Institut für Numerische Simulation
Rheinische Friedrich-Wilhelms-Universität Bonn
Endenicher Allee 19b • 53115 Bonn • Germany
phone +49 228 73-69828 • fax +49 228 73-69847
www.ins.uni-bonn.de

B. Bohn, M. Griebel, C. Rieger

A representer theorem for deep kernel learning

INS Preprint No. 1714

May 2019

A representer theorem for deep kernel learning

Bastian Bohn[†] Michael Griebel[†‡] Christian Rieger[†]

May 9, 2019

Abstract

In this paper we provide a finite-sample and an infinite-sample representer theorem for the concatenation of (linear combinations of) kernel functions of reproducing kernel Hilbert spaces. These results serve as mathematical foundation for the analysis of machine learning algorithms based on compositions of functions. As a direct consequence in the finite-sample case, the corresponding infinite-dimensional minimization problems can be recast into (nonlinear) finite-dimensional minimization problems, which can be tackled with nonlinear optimization algorithms. Moreover, we show how concatenated machine learning problems can be reformulated as neural networks and how our representer theorem applies to a broad class of state-of-the-art deep learning methods.

1 Introduction

The interpolation or regression of given function values is one of the main tasks in modern data mining and machine learning applications. Due to the famous representer theorem for empirical risk minimization in reproducing kernel Hilbert spaces (RKHS), see e.g. [16, 23, 24], various algorithms based on finite linear combinations of kernel translates have gained much popularity in the last decade, like, for example, support vector machines (SVMs) and Tikhonov-regularized least-squares in RKHS. In general, these methods work very well if the underlying problem fits the chosen reproducing kernel space H, e.g. if the given input values stem from a function $g \in H$. However, if H contains for instance only smooth functions but g has a kink or a jump, the interpolant or regressor, respectively, in H might not represent a good approximation to the true function g anymore. Then, if it is not known how to choose an appropriate kernel K of H a priorily, one usually relies on so-called multiple kernel learning (MKL) algorithms, which try to determine the optimal kernel adaptively, see e.g. [2]. But while most of these methods allow to learn a suitable kernel by simply constructing a linear or convex combination of a given set of input kernels, they still do not achieve considerably better results than standard a-priori kernel choices for many applications, see [11].
In recent years, promising new variants of kernel learning methods, namely deep kernel learning and multi-layer-MKL (MLMKL) algorithms have been developed. They have proven to be very successful in regression and classification tasks. Here, motivated by multi-layer feed-forward neural networks, a kernel function is concatenated with one or more nonlinear functions in order to achieve a highly flexible new kernel function, see e.g. [5, 7, 21, 25, 27, 28]. The main idea behind this approach is to combine the flexibility of deep neural networks, in which the feature detection in the data set is done completely automatically, with the approximation power of kernel methods, in which a feature map is determined by the chosen kernel. This way, the

[†]Institute for Numerical Simulation, University of Bonn, Wegelerstr. 6, 53115 Bonn, Germany.
[‡]Fraunhofer Center for Machine Learning, Fraunhofer Institute for Algorithms and Scientific Computing SCAI, Schloss Birlinghoven, 53754 Sankt Augustin, Germany.
The authors want to thank the anonymous referees for their suggestions and remarks and especially for pointing out the relation to [9]. The authors were partially supported by the Sonderforschungsbereich 1060 *The Mathematics of Emergent Effects* funded by the Deutsche Forschungsgemeinschaft.

neural network architecture learns the optimal kernel that best represents important features of the data for the task at hand. While first steps towards creating a mathematical framework to analyze deep neural networks - especially for image classification tasks - have been made in e.g. [17, 18, 20], deep approximation theory for kernel based approaches is still missing at large. Moreover, the underlying nonlinear minimization problem is usually tackled by simple gradient descent and heuristic backpropagation algorithms without a thorough theoretical analysis of its properties. An initial cornerstone for the analysis of chained kernel approximations has been provided by [9], where two-layer kernel networks were considered and their relation to MKL was established. However, an analysis of deeper kernel networks and their connection to MLMKL has not been considered so far.

In this paper, we consider the problem of optimal concatenated approximation in reproducing kernel Hilbert spaces, which will directly lead to a variant of multi-layer kernel learning problems and will extend the results achieved in [9]. For this class, we will prove a representer theorem, which allows us to reduce the nonlinear, potentially infinite-dimensional optimization problem to a finite-dimensional one. Consequently, standard nonlinear optimization techniques can be used to tackle this problem. At least to our knowledge, this is the first derivation of a representer theorem for concatenated function approximation in the literature. It is also valid for certain types of hidden layer neural networks and deep SVMs.

The remainder of this paper is organized as follows: In Section 2, we briefly review the interpolation and the regression problem in an (possibly infinite-dimensional) RKHS and discuss how the classical representer theorem allows to recast these problems into finite-dimensional linear equation systems. In Section 3, we introduce the optimal *concatenated* approximation problem for arbitrary loss functions and regularizers. We derive a representer theorem for this problem in the multi-layer case and discuss its relation to deep learning and multi-layer multiple kernel learning methods. Furthermore, we exemplarily derive algorithms for interpolation and least-squares regression in the two-layer case from it. The latter will be a natural generalization of the RLS2 method developed in [9], which only deals with a linear outer kernel. Section 4 illustrates the application of our concatenated interpolation and regression algorithms to two simple examples and serves as a proof of concept. Finally, we conclude with a summary and an outlook in Section 5.

2 Interpolation and regression in reproducing kernel Hilbert spaces

In this section we shortly review interpolation and least-squares regression problems, respectively, in an RKHS. To this end, we consider the standard representer theorem and show how it helps to find an interpolant/regressor.

2.1 Interpolation

Let $\Omega \subset \mathbb{R}^d$ be an open domain and let the pairwise disjoint points $X := \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_N\} \subset \Omega$ and the values $Y := \{y_1, \dots, y_N\} \subset \mathbb{R}$ be given. Let furthermore $H := H(\Omega, \mathbb{R})$ be a reproducing kernel Hilbert space of real-valued functions on Ω. The minimal norm interpolant is

$$f^*_{X,Y} := \operatorname*{arg\,min}_{f \in H} \|f\|_H \quad \text{such that} \quad f(\boldsymbol{x}_i) = y_i \quad \forall\, i = 1, \dots, N. \tag{1}$$

The classical representer theorem, see e.g. [23, 24] for scalar-valued functions and [19] for vector-valued functions, now states that $f^*_{X,Y}$ can be written as a *finite* linear combination of kernel evaluations in the data, namely

$$f^*_{X,Y}(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i^* K(\boldsymbol{x}_i, \boldsymbol{x}), \tag{2}$$

where $K : \Omega \times \Omega \to \mathbb{R}$ denotes the reproducing kernel of H and $\alpha_i^* \in \mathbb{R}, i = 1, \dots, N$, are the corresponding coefficients. For details on RKHS, see [1]. Therefore, the solution to the possibly infinite-dimensional

optimization problem (1) resides in the N-dimensional span of the functions $K(\boldsymbol{x}_i, \cdot), i = 1, \ldots, N$. To compute the coefficients, we simply have to solve the system

$$\boldsymbol{M}_{X,X} \boldsymbol{\alpha}^* = \boldsymbol{y} \tag{3}$$

of linear equations with

$$\boldsymbol{M}_{X,X} := \begin{pmatrix} K(\boldsymbol{x}_1, \boldsymbol{x}_1) & \ldots & K(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ \vdots & \ddots & \vdots \\ K(\boldsymbol{x}_N, \boldsymbol{x}_1) & \ldots & K(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{pmatrix}, \quad \boldsymbol{\alpha}^* := \begin{pmatrix} \alpha_1^* \\ \vdots \\ \alpha_N^* \end{pmatrix} \quad \text{and } \boldsymbol{y} := \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}. \tag{4}$$

Note that this $N \times N$ system admits a unique solution if the kernel K is strictly positive definite. For example, for Sobolev kernels it can be shown that the condition number of the system matrix $\boldsymbol{M}_{X,X}$ only grows moderately with the size N provided that the data points are quasi-uniformly distributed, see [8]. Moreover, for infinitely smooth kernel functions (e.g. Gaussian kernels or multiquadrics) it can be necessary to perform an appropriate basis change before solving the above equation system, see e.g. [26].

2.2 Least-squares regression

In real-world applications, the values $y_i, i = 1, \ldots, N$ are usually not exactly given, but are perturbed by some noise term. Therefore, a direct interpolation might no longer be appropriate. In this case, one considers the corresponding regularized least-squares regression problem

$$f_{X,Y}^{\lambda} := \operatorname*{arg\,min}_{f \in H} \lambda \|f\|_H^2 + \sum_{j=1}^{N} |f(\boldsymbol{x}_i) - y_i|^2, \tag{5}$$

where the side condition in (1) is substituted by a penalty term. Here, the Lagrange multiplier λ weights the importance of the norm minimization against the function evaluation error. Again, the representer theorem [19, 23, 24] tells us that $f_{X,Y}^{\lambda}$ is of the form (2), i.e.

$$f_{X,Y}^{\lambda}(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i^{\lambda} K(\boldsymbol{x}_i, \boldsymbol{x}).$$

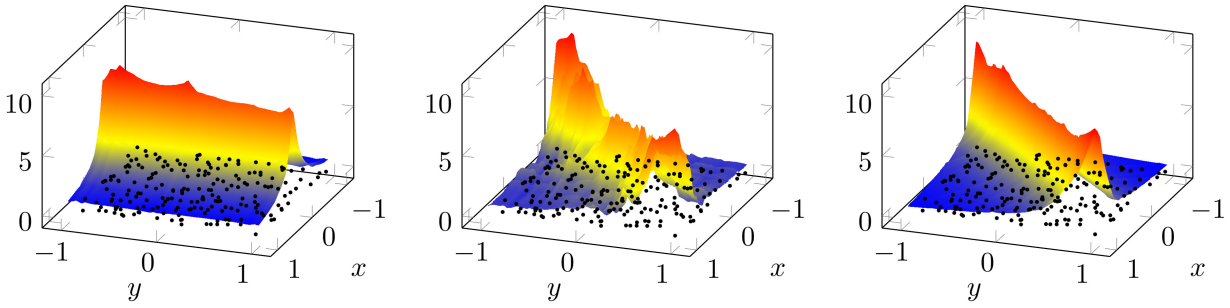
This time the coefficients $\alpha_i^{\lambda}, i = 1, \ldots, N$, are determined by

$$(\boldsymbol{M}_{X,X} + \lambda \boldsymbol{I})\, \boldsymbol{\alpha}^{\lambda} = \boldsymbol{y}, \tag{6}$$

where $\boldsymbol{I}$ denotes the $N \times N$ identity matrix. The size of the Lagrange parameter $\lambda > 0$ now also influences the condition number of the system matrix, i.e. the larger λ is, the smaller the condition number becomes.

3 Interpolation and regression with compositions of reproducing kernel Hilbert spaces

As already mentioned in the introduction, the standard interpolation and regression algorithms in RKHS work well if the samples y_i are (perturbed) evaluations of a function $g \in H$, where the reproducing kernel space H is known in the first place. However, if the appropriate RKHS H is unknown, it is advisable to resort to multiple kernel learning methods or multi-layer multiple kernel learning methods.
We now explain this aspect in more detail and, to this end, motivate a first two-dimensional, two-layer approach with an example: Let the kernel K of H be a tensor-product of two univariate Matérn Sobolev



(a) $y_i = g_1(\boldsymbol{x}_i), f \in H$ (b) $y_i = g_2(\boldsymbol{x}_i), f \in H$ (c) $y_i = g_2(\boldsymbol{x}_i), f \in \{h \circ R \mid h \in H\}$

Figure 1: Solutions to (1) in the two-variate tensor-product Matérn-kernel Sobolev space H of order one, see also [10], with 200 uniform samples $\boldsymbol{x}_i, i = 1, \ldots, 200$ (marked in black), shown in the domain $[-1, 1]^2$. (a) depicts the solution $f \in H$ for values y_i sampled from g_1, whereas (b) shows the optimal solution for y_i sampled from g_2. (c) presents the best interpolant of type $f \circ R$, where $f \in H$ and R is a rotation by 45° for y_i sampled from g_2. For reasons of comparability, we restricted our representation to $[-1, 1]^2$ here, although some data points were mapped outside of this domain by applying the rotation R and the kernel was defined on the whole $\mathbb{R}^2$.

kernels of order one on $\mathbb{R}$, see Section 4 for a definition of this kernel. The corresponding function space H is often also called Sobolev space of "mixed smoothness" of order one and it is of special importance for e.g. sparse grid discretizations, see [4], and quasi Monte Carlo quadrature, see [14]. Now, let us consider the continuous function $g_1(x, y) := (0.1 + |x|)^{-1}$, which has a kink that is perpendicular to the x-axis. It can easily be shown that $g_1 \in H$ and, therefore, the interpolant of g_1 by a function from H resembles a good approximation to g_1, see Figure 1(a). If we now look at $g_2(x, y) := (0.1 + |x - y|)^{-1}$, which has a kink along the diagonal with $x = y$, then $g_2 \notin H$. Therefore, the interpolant of g_2 by a function in H is a rather bad approximation to g_2. This can be seen in Figure 1(b). However, if we let R^{-1} be a rotation by 45°, then $g_2 \circ R^{-1} \in H$ would have an axis-aligned kink like g_1. To use this fact when interpolating g_2, we can simply look for the best interpolant in $\{f \circ R \mid f \in H\}$ in (1) instead of $f \in H$. This example is illustrated in Figure 1(c). As we can see, the interpolant in Figure 1(c) is a much better representative for g_2 than the one in Figure 1(b). This example illustrates that, already in the very simple case of employing a concatenation with a rotation, a two-layer approach can be a good choice to overcome the restrictions of a standard kernel learning algorithm. Let us remark that already this motivating example exhibits a fundamentally different setting from the one considered in [9] because of the nonlinearity of the outer kernel. While the RLS2 algorithm introduced there can be interpreted as an MKL variant, where a convex combination of given kernel functions is computed, we are looking for an inner function, which transforms the domain in such a way that it is optimal for the (possibly nonlinear) outer kernel.
Now, instead of just considering one layer of simple rotations as in the above example, we allow for a fully flexible multi-layer kernel learning approach, where we employ arbitrary functions from reproducing kernel Hilbert spaces in each layer. This approach can successfully deal with a much broader class of interpolation and regression problems, see also [21,28]. To this end, we consider *concatenated* machine learning problems. We introduce a new representer theorem for the case of multiple concatenations of functions from RKHS, which allows us to derive the related, finite-dimensional, nonlinear optimization problem.

3.1 A representer theorem for concatenated kernel learning

In this section, we show how a *concatenated representer theorem* can be derived for a very general class of problem types and an arbitrary number $L \in \mathbb{N}$ of concatenations. For more details on vector-valued

reproducing kernel Hilbert spaces, we refer the reader to [19]. For a two-layer variant of this theorem, we refer to [9].

