Electronic Journal of Statistics
Vol. 18 (2024) 553–598
ISSN: 1935-7524
https://doi.org/10.1214/23-EJS2207

Analysis of the rate of convergence of two regression estimates defined by neural features which are easy to implement

Alina Braun[1], **Michael Kohler**[1], **Jeongik Cho**[2], **and Adam Krzyżak**[*2]

[1] *Fachbereich Mathematik*
Technische Universität Darmstadt
Schlossgartenstr. 7, 64289 Darmstadt, Germany
e-mail: braun@mathematik.tu-darmstadt.de; kohler@mathematik.tu-darmstadt.de

[2] *Department of Computer Science and Software Engineering*
Concordia University
1455 De Maisonneuve Blvd. West, Montreal, Quebec, Canada H3G 1M8
e-mail: jeongik.jo.01@gmail.com; krzyzak@cs.concordia.ca

Abstract: Recent results in nonparametric regression have shown that neural network regression estimates with many hidden layers are able to achieve good rates of convergence even in case of high-dimensional predictor variables, provided suitable assumptions on the structure of the regression function are imposed. In those recent results, the estimates were defined by minimizing the empirical L_2 risk over a class of neural networks. In practice, however, it is not clear how this can be done exactly. In this article, motivated by some recent approximation results for neural networks, we introduce two new regression estimates defined by neural features where most of the neural network weights are chosen via random initialization and no training, thus sparing the costly data-dependent optimization. For the first estimate, which is defined by these neural features and an extra layer whose weights are set via least squares, we derive rates of convergence results in case the regression function is smooth. We then combine this estimate with the projection pursuit, where we choose the directions randomly, and we show that for sufficiently many repetitions we get a second regression estimate which achieves the one-dimensional rate of convergence (up to some logarithmic factor) in case that the regression function satisfies the assumptions of projection pursuit. Because the neural features are obtained by random initialization but not training of the weights, the two estimators thus defined are easy to implement.

MSC2020 subject classifications: Primary 62G08; secondary 62M45.

Keywords and phrases: Curse of dimensionality, neural networks, nonparametric regression, rate of convergence, projection pursuit.

Received January 2022.

[*]Corresponding author. Tel: +1-514-848-2424 ext. 3007, Fax: +1-514-848-2830.

1. Introduction

For many years, nonparametric classifiers and regression estimates based on multilayer perceptron (MLP) networks were considered as leading in multivariate statistical classification and regression estimation (see, e.g., the monographs Hertz, Krogh and Palmer [24], Devroye, Györfi and Lugosi [13], Anthony and Bartlett [3], Györfi et al. [19], Haykin [23] and Ripley [40]). In recent years the focus has shifted from MLPs towards deep learning, where networks with many hidden layers are fitted to observed data (see, e.g., Schmidhuber [42] and the literature cited therein).

In this article, we study neural network regression estimates in the context of nonparametric regression with random design. Here, (X, Y) is an $\mathbb{R}^d \times \mathbb{R}$-valued random vector satisfying $\mathbf{E}\{Y^2\} < \infty$, and given a random sample of (X, Y) of size n, i.e., given a data set

$$\mathcal{D}_n = \big\{(X_1, Y_1), \ldots, (X_n, Y_n)\big\}, \tag{1}$$

where $(X_1, Y_1), \ldots, (X_n, Y_n)$ are independent identically distributed (*i.i.d.*) random variables with the same distribution as (X, Y), the aim is to construct an estimate

$$m_n(\cdot) = m_n(\cdot, \mathcal{D}_n) : \mathbb{R}^d \to \mathbb{R}$$

of the regression function $m : \mathbb{R}^d \to \mathbb{R}$, defined as $m(x) = \mathbf{E}\{Y|X = x\}$, such that the L_2 error

$$\int |m_n(x) - m(x)|^2 \mathbf{P}_X(dx)$$

is "small" (see, e.g., Györfi et al. [19] for a systematic introduction to nonparametric regression and a motivation for the L_2 error).

It is well-known that smoothness assumptions on the regression function are needed in order to derive non-trivial rate of convergence results for nonparametric regression estimates (cf., e.g., Theorem 7.2 and Problem 7.2 in Devroye, Györfi and Lugosi [13] and Section 3 in Devroye and Wagner [14]). To do this, we will use the following definition.

Definition 1. Given any $p > 0$ and $C > 0$ let $p = q + s$ for some $q \in \mathbb{N}_0$ and $0 < s \leq 1$, i.e., $q = \lceil p \rceil - 1$ and $s = 1 + p - \lceil p \rceil$, where $\mathbb{N}_0$ is the set of nonnegative integers. A **function** $f : \mathbb{R}^d \to \mathbb{R}$ is called (q, s, C)**-smooth**, if for every $\alpha = (\alpha_1, \ldots, \alpha_d) \in \mathbb{N}_0^d$ with $\sum_{j=1}^d \alpha_j = q$ the partial derivative $\frac{\partial^q f}{\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}}$ exists and satisfies

$$\left| \frac{\partial^q f}{\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}}(x) - \frac{\partial^q f}{\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}}(z) \right| \leq C \cdot \|x - z\|^s$$

for all $x, z \in \mathbb{R}^d$, where $\| \cdot \|$ denotes the Euclidean norm.

The functions satisfying Definition 1 belong to a class of Hölder functions, see Adams and Fournier [1]. From now onwards we will call (q, s, C)**-smooth** functions (p, C)**-smooth** functions, which is a standard notation in statistical

literature, with $p = q + s$ as described above. Stone [44] showed that the optimal minimax rate of convergence in nonparametric regression for (p, C)-smooth functions is $n^{-2p/(2p+d)}$. In case that d is large compared to p this rate of convergence is rather slow (so-called curse of dimensionality). One way to circumvent this curse of dimensionality is to impose additional constraints on the structure of the regression function. Stone [45] assumed that the regression function is additive, i.e., that $m : \mathbb{R}^d \to \mathbb{R}$ satisfies

$$m\big(x^{(1)}, \dots, x^{(d)}\big) = m_1\big(x^{(1)}\big) + \cdots + m_d\big(x^{(d)}\big) \quad \big(x^{(1)}, \dots, x^{(d)} \in \mathbb{R}\big)$$

for some (p, C)-smooth univariate functions $m_1, \dots, m_d : \mathbb{R} \to \mathbb{R}$, and showed that, in this case, suitably defined spline estimates achieve the corresponding univariate rate of convergence. Stone [46] extended this results to interaction models, where the regression function is assumed to be a sum of functions applied to at most $d^* < d$ components of x and showed in this case that suitably defined spline estimates achieve the d^*-dimensional rate of convergence.

Barron [6, 7] proved a dimension-free rate of $n^{-1/2}$ (up to some logarithmic factor), provided the Fourier transform of the regression function has a finite first moment. This condition basically requires that the function becomes smoother with increasing dimension d of X. Furthermore, the estimates achieving this dimension-free rate of convergence were defined as the least squares estimates; however, it is not clear how to compute these nonlinear least squares estimates in practice because minimization of high dimensional nonlinear cost functions to find the global minimum typically does not lead to analytical solutions and computational cost of approximating such minima is typically prohibitively high.

Other classes of functions which enable us to achieve a better rate of convergence results include single index models (also called ridge functions in some literature), where

$$m(x) = g\big(a^T x\big) \quad \big(x \in \mathbb{R}^d\big)$$

for some $a \in \mathbb{R}^d$ and $g : \mathbb{R} \to \mathbb{R}$ (cf., e.g., Härdle and Stoker [21], Härdle, Hall and Ichimura [22], Yu and Ruppert [48], Kong and Xia [36] and Lepski and Serdyukova [37]); and projection pursuit, where

$$m(x) = \sum_{l=1}^{r} g_l\big(a_l^T x\big) \quad \big(x \in \mathbb{R}^d\big)$$

for some $r \in \mathbb{N}$, $a_l \in \mathbb{R}^d$ and $g_l : \mathbb{R} \to \mathbb{R}$ $(l = 1, \dots, r)$ (cf., e.g., Friedman and Stuetzle [18], Huber [26], Jones [29, 30], Hall [20], Zhao and Atkeson [49] and Ben-Ari and Steinberg [10]). In Section 22.3 in Györfi et al. [19] it is shown that suitably defined (nonlinear) least squares estimates in a (p, C)-smooth projection pursuit model achieve the univariate rate of convergence $n^{-2p/(2p+1)}$ up to some logarithmic factor.

A generalization of projection pursuit was considered in Horowitz and Mammen [25], who studied the case of a regression function which satisfies

$$m(x) = g\left(\sum_{l_1=1}^{L_1} g_{l_1}\left(\sum_{l_2=1}^{L_2} g_{l_1,l_2}\left(\cdots \sum_{l_r=1}^{L_r} g_{l_1,\dots,l_r}\big(x^{l_1,\dots,l_r}\big)\right)\right)\right),$$

where $g, g_{l_1}, g_{l_1,l_2}, \ldots, g_{l_1,\ldots,l_r}$ are (p, C)-smooth univariate functions and $x^{l_1,\ldots,l_r}$ are single components of $x \in \mathbb{R}^d$ (not necessarily different for different indices $(l_1, \ldots, l_r)$), was studied. With the use of a penalized least squares estimate, the rate $n^{-2p/(2p+1)}$ was proven.

The estimates in Horowitz and Mammen [25] and the one for projection pursuit in Section 22.3 in Györfi et al. [19] are nonlinear (penalized) least squares estimates; therefore, it is unclear how they can be computed exactly in practice. Friedman and Stuetzle [18] described easily implementable estimates for projection pursuit, but in their definition several heuristic simplifications are used, and as a consequence it is unclear whether for these estimates any rate of convergence result can be shown.

Recently it was shown in several papers that neural networks can achieve dimensionality reduction in case the regression function is a composition of (sums of) functions, where each of the functions is a function of at most $d^* < d$ variables. The first paper in this respect was Kohler and Krzyżak [33], where it was shown that in this case suitably defined multilayer neural networks achieve the rate of convergence $n^{-2p/(2p+d^*)}$ (up to some logarithmic factor) in case $p \leq 1$. Bauer and Kohler [9] showed that this result even holds for $p > 1$ provided the activation function is suitably chosen. Schmidt-Hieber [43] obtained similar results for neural networks with ReLU activation function, and Kohler and Langer [35] showed that the results of Bauer and Kohler [9] also hold for very simply constructed fully connected feedforward neural networks. In Kohler, Krzyżak and Langer [34] it was demonstrated that neural networks are able to circumvent the curse of dimensionality in case the regression function has a low local dimensionality. Results concerning estimation by neural networks of regression functions which are piecewise polynomials with partitions with rather general smooth boundaries have been derived in Imaizumi and Fukamizu [27].

In all articles cited above the neural network regression estimate is defined as a nonlinear least squares estimate. For instance, an estimate is defined as the function $m_n \in \mathcal{F}$ which minimizes the empirical L_2 risk

$$\frac{1}{n} \sum_{i=1}^{n} |Y_i - m_n(X_i)|^2 \tag{2}$$

over a class $\mathcal{F}$ of neural networks. In practice, it is usually not possible to find the global minimum of the empirical L_2 risk over a class of neural networks, due to their nonlinear nature, and usually one tries to find a local minimum using gradient-based optimization combined with backpropagation to compute the gradients of the empirical loss functional.

There exist quite a few papers which try to show that neural network regression estimates learned by backpropagation have nice theoretical properties. The most popular approach in this context is the so-called loss landscape approach. Choromanska et al. [12] used random matrix theory to derive a heuristic argument showing that the risk of most of the local minima of the empirical L_2 risk is not much larger than the risk of the global minimum. For neural networks with special activation function it was possible to validate this claim;

for instance, Arora et al. [4], Kawaguchi [31], and Du and Lee [15] have analyzed gradient descent for neural networks with linear or quadratic activation function. But for such neural networks there do not exist good approximation results; consequently, one cannot derive from these results rates of convergence comparable to the ones discussed above for the least squares neural network regression estimates.

Du et al. [16] analyzed gradient descent applied to neural networks with one hidden layer in case of an input X with a Gaussian distribution. They used the expected gradient instead of the gradient in their gradient descent routine, and therefore, their result cannot be used to derive a rate of convergence result similar to the results for the least squares neural network estimates cited above for an estimate learned by the gradient descent. Liang et al. [38] applied gradient descent to a modified loss function in classification, where it is assumed that the data can be interpolated by a neural network. Here, the last assumption is not satisfied in nonparametric regression and it is unclear whether the main idea (of simplifying the estimation by a modification of the loss function) can also be used in a regression setting. Recently it was shown in several papers, see, e.g., Allen-Zhu, Li and Song [2], Kawaguchi and Huang [32] and the literature cited therein, that gradient descent leads to a small empirical L_2 risk in over-parametrized neural networks. Here, it is unclear what the L_2 risk of the estimate is; while a bound on this term is necessary in order to derive results like the ones cited above for the least squares neural network regression estimates. In particular, due to the fact that the networks are over-parametrized, a bound on the empirical L_2 risk might be not useful for bounding the L_2 risk. Notice that the bound on the L_2 risk presented in Kawaguchi and Huang [32] requires that the weights in the network be small, and it is not clear whether this will be satisfied in an over-parametrized neural network learned by gradient descent.

Although the results discussed above for the least squares neural network estimates are interesting from the theoretical point of view, there is a big gap between the estimates for which there exists a result proving the above mentioned nice rate of convergence, and the estimates which can be computed in practice. Until now, the results derived in the literature for neural networks trained by backpropagation are unfortunately not strong enough to narrow this gap.

In this paper we are interested in the question of narrowing the gap between theory and practice for neural network regression estimates. We propose defining a neural network regression estimate that is possible to be implemented in practice, and which is backed by a theoretical rate of convergence result. This question was already considered in Braun, Kohler and Walk [11] who studied neural network regression estimates with one hidden layer, where the weights were chosen by minimizing a regularized empirical L_2 risk via backpropagation with starting values chosen repeatedly randomly from a special structure adapted to projection pursuit. It was shown in a (p, C)-smooth projection pursuit model, i.e., in a projection pursuit model with (p, C)-smooth functions, that this easily implementable estimate achieves (up to a logarithmic factor) the rate of convergence $n^{-2p/(2p+1)}$, provided $p \leq 1$.

In the sequel we use a different (but related) approach in order to derive the rate of convergence results for easily implementable neural network regression estimates. We use neural networks with many hidden layers, hence they can rightly be called deep neural networks; but unlike the typical deep learning practice, our strategy consists of choosing the weights of the inner layers of the network by random initialization and no training (which is therefore a data-independent way of choosing those weights); and learning the weights of the output layer via regularized least squares estimates. Here, the choice of the inner weights by random initialization alone is motivated by recent approximation results derived for deep neural networks (cf. e.g., Eckle and Schmidt-Hieber [17], Jiao et al. [28], Lu et al. [39] and Yarotsky [47]); and the use of the regularized least squares criterion for learning the weights of the output layer leads to estimates which are easy to implement because they can be computed by solving a linear equation system and that is what we mean by "easy to implement" claim. We would like to stress that "easy to implement" claim for the two neural network regression estimates defined in Section 2 and Section 3 has nothing to do with a class of estimated regression functions.

Our first main novel contribution is that we define our neural network regression estimates in Section 2 in such a way that they are easy to implement and, in addition, they achieve the same rate of convergence as linear regression estimates (e.g., kernel or spline estimates), i.e., they achieve (up to some logarithmic factor) the optimal minimax rate of convergence $n^{-2p/(2p+d)}$ in case of a (p, C)-smooth regression function, for any $p > 0$.

Our second main novel contribution is introduction in Section 3 of a projection pursuit model neural network regression estimate, in which we repeatedly choose the directions of projection pursuit randomly and define the inner weights independent of the data using these random directions, and where the weights of the output layer are computed by using the regularized least squares criterion. For this estimate we show that for sufficiently many repetitions (of the choices of the random directions) we get an estimate which achieves the one-dimensional rate of convergence (up to some logarithmic factor) in case that the regression function satisfies the assumptions of the projection pursuit. To the best of our knowledge this result is the first result in the literature which shows that there exist estimates which can be easily implemented and which achieve (up to a logarithmic factor) the rate of convergence $n^{-2p/(2p+1)}$ in a (p, C)-smooth projection pursuit model for arbitrary $p > 0$.

Throughout the paper, the following notation is used: The sets of positive integers ($\{1, 2, \ldots\}$), nonnegative integers ($\{0, 1, 2, \ldots\}$), and real numbers are denoted by $\mathbb{N}$, $\mathbb{N}_0$ and $\mathbb{R}$, respectively. For $z \in \mathbb{R}$, we denote by $\lceil z \rceil$ the 'ceiling' of z, which is defined to be the smallest integer greater than or equal to z. Furthermore we set $z_+ = \max\{z, 0\}$. The Euclidean norm of $x \in \mathbb{R}^d$ is denoted by $\|x\|$ and $\|x\|_\infty$ denotes its supremum norm. For $f : \mathbb{R}^d \to \mathbb{R}$

$$\|f\|_\infty = \sup_{x \in \mathbb{R}^d} |f(x)|$$

is its supremum norm. Let $\mathcal{F}$ be a set of functions $f : \mathbb{R}^d \to \mathbb{R}$, let $x_1, \ldots, x_n \in$

$\mathbb{R}^d$ and set $x_1^n = (x_1, \ldots, x_n)$. Let $\varepsilon \geq 0$. A finite collection $f_1, \ldots, f_N : \mathbb{R}^d \to \mathbb{R}$ is called an ε-cover of $\mathcal{F}$ on x_1^n if for any $f \in \mathcal{F}$ there exists $i \in \{1, \ldots, N\}$ such that

$$\frac{1}{n} \sum_{k=1}^{n} |f(x_k) - f_i(x_k)| < \varepsilon.$$

The ε-covering number of $\mathcal{F}$ on x_1^n is the size N of the smallest ε-cover of $\mathcal{F}$ on x_1^n and is denoted by $\mathcal{N}_1(\varepsilon, \mathcal{F}, x_1^n)$.

The outline of this paper is as follows: In Section 2 the newly proposed neural network regression estimates for (p, C)-smooth regression functions are defined and a result for the rate of convergence of these estimates is presented. In Section 3 we describe how these estimates can be combined with projection pursuit, and present a rate of convergence result where the easily computable estimate achieves (up to some logarithmic factor) the optimal one-dimensional rate of convergence if the regression function satisfies the assumptions of projection pursuit. The finite sample size performance of our newly proposed estimates on simulated data is illustrated in Section 4. The proofs are given in Section 5.

