# ReLU DNN expression of sparse gpc expansion in uncertainty quantification

Master thesis

submitted in partial fulfilment of
the requirements for the degree of

Master of Science ETH in Mathematics

ETH Zürich

Joost Aart Adriaan Opschoor*
Student ID: 16-931-776
ojoost@student.ethz.ch

Supervisor:
Prof. Dr. Christoph Schwab
Seminar for Applied Mathematics,
Department of Mathematics, ETH Zürich

Monday 3rd September, 2018

Minor revision on
Tuesday 30th October, 2018

*The author would like to thank Jakob Zech for the fruitful discussions. He is indebted to his supervisor for introducing him to the active research areas this thesis reviews and for stimulating conversations and feedback throughout the project.

## Abstract

In this thesis, we show a result on the expressive power of deep ReLU networks. We study a countably parametric elliptic diffusion equation with parameter space $U = [-1,1]^{\mathbb{N}}$ and show that its parameter-to-solution map $u : U \to H^2 \cap H_0^1((0,1)^2)$ can be approximated efficiently by a family of deep ReLU networks. We show a lower bound on the convergence rate in terms of the network size. The result we show is a generalisation of [75, Theorem 4.8 p. 22], which addresses a countably parametric diffusion equation on the spatial domain $(0,1)$. We show that the analogous result holds on $(0,1)^2$. The proof is based on the efficient approximation of products by deep ReLU networks introduced in [89] and the sparsity of the generalised polynomial chaos expansion (gpc expansion for short) of $u$ in the sense of $\ell^p$-summability of the norms of the gpc coefficients for some $0 < p < 1$ (shown in [3]). The proof is constructive if the gpc coefficients are known. The bound on the convergence rate for the approximation of $u$ only depends on the summability of the gpc coefficients and the convergence rate of deep ReLU network approximations of functions in $H^2 \cap H_0^1((0,1)^2)$. In particular, our main result also holds for parametric PDEs with infinitely many parameters. In that sense, it does not suffer from the curse of dimensionality.

# Contents

**1 Introduction** **1**
1.1 Uncertainty quantification . . . 1
1.1.1 Monte Carlo simulation . . . 1
1.1.2 Deterministic methods . . . 2
1.2 Deep learning . . . 3
1.2.1 Neural networks . . . 3
1.2.2 Non-residual feedforward neural networks . . . 6
1.2.3 Training . . . 6
1.2.4 Deep learning literature on approximation theory and on solving PDEs numerically . . . 7
1.3 Theorem 4.8 from [75] . . . 11
1.4 Notation . . . 12
1.4.1 General notation . . . 12
1.4.2 Index set $\mathcal{F}$ . . . 13
1.4.3 Parameter space $U$ . . . 13
1.4.4 Parametric functions . . . 13
1.5 Outline . . . 14

**2 ReLU neural network calculus** **15**
2.1 Formal description of non-residual feedforward ReLU neural networks . . . 15
2.1.1 Assumptions on the architecture and notation for networks with such architecture . . . 15
2.1.2 Construction of networks from subnetworks . . . 18
2.2 ReLU DNN approximation of products . . . 22
2.3 ReLU DNN approximation of products of multiple factors . . . 25

**3 ReLU DNN approximation in $\boldsymbol{H^2 \cap H_0^1((0,1)^2)}$** **32**
3.1 Continuous, piecewise bilinear interpolation in $H^2 \cap H_0^1((0,1)^2)$ . . . 32
3.2 ReLU DNN approximation of continuous, piecewise bilinear functions on $(0,1)^2$ . . . 34
3.3 Convergence rate of a family of ReLU DNN approximations of functions in $H^2 \cap H_0^1((0,1)^2)$ in terms of the network size . . . 37

**4 Properties of the Taylor gpc approximation of the solution map of a parametric diffusion equation** **39**
4.1 Properties of the non-parametric diffusion equation . . . 39
4.2 Properties of the parametric diffusion equation . . . 40
4.3 Weighted summability of the $H^1$-norms of the Taylor gpc coefficients of the solution map . . . 43
4.4 Weighted summability of the $H^2$-norms of the Taylor gpc coefficients of the solution map . . . 44

**5 ReLU DNN approximation of the solution map of a parametric diffusion equation** **47**

**6 Discussion and directions for further research** **54**
6.1 Discussion of Theorem 5.1 . . . 54
6.2 Alternatives for the ReLU DNN approximation of Taylor gpc coefficients . . . 55

6.2.1 More efficient ReLU DNN approximation of the full grid continuous, piecewise bilinear interpolants introduced in Section 3.1 . . . . . . . 55
6.2.2 Anisotropic sparse grid-based ReLU DNN approximation . . . . . . 56
6.3 Generalisations of the choice of domain . . . . . . . . . . . . . . . . . . . . . 57
6.3.1 Generalisation of Theorem 5.1 to $(0, 1)^d$ for $d > 2$ . . . . . . . . . . . 57
6.3.2 Generalisation of Theorem 5.1 to general polygonal domains . . . . . 57
6.4 Generalisation of the parametric PDE theory in Section 4 . . . . . . . . . . 58

**A Properties of concatenations and parallelisations 61**

**B Proofs 65**
B.1 Proof of Lemma 2.3 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 65
B.2 Proof of Proposition 2.12, Steps 3–5 . . . . . . . . . . . . . . . . . . . . . . 66
B.3 Proof of Proposition 2.14, Step 3 . . . . . . . . . . . . . . . . . . . . . . . . 69
B.4 Proof of Lemma 3.2 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 70
B.5 Proof of Lemma 4.3 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 71
B.6 Proof of Lemma 5.3 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 72
B.7 Proof of Proposition 5.1, Step 4 . . . . . . . . . . . . . . . . . . . . . . . . 74

# List of Figures

1.1 Example of a graph depicting a ReLU network that has two inputs, one output, depth 4 and size 12. . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

2.1 Structure of $\tilde{\times}_m$. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 24

2.2 Structure of $\tilde{\prod}_m^4$: a binary tree-structured network of $\tilde{\times}_m$-subnetworks. . . 26

3.1 Examples of univariate hat functions. . . . . . . . . . . . . . . . . . . . . . . . . . . . 33

3.2 ReLU approximation $\tilde{w}$ of $w \in V_2^{(\infty)}$. . . . . . . . . . . . . . . . . . . . . . . . . . 35

5.1 Structure of $\tilde{u}_3$; the sizes of the subnetworks $\{\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu} \in \Lambda_3} \cup \{f_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu} \in \Lambda_3} \cup \{\Phi^{\mathrm{Id}}_{2,L}, \Phi^{\mathrm{Id}}_{3,L'}\}$ depend on the approximated function $u$, this figure depicts a possible situation. By construction of Simul-Shared-Parallel in Remark A.5, there is only one identity network per Simul-Shared-Parallel-subnetwork. Moreover, some networks take the output of a hidden layer of one of the identity networks as input. . . . . . . . . . . . . . . . . . . . . . . . . . . 50

B.1 Structure of $F_4$. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 66

# 1 Introduction

In this section, we first introduce the context of this thesis. Section 1.1 introduces the field of uncertainty quantification and in particular methods to solve PDEs with random input using sparse generalised polynomial chaos expansions and approximations. Section 1.2 introduces the field of machine learning and mainly focusses on the deep neural networks we study. It also reviews some of the relevant deep learning literature. The aim of this thesis is to combine results from both fields to generalise [75, Theorem 4.8 p. 22], which is cited in Section 1.3. Section 1.4 introduces notation as well as an index set and a parameter space we will use throughout this thesis. Finally, in Section 1.5, we give an outline of Sections 2–6.

## 1.1 Uncertainty quantification

In science and engineering, many systems can be described mathematically by partial differential equations (*PDEs*), whose *data* (e.g. coefficients, source terms and boundary value data) describe the properties of the specific system under consideration. One way to investigate such a system is by solving the PDE numerically for data based on empirical information. Small deviations in the data can have an immense impact on the solution of the PDE, especially for non-linear PDEs, see e.g. [87, Section 1.1.1 pp. 1–2]. Such uncertainties may be a property of the studied system, but are also inherent in data based on empirical information. The quantitative study of such uncertainties and their influence on the exact or the numerical solution of the PDE is the active scientific field named *uncertainty quantification* (*UQ*).

