\| \mathcal{N}(\{\mathbf{W}^l\})(\mathbf{x}_i) - \mathbf{y}_i \right\|_q^{\alpha} + \frac{\beta}{2} \sum_{l=1}^{L} \sum_{(i,j)\in I_l} |\mathbf{W}_{i,j}^l|^2. \tag{2.17}$$

Here, the index sets I_l are defined as follows:

$$I_l = \begin{cases} \{(i,j) \,:\, 1 \le i \le n_l,\ 0 \le j \le n_{l-1}\}, & l < L, \\ \{(i,j) \,:\, 1 \le i \le n_l,\ 1 \le j \le n_{l-1}\}, & l = L, \end{cases} \tag{2.18}$$

which means that the weight decay term in (2.17) contains all other components of the unknown weight matrices except the final bias in $\mathbf{W}^L$ as suggested by the test results in [Kärkkäinen, 2002].

All features in the training data $\{\mathbf{x}_i, \mathbf{y}_i\}$ are preprocessed to the range $[-1, 1]$ of the k-tanh functions

$$t_k(a) = \frac{2}{1 + \exp(-2\,k\,a)} - 1 \quad \text{for } k \in \mathbb{N}, \tag{2.19}$$

which are used in the activation. In this way, we balance the scaling of unknowns (components of weight matrices at different layers) in problem (2.16) [Kärkkäinen, 2002].

In cost function (2.17), we control the generality (model complexity) of a trained network using only a single hyperparameter, the weight decay coefficient $\beta \ge 0$, which restricts the universality of the network by forcing the unknowns towards the neighborhood of zero, i.e. around the linear region of the activation functions in (2.19). Function (2.17) is also related to Bayesian statistics with compatible choices of the sample data and prior distributions (e.g., [Rögnvaldsson, 1998]). Moreover, we consider the same three combinations for the parameters q and α as in the previous section, namely $q = \alpha = 2,\ q = \alpha = 1$, and $q = 2\alpha = 2$. Hence, we conclude that the considered family of learning problem formulations for the MLP results from a compound application of robust and Bayesian statistics.

For solving the problems (2.16) we use efficient generalizations of gradient-based methods for nonsmooth problems known as bundle methods [Mäkelä and Neittaanmäki, 1992]. We recall from [Kärkkäinen, 2002] that for $\alpha = 1$ the assumptions for convergence of gradient-descent (on batch-mode Lipschitz continuity of gradient, for on-line stochastic iteration C^2

continuity), CG (Lipschitz continuity of gradient), and especially quasi-Newton methods (C^2-continuity) are violated [Haykin, 1994, Nocedal and Wright, 1999]. As documented, e.g., for MLP by [Saito and Nakano, 2000] and for image restoration by [Kärkkäinen et al., 2001] this yields nonconvergence of ordinary training algorithms, when the cost function does not fulfill the required smoothness assumptions. Furthermore, results in [Kärkkäinen et al., 2001] indicate that simple smoothing techniques, such as replacing a norm $\|\mathbf{v}\|_2$ for $\mathbf{v} \in \mathbb{R}^2$ by $\sqrt{\mathbf{v}_1^2 + \mathbf{v}_2^2 + \varepsilon}$ for $\varepsilon > 0$, are not sufficient to restore the convergence of ordinary optimization methods.

3 Sensitivity Analysis and its Consequences

Next we apply a useful technique, also presented in [Kärkkäinen, 2002], to derive the optimality conditions for the network training problem (2.16). From now on, for any vector $\mathbf{v} \in \mathbb{R}^n$, the notation $\mathrm{sign}[\mathbf{v}]$ means a componentwise application of the sign-function and the abbreviated notation $\boldsymbol{\xi} = \mathbf{v}/\|\mathbf{v}\|_2$ actually refers to

$$(\boldsymbol{\xi})_i = \frac{(\mathbf{v})_i}{\|\mathbf{v}\|_2}, \text{ for } \|\mathbf{v}\|_2 \neq 0, \quad \|\boldsymbol{\xi}\|_2 \leq 1, \text{ for } \|\mathbf{v}\|_2 = 0. \tag{3.1}$$

For simplicity, we assume that the activation functions in all function-matrices $\mathcal{F}(\cdot)$ are differentiable, although the analysis below can be extended and given algorithms applied also to nondifferentiable activation functions. Let us further emphasize that the use of nonsmooth activation functions (step-function or $a/(1+|a|)$, e.g., [Prechelt, 1998]) makes the learning problem nonsmooth even for $q = \alpha = 2$, and therefore ordinary gradient-based optimization algorithms can not be used for solving.

3.1 MLP with One Hidden Layer

For clarity, we start with MLP with only one hidden layer. Then, any local solution $(\mathbf{W}^{1*}, \mathbf{W}^{2*})$ of the minimization problem (2.16) is characterized by the conditions

$$\begin{bmatrix} \mathbf{O} \\ \mathbf{O} \end{bmatrix} \in \partial_{(\mathbf{W}^1, \mathbf{W}^2)} \mathcal{L}_{q,\beta}^{\alpha}(\mathbf{W}^{1*}, \mathbf{W}^{2*}) = \begin{bmatrix} \partial_{\mathbf{W}^1} \mathcal{L}_{q,\beta}^{\alpha}(\mathbf{W}^{1*}, \mathbf{W}^{2*}) \\ \partial_{\mathbf{W}^2} \mathcal{L}_{q,\beta}^{\alpha}(\mathbf{W}^{1*}, \mathbf{W}^{2*}) \end{bmatrix}. \tag{3.2}$$

Here, $\partial_{\mathbf{W}^l} \mathcal{L}_{q,\beta}^{\alpha}$, $l = 1, 2$, are subdifferentials presented in a similar matrix-form as the unknown weight-matrices.

We begin the derivation with some lemmata. The proofs are omitted here, because they follow exactly the same lines as the proofs of the corresponding lemmata in [Kärkkäinen, 2002], using the already introduced results on subdifferential calculus in Section 2.1.

Lemma 3.1. Let $\mathbf{v} \in \mathbb{R}^{m_1}$ and $\mathbf{y} \in \mathbb{R}^{m_2}$ be given vectors. The subgradient-matrix $\partial_{\mathbf{W}} J(\mathbf{W})$, $\mathbf{W} \in$

$\mathbb{R}^{m_2\times m_1}$, for the functional $J(\mathbf{W}) = \frac{1}{\alpha}\|\mathbf{W}\,\mathbf{v} - \mathbf{y}\|_q^\alpha$ is of the form

$$\partial_{\mathbf{W}}\, J(\mathbf{W}) = \boldsymbol{\xi}\,\mathbf{v}^T, \text{ where } \boldsymbol{\xi} = \begin{cases} [\mathbf{W}\,\mathbf{v} - \mathbf{y}], & \text{for } q = \alpha = 2, \\ \operatorname{sign}[\mathbf{W}\,\mathbf{v} - \mathbf{y}], & \text{for } q = \alpha = 1, \\ \dfrac{\mathbf{W}\,\mathbf{v} - \mathbf{y}}{\|\mathbf{W}\,\mathbf{v} - \mathbf{y}\|_2}, & \text{for } q = 2\alpha = 2. \end{cases}$$

Lemma 3.2. Let $\mathbf{W} \in \mathbb{R}^{m_2\times m_1}$ be a given matrix, $\mathbf{y} \in \mathbb{R}^{m_2}$ a given vector, and $\mathcal{F} = \operatorname{Diag}\{f_i(\cdot)\}_{i=1}^{m_1}$ a given diagonal function-matrix. The subgradient-vector $\partial_{\mathbf{u}} J(\mathbf{u}),\ \mathbf{u} \in \mathbb{R}^{m_1}$, for the functional $J(\mathbf{u}) = \frac{1}{\alpha}\|\mathbf{W}\,\mathcal{F}(\mathbf{u}) - \mathbf{y}\|_q^\alpha$ reads as

$$\partial_{\mathbf{u}} J(\mathbf{u}) = \Big(\mathbf{W}\,\mathcal{F}^{'}(\mathbf{u})\Big)^T\,\boldsymbol{\xi} = \operatorname{Diag}\{\mathcal{F}^{'}(\mathbf{u})\}\,\mathbf{W}^T\boldsymbol{\xi},$$

where

$$\boldsymbol{\xi} = \begin{cases} [\mathbf{W}\,\mathcal{F}(\mathbf{u}) - \mathbf{y}], & \text{for } q = \alpha = 2, \\ \operatorname{sign}[\mathbf{W}\,\mathcal{F}(\mathbf{u}) - \mathbf{y}], & \text{for } q = \alpha = 1, \\ \dfrac{\mathbf{W}\,\mathcal{F}(\mathbf{u}) - \mathbf{y}}{\|\mathbf{W}\,\mathcal{F}(\mathbf{u}) - \mathbf{y}\|_2}, & \text{for } q = 2\alpha = 2. \end{cases}$$

Lemma 3.3. Let $\bar{\mathbf{W}} \in \mathbb{R}^{m_2\times m_1}$ be a given matrix, $\mathcal{F} = \operatorname{Diag}\{f_i(\cdot)\}_{i=1}^{m_1}$ a given diagonal function-matrix, and $\mathbf{v} \in \mathbb{R}^{m_0}$, $\mathbf{y} \in \mathbb{R}^{m_2}$ given vectors. The subgradient-matrix $\partial_{\mathbf{W}}\, J(\mathbf{W}),\ \mathbf{W} \in \mathbb{R}^{m_1\times m_0}$, for the functional

$$J(\mathbf{W}) = \frac{1}{\alpha}\|\bar{\mathbf{W}}\,\mathcal{F}(\mathbf{W}\mathbf{v}) - \mathbf{y}\|_q^\alpha$$

is of the form $\partial_{\mathbf{W}}\, J(\mathbf{W}) = \operatorname{Diag}\{\mathcal{F}^{'}(\mathbf{W}\mathbf{v})\}\,\bar{\mathbf{W}}^T\,\boldsymbol{\xi}\,\mathbf{v}^T$, where

$$\boldsymbol{\xi} = \begin{cases} [\bar{\mathbf{W}}\,\mathcal{F}(\mathbf{W}\mathbf{v}) - \mathbf{y}], & \text{for } q = \alpha = 2, \\ \operatorname{sign}[\bar{\mathbf{W}}\,\mathcal{F}(\mathbf{W}\mathbf{v}) - \mathbf{y}], & \text{for } q = \alpha = 1, \\ \dfrac{\bar{\mathbf{W}}\,\mathcal{F}(\mathbf{W}\mathbf{v}) - \mathbf{y}}{\|\bar{\mathbf{W}}\,\mathcal{F}(\mathbf{W}\mathbf{v}) - \mathbf{y}\|_2}, & \text{for } q = 2\alpha = 2. \end{cases}$$

Now we are ready to state the actual results for the perceptron with one hidden layer. In what follows, we denote by $\mathbf{W}_1^2$ the submatrix $(\mathbf{W}^2)_{i,j}\,,\ i = 1,\ldots,n_2,\ j = 1,\ldots,n_1$, which is obtained from $\mathbf{W}^2$ by removing the first column $\mathbf{W}_0^2$ containing the bias nodes. Furthermore, the error in the ith output is denoted by $\mathbf{e}_i = \mathbf{W}^2\,\widehat{\mathcal{F}}(\mathbf{W}^1\,\hat{\mathbf{x}}_i) - \mathbf{y}_i$.