2. A first estimate: random neural features

In this section, we assume that the regression function is (p, C)-smooth. Defining a neural network requires the choice of an activation function $\sigma : \mathbb{R} \to \mathbb{R}$. Here, we use in the sequel squashing functions, which are nondecreasing and satisfy $\lim_{x \to -\infty} \sigma(x) = 0$ and $\lim_{x \to \infty} \sigma(x) = 1$. An example of a squashing function is the sigmoidal or logistic squasher

$$\sigma(x) = \frac{1}{1 + \exp(-x)} \quad (x \in \mathbb{R}). \tag{3}$$

Notice, however, that other squashing functions are considered in the literature (cf. Györfi et al. [19]). The neural network architecture $(L, \mathbf{k})$ depends on a positive integer L indicating the number of hidden layers, and a widths profile $\mathbf{k} = (k_1, \ldots, k_L) \in \mathbb{N}^L$ that describes the number of neurons in the first, second, ..., L-th hidden layer. Recall that inputs are d-dimensional. For a given choice of architecture $(L, \mathbf{k})$ and sigmoid function σ, we consider a real-valued function $f : \mathbb{R}^d \to \mathbb{R}$ defined by

$$f(x) = \sum_{i=1}^{k_L} c_i^{(L)} \cdot f_i^{(L)}(x) + c_0^{(L)} \tag{4}$$

for some coefficients $c_0^{(L)}, c_1^{(L)}, \ldots, c_{k_L}^{(L)} \in \mathbb{R}$; where the functions $f_i^{(L)} : \mathbb{R}^{k_{L-1}} \to \mathbb{R}$ are defined recursively by

$$f_i^{(r)} = \sigma\left(\sum_{j=1}^{k_{r-1}} c_{i,j}^{(r-1)} \cdot f_j^{(r-1)}(x) + c_{i,0}^{(r-1)} \right) \quad (r = 2, \ldots, L; i = 1, \ldots, k_r) \tag{5}$$

$$f_i^{(1)} = \sigma\left(\sum_{j=1}^{d} c_{i,j}^{(0)} \cdot x^{(j)} + c_{i,0}^{(0)}\right) \quad (r = 1; i = 1, \ldots, k_1) \tag{6}$$

for some coefficients $c_{i,0}^{(0)}, c_{i,1}^{(0)}, \ldots, c_{i,d}^{(0)} \in \mathbb{R}$ for the first hidden layer ($r = 1$) which acts on the input features $(x^{(1)}, \ldots, x^{(d)})$, and coefficients $c_{i,0}^{(r-1)}, c_{i,1}^{(r-1)}, \ldots, c_{i,k_{r-1}}^{(r-1)} \in \mathbb{R}$ for the other hidden layers ($r = 2, \ldots, L$). Notice that if we set $k_0 = d$ then we can unify the notation for widths to encompass the input layer (i.e. the layer of input features). The coefficients for terms $1, \ldots, k_r$ in the above sums are called 'weights' and the coefficients for the shift term (subscript 0) are called 'biases' in the deep learning literature. In the sequel we will write 'weights' to refer to all of these coefficients, for simplicity.

Formally, what we have described above defines what is called a multilayer feedforward neural network (see, e.g., Haykin [23]). In our definition, as described above, all the layers with indices $1, \ldots, L$ are 'hidden layers' and there is an extra layer on top of them which would corresponding to index $L + 1$, but the latter has been omitted in our notation, for simplicity. Notice that the said extra layer consists of a linear (plus shift) combination of functions from the L-th layer, as is evident from Eq. (4), and it completes the recursion to define the function $f : \mathbb{R}^d \to \mathbb{R}$ that maps d-dimensional input features to real-valued outputs. Notice also that all the hidden layers use the nonlinearity σ, while the output layer is linear (plus shift).

In this work we want to use the data (1) in order to choose the weights of the neural network such that the resulting function defined by (4)–(6) is a good estimate of the regression function. In particular, we aim for regression estimates defined in terms of neural networks which are easy to implement in practice, and theoretical results for these estimates that establish good rate of convergence to the regression function. To achieve both goals, we propose fixing an architecture $(L, \mathbf{k})$ and setting all the hidden layer weights in a data-free manner, and then learning the weights in the output layer using the data (1) together with the principle of (regularized) least squares.

Remark on constants Please note that there are many constants used in this work. These constants might be different in different parts of the paper; and may depend on input dimension and smoothness of estimated functions, but they do not depend on the sample size n.

2.1. Definition of the network architecture

Let $a > 0$ be fixed. The choice of the network architecture and of the values of most of the weights of our neural network is motivated by the following approximation result of a (p, C)-smooth function for $x \in [-a, a]^d$ by a local convex combination of Taylor polynomials: For $M \in \mathbb{N}$ and $\mathbf{i} = (i^{(1)}, \ldots, i^{(d)}) \in \{0, \ldots, M\}^d$ set

$$x_{\mathbf{i}} = \left(-a + i^{(1)} \cdot \frac{2a}{M}, \ldots, -a + i^{(d)} \cdot \frac{2a}{M}\right)$$

and let

$$\{\mathbf{i}_1, \dots, \mathbf{i}_{(M+1)^d}\} = \{0, \dots, M\}^d.$$

For $k \in \{1, \dots, (M+1)^d\}$ let

$$p_{\mathbf{i}_k}(x) = \sum_{\substack{j_1,\dots,j_d \in \{0,\dots,q\} \\ j_1+\cdots+j_d \le q}} \frac{1}{j_1! \cdots j_d!} \cdot \frac{\partial^{j_1+\cdots+j_d} f}{\partial^{j_1} x^{(1)} \cdots \partial^{j_d} x^{(d)}}(x_{\mathbf{i}_k}) \cdot \left(x^{(1)} - x^{(1)}_{\mathbf{i}_k}\right)^{j_1} \cdots \left(x^{(d)} - x^{(d)}_{\mathbf{i}_k}\right)^{j_d}$$

be the Taylor polynomial of f with order q around $x_{\mathbf{i}_k}$ and set

$$P(x) = \sum_{k=1}^{(M+1)^d} p_{\mathbf{i}_k}(x) \prod_{j=1}^{d} \left(1 - \frac{M}{2a} \cdot |x^{(j)} - x^{(j)}_{\mathbf{i}_k}|\right)_+, \tag{7}$$

where $z_+ = \max\{z, 0\}$ ($z \in \mathbb{R}$). Since $P(x)$ is a local convex combination of Taylor polynomials of f, it is possible to show that for a (p, C)-smooth function m we have

$$\sup_{x \in [-a,a]^d} |m(x) - P(x)| \le c_1 \cdot a^p \cdot \frac{1}{M^p} \tag{8}$$

(cf. Lemma B1 and its proof in Appendix B of the Supplemental Content of Schmidt-Hieber [43]).

We use the fact that $P(x)$ can be written

$$\sum_{k=1}^{(M+1)^d} \sum_{\substack{j_1,\dots,j_d \in \{0,\dots,q\} \\ j_1+\cdots+j_d \le q}} a_{\mathbf{i}_k, j_1, \dots, j_d} \cdot \left(x^{(1)} - x^{(1)}_{\mathbf{i}_k}\right)^{j_1} \cdots \left(x^{(d)} - x^{(d)}_{\mathbf{i}_k}\right)^{j_d} \times \prod_{j=1}^{d} \left(1 - \frac{M}{2a} \cdot |x^{(j)} - x^{(j)}_{\mathbf{i}_k}|\right)_+$$

with appropriately chosen $a_{\mathbf{i}_k, j_1, \dots, j_d} \in \mathbb{R}$. Our main insight in the sequel is to define appropriate neural networks $f_{net, j_1, \dots, j_d, \mathbf{i}_k}$ which approximate the functions

$$x \mapsto \left(x^{(1)} - x^{(1)}_{\mathbf{i}_k}\right)^{j_1} \cdots \left(x^{(d)} - x^{(d)}_{\mathbf{i}_k}\right)^{j_d} \prod_{j=1}^{d} \left(1 - \frac{M}{2a} \cdot |x^{(j)} - x^{(j)}_{\mathbf{i}_k}|\right)_+,$$

and to choose the network architecture such that neural networks of the form

$$\sum_{k=1}^{(M+1)^d} \sum_{\substack{j_1,\dots,j_d \in \{0,\dots,q\} \\ j_1+\cdots+j_d \le q}} a_{\mathbf{i}_k, j_1, \dots, j_d} \cdot f_{net, j_1, \dots, j_d, \mathbf{i}_k}(x) \quad (a_{\mathbf{i}_k, j_1, \dots, j_d} \in \mathbb{R})$$

are contained in it. To do this, we let $\sigma(x) = 1/(1+\exp(-x))$ $(x \in \mathbb{R})$ be the logistic squasher, choose $R \geq 1$ and define the following functions:

$$f_{id}(x) = 4R \cdot \sigma\left(\frac{x}{R}\right) - 2R \tag{9}$$

which approximates the identity function $f(x) = x$ for $x \in \mathbb{R}$ (cf. Lemma 1 below),

$$\begin{aligned} f_{mult}(x,y) = \frac{R^2}{4} \cdot \frac{(1+e^{-1})^3}{e^{-2}-e^{-1}} \cdot \Bigg(&\sigma\left(\frac{2(x+y)}{R}+1\right) - 2\cdot\sigma\left(\frac{x+y}{R}+1\right) \\ &- \sigma\left(\frac{2(x-y)}{R}+1\right) + 2\cdot\sigma\left(\frac{x-y}{R}+1\right)\Bigg) \end{aligned} \tag{10}$$

which approximates the function $f(x,y) = x \cdot y$ for $x, y \in \mathbb{R}$ (cf. Lemma 2 below),

$$f_{ReLU}(x) = f_{mult}\big(f_{id}(x), \sigma(R \cdot x)\big) \tag{11}$$

which approximates $f(x) = x_+$ for $x \in \mathbb{R}$ (cf. Lemma 3 below), and

$$\begin{aligned} f_{hat,y}(x) = f_{ReLU}&\left(\frac{M}{2a}\cdot(x-y)+1\right) - 2\cdot f_{ReLU}\left(\frac{M}{2a}\cdot(x-y)\right) \\ &+ f_{ReLU}\left(\frac{M}{2a}\cdot(x-y)-1\right) \end{aligned}$$

which for fixed $y \in \mathbb{R}$ approximates the function $f(x) = (1-(M/(2a))\cdot|x-y|)_+$ with $x \in \mathbb{R}$ (cf. Lemma 4 below).

With these functions of real variables defined above, we can now define $f_{net,j_1,\ldots,j_d,\mathbf{i}_k}$ recursively as follows: We choose $N \geq q$, set $s = \lceil \log_2(N+d) \rceil$ and define for $j_1, \ldots, j_d \in \{0,1,\ldots,N\}$ and $k \in \{1,\ldots,(M+1)^d\}$

$$f_{net,j_1,\ldots,j_d,\mathbf{i}_k}(x) = f_1^{(0)}(x),$$

where

$$f_k^{(l)}(x) = f_{mult}\big(f_{2k-1}^{(l+1)}(x), f_{2k}^{(l+1)}(x)\big)$$

for $k \in \{1,2,\ldots,2^l\}$ and $l \in \{0,\ldots,s-1\}$, and

$$f_k^{(s)}(x) = f_{id}\big(f_{id}\big(x^{(l)} - x_{\mathbf{i}_k}^{(l)}\big)\big)$$

for $j_1 + j_2 + \cdots + j_{l-1} + 1 \leq k \leq j_1 + j_2 + \cdots + j_l$ and $l = 1,\ldots,d$,

$$f_{j_1+j_2+\cdots+j_d+k}^{(s)}(x) = f_{hat,x_{\mathbf{i}_k}^{(k)}}\big(x^{(k)}\big)$$

for $k = 1,\ldots,d$, and

$$f_k^{(s)}(x) = 1$$

for $k = j_1 + j_2 + \cdots + j_d + d + 1, j_1 + j_2 + \cdots + j_d + d + 2, \ldots, 2^s$. It is easy to see that $f_{net,j_1,\ldots,j_d,\mathbf{i}_k}$ is a neural network with $s+2$ hidden layers and at most

$$6 \cdot 2^s,\ 12 \cdot 2^s,\ 2 \cdot 2^s, 2^s,\ \ldots, 8, 4$$

neurons in the layers $1, 2, \ldots, s+2$, resp. Consequently, this network is contained in the class of all fully connected neural networks with $s+2$ hidden layers and $24 \cdot (N+d)$ neurons in each hidden layer. Furthermore it is easy to see that all weights are bounded in absolute value by $c_2 \cdot \max\{1, M/a, R^2\}$.

2.2. Learning the output weights

We define our neural network regression estimate $\tilde{m}_n(x)$ by

$$\tilde{m}_n(x) = \sum_{k=1}^{(M+1)^d} \sum_{\substack{j_1,\ldots,j_d \in \{0,\ldots,N\} \\ j_1+\cdots+j_d \le N}} a_{\mathbf{i}_k,j_1,\ldots,j_d} \cdot f_{net,j_1,\ldots,j_d,\mathbf{i}_k}(x),$$

where the coefficients $a_{\mathbf{i}_k,j_1,\ldots,j_d}$ are chosen by minimizing

$$\frac{1}{n}\sum_{i=1}^{n} |Y_i - \tilde{m}_n(X_i)|^2 + \frac{c_3'}{n} \cdot \sum_{k=1}^{(M+1)^d} \sum_{\substack{j_1,\ldots,j_d \in \{0,\ldots,N\} \\ j_1+\cdots+j_d \le N}} a^2_{\mathbf{i}_k,j_1,\ldots,j_d} \tag{12}$$

for some constant $c_3' > 0$. This regularized linear least squares estimate can be computed by solving a linear equation system. To see this, set

$$J = (M+1)^d \cdot \binom{N+d}{d},$$

let

$$\begin{aligned}&\{B_j : j = 1, \ldots, J\} \\ &\quad = \left\{ f_{net,j_1,\ldots,j_d,\mathbf{i}_k}(x) : 1 \le k \le (1+M)^d \text{ and } 0 \le j_1 + \cdots + j_d \le N \right\}\end{aligned}$$

and set

$$\mathbf{B} = \left(B_j(X_i)\right)_{1 \le i \le n, 1 \le j \le J} \quad \text{and} \quad \mathbf{Y} = (Y_i)_{i=1,\ldots,n}.$$

It is easy to see (cf., Appendix A for a corresponding proof) that the vector of coefficients of our estimate is the unique solution of the linear equation system

$$\left(\frac{1}{n}\mathbf{B}^T\mathbf{B} + \frac{c_3'}{n} \cdot \mathbf{I}\right)\mathbf{a} = \frac{1}{n}\mathbf{B}^T\mathbf{Y}. \tag{13}$$

The value of (12) will be also less than or equal to the value which we get for coefficients equal to zero, hence we have

$$\frac{1}{n}(\mathbf{Y} - \mathbf{B}a)^T(\mathbf{Y} - \mathbf{B}a) + \frac{c_3'}{n} \cdot \mathbf{a}^T\mathbf{a} \le \frac{1}{n}\sum_{i=1}^{n} Y_i^2,$$

which will allow us to derive a bound on the maximal absolute value of our coefficients.

2.3. Rate of convergence

Theorem 1. *Assume that the distribution of (X, Y) satisfies*

$$\mathbf{E}\big(e^{c_4 \cdot |Y|^2}\big) < \infty \tag{14}$$

for some constant $c_4 > 0$ and that the distribution of X has bounded support $supp(X)$, and let $m(x) = \mathbf{E}\{Y|X = x\}$ be the corresponding regression function. Assume that m is (p, C)-smooth, where $p = q + s$ for some $q \in \mathbb{N}_0$ and $s \in (0, 1]$. Define the estimate $\tilde{m}_n$ as in Subsection 2.2, where σ is the logistic squasher and where $N \geq q$, $M = M_n = \lceil c_5 \cdot n^{1/(2p+d)} \rceil$, $R = R_n = n^{d+4}$ and $a = a_n = (\log n)^{1/(6(N+d))}$. Set $\beta_n = c_6 \cdot \log(n)$ for some suitably large constant $c_6 > 0$ and define m_n by

$$m_n(x) = T_{\beta_n} \tilde{m}_n(x)$$

where $T_\beta z = \max\{\min\{z, \beta\}, -\beta\}$ for $z \in \mathbb{R}$ and $\beta > 0$. Then m_n satisfies for n sufficiently large

$$\mathbf{E} \int |m_n(x) - m(x)|^2 \mathbf{P}_X(dx) \leq c_7 \cdot (\log n)^3 \cdot n^{-\frac{2p}{2p+d}},$$

where $c_7 > 0$ does not depend on n.

Remark 1. It follows from the proof of Theorem 1 that the result also holds for more general squashing functions than the logistic squasher. More precisely, in case that the definitions of f_{id}, f_{mult} and f_{ReLU} are modified as in Lemma 1, Lemma 2 and Lemma 3 below, it suffices to assume that σ is Lipschitz continuous and 2-admissible according to Definition 2 below.

Remark 2. We achieved the standard rate of convergence for (p, C)-smooth regression functions (cf. Stone [44]) for neural network estimates which are easy to compute. For the least squares neural network estimates which cannot be computed in practice better rates exist under additional structural assumptions on the regression function (cf. Schmidt-Hieber [43]).

3. A second estimate: random neural features and projection pursuit

In this section we assume that the regression function satisfies

$$m(x) = \sum_{l=1}^{r} g_l(\mathbf{a}_l \cdot \mathbf{x}) \quad \big(x^{(1)}, \ldots, x^{(d)} \in \mathbb{R}\big)$$

for some $r \in \mathbb{N}$, some (p, C)-smooth functions $g_l : \mathbb{R} \to \mathbb{R}$ $(l = 1, \ldots, r)$ and some $\mathbf{a}_l = (a_{(l-1)\cdot d+1}, \ldots, a_{l \cdot d})^T \in \mathbb{R}^d$ with $\|\mathbf{a}_l\| = 1$ $(l = 1, \ldots, r)$ and with indices of $\mathbf{a}_l$ assuming values $(l-1) \cdot d + 1, (l-1) \cdot d + 2, \ldots, (l-1) \cdot d + d = l \cdot d$. Our goal is to construct a neural network regression estimate of m which achieves the univariate rate of convergence. This novel architecture and its novel convergence analysis is presented below (see Theorem 2).

3.1. Definition of the network architecture

Let $A > 0$ be fixed. The choice of the network architecture and of the values of most of the weights of our neural network is motivated by the following approximation result for $x \in [-A, A]^d$: For $M \in \mathbb{N}$ and $i \in \{0, \dots, M\}$ set

$$u_i = -d \cdot A + i \cdot \frac{2 \cdot d \cdot A}{M}$$

and let

$$\{i_1, \dots, i_{M+1}\} = \{0, \dots, M\}.$$

We will see in Section 5 below that we can approximate a (p, C)-smooth projection pursuit model

$$m(x) = \sum_{l=1}^{r} g_l(\mathbf{a}_l^T x)$$

by choosing $\mathbf{b}_l$ close to $\mathbf{a}_l$ and by choosing an appropriate sum of local convex combinations of polynomials of the form

$$\sum_{l=1}^{r} \sum_{k=1}^{M+1} \sum_{\substack{j_1,\dots,j_d \in \{0,\dots,q\}, \\ j_1+\cdots+j_d \le q}} a_{i_k,j_1,\dots,j_d,\mathbf{b}_l} \cdot \left(x^{(1)}\right)^{j_1} \cdots \left(x^{(d)}\right)^{j_d} \cdot \left(1 - \frac{M}{2 \cdot d \cdot A} \cdot |\mathbf{b}_l^T x - u_{i_k}|\right)_+.$$

Our main insight in the sequel is to define appropriate neural networks $f_{net,j_1,\dots,j_d,i_k,\mathbf{b}_l}$ which approximate the functions

$$x \mapsto \left(x^{(1)}\right)^{j_1} \cdots \left(x^{(d)}\right)^{j_d} \cdot \left(1 - \frac{M}{2 \cdot d \cdot A} \cdot |\mathbf{b}_l^T x - u_{i_k}|\right)_+$$

and to choose the network architecture such that neural networks of the form

$$\sum_{l=1}^{r} \sum_{k=1}^{M+1} \sum_{\substack{j_1,\dots,j_d \in \{0,\dots,q\} \\ j_1+\cdots+j_d \le q}} a_{i_k,j_1,\dots,j_d,\mathbf{b}_l} \cdot f_{net,j_1,\dots,j_d,i_k,\mathbf{b}_l}(x) \quad (a_{i_k,j_1,\dots,j_d,\mathbf{b}_l} \in \mathbb{R})$$

are contained in it. To do this, we let $\sigma(x) = 1/(1 + \exp(-x))$ $(x \in \mathbb{R})$ be the logistic squasher, choose $R \ge 1$ and define the following neural networks: The neural network $f_{id}(x)$ as in (9) which approximates the function $f(x) = x$ (cf., Lemma 1 below), the neural network $f_{mult}(x, y)$ as in (10) which approximates the function $f(x, y) = x \cdot y$ (cf., Lemma 2 below), the neural network $f_{ReLu}(x)$ as in (11) which approximates $f(x) = x_+$ (cf., Lemma 3 below), and the neural network

$$\begin{aligned} \bar{f}_{hat,y}(x) = {} & f_{ReLU}\left(\frac{M}{2 \cdot d \cdot A} \cdot (x - y) + 1\right) - 2 \cdot f_{ReLU}\left(\frac{M}{2 \cdot d \cdot A} \cdot (x - y)\right) \\ & + f_{ReLU}\left(\frac{M}{2 \cdot d \cdot A} \cdot (x - y) - 1\right) \end{aligned}$$

which approximates for fixed $y \in \mathbb{R}$ the function $f(x) = (1 - \frac{M}{2 \cdot d \cdot A} \cdot |x - y|)_+$ (cf., Lemma 4 below).