UQ includes the development of exact and numerical PDE methods that take into account the previously described uncertainty. The uncertainty can be modelled by assuming that the data follow a probability distribution. Such spatial functions that follow a probability distribution are called *random fields*. A PDE with such a random field as data is a *PDE with random input* (*RPDE*). There are multiple ways to approach such an equation, we now discuss very briefly some of the developed RPDE methods, mainly focussing on gpc expansions and approximations introduced below. For a more complete introduction to UQ and in particular gpc expansions and approximations, including references to historically important literature, we refer to [87, especially Chapter 1 pp. 1–8] or [88]. These also serve as references for the introduction given in this section.

### 1.1.1 Monte Carlo simulation

One popular approach is that of *Monte Carlo simulation* (*MC simulation*), an example of a *sampling-based method*. In short, one computes independent draws from the probability distribution for the data and solves the deterministic PDE for those values of the data. Then, some statistical quantities of these solution samples are known to approximate those of the solution of the RPDE, e.g. the sample average of the solution samples converges to the mean solution of the RPDE. Although this method has been applied successfully in many situations, its convergence rate is generally considered slow, e.g. denoting by $N$ the number of deterministic problems that have to be solved, in general one only knows that the error of the sample average w.r.t. the mean solution is of the order $\mathcal{O}(N^{-1/2})$.

### 1.1.2 Deterministic methods

Alternatively, there exist *nonsampling methods* or *deterministic methods*, e.g. *perturbation methods* and, developed more recently, methods based on *generalised polynomial chaos expansions and approximations* (for short *gpc expansions and approximations*, named after *polynomial chaos* introduced in [85]). For both types of methods, one chooses bases for the function spaces from which the data are drawn, e.g. Karhunen Loève bases, see e.g. [29, Section 3.1 pp. 66–68]. The uncertainty in the data is modelled as uncertainty in the coefficients with respect to the chosen bases, i.e. these coefficients are modelled to be drawn from a probability distribution and the data of the RPDE are parametric functions with these coefficients as parameters. Solving the PDE for each possible value of the parameters defines a parameter-to-solution map, called *solution map.* The solution of the RPDE is determined by this parametric solution map and the probability distribution for the parameters. The remainder of this thesis is concerned with approximating the solution map of a parametric PDE, keeping in mind that results can also be applied in the context of RPDEs.

For the perturbation method, the parametric data map is Taylor expanded (in RPDE context usually around its mean) and approximated by a truncation of the expansion. The solution map is then approximated by solving the PDE for this approximation of the data, which is effective if the fluctuations of the data are small.

Alternatively, a *gpc expansion* is a polynomial expansion of the solution map in terms of the parameters. Truncation of a gpc expansion gives a *gpc approximation.* The coefficients of gpc expansions are functions in the solution space of the PDE and can themselves be approximated by Galerkin methods, for instance. This combination is referred to as a *stochastic Galerkin method* (*SG method*), cf. [30, 2, 79, 17]. Collocation methods are an alternative for Galerkin methods, the combination of a gpc approximation with a collocation method is referred to as a *stochastic collocation method* (*SC method*), cf. [1, 58, 57]. In case the number of parameters in the gpc expansion is bounded by some $M \in \mathbb{N}$, exponential convergence of an SG approximation in terms of the polynomial order used in the finite element approximation of the gpc coefficients is shown in [2, Theorems 6.2 and 6.3 p. 818]. However, the error may increase exponentially with $M$, a phenomenon called *the curse of dimensionality.* Hence, convergence rates independent of $M$ are desired. Such a result was first shown in [79].

Since then, many such results have followed based on sparsity of the gpc expansion as expressed by the $\ell^p$-summability of the norms of the gpc coefficients for some $0 < p < 1$. Early results in this direction are [17, Corollaries 7.4 p. 636 and 8.3 p. 644]. In addition, such sparsity has been derived from the so-called $(\boldsymbol{b}, \varepsilon)$-*holomorphy* of the solution map, a term that quantifies the holomorphic complex extendability of the solution map in each of the parameters. This approach is followed in e.g. [18, 14, 90] and used in [75, Theorems 2.7 p. 6, 3.9 p. 15, 3.10 p. 17 and Corollary 4.1 p. 18]. Another approach, not based on holomorphic extensions, was developed in [4, 3] and used to prove [75, Corollary 4.3 p. 18 and Theorem 4.8 p. 22]. The main topic of this thesis is [75, Theorem 4.8] (see Section 1.3 ahead), we hence review the second approach in Sections 4.2–4.4 below.

## 1.2 Deep learning

The main point in [75] is that it combines results on the sparsity of gpc expansions derived in the context of UQ with another quickly expanding field, namely that of *machine learning* (*ML*). The aim of that field is to develop algorithms that can extract features from data sets and that in new situations can make predictions based on "learned" features, but without designing the algorithms with those specific features in mind. In this context, the process by which an algorithm improves the extraction and prediction capabilities based on an available data set is called *learning* or *training* (see Section 1.2.3). Applications of machine learning are ubiquitous and include image recognition, speech recognition, autonomous driving and natural language understanding (see [73] and [32, Section 12 pp. 438–481]).

A sub-field of machine learning is *deep learning* (*DL*), which is the development and the study of *deep neural networks* (*DNNs*) for machine learning tasks. We refer to the reviews [45, 73] for an overview of the field up to 2015 and to the general introduction to the field in [32]. Although most success has been achieved in the past two decades, the concept is much older ([50]). We note that the principles behind what is now called deep learning have had different names over time, e.g. *cybernetics* introduced in [86] and used in the 1950s–1960s and *connectionism* used in the 1980s–1990s, see [32, Section 1.2.1 pp. 12–18]. The current wave of attention, under the name *deep learning*, started around 2006, see [32] and references therein, especially [32, p. 18 and Section 1.2.4 pp. 22–26] and e.g. [37, 8, 65]. Reasons for the recent success include the increased computational power of computers, the increased quantity and size of available data sets and the development of more efficient learning algorithms (see [32, Section 1.2 pp. 12–26]).

### 1.2.1 Neural networks

More specifically, *neural networks* (*NNs*) are computational algorithms that are defined in terms of a collection of *computational nodes* that each carry out a computation of specified type. The type of computation need not be the same for all computational nodes.

For most neural networks, the computational nodes are organised in *layers*. In this context, *depth* refers to the number of layers with computational nodes, we will call such layers *computational layers*. For the networks we consider, the input of the first computational layer is given externally, it is the *input* of the network. It consists of multiple numbers, called *input nodes*. Together, they constitute the *input layer*. All other layers are computational layers, the last of which is called *output layer*. The output of this layer is the *output* of the network.

Generally, for each computational layer, the type of computation is equal for all nodes in that layer. In the networks we consider, all computational layers carry out an affine transformation, which in all those layers except the last is followed by a non-linear transformation called *activation*. The activation is applied componentwise to the output of the affine transformation. See [19] for examples of networks whose first computational layer does not compute the activation function.

Currently, the most popular activation function is the *rectified linear unit* (*ReLU*) $\sigma : \mathbb{R} \to$

$\mathbb{R} : x \mapsto \max\{0, x\}$, which is globally Lipschitz continuous and hence weakly differentiable ($\sigma \in W^{1,\infty}_{\text{loc}}(\mathbb{R})$). Multiple variations of the ReLU exist, most of which are not constant on $(-\infty, 0)$. The reason why is explained in Section 1.2.3. Examples are $\mathbb{R} \to \mathbb{R} : x \mapsto \max\{0, x\} + 0.01 \min\{0, x\}$ (a *leaky ReLU*) and $\mathbb{R} \to \mathbb{R} : x \mapsto \max\{0, x\} + \min\{0, \exp(x) - 1\}$ (an *exponential linear unit*), cf. [16]. In addition, *sigmoidal functions* are a popular choice, which are functions $\psi$ that satisfy

$$\psi : \mathbb{R} \to \mathbb{R} : \lim_{x \to -\infty} \psi(x) = 0, \quad \lim_{x \to \infty} \psi(x) = 1, \tag{1.1}$$

often combined with monotonicity assumptions and regularity assumptions such as (Lipschitz) continuity or smoothness. Examples of smooth sigmoidal functions are the logistic function $x \mapsto \frac{1}{1+e^{-x}}$ and $x \mapsto \frac{2}{\pi} \arctan(\frac{\pi}{2} x)$. We note that the definition of "sigmoidal function" is not unambiguous, several different definitions can be found in the literature. Besides the ReLU and sigmoidal functions, the *binary step unit* (*BSU*) $\mathbb{R} \to \mathbb{R} : x \mapsto \mathbb{1}[x \geq 0]$ is a popular activation function. It is also called *Heaviside function.* Contrary to the ReLU and most popular sigmoidal functions, it is discontinuous and not weakly differentiable (it is not in $H^1_{\text{loc}}(\mathbb{R})$). At the end of Section 1.2.3 on DNN training, we further discuss activation functions and the importance of their regularity.