Theorem 3.1. Subgradient-matrices $\partial_{\mathbf{W}^2}\,\mathcal{L}_{q,\beta}^{\alpha}(\mathbf{W}^1,\,\mathbf{W}^2) \subset \mathbb{R}^{n_2\times(n_1+1)}$ and $\partial_{\mathbf{W}^1}\,\mathcal{L}_{q,\beta}^{\alpha}(\mathbf{W}^1,\,\mathbf{W}^2) \subset \mathbb{R}^{n_1\times(n_0+1)}$ are of the form

$$\partial_{\mathbf{W}^2}\,\mathcal{L}_{q,\beta}^{\alpha}(\mathbf{W}^1,\mathbf{W}^2) = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{\xi}_i\,[\widehat{\mathcal{F}}(\mathbf{W}^1\,\hat{\mathbf{x}}_i)]^T + \beta\,[\mathbf{0}\ \mathbf{W}_1^2], \tag{3.3}$$

$$\partial_{\mathbf{W}^1}\,\mathcal{L}_{q,\beta}^{\alpha}(\mathbf{W}^1,\mathbf{W}^2) = \frac{1}{N}\sum_{i=1}^{N}\operatorname{Diag}\{\mathcal{F}^{'}(\mathbf{W}^1\,\hat{\mathbf{x}}_i)\}\,(\mathbf{W}_1^2)^T\,\boldsymbol{\xi}_i\,\hat{\mathbf{x}}_i^T + \beta\,\mathbf{W}^1, \tag{3.4}$$

where

$$\boldsymbol{\xi}_i = \begin{cases} \mathbf{e}_i, & \text{for } q = \alpha = 2, \\ \operatorname{sign}[\mathbf{e}_i], & \text{for } q = \alpha = 1, \\ \frac{\mathbf{e}_i}{\|\mathbf{e}_i\|_2}, & \text{for } q = 2\alpha = 2. \end{cases}$$

3.2 MLP with Several Hidden Layers

Next, we generalize the previous analysis to the case of several hidden layers.

Lemma 3.4. Let $\tilde{\mathbf{W}} \in \mathbb{R}^{m_3 \times m_2}$ and $\bar{\mathbf{W}} \in \mathbb{R}^{m_2 \times m_1}$ be given matrices, $\tilde{\mathcal{F}} = \operatorname{Diag}\{\tilde{f}_i(\cdot)\}_{i=1}^{m_2}$ and $\bar{\mathcal{F}} = \operatorname{Diag}\{\bar{f}_i(\cdot)\}_{i=1}^{m_1}$ given diagonal function-matrices, and $\mathbf{v} \in \mathbb{R}^{m_0}$, $\mathbf{y} \in \mathbb{R}^{m_3}$ given vectors. The subgradient-matrix $\partial_{\mathbf{W}} J(\mathbf{W}),\ \mathbf{W} \in \mathbb{R}^{m_1 \times m_0}$, for the functional

$$J(\mathbf{W}) = \frac{1}{\alpha} \|\bar{\mathbf{W}}\, \bar{\mathcal{F}}(\tilde{\mathbf{W}}\, \tilde{\mathcal{F}}(\mathbf{W}\mathbf{v})) - \mathbf{y}\|_q^\alpha$$

is of the form

$$\partial_{\mathbf{W}} J(\mathbf{W}) = \operatorname{Diag}\{\tilde{\mathcal{F}}'(\mathbf{W}\mathbf{v})\}\, \tilde{\mathbf{W}}^T \operatorname{Diag}\{\bar{\mathcal{F}}'(\tilde{\mathbf{W}}\, \tilde{\mathcal{F}}(\mathbf{W}\mathbf{v}))\}\, \bar{\mathbf{W}}^T \boldsymbol{\xi}\, \mathbf{v}^T,$$

where

$$\boldsymbol{\xi} = \begin{cases} [\bar{\mathbf{W}}\, \bar{\mathcal{F}}(\tilde{\mathbf{W}}\, \tilde{\mathcal{F}}(\mathbf{W}\mathbf{v})) - \mathbf{y}], & \text{for } q = \alpha = 2, \\ \operatorname{sign}[\bar{\mathbf{W}}\, \bar{\mathcal{F}}(\tilde{\mathbf{W}}\, \tilde{\mathcal{F}}(\mathbf{W}\mathbf{v})) - \mathbf{y}], & \text{for } q = \alpha = 1, \\ \dfrac{\bar{\mathbf{W}}\, \bar{\mathcal{F}}(\tilde{\mathbf{W}}\, \tilde{\mathcal{F}}(\mathbf{W}\mathbf{v})) - \mathbf{y}}{\|\bar{\mathbf{W}}\, \bar{\mathcal{F}}(\tilde{\mathbf{W}}\, \tilde{\mathcal{F}}(\mathbf{W}\mathbf{v})) - \mathbf{y}\|_2}, & \text{for } q = 2\alpha = 2. \end{cases}$$

Theorem 3.2. Subgradient-matrices $\partial_{\mathbf{W}^l} \mathcal{L}_{q,\beta}^\alpha(\{\mathbf{W}^l\}),\ l = L, \ldots, 1$, read as

$$\partial_{\mathbf{W}^l} \mathcal{L}_q^\alpha(\{\mathbf{W}^l\}) = \frac{1}{N} \sum_{i=1}^N \boldsymbol{\xi}_i^l\, [\hat{\mathbf{o}}_i^{(l-1)}]^T + \beta\, \widetilde{\mathbf{W}}^l,$$

where

$$\boldsymbol{\xi}_i^L = \begin{cases} \mathbf{e}_i, & \text{for } q = \alpha = 2, \\ \operatorname{sign}[\mathbf{e}_i], & \text{for } q = \alpha = 1, \\ \dfrac{\mathbf{e}_i}{\|\mathbf{e}_i\|_2} & \text{for } q = 2\alpha = 2, \end{cases} \tag{3.5}$$

$$\boldsymbol{\xi}_i^l = \operatorname{Diag}\{(\mathcal{F}^l)'(\mathbf{W}^l\, \hat{\mathbf{o}}_i^{(l-1)})\}\, (\mathbf{W}_1^{(l+1)})^T\, \boldsymbol{\xi}_i^{(l+1)}. \tag{3.6}$$

Furthermore, $\widetilde{\mathbf{W}}^l = [\mathbf{0}\ \mathbf{W}_1^L]$ for $l = L$, and coincides with the whole matrix $\mathbf{W}^l$ for $1 \le l < L$.

The compact presentation of the optimality system in Theorem 3.2 can be readily exploited in the implementation, which practically consists of a few basic linear-algebraic operations. Moreover, the following result concerning *every* local minimizer $\mathbf{O} \in \partial_{\mathbf{W}^l} \mathcal{L}_{q,\beta}^\alpha(\{\mathbf{W}^{l*}\})$ of problem (2.16) holds.

Corollary 3.1. For locally optimal MLP-network $\{\mathbf{W}^{l^*}\}$ satisfying the conditions in Theorem 3.2:

$$\begin{cases} \dfrac{1}{N}\displaystyle\sum_{i=1}^{N} \mathbf{e}_i^* = \mathbf{0}, & \text{for } q = \alpha = 2, \\ \mathbf{0} \in \displaystyle\sum_{i=1}^{N} \operatorname{sign}[\mathbf{e}_i^*], & \text{for } q = \alpha = 1, \\ \mathbf{0} \in \displaystyle\sum_{i=1}^{N} \frac{\mathbf{e}_i^*}{\|\mathbf{e}_i^*\|_2}, & \text{for } q = 2\alpha = 2, \end{cases}$$

for all $\beta \geq 0$.