Theorem 1. *Let $\mathcal{H}_1, \ldots, \mathcal{H}_L$ be reproducing kernel Hilbert spaces of functions with finite-dimensional domains D_l and ranges $R_l \subseteq \mathbb{R}^{d_l}$ with $d_l \in \mathbb{N}$ for $l = 1, \ldots, L$ such that $R_l \subseteq D_{l-1}$ for $l = 2, \ldots, L$, $D_L = \Omega$ and $R_1 \subseteq \mathbb{R}$. Let furthermore $\mathcal{L} : \mathbb{R}^2 \to [0, \infty]$ be an arbitrary loss function and let $\Theta_1, \ldots, \Theta_L : [0, \infty) \to [0, \infty)$ be strictly monotonically increasing functions. Then, a set of minimizers $(f_l)_{l=1}^L$ with $f_l \in \mathcal{H}_l$ of*

$$J(f_1, \ldots, f_L) := \sum_{i=1}^{N} \mathcal{L}\left(y_i, f_1 \circ \ldots \circ f_L(\boldsymbol{x}_i)\right) + \sum_{l=1}^{L} \Theta_l\left(\|f_l\|_{\mathcal{H}_l}^2\right) \tag{7}$$

fulfills $f_l \in \tilde{V}_l \subset \mathcal{H}_l$ for all $l = 1, \ldots, L$ with

$$\tilde{V}_l = \operatorname{span}\left\{K_l\left(f_{l+1} \circ \ldots \circ f_L\left(\boldsymbol{x}_i\right), \cdot\right) \boldsymbol{e}_{k_l} \mid i = 1, \ldots, N \text{ and } k_l = 1, \ldots, d_l\right\},$$

where K_l denotes the reproducing kernel of $\mathcal{H}_l$ and $\boldsymbol{e}_{k_l} \in \mathbb{R}^{d_l}$ is the k_l-th unit vector.

Proof. We denote by $\Pi_{\tilde{V}_l}$ and $\Pi_{\tilde{V}_l}^{\perp}$ the projector onto $\tilde{V}_l$ and its orthogonal complement in $\mathcal{H}_l$, respectively, for $l = 1, \ldots, L$. First, we note that

$$\begin{aligned}
f_l \circ f_{l+1} \circ \ldots \circ f_L(\boldsymbol{x}_i) &= \sum_{k=1}^{d_l} \left(\Pi_{\tilde{V}_l}\left(f_l\right) + \Pi_{\tilde{V}_l^{\perp}}\left(f_l\right), K_l\left(f_{l+1} \circ \ldots \circ f_L(\boldsymbol{x}_i), \cdot\right) \boldsymbol{e}_k\right)_{\mathcal{H}_l} \cdot \boldsymbol{e}_k \\
&= \sum_{k=1}^{d_l} \left(\Pi_{\tilde{V}_l}\left(f_l\right), K_l\left(f_{l+1} \circ \ldots \circ f_L(\boldsymbol{x}_i), \cdot\right) \boldsymbol{e}_k\right)_{\mathcal{H}_l} \cdot \boldsymbol{e}_k \\
&= \sum_{k=1}^{d_l} \left(\boldsymbol{e}_k^T \Pi_{\tilde{V}_l}\left(f_l\right)\left(f_{l+1} \circ \ldots \circ f_L(\boldsymbol{x}_i)\right)\right) \cdot \boldsymbol{e}_k \\
&= \Pi_{\tilde{V}_l}\left(f_l\right)\left(f_{l+1} \circ \ldots \circ f_L(\boldsymbol{x}_i)\right)
\end{aligned}$$

for all $i = 1, \ldots, N$ and $l = 1, \ldots, L$. Since this holds for each function in the chain, we can iterate this process to obtain

$$f_l \circ f_{l+1} \circ \ldots \circ f_L(\boldsymbol{x}_i) = \Pi_{\tilde{V}_l}\left(f_l\right) \circ \Pi_{\tilde{V}_{l+1}}\left(f_{l+1}\right) \circ \ldots \circ \Pi_{\tilde{V}_L}\left(f_L\right)\left(\boldsymbol{x}_i\right) \tag{8}$$

for each $l = 1, \ldots, L$. Therefore, we have

$$\begin{aligned}
J(f_1, \ldots, f_L) = &\sum_{i=1}^{N} \mathcal{L}\left(y_i, \Pi_{\tilde{V}_1}(f_1) \circ \ldots \circ \Pi_{\tilde{V}_L}(f_L)(\boldsymbol{x}_i)\right) \\
&+ \sum_{l=1}^{L} \Theta_l\left(\|\Pi_{\tilde{V}_l}(f_l)\|_{\mathcal{H}_l}^2 + \|\Pi_{\tilde{V}_l^{\perp}}(f_l)\|_{\mathcal{H}_l}^2\right) \geq J(\Pi_{\tilde{V}_1}(f_1), \ldots, \Pi_{\tilde{V}_L}(f_L))
\end{aligned}$$

and equality only holds if $f_l \in \tilde{V}_l$ for each $l = 1, \ldots, L$ because of the strict monotonicity of each Θ_l. This completes the proof. □

Note that theorem 1 also holds for

$$J(f_1, \ldots, f_L) := \mathcal{L}\left(y_1, f_1 \circ \ldots \circ f_L(\boldsymbol{x}_1), \ldots, y_N, f_1 \circ \ldots f_L(\boldsymbol{x}_N)\right) + \sum_{l=1}^{L} \Theta_l\left(\|f_l\|_{\mathcal{H}_l}^2\right)$$

with arbitrary loss $\mathcal{L} : \left(\mathbb{R}^2\right)^N \to [0, \infty]$. However, the version we proved above is more consistent with the remainder of this paper. Furthermore, because of (8), we could also state an even more general version of Theorem 1 where the loss function $\mathcal{L}$ not only depends on the point evaluations $f_1 \circ \ldots \circ f_L(\boldsymbol{x}_i)$ for $i = 1, \ldots, N$, but also on the intermediate values $f_l \circ \ldots \circ f_L(\boldsymbol{x}_i)$ for any $l = 2, \ldots, L$. However, for the sake of readability, we proceed with (7). Theorem 1 now states that

$$(f_1, \ldots, f_L) = \operatorname*{arg\,min}_{\substack{\bar{f}_l \in \mathcal{H}_l \\ l=1,\ldots,L}} J(\bar{f}_1, \ldots, \bar{f}_L) = \operatorname*{arg\,min}_{\substack{\bar{f}_l \in \tilde{V}_l \\ l=1,\ldots,L}} J(\bar{f}_1, \ldots, \bar{f}_L) \tag{9}$$

with J from (7). This means that the (possibly) infinite-dimensional optimization problem

$$\operatorname*{arg\,min}_{\substack{\bar{f}_l \in \mathcal{H}_l \\ l=1,\ldots,L}} J(\bar{f}_1, \ldots, \bar{f}_L)$$

can be recast into the finite-dimensional optimization problem

$$\operatorname*{arg\,min}_{\substack{\bar{f}_l \in \tilde{V}_l \\ l=1,\ldots,L}} J(\bar{f}_1, \ldots, \bar{f}_L).$$

In this way, our representer theorem is a direct extension of the classical representer theorem, see Section 2 and [23], to concatenated functions. We obtain that the solution to (9) is given by a linear combination of at most N basis functions in each layer. Therefore, the overall number of degrees of freedom in the underlying optimization problem (9) is given by

$$\#\text{dof} = \sum_{l=1}^{L} \dim\left(\tilde{V}_l\right) = \sum_{l=1}^{L} N \cdot d_l = N \cdot \left(1 + \sum_{l=2}^{L} d_l\right).$$

According to Theorem 1, we can write f_1 as

$$f_1(\cdot) = \sum_{j=1}^{N} \alpha_j K_1\left(f_2 \circ \ldots \circ f_L(\boldsymbol{x}_j), \cdot\right)$$

for some coefficients $\alpha_j \in \mathbb{R}$. Therefore, the concatenated function $h(\cdot) = f_1 \circ \ldots \circ f_L(\cdot)$, which we are interested in, can be expressed as

$$h(\cdot) = \sum_{j=1}^{N} \alpha_j \mathcal{K}^L(\boldsymbol{x}_j, \cdot)$$

with the deep kernel

$$\mathcal{K}^L(\boldsymbol{x}, \boldsymbol{y}) = K_1\left(f_2 \circ \ldots \circ f_L(\boldsymbol{x}), f_2 \circ \ldots \circ f_L(\boldsymbol{y})\right). \tag{10}$$

Due to the definition of $\tilde{V}_l$ for $l = 1, \ldots, L$, the corresponding f_l is defined recursively. In general, it is thus not possible to simply write down a closed formula for $\mathcal{K}^L$ for arbitrary L. To illustrate the structure of the kernel $\mathcal{K}^L$, we therefore consider a two-layer example with $L = 2$ in the following. In this case, we obtain $\tilde{V}_2 = \operatorname{span}\{K_2(\boldsymbol{x}_i, \cdot)\boldsymbol{e}_{k_2} \mid i = 1, \ldots, N \text{ and } k_2 = 1, \ldots, d_2\}$. From Theorem 1, we know that

$$f_2(\cdot) = \sum_{i=1}^{N} \sum_{k_2=1}^{d_2} c_{i,k_2} K_2(\boldsymbol{x}_i, \cdot)\boldsymbol{e}_{k_2}$$

for certain coefficients $c_{i,k_2} \in \mathbb{R}$. Furthermore, we have that $f_1 \in \tilde{V}_1 = \operatorname{span}\{K_1(f_2(\boldsymbol{x}_i), \cdot) \mid i = 1, \ldots, N\}$ and thus

$$f_1(\cdot) = \sum_{j=1}^{N} \alpha_j K_1\left(\sum_{i=1}^{N} \sum_{k_2=1}^{d_2} c_{i,k_2} K_2(\boldsymbol{x}_i, \boldsymbol{x}_j)\boldsymbol{e}_{k_2}, \cdot\right).$$

The concatenated function is then given by $h(\cdot) := f_1 \circ f_2(\cdot) = \sum_{j=1}^{N} \alpha_j \mathcal{K}^2(\boldsymbol{x}_j, \cdot)$ with the composition kernel

$$\mathcal{K}^2(\boldsymbol{x}, \boldsymbol{y}) = K_1 \left(\sum_{i=1}^{N} \sum_{k_2=1}^{d_2} c_{i,k_2} K_2(\boldsymbol{x}_i, \boldsymbol{x}) \boldsymbol{e}_{k_2}, \sum_{i=1}^{N} \sum_{k_2=1}^{d_2} c_{i,k_2} K_2(\boldsymbol{x}_i, \boldsymbol{y}) \boldsymbol{e}_{k_2} \right). \tag{11}$$

Therefore, instead of considering the infinite-dimensional optimization problem of finding $f_1 \in \mathcal{H}_1$ and $f_2 \in \mathcal{H}_2$ that minimize

$$J(f_1, f_2) = \sum_{i=1}^{N} \mathcal{L}\left(y_i, f_1(f_2(\boldsymbol{x}_i))\right) + \Theta_1\left(\|f_1\|_{\mathcal{H}_1}^2\right) + \Theta_2\left(\|f_2\|_{\mathcal{H}_2}^2\right),$$

we can restrict ourselves to finding the $N + N \cdot d_2$ coefficients α_j, c_{i,k_2} for $i, j = 1, \ldots, N$ and $k_2 = 1, \ldots, d_2$. Note at this point that the problem of finding these coefficients is highly nonlinear and becomes more complicated for a larger number of layers L. While the corresponding problem of optimizing the outermost coefficients, i.e. α_j for $j = 1, \ldots, N$ in our example, is still convex if the loss $\mathcal{L}$ and the penalty terms Θ_1, Θ_2 are convex, the optimization of the inner coefficients, i.e. c_{i,k_2} for $i = 1, \ldots, N$ and $k_2 = 1, \ldots, d_2$, is usually not convex anymore and can have many local minima. Here, finding a global minimum is an issue because standard (iterative) optimization methods strongly depend on the chosen initial value and usually just deliver some local minimum.
If the optimization functional J is smooth, one can rely on a Newton-type minimizer such as BFGS to solve the underlying optimization problem. However, if one deals with nonsmooth loss functionals or penalty terms, one should resort to specifically designed stochastic gradient algorithms which fit the problem at hand, see e.g. [22].
It remains to note that our representer theorem covers much more than just interpolation or least-squares regression algorithms. In the same fashion as the standard representer theorem in [23], it can directly be applied to more involved settings such as regression with a concatenation of support vector machines for instance. To this end, just choose $\mathcal{L}$ to be the ε-insensitive loss function and $\Theta_1(x) = \ldots = \Theta_L(x) = x$. Furthermore, the choice of the additive penalties $\Theta_1, \ldots, \Theta_L$ in (7) is rather arbitrary and one could think of more complex interactions between the penalties for each function $f_l, l = 1, \ldots, L$, as long as the arguments in the proof of Theorem 1 remain valid.

3.2 An infinite-sample representer theorem for concatenated kernel learning

After deriving the representer theorem 1 for the case of multi-layer kernel approximations, we now extend our results to the case of infinitely many samples. This has to be understood in analogy to the results in chapter 5 of [24], where such an infinite-sample representer theorem is provided for the single-layer case. Although such a result can usually not directly be applied to a practical problem unless the distribution of the data points is known, it can serve as a cornerstone for the analysis of robustness with respect to a measure change and can lead to a-priori convergence results, see [24]. We will restrict the loss function to be an L-times differentiable Nemitski loss for the following theorem. For a definition, we refer to [24] or our appendix, where we define an even more general type of *Nemitski vector loss*. Note that, when we refer to convexity or differentiability of Nemitski losses or reproducing kernels, this should always be understood with respect to the second argument, i.e. $\mathrm{d}K(\boldsymbol{x}, \boldsymbol{z})$ should be understood as $\frac{\partial}{\partial \boldsymbol{z}} K(\boldsymbol{x}, \boldsymbol{z})$. In the following, we denote by $\mathcal{B}(X, Y)$ the space of bounded linear operators from X to Y, endowed with the standard operator norm.

Theorem 2. *Let $\mathcal{H}_1, \ldots, \mathcal{H}_L$ and the domains and ranges of their elements be as in theorem 1 and let $\lambda_1, \ldots, \lambda_L > 0$. Let, furthermore, the kernel K_l of $\mathcal{H}_l$ fulfill $K_l \in C^1(D_l \times D_l)$ together with*

$$\sup_{\boldsymbol{x} \in D_l} \|K_l(\boldsymbol{x}, \boldsymbol{x})\|_2 \leq c_l \quad \text{and} \quad \sup_{\boldsymbol{x}, \boldsymbol{z} \in D_l} \|\mathrm{d}K_l(\boldsymbol{x}, \boldsymbol{z})\|_{\mathcal{B}\left(D_l, \mathbb{R}^{d_l \times d_l}\right)} \leq c_l \tag{12}$$

for some $c_l < \infty$ and all $l = 1, \dots, L$. Let $\mathbb{P}$ be a distribution on $\Omega \times R_1$ and let $\mathcal{L} : R_1 \times \mathbb{R} \to [0, \infty)$ be a convex, $\mathbb{P}$-integrable and 1-times differentiable (w.r.t. the second variable) Nemitski loss such that the absolute value of the derivative is also a $\mathbb{P}$-integrable Nemitski loss, which fulfills

$$\left|\mathcal{L}^{(k)}(y, z)\right| \leq b_k(y) + h_k(|z|) \quad \text{for all } (y, z) \in R_1 \times \mathbb{R}$$

for some $L_{1,\mathbb{P}_{R_1}}$-integrable[1] $b_k : R_1 \to [0, \infty)$ and some increasing $h_k : [0, \infty) \to [0, \infty)$ for $k = 0, 1$. Then, if we assume that a set of minimizers $(f_l)_{l=1}^L$ with $f_l \in \mathcal{H}_l$ of

$$J(f_1, \dots, f_L) := \int_{\Omega \times R_1} \mathcal{L}\left(y, f_1 \circ \dots \circ f_L(\boldsymbol{x})\right) \, \mathrm{d}\mathbb{P}(\boldsymbol{x}, y) + \sum_{l=1}^{L} \lambda_l \|f_l\|_{\mathcal{H}_l}^2 \tag{13}$$

exists, it fulfills the Bochner-type integral equation

$$f_l(\cdot) = -\frac{1}{2\lambda_i} \int_{\Omega \times R_1} K_l\left(\cdot, f_{l+1} \circ \dots \circ f_L(\boldsymbol{x})\right) A_{f_l, f_{l+1}, \dots, f_L}(\boldsymbol{x}, y) \, \mathrm{d}\mathbb{P}(\boldsymbol{x}, y) \tag{14}$$

for some $A_{f_l, f_{l+1}, \dots, f_L} \in L_{1,\mathbb{P}}(\Omega \times R_1; R_l)$ for all $l = 1, \dots, L$.