Computational layers in which nodes compute the activation function are called *hidden layers.* That is, for the networks we consider, all computational layers except for the output layer are hidden layers. Neural networks with not more than one hidden layer are called *shallow*, while networks with multiple hidden layers are called *deep.* In addition, we define the *size* of a network as the number of computational nodes applying the activation function.

The term "neural network" expresses the inspiration the field has found in neuroscience. For example, it builds on the idea that intelligent behaviour can result from complexly structured networks of interacting *neurons*, which are themselves simple computational nodes, and that learning is realised by properties of the nodes that change under external input (cf. Section 1.2.3 below). Although neural networks have been used to gain understanding of biological brains (see e.g. [38]), the currently popular neural networks are designed for machine learning tasks and do not faithfully model biological neural networks. To distinguish abstract neural networks from biological ones, the former are called *artificial neural networks* (*ANNs*). We only consider ANNs. The heuristic for the term "network" is that the dependencies between computational nodes can be represented by a graph: the dependence of the input of a computational node on the output of another node can be represented by an edge connecting the two nodes. An example of such a graph is shown in Figure 1.1.

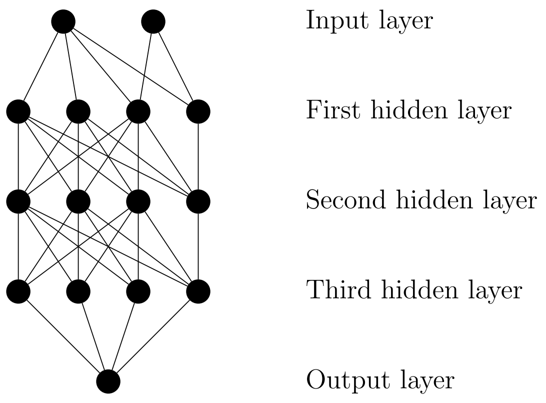


Figure 1.1: Example of a graph depicting a ReLU network that has two inputs, one output, depth 4 and size 12.

The network properties described so far, i.e. the collection of nodes organised into layers and the type of computation carried out by each computational node, are part of the *architecture* of the network. In addition, the architecture includes a rule that for each computational layer describes on the output of which layers its input may depend. The rule we use is described in Section 1.2.2 ahead. In particular, the architecture prescribes the size of each layer, which is the number of nodes.

In addition to the architecture, the computation carried out by a computational node also depends on a number of parameters, which are different for each node and are called *coefficients*. For each allowed value of the input of the network, the architecture and the coefficients of all computational nodes together determine the output of the network.

In the literature, the term “neural network” is used to refer to different objects. For example, it can refer to the architecture, the collection of all coefficients or the function that sends each allowed input value to the corresponding output value. In this thesis we will use “neural network” (or simply “network”) to refer to the combination of an architecture and a collection of corresponding coefficients. They together uniquely define a function from the set of allowed input values to the set of possible output values, which we call the *realisation* of the network, as in [62]. Therefore, we can speak about networks approximating a function.

Although each network has a unique realisation, the converse is not true: for each function that can be implemented by a ReLU network of the type described above, there are many other networks that implement the same function. This first of all follows from the positive homogeneity of the ReLU:

$$\forall \lambda \in \mathbb{R}_{>0}, x \in \mathbb{R} : \sigma(\lambda x) = \lambda \sigma(x).$$

In addition, Definition 2.4 below introduces networks that implement the identity operator. As will be discussed in Remark A.2, they can be used to extend networks without changing their realisation. Example 2.6 below shows a much less trivial case of two networks that have the same realisation, namely the standard hat function. Similar hat functions will be used in Sections 2.2 and 3.

Having fixed a realisation, the freedom in coefficients resulting from positive homogeneity could be removed by rescaling the coefficients. The other two types of freedom cannot simply be removed. In fact, such freedom shows that the relation between networks and their realisation is very complex. On the one hand, exploiting this complex relationship could lead to efficient DNN approximations (cf. Section 6.2.2). On the other hand, as a result, it might be difficult to bound from below the convergence rates of DNN training algorithms (training is further discussed in Section 1.2.3 below).

### 1.2.2 Non-residual feedforward neural networks

We restrict ourselves to *feedforward ReLU NNs* (*FF ReLU NNs*), which are ReLU networks with finitely many linearly ordered layers. The input of each layer may only depend on the outputs of the previous layers. Networks without this property are called *recurrent NNs*. They may have feedback connections from the output of a layer to the input of that layer or the input of a previous layer. Although in some situations recurrent networks have shown to outperform feedforward networks (cf. [73, e.g. Section 5.13 pp. 95–96]), all results in this thesis will be obtained without recurrence.

In addition, the networks we use are *non-residual*, i.e. the input of computational nodes in a certain layer may only depend on the output of the layer directly before, not on the output of layers before that layer. Connections between two non-consecutive layers appearing in residual NNs are called *skip connections*. Because skip connections could be implemented by identity networks as defined in Definition 2.4, the assumption that networks are non-residual is not very restrictive for the derivation of convergence rate bounds in terms of the network size or the number of non-zero coefficients. It only increases the number of computational nodes and the number of non-zero coefficients needed to implement certain functions.

In Section 2.1 ahead, we introduce a formalism to describe the non-residual FF ReLU NNs we use throughout this thesis.

### 1.2.3 Training

For DNN training, the architecture is generally fixed beforehand. The coefficients are not fixed, the aim of training is to optimise the coefficients for the task at hand. Two of the first models that could be trained were the *perceptron* introduced in [68] and *Adaline* (which stands for *adaptive linear*) introduced in [84], cf. also [82, 83].

In most cases, the aim of the training process is to find a network that minimises a certain loss function, which expresses the difference between the output of the network and the "true" value it needs to approximate. This type of learning uses a data set comprising pairs of input values and corresponding "true" output values and is called *supervised learning*.

An example of a supervised learning algorithm is *backpropagation* introduced in [71]. In each learning step, the loss function is evaluated on elements of the training data set. The outcome is used to compute the gradient of the loss function with respect to the coefficients. At the end of each step, the coefficients are changed along the negative gradient. This deterministic method is an example of *(deterministic) gradient descent*. Better results have been achieved with *stochastic gradient descent* (*SGD*), in which in each training step the coefficients are changed along the sum of the negative gradient of the loss function and a small random term. Choosing the size of the change in coefficients based on the size of the gradients observed in previous steps further improves the results. An example of an algorithm that does so is *Adam*. It was introduced in [42, Algorithm 1 p. 2] and has since received much attention, cf. also the review of optimisation methods based on deterministic or stochastic gradient descent in [70, Section 4.6 pp. 7–8] and the recent article [66] on possible improvements of Adam.

Gradient descent requires differentiability of the activation function. It is desirable that the activation function is continuously differentiable. In addition, it is undesirable that the activation function is constant on a part of its domain. It being constant can result in the vanishing of the gradient of the loss function with respect to the coefficients of a neuron, which means that the coefficients cannot be optimised anymore by gradient descent. Such nodes are called *dead*. In practice, this means that the size of networks with such activation functions needs to be chosen much larger than theoretically needed, because a significant part of the nodes will die during the training process.

The ReLU does not have the desired properties, it is only weakly differentiable and it is constant on $(-\infty, 0)$. Variations of the ReLU can be used to overcome one or both of these issues. The BSU is much less regular, it is discontinuous and its pointwise derivative vanishes everywhere except in $x = 0$, where it is not defined.

### 1.2.4 Deep learning literature on approximation theory and on solving PDEs numerically

Much literature is available on deep learning. On the one hand, many empirical works describe the results achieved with state-of-the-art architectures and training algorithms. Most of those results still lack a theoretical explanation and a proof guaranteeing good results. On the other hand, much theoretical research has been conducted, building on results obtained in e.g. information theory, statistics and numerical analysis. Our work is of the latter form. We will now introduce some of the theoretical results that are related to the work in this thesis.