Proof. The optimality condition

$$\mathbf{O} \in \partial_{\mathbf{W}^L} \mathcal{L}^{\alpha}_{q,\beta}(\{\mathbf{W}^{l^*}\}) = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{\xi}_i^{L^*} \, [\hat{\mathbf{o}}_i^{(L-1)}]^T + \beta \, [\mathbf{0} \; \mathbf{W}_1^{L^*}]$$

(with the abbreviation $\hat{\mathbf{o}}_i^{(L-1)} = \hat{\mathbf{o}}_i^{(L-1)^*}$) in Theorem 3.2 can be written in the non-extended form as

$$\mathbf{O} \in \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{\xi}_i^{L^*} \, [1 \; (\mathbf{o}_i^{(L-1)})^T] + \beta \, [\mathbf{0} \; \mathbf{W}_1^{L^*}].$$

By taking the transpose on the right-hand-side we obtain

$$\frac{1}{N}\sum_{i=1}^{N} \begin{bmatrix} 1 \\ \mathbf{o}_i^{(L-1)} \end{bmatrix} (\boldsymbol{\xi}_i^{L^*})^T + \begin{bmatrix} \mathbf{0}^T \\ \beta \, (\mathbf{W}_1^{L^*})^T \end{bmatrix} = \frac{1}{N}\sum_{i=1}^{N} \begin{bmatrix} (\boldsymbol{\xi}_i^{L^*})^T \\ \mathbf{o}_i^{(L-1)} \, (\boldsymbol{\xi}_i^{L^*})^T + \beta \, (\mathbf{W}_1^{L^*})^T \end{bmatrix}.$$

Finally, by using the definitions in (3.5) for $\boldsymbol{\xi}_i^{L^*}$ in the first row shows the results. □

The interpretation of Corollary 3.1 is significant: The special choice of MLP-architecture with linear final layer together with the proposed choice of quadratic regularization without final bias of $\mathbf{W}^L$ allows one to generate MLP-mappings which obey the robustness properties of the norms for the fitting. Namely, according to Corollary 3.1, for all $\beta \geq 0$:

1. Every local minimizer of $\mathcal{L}^2_{2,\beta}(\{\mathbf{W}^l\})$ generates function with average error over the learning data equal to zero.
2. Local minimizers of $\mathcal{L}^1_{1,\beta}(\{\mathbf{W}^l\})$ yield zero median error for the MLP-function.
3. Minimization of $\mathcal{L}^1_{2,\beta}(\{\mathbf{W}^l\})$ enforces spatial median error of the MLP-transformation to zero.

To this end, we notice that these results are actually valid for *all kind of regression approximators with separate bias*.

4 Numerical experiments

4.1 Univariate single-valued regression

In the first test setting, we study the use of the MLP-network in the reconstruction of a given single-valued function of one variable, which is disturbed by random noise. We train the network by solving the optimization problem (2.16) both with $\alpha = q = 2$ and $\alpha = q = 1$, and we compare the results given by these two approaches. We remark that in this case $n_L = 1$, and thus, the functionals $\mathcal{L}^1_{2,\beta}$ and $\mathcal{L}^1_{1,\beta}$ are identical. The minimizations are performed by the proximity control bundle method, which is applicable both to the smooth functional $\mathcal{L}^2_{2,\beta}$ and to the nonsmooth functional $\mathcal{L}^1_{1,\beta}$ [Mäkelä and Neittaanmäki, 1992].

Definition of the test problem

We consider the reconstruction of the function $f(x) = \sin(x)$ in the interval $x \in [0, 2\pi]$. The input-vectors of the training data are chosen to be N uniformly spaced values $\mathbf{x}_i$ from the interval $[0, 2\pi]$ given by $\mathbf{x}_i = (i-1)\,2\pi/(N-1)$. The samples of function values involve two types of random noise: Low-amplitude normally distributed noise affects the values over the whole interval $[0, 2\pi]$, while at some isolated points, the values are disturbed by high-amplitude uniformly distributed noise (outliers). Hence, we choose

$$\mathbf{y}_i = \sin(\mathbf{x}_i) + \delta\,\varepsilon_i + \zeta\,\eta_i, \tag{4.1}$$

where $\varepsilon_i \sim \mathcal{N}(0,1)$ and

$$\eta_i = \begin{cases} \mathcal{U}(-1,1), & i \in O, \\ 0, & i \notin O. \end{cases} \tag{4.2}$$

Here, $\mathcal{U}(-1,1)$ denotes the uniform distribution on $(-1,1)$ and O is an index set of outliers such that $O \subset \{1, 2, \ldots, N\}$.

We use the MLP with one hidden layer (i.e., $L = 2$) considered in Section 3.1. The activation is performed with the k-tanh functions (2.19) such that $\mathcal{F}(\cdot) = \mathrm{Diag}\{t_i(\cdot)\}_{i=1}^{n_1}$. The input and output dimensions are both equal to one ($n_0 = n_2 = 1$), and we use the values 5, 10, and 20 for the dimension n_1 of the hidden layer. The size N of the training data is 30, 60, or 120, and correspondingly the index set O is chosen to contain 3, 5, or 10 randomly selected indices between 1 and N. The amplitudes of the normally and uniformly distributed noise are $\delta = 0.3$ and $\zeta = 2$, respectively. The training dataset $\{\mathbf{x}_i, \mathbf{y}_i\}$ in the case $N = 30$, created according to the definitions above (before scaling to the range of the activation functions), is illustrated in Figure 4.

Comparison of formulations

Our goal is to estimate the approximation capability of the MLP-networks corresponding to the minimization of the two functionals $\mathcal{L}^1_{1,\beta}$ and $\mathcal{L}^2_{2,\beta}$ with different values of the param-

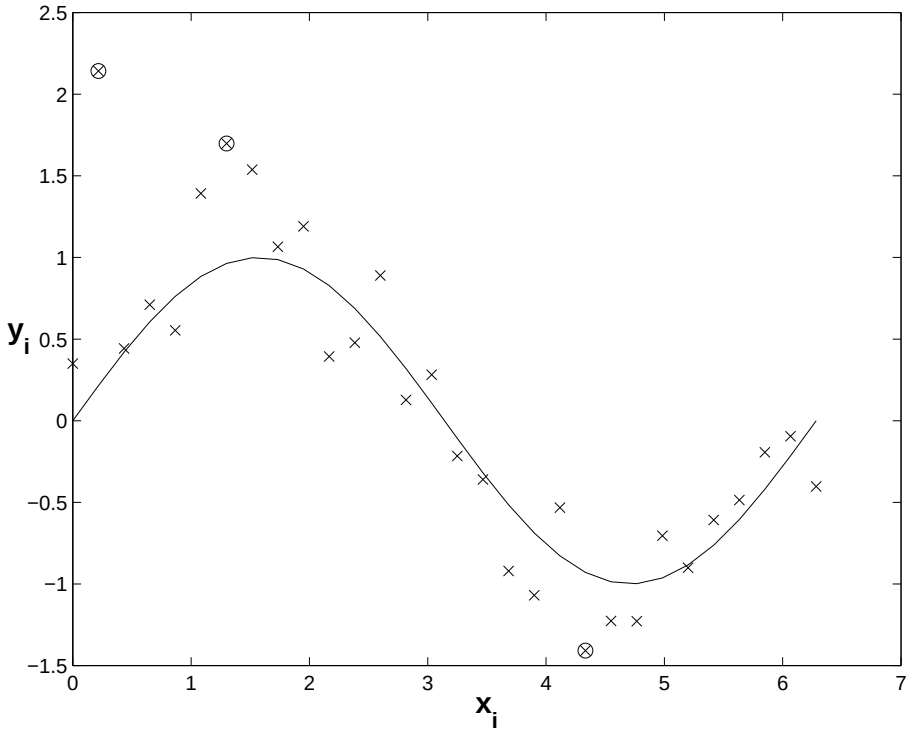

Figure 4: The training dataset $\{\mathbf{x}_i, \mathbf{y}_i\}$ in the case $N = 30$ and the exact graph of function f. The circled markers involve also uniformly distributed noise.

eters N, n_1, and β. For this purpose, we define a validation set of input-values by $\hat{\mathbf{x}}_i = (i-1)\,2\pi/(N_t-1),\ N_t = 257$, which do not coincide with the input-values $\mathbf{x}_i$ of the training data. The difference between the MLP-approximation and the exact function f is then calculated by using the norm

$$\text{err}\,(\mathbf{W}^1, \mathbf{W}^2) = \frac{1}{N_t} \sum_{i=1}^{N_t} \left| \mathbf{W}^2\, \widehat{\mathcal{F}}(\mathbf{W}^1\, \hat{\mathbf{x}}_i) - f(\hat{\mathbf{x}}_i) \right|. \tag{4.3}$$

Let us emphasize that the choice of error measure is not based on favoring $\mathcal{L}^1_{1,\beta}$ but on the fact that this form weights equally both small and large deviations from the exact function.

We performed a series of tests with the three different values for N and n_1. For fixed N and n_1, the value of the regularization parameter β varied in the interval $[0, 1]$, and for each β, we repeated the optimization algorithm 100 times with randomly created initial values for the weight matrices $\{\mathbf{W}^1, \mathbf{W}^2\}$ and computed the minimum and average values of the errors (4.3). We remind that it is well-known and tested that the optimization problems to be solved for training the MLP are nonconvex, and thus, there is a large number of local minima (all satisfying Corollary 3.1) in the error surface to be explored by random initialization.

We computed also the value of the regularization term

$$r(\{\mathbf{W}^l\}) = \frac{\beta}{2} \sum_{l=1}^{L} \sum_{(i,j)\in I_l} |\mathbf{W}^l_{i,j}|^2 \tag{4.4}$$

		$\mathcal{L}^1_{1,\beta}$		$\mathcal{L}^2_{2,\beta}$	
N	n_1	β^*	err*	β^*	err*
30	5	2.0e-2	1.2e-1	7.7e-3	1.6e-1
	10	4.1e-2	1.2e-1	2.0e-2	1.6e-1
	20	1.0e-1	1.3e-1	4.1e-2	1.6e-1
60	5	2.0e-5	5.9e-2	6.4e-4	9.2e-2
	10	2.0e-2	6.9e-2	2.6e-3	9.4e-2
	20	4.1e-2	6.6e-2	6.4e-3	1.0e-1
120	5	1.9e-3	4.3e-2	1.2e-4	5.7e-2
	10	1.3e-2	4.7e-2	6.4e-4	5.9e-2
	20	3.1e-2	4.8e-2	2.6e-3	6.2e-2

Table 1: Optimal values β^* of the regularization parameter for different values of N and n_1 with the functionals $\mathcal{L}^1_{1,\beta}$ and $\mathcal{L}^2_{2,\beta}$. Column err* gives the minimum error obtained with the value β^* over 100 tests.

of the functional $\mathcal{L}^\alpha_{q,\beta}$ corresponding to the MLP with minimum error in (4.3) for finding an effective way to choose the parameter β. This computation was motivated by the fact that our previous studies in image restoration with similar functions to be optimized have shown a strong correlation between the reconstruction error and the value of the regularization term [Kärkkäinen and Majava, 2000]. In addition to the well-known cross-validation techniques, simpler heuristics for this purpose have been proposed and tested with the backpropagation-algorithm, e.g., in [Rögnvaldsson, 1998].