Proof. The proof works layer-wise and it is an extension of the proof of theorem 5.8 of [24] to the multi-layer case and to Nemitski vector loss functions, see also definition 5. Let $g_i \in \mathcal{H}_i$ be arbitrary for all $i = 1, \dots, L$. Let $G_1 : \Omega \times R_1 \to R_2 \times R_1$ be defined by $G_1(\boldsymbol{x}, y) = (g_2 \circ \dots \circ g_L(\boldsymbol{x}), y)$. Obviously, G_1 is a measurable map and we can define the pushforward $G_{1,\star}(\mathbb{P})$ of $\mathbb{P}$ onto $R_2 \times R_1$. With this we obtain

$$\int_{\Omega \times R_1} \mathcal{L}\left(y, g_1 \circ \dots \circ g_L(\boldsymbol{x})\right) \, \mathrm{d}\mathbb{P}(\boldsymbol{x}, y) = \int_{R_2 \times R_1} \mathcal{L}\left(y, g_1(\boldsymbol{\xi})\right) \, \mathrm{d}G_{1,\star}(\mathbb{P})(\boldsymbol{\xi}, y).$$

Now, with the functional $J_{g_2, \dots, g_L} : \mathcal{H}_1 \to [0, \infty)$ defined by

$$J_{g_2, \dots, g_L}(g_1) := \int_{R_2 \times R_1} \mathcal{L}\left(y, g_1(\boldsymbol{\xi})\right) \, \mathrm{d}G_{1,\star}(\mathbb{P})(\boldsymbol{\xi}, y) + \lambda_1 \|g_1\|_{\mathcal{H}_1}^2,$$

we can reformulate the minimization problem as

$$\min_{g_1 \in \mathcal{H}_1, \dots, g_L \in \mathcal{H}_L} J(g_1, \dots, g_L) = \min_{g_2 \in \mathcal{H}_2, \dots, g_L \in \mathcal{H}_L} \left(\min_{g_1 \in \mathcal{H}_1} J_{g_2, \dots, g_L}(g_1) \right) + \sum_{l=2}^{L} \lambda_l \|g_l\|_{\mathcal{H}_l}^2.$$

Since G_1 leaves the second argument unchanged, it directly follows from the $\mathbb{P}$-integrability that $\mathcal{L}$ is also a $G_{1,\star}(\mathbb{P})$-integrable Nemitski loss. Therefore, the application of the infinite-sample representer theorem 5.8 in [24] states that the minimizer $g_1^\star$ of $J_{g_2, \dots, g_L}$ can be written as

$$\begin{aligned} g_1^\star(\cdot) &= -\frac{1}{2\lambda_1} \int_{R_2 \times R_1} \mathcal{L}^{(1)}\left(y, g_1^\star(\boldsymbol{\xi})\right) K_1(\cdot, \boldsymbol{\xi}) \, \mathrm{d}G_{1,\star}(\mathbb{P})(\boldsymbol{\xi}, y) \\ &= -\frac{1}{2\lambda_1} \int_{\Omega \times R_1} \mathcal{L}^{(1)}\left(y, g_1^\star \circ g_2 \circ \dots \circ g_L(\boldsymbol{x})\right) K_1(\cdot, g_2 \circ \dots \circ g_L(\boldsymbol{x})) \, \mathrm{d}\mathbb{P}(\boldsymbol{x}, y), \end{aligned}$$

where $\mathcal{L}^{(1)}$ denotes the first derivative of $\mathcal{L}$ w.r.t. the second argument. For the choice $g_i = f_i$ for $i = 2, \dots, L$, we obtain the minimizer $f_1 = g_1^\star$. Note that f_1 is continuous and $\|f_1\|_\infty := \sup_{\boldsymbol{x} \in D_1} |f_1(\boldsymbol{x})| < \infty$ since $\mathcal{H}_1 \hookrightarrow C(D_1)$ follows directly by (12). Therefore, (14) is true for $l = 1$ since

$$\begin{aligned} |A_{f_1, \dots, f_L}(\cdot)| := \left|\mathcal{L}^{(1)}\left(y, f_1 \circ f_2 \circ \dots \circ f_L(\cdot)\right)\right| &\leq b_1(y) + h_1\left(|f_1 \circ f_2 \circ \dots \circ f_L(\cdot)|\right) \\ &\leq b_1(y) + h_1\left(\|f_1\|_\infty\right) \end{aligned}$$

[1] Here, $\mathbb{P}_{R_1}$ denotes the marginal distribution of $\mathbb{P}$ w.r.t. the second variable.

is in $L_{1,\mathbb{P}}$ since $b_1 \in L_{1,\mathbb{P}_{R_1}}(R_1)$.
To tackle the next layer, we define $\tilde{\mathcal{L}} : R_1 \times R_2 \to [0,\infty)$ by

$$\tilde{\mathcal{L}}(y, \boldsymbol{z}) := \mathcal{L}(y, f_1(\boldsymbol{z})).$$

We proceed by showing that $\tilde{\mathcal{L}}$ is a $\mathbb{P}$-integrable and 1-times differentiable Nemitski vector loss. Then we show that we can use analogous techniques as in [24] - but for vector-valued functions - to ensure the representation (14) for $l = 2$. These arguments can then be iterated until we reach the innermost layer and the proof is completed. Since the details are quite technical, we outsourced them into appendix 5. □

Theorem 2 states that the solution f_l in the l-th layer of (13) is an element of the range of the integral operator defined by the kernel $K_l\left(\cdot, f_{l+1} \circ \ldots \circ f_L(\cdot)\right) : D_l \times \Omega \to \mathbb{R}^{d_l \times d_l}$. Note that the statement of theorem 1 can be derived by choosing a sum of finitely many Dirac measures $\delta_{\boldsymbol{x}_i, y_i}$ as $\mathbb{P}$ in theorem 2. In this special case, the result boils down to f_l being in the span of the kernel evaluations in the data points. Note furthermore that - in contrast to the finite sample case - f_l is defined as a convolution with the asymmetric kernel in (14). This can be interpreted as a smoothing step for many kernel choices. In this sense, we can expect the solutions f_l of (13) to employ a higher degree of smoothness than in the case of (7), where the solutions are only finite linear combinations of kernels. However, this of course comes at the cost of the regularity condition on the kernels in the requirements of theorem 2.

3.3 Relation to neural networks and deep learning

We now come back to the finite sample case in this section and discuss the relation of our representer theorem 1 to two of the most common approaches in deep learning with kernels, namely multi-layer multiple kernel learning (MLMKL) and deep kernel networks (DKN), see e.g. [5, 7, 21, 25, 27, 28]. For reasons of simplicity, we restrict ourselves to the two-layer case $L = 2$ here.

3.3.1 Relation to hidden layer neural networks

Let us first illustrate how our approach can be encoded as a hidden layer feed-forward neural network. The idea behind artificial neural networks is the same as for multi-layer kernel learning, namely using concatenations of functions to compute good approximations. More precisely, the so-called universal approximation theorem states that already a two-layer neural network can approximate any continuous function arbitrarily well, see [6, 15]. For more details on artificial neural networks and deep learning, we refer the reader to [12]. As mentioned in the two-layer case above, we are aiming to find a function $h(\cdot) = f_1 \circ f_2(\cdot) = \sum_{j=1}^{N} \alpha_j \mathcal{K}^2(\boldsymbol{x}_j, \cdot)$ with $f_1 \in \mathcal{H}_1$ and $f_2 \in \mathcal{H}_2$ and associated K_1 and K_2, respectively, where the kernel $\mathcal{K}^2$ is given by (11). The construction of h can be easily encoded as a feed-forward neural network with one hidden layer if K_1 is a radial basis function (RBF) kernel for instance[2]. We illustrate[3] the case $d_2 = 1$ with an RBF kernel $K_1(z_1, z_2) = a(|z_1 - z_2|)$ for some function $a : \mathbb{R} \to \mathbb{R}$ in Figure 2. The first layer is split into the input layer with values $K_2(\boldsymbol{x}_i, \boldsymbol{x})$ for $i = 1, \ldots, N$ and an artificial "always on" layer with neuron-clusters that supply the constant values $K_2(\boldsymbol{x}_i, \boldsymbol{x}_j)$ with weights $-c_j$ for $i, j = 1, \ldots, N$. Note that the i-th cluster $K_2(\boldsymbol{x}_i, \boldsymbol{x}_j)$ of the "always on" layer is only connected to the i-th neuron of the hidden layer. Note furthermore that the inputs $K_2(\boldsymbol{x}_i, \boldsymbol{x})$ can also easily be computed by a neural network with fixed weights if K_2 is a radial basis kernel. If we consider a "deeper" concatenation, we would need a deeper neural network with additional layers, i.e. for $f_1 \circ \ldots \circ f_L$, we need $L - 1$ hidden layers.

[2]For many other types of kernels, e.g. tensor products of RBF kernels, one can still construct a more complex Sigma-Pi neural network for the computation of the output values.

[3]Note that we only choose $d_2 = 1$ for illustrative reasons. For $d_2 > 1$, a neural network can be built analogously with an additional hidden layer to compute the norm of the difference of d_2-dimensional vectors. However, this additional layer, which just computes $\|\boldsymbol{x} - \boldsymbol{y}\|_2$ for given $\boldsymbol{x}$ and $\boldsymbol{y}$, has fixed weights and does not play any role for the optimization of the neural network.

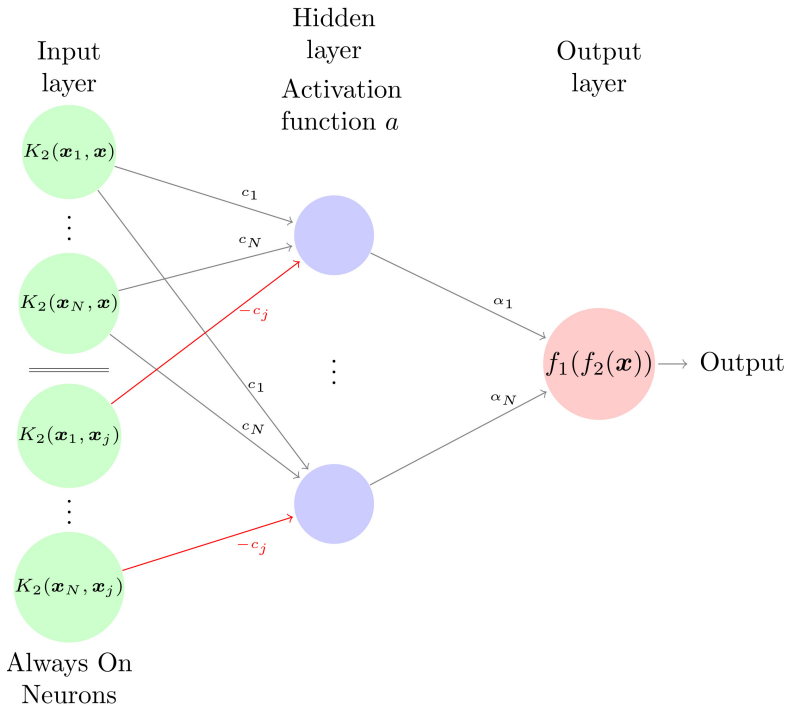


Figure 2: A hidden layer, feed-forward neural network to simulate the concatenation of two functions f_1 and f_2 from reproducing kernel Hilbert spaces. For reasons of readability, we choose $d_2 = 1$ and write $c_i := c_{i,1}$. The outer kernel is $K_1(z_1, z_2) = a(|z_1 - z_2|)$. Note that the i-th artificial "always on" neuron-cluster in the lower half of the first layer is written as $K_2(\boldsymbol{x}_i, \boldsymbol{x}_j)$, which stands for N single neurons with values $K_2(\boldsymbol{x}_i, \boldsymbol{x}_1), \ldots, K_2(\boldsymbol{x}_i, \boldsymbol{x}_N)$. The cluster $K_2(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is only connected to the i-th neuron of the hidden layer with weights $-c_j$ (red lines). This means that the value $\sum_{j=1}^{N} -c_j K_2(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is forwarded to the i-th neuron of the hidden layer.

3.3.2 Relation to multi-layer multiple kernel learning

The common idea in MLMKL methods is to learn a kernel $\tilde{K}$, which consists of a chain of linear combinations of functions and an inner kernel, e.g.

$$\tilde{K}(\boldsymbol{x}, \boldsymbol{y}) = \sum_{\ell=1}^{n_1} \nu_{1,\ell} k_{1,\ell} \left(\sum_{i=1}^{n_2} \nu_{2,i} K_{2,i}(\boldsymbol{x}, \boldsymbol{y}) \right)$$

in the two-layer case, where $k_{1,\ell}$ are real-valued functions for $\ell = 1, \ldots, n_1$ and $K_{2,i}$ are different scalar-valued kernels for $i = 1, \ldots, n_2$. Note that the functions $k_{1,\ell}$ are chosen such that $\tilde{K}$ is still a kernel. In the case of linear $k_{1,\ell}$, [9] has shown that the resulting algorithm becomes a standard MKL procedure and can be interpreted as a two-layer kernel network with a linear outer kernel. However, for arbitrary $k_{1,\ell}$ this is not the case and we are dealing with a true MLMKL approach. The specific MLMKL algorithm then aims to find the optimal values for the coefficients $\nu_{1,\ell}, \nu_{2,i}$ in order to determine the best $\tilde{K}$ for a regression of the given data X and Y with e.g. a support vector regression algorithm. Note that the kernels and the k-functions are usually chosen heuristically, e.g. as polynomials, Gaussians, sigmoidals, etc., see [21, 28].
To apply our result to the two-layer MKL method above, let us consider the case $n_1 = 1$ and $n_2 = N$. We set $\nu_{1,1} = 1$ without loss of generality. We let the outer function $k_{1,1}(z) = a(|z|)$ be the radial basis function

used for the outer kernel (i.e. middle layer) in Figure 2. Furthermore, we set

$$K_{2,i}(\boldsymbol{x}, \boldsymbol{y}) := K_2(\boldsymbol{x}_i, \boldsymbol{x}) - K_2(\boldsymbol{x}_i, \boldsymbol{y}).$$

Note that the $K_{2,i}$ are no longer kernels anymore in this setting. However, they are now directly connected to our concatenated function learning approach since

$$\begin{aligned}\tilde{K}(\boldsymbol{x}, \boldsymbol{y}) &= k_{1,1}\left(\sum_{i=1}^{N} \nu_{2,i} K_{2,i}(\boldsymbol{x}, \boldsymbol{y})\right) = a\left(\left|\sum_{i=1}^{N} \nu_{2,i} K_2(\boldsymbol{x}_i, \boldsymbol{x}) - \nu_{2,i} K_2(\boldsymbol{x}_i, \boldsymbol{y})\right|\right) \\ &= K_1\left(\sum_{i=1}^{N} \nu_{2,i} K_2(\boldsymbol{x}_i, \boldsymbol{x}), \sum_{i=1}^{N} \nu_{2,i} K_2(\boldsymbol{x}_i, \boldsymbol{y})\right) = \mathcal{K}^2(\boldsymbol{x}, \boldsymbol{y})\end{aligned}$$

from (11) with $c_i = \nu_{2,i}$ and the kernels K_1 and K_2 used in Figure 2. Altogether, we thus see that an MLMKL algorithm with these parameters already determines the optimal solution (provided that the right hand side of (9) is solved exactly) among all functions of type $h = f_1 \circ f_2$ with $f_1 \in \mathcal{H}_1$ and $f_2 \in \mathcal{H}_2$ according to Theorem 1. This way, our representer theorem for concatenated functions directly applies to a special case of MLMKL networks. Note however that a generalization of our arguments to more layers, i.e. $L > 2$, is not straightforward for MLMKL.