With these functions of real variables defined above we can now define the neural networks $f_{net,j_1,\ldots,j_d,i_k,\mathbf{b}_l}$ recursively as follows: We choose $N \geq q$, set $s = \lceil \log_2(N+1) \rceil$ and define for $l \in \{1,\ldots,r\}$, $j_1,\ldots,j_d \in \{0,1,\ldots,N\}$ and $k \in \{1,\ldots,M+1\}$

$$f_{net,j_1,\ldots,j_d,i_k,\mathbf{b}_l}(x) = f_1^{(0)}(x),$$

where

$$f_k *^{(t)} (x) = f_{mult}\big(f_{2k*-1}^{(t+1)}(x), f_{2k*}^{(t+1)}(x)\big)$$

for $k* \in \{1,2,\ldots,2^t\}$ and $t \in \{0,\ldots,s-1\}$, and

$$f_k *^{(s)} (x) = f_{id}\big(f_{id}\big(x^{(t)}\big)\big)$$

for $j_1 + j_2 + \cdots + j_{t-1} + 1 \leq k* \leq j_1 + j_2 + \cdots + j_t$ and $t = 1,\ldots,d$,

$$f_{j_1+j_2+\cdots+j_d+1}^{(s)}(x) = \bar{f}_{hat,u_{i_k*}}\big(\mathbf{b}_l^T x\big),$$

and

$$f_k^{(s)} * (x) = 1$$

for $k* = j_1 + j_2 + \cdots + j_d + 2, j_1 + j_2 + \cdots + j_d + 3, \ldots, 2^s$. As before, it is easy to see that $f_{net,k,j_1,\ldots,j_d,\mathbf{b}_l}$ is a neural network with $s+2$ hidden layers and at most

$$6 \cdot 2^s,\ 12 \cdot 2^s,\ 2 \cdot 2^s, 2^s,\ \ldots, 8, 4$$

neurons in the layers $1,2,\ldots,s+2$, resp. Consequently, this network is contained in the class of all fully connected neural networks with $s+2$ hidden layers and $24 \cdot (N+1)$ neurons in each hidden layer. Furthermore it is easy to see that all weights are bounded in absolute value by $c_8 \cdot \max\{1, M/A, R^2\}$.

3.2. Learning the output weights

For given directions $\mathbf{b}_l$ $(l = 1,\ldots,r)$ we define our neural network estimate $\tilde{m}_n(x)$ by

$$\tilde{m}_n(x) = \sum_{l=1,\ldots,r} \sum_{k=1}^{M+1} \sum_{\substack{j_1,\ldots,j_d \in \{0,\ldots,N\} \\ j_1+\cdots+j_d \leq N}} a_{i_k,j_1,\ldots,j_d,\mathbf{b}_l} \cdot f_{net,j_1,\ldots,j_d,i_k,\mathbf{b}_l}(x),$$

where the coefficients $a_{i_k,j_1,\ldots,j_d,\mathbf{b}_l}$ are chosen by minimizing

$$\frac{1}{n} \sum_{i=1}^{n} |Y_i - \tilde{m}_n(X_i)|^2 + \frac{c_3''}{n} \cdot \sum_{l=1}^{r} \sum_{k=1}^{M+1} \sum_{\substack{j_1,\ldots,j_d \in \{0,\ldots,N\} \\ j_1+\cdots+j_d \leq N}} a_{i_k,j_1,\ldots,j_d,\mathbf{b}_l}^2 \tag{15}$$

for some constant $c_3'' > 0$. This regularized linear least squares estimate can be computed by solving a linear equation system. To see this, set

$$J = r \cdot (M+1) \cdot \binom{N+d}{d},$$

let

$$\{B_j : j = 1, \dots, J\}$$
$$= \left\{ f_{net, j_1, \dots, j_d, i_k, \mathbf{b}_l}(x) : 1 \le l \le r,\ 1 \le k \le M+1 \text{ and } 0 \le j_1 + \cdots + j_d \le N \right\}$$

and set

$$\mathbf{B} = \big(B_j(X_i)\big)_{1 \le i \le n, 1 \le j \le J} \quad \text{and} \quad \mathbf{Y} = (Y_i)_{i=1,\dots,n}.$$

As in Subsection 2.3 it is easy to see that the vector of coefficients of our estimate is the unique solution of the linear equation system

$$\left(\frac{1}{n} \mathbf{B}^T \mathbf{B} + \frac{c_3''}{n} \cdot \mathbf{I} \right) \mathbf{a} = \frac{1}{n} \mathbf{B}^T \mathbf{Y}. \tag{16}$$

The value of (15) will be also less than or equal to the value which we get for coefficients equal to zero, hence we have

$$\frac{1}{n} (\mathbf{Y} - \mathbf{B}a)^T (\mathbf{Y} - \mathbf{B}a) + \frac{c_3''}{n} \cdot \mathbf{a}^T \mathbf{a} \le \frac{1}{n} \sum_{i=1}^{n} Y_i^2, \tag{17}$$

which will allow us to derive a bound on the maximal absolute value of our coefficients.

3.2.1. Choice of the directions

In order to choose $\mathbf{b}_l$ $(l = 1, \dots, r)$, we choose them I_n times independent randomly according to a uniform distribution on $[-1,1]^d$, compute each time the corresponding outer weights as in Subsection 3.2, and choose the directions and the corresponding outer weights for our estimate $\tilde{m}_n$, where the empirical L_2 risk of the estimate is minimal.

3.3. Rate of convergence

Theorem 2. *Assume that the distribution of (X,Y) satisfies (14) for some constant $c_4 > 0$ and that the distribution of X has bounded support $supp(X)$, and let $m(x) = \mathbf{E}\{Y|X = x\}$ be the corresponding regression function. Let $r \in \mathbb{N}$, $p > 0$ and $C > 0$, and assume that the regression function satisfies*

$$m(x) = \sum_{l=1}^{r} g_l\big(\mathbf{a}_l^T x\big) \quad \big(x \in \mathbb{R}^d\big)$$

for some (p, C)*-smooth functions* $g_l : \mathbb{R} \to \mathbb{R}$ *and some* $\mathbf{a}_l \in \mathbb{R}^d$ *with* $\|\mathbf{a}_l\| = 1$ $(l = 1, \ldots, r)$.

Define the estimate $\tilde{m}_n$ *as in Subsections 3.1–3.2.1, where* σ *is the logistic squasher and where* $I_n = \lceil c_9 \cdot (\log n)^2 \cdot n^{\frac{r \cdot d}{2p+1}} \rceil$, $N \geq p$, $M = M_n = \lceil c_{10} \cdot n^{1/(2p+1)} \rceil$, $R = R_n = n^3$ *and* $A = A_n = (\log n)^{1/(6(N+d))}$. *Set* $\beta_n = c_6 \cdot \log(n)$ *for some suitably large constant* $c_6 > 0$ *and define* m_n *by*

$$m_n(x) = T_{\beta_n} \tilde{m}_n(x).$$

Then m_n *satisfies for* n *sufficiently large*

$$\mathbf{E} \int |m_n(x) - m(x)|^2 \mathbf{P}_X(dx) \leq c_{11} \cdot (\log n)^3 \cdot n^{-\frac{2p}{2p+1}},$$

where $c_{11} > 0$ *does not depend on* n.

Remark 3. In order to compute our estimate, we have to solve I_n times a linear equation system with a square matrix of size M_n, for which computing time is proportional to

$$I_n \cdot M_n^2 \approx (\log n)^2 \cdot n^{\frac{r \cdot d + 2}{2p+1}}.$$

Hence in case

$$r \cdot d < 4 \cdot p$$

computing time is $O(n^2)$, so in case that the number r of terms in the projection pursuit model and the dimension d of X are not too large in comparison with the smoothness p of the regression function, our estimate can be computed in $O(n^2)$ time. Because d and r might be in fact arbitrarily large in our paper in case that p is large, we do not have an upper bound on p in our theory.

4. Application to simulated data

In this section we illustrate the finite sample size performance of our newly proposed estimate by applying it to simulated data.

The simulated data which we use is defined as follows: We choose X uniformly distributed on $[-1, 1]^d$, where d is the dimension of the input, ϵ standard normal and independent of X, and we define Y by

$$Y = m_j(X) + \sigma \cdot \lambda_j \cdot \epsilon, \tag{18}$$

where $m_j : [-1, 1]^d \to \mathbb{R}$ is described below, $\lambda_j > 0$ is a scaling value defined below and σ is chosen from $\{0.05, 0.10\}$ $(j \in \{1, 2, 3, 4\})$. As regression functions we use

$$m_1(x_1, x_2) = \log\bigl((0.2 \cdot x_1 + 0.9 \cdot x_2)^2\bigr) + \cos\left(\frac{\pi}{\log(0.5 \cdot x_1 + 0.3 \cdot x_2)^2}\right)$$

$$
\begin{aligned}
&+\exp\left(\frac{1}{50}\cdot(0.7\cdot x_1+0.7\cdot x_2)\right)+\frac{\tan(\pi\cdot(0.1\cdot x_1+0.3\cdot x_2)^4)}{(0.1\cdot x_1+0.3\cdot x_2)^2},\\
&m_2(x_1,x_2,x_3,x_4)\\
&\quad=\tan\bigl(\sin\bigl(\pi\cdot(0.2\cdot x_1+0.5\cdot x_2-0.6\cdot x_3+0.2\cdot x_4)\bigr)\bigr)\\
&\qquad+\bigl(0.5\cdot(x_1+x_2+x_3+x_4)\bigr)^3\\
&\qquad+\frac{1}{(0.5\cdot x_1+0.3\cdot x_2-0.3\cdot x_3+0.25\cdot x_4)^2+4},\\
&m_3(x_1,x_2,x_3,x_4,x_5)\\
&\quad=\log\bigl(0.5\cdot(x_1+0.3\cdot x_2+0.6\cdot x_3+x_4-x_5)^2\bigr)\\
&\qquad+\sin\bigl(\pi\cdot(0.7\cdot x_1+x_2-0.3\cdot x_3-0.4\cdot x_4-0.8\cdot x_5)\bigr)\\
&\qquad+\cos\left(\frac{\pi}{1+\sin(0.5\cdot(x_2+0.9\cdot x_3-x_5))}\right)
\end{aligned}
$$

and

$$
\begin{aligned}
&m_4(x_1,x_2,x_3,x_4,x_5,x_6)\\
&\quad=\exp\bigl(0.2\cdot(x_1+x_2+x_3+x_4+x_5+x_6)\bigr)\\
&\qquad+\sin\left(\frac{\pi}{2}\cdot(x_1-x_2-x_3+x_4-x_5-x_6)\right)\\
&\qquad+\frac{1}{(0.3\cdot x_1-0.2\cdot x_2+0.8\cdot x_3-0.5\cdot x_4+0.6\cdot x_5-0.2\cdot x_6)^2+6}\\
&\qquad+0.5\cdot(x_1+x_3-x_5)^3
\end{aligned}
$$

λ_j is chosen approximately as IQR of a sample of size 100 of $m(X)$, and we use the values $\lambda_1 = 2.20$, $\lambda_2 = 1.96$, $\lambda_3 = 2.85$, and $\lambda_4 = 1.59$. From distribution defined by (18) we generate a sample of size $n = 100, 200, 400$ and apply our newly proposed neural network regression estimate and compare our results to that of six alternative regression estimates on the same data. Then we compute the L_2 errors of these estimates approximately by using the empirical L_2 error $\varepsilon_{L_2,\bar{N}}(\cdot)$ on an independent sample of X of size $\bar{N} = 10{,}000$. Since this error strongly depends on the behavior of the correct function m_j, we consider it in relation to the error of the simplest estimate for m_j we can think of, a completely constant function (whose value is the average of the observed data according to the least squares approach). Thus, the scaled error measure we use for evaluation of the estimates is $\varepsilon_{L_2,\bar{N}}(m_{n,i})/\bar{\varepsilon}_{L_2,\bar{N}}(avg)$, where $\bar{\varepsilon}_{L_2,\bar{N}}(avg)$ is the median of 50 independent realizations of the value one obtains if one plugs the average of n observations into $\varepsilon_{L_2,\bar{N}}(\cdot)$. To a certain extent, this quotient can be interpreted as the relative part of the error of the constant estimate that is still contained in the more sophisticated approaches. The resulting scaled errors of course depend on the random sample of (X, Y), and to be able to compare these values nevertheless we repeat the whole computation 50 times and report the median and the interquartile range of the 50 scaled errors for each of our estimates.

We choose the parameters for each of the estimates by splitting of the sample. Here we split our sample in a learning sample of size $n_l = 0.8 \cdot n$ and a testing sample of size $n_t = 0.2 \cdot n$. We compute the estimate for all parameter values from the sets described below using the learning sample, compute the corresponding empirical L_2 risk on the testing sample and choose the parameter value which leads to the minimal empirical L_2 risk on the testing sample.

Our first three estimates are fully connected neural network estimates where the number of layers is fixed (with the same number of neurons in each layer) and the number of neurons per layer is chosen adaptively. We implemented FcNeural ourselves with Tensorflow API. We used SGD optimizer with $learningrate = 0.02$, sigmoidal activation function and number of epochs parameter $epoch = 10^5$. The estimate *fc-neural-1* has one hidden layer, estimate *fc-neural-3* has three hidden layers and estimate *fc-neural-6* has six hidden layers. The number of neurons per layer is chosen from the set $\{5, 10, 25, 50, 75\}$, $\{3, 6, 9, 12, 15\}$, $\{2, 4, 6, 8, 10\}$, respectively.

Our fourth estimate *kernel* is the Nadaraya-Watson kernel regression estimate with the so-called naive kernel where the bandwidth is chosen from the set $\{2^k : k = -5, -4, \ldots, 5\}$. We used kernel regression estimate from *Statsmodel Library*.

Our fifth estimate *KNN* is the nearest neighbor estimate with the number of nearest neighbors set to $1, 2, 3, 4, 8, 12, 16, 20$. We used *KNN* implementation from *Sklearn Scikit Library* with the following parameters *weights=uniform, algorithm=auto, leaf_size=30, p=2, metric=minkowski*. These parameters are the default values of the library.

Our sixth estimate *RBF* is an interpoland with the radial basis function $\Phi(r)$ selected from the following list: *linear* $\Phi(r) = -r$, *cubic* $\Phi(r) = r^3$, *quintic* $\Phi(r) = -r^5$ and a *thin_plate_spline* $\Phi(r) = r^2 \cdot \log(r)$, where $r = \|x - c\|$, c is the center of the RBF and $\|\cdot\|$ is the Euclidean norm. We used *RBF* implementation from *Scipy Library*. The RBF functions used in our experiments are pre-implemented functions in the library.

Our seventh estimate *MARS* is a method which makes use of multivariate adaptive regression splines. We used *MARS* implementation from *Py-earth Library*.

Our last estimate *proj-neural* is our newly proposed neural network estimate presented in this paper. The parameters of the estimate were chosen as follows: N is set to 2, A is set to 1, and R is set to 10^6, and r is set to 4. Parameter M of the estimate is chosen from the set $\{2, 4, 8, 16\}$. I_n was set to 400. c_3'' was set to 1.

The results are summarized in Table 1, Table 2 and Table 3. In our experiments, the kernel estimator sometimes returns NaN. The model of our method uses "$J + r \cdot d$" parameters. Table 4 shows a minimum and a maximum number of parameters of the ProjNeural model used in the experiment.

As we can see from the reported scaled errors, our newly proposed neural network estimate performs pretty well. Although it is systematically outperformed by a fully connected network the scaled errors of our estimate lie within a small range of the best error value. Despite the fact that our estimate is slightly out-

TABLE 1
Median and IQR of the scaled empirical L_2 error of estimates for sample size $n = 100$.

100 Samples	M1				M2			
Noise scale	2.2				1.96			
	Noise 5%		Noise 10%		Noise 5%		Noise 10%	
Loss scale	4.0842		4.1229		1.5530		1.5849	
After loss scale	Median	IQR	Median	IQR	Median	IQR	Median	IQR
FcNeural-1	0.0880	0.0423	0.0977	0.0385	0.0305	0.0141	0.0525	0.0225
FcNeural-3	0.0992	0.0681	0.1124	0.0850	0.0356	0.0140	0.0612	0.0322
FcNeural-6	0.9696	0.6124	0.9610	0.8106	0.0787	0.1015	0.0977	0.0575
Kernel	0.2796	0.0844	0.2748	0.0989	0.2263	0.0388	NaN	NaN
KNN	0.3095	0.1073	0.3246	0.0879	0.2852	0.0812	0.2986	0.0947
MARS	0.3727	0.0608	0.3717	0.0436	0.4928	0.1177	0.4610	0.0852
ProjNeural (ours)	0.1474	0.0825	0.1687	0.0631	0.0791	0.0347	0.0943	0.0400
RBF	0.2146	0.0677	0.2230	0.0631	0.0739	0.0294	0.1015	0.0390
100 Samples	M3				M4			
Noise scale	2.85				1.59			
	Noise 5%		Noise 10%		Noise 5%		Noise 10%	
Loss scale	5.9496		6.0032		3.3927		3.3975	
After loss scale	Median	IQR	Median	IQR	Median	IQR	Median	IQR
FcNeural-1	0.4680	0.2636	0.4907	0.2572	0.2090	0.1121	0.2482	0.1885
FcNeural-3	0.4646	0.4514	0.6035	0.3580	0.1208	0.0939	0.1507	0.0947
FcNeural-6	0.6741	0.4977	0.6575	0.5332	0.2007	0.1536	0.2032	0.1788
Kernel	0.7619	0.1454	NaN	NaN	NaN	NaN	0.4858	0.1249
KNN	0.7986	0.1586	0.7950	0.1268	0.5187	0.1146	0.5268	0.0888
MARS	1.0268	0.1082	1.0083	0.0863	0.4765	0.1129	0.4688	0.1058
ProjNeural (ours)	0.5687	0.1054	0.5716	0.1365	0.2664	0.1141	0.2773	0.1079
RBF	0.6500	0.0749	0.6390	0.0970	0.2988	0.0967	0.2897	0.0680

TABLE 2
Median and IQR of the scaled empirical L_2 error of estimates for sample size $n = 200$.

200 Samples	M1				M2			
Noise scale	2.2				1.96			
	Noise 5%		Noise 10%		Noise 5%		Noise 10%	
Loss scale	4.0809		4.1308		1.5494		1.5654	
After loss scale	Median	IQR	Median	IQR	Median	IQR	Median	IQR
FcNeural-1	0.0698	0.0182	0.0769	0.0191	0.0179	0.0105	0.0241	0.0092
FcNeural-3	0.0549	0.0246	0.0629	0.0331	0.0181	0.0065	0.0250	0.0079
FcNeural-6	0.9959	0.0465	0.9824	0.0465	0.0468	0.0380	0.0500	0.0684
Kernel	0.1901	0.0365	0.1949	0.0389	0.1460	0.0397	0.1467	0.0343
KNN	0.2141	0.0308	0.2273	0.0491	0.1859	0.0328	0.1926	0.0262
MARS	0.3348	0.0287	0.3402	0.0329	0.4506	0.0405	0.4715	0.0606
ProjNeural (ours)	0.1036	0.0239	0.1126	0.0240	0.0345	0.0160	0.0426	0.0124
RBF	0.1425	0.0303	0.1578	0.0313	0.0328	0.0103	0.0522	0.0138
200 Samples	M3				M4			
Noise scale	2.85				1.59			
	Noise 5%		Noise 10%		Noise 5%		Noise 10%	
Loss scale	5.9577		5.9793		3.3676		3.3969	
After loss scale	Median	IQR	Median	IQR	Median	IQR	Median	IQR
FcNeural-1	0.1717	0.0712	0.1932	0.0426	0.0547	0.0374	0.0562	0.0465
FcNeural-3	0.2353	0.1280	0.2802	0.0980	0.0394	0.0208	0.0421	0.0201
FcNeural-6	0.3156	0.2953	0.3267	0.6323	0.0560	0.0376	0.0631	0.0388
Kernel	NaN	NaN	0.6447	0.0736	0.3834	0.0751	0.3762	0.0693
KNN	0.6621	0.0712	0.6580	0.0858	0.4089	0.0851	0.4042	0.0756
MARS	1.0276	0.0823	1.0056	0.0565	0.4426	0.0464	0.4435	0.0683
ProjNeural (ours)	0.4630	0.0976	0.4605	0.0783	0.1156	0.0467	0.1237	0.0397
RBF	0.5169	0.0555	0.5194	0.0490	0.1471	0.0329	0.1483	0.0445

TABLE 3
Median and IQR of the scaled empirical L_2 error of estimates for sample size $n = 400$.