The results are collected in Figures 9–20. Each figure includes three graphs corresponding to the three dimensions n_1 of the hidden layer. For certain functional and fixed value of N, the graphs represent either the average value of the errors in the 100 tests or the value of the regularization term. In each case, the norm (4.3) obtains minimum value at certain β, and these optimal points and minimal values of 'err' are listed in Table 1. The optimal points are marked also in the graphs by vertical dashed segments.

We see that in each test case the MLP-network based on the minimization of the functional $\mathcal{L}^1_{1,\beta^*}$ leads to a better approximation of the exact function f than the MLP based on $\mathcal{L}^2_{2,\beta}$. We can also make the natural conclusion that the error is reduced by increasing N. The figures show that with larger dimension of the hidden layer the overall values of the error become smaller, but the minimum value remains essentially the same. Moreover, with larger value of n_1, the error of the MLP-approximation becomes less sensitive to the choice of the regularization parameter. However, there is a remarkable difference in the behaviour of the two learning problem formulations for $n_1 = 20$ (i.e., with high representation capability of MLP): When β grows from β^*, the average error for $\mathcal{L}^1_{1,\beta}$ essentially stays on the same level whereas for $\mathcal{L}^2_{2,\beta}$ there is approximately a linear increase. In addition, for $\mathcal{L}^2_{2,\beta}$ small deviations from the optimal regularization parameter β^* may lead to a large increase in the error. Interestingly this suggests that the well-known approach in statistics "to integrate over the nuisance parameters"

like β would here yield a poorer results (especially for $\mathcal{L}^2_{2,\beta}$) than choosing an appropriate single value.

From the graphs of the regularization term we conclude that the strong oscillation of $r(\{\mathbf{W}^l\})$ indicates that the value of β is smaller than the optimal value β^*. In other words, the MLP is too complex with unnecessary variance. However, otherwise the reconstruction error and the regularization term are not clearly correlated, and thereby the value of $r(\{\mathbf{W}^l\})$ does not contain enough information in choosing the parameter β exactly. For $q = \alpha = 1$ and $n_1 = 20$ there seems to be some similarity in the graphs for different values of N to indicate β^*, although this visual information is difficult to quantify precisely.

4.2 Bivariate vector-valued regression

In the second set of experiments, we consider the reconstruction of a vector-valued function from noisy data. We use again the MLP-network with one hidden layer and k-tanh activation and train the network by minimizing the functional $\mathcal{L}^\alpha_{q,\beta}$ with the three choices $q = \alpha = 2$, $q = \alpha = 1$, and $q = 2\alpha = 2$. Since the output-dimension n_L of the network is larger than one, the function $\mathcal{L}^1_{2,\beta}$ differs from $\mathcal{L}^1_{1,\beta}$ unlike in the case $n_L = 1$.

The test function is formed as a sum of a global term, which affects the function values over the whole domain, and a local term, which is nonzero only in a small part of the domain. It is well-known that MLP is efficient in approximating the global behaviour of a function, but due to its structure it tends to ignore the local variations. Another commonly used neural architecture is the radial basis function network (RBFN) [Broomhead and Lowe, 1988], which builds approximations with local basis functions. Therefore, when properly focused, RBFN can catch the local term but gives poor approximations to the global term.

A simple idea to combine the advantages of these two types of networks is to augment the input of the MLP by the squares of the input-values. Flake refers to such MLP-networks as SQUARE-MLP (square unit augmented, radially extended, multilayer perceptron) [Flake, 1998] (see also [Sarajedini and Hecht-Nielsen, 1992]) . Such networks retain the ability to form global representations, but they can also form local approximations with a single hidden node.

Definition of the test problem

We define the vector-valued function $\mathbf{f} : \mathbb{R}^2 \to \mathbb{R}^2$, $\mathbf{f}(x, y) = (\mathbf{f}_1(x, y), \mathbf{f}_2(x, y))$ as

$$\mathbf{f}_1(x, y) = 4 \exp\left(-\frac{(x - x_0)^2 + (y - y_0)^2}{0.7}\right) - \frac{2}{1 + \exp(-6x)} + 1, \tag{4.5}$$

and $\mathbf{f}_2(x, y) = \mathbf{f}_1(y, -x)$ with $x_0 = y_0 = 2.5$. The function (4.5) is an example of a "Hill-Plateau" surface [Sarle, 1997], which is a sum of a local Gaussian and a global tanh-function. The approximation of such a function is known to be difficult for both MLP and RBFN.

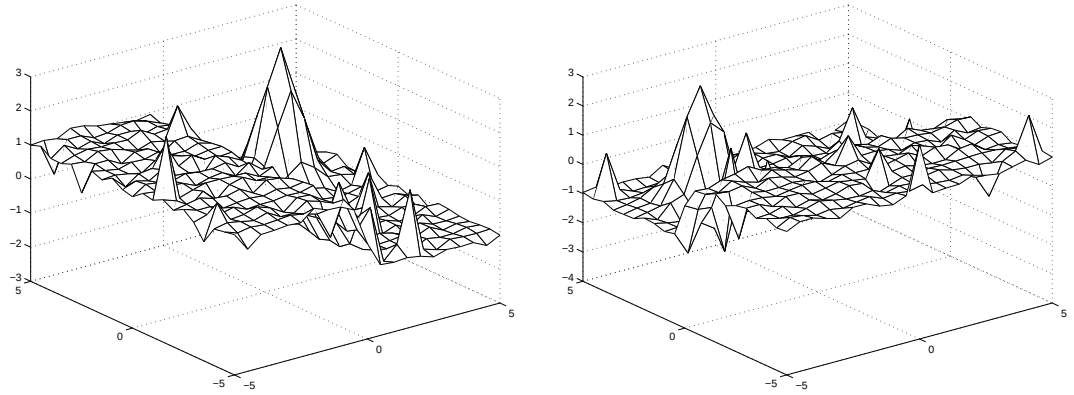

Figure 5: The components of the output-vectors in the training data ($\mathbf{f}_1$ left, $\mathbf{f}_2$ right).

We attempt to reconstruct the function $\mathbf{f}$ in $\Pi = [-5, 5] \times [-5, 5]$. The input-vectors of the training data are obtained by first constructing a uniform grid in Π with the grid points given by

$$(x_i, y_j) = ((i-1)\,h - 5, (j-1)\,h - 5), \quad i, j = 1, \ldots, n_g, \tag{4.6}$$

where $h = 10/n_g$. For the tests, we choose $n_g = 21$ as in [Flake, 1998]. These coordinate values are then prescaled to the range $[-1, 1]$ of the activation functions, and they are included in the input-vector together with the squares of the scaled coordinates. Thus, the input-dimension of the MLP-network becomes $n_0 = 4$.

As in the previous section, all output-vectors involve low-amplitude normally distributed noise, while some isolated outputs are also disturbed by high-amplitude outliers. More precisely, the output corresponding to the input $\mathbf{x}^{i,j} = (x_i, y_j, x_i^2, y_j^2)^T$ is of the form $\mathbf{y}^{i,j} = (\mathbf{y}_1^{i,j}, \mathbf{y}_2^{i,j})^T$ with

$$\mathbf{y}_k^{i,j} = \mathbf{f}_k(x_i, y_j) + \delta\,\varepsilon_{i,j} + \zeta\,\eta_{i,j}, \tag{4.7}$$

where $\varepsilon_{i,j} \sim \mathcal{N}(0, 1)$ and

$$\eta_{i,j} = \begin{cases} \mathcal{U}(-1, 1), & (i, j) \in O, \\ 0, & (i, j) \notin O. \end{cases} \tag{4.8}$$

In the tests, the index set O is chosen to include approximately $0.05\,n_g^2$ randomly selected elements, $\delta = 0.1$, and $\zeta = 2$. The training data created according to the definitions above (before prescaling to the range of the activation function) is illustrated in Figure 5.

Comparison of formulations

We performed tests by using the values 5, 10, and 20 for n_1 with the three different training formulations (2.16), initially without regularization (i.e., $\beta = 0$). The error of the MLP-approximations was measured using a uniform 49×49 validation grid over Π. Let us denote the input-vectors of the validation data by $\hat{\mathbf{x}}^{i,j}$ and the corresponding grid points by $(\hat{x}_i, \hat{y}_i)$. Then,

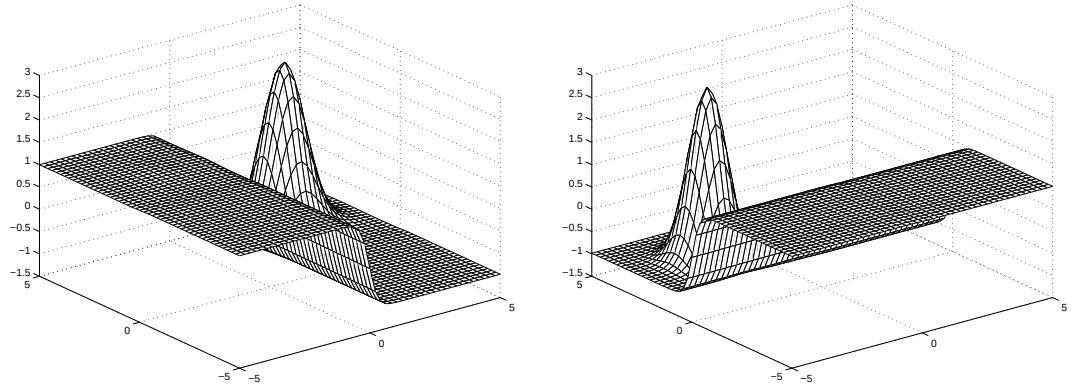

Figure 6: The components of the best reconstruction, which was obtained by minimizing $\mathcal{L}^1_{2,\beta}$ with $n_1 = 5$ ($\mathbf{f}_1$ left, $\mathbf{f}_2$ right).

the error is calculated by

$$\text{err}\,(\mathbf{W}^1, \mathbf{W}^2) = \frac{1}{49^2} \sum_{i=1}^{49} \sum_{j=1}^{49} \left\| \mathbf{W}^2\, \widehat{\mathcal{F}}(\mathbf{W}^1\, \hat{\mathbf{x}}^{i,j}) - \mathbf{f}(\hat{x}_i, \hat{y}_i) \right\|_2 . \tag{4.9}$$

With fixed n_1, we again repeated the optimization algorithm 100 times with random initialization and computed the minimum and average values of the errors (4.9).