3.3.3 Relation to deep kernel learning approaches

The class of DKN methods consists of algorithms which build a kernel by nonlinearly transforming the input vectors before applying an outer kernel function. This is in contrast to the MLMKL approach, where only the innermost function is a two-variate kernel and its evaluations are modified by some nonlinear outer functions. The models in this class range from simple feature map powers for some function Ψ, i.e.

$$\tilde{K}(x, y) = \underbrace{\Psi \circ \ldots \circ \Psi}_{L-1 \text{ times}}(\boldsymbol{x}) \cdot \underbrace{\Psi \circ \ldots \circ \Psi}_{L-1 \text{ times}}(\boldsymbol{y}),$$

see [5], to more general variants like

$$\tilde{K}(x, y) = K\left(f_2 \circ \ldots \circ f_L(\boldsymbol{x}), f_2 \circ \ldots \circ f_L(\boldsymbol{y})\right)$$

with nonlinear functions $f_2, \ldots, f_L$, see [27]. If we assume that $f_l \in \mathcal{H}_l$ for $l = 2, \ldots, L$ stem from reproducing kernel Hilbert spaces with associated kernels K_l, we can apply Theorem 1 to this approach and obtain that each f_l can be written as a finite linear combination of evaluations of the kernel K_l. Thus, we can directly apply our representer theorem for L-layer DKN algorithms.

3.4 The two-layer interpolation problem

After analyzing the general multi-layer kernel concatenation problem in Theorem 1, we now have a closer, more detailed look at the main component of it, namely the *concatenation of two functions*. To this end, we specifically consider the interpolation problem for $L = 2$. This simple, illustrative setting gives further insights into the way concatenation works in machine learning problems.

3.4.1 Definition of the problem

We slightly adapt our notation to this special case to obtain a direct relation to the single-layer interpolation problem from Section 2. To this end, let $D := d_2$ and consider the domain $\Phi := D_1 \subseteq \mathbb{R}^D$ together with the two function spaces

$$\begin{aligned} &H(\Phi, \mathbb{R}) := \mathcal{H}_1 \subset C(\Phi) := \{f : \Phi \to \mathbb{R} \mid f \text{ continuous}\} && \text{– “outer” space,} \\ &\boldsymbol{H}(\Omega, \Phi) := \mathcal{H}_2 \subset \left\{\boldsymbol{g} = (g_1, \ldots, g_D)^T : \Omega \to \Phi \mid \boldsymbol{g} \text{ continuous}\right\} && \text{– “inner” space.} \end{aligned}$$

Both spaces are supposed to be reproducing kernel Hilbert spaces, i.e. there is an (outer) kernel $K := K_1 : \Phi \times \Phi \to \mathbb{R}$ for $H(\Phi, \mathbb{R})$ such that

$$\begin{aligned} K(\boldsymbol{x}, \cdot) &\in H(\Phi, \mathbb{R}) && \text{for all } \boldsymbol{x} \in \Omega, \\ f(\boldsymbol{x}) &= (f, K(\boldsymbol{x}, \cdot))_{H(\Phi,\mathbb{R})} && \text{for all } \boldsymbol{x} \in \Omega \text{ and all } f \in H(\Phi, \mathbb{R}). \end{aligned}$$

The function space $\boldsymbol{H}(\Omega, \Phi)$ is assumed to be a vector-valued RKHS, i.e. there is an (inner) kernel $\boldsymbol{K} : \Omega \times \Omega \to \mathbb{R}^{D \times D}$ such that

$$\begin{aligned} \boldsymbol{K}(\boldsymbol{x}, \cdot)\, \boldsymbol{c} &\in \boldsymbol{H}(\Omega, \Phi) && \text{for all } \boldsymbol{x} \in \Omega \text{ and all } \boldsymbol{c} \in \mathbb{R}^D, \\ \boldsymbol{c}^T \boldsymbol{g}(\boldsymbol{x}) &= (\boldsymbol{g}, \boldsymbol{K}(\boldsymbol{x}, \cdot)\, \boldsymbol{c})_{\boldsymbol{H}(\Omega,\Phi)} && \text{for all } \boldsymbol{x} \in \Omega, \text{ all } \boldsymbol{c} \in \mathbb{R}^D \text{ and all } \boldsymbol{g} \in \boldsymbol{H}(\Omega, \Phi). \end{aligned}$$

To formulate the concatenated interpolation problem in the spirit of (1), we have to define an appropriate functional and propose an appropriate search set for the minimization task. To this end, we consider the functional $J : H(\Phi, \mathbb{R}) \times \boldsymbol{H}(\Omega, \Phi) \to \mathbb{R}$ given by

$$J(f, \boldsymbol{g}) := \|f\|^2_{H(\Phi,\mathbb{R})} + \|\boldsymbol{g}\|^2_{\boldsymbol{H}(\Omega,\Phi)},$$

which penalizes the norms of both the outer and the inner function, and the admissible set

$$\mathcal{A}_{X,Y} := \{(f, \boldsymbol{g}) \in H(\Phi, \mathbb{R}) \times \boldsymbol{H}(\Omega, \Phi) \mid f \circ \boldsymbol{g}(\boldsymbol{x}_j) = y_j \ 1 \le j \le N\} \subset H(\Phi, \mathbb{R}) \times \boldsymbol{H}(\Omega, \Phi),$$

i.e. the set of all concatenations of functions from $H(\Phi, \mathbb{R})$ and $\boldsymbol{H}(\Omega, \Phi)$ which interpolate the data. With this notation, we can define the following variational optimization problem

$$J(f, \boldsymbol{g}) \to \min \quad \text{for } (f, \boldsymbol{g}) \in \mathcal{A}_{X,Y} \tag{P}$$

As explained in Section 2, the solution $f^*_{X,Y}$ to the standard interpolation problem (1) can be computed by solving the system (3) of linear equations for a given set of fixed and pairwise disjoint input data points $X := \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$. Therefore, if we assume for a moment the inner function $\boldsymbol{g}$ in (P) to be fixed and $Z := \boldsymbol{g}(X) = \{\boldsymbol{z}_i = \boldsymbol{g}(\boldsymbol{x}_i) \mid i = 1, \ldots, N\}$, then we obtain that the solution $f^*_{Z,Y}$ to (1) with data points Z is the only admissible minimizer of the concatenated interpolation problem (P), i.e.

$$f^*_{Z,Y} = \operatorname*{arg\,min}_{f \in \{h \in H(\Phi,\mathbb{R}) \mid (h, \boldsymbol{g}) \in \mathcal{A}_{X,Y}\}} \|f\|^2_{H(\Phi,\mathbb{R})}.$$

Note that the coefficients $\boldsymbol{\alpha}^* \in \mathbb{R}^N$ of $f^*_{Z,Y} = \sum_{i=1}^N \alpha^*_i K(\boldsymbol{z}_i, \cdot)$ can be computed by solving the system

$$\boldsymbol{M}_{Z,Z} \boldsymbol{\alpha}^* = \boldsymbol{y}$$

and the value of the optimal energy, i.e. the squared norm, is given by

$$\left\| f^*_{Z,Y} \right\|^2_{H(\Phi,\mathbb{R})} = {\boldsymbol{\alpha}^*}^T \boldsymbol{M}_{Z,Z} \boldsymbol{\alpha}^* = \boldsymbol{y}^T \boldsymbol{M}^{-1}_{Z,Z} \boldsymbol{y}.$$

3.4.2 Application of the representer theorem

In order to rewrite the concatenated interpolation problem (P) into an unconstrained minimization problem by applying the above result, we first have to discuss what happens if $\boldsymbol{g}(\boldsymbol{x}_j) = \boldsymbol{g}(\boldsymbol{x}_k)$ for two indices $j \neq k$. If equality holds also for the corresponding values from Y, i.e. $y_j = y_k$, we can simply remove the pair $(x_j, y_j) \in X \times Y$ from the input data and with it also the corresponding condition from the admissible set. However, if $y_j \neq y_k$, there cannot be an $f \in H(\Phi, \mathbb{R})$ such that $(f, \boldsymbol{g}) \in \mathcal{A}_{X,Y}$. In this case, we simply set $J(f, \boldsymbol{g}) = \infty$. Using this convention, we can recast (P) into the unrestricted optimization problem

$$J\left(f^*_{\boldsymbol{g}(X),Y}, \boldsymbol{g}\right) = \boldsymbol{y}^T \boldsymbol{M}^{-1}_{\boldsymbol{g}(X),\boldsymbol{g}(X)} \boldsymbol{y} + \|\boldsymbol{g}\|^2_{\boldsymbol{H}(\Omega,\Phi)} \to \min \quad \text{for } \boldsymbol{g} \in \boldsymbol{H}(\Omega, \Phi). \tag{uP}$$

Therefore, we only have to consider the minimization with respect to $\boldsymbol{g} \in \boldsymbol{H}(\Omega, \Phi)$ since the optimal outer function $f^*_{\boldsymbol{g}(X),Y}$ is completely determined by the inner function values $\boldsymbol{g}(X)$ and Y.
Note that the side condition $\boldsymbol{g}(\boldsymbol{x}_j) \neq \boldsymbol{g}(\boldsymbol{x}_k)$ for $j \neq k$ can also be enforced by adding a penalty term of type $\sum_{i<j} W\left(\|\boldsymbol{g}(\boldsymbol{x}_i) - \boldsymbol{g}(\boldsymbol{x}_j)\|_2^2\right)$ to J, where W is a smooth function with $W(0) = \infty$, e.g. $W(x) = \coth(x)$. This can also remedy the problem of small condition numbers of $\boldsymbol{M}_{\boldsymbol{g}(X),\boldsymbol{g}(X)}$ for large sample sizes since it maximizes distances between the point evaluations of $\boldsymbol{g}$. Adding this to (uP), we obtain

$$J_\gamma\left(f^*_{\boldsymbol{g}(X),Y}, \boldsymbol{g}\right) := J\left(f^*_{\boldsymbol{g}(X),Y}, \boldsymbol{g}\right) + \gamma \sum_{1 \leq i < j \leq N} \coth\left(\|\boldsymbol{g}(\boldsymbol{x}_i) - \boldsymbol{g}(\boldsymbol{x}_j)\|_2^2\right) \tag{15}$$
$$\to \min \quad \text{for } \boldsymbol{g} \in \boldsymbol{H}(\Omega, \Phi).$$

However, since using $J_0 = J$ in our experiments in Section 4 works out already well and the side condition does not seem to affect the results for moderate sample sizes, we restrict ourselves to the problem (uP) in the following.
Although the above considerations seem to simplify the concatenated interpolation problem, we still have to solve a highly nonlinear optimization problem over the (possibly) infinite-dimensional RKHS $\boldsymbol{H}(\Omega, \Phi)$. Nonetheless, by applying Theorem 1 to the unrestricted concatenated interpolation problem (uP), we can restrict the search space $\boldsymbol{H}(\Omega, \Phi)$ to the span of the kernel translates in the input data.

Corollary 3. *Let $V_X := \operatorname{span}\{\boldsymbol{K}(\boldsymbol{x}_i, \cdot)\boldsymbol{e}_j \mid i = 1, \ldots, N \text{ and } j = 1, \ldots, D\}$, where $\boldsymbol{e}_j$ denotes the j-th unit vector in $\mathbb{R}^D$. Then, the solution $\boldsymbol{g}^*$ to the unconstrained concatenated interpolation problem* (uP) *fulfills $\boldsymbol{g}^* \in V_X$.*

Proof. We apply Theorem 1 with $L = 2$, $\Theta_1(x) = \Theta_2(x) = x$ and

$$\mathcal{L}(y_i, f \circ \boldsymbol{g}(\boldsymbol{x}_i)) = \begin{cases} 0 & \text{if } f \circ \boldsymbol{g}(\boldsymbol{x}_i) = y_i \\ \infty & \text{else,} \end{cases}$$

which exactly resembles the interpolation problem (uP). □

Due to Corollary 3, we can recast the unrestricted concatenated interpolation problem (uP) into

$$J\left(f^*_{\boldsymbol{g}(X),Y}, \boldsymbol{g}\right) = \boldsymbol{y}^T \boldsymbol{M}^{-1}_{\boldsymbol{g}(X),\boldsymbol{g}(X)} \boldsymbol{y} + \|\boldsymbol{g}\|^2_{\boldsymbol{H}(\Omega,\Phi)} \to \min \text{ for } \boldsymbol{g} \in V_X \subset \boldsymbol{H}(\Omega, \Phi). \tag{uP-X}$$

This is a nonlinear, finite-dimensional and unrestricted optimization problem. We fix the kernel basis $\{\boldsymbol{K}(\boldsymbol{x}_j, \cdot)\boldsymbol{e}_\ell \mid (j, \ell) \in \mathcal{I}\}$ with $\mathcal{I} := \{(j, \ell) \in \mathbb{N}^2 \mid 1 \leq j \leq N, 1 \leq \ell \leq D\}$ to solve (uP-X). Then, the optimal solution can be written as

$$\boldsymbol{g}^*(\cdot) = \sum_{(j,\ell) \in \mathcal{I}} c^*_{j,\ell} \boldsymbol{K}(\boldsymbol{x}_j, \cdot)\boldsymbol{e}_\ell. \tag{16}$$

In order to express the minimization problem (uP-X) with respect to the coefficients $\boldsymbol{c}^* = \left(c^*_{1,1}, \ldots, c^*_{N,D}\right)^T$, we introduce

$$\boldsymbol{Q}_{X,X}(\boldsymbol{c}) = \boldsymbol{M}_{\boldsymbol{g}(X),\boldsymbol{g}(X)} = \left(K\left(\sum_{(j,\ell)\in I} c_{j,\ell} \boldsymbol{K}(\boldsymbol{x}_j, \boldsymbol{x}_n)\boldsymbol{e}_\ell, \sum_{(j,\ell)\in I} c_{j,\ell} \boldsymbol{K}(\boldsymbol{x}_j, \boldsymbol{x}_m)\boldsymbol{e}_\ell\right)\right)_{1 \leq n,m \leq N} \tag{17}$$

and the corresponding quadratic form

$$\mathcal{Q} : \mathbb{R}^{ND} \to \mathbb{R}, \quad \boldsymbol{c} \mapsto \boldsymbol{y}^T \boldsymbol{Q}_{X,X}(\boldsymbol{c})^{-1} \boldsymbol{y}.$$

Furthermore, to express $\|\boldsymbol{g}^*\|^2_{\boldsymbol{H}(\Omega,\Phi)}$ with respect to $\boldsymbol{c}^*$, we need

$$\mathcal{N} : \mathbb{R}^{ND} \to \mathbb{R}, \quad \boldsymbol{c} \mapsto \sum_{j,k=1}^{N} \begin{pmatrix} c_{j,1} \\ \vdots \\ c_{j,D} \end{pmatrix}^T \boldsymbol{K}\left(\boldsymbol{x}_j, \boldsymbol{x}_k\right) \begin{pmatrix} c_{k,1} \\ \vdots \\ c_{k,D} \end{pmatrix}. \tag{18}$$

Finally, we obtain the finite-dimensional optimization problem

$$\boldsymbol{c}^* = \underset{\boldsymbol{c}\in\mathbb{R}^{ND}}{\arg\min} \quad \underbrace{\mathcal{Q}\left(\boldsymbol{c}\right)}_{\|f^*_{\boldsymbol{g}(X),Y}\|^2_{H(\Phi,\mathbb{R})}} + \underbrace{\mathcal{N}\left(\boldsymbol{c}\right)}_{\|\boldsymbol{g}\|^2_{\boldsymbol{H}(\Omega,\Phi)}}. \tag{Int}$$