If not specified otherwise, the networks discussed in this section are FF NNs whose computational layers compute an affine transformation, in all computational layers except the output layer followed by activation.

#### Universality

For a given neural network architecture, the first question to ask is which functions can be implemented by networks of that architecture. For other functions, one wants to know how well they can be approximated. A collection of networks with the same architecture, except for the fact that their layer sizes may differ, is called *universal* if any function from a certain function space (e.g. the space of continuous functions or the space of measurable functions) can be approximated arbitrarily well by networks from that collection.

For $n \in \mathbb{N}$, one hidden layer NNs with continuous sigmoidal activation function are universal in the sense that their realisations are dense in $C([0, 1]^n)$ in the topology of uniform convergence on compacta ([20, Theorem 1 p. 306]). See [20, Table 1 p. 312] for an overview of the results in that paper and similar results at that time. It was shown in [39, Theorem 2.4 p. 362] that the realisations of one hidden layer NNs with non-decreasing sigmoidal activation function are dense in the space of Borel measurable functions in the sense of global convergence in measure (the used metric is given in [39, Definition 2.9 and Lemma 2.1 p. 361]). Similar results were obtained in e.g. [28, 6].

An important survey of universality was given in [63]. Based on [47], it was shown in [63, Theorem 3.1 p. 153] that one hidden layer neural networks with continuous, non-polynomial activation function are universal, i.e. their realisations are dense in the space of continuous functions in the topology of uniform convergence on compacta. In the context of ReLU networks, this result implies universality of ReLU networks with more than one hidden layer: as discussed in Remark A.2 in Appendix A, the depth of ReLU networks can easily be increased without changing the realisation.

### Expressive power

Universality results, however, do not give any information on the size of the network or the number of non-zero coefficients needed for the implementation of a function or the approximation up to a prescribed accuracy. The ability of networks of a given architecture (i.e. with given depth and size) to implement or approximate functions is called *expressive power*. It is one of the main topics in deep learning research. Early results in this direction were given in [5, 52]. There, bounds were shown on the convergence rate of shallow network approximations of functions of a certain smoothness in terms of the network size. For similar results, see the references made there.

More recently, the expressive power of deep networks was studied in e.g. [10] and its published summary [9]. They showed the efficient representability of so-called affine systems by neural networks with as activation function a smooth version of the ReLU or a sigmoidal function of order $k \in \mathbb{N}_{\geq 2}$ as defined in [10, Definition 5.1 p. 19] (their definition of "sigmoidal" is more general than ours).

The results on expressive power stated above and below have in common that they show approximation properties in terms of the depth, the size or the number of non-zero coefficients of the networks. They do not show how to find such networks through training. In addition, in general, no restrictions are posed on the relation between the function to be approximated and the coefficients of the approximating networks. This relation may be very complicated, the coefficients need not depend continuously on the approximated function. In practice, this relation between function and coefficients has to be learned through DNN training, i.e. it depends on the performance of training algorithms whether the theoretical approximation results can easily be achieved in practice. Theoretical results on expressive power serve as benchmarks for the performance of training algorithms.

### Efficient ReLU approximation of arithmetics

Many recent results on ReLU DNN convergence rates in terms of the network size (e.g. the results obtained in [89, 62, 54, 75, 61, 24] and this thesis) depend on the efficient approximation of products by ReLU DNNs as first proposed in [89, Proposition 3 p. 106]. It exploits the fact that using the results in [77, Section 2.2 pp. 3–4] the function $[0,1] \ni x \mapsto x^2$ can be approximated efficiently by deep ReLU networks ([89, Proposition 2 p. 105]). We discuss this in more detail in Sections 2.2–2.3 below. A similar result on the efficient DNN approximation of products is [48, Theorem 1 p. 4], which uses networks with ReLUs and BSUs. A related result, but not used in [75] or this thesis, is the fact that division can be implemented efficiently by ReLU networks, which follows from [78, Theorem 1.1 p. 1]. That theorem shows a more general statement, namely that rational functions can be approximated efficiently by ReLU networks. Interestingly, it also shows the converse, namely that the realisations of ReLU networks can be approximated efficiently by rational functions.

**Expressive efficiency of depth**

A not yet fully understood issue concerning the expressive power of neural networks is the (dis)advantage of depth. Twenty years ago it was still unclear whether depth was advantageous in general, as depth is not needed for universality ([63, Section 7 pp. 182–187]). In the mean time, many promising results have been achieved on the *expressive efficiency of depth*. The aim is to show that for a fixed network size or a fixed number of non-zero coefficients deep networks outperform shallow (or less deep) networks at implementing and approximating functions, e.g. that for deep network approximations the convergence rate in terms of the network size or the number of non-zero coefficients is better than for shallow network approximations.

We now recall some results showing the expressive efficiency of depth. One hidden layer BSU NNs cannot provide localised approximations in Euclidean space of dimension more than one ([15, Theorem 2.2 p. 609]), whereas two hidden layer BSU NNs can ([15, Theorem 2.3 p. 610]). In [60, 55], it was shown that the realisations of ReLU DNNs are piecewise linear on a finite partition of their domain. In addition, lower bounds on the maximum possible number of linear regions were given in terms of depth and layer sizes (e.g. [55, Theorem 5 p. 7]). It was shown that for networks with fixed layer size this lower bound grows exponentially with depth ([55, Corollary 6 p. 7]), whereas the maximum number of linear regions that can be achieved by a single hidden layer network grows polynomially with the size of the network ([60, Proposition 2 p. 6]), i.e. deep networks can achieve exponentially more linear regions than shallow networks. The references [51, 53, 64] showed that compositional functions can be approximated efficiently by DNNs, not by shallow networks. In [89, 48], DNNs that approximate polynomials were constructed. In both works, such networks were used to show that deep networks are more efficient at approximating functions from certain smoothness classes than shallow networks. The results in [89] were achieved using ReLU networks, whereas in [48] networks with ReLUs and BSUs were used. For the approximation of a multivariate polynomial in $n \in \mathbb{N}$ variables on a bounded domain by networks with $n$ times differentiable activation functions with non-zero Taylor coefficients, it was shown in [67] that the required depth of networks with fixed layer size grows linearly with $n$, whereas the required layer size for finite depth networks grows exponentially with $n$. The optimal approximability of piecewise smooth functions by ReLU DNNs was studied in [62]. In particular, they showed a lower bound on the depth of networks that achieve the optimal convergence rate in terms of the number of non-zero coefficients. In [61], it was shown that ReLU DNNs are rate-distortion optimal for the approximation of sinusoidal functions, one-dimensional oscillatory textures and the Weierstrass function. In particular, in [61, Theorems 6.4 and 6.6 p. 9], it was proved that deep networks with finite width outperform networks with finite depth on the approximation problems considered in those two theorems.

A more general type of result was obtained in [19], which studies *sum-product networks*, which are also called *convolutional arithmetic circuits* and have a different architecture than the networks we consider. In [19, Corollary 2 p. 10], the coefficients of sum-product networks were interpreted as tensors. Under this correspondence, shallow sum-product networks implement the canonical tensor decomposition and deep sum-product networks implement the *hierarchical Tucker tensor decomposition* (*HT decomposition*, introduced in [35]), see also [34]. It was shown that Lebesgue-almost every tensor in HT format has canonical rank growing exponentially with the order of the tensor ([19, Theorem 1 p. 9]), which implies that for any given deep sum-product architecture the realisation of Lebesgue almost every network of that architecture can only be implemented or approximated arbitrarily closely by shallow sum-product networks of size exponentially growing with the depth of the deep network ([19, Corollary 2 p. 10]). The strength of this result is that it shows that *almost every* deep sum-product network cannot be approximated efficiently by a shallow network, whereas the previously cited results merely showed this for specific classes of networks. On the other hand, it is possible that most networks of interest are elements of the exceptional set of zero Lebesgue measure whose elements can be approximated by smaller shallow networks. An analogous result for recurrent neural networks was shown in [41], by showing that deep recurrent neural networks implement the *tensor train decomposition* (*TT decomposition*) introduced in [59].

**Numerically solving differential equations with neural networks**

We now discuss some results that use neural networks to solve PDEs numerically. The idea of using neural networks to solve differential equations numerically has been around for decades, see [46, 43, 44, 49]. Cf. also [76, Section 1.2 p. 3], which discusses more recent results that use deep learning to solve PDEs.