The results are collected in Table 2. We conclude that the functionals $\mathcal{L}^1_{1,0}$ and $\mathcal{L}^1_{2,0}$ are more accurate and clearly more robust with respect to the initial guess than the smooth functional $\mathcal{L}^2_{2,0}$. In all cases, the smallest minimum error is achieved with the choice $q = 2\alpha = 2$, while the dimension n_1 does not have a strong effect on the accuracy. The best reconstruction, given by the functional $\mathcal{L}^1_{2,0}$, is illustrated in Figure 6.

Determination of the regularization parameter

In the previous section, the regularization parameter β was equal to zero. However, as already pointed out by the results in Section 4.1, the value of β has a strong effect on the accuracy of results, and thereby, it is important to be able to choose the value correctly. Next, we describe another strategy for choosing the value of β and estimate the quality of the resultant MLP network.

The dimension of the hidden layer is fixed to $n_1 = 20$ and we use the choice $q = 2\alpha = 2$.

	$\mathcal{L}^2_{2,\beta}$		$\mathcal{L}^1_{1,\beta}$		$\mathcal{L}^1_{2,\beta}$	
n_1	err*	$\overline{\text{err}}$	err*	$\overline{\text{err}}$	err*	$\overline{\text{err}}$
5	9.6e-2	2.8e-1	3.7e-2	1.7e-1	3.4e-2	1.6e-1
10	1.2e-1	2.4e-1	4.0e-2	1.5e-1	3.6e-2	1.4e-1
20	1.1e-1	2.3e-1	4.7e-2	1.4e-1	4.4e-2	1.4e-1

Table 2: Minimum and average errors with the functionals $\mathcal{L}^2_{2,\beta}$, $\mathcal{L}^1_{1,\beta}$, and $\mathcal{L}^1_{2,\beta}$.

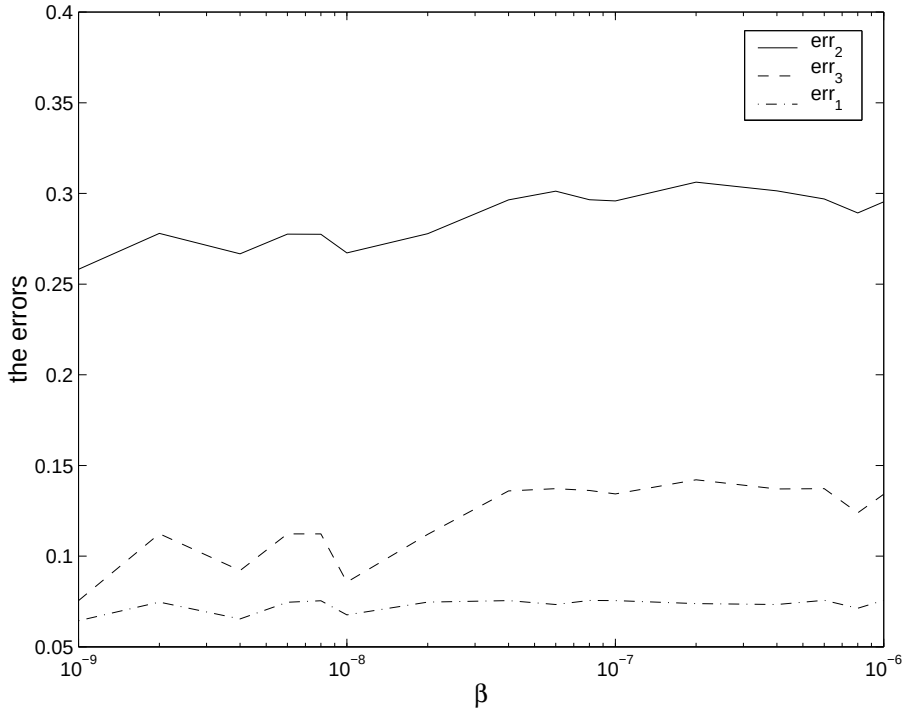

Figure 7: Graphs of the three errors err_k as functions of β.

The learning data is exactly the same as in the previous tests, and it is first divided into two disjoint parts C_1 and C_2 of equal sizes. More precisely, for $N = 441$ the randomly chosen sets C_1 and C_2 contain $N_1 = 221$ and $N_2 = 220$ elements, respectively. Only the input-output pairs $(\mathbf{x}^{i,j}, \mathbf{y}^{i,j})$ in the set C_1 are used in the functional (2.17), while the set C_2 is reserved for testing. This choice originates from the relaxed requirements concerning the necessary amount of data for robust training. We define the two errors

$$\text{err}_k\,(\mathbf{W}^1, \mathbf{W}^2) = \frac{1}{N_k} \sum_{(\mathbf{x}^{i,j}, \mathbf{y}^{i,j}) \in C_k} \left\| \mathbf{W}^2\, \widehat{\mathcal{F}}(\mathbf{W}^1\, \hat{\mathbf{x}}^{i,j}) - \mathbf{y}^{i,j} \right\|_2, \quad k = 1, 2, \tag{4.10}$$

while err_3 refers to the already defined validation error in (4.9).

We search for an optimal nonzero β in the interval $[10^{-9}, 10^{-6}]$, which can be determined by monitoring the value of the regularization term as described within the univariate test. The interval is covered with a predefined set of values $l \cdot 10^{-s},\ l = 1, 2, 4, 6, 8; s = 9, 8, 7$, and, for each fixed β, the optimization algorithm is repeated 50 times with random initialization. We compute the errors err_1 and err_2 in all 50 tests and choose the MLP network with the smallest err_2 to be the best one. For this MLP, we compute also the error err_3.

The results of the first stage of our strategy are illustrated by the graphs in Figure 7. We see that all three curves have similar behavior, which suggests that the errors err_1 and err_2, which can be computed without knowing the exact function $\mathbf{f}$, contain the same information as the true error err_3. Based on err_1 and err_2 we now limit the complexity of MLP by choosing $\beta = 10^{-8}$.

After choosing the value of the regularization parameter we proceed to the second stage of

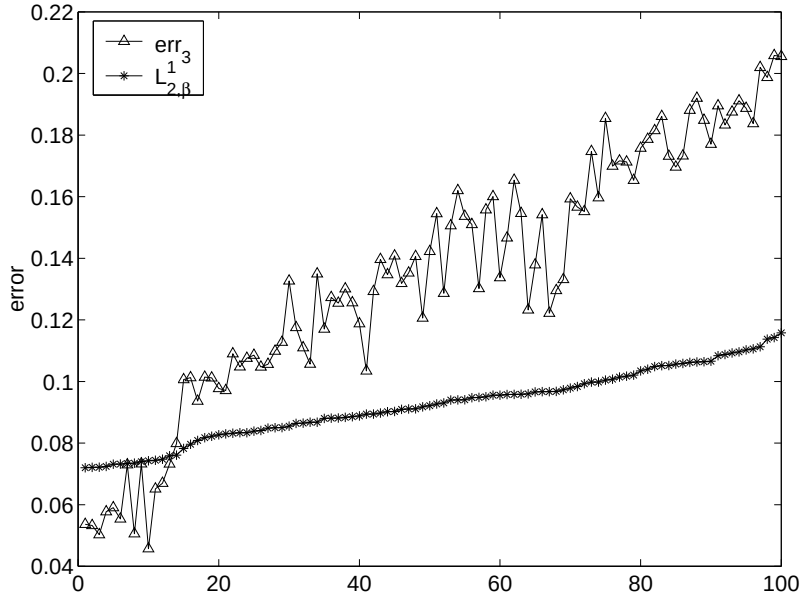

Figure 8: The minimal value of $\mathcal{L}^1_{2,\beta}$ and err_3 in the final 100 tests.

our strategy, which is the determination of the final MLP network. Because the model complexity of MLP is now fixed along with β, we are able to use the whole data in the learning problem. We perform the minimization 100 times and choose the final MLP to be the one which corresponds to the smallest value of local minima for $\mathcal{L}^1_{2,\beta}$ (i.e., the best candidate for global minimum).

We evaluated the quality of the obtained MLP by computing the corresponding err_3, which was approximately 0.05. By comparing this value to the graph of Figure 7 we see that approximation is improved from the first stage and we obtain a very good overall error level. We also computed err_3 in each 100 tests and compared these values to the local minima of $\mathcal{L}^1_{2,\beta}$. To study the correlation of these values and thus the validity of the final choice, we then sorted the 100 tests in ascending order according to $\mathcal{L}^1_{2,\beta}$. The results of this procedure together with the corresponding values of err_3 are given in Figure 8. The increse of err_3 from its smallest value stresses the importance of the last choice, which recovered almost the best alternative among the 100 candidates.

5 Conclusions

We considered robust learning problem formulations for the MLP network with regularization-based pruning. The MLP-transformation was presented in a layer-wise form, which yielded a compact representation of the optimality systems and allowed a straightforward analysis and computer implementation of the proposed techniques.

Different learning problem formulations were tested numerically for studying the effect of noise with outliers. We also proposed and tested two novel strategies for blind determination of the regularization parameter and thus the generality of MLP. Altogether, combination of robust training, square unit augmentation of input and effective control of model complexity yielded very promising computational results with simulated data.

Acknowledgements

The authors would like to thank Professor Hannu Oja and Docent Marko M. Mäkelä for their help during the course of this research.

References

[Broomhead and Lowe, 1988] Broomhead, D. S. and Lowe, D. (1988). Multivariable functional interpolation and adaptive networks. *Complex Systems*, 2(3):321–355.

[Chen and Jain, 1994] Chen, D. S. and Jain, R. C. (1994). A robust backpropagation learning algorithm for function approximation. *IEEE Transactions on Neural Networks*, 5(3):467–479.

[Clarke, 1983] Clarke, F. H. (1983). *Optimization and nonsmooth analysis*. John Wiley & Sons Inc., New York. A Wiley-Interscience Publication.