3.4.3 Solving the minimization problem

The unconstrained problem (Int) is highly nonlinear because the coefficients $c_{j,\ell}$ are transformed by the outer kernel function K. It can be tackled by any suitable iterative optimization algorithm. If the kernels K and $\boldsymbol{K}$ are differentiable, a quasi-Newton approach is appropriate. If this is not the case, a derivative-free optimizer should be chosen.
Note that we can restrict the minimization in (Int) to a compact subset of $\mathbb{R}^{ND}$ without loss of generality. To this end, let $\underline{\boldsymbol{K}} \in \mathbb{R}^{ND\times ND}$ be the $N \times N$ matrix of matrices $\boldsymbol{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathbb{R}^{D\times D}$ and note that

$$\mathcal{Q}(\boldsymbol{c}) + \mathcal{N}(\boldsymbol{c}) \geq 0 + \lambda_{\min}\left(\underline{\boldsymbol{K}}\right) \cdot \|\boldsymbol{c}\|_2^2 \overset{\|\boldsymbol{c}\|_2\to\infty}{\longrightarrow} \infty,$$

where $\lambda_{\min}\left(\underline{\boldsymbol{K}}\right) > 0$ denotes the smallest eigenvalue of $\underline{\boldsymbol{K}}$. Therefore, we can restrict our search to the compact set $A := \left\{\boldsymbol{c} \in \mathbb{R}^{ND} \mid \|\boldsymbol{c}\|_2 \leq C\right\}$ for a large enough $C > 0$. Unfortunately, we cannot directly obtain the existence of a minimizer from this since (Int) is not continuous. However, if we add a smooth term

$$\begin{aligned}\mathcal{P}^\gamma(\boldsymbol{c}) &= \gamma \sum_{1\leq m<n\leq N} \coth\left(\|\boldsymbol{g}(\boldsymbol{x}_m) - \boldsymbol{g}(\boldsymbol{x}_n)\|_2^2\right) \\ &= \gamma \sum_{1\leq m<n\leq N} \coth\left(\left\|\sum_{(j,\ell)\in\mathcal{I}} c_{j,\ell}\left(\boldsymbol{K}\left(\boldsymbol{x}_j, \boldsymbol{x}_m\right) - \boldsymbol{K}\left(\boldsymbol{x}_j, \boldsymbol{x}_n\right)\right)\boldsymbol{e}_\ell\right\|_2^2\right),\end{aligned}$$

which is equivalent to (15), for $\gamma > 0$, we can deduce the existence of a minimizer with the direct method from the calculus of variations. To this end, note that for a minimizing sequence $(\boldsymbol{c}_i)_{i=1}^\infty$ of $\mathcal{Q} + \mathcal{N} + \mathcal{P}^\gamma$, there necessarily exist $i_0 \in \mathbb{N}$ and $C_0 > 0$ such that all mutual squared distances $\|\boldsymbol{g}(\boldsymbol{x}_m) - \boldsymbol{g}(\boldsymbol{x}_n)\|_2^2$ with $1 \leq m < n \leq N$ are larger than C_0 for all $\boldsymbol{c}_i$ with $i > i_0$. Therefore, we can restrict the minimization to the compact subdomain

$$A \cap \left\{\boldsymbol{c} \in \mathbb{R}^{ND} \mid \|\boldsymbol{g}(\boldsymbol{x}_m) - \boldsymbol{g}(\boldsymbol{x}_n)\|_2^2 \geq C_0 \ \text{ for } 1 \leq m < n \leq N\right\},$$

on which $\mathcal{Q} + \mathcal{N} + \mathcal{P}^\gamma$ is continuous, and the existence of a minimizer follows. Nevertheless, as we explained above, the critical condition $\mathcal{P}^\gamma(\boldsymbol{c}) = \infty$ is practically never met for moderate data set sizes and, therefore, it is safe to assume that there also exists a minimizer for (Int). Note however that, depending on the kernels and the data at hand, there usually might exist many minimizers and the solution to (Int) might not be unique. To reduce the chance of getting stuck in a local minimum, we propose to restart the minimization procedure several times with different starting values for $\boldsymbol{c}^*$.
Since we will be dealing with differentiable kernel functions in Section 4 and since the derivatives of these kernels can be computed explicitly, we propose a BFGS minimization algorithm to solve (Int). To this end,

note that the only derivatives we need are essentially the derivative of the inverse of $\boldsymbol{Q}_{X,X}(\boldsymbol{c})$, i.e.

$$\frac{\partial}{\partial c_{m,n}}\boldsymbol{Q}_{X,X}^{-1}(\boldsymbol{c}) = -\boldsymbol{Q}_{X,X}^{-1}(\boldsymbol{c})\frac{\partial}{\partial c_{m,n}}\boldsymbol{Q}_{X,X}(\boldsymbol{c})\boldsymbol{Q}_{X,X}^{-1}(\boldsymbol{c}),$$

and the derivative of $\boldsymbol{Q}_{X,X}(\boldsymbol{c})$. The latter consists of the derivative of the outer kernel K, which is known analytically for all kernel choices that we discuss in Section 4, and

$$\frac{\partial}{\partial c_{m,n}}\boldsymbol{g}(\boldsymbol{x}) = \frac{\partial}{\partial c_{m,n}}\sum_{(j,\ell)\in\mathcal{I}} c_{j,\ell}\boldsymbol{K}\left(\boldsymbol{x}_j,\boldsymbol{x}\right)\boldsymbol{e}_\ell = \boldsymbol{K}\left(\boldsymbol{x}_m,\boldsymbol{x}\right)\boldsymbol{e}_n$$

for each $(m,n)\in\mathcal{I}$. The overall computational cost complexity for one BFGS step, i.e. the evaluation of $\mathcal{Q},\mathcal{N}$ and their derivatives, is bounded by $\mathcal{O}\left(N^3D+(ND)^2\right)$.

3.5 Two-layer Least-squares regression

After the discussion of the two-layer interpolation problem in the last section, we now consider the regularized two-layer least-squares problem in more detail. This is a natural extension of the two-layer least-squares problem RLS2 considered in [9] to the case of nonlinear outer kernels.

3.5.1 Definition of the problem

For concatenated, regularized least-squares regression, the minimization task changes to

$$J_{\lambda,\mu}\left(f,\boldsymbol{g}\right) := \sum_{j=1}^{N}\left|f\circ\boldsymbol{g}\left(\boldsymbol{x}_j\right)-y_j\right|^2 + \lambda\left\|f\right\|_{H(\Phi,\mathbb{R})}^2 + \mu\left\|\boldsymbol{g}\right\|_{\boldsymbol{H}(\Omega,\Phi)}^2 \qquad \text{(R)}$$
$$\to \min \text{ for } f\in H\left(\Phi,\mathbb{R}\right), g\in\boldsymbol{H}\left(\Omega,\Phi\right)$$

with $\lambda,\mu>0$, which is in the same fashion as the standard least-squares regression problem (5).
Analogously to our considerations in Section 3.4, we find that, for fixed inner points $Z=\boldsymbol{g}(X)\subset\Phi$, the function $f_{Z,Y}^{\lambda}$, see (5), is the solution of the problem

$$\sum_{j=1}^{N}\left|f(\boldsymbol{z}_j)-y_j\right|^2 + \lambda\left\|f\right\|_{H(\Phi,\mathbb{R})}^2 \to \min \text{ for } f\in H\left(\Phi,\mathbb{R}\right).$$

The corresponding coefficients $\boldsymbol{\alpha}^{\lambda}\in\mathbb{R}^N$ with respect to the basis $\{K(\boldsymbol{z}_j,\cdot)\mid j=1,\ldots,N\}$ are computed by solving

$$\left(\boldsymbol{M}_{Z,Z}+\lambda\boldsymbol{I}\right)\boldsymbol{\alpha}^{\lambda} = \boldsymbol{y}.$$

Therefore, each of the terms of the optimal energy can be expressed as

$$\begin{aligned}\left\|f_{Z,Y}^{\lambda}\right\|_{H(\Phi,\mathbb{R})}^2 &= {\boldsymbol{\alpha}^{\lambda}}^T\boldsymbol{M}_{Z,Z}\boldsymbol{\alpha}^{\lambda} = \boldsymbol{y}^T\left(\boldsymbol{M}_{Z,Z}+\lambda\boldsymbol{I}\right)^{-1}\boldsymbol{M}_{Z,Z}\left(\boldsymbol{M}_{Z,Z}+\lambda\boldsymbol{I}\right)^{-1}\boldsymbol{y},\\ \sum_{j=1}^{N}\left|f_{Z,Y}^{\lambda}\left(\boldsymbol{z}_j\right)-y_j\right|^2 &= \left\|\boldsymbol{M}_{Z,Z}\boldsymbol{\alpha}^{\lambda}-\boldsymbol{y}\right\|_2^2 = \left\|\left(\boldsymbol{I}-\boldsymbol{M}_{Z,Z}\left(\boldsymbol{M}_{Z,Z}+\lambda\boldsymbol{I}\right)^{-1}\right)\boldsymbol{y}\right\|_2^2.\end{aligned}$$

3.5.2 Application of the representer theorem

Analogously to (uP), we can use

$$J_{\lambda,\mu}\left(f^{\lambda}_{\boldsymbol{g}(X),Y},\boldsymbol{g}\right) = \lambda \boldsymbol{y}^T \left(\boldsymbol{M}_{\boldsymbol{g}(X),\boldsymbol{g}(X)} + \lambda \boldsymbol{I}\right)^{-1} \boldsymbol{M}_{\boldsymbol{g}(X),\boldsymbol{g}(X)} \left(\boldsymbol{M}_{\boldsymbol{g}(X),\boldsymbol{g}(X)} + \lambda \boldsymbol{I}\right)^{-1} \boldsymbol{y} + \mu \left\|\boldsymbol{g}\right\|^2_{\boldsymbol{H}(\Omega,\Phi)} + \left\| \left(\boldsymbol{I} - \boldsymbol{M}_{\boldsymbol{g}(X),\boldsymbol{g}(X)} \left(\boldsymbol{M}_{\boldsymbol{g}(X),\boldsymbol{g}(X)} + \lambda \boldsymbol{I}\right)^{-1}\right) \boldsymbol{y} \right\|_2^2 \tag{19}$$

to reformulate (R) as

$$J_{\lambda,\mu}\left(f^{\lambda}_{\boldsymbol{g}(X),Y},\boldsymbol{g}\right) \to \min \text{ for } \boldsymbol{g} \in \boldsymbol{H}(\Omega,\Phi). \tag{uR}$$

Corollary 4. *The solution $\boldsymbol{g}^{\lambda,\mu}$ to the unconstrained concatenated regression problem* (uR) *fulfills $\boldsymbol{g}^{\lambda,\mu} \in V_X$.*

Proof. We apply Theorem 1 with $L = 2$, $\Theta_1(x) = \lambda \cdot x$, $\Theta_2(x) = \mu \cdot x$ and

$$\mathcal{L}\left(y_i, f \circ \boldsymbol{g}(\boldsymbol{x}_i)\right) = \left|f \circ \boldsymbol{g}\left(\boldsymbol{x}_i\right) - y_i\right|^2,$$

which resembles the regression problem (R). □

Hence, as for interpolation, we obtain a representer theorem for concatenated least-squares regression, which allows us to replace the infinite-dimensional optimization problem (R) with the finite-dimensional problem

$$J_{\lambda,\mu}\left(f^{\lambda}_{\boldsymbol{g}(X),Y},\boldsymbol{g}\right) \to \min \text{ for } \boldsymbol{g} \in V_X \subset \boldsymbol{H}(\Omega,\Phi). \tag{uR-X}$$

Finally, we want to express (uR-X) in terms of the coefficients $\boldsymbol{c}^{\lambda,\mu} = \left(c^{\lambda,\mu}_{1,1}, \ldots, c^{\lambda,\mu}_{N,D}\right)^T$ of $\boldsymbol{g}^{\lambda,\mu}$ with respect to the basis $\{\boldsymbol{K}(\boldsymbol{x}_j,\cdot)\,\boldsymbol{e}_\ell \mid (j,\ell) \in \mathcal{I}\}$. To this end, we set $\boldsymbol{A} := \left(\boldsymbol{Q}_{X,X}(\boldsymbol{c}) + \lambda \boldsymbol{I}\right)^{-1}$ and define the quadratic forms

$$\begin{aligned}
\mathcal{Q}^{\lambda} &: \mathbb{R}^{ND} \to \mathbb{R}, \quad \boldsymbol{c} \mapsto \lambda \cdot \boldsymbol{y}^T \boldsymbol{A} \boldsymbol{Q}_{X,X}(\boldsymbol{c})\, \boldsymbol{A} \boldsymbol{y}, \\
\mathcal{N}^{\mu} &: \mathbb{R}^{ND} \to \mathbb{R}, \quad \boldsymbol{c} \mapsto \mu \cdot \mathcal{N}(\boldsymbol{c}) \quad \text{and} \\
\mathcal{C}^{\lambda} &: \mathbb{R}^{ND} \to \mathbb{R}, \quad \boldsymbol{c} \mapsto \boldsymbol{y}^T \left(\boldsymbol{I} - \boldsymbol{Q}_{X,X}(\boldsymbol{c})\, \boldsymbol{A}\right)^T \left(\boldsymbol{I} - \boldsymbol{Q}_{X,X}(\boldsymbol{c})\, \boldsymbol{A}\right) \boldsymbol{y}
\end{aligned}$$

with the help of (17) and (18). Subsequently, we arrive at the optimization problem

$$\boldsymbol{c}^{\lambda,\mu} = \operatorname*{arg\,min}_{\boldsymbol{c} \in \mathbb{R}^{ND}} \mathcal{Q}^{\lambda}(\boldsymbol{c}) + \mathcal{N}^{\mu}(\boldsymbol{c}) + \mathcal{C}^{\lambda}(\boldsymbol{c}), \tag{Reg}$$

which is the equivalent to (uR-X).

3.5.3 Solving the minimization problem

Note that the existence of a minimizer follows by similar arguments as in the previous section for the interpolation problem, i.e.

$$\mathcal{Q}^{\lambda}(\boldsymbol{c}) + \mathcal{N}^{\mu}(\boldsymbol{c}) + \mathcal{C}^{\lambda}(\boldsymbol{c}) \geq \mu \cdot \lambda_{\min}\left(\underline{\boldsymbol{K}}\right) \cdot \left\|\boldsymbol{c}\right\|_2^2 \overset{\|\boldsymbol{c}\|_2 \to \infty}{\longrightarrow} \infty$$

and we can thus restrict the search for a minimizer to a compact subset of $\mathbb{R}^{ND}$. For regression we need the inverse of $\boldsymbol{Q}_{X,X}(\boldsymbol{c}) + \lambda \boldsymbol{I}$ to compute $\mathcal{Q}^{\lambda}$, which is positive definite for every $\lambda > 0$ and, therefore, there are no pathological cases as in the interpolation setting. Thus, the functions $\mathcal{Q}^{\lambda}, \mathcal{N}^{\mu}, \mathcal{C}^{\lambda}$ are continuous and the minimization of (Reg) over a compact subset of $\mathbb{R}^{ND}$ has a minimizer. Nevertheless, also in this case the minimizer is not necessarily unique.