In order to train neural networks, a loss function is defined that evaluates how well a function from the solution space of the PDE satisfies the PDE, its boundary conditions and possibly initial conditions. The loss function can simply be based on the PDE itself, but also on a variational formulation of the PDE similar to that in Proposition 4.2 below. The latter approach was used in [23]. Often, e.g. in [46, 43, 44, 49], the loss function is evaluated on a mesh. Alternatively, in [76] it was proposed to train DNNs by SGD based on evaluation of the objective function in randomly drawn points in its domain.

Another research topic is the development of training algorithms for the specific purpose of solving PDEs numerically, see e.g. [69].

Recently, DNNs have even proved successful in numerically solving some specific examples of high-dimensional parametric PDEs, i.e. methods have been developed that in some situations can overcome the curse of dimensionality, at least up to a large extent. As discussed in Section 1.1.2, this can be used to solve RPDEs. An early result in this direction is [40]. An RPDE is solved by first calculating independent draws for the data of the underlying PDE. The PDE is then solved numerically for those values of the data, which gives data-solution pairs with which a DNN can be trained. The references [22, 36, 27] address high-dimensional non-linear PDEs and backward stochastic differential equations

(*BSDEs*) by showing that they correspond to a stochastic control problem, whose solution can be approximated using deep reinforcement learning. Numerical experiments show that problems with dimension of the order 100 can be handled. Another work following a roughly similar approach is [7]. It also relies on the correspondence of certain fully non-linear PDEs to BSDEs.

Very recently, the existence of a family of DNN approximations of the solution of a high-dimensional PDE was shown in [24, Theorem 7.3 p. 30], together with a lower bound on the convergence rate in terms of the number of non-zero coefficients that does not suffer from the curse of dimensionality.

**The contribution of [75] and this thesis**

The results in [75] and this thesis address the expressive power of DNNs and contribute to the area using DNNs to solve PDEs numerically. The results use the sparsity of gpc expansions and the efficient ReLU approximation of products to show the existence of a family of ReLU DNNs approximating the solution map of a parametric PDE and to give a lower bound on the convergence rate of the approximations in terms of the network size. The proofs are constructive: the architecture and the coefficients of the approximating family of networks are constructed explicitly, assuming that the gpc coefficients are known.

## 1.3 Theorem 4.8 from [75]

The aim of this thesis is to review [75, Theorem 4.8 p. 22] and its extensibility. The theorem shows the existence of a family of non-residual FF ReLU DNNs (see Section 1.2.2) approximating the solution map of a parametric elliptic diffusion equation on the domain $(0,1)$ with a scalar diffusion coefficient that is uniformly elliptic with respect to the parameters. More precisely, with $D = (0,1)$, $U = [-1,1]^{\mathbb{N}}$ (see Section 1.4.3 below) and spaces of real-valued functions $V = H_0^1(D)$ and $X = H^2 \cap H_0^1(D)$, it reads:

**Theorem 1.1** ([75, Theorem 4.8 p. 22])**.** *"Let $0 < q_V \leq q_X < 2$ and denote $p_V := (1/q_V + 1/2)^{-1} \in (0,1)$ and $p_X := (1/q_X + 1/2)^{-1} \in (0,1)$. Let $\boldsymbol{\beta}_V = (\beta_{V;j})_{j\in\mathbb{N}} \in (0,1)^{\mathbb{N}}$ and $\boldsymbol{\beta}_X = (\beta_{X;j})_{j\in\mathbb{N}} \in (0,1)^{\mathbb{N}}$ be two monotonically decreasing sequences such that $\boldsymbol{\beta}_V \in \ell^{q_V}(\mathbb{N})$ and $\boldsymbol{\beta}_X \in \ell^{q_X}(\mathbb{N})$, and such that"*

$$\left\| \frac{\sum_{j\in\mathbb{N}} \beta_{V;j}^{-1} |\psi_j(\cdot)|}{\bar{a}(\cdot)} \right\|_{L^\infty(D)} < 1$$

*and*

$$\left\| \frac{\sum_{j\in\mathbb{N}} \beta_{X;j}^{-1} |\psi_j(\cdot)|}{\bar{a}(\cdot)} \right\|_{L^\infty(D)} < 1, \qquad \left\| \sum_{j\in\mathbb{N}} \beta_{X;j}^{-1} |\psi_j'(\cdot)| \right\|_{L^\infty(D)} < \infty$$

*for $\bar{a}$ and all $\psi_j$ belonging to $W^{1,\infty}(D)$ and with $\operatorname{ess\,inf} \bar{a} > 0$. Assume that in*

$$\int_D a(\boldsymbol{y},x) \nabla u(\boldsymbol{y},x) \cdot \nabla v(x) \,\mathrm{d}x = {}_{V'}\langle f, v\rangle_V, \quad \forall v \in H_0^1(D) \tag{1.2}$$

$f \in L^2(D)$. *"[...] Denote for every $\boldsymbol{y} \in U$ by $u(\boldsymbol{y}, \cdot) \in V$ the solution of (1.2) for the affine-parametric diffusion coefficient*

$$a(\boldsymbol{y}, x) = \bar{a}(x) + \sum_{j \in \mathbb{N}} y_j \psi_j(x), \quad x \in D.$$

*Then, there exists a constant $C > 0$ such that for every $n \in \mathbb{N}$ there exists a ReLU network $\tilde{u}_n(y_1, \ldots, y_n, x)$ with $n+1$ input units and for a number $\mathcal{N}_n \geq n$ with $r = \min\{1, (p_V^{-1} - 1)/(1 + p_V^{-1} - p_X^{-1})\}$ there holds the bound*

$$\sup_{\boldsymbol{y} \in U} \|u(\boldsymbol{y}, \cdot) - \tilde{u}_n(y_1, \ldots, y_n, \cdot)\|_V \leq C \mathcal{N}_n^{-r}.$$

*Moreover, for every $n \in \mathbb{N}$,*

$$\operatorname{size}(\tilde{u}_n) \leq C(1 + \mathcal{N}_n \log(\mathcal{N}_n) \log\log(\mathcal{N}_n)),$$

$$\operatorname{depth}(\tilde{u}_n) \leq C(1 + \log(\mathcal{N}_n) \log\log(\mathcal{N}_n)).\text{"}$$

Theorem 5.1 in Section 5 shows that an analogous result holds on the domain $(0,1)^2$. Possibilities for further extensions are discussed in Section 6.

## 1.4 Notation

### 1.4.1 General notation

We will use the following notation: $\mathbb{N} = \{1, 2, \ldots\}$, $\mathbb{N}_0 = \{0, 1, 2, \ldots\}$. Cartesian powers are denoted by superscripts, e.g. $\mathbb{N}_0^3 = \mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{N}_0$, while certain subsets of $\mathbb{N}_0$ are denoted by subscripts, e.g. $\mathbb{N}_{\geq 2} = \{2, 3, \ldots\} = \mathbb{N}_{>1}$, $\mathbb{N}_{\leq k} = \{1, 2, \ldots, k\}$, $\mathbb{N}_{<k} = \{1, 2, \ldots, k-1\}$ and $(\mathbb{N}_0)^2_{\leq 2^n} = \{0, 1, 2, \ldots, 2^n\} \times \{0, 1, 2, \ldots, 2^n\}$. For any set $\Lambda$, we denote its cardinality by $|\Lambda|$.

For statements, we write

$$\mathbb{1}[\text{statement}] = \begin{cases} 1 & \text{statement is true,} \\ 0 & \text{statement is false.} \end{cases}$$

Vectors, sequences and multiindices are written in boldface, e.g. $\boldsymbol{x} \in \mathbb{R}^2$, $\boldsymbol{y} \in [-1,1]^{\mathbb{N}}$. Numbers in boldface denote sequences or vectors, e.g. $\boldsymbol{1} = (1, \ldots, 1) \in \mathbb{R}^d$ for $d \in \mathbb{N}$. For countable sequences $\boldsymbol{b} = (b_j)_{j \in J}$, we write $|\boldsymbol{b}|_1 := \sum_{j \in J} b_j$. In addition, for all $d, d_1, d_2 \in \mathbb{N}$, we write for vectors in $\mathbb{R}^d$ resp. matrices in $\mathbb{R}^{d_1 \times d_2}$

$$\forall \boldsymbol{b} \in \mathbb{R}^d : \|\boldsymbol{b}\|_{\ell^0} := \sum_{i=1}^{d} \mathbb{1}[b_i \neq 0], \qquad \forall A \in \mathbb{R}^{d_1 \times d_2} : \|A\|_{\ell^0} := \sum_{i=1}^{d_1} \sum_{j=1}^{d_2} \mathbb{1}[A_{i,j} \neq 0].$$

For $d \in \mathbb{N}$, the $d \times d$ identity matrix is denoted by $\mathrm{Id}_{\mathbb{R}^d}$. In addition, for any set $S$, the identity function on $S$ is denoted by $\mathrm{Id}_S$.