[Flake, 1998] Flake, G. W. (1998). Square unit augmented, radially extended, multilayer perceptrons. In [Orr and Müller, 1998], pages 145–163.

[Hagan and Menhaj, 1994] Hagan, M. and Menhaj, M. (1994). Training feedforward networks with the Marquardt algorithm. *IEEE Transactions on Neural Networks*, 5(6):989–993.

[Haykin, 1994] Haykin, S. (1994). *Neural Networks; A Comprehensive Foundation*. Macmillan College Publishing Company, New York.

[Hettmansperger and McKean, 1998] Hettmansperger, T. P. and McKean, J. W. (1998). *Robust nonparametric statistical methods*. Edward Arnold, London.

[Huber, 1981] Huber, P. J. (1981). *Robust statistics*. John Wiley & Sons Inc., New York. Wiley Series in Probability and Mathematical Statistics.

[Kärkkäinen, 2002] Kärkkäinen, T. (2002). MLP-network in a layer-wise form with applications to weight decay. *Neural Computation*, 14(6):1451–1480.

[Kärkkäinen and Majava, 2000] Kärkkäinen, T. and Majava, K. (2000). Determination of regularization parameter in monotone active set method for image restoration. In Neittaanmäki, P., Tiihonen, T., and Tarvainen, P., editors, *Proceedings of the Third European Conference on Numerical Mathematics and Advanced Applications*, pages 641–648, Singapore. World Scientific.

[Kärkkäinen et al., 2001] Kärkkäinen, T., Majava, K., and Mäkelä, M. M. (2001). Comparison of formulations and solution methods for image restoration problems. *Inverse Problems*, 17(6):1977–1995.

[Kosko, 1992] Kosko, B. (1992). *Neural Networks and Fuzzy Systems; A Dynamical Systems Approach to Machine Intelligence*. Prentice-Hall, Englewood Cliffs, N.J.

[Liano, 1996] Liano, K. (1996). Robust error measure for supervised neural network learning with outliers. *IEEE Transactions on Neural Networks*, 7(1):246–250.

[Mäkelä and Neittaanmäki, 1992] Mäkelä, M. M. and Neittaanmäki, P. (1992). *Nonsmooth optimization*. World Scientific Publishing Co. Inc., River Edge, NJ. Analysis and algorithms with applications to optimal control.

[Nocedal and Wright, 1999] Nocedal, J. and Wright, S. J. (1999). *Numerical optimization*. Springer-Verlag, New York.

[Oja, 1999] Oja, H. (1999). Affine invariant multivariate sign and rank tests and corresponding estimates: a review. *Scand. J. Statist.*, 26(3):319–343.

[Orr and Müller, 1998] Orr, G. B. and Müller, K.-R., editors (1998). *Neural Networks: Tricks of the Trade*, volume 1524 of *Lecture Notes in Computer Science*, Berlin Heidelberg. Springer-Verlag.

[Prechelt, 1998] Prechelt, L. (1998). Early stopping - but when? In [Orr and Müller, 1998], pages 55–70.

[Rao, 1988] Rao, C. R. (1988). Methodology based on the L_1-norm, in statistical inference. *Sankhyā Ser. A*, 50(3):289–313.

[Rögnvaldsson, 1998] Rögnvaldsson, T. S. (1998). A simple trick for estimating the weight decay parameter. In [Orr and Müller, 1998], pages 71–92.

[Rousseeuw and Leroy, 1987] Rousseeuw, P. J. and Leroy, A. M. (1987). *Robust regression and outlier detection*. John Wiley & Sons Inc., New York.

[Saito and Nakano, 2000] Saito, K. and Nakano, R. (2000). Second-order learning algorithm with squared penalty term. *Neural Computation*, 12(3):709–729.

[Sarajedini and Hecht-Nielsen, 1992] Sarajedini, A. and Hecht-Nielsen, R. (1992). The best of both worlds: Casasent networks integrate multilayer perceptrons and radial basis functions. In *Neural Networks, 1992. IJCNN., International Joint Conference on*, volume 3, pages 905–910. IEEE.

[Sarle, 1997] Sarle, W. (1997). The `comp.ai.neural-nets` frequently asked questions list.

Appendix: Error and regularization figures

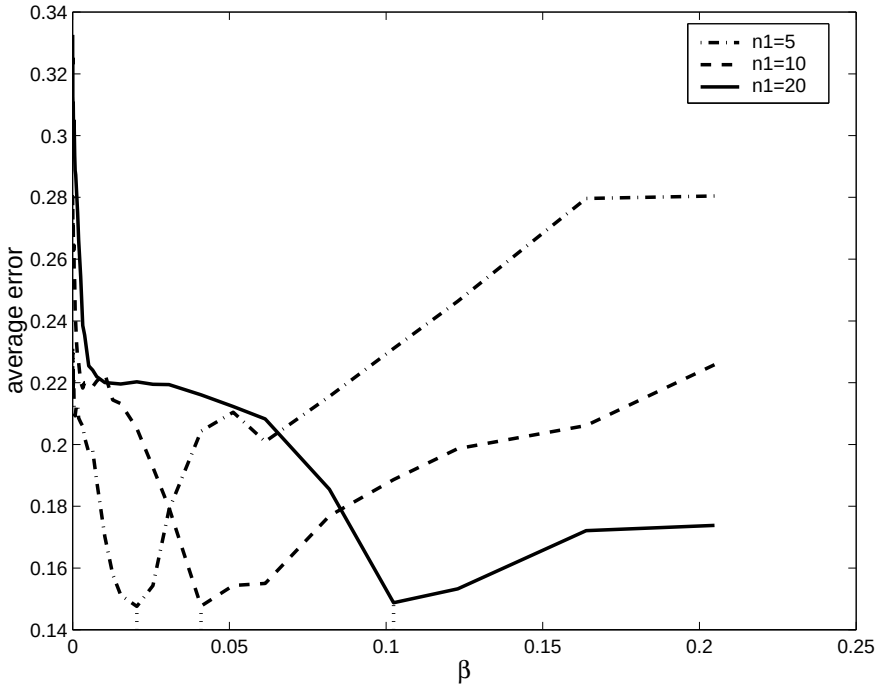

Figure 9: Average error with the functional $\mathcal{L}^1_{1,\beta}$ and $N = 30$.

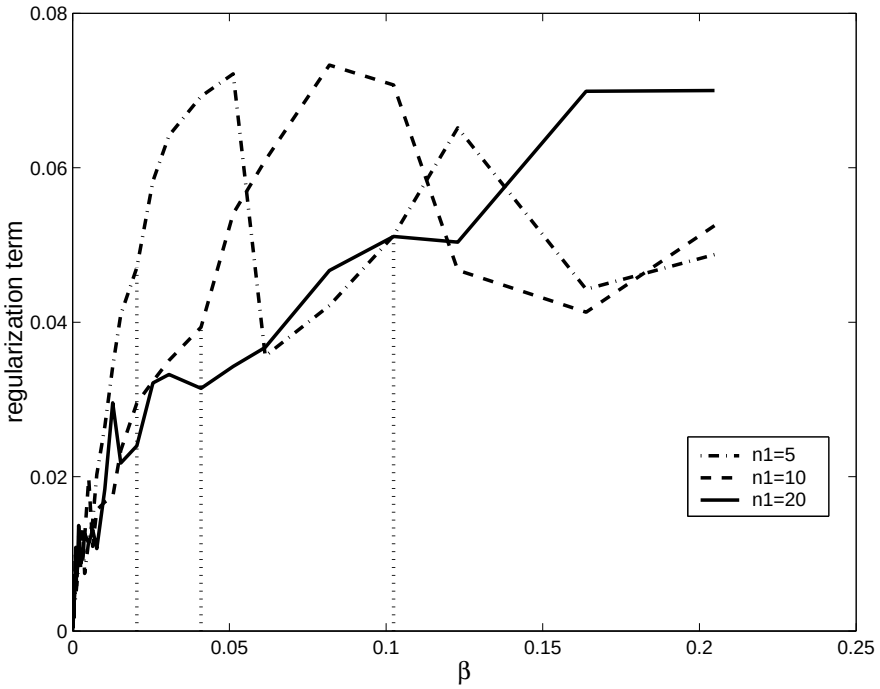

Figure 10: Regularization term with the functional $\mathcal{L}^1_{1,\beta}$ and $N = 30$.

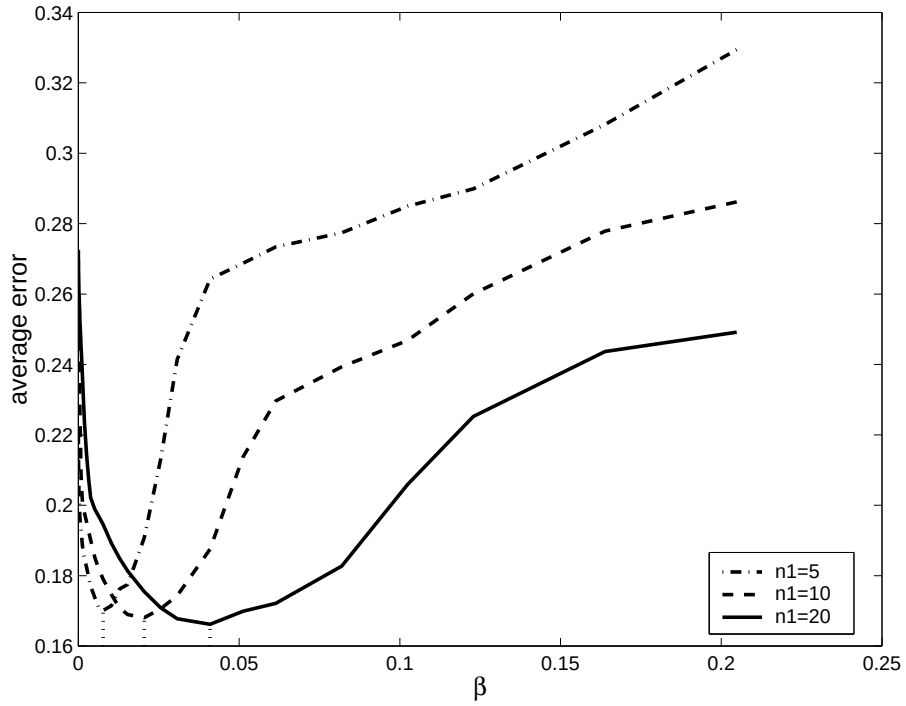

Figure 11: Average error with the functional $\mathcal{L}^2_{2,\beta}$ and $N = 30$.