While the optimization for the coefficients in the RLS2 algorithm proposed in [9] boils down to a simplex-constrained linear least-squares problem, we have to deal with a high degree of nonlinearity here. Nevertheless, if the kernel functions are differentiable, we can again - as in the interpolation case - employ a BFGS algorithm with several restarts to approximately find the optimal coefficients $\boldsymbol{c}^{\lambda,\mu}$. To this end, note that $\mathcal{Q}^\lambda$ and $\mathcal{N}^\mu$ can be computed similarly as $\mathcal{Q}$ and $\mathcal{N}$ in the interpolation case. Furthermore, also the derivative of $\mathcal{C}^\lambda$ can be computed with the same techniques since we essentially only need the derivatives of $\boldsymbol{Q}_{X,X}(\boldsymbol{c})$ and $\left(\boldsymbol{Q}_{X,X}(\boldsymbol{c}) + \lambda \boldsymbol{I}\right)^{-1}$. While the number of terms is larger than in the interpolation case, the asymptotic computational runtime is still bounded by $\mathcal{O}\left(N^3 D + (ND)^2\right)$. Furthermore, the condition number of the matrix $\boldsymbol{Q}_{X,X}(\boldsymbol{c}) + \lambda \boldsymbol{I}$ is smaller than the one of $\boldsymbol{Q}_{X,X}$, which had to be inverted for interpolation. Therefore, computing $\mathcal{Q}^\lambda(\boldsymbol{c})$ with an iterative solver for the application of $\left(\boldsymbol{Q}_{X,X}(\boldsymbol{c}) + \lambda \boldsymbol{I}\right)^{-1}$ needs fewer computational steps than computing $\mathcal{Q}(\boldsymbol{c})$ in the interpolation case.
Finally, let us remark that for both interpolation and least-squares regression there exists another possibility to obtain a finite-dimensional optimization problem from (P) and (R), respectively, without using the representer theorem. We could discretize the functions $f_1 \in \mathcal{H}_1$ and $f_2 \in \mathcal{H}_2$ by $\tilde{f}_1 \in V_1$ and $\tilde{f}_2 \in V_2$ with finite-dimensional spaces V_1, V_2, see e.g. [3] for an error analysis of this scenario for single-layer regression. However, when following this approach, the choice of the specific discretization can severely influence the results of the minimization. Furthermore, we are limited by the size of the dimensions of the discretization spaces V_1, V_2, which influences the computational costs for solving the underlying optimization problem.

4 The effects of concatenated learning

This section serves to illustrate the main operating principle behind the concatenated interpolation and regression algorithms presented in the previous section. Note that our brief considerations in this section are not meant to provide a thorough numerical analysis of the performance of the algorithms but are rather thought to aid the understanding of their internal mechanisms. For benchmarks of highly performant variants of our basic algorithms on real-world data we refer the interested reader to [7, 21, 28].

4.1 Kernel choice

For reasons of simplicity, we will stick to the two-layer case and to outer function spaces $H(\Phi, \mathbb{R})$ with associated kernel K which are defined on the whole space $\mathbb{R}^D$. This way, the image Φ of the inner function space is automatically contained in the domain of the outer function space. Furthermore, if not stated otherwise, we assume that the matrix-valued kernel $\boldsymbol{K} : \Omega \times \Omega \to \mathbb{R}^{D \times D}$ of the inner RKHS can be written as

$$\boldsymbol{K}(\boldsymbol{x}, \boldsymbol{y}) = K_{\mathcal{I}}(\boldsymbol{x}, \boldsymbol{y}) \cdot \operatorname{diag}(\boldsymbol{a}) \tag{20}$$

for some weight vector $\boldsymbol{a} \in \mathbb{R}^D_+$. Here, $\operatorname{diag}(\boldsymbol{a})$ denotes the diagonal matrix $\boldsymbol{A}$ with $\boldsymbol{A}_{ii} = \boldsymbol{a}_i$ and $K_{\mathcal{I}} : \Omega \times \Omega \to \mathbb{R}$ is a scalar-valued kernel function.
Possible outer and inner kernel functions K and $K_{\mathcal{I}}$ are the polynomial kernel

$$K_{\mathrm{Poly},p}(\boldsymbol{x}, \boldsymbol{y}) := \left(\boldsymbol{x}^T \boldsymbol{y} + 1\right)^p,$$

the Gaussian kernel

$$K_{\mathrm{Gauss},\sigma}(\boldsymbol{x}, \boldsymbol{y}) := \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{y}\|^2}{2\sigma^2}\right)$$

and the tensor-product Matérn kernel

$$K_{\mathrm{TensorMatérn},s}(\boldsymbol{x}, \boldsymbol{y}) := \prod_{i=1}^{d} \kappa_{\frac{2s-1}{2}}\left(|x_i - y_i|\right) \cdot |x_i - y_i|^{\frac{2s-1}{2}}.$$

where κ_α denotes the modified (hyperbolic) Bessel function of the second kind with parameter α. Note that the latter characterizes the Sobolev space of dominating mixed smoothness of order $s \in \mathbb{N}$, see e.g. [10, 13] for a bi-variate version. These Sobolev spaces play an important role for hyperbolic cross or sparse grid approximations for instance, see e.g. [4]. Note that the Gaussian kernel is already a tensor product kernel by nature.

4.2 Experiment design

Let us choose $\Omega = [-1, 1]^2$. We will evaluate our method for the two test functions

$$\begin{aligned} h_1 : \Omega \to \mathbb{R} \qquad & h_1(x, y) := (0.1 + |x - y|)^{-1} \\ h_2 : \Omega \to \mathbb{R} \qquad & h_2(x, y) := \begin{cases} 1 & \text{if } x \cdot y > \frac{3}{20} \\ 0 & \text{else} \end{cases} . \end{aligned}$$

The function h_1 employs a kink-like structure along the diagonal of the domain, while h_2 represents an indicator function with a jump. Neither of these two functions is an element of a reproducing kernel space spanned by any of the above kernel functions for arbitrary parameters $p, s \in \mathbb{N}, \sigma \in (0, \infty)$. Therefore they cannot be approximated too well by a single-layer method. The approximation of such functions with kinks or jumps by (a composition of) smooth functions plays an important role in applications from econometrics, finance or two-phase flow problems for example.
We choose $D = d = 2$, i.e. $\Omega, \Phi \subset \mathbb{R}^2$, and $\boldsymbol{a} = (1\ 1)^T$. Then, we independently draw $N = 100$ random equidistributed points $\{\boldsymbol{x}_1, \dots, \boldsymbol{x}_N\} \subset \Omega$ and set $y_i := h_*(\boldsymbol{x}_i) + \varepsilon_i$ for all $i = 1, \dots, N$ for the function $h_* \in \{h_1, h_2\}$. Here, ε_i are additive noise perturbations which are drawn i.i.d. according to a centered Gaussian distribution with standard deviation 0.01. To solve (Int) or (Reg), respectively, we use a BFGS algorithm with random initialization of the coefficient vector $\boldsymbol{c}$ of the inner function, see also (16). As the goal functions employ many local minima, we run the algorithm sufficiently many times to achieve a good approximation to the global minimum. It turned out that 64 runs were sufficient for our case of 100 data points in 2 dimensions. From the 64 runs we pick the vector $\boldsymbol{c}$ (and with this the functions f and $\boldsymbol{g}$) for which the smallest goal function value in (Int) or (Reg), respectively, is achieved.
To be able to compare our computed $f(g(\cdot))$, which approximates the true solution $f^*_{\boldsymbol{g}(X),Y}(\boldsymbol{g}^*(\cdot))$ or $f^\lambda_{\boldsymbol{g}(X),Y}\left(\boldsymbol{g}^{\lambda,\mu}(\cdot)\right)$, respectively, to the result of a standard kernel interpolation/regression, we also calculate the interpolant/regressor $w \in \left\{f^*_{X,Y}, f^\lambda_{X,Y}\right\}$. This resembles the solution to (1) or (5), respectively, for the reproducing kernel Hilbert space $H(\Omega, \mathbb{R})$ which employs the same kernel type and parameters as $H(\Phi, \mathbb{R})$ but on the domain Ω instead of Φ. We then define $\boldsymbol{t}_i, i = 1, \dots, n_t$, as the points of a uniform grid of meshwidth $\frac{1}{50}$ over $\Omega = [-1, 1]^2$, i.e. $n_t = 101^2$, and consider the *pointwise error*

$$|(f \circ \boldsymbol{g} - h_*)(\boldsymbol{t}_i)| \quad \text{and} \quad |(w - h_*)(\boldsymbol{t}_i)|,$$

which we visualize in a two-dimensional contour plot.

4.2.1 Interpolation

We first compare the results for two-layer interpolation, see (Int), with the results for single-layer interpolation, see (1). To this end, we choose an outer Matérn kernel $K = K_{\text{TensorMatérn},s}$ with $s = 1$ and an inner polynomial kernel $K_{\mathcal{I}} = K_{\text{Poly},p}$ with $p = 1$ or $p = 2$. In Figure 3 we display the pointwise errors. We observe that there is a visible improvement in the error when dealing with two-layer interpolation instead of single-layer interpolation. While the benefits of two-layer interpolation are already observable for the test function h_2, they become even more obvious for h_1. As explained in the beginning of Section 3, the fact that the kink of h_1 is not parallel to a coordinate axis poses a problem when dealing with the tensor-product kernel. Since a linear transformation (rotation) would suffice to remedy this problem, the polynomial kernel

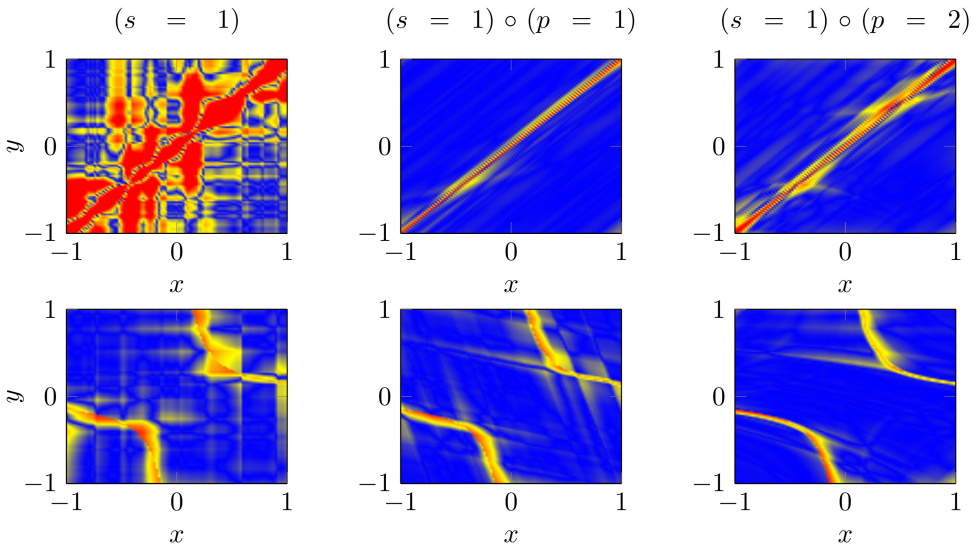


Figure 3: The pointwise error for standard interpolation with $w = f^*_{X,Y}$ (left) and for concatenated interpolation with outer kernel $K_{\text{TensorMatérn},1}$ and inner kernel $K_{\text{Poly},1}$ (mid) or $K_{\text{Poly},2}$ (right), respectively. We plotted both, the error for h_1 (top) and h_2 (bottom). The color scale ranges from blue (0% error) to red (more than 10% error), where the percentage has to be understood with respect to the $\|\cdot\|_{L_\infty}$ norm of h_1 or h_2, respectively.

of degree $p = 1$ already suffices to obtain a better error behavior. Therefore, $p = 2$ can already lead to a small overfitting effect as we observe in Figure 3. Nevertheless, the error is still significantly better than in the single-layer case. In the case of h_2, however, we have a jump along two nonlinear curves. Here, $p = 2$ seems to be more appropriate to deal with this problem. Overall, we come to the conclusion that interpolation in reproducing kernel Hilbert spaces can significantly benefit from a two-layer approach if the reproducing kernel at hand does not suit the underlying function.

4.2.2 Regression

Now we have a look at solving the least-squares regression problem (Reg). To determine the optimal parameters λ and μ, we run a 5-fold cross-validation on the input data for all possible choices $\lambda, \mu \in \{2^{-2t+1} \mid t = 1, \dots, 10\}$. Subsequently, we use the parameter pair (λ, μ) for which the smallest function value of (Reg) is achieved and run the regression algorithm on the whole input data set to obtain our final results. We compare the two-layer case with the single-layer regression, see also (5), with the parameter λ, which achieves the smallest error, i.e. we compare to the best possible single-layer solution.
Since the results for interpolation and least-squares regression with the same kernel choices as above happen to be similar, we employ an outer kernel of Gaussian type $K = K_{\text{Gauss},\sigma}$ with $\sigma = 0.1$ instead of Matérn type here. For the inner kernel we again choose $K_{\mathcal{I}} = K_{\text{Poly},p}$ with $p = 1, 2$. As we observe in Figure 4, there is a significant improvement of the two-layer approach over the single-layer one. Note that we deliberately employ the kernel width $\sigma = 0.1$, which appears to be too small for single-layer regression. However, the two-layer approach seems to remedy this bad choice automatically by adjusting the inner transformation accordingly. In this regard, the algorithm can also be understood as an implicit hyperparameter tuner.

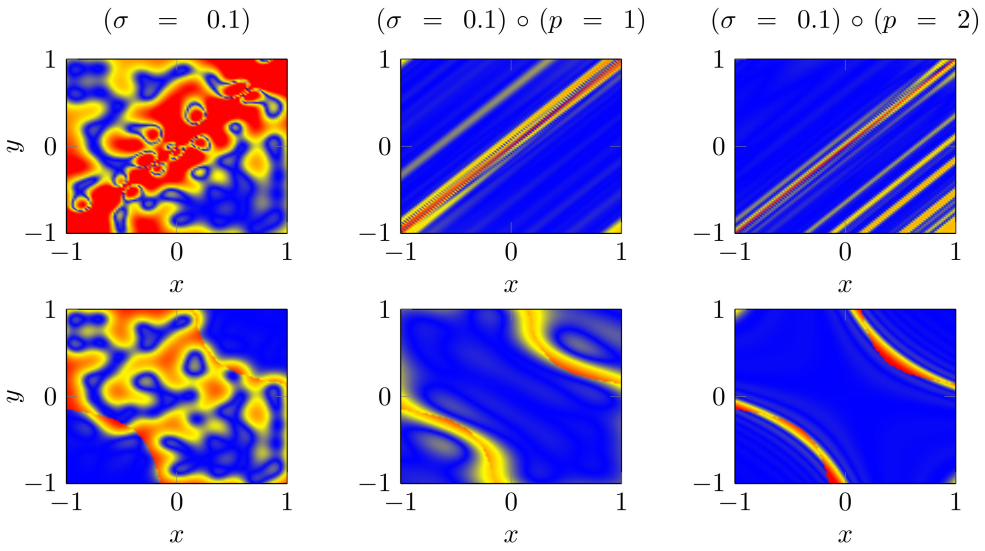


Figure 4: The pointwise error for standard least-squares with $w = f^{\lambda}_{X,Y}$ (left) and for concatenated least-squares with outer kernel $K_{\mathrm{Gauss},0.1}$ and inner kernel $K_{\mathrm{Poly},1}$ (mid) or $K_{\mathrm{Poly},2}$ (right), respectively. We plotted both, the error for h_1 (top) and h_2 (bottom). The color scale ranges from blue (0% error) to red (more than 10% error), where the percentage has to be understood with respect to the $\|\cdot\|_{L_\infty}$ norm of h_1 or h_2, respectively.

4.2.3 Linear outer kernel

In this section, we again want to emphasize the difference of our approach, which allows for nonlinear outer kernels, to the MKL-type RLS2 algorithm of [9], where only a linear outer kernel is considered and the inner kernel is given by a diagonal matrix with its entries being different scalar-valued (nonlinear) kernels. To this end, we run our two-layer least-squares regression approach for the following two settings:

(1) Outer polynomial kernel $K = K_{\mathrm{Poly},1}$ of order 1, inner mixture kernel $\boldsymbol{K}(\boldsymbol{x}, \boldsymbol{y})$,

(2) Outer Matérn kernel $K = K_{\mathrm{TensorMatérn},1}$ of order 1, inner mixture kernel $\boldsymbol{K}(\boldsymbol{x}, \boldsymbol{y})$.