400 Samples	M1				M2			
Noise scale	2.2				1.96			
	Noise 5%		Noise 10%		Noise 5%		Noise 10%	
Loss scale	4.0246		4.0578		1.5454		1.5745	
After loss scale	Median	IQR	Median	IQR	Median	IQR	Median	IQR
FcNeural-1	0.0625	0.0121	0.0648	0.0123	0.0121	0.0085	0.0160	0.0080
FcNeural-3	0.0402	0.0153	0.0436	0.0190	0.0131	0.0044	0.0153	0.0038
FcNeural-6	0.9963	0.0321	0.9891	0.0493	0.0442	0.0266	0.0410	0.0203
Kernel	0.1273	0.0371	0.1313	0.0304	0.1048	0.0213	0.1032	0.0180
KNN	0.1434	0.0266	0.1486	0.0231	0.1238	0.0162	0.1314	0.0156
MARS	0.3170	0.0322	0.3141	0.0298	0.4494	0.0335	0.4408	0.0304
ProjNeural (ours)	0.0834	0.0123	0.0849	0.0146	0.0185	0.0088	0.0238	0.0061
RBF	0.1023	0.0316	0.1030	0.0244	0.0179	0.0028	0.0294	0.0057
400 Samples	M3				M4			
Noise scale	2.85				1.59			
	Noise 5%		Noise 10%		Noise 5%		Noise 10%	
Loss scale	5.8926		5.9618		3.3786		3.3756	
After loss scale	Median	IQR	Median	IQR	Median	IQR	Median	IQR
FcNeural-1	0.0986	0.0182	0.1010	0.0250	0.0257	0.0187	0.0305	0.0242
FcNeural-3	0.1092	0.0396	0.1198	0.0572	0.0161	0.0062	0.0201	0.0062
FcNeural-6	0.7727	0.7861	0.2874	0.7646	0.0272	0.0089	0.0286	0.0130
Kernel	0.5078	0.0495	0.4992	0.0406	0.2823	0.0347	0.2781	0.0458
KNN	0.5556	0.0460	0.5540	0.0441	0.3179	0.0430	0.3120	0.0488
MARS	1.0091	0.0469	0.9973	0.0444	0.4177	0.0333	0.4122	0.0298
ProjNeural (ours)	0.3585	0.0469	0.3576	0.0705	0.0554	0.0181	0.0502	0.0215
RBF	0.4371	0.0363	0.4285	0.0263	0.0659	0.0151	0.0725	0.0128

TABLE 4
Number of parameters in ProjNeural. One can see that the number of parameters is affected by hyperparameter M and input dimension d.

ProjNeural N Params	Min ($M = 2$)	Max ($M = 16$)
M1 ($d = 2$)	80	416
M2 ($d = 4$)	196	1036
M3 ($d = 5$)	272	1448
M4 ($d = 6$)	360	1928

performed by some other estimates its main advantage remains that it can be theoretically analyzed in the way it is implemented (which is not true for the implementations of the other neural network estimates).

5. Proofs

5.1. Approximation results for neural networks

We will use the following assumption on the activation function of our neural network.

Definition 2. Let $N \in \mathbb{N}_0$. A function $\sigma : \mathbb{R} \to [0, 1]$ is called **N-admissible**, if it is nondecreasing and Lipschitz continuous and if, in addition, the following three conditions are satisfied:

(i) The function σ is $N+1$ times continuously differentiable with bounded derivatives.
(ii) A point $t_\sigma \in \mathbb{R}$ exists, where all derivatives up to order N of σ are nonzero.
(iii) If $y > 0$, the relation $|\sigma(y)-1| \leq \frac{1}{y}$ holds. If $y < 0$, the relation $|\sigma(y)| \leq \frac{1}{|y|}$ holds.

It is easy to see that the logistic squasher (3) is N-admissible for any $N \in \mathbb{N}$ (cf., e.g., Bauer and Kohler [9]).

Lemma 1. *Let $\sigma : \mathbb{R} \to \mathbb{R}$ be a function, let $R, a > 0$.*

a) *Assume that σ is two times continuously differentiable and let $t_{\sigma,id} \in \mathbb{R}$ be such that $\sigma'(t_{\sigma,id}) \neq 0$. Then*

$$f_{id}(x) = \frac{R}{\sigma'(t_{\sigma,id})} \cdot \left(\sigma\left(\frac{x}{R} + t_{\sigma,id} \right) - \sigma(t_{\sigma,id}) \right)$$

satisfies for any $x \in [-a, a]$:

$$|f_{id}(x) - x| \leq \frac{\|\sigma''\|_\infty \cdot a^2}{2 \cdot |\sigma'(t_{\sigma,id})|} \cdot \frac{1}{R}.$$

b) *Assume that σ is three times continuously differentiable and let $t_{\sigma,sq} \in \mathbb{R}$ be such that $\sigma''(t_{\sigma,sq}) \neq 0$. Then*

$$f_{sq}(x) = \frac{R^2}{\sigma''(t_{\sigma,sq})} \cdot \left(\sigma\left(\frac{2x}{R} + t_{\sigma,sq} \right) - 2 \cdot \sigma\left(\frac{x}{R} + t_{\sigma,sq} \right) + \sigma(t_{\sigma,sq}) \right)$$

satisfies for any $x \in [-a, a]$:

$$|f_{sq}(x) - x^2| \leq \frac{5 \cdot \|\sigma'''\|_\infty \cdot a^3}{3 \cdot |\sigma''(t_{\sigma,sq})|} \cdot \frac{1}{R}.$$

Proof. The result follows in a straightforward way from the proof of Theorem 2 in Scarselli and Tsoi [41], cf. Lemma 1 in Kohler, Krzyżak and Langer [34]. □

Remark 4. In case of the logistic squasher it is easy to see that with the choice $t_{\sigma,id} = 0$ the network f_{id} in Lemma 1 is given by (9).

Lemma 2. *Let $\sigma : \mathbb{R} \to [0, 1]$ be 2-admissible according to Definition 2. Then for any $R > 0$ and any $a > 0$ the neural network*

$$f_{mult}(x, y) = \frac{R^2}{4 \cdot \sigma''(t_\sigma)} \cdot \left(\sigma\left(\frac{2 \cdot (x+y)}{R} + t_\sigma \right) - 2 \cdot \sigma\left(\frac{x+y}{R} + t_\sigma \right) - \sigma\left(\frac{2 \cdot (x-y)}{R} + t_\sigma \right) + 2 \cdot \sigma\left(\frac{x-y}{R} + t_\sigma \right) \right)$$

satisfies for any $x \in [-a, a]$:

$$|f_{mult}(x, y) - x \cdot y| \leq \frac{20 \cdot \|\sigma'''\|_\infty \cdot a^3}{3 \cdot |\sigma''(t_\sigma)|} \cdot \frac{1}{R}.$$

Proof. See Lemma 2 in Kohler, Krzyżak and Langer [34]. □

Remark 5. In case of the logistic squasher it is easy to see that with the choice $t_\sigma = 1$ the network f_{mult} in Lemma 2 is given by (10).

Lemma 3. *Let $\sigma : \mathbb{R} \to [0,1]$ be 2-admissible according to Definition 2. Let f_{mult} be the neural network from Lemma 2 and let f_{id} be the network from Lemma 1. Assume*

$$a \geq 1 \quad \text{and} \quad R \geq \frac{\|\sigma''\|_\infty \cdot a}{2 \cdot |\sigma'(t_{\sigma.id})|}. \tag{19}$$

Then the neural network

$$\begin{aligned} f_{ReLU}(x) &= f_{mult}\big(f_{id}(x), \sigma(R \cdot x)\big) \\ &= \sum_{k=1}^{4} d_k \cdot \sigma\left(\sum_{i=1}^{2} b_{k,i} \cdot \sigma(a_i \cdot x + t_\sigma) + b_{k,3} \cdot \sigma(a_3 \cdot x) + t_\sigma\right) \end{aligned}$$

satisfies

$$|f_{ReLU}(x) - \max\{x, 0\}| \leq 56 \cdot \frac{\max\{\|\sigma''\|_\infty, \|\sigma'''\|_\infty, 1\}}{\min\{2 \cdot |\sigma'(t_{\sigma.id})|, |\sigma''(t_\sigma)|, 1\}} \cdot a^3 \cdot \frac{1}{R}$$

for all $x \in [-a, a]$.

Proof. See Lemma 3 in Kohler, Krzyżak and Langer [34]. □

Lemma 4. *Let $M \in \mathbb{N}$ and let $\sigma : \mathbb{R} \to [0,1]$ be 2-admissible according to Definition 2. Let $a > 0$ and*

$$R \geq \frac{\|\sigma''\|_\infty \cdot (M+1)}{2 \cdot |\sigma'(t_{\sigma.id})|},$$

let $y \in [-a, a]$ and let f_{ReLU} be the neural network of Lemma 3. Then the network

$$\begin{aligned} f_{hat,y}(x) = {} & f_{ReLU}\left(\frac{M}{2a} \cdot (x - y) + 1\right) - 2 \cdot f_{ReLU}\left(\frac{M}{2a} \cdot (x - y)\right) \\ & + f_{ReLU}\left(\frac{M}{2a} \cdot (x - y) - 1\right) \end{aligned}$$

satisfies

$$\left| f_{hat,y}(x) - \left(1 - \frac{M}{2a} \cdot |x - y|\right)_+ \right| \leq 1792 \cdot \frac{\max\{\|\sigma''\|_\infty, \|\sigma'''\|_\infty, 1\}}{\min\{2 \cdot |\sigma'(t_{\sigma.id})|, |\sigma''(t_\sigma)|, 1\}} \cdot M^3 \cdot \frac{1}{R}$$

for all $x \in [-a, a]$.

Proof. Since

$$\left(1 - \frac{M}{2a} \cdot |x|\right)_+$$

$$= \max\left\{\frac{M}{2a}\cdot x+1,0\right\} - 2\cdot\max\left\{\frac{M}{2a}\cdot x,0\right\} + \max\left\{\frac{M}{2a}\cdot x-1,0\right\} \quad (x\in\mathbb{R})$$

the result is an easy consequence of Lemma 3 (applied with $M+1$ instead of a). □

Lemma 5. *Let $M\in\mathbb{N}$ and let $\sigma:\mathbb{R}\to[0,1]$ be 2-admissible according to Definition 2. Let $a\geq 1$ and*

$$R\geq \max\left\{\frac{\|\sigma''\|_\infty\cdot(M+1)}{2\cdot|\sigma'(t_{\sigma,id})|}, \frac{9\cdot\|\sigma''\|_\infty\cdot a}{|\sigma'(t_{\sigma,id})|}, \frac{20\cdot\|\sigma'''\|_\infty}{3\cdot|\sigma''(t_\sigma)|}\cdot 3^{3\cdot 3^s}\cdot a^{3\cdot 2^s}, 1792\cdot\frac{\max\{\|\sigma''\|_\infty,\|\sigma'''\|_\infty,1\}}{\min\{2\cdot|\sigma'(t_{\sigma,id})|,|\sigma''(t_\sigma)|,1\}}\cdot M^3\right\} \tag{20}$$

and let $y\in[-a,a]^d$. Let $N\in\mathbb{N}$ and let $j_1,\dots,j_d\in\mathbb{N}_0$ such that $j_1+\cdots+j_d\leq N$, and set $s=\lceil\log_2(N+d)\rceil$. Let f_{id}, f_{mult} and $f_{hat,z}$ (for $z\in\mathbb{R}$) be the neural networks defined in Lemma 1, Lemma 2 and Lemma 4, resp. Define the network $f_{net,j_1,\dots,j_d,y}$ by

$$f_{net,j_1,\dots,j_d,y}(x) = f_1^{(0)}(x),$$

where $f_1^{(0)}$ is defined by backward recursion as follows:

$$f_k^{(l)}(x) = f_{mult}\big(f_{2k-1}^{(l+1)}(x), f_{2k}^{(l+1)}(x)\big)$$

for $k\in\{1,2,\dots,2^l\}$ and $l\in\{0,\dots,s-1\}$, and

$$f_k^{(s)}(x) = f_{id}\big(f_{id}\big(x^{(l)}-y^{(l)}\big)\big)$$

for $j_1+j_2+\cdots+j_{l-1}+1\leq k\leq j_1+j_2+\cdots+j_l$ and $l=1,\dots,d$,

$$f_{j_1+j_2+\cdots+j_d+k}^{(s)}(x) = f_{hat,y^{(k)}}\big(x^{(k)}\big)$$

for $k=1,\dots,d$, and

$$f_k^{(s)}(x)=1$$

for $k=j_1+j_2+\cdots+j_d+d+1, j_1+j_2+\cdots+j_d+d+2,\dots,2^s$. Then we have for any $x\in[-a,a]^d$:

$$\left|f_{net,y}(x) - \big(x^{(1)}-y^{(1)}\big)^{j_1}\cdots\big(x^{(d)}-y^{(d)}\big)^{j_d}\prod_{j=1}^d\left(1-\frac{M}{2a}\cdot|x^{(j)}-y^{(j)}|\right)_+\right| \leq c_{12}\cdot 3^{3\cdot 3^s}\cdot a^{3\cdot 2^s}\cdot M^3\cdot\frac{1}{R}.$$

Proof. The result follows from Lemma 1, Lemma 2 and Lemma 4 in a straightforward but technical way using an induction. A complete proof can be found in the Appendix A. □

Remark 6. The result can be analogously stated for our estimate in the context of the projection pursuit model. The corresponding statement and a complete proof can be found in the Appendix A.

5.2. Approximation of a projection pursuit model by piecewise polynomials

Lemma 6. *Let $p = q + s$ for some $q \in \mathbb{N}_0$ and $s \in (0, 1]$. Let $C > 0$, $r \in \mathbb{N}$, $g_l : \mathbb{R} \to \mathbb{R}$ (p, C)-smooth functions $(l = 1, \dots, r)$ and $\mathbf{a}_l \in \mathbb{R}^d$ $(l = 1, \dots, r)$. Set*

$$m(x) = \sum_{l=1}^{r} g_l\big(\mathbf{a}_l^T x\big) \quad \big(x \in \mathbb{R}^d\big).$$

For $\mathbf{b}_l \in \mathbb{R}^d$ $(l = 1, \dots, r)$ set

$$g(x) = \sum_{l=1}^{r} \sum_{j=0}^{q} \frac{g_l^{(j)}(\mathbf{b}_l^T x)}{j!} \cdot \big((\mathbf{a}_l - \mathbf{b}_l)^T x\big)^j,$$

where $g_l^{(j)}$ denotes the j-th derivative of g_l. Then we have for any $x \in \mathbb{R}^d$

$$|m(x) - g(x)| \le \frac{r \cdot d^p \cdot C}{q!} \cdot \|x\|_\infty^p \cdot \Big(\max_l \|\mathbf{a}_l - \mathbf{b}_l\|_\infty\Big)^p.$$

Proof. By the proof of Lemma 11.1 in Györfi et al. [19] we have for any $z \in \mathbb{R}$

$$\left| g_l(u) - \sum_{j=0}^{q} \frac{g_l^{(j)}(z)}{j!} \cdot (u - z)^j \right| \le \frac{1}{q!} \cdot C \cdot |u - z|^p \quad (u \in \mathbb{R}).$$

Applying this with $u = \mathbf{a}_l^T x$ and $z = \mathbf{b}_l^T x$ we get

$$\begin{aligned}
&|m(x) - g(x)| \\
&\quad \le \sum_{l=1}^{r} \left| g_l\big(\mathbf{a}_l^T x\big) - \sum_{j=0}^{q} \frac{g_l^{(j)}(\mathbf{b}_l^T x)}{j!} \cdot \big(\mathbf{a}_l^T x - \mathbf{b}_l^T x\big)^j \right| \\
&\quad \le \sum_{l=1}^{r} \frac{1}{q!} \cdot C \cdot |\mathbf{a}_l^T x - \mathbf{b}_l^T x|^p \\
&\quad \le \frac{r \cdot d^p \cdot C}{q!} \cdot \|x\|_\infty^p \cdot \Big(\max_l \|\mathbf{a}_l - \mathbf{b}_l\|_\infty\Big)^p.
\end{aligned}$$

□

Lemma 7. *Let $p = q + s$ for some $q \in \mathbb{N}_0$ and $s \in (0, 1]$. Let $C > 0$, $r \in \mathbb{N}$, $g_l : \mathbb{R} \to \mathbb{R}$ (p, C)-smooth functions $(l = 1, \dots, r)$ and $\mathbf{a}_l \in \mathbb{R}^d$ with $\|\mathbf{a}_l\| = 1$ and $\mathbf{b}_l \in [-1, 1]^d$ $(l = 1, \dots, r)$. Let $A \ge 1$, $M \in \mathbb{N}$, set*

$$u_i = -d \cdot A + i \cdot \frac{2 \cdot d \cdot A}{M} \quad (i = 0, \dots, M)$$

and $\{i_1, \dots, i_{M+1}\} = \{0, \dots, M\}$. Then there exist polynomials $p_{i_k,l} : \mathbb{R}^d \to \mathbb{R}$ of total degree q, which depend on $\mathbf{a}_l$ and $\mathbf{b}_l$ and where all coefficients are bounded in absolute value by

$$(q+1) \cdot 2^p \cdot d^{3p/2} \cdot A^p \cdot \max_{l \in \{1,\dots,r\}, j \in \{0,\dots,q\}} \|g_l^{(j)}\|_\infty, \cdot \Big(\max\Big\{\max_{l=1,\dots,r} \|\mathbf{a}_l - \mathbf{b}_l\|_\infty, 1\Big\}\Big)^p,$$

such that we have for all $x \in [-A, A]^d$

$$\begin{aligned}
&\left| \sum_{l=1}^{r} \sum_{j=0}^{q} \frac{g_l^{(j)}(\mathbf{b}_l^T x)}{j!} \cdot \left((\mathbf{a}_l - \mathbf{b}_l)^T x\right)^j \right. \\
&\qquad \left. - \sum_{l=1}^{r} \sum_{k=1}^{M+1} p_{i_k,l}(x) \cdot \left(1 - \frac{M}{2 \cdot d \cdot A} \cdot |\mathbf{b}_l^T x - u_{i_k}|\right)_+ \right| \\
&\quad \leq r \cdot 2^p \cdot (p+1) \cdot C \cdot d^{3p/2} \cdot A^{2p} \cdot \left(\max\left\{ \frac{1}{M}, \max_{l=1,\ldots,r} \|\mathbf{a}_l - \mathbf{b}_l\|_\infty \right\} \right)^p.
\end{aligned}$$