We use the componentwise ordering on $(\mathbb{R}_{\geq 0})^{\mathbb{N}}$, i.e. $\boldsymbol{\nu} \leq \boldsymbol{\mu}$ if and only if $\forall j \in \mathbb{N} : \nu_j \leq \mu_j$. In addition, we define $\boldsymbol{\nu} < \boldsymbol{\mu}$ as $(\boldsymbol{\nu} \leq \boldsymbol{\mu}) \wedge (\boldsymbol{\nu} \neq \boldsymbol{\mu})$.

A set $\Lambda \subset \mathbb{N}_0^{\mathbb{N}}$ is *downward closed* if all $\boldsymbol{\mu} \in \mathbb{N}_0^{\mathbb{N}}$ that satisfy $\boldsymbol{\mu} \leq \boldsymbol{\lambda}$ for some $\boldsymbol{\lambda} \in \Lambda$ are also contained in $\Lambda$. For such $\Lambda$, we know that $|\{j \in \mathbb{N} : \exists \boldsymbol{\lambda} \in \Lambda : \lambda_j \neq 0\}| \leq |\Lambda|$, because we have $\boldsymbol{e}_j = (0, \ldots, 0, 1, 0, \ldots) \in \Lambda$ for each $j$ satisfying $(\exists \boldsymbol{\lambda} \in \Lambda : \lambda_j \neq 0)$, where $(\boldsymbol{e}_j)_j = 1$ and $\forall i \in \mathbb{N}, i \neq j : (\boldsymbol{e}_j)_i = 0$.

### 1.4.2 Index set $\mathcal{F}$

We define $\mathcal{F} := \{\boldsymbol{\nu} \in \mathbb{N}_0^{\mathbb{N}} : \nu_j \neq 0$ for only finitely many $j \in \mathbb{N}\}$, i.e. $\mathcal{F}$ is the set of *finitely supported* sequences with values in $\mathbb{N}_0$. For $\boldsymbol{\nu} \in \mathcal{F}$, we define

$$\operatorname{supp} \boldsymbol{\nu} := \{j \in \mathbb{N} : \nu_j \neq 0\}.$$

Note that $\mathbb{N}^{\mathbb{N}}$ is uncountable (the famous result by Cantor, cf. [25, Theorem 6B p. 132]), hence so is $\mathbb{N}_0^{\mathbb{N}}$. However, $\mathcal{F}$ is countable, as we will use in many of our results. It follows from

$$\mathcal{F} = \bigcup_{n \in \mathbb{N}_0} \bigcup_{m \in \mathbb{N}} \left\{ \boldsymbol{\nu} \in \mathbb{N}_0^{\mathbb{N}} : |\boldsymbol{\nu}|_1 = n, \operatorname{supp} \boldsymbol{\nu} \subset \{1, \ldots, m\} \right\},$$

which shows that $\mathcal{F}$ is a countable union of finite sets, hence countable.

For a subset $\Lambda \subset \mathcal{F}$, we denote its complement in $\mathcal{F}$ by $\Lambda^c := \mathcal{F} \backslash \Lambda$. For $\boldsymbol{\nu} \in \mathcal{F}$, we write $\boldsymbol{\nu}! = \prod_{j \in \mathbb{N}} \nu_j!$, using that $0! = 1$. For $\boldsymbol{\nu} \in \mathcal{F}$ and $\boldsymbol{b} \in \mathbb{R}^{\mathbb{N}}$, we define $\boldsymbol{b}^{\boldsymbol{\nu}} := \prod_{j \in \mathbb{N}} b_j^{\nu_j}$, using that $0^0 = 1$. Although these products formally have infinitely many factors, $\boldsymbol{\nu} \in \mathcal{F}$ ensures that only finitely many factors differ from 1.

### 1.4.3 Parameter space $U$

We will study parametric functions with parameters $\boldsymbol{y} = (y_j)_{j \in \mathbb{N}}$ in $U := [-1, 1]^{\mathbb{N}}$. We will always endow $U$ with the product topology. In 1930 Tychonoff showed that the Cartesian product of any cardinality of unit intervals endowed with the product topology is compact Hausdorff ([80, Sections 1 and 2 pp. 546–550]). Second countable compact Hausdorff spaces are metrisable by Urysohn's metrisation theorem ([81, Hauptsatz p. 310]). Hence, $U$ is a compact metrisable space, as is any other countable Cartesian product of compact intervals. An example of a metric inducing the product topology on $[-1, 1]^{\mathbb{N}}$ is

$$d(\boldsymbol{y}, \boldsymbol{y}') = \sum_{j \in \mathbb{N}} 2^{-j} \frac{|y_j - y_j'|}{1 + |y_j - y_j'|}$$

([72, Examples 3.3.20.(b) p. 84]), convergence in the product topology corresponds to componentwise convergence.

### 1.4.4 Parametric functions

For a domain $D \subset \mathbb{R}^d$ with $d \in \mathbb{N}$, we denote points in $D$ by $\boldsymbol{x} \in D$ or by $x \in D$ if $d = 1$. For $i \in \{1, \ldots, d\}$, the $i$'th Euclidean unit vector is denoted by $\boldsymbol{e}_i$.

For a Banach space $V$ of functions defined on $D$, we will study parametric functions $a : U \to V : \boldsymbol{y} \mapsto a(\boldsymbol{y})$. In general, we write $a(\boldsymbol{y}) \in V$, suppressing the spatial variable $\boldsymbol{x} \in D$ in notation. Otherwise, we write $a(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{R}$. Weak and strong partial derivatives with respect to the spatial variable $\boldsymbol{x} \in D$ are both denoted by $\frac{\partial}{\partial x_i} a$ for $i \in \{1, \dots, d\}$ and, more generally, by

$$\forall \boldsymbol{\alpha} \in \mathbb{N}_0^d : D^{\boldsymbol{\alpha}} a := D^{\boldsymbol{\alpha}}_{\boldsymbol{x}} a := \frac{\partial^{|\boldsymbol{\alpha}|_1}}{\partial x_1^{\alpha_1} \partial x_2^{\alpha_2} \cdots \partial x_d^{\alpha_d}} a(\boldsymbol{x}, \boldsymbol{y}).$$

We denote partial derivatives of $a$ with respect to the parameters $\boldsymbol{y} \in U$ by

$$\forall \boldsymbol{\nu} \in \mathcal{F} : \partial^{\boldsymbol{\nu}} a := \partial^{\boldsymbol{\nu}}_{\boldsymbol{y}} a := \frac{\partial^{|\boldsymbol{\nu}|_1}}{\partial y_1^{\nu_1} \partial y_2^{\nu_2} \cdots} a(\boldsymbol{x}, \boldsymbol{y}).$$

## 1.5 Outline

In Section 2, a formalism for non-residual feedforward deep ReLU networks (defined in Sections 1.2.1 and 1.2.2) is introduced and ReLU approximations of products of two or more real numbers are studied. Section 3 introduces ReLU approximations of functions in $H^2 \cap H^1_0((0,1)^2)$, based on square grid continuous, piecewise bilinear interpolation. For the ReLU approximation of the interpolants, the ReLU product networks from Section 2 are used. The main result of Section 3 shows the convergence rate of the ReLU approximations in terms of the ReLU network size. In Section 4, we study a parametric elliptic diffusion equation on the domain $(0,1)^2$ and we derive properties of the corresponding solution map. This section gives sufficient conditions for the Taylor gpc expansion of the solution map to be sparse in the sense that the $H^1$-norms of the Taylor gpc coefficients are $\ell^p$-summable for some $0 < p < 1$. In addition, it is shown that under slightly stronger assumptions the same holds for the $H^2$-norms of the Taylor gpc coefficients, where $p \in (0,1)$ may be larger than for the summability of the $H^1$-norms. Truncations of the Taylor gpc expansion are proposed as approximations of the solution map. The sparsity of the Taylor gpc expansion is used to show lower bounds on the convergence rate of these Taylor gpc approximations. Section 5 applies the ReLU approximation of functions in $H^2 \cap H^1_0((0,1)^2)$ to the coefficients of the Taylor gpc approximations of the solution map. Combined with the efficient ReLU approximation of products shown in Section 2, it follows that the solution map can be approximated efficiently by a family of ReLU networks. This is expressed by a lower bound on the convergence rate in terms of the network size. This convergence rate bound depends on the convergence rate of the ReLU approximations of functions in $H^2 \cap H^1_0((0,1)^2)$ and the summability exponents of the $H^1$- and the $H^2$-norms of the Taylor gpc coefficients. Finally, Section 6 discusses the main result of this thesis (Theorem 5.1) and gives directions for further research. This includes two alternatives for the ReLU approximation of gpc coefficients, two generalisations of the spatial domain $(0,1)^2$ and generalisations of the parametric PDE theory showing sparsity of the gpc expansion.