For the inner mixture kernel, we deviate from (20) and from $D = 2$ here. To this end, we set $D = 5$ and use a diagonal kernel $\boldsymbol{K}$ with different scalar-valued kernels as entries. For the five scalar-valued kernels we choose three Gaussian kernels $K_{\mathrm{Gauss},\sigma}$ with $\sigma = 0.1, 1, 10$ and two polynomial kernels $K_{\mathrm{Poly},p}$ with $p = 1, 2$. Setting (1) serves to represent the RLS2 algorithm[4], where similar choices for the inner kernel have been made, see [9]. To determine the optimal parameters $\lambda, \mu \in \{10^{-2t+1} \mid t = 1, \ldots, 6\}$, we again run a 5-fold crossvalidation[5]. The results can be found in figure 5. As we have already seen for interpolation, the structure of the function h_1 admits a good representation by a two-layer kernel discretization of type (2). However, despite the quite generic choice of the inner kernel in setting (1), the two-layer kernel approach with a linear outer kernel is not able to find a good representation of the function. This shows that a nonlinear choice for the outer kernel can be necessary to find suitable approximations by the two-layer algorithm. Although the results do not differ that much for h_2, we again see that there is a slight advantage in approximating with a nonlinear outer kernel.

[4]Note however that we did not use a diagonal scaling of the linear kernel and our optimization algorithm is different from the one used in [9], which is adjusted to the problem with a linear outer kernel.

[5]Note that we scan a coarser (but wider) range than in the previous section, which seemed to be appropriate here.

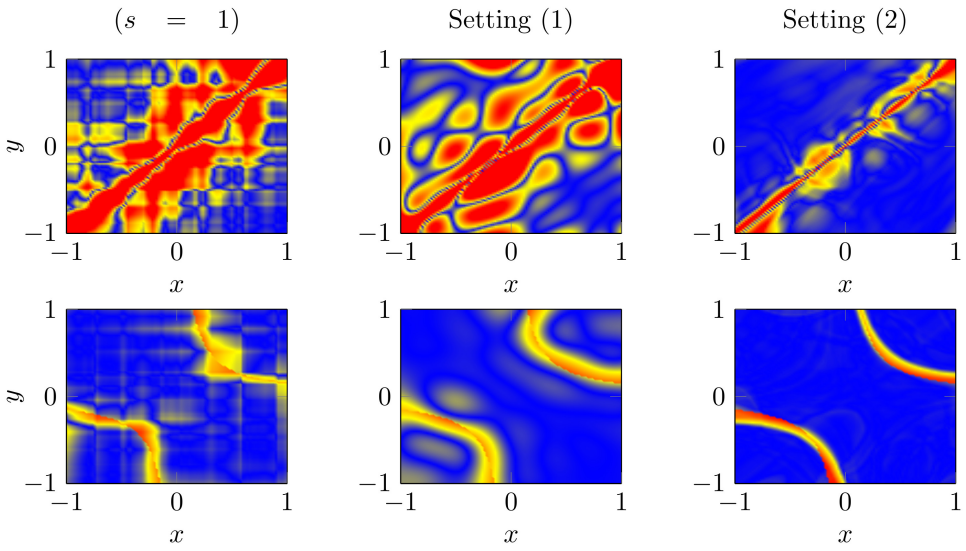


Figure 5: The pointwise error for standard least-squares with Matérn kernel and $w = f^{\lambda}_{X,Y}$ (left) and for concatenated least-squares with setting (1) (mid) and setting (2) (right) from section 4.2.3. We plotted both, the error for h_1 (top) and h_2 (bottom). The color scale ranges from blue (0% error) to red (more than 10% error), where the percentage has to be understood with respect to the $\|\cdot\|_{L_\infty}$ norm of h_1 or h_2, respectively.

4.3 Transformation by the inner function

To get a better impression on how the two-layer algorithms work, we exemplarily inspect the inner function $\boldsymbol{g}$ in the case of interpolation with $K = K_{\text{TensorMatérn},s}$ for $s = 1$ and $K_{\mathcal{I}} = K_{\text{Poly},p}$ for $p = 1$ or $p = 2$, i.e. for the setting from Section 4.2.1. To this end, we depict isotropic grid points in $\Omega = [-1, 1]^2$ and have a look at how these points are transformed by $\boldsymbol{g}$ in Figure 6.
We observe that for h_1 in both cases $p = 1$ and $p = 2$, the inner function aligns the kink almost perpendicular to the y-axis. Therefore, one can easily characterize the kink by the y-coordinate after the inner transformation. This reduces the original two-dimensional kink description $x - y = 0$ to just the one-dimensional description $y = 0$. While the function with the kink along the diagonal does not reside in the tensor-product Matérn space of order 1, which corresponds to the outer kernel in this example, a function with a kink parallel to one of the coordinate axes does. Therefore, the inner function $\boldsymbol{g}$ transforms the domain in such a way that the result resides in the RKHS to which the outer function belongs.
Considering the test function h_2, we see that a linear inner transformation, i.e. $p = 1$, essentially just rotates and shears the domain and does not change the alignment of the jump very much. However, in the case $p = 2$, the inner function $\boldsymbol{g}$ manages to transform the domain in such a way that the jump is now almost parallel to the y-axis. We observe that the pointwise errors in Figure 3 really benefit from this transformation and the jump is resolved almost perfectly. Overall, we see that the inner function $\boldsymbol{g}$ tries to align the features of the original test function in such a way that they can be easily resolved by the outer function f.

5 Conclusion

In this paper, we presented both a finite- and an infinite-sample representer theorem for concatenated machine learning problems. In the finite-sample case, the statement essentially boils down to the fact that the a priori infinite-dimensional optimization problem, which appears when dealing with function compositions from reproducing kernel Hilbert spaces, can be recast into a finite-dimensional optimization problem, where

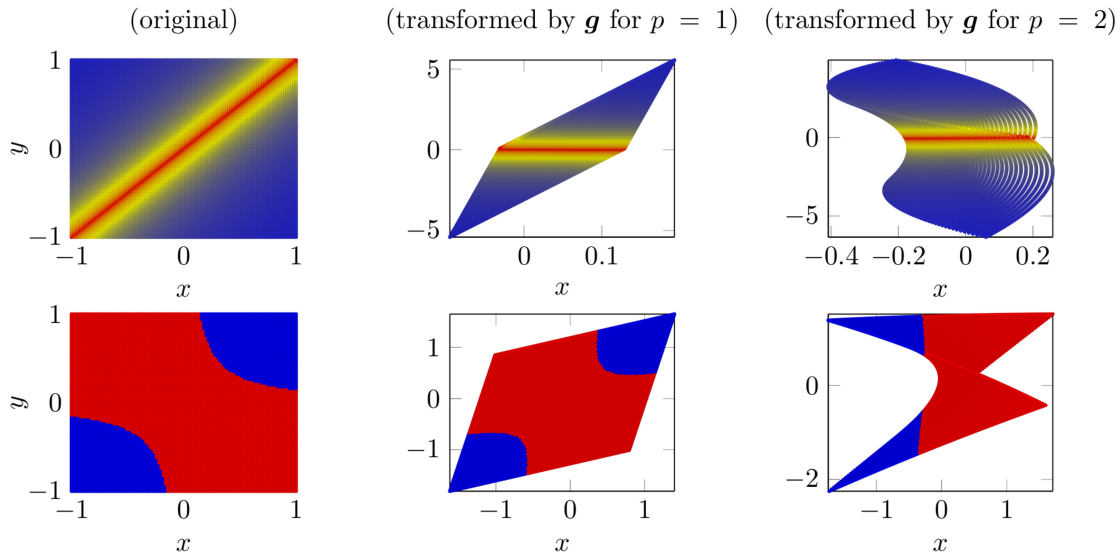


Figure 6: The transformation of the isotropic grid points (left) by the inner function with $p = 1$ (mid) and $p = 2$ (right). The underlying problem is interpolation of h_1 (top) and h_2 (bottom) for the outer kernel $K_{\text{TensorMatérn},1}$ and the inner kernel $K_{\text{Poly},p}$. The color scale represents the values of h_1 or h_2, respectively.

we only have to deal with at most N kernel translates in each layer of the composition. Here, N denotes the number of input data points. In the infinite-sample case, we derived an analogous result stating that the solution in each layer is an element of the image space of the integral operator defined by the corresponding kernel evaluated at the innermost functions. We introduced a simple neural network architecture, which represents the concatenated functions we are dealing with. Furthermore, we established a connection between our representer theorem and two types of state-of-the-art deep learning algorithms, namely multi-layer multiple kernel learning and deep kernel networks. Finally, we presented a detailed analysis on a two-layer interpolation and a two-layer least-squares regression algorithm, which can directly be derived from our representer theorem. We illustrated the operating principles of these algorithms with the help of two artificial test functions and explained why the two-layer approach is able to remedy the shortcomings of a single-layer variant. Furthermore, we highlighted that the use of a nonlinear outer kernel, instead of a linear one as in [9], can be inevitable to obtain good two-layer approximations. Nevertheless, the nonlinearity of the outer layer makes the numerical treatment of the underlying optimization problem more difficult.
While we presented specific two-layer ($L = 2$) algorithms and applied them to two-dimensional ($d = 2$) toy problems for illustrative reasons, our representer theorems can also be applied in the high-dimensional case with an arbitrary number of layers. Note furthermore that, apart from interpolation and least-squares regression, also more general choices of the loss function $\mathcal{L}$ and the regularizers Θ_l are allowed in (7). Therefore, one can also think of multi-layer support vector machines for instance. The construction of such efficient deep kernel learning algorithms for high-dimensional problems and a thorough analysis of the interplay between the number of layers L and the dimension d will be future work.

Appendix A: Remainder of the proof of theorem 2

To continue the proof of theorem 2, we note that we already showed that f_1 has the desired structure (14). Let us assume we have shown (14) for all $f_1, \dots, f_{l-1}$ for an $l \in \{2, \dots, L\}$. To obtain (14) for f_l, we proceed in the same fashion as in the first part of the proof in section 3.2. To this end, we now define

$J_{g_{l+1},\dots,g_L} : \mathcal{H}_l \to [0,\infty)$ by

$$J_{g_{l+1},\dots,g_L}(g_l) := \int_{R_{l+1}\times R_1} \tilde{\mathcal{L}}_l\left(y, g_l(\boldsymbol{\xi})\right)\ \mathrm{d}G_{l,\star}(\mathbb{P})(\boldsymbol{\xi}, y) + \lambda_l \|g_l\|^2_{\mathcal{H}_l},$$

where $\tilde{\mathcal{L}}_l(y,\boldsymbol{z}) := \mathcal{L}(y, f_1 \circ \ldots \circ f_{l-1}(\boldsymbol{z}))$ and $G_{l,\star}(\mathbb{P})$ is the pushforward of $\mathbb{P}$ onto $R_{l+1}\times R_1$ defined by $G_l(\boldsymbol{x}, y) = (g_{l+1}\circ\ldots\circ g_L(\boldsymbol{x}), y)$. Then it holds

$$\begin{aligned}&\min_{g_l\in\mathcal{H}_l,\dots,g_L\in\mathcal{H}_L} J(f_1, f_2, \dots, f_{l-1}, g_l, g_{l+1}, \dots, g_L)\\ =\ &\min_{g_{l+1}\in\mathcal{H}_{l+1},\dots,g_L\in\mathcal{H}_L}\left(\min_{g_l\in\mathcal{H}_l} J_{g_{l+1},\dots,g_{l+1}}(g_l)\right) + \sum_{i=l+1}^{L}\lambda_i\|g_i\|^2_{\mathcal{H}_i}\end{aligned}$$

and we need to show that a minimizer of $J_{g_{l+1},\dots,g_L}$ admits a representation of type (14). To this end, we begin by defining a Nemitski vector loss function and we subsequently prove that these loss functions admit the representation we need.

Definition 5. *Let $\mathcal{L} : R_1 \times D \to [0,\infty)$ for some domain $D \subset \mathbb{R}^d$. Let $\mathbb{P}_{R_1}$ denote the marginal distribution of $\mathbb{P}$ w.r.t. the second variable. We call $\mathcal{L}$ a $\mathbb{P}$-integrable Nemitski vector loss, if there exist $b : R_1 \to [0,\infty)$ with $b \in L_{1,\mathbb{P}_{R_1}}(R_1)$ and a measurable, increasing $h : [0,\infty) \to [0,\infty)$ such that*

$$\mathcal{L}(y,\boldsymbol{z}) \le b(y) + h(\|\boldsymbol{z}\|) \quad \text{for all } (y,\boldsymbol{z}) \in R_1 \times D.$$

If $\mathcal{L}$ is k-times differentiable w.r.t. the second variable for all $y \in R_1$, we call it a k-times differentiable Nemitski vector loss.

Lemma 6. *Let $l \in \{2,\dots,L\}$ and let $\mathbb{P}^l$ be a distribution[6] on $R_{l+1}\times R_1$ and let $\mathcal{L}^\star$ be a $\mathbb{P}^l$-integrable and 1-differentiable Nemitski vector loss on $R_1 \times R_l$ such that the derivative w.r.t. the second argument $\nabla_2\mathcal{L}$ fulfills*

$$\|\nabla_2\mathcal{L}^\star(y,\boldsymbol{z})\| \le b^\star(y) + h^\star(\|\boldsymbol{z}\|) \quad \text{for all } (y,\boldsymbol{z}) \in R_1 \times R_l$$

for some $b^\star \in L_{1,\mathbb{P}^l_{R_1}}(R_1)$ and a measurable, increasing $h^\star : [0,\infty) \to [0,\infty)$. Then, the functional $\mathcal{R}_{l,\mathbb{P}^l} : \mathcal{H}_l \to [0,\infty)$ defined by

$$\mathcal{R}_{l,\mathbb{P}^l}(f) := \int_{R_{l+1}\times R_1} \mathcal{L}^\star(y, f(\boldsymbol{z}))\ \mathrm{d}\mathbb{P}^l(\boldsymbol{z}, y)$$

is Frechet differentiable and the derivative $\mathrm{d}\mathcal{R}_{l,\mathbb{P}^l} : \mathcal{H}_l \to \mathcal{B}(\mathcal{H}_l, \mathbb{R})$ is given by

$$\mathrm{d}\mathcal{R}_{l,\mathbb{P}^l}(f)(g) = \int_{R_{l+1}\times R_1} \nabla_2\mathcal{L}^\star(y, f(\boldsymbol{z}))^T \cdot g(\boldsymbol{z})\ \mathrm{d}\mathbb{P}^l(\boldsymbol{z}, y). \tag{21}$$

Furthermore, a critical point of $J^\star : \mathcal{H}_l \to [0,\infty)$ defined by

$$J^\star(f) := \mathcal{R}_{l,\mathbb{P}^l}(f) + \lambda_l\|f\|^2_{\mathcal{H}_l}$$

is given by

$$f(\cdot) = -\frac{1}{2\lambda_l}\int_{R_{l+1}\times R_1} K_l(\cdot,\boldsymbol{z}) \cdot \nabla_2\mathcal{L}^\star(y, f(\boldsymbol{z}))\ \mathrm{d}\mathbb{P}^l(\boldsymbol{z}, y). \tag{22}$$

[6]Note that we set $R_{L+1} := D_L = \Omega$.