Proof. Let p_{l,j,i_k} be the Taylor polynomial of $g_l^{(j)}$ of degree $q-j$ around u_{i_k}. Because of the $(p-j, C)$-smoothness of $g_l^{(j)}$ Lemma 11.1 in Györfi et al. [19] implies

$$\left| g_l^{(j)}\left(\mathbf{b}_l^T x\right) - p_{l,j,i_k}\left(\mathbf{b}_l^T x\right) \right| \leq \frac{1}{(q-j)!} \cdot C \cdot |\mathbf{b}_l^T x - u_{i_k}|^{(p-j)}.$$

From this we can conclude for $x \in [-A, A]^d$

$$\begin{aligned}
&\left| \frac{g_l^{(j)}(\mathbf{b}_l^T x)}{j!} \cdot \left((\mathbf{a}_l - \mathbf{b}_l)^T x\right)^j - \frac{p_{l,j,i_k}(\mathbf{b}_l^T x)}{j!} \cdot \left((\mathbf{a}_l - \mathbf{b}_l)^T x\right)^j \right| \\
&\quad \leq \frac{1}{(q-j)!} \cdot C \cdot d^j \cdot A^j \cdot \left(\max\left\{ |\mathbf{b}_l^T x - u_{i_k}|, \|\mathbf{a}_l - \mathbf{b}_l\|_\infty \right\}\right)^p.
\end{aligned}$$

Using

$$\sum_{k=1}^{M+1} \left(1 - \frac{M}{2 \cdot d \cdot A} \cdot |\mathbf{b}_l^T x - u_{i_k}|\right)_+ = 1$$

for $x \in [-A, A]^d$, this in turn implies for $x \in [-A, A]^d$

$$\begin{aligned}
&\left| \sum_{j=0}^{q} \frac{g_l^{(j)}(\mathbf{b}_l^T x)}{j!} \cdot \left((\mathbf{a}_l - \mathbf{b}_l)^T x\right)^j \right. \\
&\qquad \left. - \sum_{j=0}^{q} \sum_{k=1}^{M+1} \frac{p_{l,j,i_k}(\mathbf{b}_l^T x)}{j!} \cdot \left((\mathbf{a}_l - \mathbf{b}_l)^T x\right)^j \cdot \left(1 - \frac{M}{2 \cdot d \cdot A} \cdot |\mathbf{b}_l^T x - u_{i_k}|\right)_+ \right| \\
&\quad \leq \sum_{j=0}^{q} \sum_{k=1}^{M+1} \left| \frac{g_l^{(j)}(\mathbf{b}_l^T x)}{j!} \cdot \left((\mathbf{a}_l - \mathbf{b}_l)^T x\right)^j - \frac{p_{l,j,i_k}(\mathbf{b}_l^T x)}{j!} \cdot \left((\mathbf{a}_l - \mathbf{b}_l)^T x\right)^j \right| \\
&\qquad \cdot \left(1 - \frac{M}{2 \cdot d \cdot A} \cdot |\mathbf{b}_l^T x - u_{i_k}|\right)_+ \\
&\quad \leq \sum_{j=0}^{q} \max_{\substack{i_k \in \{0,\ldots,M\}, \\ |\mathbf{b}_l^T x - u_{i_k}| \leq 2 \cdot d \cdot A / M}} \left| \frac{g_l^{(j)}(\mathbf{b}_l^T x)}{j!} \cdot \left((\mathbf{a}_l - \mathbf{b}_l)^T x\right)^j \right. \\
&\qquad \left. - \frac{p_{l,j,i_k}(\mathbf{b}_l^T x)}{j!} \cdot \left((\mathbf{a}_l - \mathbf{b}_l)^T x\right)^j \right|
\end{aligned}$$

$$\leq (q+1)\cdot C\cdot d^q\cdot A^q\left(\max\left\{\frac{2\cdot d\cdot A}{M},\|\mathbf{a}_l-\mathbf{b}_l\|_\infty\right\}\right)^p.$$

With

$$p_{i_k,l}(x)=\sum_{j=0}^{q}\frac{p_{l,j,i_k}(\mathbf{b}_l^T x)}{j!}\cdot\left((\mathbf{a}_l-\mathbf{b}_l)^T x\right)^j$$

we get the assertion. □

5.3. Auxiliary results

Lemma 8. *Let $\beta_n=c_6\cdot\log(n)$ for some suitably large constant $c_6>0$. Assume that the distribution of (X,Y) satisfies (14) for some constant $c_4>0$ and that the regression function m is bounded in absolute value. Let $\mathcal{F}_n$ be a set of functions $f:\mathbb{R}^d\to\mathbb{R}$ and assume that the estimate m_n satisfies*

$$m_n=T_{\beta_n}\tilde{m}_n$$

and

$$\tilde{m}_n(\cdot)=\tilde{m}_n\big(\cdot,(X_1,Y_1),\ldots,(X_n,Y_n)\big)\in\mathcal{F}_n$$

and

$$\frac{1}{n}\sum_{i=1}^{n}|Y_i-\tilde{m}_n(X_i)|^2\leq\min_{l\in\Theta_n}\left(\frac{1}{n}\sum_{i=1}^{n}|Y_i-g_{n,l}(X_i)|^2+pen_n(g_{n,l})\right)$$

for some nonempty parameter set Θ_n, some random functions $g_{n,l}:\mathbb{R}^d\to\mathbb{R}$ and some deterministic penalty terms $pen_n(g_{n,l})\geq 0$, where the random function $g_{n,l}:\mathbb{R}^d\to\mathbb{R}$ depend only on random variables

$$\mathbf{b}_1^{(1)},\ldots,\mathbf{b}_r^{(1)},\ldots,\mathbf{b}_1^{(I_n)},\ldots,\mathbf{b}_r^{(I_n)},$$

which are independent of $(X_1,Y_1),(X_2,Y_2),\ldots$.

Then m_n satisfies

$$\begin{aligned}&\mathbf{E}\int|m_n(x)-m(x)|^2\mathbf{P}_X(dx)\\&\quad\leq\frac{c_{13}\cdot(\log n)^2\cdot(\log(\sup_{x_1^n\in(supp(X))^n}\mathcal{N}_1(\frac{1}{n\cdot\beta_n},\mathcal{F}_n,x_1^n))+1)}{n}\\&\qquad+2\cdot\mathbf{E}\left(\min_{l\in\Theta_n}\int|g_{n,l}(x)-m(x)|^2\mathbf{P}_X(dx)+pen_n(g_{n,l})\right)\end{aligned}$$

for $n>1$ and some constant $c_{13}>0$, which does not depend on n.

Proof. This lemma follows in a straightforward way from the proof of Theorem 1 in Bagirov et al. [5]. A complete version of the proof is given in the Appendix A. □

In order to bound the covering number $\mathcal{N}_1(\frac{1}{n\cdot\beta_n}, \mathcal{F}_n, x_1^n)$ we will use the following lemma.

Lemma 9. *Let $a > 0$ and let $d, N, J_n \in \mathbb{N}$ be such that $J_n \leq n^{c_{14}}$ and set $\beta_n = c_6 \cdot \log n$. Let σ be 2-admissible according to Definition 2. Let $\mathcal{F}$ be the set of all functions defined by (4), (5) and (6) where $k_1 = k_2 = \cdots = k_L = 24 \cdot (N+d)$ and the weights are bounded in absolute value by $c_{15} \cdot n^{c_{16}}$. Set*

$$\mathcal{F}^{(J_n)} = \left\{ \sum_{j=1}^{J_n} a_j \cdot f_j \,:\, f_j \in \mathcal{F} \quad \text{and} \quad \sum_{j=1}^{J_n} a_j^2 \leq c_{17} \cdot n^{c_{18}} \right\}.$$

Then we have for $n > 1$

$$\log\left(supp_{x_1^n \in [-a,a]^{d\cdot n}} \mathcal{N}_1\left(\frac{1}{n \cdot \beta_n}, \mathcal{F}^{(J_n)}, x_1^n \right) \right) \leq c_{19} \cdot \log n \cdot J_n$$

for some constant c_{19} which depends only on L, N, a and d.

Proof. Since the networks in $\mathcal{F}^{(J_n)}$ are linear combinations of J_n fully connected neural networks with L hidden layers, a bounded number of neurons in each hidden layers and all weights bounded by a polynomial in n, the result follows by combining Lemma 16.6 in Györfi et al. [19] with Lemma 2 in the Supplement of Bauer et al. [8]. □

5.4. Proof of Theorem 1

Since $supp(\mathbf{P}_X)$ is bounded and m is (p, C)-smooth, we conclude that m is bounded in absolute value, and we can assume without loss of generality that $supp(X) \subseteq [-a_n, a_n]^d$ and $\|m\|_\infty \leq \beta_n$.

Let $\mathcal{F}$ be the set of all functions defined by (4), (5) and (6) where $L = s+2 = \lceil \log_2(N+d) \rceil + 2$, where $k_1 = k_2 = \cdots = k_L = 24 \cdot (N+d)$ and where the weights are bounded in absolute value by $n^{c_{20}}$. Set

$$\mathcal{F}^{(J_n)} = \left\{ \sum_{j=1}^{J_n} a_j \cdot f_j \,:\, f_j \in \mathcal{F} \quad \text{and} \quad \sum_{j=1}^{J_n} a_j^2 \leq c_{21} \cdot n \right\}$$

for c_{21} chosen below, where

$$J_n = (M_n+1)^d \cdot \left| \left\{ (j_1, \ldots, j_d) \,:\, j_1, \ldots, j_d \in \{0, \ldots, N\},\, j_1 + \cdots + j_d \leq N \right\} \right|.$$

Then $J_n \leq (M_n+1)^d \cdot (N+1)^d$.

Let

$$g_n(x) = \sum_{k=1}^{(M_n+1)^d} \sum_{\substack{j_1,\ldots,j_d \in \{0,\ldots,q\} \\ j_1+\cdots+j_d \leq q}} \frac{1}{j_1! \cdots j_d!} \cdot \frac{\partial^{j_1+\cdots+j_d} m}{\partial^{j_1} x^{(1)} \ldots \partial^{j_d} x^{(d)}}(x_{\mathbf{i}_k}) \cdot f_{net, j_1, \ldots, j_d, \mathbf{i}_k}(x).$$

Because of the (p, C)-smoothness of m we know that

$$\max_{k\in\{1,\dots,(M_n+1)^d, j_1,\dots,j_d\in\{0,\dots,q\}, j_1+\cdots+j_d\le q}\left|\frac{\partial^{j_1+\cdots+j_d}m}{\partial^{j_1}x^{(1)}\cdots\partial^{j_d}x^{(d)}}(x_{\mathbf{i}_k})\right| < \infty. \quad (21)$$

Set

$$c_{21} = \max\left\{\frac{1+\mathbf{E}\{Y^2\}}{c_3'}, (N+1)^d \cdot \max\left\{\left|\frac{1}{j_1!\cdots j_d!}\cdot\frac{\partial^{j_1+\cdots+j_d}m}{\partial^{j_1}x^{(1)}\cdots\partial^{j_d}x^{(d)}}(x_{\mathbf{i}_k})\right|^2 : j_1,\dots,j_d\in\{0,\dots,q\},\ j_1+\cdots+j_d\le q\right\}\right\} \quad (22)$$

and let A_n be the event that

$$\frac{1}{n}\sum_{i=1}^n Y_i^2 \le 1+\mathbf{E}\{Y^2\} \quad (23)$$

holds. Then

$$\mathbf{P}(A_n^c) \le \frac{\mathbf{Var}\{Y^2\}}{n} \le \frac{c_{22}}{n}$$

by Chebyshev's inequality.

Set $\hat{m}_n = T_{\beta_n}\tilde{m}_n = m_n$ in case that A_n holds and set $\hat{m}_n = T_{\beta_n}g_n$ otherwise. Then

$$\begin{aligned}
&\mathbf{E}\int |m_n(x)-m(x)|^2\mathbf{P}_X(dx)\\
&\quad\le 4\beta_n^2\cdot\mathbf{P}\{A_n^c\} + \mathbf{E}\left\{\int |m_n(x)-m(x)|^2\mathbf{P}_X(dx)\cdot 1_{A_n}\right\}\\
&\quad\le \frac{4\cdot c_{22}\cdot\beta_n^2}{n} + \mathbf{E}\int |\hat{m}_n(x)-m(x)|^2\mathbf{P}_X(dx).
\end{aligned}$$

The definition of the estimate $\tilde{m}_n$ implies

$$\tilde{m}_n(x) = \sum_{j=1}^{J_n}\hat{a}_j\cdot f_j$$

for some $f_j\in\mathcal{F}$ and some $\hat{a}_j$ satisfying

$$\sum_{j=1}^{J_n}\hat{a}_j^2 \le \frac{1}{n}\sum_{i=1}^n Y_i^2\cdot\frac{n}{c_3'}.$$

Hence on A_n we have

$$\sum_{j=1}^{J_n}\hat{a}_j^2 \le \frac{1+\mathbf{E}Y^2}{c_3'}\cdot n,$$

and consequently we can assume w.l.o.g. that m_n satisfies $m_n = T_{\beta_n}\bar{m}_n$ for some $\bar{m}_n \in \mathcal{F}^{(J_n)}$. And since

$$\begin{aligned}
&\frac{1}{n}\sum_{i=1}^{n}|Y_i - \tilde{m}_n(X_i)|^2 \\
&\quad\le \frac{1}{n}\sum_{i=1}^{n}|Y_i - \tilde{m}_n(X_i)|^2 + \frac{c_3'}{n}\cdot \sum_{k=1}^{(M_n+1)^d} \sum_{\substack{j_1,\dots,j_d\in\{0,\dots,N\}\\ j_1+\cdots+j_d\le N}} a^2_{\mathbf{i}_k,j_1,\dots,j_d} \\
&\quad\le \frac{1}{n}\sum_{i=1}^{n}|Y_i - g_n(X_i)|^2 \\
&\qquad + \frac{c_3'}{n}\cdot \sum_{k=1}^{(M_n+1)^d} \sum_{\substack{j_1,\dots,j_d\in\{0,\dots,q\}\\ j_1+\cdots+j_d\le q}} \left|\frac{1}{j_1!\cdots j_d!}\cdot \frac{\partial^{j_1+\cdots+j_d} m}{\partial^{j_1}x^{(1)}\cdots\partial^{j_d}x^{(d)}}(x_{\mathbf{i}_k})\right|^2
\end{aligned}$$

(by definition of $\tilde{m}_n$) and (21), we also have

$$\frac{1}{n}\sum_{i=1}^{n}|Y_i - \bar{m}_n(X_i)|^2 \le \frac{1}{n}\sum_{i=1}^{n}|Y_i - g_n(X_i)|^2 + c_{23}\cdot\frac{(M_n+1)^d}{n}.$$

Set

$$\begin{aligned}
P_n(x) = &\sum_{k=1}^{(M_n+1)^d} \sum_{\substack{j_1,\dots,j_d\in\{0,\dots,q\}\\ j_1+\cdots+j_d\le q}} \frac{1}{j_1!\cdots j_d!}\cdot \frac{\partial^{j_1+\cdots+j_d} m}{\partial^{j_1}x^{(1)}\cdots\partial^{j_d}x^{(d)}}(x_{\mathbf{i}_k}) \\
&\cdot \left(x^{(1)} - x^{(1)}_{\mathbf{i}_k}\right)^{j_1}\cdots\left(x^{(d)} - x^{(d)}_{\mathbf{i}_k}\right)^{j_d}\cdot\prod_{j=1}^{d}\left(1 - \frac{M_n}{2a}\cdot|x^{(j)} - x^{(j)}_{\mathbf{i}_k}|\right)_+.
\end{aligned}$$

Application of Lemma 8 (with $|\Theta_n| = 1$ and $g_{n,1} = g_n$ deterministic) yields

$$\begin{aligned}
&\mathbf{E}\int |m_n(x) - m(x)|^2 \mathbf{P}_X(dx) \\
&\quad\le \frac{c_{23}\cdot(\log n)^2\cdot(\log(\sup_{x_1^n\in supp(X)^n}\mathcal{N}_1(\frac{1}{n\cdot\beta_n},\mathcal{F}^{(J_n)},x_1^n))+1)}{n} \\
&\qquad + 2\cdot\int |g_n(x) - m(x)|^2\mathbf{P}_X(dx) + 2\cdot c_{21}\cdot\frac{(M_n+1)^d}{n}.
\end{aligned}$$

By Lemma 9 we know that

$$\begin{aligned}
&\frac{c_{23}\cdot\log(n)^2\cdot(\log(\sup_{x_1^n\in supp(X)^n}\mathcal{N}_1(\frac{1}{n\cdot\beta_n},\mathcal{F}^{(J_n)},x_1^n))+1)}{n} \\
&\quad\le c_{24}\cdot\frac{(\log n)^3\cdot(N+1)^d\cdot(M_n+1)^d}{n}.
\end{aligned}$$

Furthermore we have

$$\int |g_n(x) - m(x)|^2 \mathbf{P}_X(dx)$$
$$\leq 2 \cdot \sup_{x \in [-a_n, a_n]^d} |g_n(x) - P_n(x)|^2 + 2 \cdot \sup_{x \in [-a_n, a_n]^d} |P_n(x) - m(x)|^2.$$

By Lemma 5 we know

$$\sup_{x \in [-a_n, a_n]^d} |g_n(x) - P_n(x)| \leq (M_n + 1)^d \cdot (q+1)^d \cdot c_{25} \cdot a_n^{6(N+d)} \cdot M_n^3 \frac{1}{R_n}$$
$$\leq (M_n + 1)^d \cdot (q+1)^d \cdot c_{25} \cdot (\log n) \cdot \frac{M_n^3}{R_n},$$

and Lemma 5 in Schmidt-Hieber [43] implies

$$\sup_{x \in [-a_n, a_n]^d} |P_n(x) - m(x)| \leq c_{26} \cdot \frac{a_n^p}{M_n^p} \leq c_{26} \cdot (\log n) \cdot \frac{1}{M_n^p}.$$

Plugging in the values for R_n and M_n we get the assertion. □

5.5. *Proof of Theorem 2*

W.l.o.g. we assume $supp(X) \subseteq [-A_n, A_n]^d$.