# 2 ReLU neural network calculus

We introduce a formal description of non-residual feedforward ReLU neural networks in Section 2.1 and discuss the approximation of products by such networks in Sections 2.2 and 2.3. Section 2.2 studies ReLU approximations of products of two factors, whereas Section 2.3 uses such networks for the approximation of products of multiple factors.

## 2.1 Formal description of non-residual feedforward ReLU neural networks

Non-residual FF ReLU NNs have been introduced in Sections 1.2.1–1.2.2. We now introduce a formal description of such networks that is similar to but not equal to the formalisms introduced in [62, Section 2 pp. 6–8] and [24, Section 5 pp. 16–19].

As discussed in Section 1.2.1, most neural networks consist of computational nodes organised in layers. The formalism we use describes the network in terms of layers, the decomposition of layers into nodes is not used explicitly. At the end of Notation 2.2 below, we discuss how this formalism can be rephrased in terms of computational nodes.

We first introduce our assumptions on the network architecture and notation for networks of that architecture in Section 2.1.1. Then, we discuss how such networks can be constructed from multiple subnetworks. Section 2.1.2 mainly contains definitions of such constructions, many of their basic properties are given in Appendix A.

### 2.1.1 Assumptions on the architecture and notation for networks with such architecture

**Assumption 2.1** (Architecture, cf. [24, Setting 5.1 pp. 16–17])**.** *The architecture of networks we study in this thesis is as follows:*

*We study non-residual FF NNs with inputs $x_1, \dots, x_{N_0} \in \mathbb{R}$ for $N_0 \in \mathbb{N}$, also denoted as a vector $\boldsymbol{x} = (x_1, \dots, x_{N_0}) \in \mathbb{R}^{N_0}$. The inputs form the input layer, which does not carry out any computations. It serves as input for the first computational layer. The input layer is followed by $L \in \mathbb{N}$ computational layers. For $l \in \{1, \dots, L\}$, layer $l$ has size $N_l \in \mathbb{N}$, which equals the dimension of the output of that layer.*

*We study networks in which computational layers $1, \dots, L-1$ carry out an affine transformation followed by a non-linear activation. We use the ReLU activation function $\sigma : \mathbb{R} \to \mathbb{R} : x \mapsto \max\{0, x\}$. In addition, we denote*

$$\sigma^* : \bigcup_{d \in \mathbb{N}} \mathbb{R}^d \to \bigcup_{d \in \mathbb{N}} \mathbb{R}^d : (z_1, \dots, z_d) \mapsto (\sigma(z_1), \dots, \sigma(z_d)).$$

*The output layer, which is the last computational layer, carries out an affine transformation not followed by activation.*

*Computational layers that apply the activation function are called hidden layers, i.e. computational layers $1, \dots, L-1$ are hidden layers. The output layer is the only computational layer that is not a hidden layer.*

We now introduce notation for networks whose architecture satisfies Assumption 2.1. In addition to introducing notation for the coefficients, the depth, the size and the number of non-zero coefficients of the network, we also give explicit expressions for the computations carried out by layers of the network. That is, for a network $\Phi$ that satisfies Assumption 2.1 and has $L$ computational layers, for $l \in \{1, \ldots, L\}$, we will give an explicit expression for the function $R_{\sigma,l}(\Phi) : \mathbb{R}^{N_{l-1}} \to \mathbb{R}^{N_l}$ implemented by the $l$'th computational layer of $\Phi$ and for $R_\sigma(\Phi) : \mathbb{R}^{N_0} \to \mathbb{R}^{N_L}$, which denotes the realisation of $\Phi$. For these functions, the subscript $\sigma$ denotes that $\Phi$ has the ReLU activation function.

**Notation 2.2** (Cf. [62, Definition 2.1 p. 6] and [24, Setting 5.1 pp. 16–17])**.** *The coefficients of a network $\Phi$ with the architecture of Assumption 2.1 are for each $l \in \{1, \ldots, L\}$ a $N_l \times N_{l-1}$* weight matrix *$A_l \in \mathbb{R}^{N_l \times \mathbb{N}_{l-1}}$ whose elements are called* weights *and a* bias vector *$\boldsymbol{b}_l \in \mathbb{R}^{N_l}$ whose components are called* biases. *As mentioned in Section 1.2.1, a network is determined by its architecture and its coefficients. Given that a network $\Phi$ has the architecture of Assumption 2.1, we identify it with its coefficients: with*

$$\mathcal{N}_L^{N_0,N_1,\ldots,N_L} := \bigtimes_{l\in\{1,\ldots,L\}} \left(\mathbb{R}^{N_l \times N_{l-1}} \times \mathbb{R}^{N_l}\right),$$

*we write*

$$\Phi = ((A_1, \boldsymbol{b}_1), (A_2, \boldsymbol{b}_2), \ldots, (A_L, \boldsymbol{b}_L)) \in \mathcal{N}_L^{N_0,N_1,\ldots,N_L}. \tag{2.1}$$

*In addition, we define*

$$\mathcal{R} := \bigcup_{L\in\mathbb{N}} \bigcup_{N_0,\ldots,N_L\in\mathbb{N}} \mathcal{N}_L^{N_0,N_1,\ldots,N_L}. \tag{2.2}$$

*As mentioned in Section 1.2.1, neural networks define an input-to-output map called realisation. We now describe the functions implemented by each computational layer and the function implemented by the whole network.*

*For $\Phi$ as in Equation (2.1) and for $l \in \{1, \ldots, L-1\}$, computational layer $l$ implements*

$$R_{\sigma,l}(\Phi) : \mathbb{R}^{N_{l-1}} \to \mathbb{R}^{N_l} : \boldsymbol{z}^{l-1} \mapsto \sigma^*(A_l \boldsymbol{z}^{l-1} + \boldsymbol{b}_l), \tag{2.3}$$

*whereas the output layer implements*

$$R_{\sigma,L}(\Phi) : \mathbb{R}^{N_{L-1}} \to \mathbb{R}^{N_L} : \boldsymbol{z}^{L-1} \mapsto A_L \boldsymbol{z}^{L-1} + \boldsymbol{b}_L. \tag{2.4}$$

*All layers together implement the realisation of $\Phi$*

$$R_\sigma(\Phi) : \mathbb{R}^{N_0} \to \mathbb{R}^{N_L} : \boldsymbol{x} \mapsto \big(R_{\sigma,L}(\Phi) \circ R_{\sigma,L-1}(\Phi) \circ \cdots \circ R_{\sigma,1}(\Phi)\big)(\boldsymbol{x}). \tag{2.5}$$

*The computational layers of a neural network are often thought of as composed of computational nodes (see Section 1.2.1). In terms of Equations (2.3)–(2.4), a computational node outputs one component of the output of a layer, i.e. denoting the elements of $A_l$ by $\{A_{l;j,k}\}_{j\in\{1,\ldots,N_l\},k\in\{1,\ldots,N_{l-1}\}}$ and the components of $\boldsymbol{z}^{l-1}$ and $\boldsymbol{b}_l$ by $\{z_k^{l-1}\}_{k\in\{1,\ldots,N_{l-1}\}}$ and $\{b_{l;j}\}_{j\in\{1,\ldots,N_l\}}$, for $l \in \{1, \ldots, L\}$ and $j \in \{1, \ldots, N_l\}$, the $j$'th node of the $l$'th computational layer computes for $\boldsymbol{z}^{l-1} \in \mathbb{R}^{N_{l-1}}$*

$$\big(R_{\sigma,l}(\Phi)(\boldsymbol{z}^{l-1})\big)_j = \begin{cases} \sigma\left(\sum_{k=1}^{N_{l-1}} A_{l;j,k} z_k^{l-1} + b_{l;j}\right) & l \in \{1, \ldots, L-1\}, \\ \sum_{k=1}^{N_{l-1}} A_{l;j,k} z_k^{l-1} + b_{l;j} & l = L. \end{cases} \tag{2.6}$$