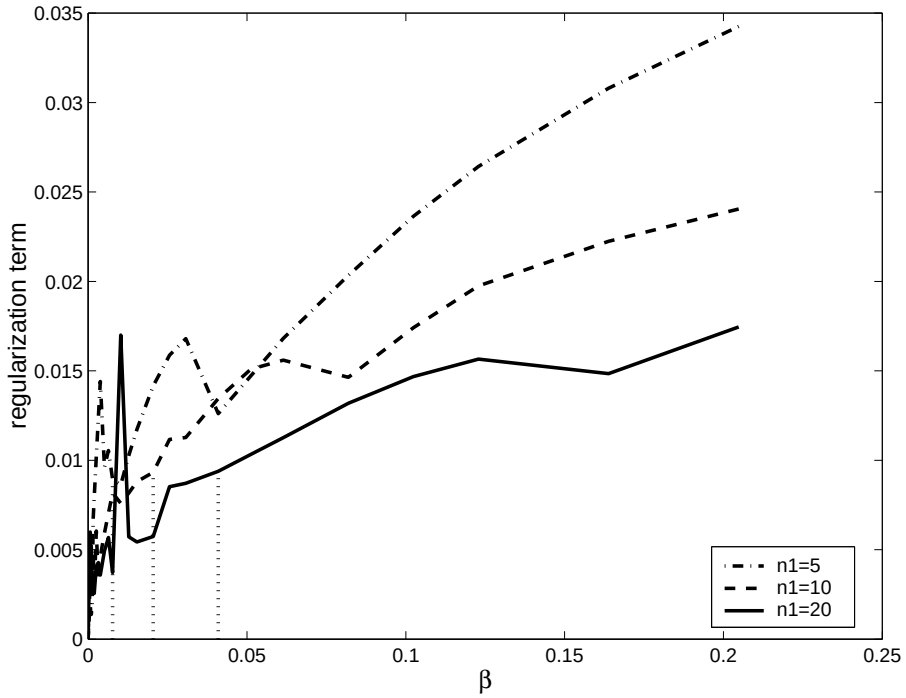

Figure 12: Regularization term with the functional $\mathcal{L}^2_{2,\beta}$ and $N = 30$.

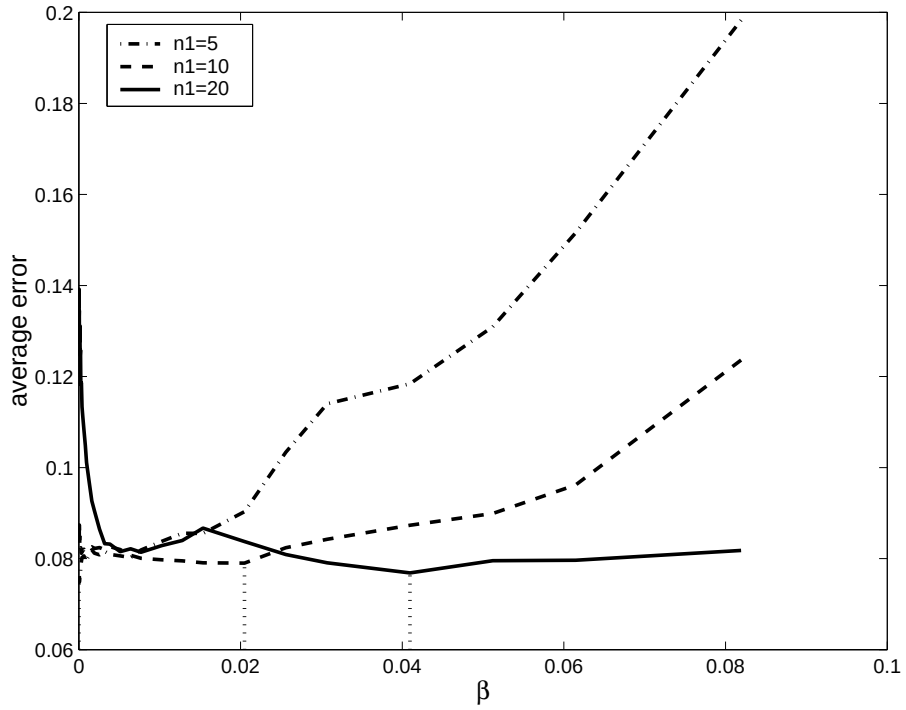

Figure 13: Average error with the functional $\mathcal{L}^1_{1,\beta}$ and $N = 60$.

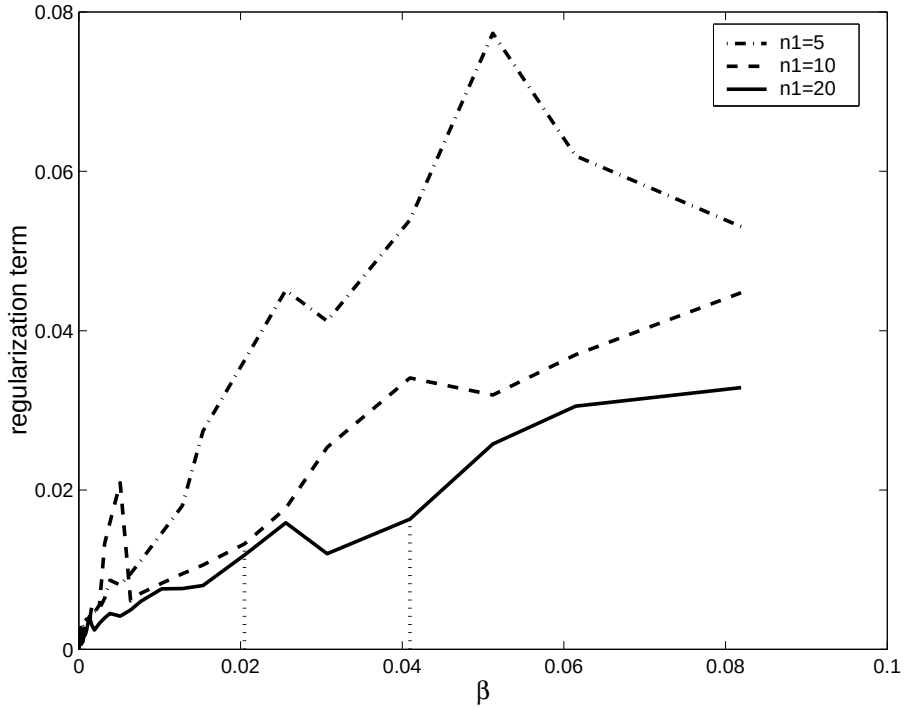

Figure 14: Regularization term with the functional $\mathcal{L}^1_{1,\beta}$ and $N = 60$.

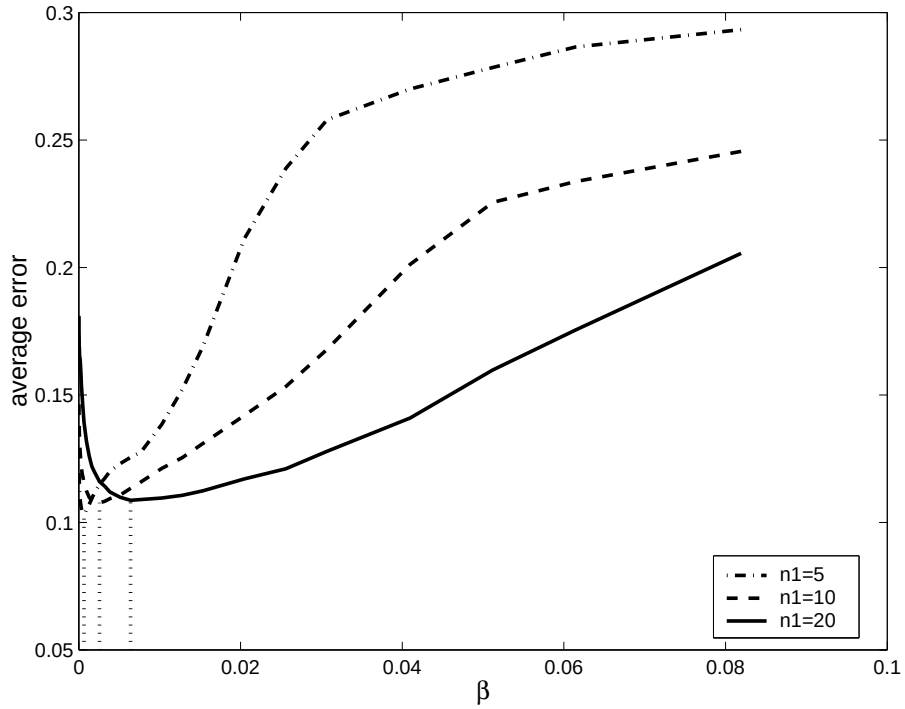

Figure 15: Average error with the functional $\mathcal{L}^2_{2,\beta}$ and $N = 60$.