Define the estimate $\bar{m}_n$ exactly like m_n except that for given directions $\mathbf{b}_l$ $(l = 1, \ldots, r)$ we define the neural network estimate $\tilde{m}_n(x)$ by

$$\tilde{m}_n(x) = \sum_{l=1}^{r} \sum_{k=1}^{M_n+1} \sum_{\substack{j_1, \ldots, j_d \in \{0, \ldots, N\} \\ j_1 + \cdots + j_d \leq N}} a_{i_k, j_1, \ldots, j_d, \mathbf{b}_l} \cdot f_{net, j_1, \ldots, j_d, i_k, \mathbf{b}_l}(x),$$

where the coefficients $a_{k, j_1, \ldots, j_d, \mathbf{b}_l}$ are chosen from the set

$$\left\{ (a_{k, j_1, \ldots, j_d, \mathbf{b}_l})_{k, j_1, \ldots, j_d, l} : \sum_{k, j_1, \ldots, j_d, l} a_{k, j_1, \ldots, j_d, \mathbf{b}_l}^2 \leq c_{27} \cdot n^2 \right\}$$

by minimizing

$$\frac{1}{n} \sum_{i=1}^{n} |Y_i - \tilde{m}_n(X_i)|^2 + \frac{c_3''}{n} \cdot \sum_{l=1}^{r} \sum_{k=0}^{K} \sum_{\substack{j_1, \ldots, j_d \in \{0, \ldots, N\} \\ j_1 + \cdots + j_d \leq N}} a_{k, j_1, \ldots, j_d, \mathbf{b}_l}^2$$

for some constant $c_3'' > 0$. Then $\bar{m}_n$ satisfies

$$\bar{m}_n \in \left\{ T_{\beta_n} f : f \in \mathcal{F}^{(J_n)} \right\},$$

where $\mathcal{F}^{(J_n)}$ (with $J_n = r \cdot (M_n + 1) \cdot \binom{N+d}{d}$) is the function space defined in Lemma 9. On the event

$$B_n = \{ |Y_i| \leq \sqrt{n} : i = 1, \ldots, n \}$$

we know by (17) that we have $m_n = \bar{m}_n$ (provided $c_{27} \geq 1/c_3''$). Hence

$$\int |m_n(x) - m(x)|^2 \mathbf{P}_X(dx) \leq \int |\bar{m}_n(x) - m(x)|^2 \mathbf{P}_X(dx) + 4\beta_n^2 \cdot 1_{B_n^c}.$$

By Markov inequality we know

$$\mathbf{P}\{B_n^c\} \leq n \cdot \mathbf{P}\{|Y| > \sqrt{n}\} \leq \frac{n \cdot \mathbf{E}\{e^{c_3'' \cdot Y^2}\}}{\exp(c_3'' \cdot n)},$$

therefore (14) implies that it suffices to show the assertion under the additional assumption

$$\tilde{m}_n(\cdot, (X_1, Y_1), \ldots, (X_n, Y_n)) \in \mathcal{F}^{(J_n)}. \tag{24}$$

By Lemma 7 we know that for each $i \in \{1, \ldots, I_n\}$ there exist coefficients $a_{k,j_1,\ldots,j_d,l}^{(i)} \in [-c_{28} \cdot A_n^p, c_{28} \cdot A_n^p]$, which depend on $\mathbf{a}_l$ and on $\mathbf{b}_l^{(i)}$, but which are independent of (X_1, Y_1), …, (X_n, Y_n), such that we have for all $x \in [-A_n, A_n]^d$

$$\begin{aligned}
&\left| \sum_{l=1}^{r} \sum_{j=0}^{q} \frac{g_l^{(j)}((\mathbf{b}_l^{(i)})^T x)}{j!} \cdot \left((\mathbf{a}_l - \mathbf{b}_l^{(i)})^T x\right)^j \right. \\
&\quad - \sum_{l=0}^{r} \sum_{k=1}^{M_n+1} \sum_{\substack{j_1,\ldots,j_d \in \{0,\ldots,N\} \\ j_1+\cdots+j_d \leq N}} a_{i_k,j_1,\ldots,j_d,l}^{(i)} \cdot \left(x^{(1)}\right)^{j_1} \cdots \left(x^{(d)}\right)^{j_d} \\
&\quad \left. \cdot \left(1 - \frac{M_n}{2 \cdot d \cdot A_n} \cdot |(\mathbf{b}_l^{(i)})^T x - u_{i_k}| \right)_+ \right| \\
&\leq r \cdot 2^p \cdot (p+1) \cdot C \cdot A_n^{2p} \cdot \left(\max\left\{ \frac{1}{M_n}, \max_{l=1,\ldots,r} \|\mathbf{a}_l - \mathbf{b}_l^{(i)}\|_\infty \right\} \right)^p.
\end{aligned} \tag{25}$$

From the definition of the estimate we get

$$\begin{aligned}
&\frac{1}{n} \sum_{i=1}^{n} |Y_i - \tilde{m}_n(X_i)|^2 \\
&\leq \min_{t=1,\ldots,I_n} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left| Y_i - \sum_{l=1}^{r} \sum_{k=1}^{M_n+1} \sum_{\substack{j_1,\ldots,j_d \in \{0,\ldots,N\} \\ j_1+\cdots+j_d \leq N}} a_{i_k,j_1,\ldots,j_d,l}^{(t)} \cdot f_{net,j_1,\ldots,j_d,i_k,\mathbf{b}_l^{(t)}}(X_i) \right|^2 \right. \\
&\quad \left. + \frac{c_3''}{n} \cdot \sum_{l=1}^{r} \sum_{k=1}^{M_n+1} \sum_{\substack{j_1,\ldots,j_d \in \{0,\ldots,N\} \\ j_1+\cdots+j_d \leq N}} \left(a_{i_k,j_1,\ldots,j_d,l}^{(t)}\right)^2 \right\} \\
&\leq \min_{t=1,\ldots,I_n} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left| Y_i - \sum_{l=1}^{r} \sum_{k=1}^{M_n+1} \sum_{\substack{j_1,\ldots,j_d \in \{0,\ldots,N\} \\ j_1+\cdots+j_d \leq N}} a_{i_k,j_1,\ldots,j_d,l}^{(t)} \cdot f_{net,j_1,\ldots,j_d,i_k,\mathbf{b}_l^{(t)}}(X_i) \right|^2 \right. \\
&\quad \left. + c_{29} \cdot A_n^{2p} \cdot r \cdot \binom{N+d}{d} \cdot \frac{M_n}{n} \right\}.
\end{aligned}$$

Hence, application of Lemma 8 and Lemma 9 (together with (24)) yields

$$\begin{aligned}
&\mathbf{E}\int |m_n(x)-m(x)|^2 \mathbf{P}_X(dx)\\
&\quad \le c_{30}\cdot \frac{(\log n)^3\cdot M_n}{n}\\
&\qquad + 2\cdot \mathbf{E}\left(\min_{t=1,\dots,I_n}\int \left|\sum_{l=1}^{r}\sum_{k=1}^{M_n+1}\sum_{\substack{j_1,\dots,j_d\in\{0,\dots,N\}\\ j_1+\cdots+j_d\le N}} a^{(t)}_{i_k,j_1,\dots,j_d,l}\cdot f_{net,j_1,\dots,j_d,i_k,\mathbf{b}_l^{(t)}}(x)\right.\right.\\
&\qquad \left.\left. - m(x)\right|^2 \mathbf{P}_X(dx)\right) + c_{31}\cdot(\log n)\cdot n^{-\frac{2p}{2p+1}}.
\end{aligned}$$

Because of $(a+b+c)^2 \le 3a^2+3b^2+3c^2$ $(a,b,c\in\mathbb{R})$ we have

$$\begin{aligned}
&\int \left|\sum_{l=1}^{r}\sum_{k=1}^{M_n+1}\sum_{\substack{j_1,\dots,j_d\in\{0,\dots,N\}\\ j_1+\cdots+j_d\le N}} a^{(t)}_{i_k,j_1,\dots,j_d,l}\cdot f_{net,j_1,\dots,j_d,i_k,\mathbf{b}_l^{(t)}}(x) - m(x)\right|^2 \mathbf{P}_X(dx)\\
&\quad \le 3\cdot\int \left|\sum_{l=1}^{r}\sum_{k=1}^{M_n+1}\sum_{\substack{j_1,\dots,j_d\in\{0,\dots,N\}\\ j_1+\cdots+j_d\le N}} a^{(t)}_{i_k,j_1,\dots,j_d,l}\cdot f_{net,j_1,\dots,j_d,i_k,\mathbf{b}_l^{(t)}}(x)\right.\\
&\qquad - \sum_{l=1}^{r}\sum_{k=1}^{M_n+1}\sum_{\substack{j_1,\dots,j_d\in\{0,\dots,N\}\\ j_1+\cdots+j_d\le N}} a^{(t)}_{i_k,j_1,\dots,j_d,l}\cdot \left(x^{(1)}\right)^{j_1}\cdots\left(x^{(d)}\right)^{j_d}\\
&\qquad \left.\cdot\left(1-\frac{M_n}{2\cdot d\cdot A_n}\cdot|\left(\mathbf{b}_l^{(t)}\right)^T x-u_{i_k}|\right)_+\right|^2 \mathbf{P}_X(dx)\\
&\qquad + 3\cdot\int \left|\sum_{l=1}^{r}\sum_{k=1}^{M_n+1}\sum_{\substack{j_1,\dots,j_d\in\{0,\dots,N\}\\ j_1+\cdots+j_d\le N}} a^{(t)}_{i_k,j_1,\dots,j_d,l}\cdot \left(x^{(1)}\right)^{j_1}\cdots\left(x^{(d)}\right)^{j_d}\right.\\
&\qquad \cdot\left(1-\frac{M_n}{2\cdot d\cdot A_n}\cdot|\left(\mathbf{b}_l^{(t)}\right)^T x-u_{i_k}|\right)_+\\
&\qquad \left.- \sum_{l=1}^{r}\sum_{j=0}^{q}\frac{g_l^{(j)}((\mathbf{b}_l^{(t)})^T x)}{j!}\cdot\left(\left(\mathbf{a}_l-\mathbf{b}_l^{(t)}\right)^T x\right)^j\right|^2 \mathbf{P}_X(dx)\\
&\qquad + 3\cdot\int \left|\sum_{l=1}^{r}\sum_{j=0}^{q}\frac{g_l^{(j)}(\mathbf{b}_l^T x)}{j!}\cdot\left(\left(\mathbf{a}_l-\mathbf{b}_l^{(t)}\right)^T x\right)^j - m(x)\right|^2 \mathbf{P}_X(dx).
\end{aligned}$$

Application of Lemma 5 implies for all $x\in[-A_n,A_n]^d$

$$\left|\sum_{l=1}^{r}\sum_{k=1}^{M_n+1}\sum_{\substack{j_1,\dots,j_d\in\{0,\dots,N\}\\ j_1+\cdots+j_d\le N}} a^{(t)}_{k,j_1,\dots,j_d,l}\cdot f_{net,k,j_1,\dots,j_d,\mathbf{b}_l^{(t)}}(x)\right.$$

$$
\begin{aligned}
&- \sum_{l=1}^{r} \sum_{k=1}^{M_n+1} \sum_{\substack{j_1,\dots,j_d \in \{0,\dots,N\} \\ j_1+\dots+j_d \le N}} a_{k,j_1,\dots,j_d,l}^{(t)} \cdot \left(x^{(1)}\right)^{j_1} \cdots \left(x^{(d)}\right)^{j_d} \\
&\cdot \left(1 - \frac{K}{2 \cdot d \cdot A} \cdot |(\mathbf{b}_l^{(t)})^T x - u_k|\right)_+ \Bigg|^2 \\
&\le r^2 \cdot (M_n+1)^2 \cdot (N+d)^{2d} \cdot c_{28}^2 \cdot A_n^{2p} \cdot c_{12}^2 \cdot 3^{6 \cdot 3^s} \cdot A_n^{6 \cdot 2^s} \cdot M_n^6 \cdot \frac{1}{R_n^2} \\
&\le c_{32} \cdot \frac{(\log n)^2}{n}.
\end{aligned}
$$

By Lemma 6 we have for all $x \in [-A_n, A_n]^d$

$$
\left| \sum_{l=1}^{r} \sum_{j=0}^{q} \frac{g_l^{(j)}((\mathbf{b}_l^{(t)})^T x)}{j!} \cdot \left((\mathbf{a}_l - \mathbf{b}_l)^T x\right)^j - m(x) \right|^2 \le c_{33} \cdot A_n^{2p} \cdot \|\mathbf{a}_l - \mathbf{b}_l^{(t)}\|_\infty^{2p}.
$$

Using this together with (25) we see that it remains to show

$$
\mathbf{E}\left\{ \min_{i=1,\dots,I_n} \max_{s=1,\dots,r} \|\mathbf{b}_s^{(i)} - \mathbf{a}_s\|_\infty^{2p} \right\} \le c_{34} \cdot (\log n)^2 \cdot n^{-\frac{2p}{2p+1}}.
$$

By the random choice of the $\mathbf{b}_l^{(i)}$ we know for any $t \in (0,1]$

$$
\begin{aligned}
\mathbf{P}\left\{ \min_{i=1,\dots,I_n} \max_{l=1,\dots,r} \|\mathbf{b}_l^{(i)} - \mathbf{a}_l\|_\infty > t \right\} &= \prod_{i=1}^{I_n} \left(1 - \mathbf{P}\left\{ \max_{l=1,\dots,r} \|\mathbf{b}_l^{(i)} - \mathbf{a}_l\|_\infty \le t \right\}\right) \\
&\le \left(1 - \left(\frac{t}{2}\right)^{r \cdot d}\right)^{I_n}
\end{aligned}
$$

from which we conclude

$$
\begin{aligned}
&\mathbf{E}\left\{ \min_{i=1,\dots,I_n} \max_{l=1,\dots,r} \|\mathbf{b}_l^{(i)} - \mathbf{a}_l\|_\infty^{2p} \right\} \\
&\le \left(\frac{\log n}{n}\right)^{\frac{2p}{2p+1}} + 2^{2p} \cdot \mathbf{P}\left\{ \min_{i=1,\dots,I_n} \max_{l=1,\dots,r} \|\mathbf{b}_l^{(i)} - \mathbf{a}_l\|_\infty > \left(\frac{\log n}{n}\right)^{\frac{1}{2p+1}} \right\} \\
&\le \left(\frac{\log n}{n}\right)^{\frac{2p}{2p+1}} + 2^{2p} \cdot \left(1 - \cdot \frac{1}{2^{r \cdot d}} \left(\frac{\log n}{n}\right)^{\frac{r \cdot d}{2p+1}}\right)^{I_n} \\
&\le c_{35} \cdot \exp\left(-I_n \cdot \frac{1}{2^{r \cdot d}} \cdot \left(\frac{\log n}{n}\right)^{\frac{r \cdot d}{2p+1}}\right) \\
&= c_{35} \cdot \exp\left(-\frac{c_9}{2^{r \cdot d}} \cdot (\log n)^2\right) \\
&\le c_{35} \cdot \left(\frac{\log n}{n}\right)^{\frac{2p}{2p+1}}
\end{aligned}
$$

where the last inequality follows from

$$I_n \geq c_9 \cdot (\log n)^2 \cdot \left(\frac{n}{\log n}\right)^{\frac{r \cdot d}{2p+1}}.$$

Putting together the above results we get the assertion. □

Appendix A: Supplementary Material

A.1. Computation of the linear neural network estimate

The estimate in Subsection 2.2 is given by

$$\tilde{m}_n(x) = \sum_{j=1}^{J} a_j \cdot B_j(x) \tag{26}$$

where $\mathbf{a} = (a_j)_{j=1,\ldots,J} \in \mathbb{R}^J$ minimizes

$$\begin{aligned}&\frac{1}{n}(\mathbf{Y} - \mathbf{B}a)^T(\mathbf{Y} - \mathbf{B}a) + \frac{c_3'}{n} \cdot \mathbf{a}^T\mathbf{a} \\ &\quad = \frac{1}{n}\big(\mathbf{Y}^T\mathbf{Y} - 2\mathbf{Y}^T\mathbf{B}a\big) + \mathbf{a}^T\left(\frac{1}{n}\mathbf{B}^T\mathbf{B} + \frac{c_3'}{n} \cdot \mathbf{I}\right)\mathbf{a}.\end{aligned}$$

Since the matrix

$$\mathbf{A} = \frac{1}{n}\mathbf{B}^T\mathbf{B} + \frac{c_3'}{n} \cdot \mathbf{I}$$

is positive definite, its inverse matrix $\mathbf{A}^{-1}$ exists and it is easy to see that we have

$$\begin{aligned}&\frac{1}{n} \cdot \big(\mathbf{Y}^T\mathbf{Y} - 2\mathbf{Y}^T\mathbf{B}a\big) + \mathbf{a}^T\left(\frac{1}{n}\mathbf{B}^T\mathbf{B} + \frac{c_3'}{n} \cdot \mathbf{I}\right)\mathbf{a} \\ &\quad = \left(\mathbf{a} - \frac{1}{n} \cdot \mathbf{A}^{-1}\mathbf{B}^T\mathbf{Y}\right)^T \mathbf{A}\left(\mathbf{a} - \frac{1}{n} \cdot \mathbf{A}^{-1}\mathbf{B}^T\mathbf{Y}\right) \\ &\qquad + \frac{1}{n}\mathbf{Y}^T\mathbf{Y} - \frac{1}{n^2} \cdot \mathbf{Y}^T\mathbf{B}\mathbf{A}^{-1}\mathbf{B}^T\mathbf{Y}.\end{aligned}$$

The last expression is minimal for $\mathbf{a} = \frac{1}{n} \cdot \mathbf{A}^{-1}\mathbf{B}^T\mathbf{Y}$, which proves that the vector of coefficients of our estimate (26) is the unique solution of the linear equation system (13).

A.2. Proof of Lemma 5

Define $g_1^{(0)}$ by backward recursion:

$$g_k^{(s)}(x) = x^{(l)} - y^{(l)}$$

for $j_1 + j_2 + \cdots + j_{l-1} + 1 \leq k \leq j_1 + j_2 + \cdots + j_l$ and $l = 1, \ldots, d$,

$$g^{(s)}_{j_1+j_2+\cdots+j_d+k}(x) = \left(1 - \frac{M}{2a} \cdot |x^{(k)} - y^{(k)}|\right)_+$$

for $k = 1, \ldots, d$, and

$$g^{(s)}_k(x) = 1$$

for $k = j_1 + j_2 + \cdots + j_d + d + 1, j_1 + j_2 + \cdots + j_d + d + 2, \ldots, 2^s$, and

$$g^{(l)}_k(x) = g^{(l+1)}_{2k-1}(x) \cdot g^{(l+1)}_{2k}(x)$$

for $k \in \{1, 2, \ldots, 2^l\}$ and $l \in \{0, \ldots, s-1\}$.

Then we have for any $l \in \{0, \ldots, s\}$, $k \in \{1, \ldots, 2^l\}$ and $x \in [-a, a]$

$$|g^{(l)}_k(x)| \leq (2a)^{2^{s-l}}.$$

By Lemma 2 the network f_{mult} satisfies for any $l \in \{0, \ldots, s\}$ and $x, y \in [-3^{3^{s-l}} \cdot a^{2^{s-l}}, 3^{3^{s-l}} \cdot a^{2^{s-l}}]$

$$|f_{mult}(x, y) - x \cdot y| \leq \frac{20 \cdot \|\sigma'''\|_\infty}{3 \cdot |\sigma''(t_\sigma)|} \cdot 3^{3 \cdot 3^{s-l}} \cdot a^{3 \cdot 2^{s-l}} \cdot \frac{1}{R}.$$

Furthermore we have by Lemma 1 and Lemma 4 for any $x \in [-3a, 3a]$

$$|f_{id}(x) - x| \leq \frac{9 \cdot \|\sigma''\|_\infty \cdot a^2}{2 \cdot |\sigma'(t_{\sigma,id})|} \cdot \frac{1}{R} \tag{27}$$

and for any $x \in [-a, a]^d$

$$\begin{aligned} &\left| f_{hat,y}(x) - \left(1 - \frac{M}{2a} \cdot |x - y|\right)_+ \right| \\ &\quad \leq 1792 \cdot \frac{\max\{\|\sigma''\|_\infty, \|\sigma'''\|_\infty, 1\}}{\min\{2 \cdot |\sigma'(t_{\sigma.id})|, |\sigma''(t_\sigma)|, 1\}} \cdot M^3 \cdot \frac{1}{R}. \end{aligned} \tag{28}$$

From this and (20) we can recursively conclude

$$|f^{(l)}_k(x)| \leq 3^{3^{s-l}} \cdot a^{2^{s-l}}$$

for $k \in \{1, \ldots, 2^l\}$ and $l \in \{0, \ldots, s\}$.