Proof. We have

$$\begin{aligned}
&\lim_{\|g\|_{\mathcal{H}_l}\to 0} \frac{\mathcal{R}_{l,\mathbb{P}^l}(f+g) - \mathcal{R}_{l,\mathbb{P}^l}(f) - \int_{R_{l+1}\times R_1} \nabla_2\mathcal{L}^\star(y, f(\boldsymbol{z}))^T \cdot g(\boldsymbol{z}) \, \mathrm{d}\mathbb{P}^l(\boldsymbol{z}, y)}{\|g\|_{\mathcal{H}_l}} \\
= &\lim_{\|g\|_{\mathcal{H}_l}\to 0} \int_{R_{l+1}\times R_1} \frac{\mathcal{L}^\star(y, f(\boldsymbol{z}) + g(\boldsymbol{z})) - \mathcal{L}^\star(y, f(\boldsymbol{z})) - \nabla_2\mathcal{L}^\star(y, f(\boldsymbol{z}))^T \cdot g(\boldsymbol{z})}{\|g\|_{\mathcal{H}_l}} \, \mathrm{d}\mathbb{P}^l(\boldsymbol{z}, y) \\
\overset{(*)}{=} &\int_{R_{l+1}\times R_1} \lim_{\|g\|_{\mathcal{H}_l}\to 0} \frac{\mathcal{L}^\star(y, f(\boldsymbol{z}) + g(\boldsymbol{z})) - \mathcal{L}^\star(y, f(\boldsymbol{z})) - \nabla_2\mathcal{L}^\star(y, f(\boldsymbol{z}))^T \cdot g(\boldsymbol{z})}{\|g\|_{\mathcal{H}_l}} \, \mathrm{d}\mathbb{P}^l(\boldsymbol{z}, y) = 0,
\end{aligned}$$

where the last equation follows from the differentiability of $\mathcal{L}^\star$ and $(*)$ follows by the dominated convergence theorem since the integrand is bounded by

$$\begin{aligned}
&\left| \frac{\mathcal{L}^\star(y, f(\boldsymbol{z}) + g(\boldsymbol{z})) - \mathcal{L}^\star(y, f(\boldsymbol{z})) - \nabla_2\mathcal{L}^\star(y, f(\boldsymbol{z}))^T \cdot g(\boldsymbol{z})}{\|g\|_{\mathcal{H}_l}} \right| \\
= &\left| \frac{\nabla_2\mathcal{L}^\star(y, cf(\boldsymbol{z}) + (1-c)g(\boldsymbol{z}))^T \cdot g(\boldsymbol{z}) - \nabla_2\mathcal{L}^\star(y, f(\boldsymbol{z}))^T \cdot g(\boldsymbol{z})}{\|g\|_{\mathcal{H}_l}} \right| \\
\leq &\, 2b^\star(y) + h^\star(\|cf(\boldsymbol{z}) + (1-c)g(\boldsymbol{z})\|) + h^\star(\|f(\boldsymbol{z})\|)
\end{aligned}$$

for some $c \in [0,1]$ due to the mean value theorem. Since the last line is bounded by $2b^\star(y) + 2h^\star(\|f(\boldsymbol{z})\| + 1)$ independently of g for any g with $\|g\|_{\mathcal{H}_l} \leq 1$, the dominated convergence theorem can be applied, which proves (21).
Since a critical point f of $J^\star$ fulfills

$$0 = \mathrm{d}J^\star(f)(g) = \mathrm{d}\mathcal{R}_{l,\mathbb{P}^l}(f)(g) + 2\lambda_l \langle f, g\rangle_{\mathcal{H}_l},$$

for all $g \in \mathcal{H}_l$, we obtain

$$\begin{aligned}
\langle f, g\rangle_{\mathcal{H}_l} = &-\frac{1}{2\lambda_l}\mathrm{d}\mathcal{R}_{l,\mathbb{P}^l}(f)(g) \\
= &-\frac{1}{2\lambda_l}\int_{R_{l+1}\times R_1} \nabla_2\mathcal{L}^\star(y, f(\boldsymbol{z}))^T \cdot g(\boldsymbol{z}) \, \mathrm{d}\mathbb{P}^l(\boldsymbol{z}, y) \\
= &-\frac{1}{2\lambda_l}\int_{R_{l+1}\times R_1} \nabla_2\mathcal{L}^\star(y, f(\boldsymbol{z}))^T \cdot \sum_{i=1}^{d_l} \langle g, K_l(\cdot, \boldsymbol{z})\boldsymbol{e}_i\rangle_{\mathcal{H}_l} \cdot \boldsymbol{e}_i \, \mathrm{d}\mathbb{P}^l(\boldsymbol{z}, y) \\
= &-\frac{1}{2\lambda_l}\sum_{i=1}^{d_l} \left\langle \int_{R_{l+1}\times R_1} \nabla_2\mathcal{L}^\star(y, f(\boldsymbol{z}))^T K_l(\cdot, \boldsymbol{z})\boldsymbol{e}_i \, \mathrm{d}\mathbb{P}^l(\boldsymbol{z}, y), g \right\rangle_{\mathcal{H}_l} \cdot \boldsymbol{e}_i
\end{aligned}$$

with the reproducing property of K_l, which is equivalent to the Bochner-type integral formulation (22). This finishes the proof. □

Now, we can apply lemma 6 with $\mathbb{P}^l = G_{l,\star}(\mathbb{P})$ and $\mathcal{L}^\star = \tilde{\mathcal{L}}_l$, which shows that a critical point $g_l^\star$ of $J_{g_{l+1},\ldots,g_L}$ can be written as

$$\begin{aligned}
g_l^\star(\cdot) = &-\frac{1}{2\lambda_l}\int_{R_{l+1}\times R_1} K_l(\cdot, \boldsymbol{\xi}) \cdot \nabla_2\tilde{\mathcal{L}}_l(y, g_l^\star(\boldsymbol{\xi})) \, \mathrm{d}G_{l,\star}(\mathbb{P})(\boldsymbol{\xi}, y) \qquad (23) \\
= &-\frac{1}{2\lambda_l}\int_{\Omega\times R_1} K_l(\cdot, g_{l+1}\circ\ldots\circ g_L(\boldsymbol{x})) \cdot \nabla_2\tilde{\mathcal{L}}_l(y, g_l^\star \circ g_{l+1}\circ\ldots\circ g_L(\boldsymbol{x})) \, \mathrm{d}\mathbb{P}(\boldsymbol{x}, y),
\end{aligned}$$

which is of type (14). Therefore, it just remains to show that $\tilde{\mathcal{L}}_l$ fulfills the prerequisites of lemma 6 and that $A_{f_l,f_{l+1},\ldots,f_L}(\boldsymbol{x}, y) := \nabla_2\tilde{\mathcal{L}}_l(y, f_l \circ f_{l+1}\circ\ldots\circ f_L(\boldsymbol{x})) \in L_{1,\mathbb{P}}$.

Lemma 7. *$\tilde{\mathcal{L}}_l$ is a $G_{l,\star}(\mathbb{P})$-integrable and 1-differentiable Nemitski loss and the derivative w.r.t. the second argument fulfills*

$$\left\|\nabla_2\tilde{\mathcal{L}}_l(y,\boldsymbol{z})\right\| \leq \tilde{b}(y) + \tilde{h}(\|\boldsymbol{z}\|) \quad \text{for all } (y,\boldsymbol{z}) \in R_1 \times R_l \tag{24}$$

for a $\tilde{b} \in L_{1,G_{l,\star}(\mathbb{P})_{R_1}}(R_1)$ and a measurable, increasing $\tilde{h} : [0,\infty) \to [0,\infty)$.

Proof. Since

$$|\tilde{\mathcal{L}}_l(y,\boldsymbol{z})| = |\mathcal{L}(y, f_1 \circ \ldots \circ f_{l-1}(\boldsymbol{z}))| \leq b_0(y) + h_0(f_1 \circ \ldots \circ f_{l-1}(\boldsymbol{z})) \leq b_0(y) + h_0(\|f_1\|_\infty),$$

$\tilde{\mathcal{L}}_l$ is a $G_{l,\star}(\mathbb{P})$-integrable Nemitski-loss. Here, we again used that $f_1 \in \mathcal{H}_1 \hookrightarrow C(D_1)$ because of (12). Since $f_1, \ldots, f_{l-1}$ are differentiable because the respective kernels are in C^1, $\tilde{\mathcal{L}}_l$ is also 1-differentiable by the chain rule. It remains to show (24). To this end, note that the chain rule gives us

$$\begin{aligned}
&\nabla_2\tilde{\mathcal{L}}_l(y,\boldsymbol{z})(\cdot) = \frac{\partial}{\partial \boldsymbol{z}}\left(\mathcal{L}\left(y, f_1 \circ \ldots \circ f_{l-1}(\boldsymbol{z})\right)\right) \\
&= \mathcal{L}^{(1)}(y, f_1 \circ \ldots \circ f_{l-1}(\boldsymbol{z})) \cdot \mathrm{d}f_1(f_2 \circ \ldots \circ f_{l-1}(\boldsymbol{z}))\left(\mathrm{d}f_2(f_3 \circ \ldots \circ f_{l-1}(\boldsymbol{z}))\left(\ldots \mathrm{d}f_{l-1}(\boldsymbol{z})(\cdot)\right)\right),
\end{aligned}$$

which leads to

$$\begin{aligned}
\|\nabla_2\tilde{\mathcal{L}}_l(y,\boldsymbol{z})\| &\leq \left(b_1(y) + h_1(|f_1 \circ \ldots \circ f_{l-1}(\boldsymbol{z})|)\right) \cdot \prod_{i=1}^{l-1} \sup_{\boldsymbol{x}_i \in D_i} \|\mathrm{d}f_i(\boldsymbol{x}_i)\|_{\mathcal{B}(D_i,R_i)} \\
&\leq \left(b_1(y) + h_1(\|f_1\|_\infty)\right) \cdot \prod_{i=1}^{l-1} \sup_{\boldsymbol{x}_i \in D_i} \|\mathrm{d}f_i(\boldsymbol{x}_i)\|_{\mathcal{B}(D_i,R_i)}.
\end{aligned} \tag{25}$$

Because of our assumption that we already showed (14) for $f_1, \ldots, f_{l-1}$ and because of (12), we get by the dominated convergence theorem that

$$\sup_{\boldsymbol{x}_i \in D_i} \|\mathrm{d}f_i(\boldsymbol{x}_i)\|_{\mathcal{B}(D_i,R_i)} \leq \frac{1}{2\lambda_i} c_i \|A_{f_i,\ldots,f_L}\|_{L_{1,\mathbb{P}}}$$

for all $i = 1, \ldots, l-1$. Therefore, by setting $\tilde{b}(y) := c \cdot b_1(y)$ and choosing a constant $\tilde{h} := c \cdot h_1(\|f_1\|_\infty)$ with $c := \prod_{i=1}^{l-1} \frac{1}{2\lambda_i} c_i \|A_{f_i,\ldots,f_L}\|_{L_{1,\mathbb{P}}} < \infty$, we obtain (24). □

Applying lemma 6 and lemma 7 shows us that f_l fulfills the integral equation (23). To conclude the proof of theorem 2, we note that

$$A_{f_l,f_{l+1},\ldots,f_L}(\boldsymbol{x},y) := \nabla_2\tilde{\mathcal{L}}_l(y, f_l \circ f_{l+1} \circ \ldots \circ f_L(\boldsymbol{x})) \in L_{1,\mathbb{P}},$$

which directly follows from (25) and the fact that $b_1 \in L_{1,\mathbb{P}_{R_1}}$. This finally shows that f_l admits a representation of type (14). Since the argument is valid for each $l = 2, \ldots, L$ and we already proved (14) for $l = 1$ in section 3.2, this finishes the proof of theorem 2.

References

[1] N. Aronszajn, *Theory of reproducing kernels*, Transactions of the American Mathematical Society **68** (1950), no. 3, 337–404.

[2] F. Bach, G. Lanckriet, and M. Jordan, *Multiple kernel learning, conic duality, and the SMO algorithm*, Proceedings of the 21st International Conference on Machine Learning, 2004, pp. 1–9.

[3] B. Bohn and M. Griebel, *Error estimates for multivariate regression on discretized function spaces*, SIAM Journal on Numerical Analysis **55** (2017), no. 4, 1843–1866.

[4] H.-J. Bungartz and M. Griebel, *Sparse grids*, Acta Numerica **13** (2004), 147–269.

[5] Y. Cho and L. Saul, *Kernel methods for deep learning*, Advances in Neural Information Processing Systems (Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta, eds.), vol. 22, Curran Associates, Inc., 2009, pp. 342–350.

[6] G. Cybenko, *Approximations by superpositions of sigmoidal functions*, Mathematics of Control, Signals, and Systems **2** (1989), no. 4, 303–314.

[7] A. Damianou and N. Lawrence, *Deep Gaussian processes*, Proceedings of the 16th International Conference on Artificial Intelligence and Statistics, 2013, pp. 207–215.

[8] S. de Marchi and R. Schaback, *Stability of kernel-based interpolation*, Adv. Comput. Math. **32** (2010), 155–161.

[9] F. Dinuzzo, *Learning functions with kernel methods*, Ph.D. thesis, University of Pavia, Pavia, Italy, 2011.

[10] G. Fasshauer and Q. Ye, *Reproducing kernels of generalized Sobolev spaces via a Green function approach with distributional operators*, Numerische Mathematik **119** (2011), no. 3, 585–611.

[11] M. Gönen and E. Alpaydin, *Multiple kernel learning algorithms*, JMLR **12** (2011), 2211–2268.

[12] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.

[13] M. Griebel and H. Harbrecht, *Approximation of bi-variate functions: singular value decomposition versus sparse grids*, IMA J. Numer. Anal. **34** (2014), no. 1, 28–54.

[14] A. Hinrichs, L. Markhasin, J. Oettershagen, and T. Ullrich, *Optimal quasi-Monte Carlo rules on higher order digital nets for the numerical integration of multivariate periodic functions*, Numerische Mathematik **134** (2016), no. 1, 163–196.

[15] K. Hornik, *Approximation capabilities of multilayer feedforward networks*, Neural Networks **4** (1991), no. 2, 251–257.

[16] G. Kimeldorf and G. Wahba, *A correspondence between Bayesian estimation on stochastic processes and smoothing by splines*, The Annals of Mathematical Statistics **41** (1970), no. 2, 495–502.

[17] S. Mallat, *Understanding deep convolutional networks*, Phil. Trans. R. Soc. A **374** (2016), no. 2065.

[18] H. Mhaskar, Q. Liao, and T. Poggio, *When and why are deep networks better than shallow ones?*, Proceedings of the 31st AAAI Conference on Artificial Intelligence, 2017, pp. 2343–2349.

[19] C. Micchelli and M. Pontil, *On learning vector-valued functions*, Neural Computation **17** (2005), 177–204.

[20] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K.-R. Müller, *Explaining nonlinear classification decisions with deep Taylor decomposition*, Pattern Recognition **65** (2017), 211–222.

[21] I. Rebai, Y. Benayed, and W. Mahdi, *Deep multilayer multiple kernel learning*, Neural Computing and Applications **27** (2016), no. 8, 2305–2314.

[22] S. Reddi, S. Sra, B. Poczos, and A. Smola, *Proximal stochastic methods for nonsmooth nonconvex finite-sum optimization*, Advances in Neural Information Processing Systems 29 (D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, eds.), Curran Associates, Inc., 2016, pp. 1145–1153.

[23] B. Schölkopf and A. Smola, *Learning with Kernels – Support Vector Machines, Regularization, Optimization, and Beyond*, The MIT Press – Cambridge, Massachusetts, 2002.

[24] I. Steinwart and A. Christmann, *Support vector machines*, Springer, New York, 2008.

[25] E. Strobl and S. Visweswaran, *Deep multiple kernel learning*, Proceedings of the 12th International Conference on Machine Learning and Applications, 2013, pp. 414–417.

[26] H. Wendland, *Scattered Data Approximation*, Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, 2005.

[27] A. Wilson, Z. Hu, R. Salakhutdinov, and E. Xing, *Deep kernel learning*, Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, 2016, pp. 370–378.

[28] J. Zhuang, I. Tsang, and S. Hoi, *Two-layer multiple kernel learning*, Proceedings of the 14th International Conference on Artificial Intelligence and Statistics, 2011, pp. 909–917.