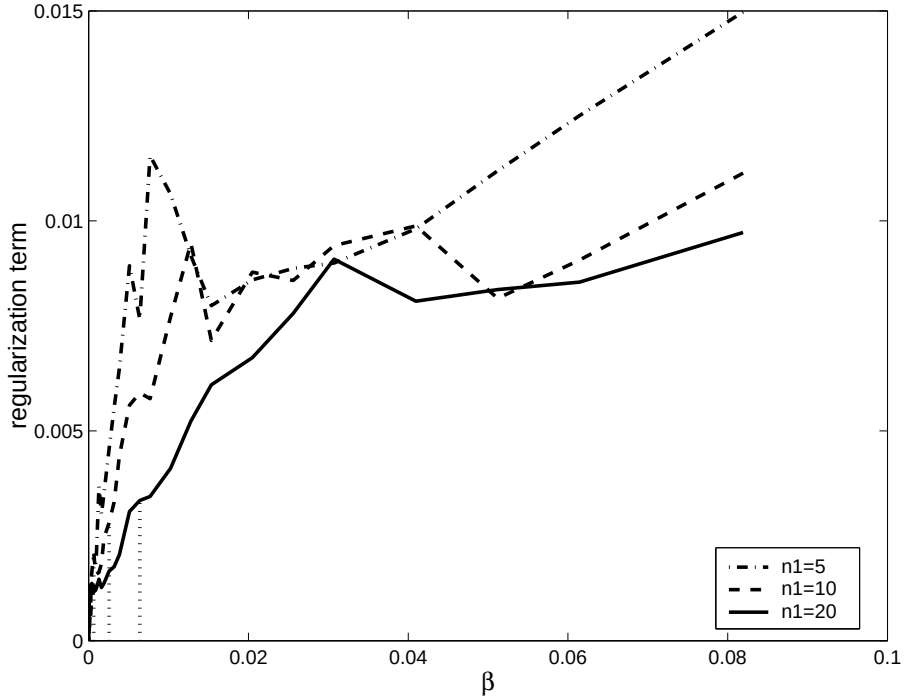

Figure 16: Regularization term with the functional $\mathcal{L}^2_{2,\beta}$ and $N = 60$.

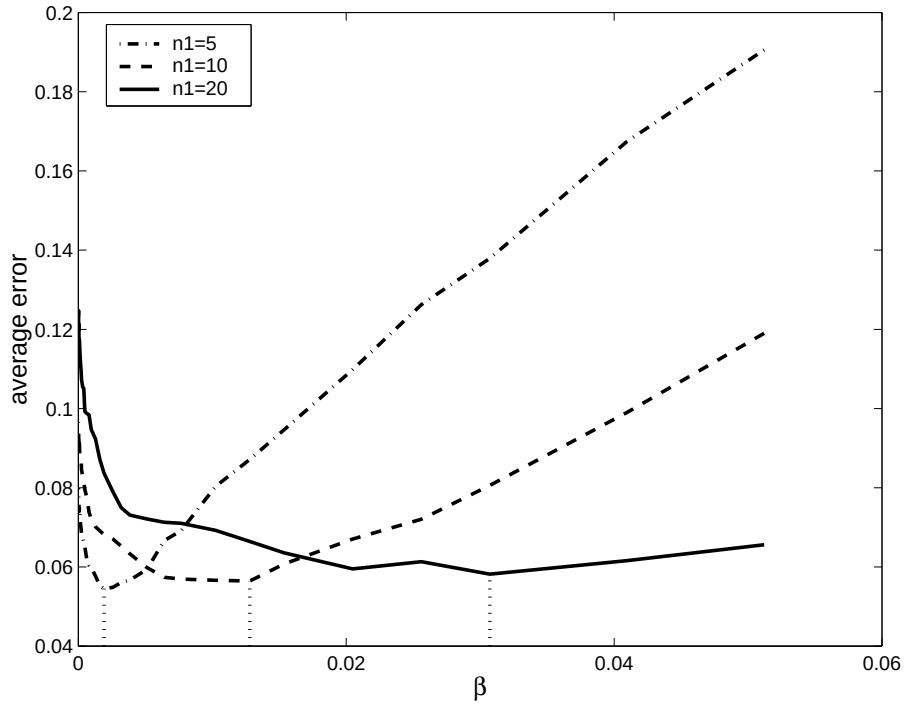

Figure 17: Average error with the functional $\mathcal{L}^1_{1,\beta}$ and $N = 120$.

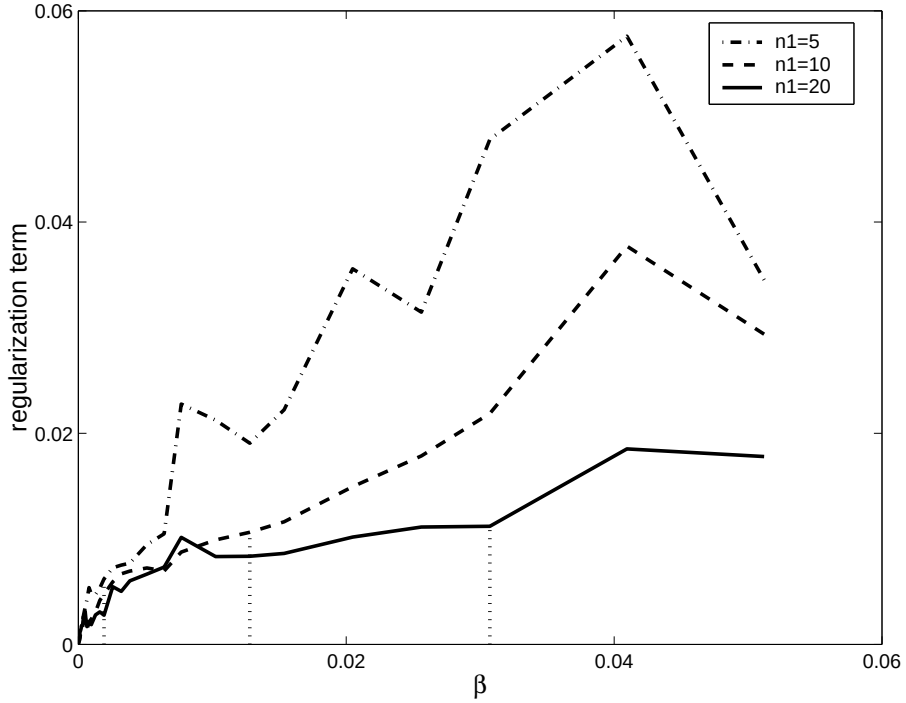

Figure 18: Regularization term with the functional $\mathcal{L}^1_{1,\beta}$ and $N = 120$.

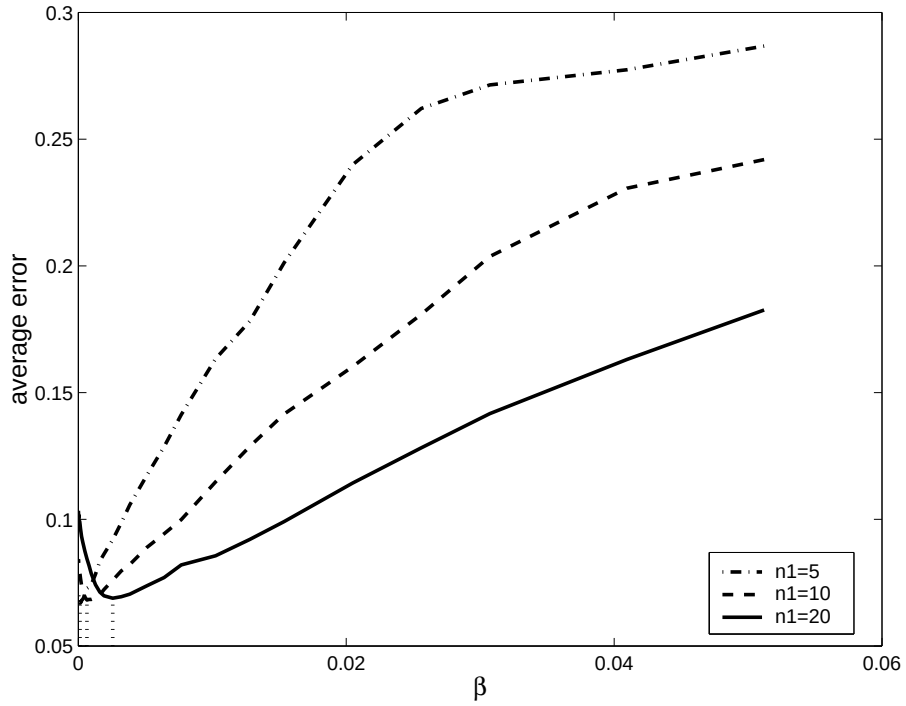

Figure 19: Average error with the functional $\mathcal{L}^2_{2,\beta}$ and $N = 120$.

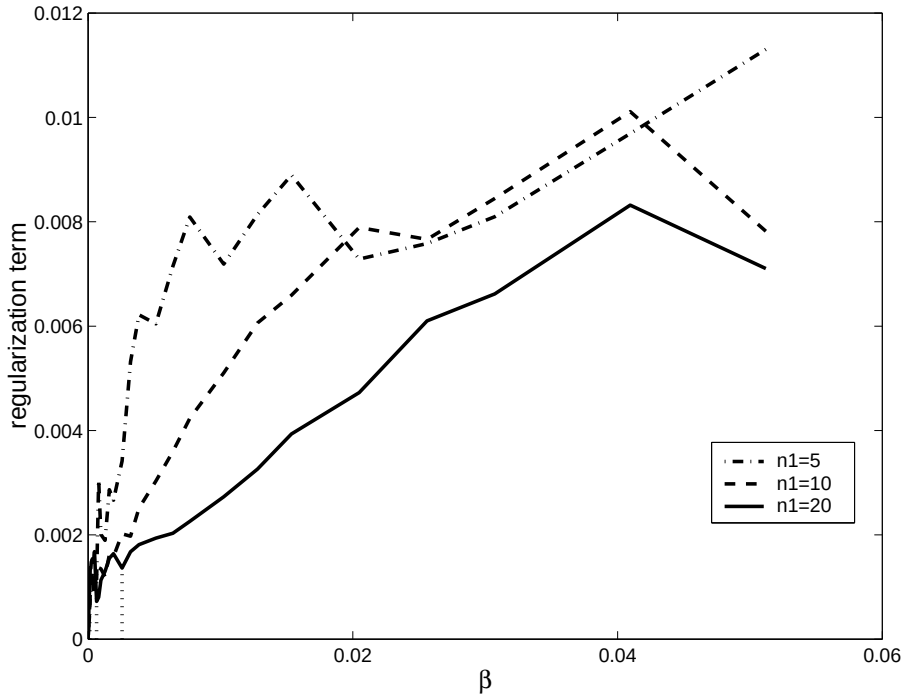

Figure 20: Average error with the functional $\mathcal{L}^2_{2,\beta}$ and $N = 120$.