In order to prove the assertion of Lemma 5 we show in the sequel

$$|f^{(l)}_k(x) - g^{(l)}_k(x)| \leq c_{36} \cdot 3^{3 \cdot 3^{s-l}} \cdot a^{3 \cdot 2^{s-l}} \cdot M^3 \cdot \frac{1}{R}$$

for $k \in \{1, \ldots, 2^l\}$ and $l \in \{0, \ldots, s\}$, where

$$c_{36} = \max\left\{\frac{20 \cdot \|\sigma'''\|_\infty}{3 \cdot |\sigma''(t_\sigma)|}, \frac{9 \cdot \|\sigma''\|_\infty}{|\sigma'(t_{\sigma,id})|}, 1792 \cdot \frac{\max\{\|\sigma''\|_\infty, \|\sigma'''\|_\infty, 1\}}{\min\{2 \cdot |\sigma'(t_{\sigma.id})|, |\sigma''(t_\sigma)|, 1\}}\right\}.$$

For $s = l$ this is a consequence of (27), and (28). For $l \in \{0, 1, \dots, s-1\}$ we can conclude via induction

$$\begin{aligned}
&|f_k^{(l)}(x) - g_k^{(l)}(x)| \\
&\quad \le |f_{mult}\big(f_{2k-1}^{(l+1)}(x), f_{2k}^{(l+1)}(x)\big) - f_{2k-1}^{(l+1)}(x) \cdot f_{2k}^{(l+1)}(x)| \\
&\qquad + |f_{2k-1}^{(l+1)}(x) \cdot f_{2k}^{(l+1)}(x) - g_{2k-1}^{(l+1)}(x) \cdot f_{2k}^{(l+1)}(x)| \\
&\qquad + |g_{2k-1}^{(l+1)}(x) \cdot f_{2k}^{(l+1)}(x) - g_{2k-1}^{(l+1)}(x) \cdot g_{2k}^{(l+1)}(x)| \\
&\quad \le c_{36} \cdot 3^{3 \cdot 3^{s-l-1}} \cdot a^{3 \cdot 2^{s-l-1}} \cdot \frac{1}{R} + 3^{3^{s-l-1}} \cdot a^{2^{s-l-1}} \cdot 2 \cdot c_{36} \cdot 3^{3 \cdot 3^{s-l-1}} \\
&\qquad \cdot a^{3 \cdot 2^{s-l-1}} \cdot M^3 \cdot \frac{1}{R} \\
&\quad \le c_{36} \cdot \big(3^{3^{s-l}} + 2 \cdot 3^{4 \cdot 3^{s-l-1}}\big) \cdot a^{3 \cdot 2^{s-l}} \cdot M^3 \cdot \frac{1}{R} \\
&\quad \le c_{36} \cdot 3^{3 \cdot 3^{s-l}} \cdot a^{3 \cdot 2^{s-l}} \cdot M^3 \cdot \frac{1}{R}.
\end{aligned}$$

□

A.3. Lemma 5 in the context of projection pursuit

Lemma 10. *Let $M \in \mathbb{N}$ and let $\sigma : \mathbb{R} \to [0,1]$ be 2-admissible according to Definition 2. Let $A \ge 1$, $\mathbf{b} \in \mathbb{R}^d$ with $\|\mathbf{b}\| \le 1$ and*

$$\begin{aligned}
R \ge \max\Bigg\{ &\frac{\|\sigma''\|_\infty \cdot (M+1)}{2 \cdot |\sigma'(t_{\sigma,id})|}, \frac{9 \cdot \|\sigma''\|_\infty \cdot A}{|\sigma'(t_{\sigma,id})|}, \\
&\frac{20 \cdot \|\sigma'''\|_\infty}{3 \cdot |\sigma''(t_\sigma)|} \cdot 3^{3 \cdot 3^s} \cdot A^{3 \cdot 2^s}, 1792 \cdot \frac{\max\{\|\sigma''\|_\infty, \|\sigma'''\|_\infty, 1\}}{\min\{2 \cdot |\sigma'(t_{\sigma,id})|, |\sigma''(t_\sigma)|, 1\}} \cdot d^{3/2} \cdot M^3 \Bigg\}
\end{aligned} \tag{29}$$

and let $y \in [-A, A]$. Let $N \in \mathbb{N}$ and let $j_1, \dots, j_d \in \mathbb{N}_0$ such that $j_1 + \cdots + j_d \le N$, and set $s = \lceil \log_2(N+1) \rceil$. Let f_{id}, f_{mult} and $\bar{f}_{hat,z}$ (for $z \in \mathbb{R}$) be the neural networks defined in Subsection 3.2. (So in particular $\bar{f}_{hat,z}$ is the neural network from Lemma 4 with $y = z$ and $a = d \cdot A$.) Define the network $f_{net,j_1,\dots,j_d,y}$ by

$$f_{net,j_1,\dots,j_d,y}(x) = f_1^{(0)}(x),$$

where $f_1^{(0)}$ is defined by backward recursion as follows:

$$f_k^{(l)}(x) = f_{mult}\big(f_{2k-1}^{(l+1)}(x), f_{2k}^{(l+1)}(x)\big)$$

for $k \in \{1, 2, \dots, 2^l\}$ and $l \in \{0, \dots, s-1\}$, and

$$f_k^{(s)}(x) = f_{id}\big(f_{id}\big(x^{(l)}\big)\big)$$

for $j_1 + j_2 + \cdots + j_{l-1} + 1 \le k \le j_1 + j_2 + \cdots + j_l$ and $l = 1, \dots, d$,

$$f_{j_1+j_2+\cdots+j_d+1}^{(s)}(x) = \bar{f}_{hat,y}\big(\mathbf{b}^T x\big),$$

and

$$f_k^{(s)}(x) = 1$$

for $k = j_1 + j_2 + \cdots + j_d + 2, j_1 + j_2 + \cdots + j_d + 3, \ldots, 2^s$. *Then we have for any* $x \in [-A, A]^d$*:*

$$\left| f_{net,y}(x) - \left(x^{(1)}\right)^{j_1} \cdots \left(x^{(d)}\right)^{j_d} \cdot \left(1 - \frac{M}{2 \cdot d \cdot A} \cdot |\mathbf{b}^T x - y|\right)_+ \right|$$
$$\leq c_{37} \cdot 3^{3 \cdot 3^s} \cdot A^{3 \cdot 2^s} \cdot M^3 \cdot \frac{1}{R}.$$

Proof. Define $g_1^{(0)}$ by backward recursion:

$$g_k^{(s)}(x) = x^{(l)}$$

for $j_1 + j_2 + \cdots + j_{l-1} + 1 \leq k \leq j_1 + j_2 + \cdots + j_l$ and $l = 1, \ldots, d$,

$$g_{j_1 + j_2 + \cdots + j_d + 1}^{(s)}(x) = \left(1 - \frac{M}{2 \cdot d \cdot A} \cdot |\mathbf{b}^T x - y|\right)_+,$$

and

$$g_k^{(s)}(x) = 1$$

for $k = j_1 + j_2 + \cdots + j_d + 2, j_1 + j_2 + \cdots + j_d + 3, \ldots, 2^s$, and

$$g_k^{(l)}(x) = g_{2k-1}^{(l+1)}(x) \cdot g_{2k}^{(l+1)}(x)$$

for $k \in \{1, 2, \ldots, 2^l\}$ and $l \in \{0, \ldots, s-1\}$.

Then we have for any $x \in [-A, A]^d$

$$|g_k^{(l)}(x)| \leq A^{2^{s-l}}.$$

By Lemma 2 the network f_{mult} satisfies for any $l \in \{0, \ldots, s\}$ and $x, y \in [-3^{3^{s-l}} \cdot A^{2^{s-l}}, 3^{3^{s-l}} \cdot A^{2^{s-l}}]$

$$|f_{mult}(x, y) - x \cdot y| \leq \frac{20 \cdot \|\sigma'''\|_\infty}{3 \cdot |\sigma''(t_\sigma)|} \cdot 3^{3 \cdot 3^{s-l}} \cdot A^{3 \cdot 2^{s-l}} \cdot \frac{1}{R}.$$

Furthermore we have by Lemma 1 and Lemma 4 for any $x \in [-3A, 3A]$

$$|f_{id}(x) - x| \leq \frac{9 \cdot \|\sigma''\|_\infty \cdot A^2}{2 \cdot |\sigma'(t_{\sigma,id})|} \cdot \frac{1}{R} \tag{30}$$

and for any $x \in [-A, A]^d$

$$\left| \bar{f}_{hat,y}(x) - \left(1 - \frac{M}{2 \cdot d \cdot A} \cdot |\mathbf{b}^T x - y|\right)_+ \right|$$
$$\leq 1792 \cdot \frac{\max\{\|\sigma''\|_\infty, \|\sigma'''\|_\infty, 1\}}{\min\{2 \cdot |\sigma'(t_{\sigma.id})|, |\sigma''(t_\sigma)|, 1\}} \cdot M^3 \cdot \frac{1}{R}. \tag{31}$$

From this and (29) we can recursively conclude

$$|f_k^{(l)}(x)| \le 3^{3^{s-l}} \cdot A^{2^{s-l}}$$

for $k \in \{1, \dots, 2^l\}$ and $l \in \{0, \dots, s\}$.

In order to prove the assertion of Lemma 5 we show in the sequel

$$|f_k^{(l)}(x) - g_k^{(l)}(x)| \le c_{37} \cdot 3^{3 \cdot 3^{s-l}} \cdot A^{3 \cdot 2^{s-l}} \cdot M^3 \cdot \frac{1}{R}$$

for $k \in \{1, \dots, 2^l\}$ and $l \in \{0, \dots, s\}$, where

$$c_{37} = \max\left\{ \frac{20 \cdot \|\sigma'''\|_\infty}{3 \cdot |\sigma''(t_\sigma)|}, \frac{9 \cdot \|\sigma''\|_\infty}{|\sigma'(t_{\sigma,id})|}, 1792 \cdot \frac{\max\{\|\sigma''\|_\infty, \|\sigma'''\|_\infty, 1\}}{\min\{2 \cdot |\sigma'(t_{\sigma.id})|, |\sigma''(t_\sigma)|, 1\}} \right\}.$$

For $s = l$ this is a consequence of (30) and (31). For $l \in \{0, 1, \dots, s-1\}$ we can conclude via induction

$$\begin{aligned}
&\big|f_k^{(l)}(x) - g_k^{(l)}(x)\big| \\
&\quad \le \big|f_{mult}\big(f_{2k-1}^{(l+1)}(x), f_{2k}^{(l+1)}(x)\big) - f_{2k-1}^{(l+1)}(x) \cdot f_{2k}^{(l+1)}(x)\big| \\
&\qquad + \big|f_{2k-1}^{(l+1)}(x) \cdot f_{2k}^{(l+1)}(x) - g_{2k-1}^{(l+1)}(x) \cdot f_{2k}^{(l+1)}(x)\big| \\
&\qquad + \big|g_{2k-1}^{(l+1)}(x) \cdot f_{2k}^{(l+1)}(x) - g_{2k-1}^{(l+1)}(x) \cdot g_{2k}^{(l+1)}(x)\big| \\
&\quad \le c_{37} \cdot 3^{3 \cdot 3^{s-l-1}} \cdot A^{3 \cdot 2^{s-l-1}} \cdot \frac{1}{R} + 3^{3^{s-l-1}} \cdot A^{2^{s-l-1}} \cdot 2 \cdot c_{37} \cdot 3^{3 \cdot 3^{s-l-1}} \\
&\qquad \cdot A^{3 \cdot 2^{s-l-1}} \cdot M^3 \cdot \frac{1}{R} \\
&\quad \le c_{37} \cdot \big(3^{3^{s-l}} + 2 \cdot 3^{4 \cdot 3^{s-l-1}}\big) \cdot A^{3 \cdot 2^{s-l}} \cdot M^3 \cdot \frac{1}{R} \\
&\quad \le c_{37} \cdot 3^{3 \cdot 3^{s-l}} \cdot A^{3 \cdot 2^{s-l}} \cdot M^3 \cdot \frac{1}{R}.
\end{aligned}$$

□

A.4. Proof of Lemma 8

In the proof we use the following error decomposition:

$$\begin{aligned}
&\int |m_n(x) - m(x)|^2 \mathbf{P}_X(dx) \\
&\quad = \big[\mathbf{E}\big\{|m_n(X) - Y|^2 | \mathcal{D}_n\big\} - \mathbf{E}\big\{|m(X) - Y|^2\big\} \\
&\qquad - \big(\mathbf{E}\big\{|m_n(X) - T_{\beta_n} Y|^2 | \mathcal{D}_n\big\} - \mathbf{E}\big\{|m_{\beta_n}(X) - T_{\beta_n} Y|^2\big\}\big)\big] \\
&\qquad + \Bigg[\mathbf{E}\big\{|m_n(X) - T_{\beta_n} Y|^2 | \mathcal{D}_n\big\} - \mathbf{E}\big\{|m_{\beta_n}(X) - T_{\beta_n} Y|^2\big\} \\
&\qquad - 2 \cdot \frac{1}{n} \sum_{i=1}^n \big(|m_n(X_i) - T_{\beta_n} Y_i|^2 - |m_{\beta_n}(X_i) - T_{\beta_n} Y_i|^2\big)\Bigg]
\end{aligned}$$

$$
\begin{aligned}
&+\left[2\cdot\frac{1}{n}\sum_{i=1}^{n}|m_n(X_i)-T_{\beta_n}Y_i|^2-2\cdot\frac{1}{n}\sum_{i=1}^{n}|m_{\beta_n}(X_i)-T_{\beta_n}Y_i|^2\right.\\
&\left.-\left(2\cdot\frac{1}{n}\sum_{i=1}^{n}|m_n(X_i)-Y_i|^2-2\cdot\frac{1}{n}\sum_{i=1}^{n}|m(X_i)-Y_i|^2\right)\right]\\
&+\left[2\left(\frac{1}{n}\sum_{i=1}^{n}|m_n(X_i)-Y_i|^2-\frac{1}{n}\sum_{i=1}^{n}|m(X_i)-Y_i|^2\right)\right]\\
&=\sum_{i=1}^{4}T_{i,n},
\end{aligned}
$$

where $T_{\beta_n}Y$ is the truncated version of Y and m_{β_n} is the regression function of $T_{\beta_n}Y$, i.e.,

$$m_{\beta_n}(x)=\mathbf{E}\{T_{\beta_n}Y|X=x\}.$$

We start with bounding $T_{1,n}$. By using $a^2-b^2=(a-b)(a+b)$ we get

$$
\begin{aligned}
T_{1,n}&=\mathbf{E}\big\{|m_n(X)-Y|^2-|m_n(X)-T_{\beta_n}Y|^2|\mathcal{D}_n\big\}\\
&\quad-\mathbf{E}\big\{|m(X)-Y|^2-|m_{\beta_n}(X)-T_{\beta_n}Y|^2\big\}\\
&=\mathbf{E}\big\{(T_{\beta_n}Y-Y)\big(2m_n(X)-Y-T_{\beta_n}Y\big)|\mathcal{D}_n\big\}\\
&\quad-\mathbf{E}\big\{\big(\big(m(X)-m_{\beta_n}(X)\big)+(T_{\beta_n}Y-Y)\big)\big(m(X)+m_{\beta_n}(X)-Y-T_{\beta_n}Y\big)\big\}\\
&=T_{5,n}+T_{6,n}.
\end{aligned}
$$

With the Cauchy-Schwarz inequality and

$$I_{\{|Y|>\beta_n\}}\le\frac{\exp(c_4/2\cdot|Y|^2)}{\exp(c_4/2\cdot\beta_n^2)}\tag{32}$$

we conclude

$$
\begin{aligned}
|T_{5,n}|&\le\sqrt{\mathbf{E}\big\{|T_{\beta_n}Y-Y|^2\big\}}\cdot\sqrt{\mathbf{E}\big\{|2m_n(X)-Y-T_{\beta_n}Y|^2|\mathcal{D}_n\big\}}\\
&\le\sqrt{\mathbf{E}\big\{|Y|^2\cdot I_{\{|Y|>\beta_n\}}\big\}}\cdot\sqrt{\mathbf{E}\big\{2\cdot|2m_n(X)-T_{\beta_n}Y|^2+2\cdot|Y|^2|\mathcal{D}_n\big\}}\\
&\le\sqrt{\mathbf{E}\left\{|Y|^2\cdot\frac{\exp(c_4/2\cdot|Y|^2)}{\exp(c_4/2\cdot\beta_n^2)}\right\}}\\
&\quad\cdot\sqrt{\mathbf{E}\big\{2\cdot|2m_n(X)-T_{\beta_n}Y|^2|\mathcal{D}_n\big\}+2\mathbf{E}\big\{|Y|^2\big\}}\\
&\le\sqrt{\mathbf{E}\big\{|Y|^2\cdot\exp\big(c_4/2\cdot|Y|^2\big)\big\}}\cdot\exp\left(-\frac{c_4\cdot\beta_n^2}{4}\right)\cdot\sqrt{2(3\beta_n)^2+2\mathbf{E}\big\{|Y|^2\big\}}.
\end{aligned}
$$

With $x\le\exp(x)$ for $x\in\mathbb{R}$ we get

$$|Y|^2\le\frac{2}{c_4}\cdot\exp\left(\frac{c_4}{2}\cdot|Y|^2\right)$$

and hence $\mathbf{E}\{|Y|^2 \cdot \exp(c_4/2 \cdot |Y|^2)\}$ is bounded by

$$\mathbf{E}\Big(\frac{2}{c_4} \cdot \exp\big(c_4/2 \cdot |Y|^2\big) \cdot \exp\big(c_4/2 \cdot |Y|^2\big)\Big) \leq \mathbf{E}\Big(\frac{2}{c_4} \cdot \exp\big(c_4 \cdot |Y|^2\big)\Big) \leq c_{38}$$

which is less than infinity by the assumptions of the lemma. Furthermore the third term is bounded by $\sqrt{18\beta_n^2 + c_{39}}$ because

$$\mathbf{E}\big(|Y|^2\big) \leq \mathbf{E}(1/c_4 \cdot \exp\big(c_4 \cdot |Y|^2\big) \leq c_{39} < \infty, \tag{33}$$

which follows again as above. With the setting $\beta_n = c_6 \cdot \log(n)$ it follows for some constants $c_{40}, c_{41} > 0$ that

$$|T_{5,n}| \leq \sqrt{c_{38}} \cdot \exp\big(-c_{40} \cdot \log(n)^2\big) \cdot \sqrt{\big(18 \cdot c_6^2 \cdot (\log n)^2 + c_{39}\big)} \leq c_{41} \cdot \frac{\log(n)}{n}.$$

By the Cauchy-Schwarz inequality we get

$$\begin{aligned} T_{6,n} \leq \sqrt{2 \cdot \mathbf{E}\big\{|\big(m(X) - m_{\beta_n}(X)\big)|^2\big\} + 2 \cdot \mathbf{E}\big\{|(T_{\beta_n} Y - Y)|^2\big\}} \\ \cdot \sqrt{\mathbf{E}\big\{|m(X) + m_{\beta_n}(X) - Y - T_{\beta_n} Y|^2\big\}}, \end{aligned}$$

where we can bound the second factor on the right-hand side in the above inequality in the same way we have bounded the second factor in $T_{5,n}$, because by assumption $\|m\|_\infty$ is bounded and furthermore m_{β_n} is bounded by β_n. Thus we get for some constant $c_{42} > 0$

$$\sqrt{\mathbf{E}\big\{|m(X) + m_{\beta_n}(X) - Y - T_{\beta_n} Y|^2\big\}} \leq c_{42} \cdot \log(n).$$

Next we consider the first term. By Jensen's inequality it follows that

$$\mathbf{E}\big\{|m(X) - m_{\beta_n}(X)|^2\big\} \leq \mathbf{E}\big\{\mathbf{E}\big(|Y - T_{\beta_n} Y|^2 | X\big)\big\} = \mathbf{E}\big\{|Y - T_{\beta_n} Y|^2\big\}.$$

Hence we get

$$T_{6,n} \leq \sqrt{4 \cdot \mathbf{E}\big\{|Y - T_{\beta_n} Y|^2\big\}} \cdot c_{42} \cdot \log(n)$$

and therefore with the calculations from $T_{5,n}$ it follows that $T_{6,n} \leq c_{43} \cdot \log(n)/n$ for some constant $c_{43} > 0$. Altogether we get

$$T_{1,n} \leq c_{44} \cdot \frac{\log(n)}{n}$$

for some constant $c_{44} > 0$.

Next we consider $T_{2,n}$ and conclude for $t > 0$

$$\begin{aligned} &\mathbf{P}\{T_{2,n} > t\} \\ &\quad \leq \mathbf{P}\left\{\exists f \in T_{\beta_n, supp(X)} \mathcal{F}_n : \mathbf{E}\left(\left|\frac{f(X)}{\beta_n} - \frac{T_{\beta_n} Y}{\beta_n}\right|^2\right) - \mathbf{E}\left(\left|\frac{m_{\beta_n}(X)}{\beta_n} - \frac{T_{\beta_n} Y}{\beta_n}\right|^2\right)\right. \end{aligned}$$

$$-\frac{1}{n}\sum_{i=1}^{n}\left(\left|\frac{f(X_i)}{\beta_n}-\frac{T_{\beta_n}Y_i}{\beta_n}\right|^2-\left|\frac{m_{\beta_n}(X_i)}{\beta_n}-\frac{T_{\beta_n}Y_i}{\beta_n}\right|^2\right)$$
$$>\frac{1}{2}\left(\frac{t}{\beta_n^2}+\mathbf{E}\left(\left|\frac{f(X)}{\beta_n}-\frac{T_{\beta_n}Y}{\beta_n}\right|^2\right)-\mathbf{E}\left(\left|\frac{m_{\beta_n}(X)}{\beta_n}-\frac{T_{\beta_n}Y}{\beta_n}\right|^2\right)\right)\Bigg\},$$

where $T_{\beta_n,supp(X)}\mathcal{F}_n$ is defined as $\{T_{\beta_n}f\cdot 1_{supp(X)} : f\in\mathcal{F}_n\}$. Theorem 11.4 in Györfi et al. [19] and the relation

$$\mathcal{N}_1\left(\delta,\left\{\frac{1}{\beta_n}g : g\in\mathcal{G}\right\},x_1^n\right)\le\mathcal{N}_1\left(\delta\cdot\beta_n,\mathcal{G},x_1^n\right)$$

for an arbitrary function space $\mathcal{G}$ and $\delta>0$ lead to

$$\mathbf{P}\{T_{2,n}>t\}\le 14\cdot\sup_{x_1^n\in supp(X)^n}\mathcal{N}_1\left(\frac{t}{80\cdot\beta_n},\mathcal{F}_n,x_1^n\right)\cdot\exp\left(-\frac{n}{5136\cdot\beta_n^2}\cdot t\right).$$

Since the covering number is decreasing in t, we can conclude for $\varepsilon_n\ge\frac{80}{n}$

$$\begin{aligned}\mathbf{E}(T_{2,n})&\le\mathbf{E}\big(\max\{T_{2,n},0\}\big)=\int_0^\infty\mathbf{P}\big\{\max\{T_{2,n},0\}>t\big\}\,dt=\int_0^\infty\mathbf{P}\{T_{2,n}>t\}\,dt\\&\le\varepsilon_n+\int_{\varepsilon_n}^\infty\mathbf{P}\{T_{2,n}>t\}dt\\&\le\varepsilon_n+14\cdot\sup_{x_1^n\in supp(X)^n}\mathcal{N}_1\left(\frac{1}{n\cdot\beta_n},\mathcal{F}_n,x_1^n\right)\cdot\exp\left(-\frac{n}{5136\cdot\beta_n^2}\cdot\varepsilon_n\right)\\&\quad\cdot\frac{5136\cdot\beta_n^2}{n}.\end{aligned}$$

Choosing

$$\varepsilon_n=\frac{5136\cdot\beta_n^2}{n}\cdot\log\left(14\cdot\sup_{x_1^n\in supp(X)^n}\mathcal{N}_1\left(\frac{1}{n\cdot\beta_n},\mathcal{F}_n,x_1^n\right)\right)$$

(which satisfies the necessary condition $\varepsilon_n\ge\frac{80}{n}$ if the constant c_6 in the definition of β_n is not too small) minimizes the right-hand side and implies

$$\mathbf{E}(T_{2,n})\le\frac{c_{45}\cdot\log(n)^2\cdot\log(\sup_{x_1^n\in supp(X)^n}\mathcal{N}_1(\frac{1}{n\cdot\beta_n},\mathcal{F}_n,x_1^n))}{n}.$$

By bounding $T_{3,n}$ similarly to $T_{1,n}$ we get

$$\mathbf{E}(T_{3,n})\le c_{46}\cdot\frac{\log(n)}{n}$$

for some large enough constant $c_{46}>0$ and hence we get in total

$$\mathbf{E}\left(\sum_{i=1}^{3}T_{i,n}\right)\le\frac{c_{47}\cdot\log(n)^2\cdot(\log(\sup_{x_1^n\in supp(X)^n}\mathcal{N}_1(\frac{1}{n\cdot\beta_n},\mathcal{F}_n,x_1^n))+1)}{n}$$

for some sufficient large constant $c_{47} > 0$.

We finish the proof by bounding $T_{4,n}$. Let A_n be the event, that there exists $i \in \{1, \dots, n\}$ such that $|Y_i| > \beta_n$ and let I_{A_n} be the indicator function of A_n. Then we get

$$\begin{aligned}
\mathbf{E}(T_{4,n}) &\leq 2 \cdot \mathbf{E}\left(\frac{1}{n} \sum_{i=1}^{n} |m_n(X_i) - Y_i|^2 \cdot I_{A_n} \right) \\
&\quad + 2 \cdot \mathbf{E}\left(\frac{1}{n} \sum_{i=1}^{n} |m_n(X_i) - Y_i|^2 \cdot I_{A_n^c} - \frac{1}{n} \sum_{i=1}^{n} |m(X_i) - Y_i|^2 \right) \\
&= 2 \cdot \mathbf{E}\big(|m_n(X_1) - Y_1|^2 \cdot I_{A_n} \big) \\
&\quad + 2 \cdot \mathbf{E}\left(\frac{1}{n} \sum_{i=1}^{n} |m_n(X_i) - Y_i|^2 \cdot I_{A_n^c} - \frac{1}{n} \sum_{i=1}^{n} |m(X_i) - Y_i|^2 \right) \\
&= T_{7,n} + T_{8,n}.
\end{aligned}$$

By the Cauchy-Schwarz inequality we get for $T_{7,n}$

$$\begin{aligned}
\frac{1}{2} \cdot T_{7,n} &\leq \sqrt{\mathbf{E}\big(\big(|m_n(X_1) - Y_1|^2 \big)^2 \big)} \cdot \sqrt{\mathbf{P}(A_n)} \\
&\leq \sqrt{\mathbf{E}\big(\big(2|m_n(X_1)|^2 + 2|Y_1|^2 \big)^2 \big)} \cdot \sqrt{n \cdot \mathbf{P}\{|Y_1| > \beta_n\}} \\
&\leq \sqrt{\mathbf{E}\big(8|m_n(X_1)|^4 + 8|Y_1|^4 \big)} \cdot \sqrt{n \cdot \frac{\mathbf{E}(\exp(c_4 \cdot |Y_1|^2))}{\exp(c_4 \cdot \beta_n^2)}},
\end{aligned}$$

where the last inequality follows as in the proof of inequality (32). Using $x \leq \exp(x)$ for $x \in \mathbb{R}$ we get

$$\begin{aligned}
\mathbf{E}\big(|Y|^4 \big) = \mathbf{E}\big(|Y|^2 \cdot |Y|^2 \big) &\leq \mathbf{E}\left(\frac{2}{c_4} \cdot \exp\left(\frac{c_4}{2} \cdot |Y|^2 \right) \cdot \frac{2}{c_4} \cdot \exp\left(\frac{c_4}{2} \cdot |Y|^2 \right) \right) \\
&= \frac{4}{c_4^2} \cdot \mathbf{E}\big(\exp\big(c_4 \cdot |Y|^2 \big) \big),
\end{aligned}$$

which is finite by assumption (14) of the lemma. Furthermore $\|m_n\|_\infty$ is bounded by β_n and therefore the first factor is bounded by

$$c_{48} \cdot \beta_n^2 = c_{49} \cdot (\log n)^2$$

for some constant $c_{49} > 0$. The second factor is bounded by $1/n$, because by the assumptions of the lemma $\mathbf{E}(\exp(c_4 \cdot |Y_1|^2))$ is bounded by some constant $c_{50} < \infty$ and hence we get

$$\sqrt{n \cdot \frac{\mathbf{E}(\exp(c_4 \cdot |Y_1|^2))}{\exp(c_4 \cdot \beta_n^2)}} \leq \sqrt{n} \cdot \frac{\sqrt{c_{49}}}{\sqrt{\exp(c_4 \cdot \beta_n^2)}} \leq \frac{\sqrt{n} \cdot \sqrt{c_{50}}}{\exp((c_4 \cdot c_6^2 \cdot (\log n)^2)/2)}.$$

Since $\exp(-c \cdot \log(n)^2) = O(n^{-2})$ for any $c > 0$, we get altogether

$$T_{7,n} \leq c_{51} \cdot \frac{(\log n)^2 \sqrt{n}}{n^2} \leq c_{52} \cdot \frac{(\log n)^2}{n}.$$

With the definition of A_n^c and $\tilde{m}_n$ defined as in the assumptions of this lemma we conclude

$$\begin{aligned}
T_{8,n} &\le 2 \cdot \mathbf{E}\left(\frac{1}{n}\sum_{i=1}^{n} |\tilde{m}_n(X_i) - Y_i|^2 \cdot I_{A_n^c} - \frac{1}{n}\sum_{i=1}^{n} |m(X_i) - Y_i|^2 \right) \\
&\le 2 \cdot \mathbf{E}\left(\frac{1}{n}\sum_{i=1}^{n} |\tilde{m}_n(X_i) - Y_i|^2 - \frac{1}{n}\sum_{i=1}^{n} |m(X_i) - Y_i|^2 \right) \\
&\le 2 \cdot \mathbf{E}\left(\min_{l \in \Theta_n} \frac{1}{n}\sum_{i=1}^{n} |g_n(X_i) - Y_i|^2 + pen_n(g_{n,l}) - \frac{1}{n}\sum_{i=1}^{n} |m(X_i) - Y_i|^2 \right) \\
&\le 2 \cdot \mathbf{E}\left(\min_{l \in \Theta_n} \mathbf{E}\left(\frac{1}{n}\sum_{i=1}^{n} |g_{n,l}(X_i) - Y_i|^2 + pen_n(g_{n,l}) \right.\right. \\
&\quad \left.\left. - \frac{1}{n}\sum_{i=1}^{n} |m(X_i) - Y_i|^2 | \mathbf{b}_1^{(1)}, \dots, \mathbf{b}_r^{(1)}, \dots, \mathbf{b}_1^{(I_n)}, \dots, \mathbf{b}_r^{(I_n)} \right) \right) \\
&\le 2 \cdot \mathbf{E}\left(\min_{l \in \Theta_n} \int |g_{n,l}(x) - m(x)|^2 \mathbf{P}_X(dx) + pen_n(g_{n,l}) \right)
\end{aligned}$$

because $|T_\beta z - y| \le |z - y|$ holds for $|y| \le \beta$. Hence

$$\mathbf{E}(T_{4,n}) \le c_{53} \cdot \frac{(\log n)^2}{n} + 2 \cdot \mathbf{E}\left(\min_{l \in \Theta_n} \int |g_{n,l}(x) - m(x)|^2 \mathbf{P}_X(dx) + pen_n(g_{n,l}) \right)$$

holds. Thus the proof of Lemma 8 is complete. □

Acknowledgments

The authors would like to thank the Editor and the two anonymous referees for their very useful suggestions which helped to substantially improve the manuscript.

Funding

Supported by NSERC Grant RGPIN-2020-06793.

References

[1] Adams, R. A. and Fournier, J. J. F. (2003). *Sobolev Spaces*, 2nd ed. Academic Press, Amsterdam, The Netherlands. MR2424078

[2] Allen-Zhu, Z., Li, Y., and Song, Z. (2019). A convergence theory for deep kearning via over-parameterization. In: *Proceedings of the 36th International Conference on Machine Learning (PMLR 2019)*, **97**, pp. 242–252. Long Beach, California.

[3] Anthony, M. and Bartlett, P. L. (1999). *Neural Networks and Learning: Theoretical Foundations*. Cambridge University Press, Cambridge, UK. MR1741038

[4] Arora, S., Cohen, N., Golowich, N., and Hu, W. (2018). A convergence analysis of gradient descent for deep linear neural networks. In: *International Conference on Learning Representations (ICLR 2019)*. New Orleans, Louisiana.

[5] Bagirov, A. M., Clausen, C., and Kohler, M. (2009). Estimation of a regression function by maxima of minima of linear functions. *IEEE Transactions on Information Theory*, **55**, 833–845. MR2597271

[6] Barron, A. R. (1993). Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory* **39**, 930–944. MR1237720

[7] Barron, A. R. (1994). Approximation and estimation bounds for artificial neural networks. *Machine Learning* **14**, 115–133.

[8] Bauer, B., Heimrich, F., Kohler, M., and Krzyżak, A. (2019). On estimation of surrogate models for high-dimensional computer experiments. *Annals of the Institute of Statistical Mathematics* **71**, 107–136. MR3898428

[9] Bauer, B. and Kohler, M. (2019). On deep learning as a remedy for the curse of dimensionality in nonparametric regression. *Annals of Statistics* **47**, 2261–2285. MR3953451

[10] Ben-Ari, E. N. and Steinberg, D. M. (2007). Modeling data from computer experiments: an empirical comparison of kriging with MARS and projection pursuit regression. *Quality Engineering*, **19**, 327–338.

[11] Braun, A., Kohler, M., and Walk, H. (2019). On the rate of convergence of a neural network regression estimate learned by gradient descent. arXiv: 1912.03921.

[12] Choromanska, A., Henaff, M., Mathieu, M., Arous, G. B., and LeCun, Y. (2015). The loss surface of multilayer networks. In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2015, San Diego, CA, USA. Proceeding of Machine Learning Research **38**, pp. 192–204.

[13] Devroye, L., Györfi, L., and Lugosi, G. (1996). *A Probabilistic Theory of Pattern Recognition*. Springer-Verlag, New York, US. MR1383093

[14] Devroye, L. and Wagner, T. J. (1980). Distribution-free consistency results in nonparametric discrimination and regression function estimation. *Annals of Statistics*, **8**, 231–239. MR0560725

[15] Du, S. and Lee, J. (2018). On the power of over-parametrization in neural networks with quadratic activation. In: *Proceedings of the 35th International Conference on Machine Learning (PMLR 2018)*, **80**, pp. 1329–1338. Stockholm, Sweden.

[16] Du, S., Lee, J., Tian, Y., Poczos, B., and Singh, A. (2018). Gradient descent learns one-hidden-layer CNN: don't be afraid of spurious local minima. In: *Proceedings of the 35th International Conference on Machine Learning (PMLR 2018)*, **80**, 1339–1348. Stockholm, Sweden.

[17] Eckle, K. and Schmidt-Hieber, J. (2019). A comparison of deep networks with ReLU activation function and linear spline-type methods. *Neural Net-*

works, **110**, 232–242.
[18] Friedman, J. H. and Stuetzle, W. (1981). Projection pursuit regression. *Journal of the American Statistical Association*, **76**, 817–823. MR0650892
[19] Györfi, L., Kohler, M., Krzyżak, A., and Walk, H. (2002). *A Distribution-Free Theory of Nonparametric Regression*. Springer.
[20] Hall, P. (1989). On projection pursuit regression. *Annals of Statistics*, **17**, 573–588. MR0994251
[21] Härdle, W. and Stoker, T. M. (1989). Investigating smooth multiple regression by the method of average derivatives. *Journal of the American Statistical Association*, **84**, 986–995. MR1134488
[22] Härdle, W., Hall, P., and Ichimura, H. (1993). Optimal smoothing in single-index models. *Annals of Statistics*, **21**, 157–178. MR1212171
[23] Haykin, S. O. (2008). *Neural Networks and Learning Machines*, 3rd ed. Prentice-Hall, New York, US.
[24] Hertz, J., Krogh, A., and Palmer, R. G. (1991). *Introduction to the Theory of Neural Computation*. Addison-Wesley, Redwood City, California, US. MR1096298
[25] Horowitz, J. L. and Mammen, E. (2007). Rate-optimal estimation for a general class of nonparametric regression models with unknown link functions. *Annals of Statistics*, **35**, 2589–2619. MR2382659
[26] Huber, P. J. (1985). Projection pursuit. *Annals of Statistics*, **13**, 435–475. MR0790553
[27] Imaizumi, M. and Fukamizu, K. (2019). Deep neural networks learn non-smooth functions effectively. In: *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS 2019)*. Naha, Okinawa, Japan.
[28] Jiao, Y., Lai, Y., Lu, X., Wang, F., Yang, J. Z., and Yang, Y. (2023). Deep neural networks with relu-sine-exponential activations break curse of dimensionality in approximation on Hölder class. *SIAM Journal on Mathematical Analysis*, **55**, 3635–3649. MR4629853
[29] Jones, L. K. (1987). On a conjecture of Huber concerning the convergence of projection pursuit regression. *Annals of Statistics*, **15**, 880–882. MR0888447
[30] Jones, L. K. (1992). A simple lemma on greedy approximation in Hilbert space and convergence rates for projection pursuit regression and neural network training. *Annals of Statistics*, **20**, 608–613. MR1150368
[31] Kawaguchi, K. (2016). Deep learning without poor local minima. In: *30th Conference on Neural Information Processing Systems (NIPS 2016)*. Barcelona, Spain.
[32] Kawaguchi, K. and Huang, J. (2019). Gradient descent finds global minima for generalizable deep neural networks of practical sizes. arXiv:1908.02419v1.
[33] Kohler, M. and Krzyżak, A. (2017). Nonparametric regression based on hierarchical interaction models. *IEEE Transaction on Information Theory*, **63**, 1620–1630. MR3625984
[34] Kohler, M., Krzyżak, A., and Langer, S. (2022). Estimation of a function of low local dimensionality by deep neural networks. *IEEE Transaction on*

Information Theory, **68**, 4032–4042. MR4433267
[35] Kohler, M. and Langer, S. (2021). On the rate of convergence of fully connected deep neural network regression estimates *Annals of Statistics* **49**, 2231–2249. MR4319248
[36] Kong, E. and Xia, Y. (2007). Variable selection for the single-index model *Biometrika*, **94**, 217–229. MR2367831
[37] Lepski, O and Serdyukova, O. (2014). Adaptive estimation under single-index constraint in a regression model. *Annals of Statistics*, **42**, 1–28. MR3161459
[38] Liang, S., Sun, R., Lee, J., and Srikant, R. (2018). Adding one neuron can eliminate all bad local minima. In: *Proceedings of the 32nd Conference on Neural Information Processing Systems (NIPS 2018)*, pp. 4355–4365. Montreal, Canada.
[39] Lu, J., Shen, Z., Yang, H., and Zhang, S. (2021). Deep network approximation for smooth functions. *SIAM Journal on Mathematical Analysis*, **53**, 5465–5506. MR4319100
[40] Ripley, B. D. (2008). *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge, UK. MR2451352
[41] Scarselli, F. and Tsoi, A. C. (1998). Universal Approximation Using Feedforward Neural Networks: A Survey of Some Existing Methods, and Some New Results. *Neural Networks*, **11**, 15–37.
[42] Schmidhuber, J. (2015). Deep learning in neural networks: an overview. *Neural Networks*, **61**, 85–117.
[43] Schmidt-Hieber, J. (2020). Nonparametric regression using deep neural networks with ReLU activation function. *Annals of Statistics*, **48**, 1875–1897 (with discussion). MR4134774
[44] Stone, C. J. (1982). Optimal global rates of convergence for nonparametric regression. *Annals of Statistics*, **10**, 1040–1053. MR0673642
[45] Stone, C. J. (1985). Additive regression and other nonparametric models. *Annals of Statistics*, **13**, 689–705. MR0790566
[46] Stone, C. J. (1994). The use of polynomial splines and their tensor products in multivariate function estimation. *Annals of Statistics*, **22**, 118–184. MR1272079
[47] Yarotsky, D. (2017). Error bounds for approximations with deep ReLU networks. *Neural Networks*, **94**, 103–114.
[48] Yu, Y. and Ruppert, D. (2002). Penalized spline estimation for partially linear single-index models. *Journal of the American Statistical Association*, **97**, 1042–1054. MR1951258
[49] Zhao, Y. and Atkeson, C. G. (1992). Some approximation properties of projection pursuit learning networks. In: *Advances in Neural Information Processing Systems*, pp. 936–943.