*The number of ReLUs in the network equals the number of nodes in the hidden layers: each such node applies the ReLU activation function once, nodes in other layers do not contain ReLUs.*

*Finally, we will use the following definitions:*

$$\operatorname{depth}(\Phi) := L, \qquad \operatorname{size}(\Phi) := \sum_{l=1}^{L-1} N_l, \qquad \mathcal{M}(\Phi) := \sum_{l=1}^{L} (\|A_l\|_{\ell^0} + \|\boldsymbol{b}_l\|_{\ell^0}),$$

*i.e.* $\operatorname{depth}(\Phi)$ *denotes the number of computational layers of* $\Phi$*,* $\operatorname{size}(\Phi)$ *the number of ReLUs and* $\mathcal{M}(\Phi)$ *the number of non-zero coefficients.*

The following lemma holds, cf. [77, Lemmas 2.1 p. 2 and 2.3 p. 3] for non-residual FF ReLU NNs with input size and output size equal to one and [55, Proposition 4 p. 6] for non-residual FF ReLU NNs with arbitrary input and output size. The statements in [60, 55] are more precise, for us the result below suffices.

**Lemma 2.3** (Cf. [55, Proposition 4 p. 6]). *Every function $f : \mathbb{R}^{N_0} \to \mathbb{R}$ that can be implemented by a ReLU network as described in Assumption 2.1 and Notation 2.2 has the following property: there exists a family $\mathcal{T} = \{T_j\}_{j\in\{1,\dots,J\}}$ of finitely many convex open sets such that $\forall j \in \{1,\dots,J\} : T_j \subset \mathbb{R}^{N_0}$, $\mathbb{R}^{N_0} = \bigcup_{j\in\{1,\dots,J\}} \overline{T_j}$ and such that $f$ is linear on $T_j$ for each $j \in \{1,\dots,J\}$.*

The idea of the proof is that of [60, Lemma 1 pp. 3–4], which gives the result for one hidden layer. The proof of Lemma 2.3 is given in Appendix B.1.

We now define networks with architecture as in Assumption 2.1 that implement the identity operator. We will use them to increase the depth of a network without changing the realisation (Remark A.2). This is sometimes needed when constructing networks from subnetworks of unequal depth.

**Definition 2.4** ([62, Lemma 2.3 and Remark 2.4 p. 7] and [24, Setting 5.2 pp. 17–18]). *Using Notation 2.2, for $d \in \mathbb{N}$ and $L \in \mathbb{N}_{\geq 2}$, we define the* identity network $\Phi^{\mathrm{Id}}_{d,L} \in \mathcal{N}_L^{d,2d,\dots,2d,d}$ *with architecture satisfying Assumption 2.1 as*

$$\Phi^{\mathrm{Id}}_{d,L} := \left( \left( \begin{pmatrix} \mathrm{Id}_{\mathbb{R}^d} \\ -\mathrm{Id}_{\mathbb{R}^d} \end{pmatrix}, \mathbf{0} \right), (\mathrm{Id}_{\mathbb{R}^{2d}}, \mathbf{0}), \dots, (\mathrm{Id}_{\mathbb{R}^{2d}}, \mathbf{0}), \left( \begin{pmatrix} \mathrm{Id}_{\mathbb{R}^d} & -\mathrm{Id}_{\mathbb{R}^d} \end{pmatrix}, \mathbf{0} \right) \right). \tag{2.7}$$

*For $d \in \mathbb{N}$ and $L = 1$, we can implement the identity operator by the* identity network $\Phi^{\mathrm{Id}}_{d,1} := ((\mathrm{Id}_d, \mathbf{0})) \in \mathcal{N}_1^{d,d}$*.*

**Remark 2.5** ([62, Lemma 2.3 and Remark 2.4 p. 7], see also [24, Setting 5.2 pp. 17–18]). *For $d \in \mathbb{N}$, $L \in \mathbb{N}_{\geq 2}$ and $M \in \mathbb{R}_{>0}$, the networks from Definition 2.4 satisfy*

$$\operatorname{depth}(\Phi^{\mathrm{Id}}_{d,L}) = L, \qquad \operatorname{size}(\Phi^{\mathrm{Id}}_{d,L}) = 2(L-1)d, \qquad \mathcal{M}(\Phi^{\mathrm{Id}}_{d,L}) = 2Ld, \tag{2.8}$$

$$\operatorname{depth}(\Phi^{\mathrm{Id}}_{d,1}) = 1, \qquad \operatorname{size}(\Phi^{\mathrm{Id}}_{d,1}) = 0, \qquad \mathcal{M}(\Phi^{\mathrm{Id}}_{d,1}) = d. \tag{2.9}$$

*With $\sigma$ the ReLU as in Assumption 2.1, it follows from $\forall y \in \mathbb{R} : y = \sigma(y) - \sigma(-y)$ that* *for $d \in \mathbb{N}$ and $L \in \mathbb{N}_{\geq 2}$*

$$\forall \boldsymbol{x} \in \mathbb{R}^d : R_\sigma(\Phi^{\mathrm{Id}}_{d,L})(\boldsymbol{x}) = (\sigma^*)^{\circ L-1}(\boldsymbol{x}) - (\sigma^*)^{\circ L-1}(-\boldsymbol{x}) = \boldsymbol{x},$$

*i.e. $R_\sigma(\Phi^{\mathrm{Id}}_{d,L}) = \mathrm{Id}_{\mathbb{R}^d}$. In addition, $R_\sigma(\Phi^{\mathrm{Id}}_{d,1}) = \mathrm{Id}_{\mathbb{R}^d}$.*

As was mentioned in Section 1.2.1, for each function that can be implemented by a ReLU network, there exist multiple networks that do so. We mentioned two relatively straightforward types of network modifications that do not affect the realisation. In the following example we show a simple but much less trivial example of two networks with the same realisation.

**Example 2.6.** *In this example we discuss ReLU networks that satisfy Assumption 2.1. We use Notation 2.2 to denote them.*

*The* standard hat function

$$g(x) = \begin{cases} 2x & 0 \leq x \leq \frac{1}{2}, \\ 2(1-x) & \frac{1}{2} < x \leq 1, \\ 0 & x < 0 \text{ and } x > 1 \end{cases} \tag{2.10}$$

*can be implemented by the depth two network*

$$\Phi^1 := \left( \left( \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ -\frac{1}{2} \\ -1 \end{pmatrix} \right), \left( \begin{pmatrix} 2 & -4 & 2 \end{pmatrix}, (0) \right) \right), \tag{2.11}$$

*as it computes $\forall x \in \mathbb{R} : R_\sigma(\Phi^1)(x) = 2\sigma(x) - 4\sigma(x - \frac{1}{2}) + 2\sigma(x-1) = g(x)$.*

*Alternatively, the depth three network*

$$\Phi^2 := \left( \left( \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \begin{pmatrix} -\frac{1}{2} \\ \frac{1}{2} \end{pmatrix} \right), \left( \begin{pmatrix} -2 & -2 \end{pmatrix}, \begin{pmatrix} 1 \end{pmatrix} \right), \left( \begin{pmatrix} 1 \end{pmatrix}, \begin{pmatrix} 0 \end{pmatrix} \right) \right) \tag{2.12}$$

*has realisation $\forall x \in \mathbb{R} : R_\sigma(\Phi^2)(x) = \sigma\big(1 - 2\sigma(x - \frac{1}{2}) - 2\sigma(\frac{1}{2} - x)\big) = g(x)$.*

*Note that there is no natural way to find the coefficients of $\Phi^2$ from those of $\Phi^1$ or vice versa.*

### 2.1.2 Construction of networks from subnetworks

We will now discuss how multiple subnetworks can be combined into one network. This includes the concatenation of subnetworks, implementing the composition of realisations, and the parallelisation of multiple subnetworks. This section mainly contains definitions, basic properties of the defined networks are given in Appendix A.

In this section we use Assumption 2.1 and Notation 2.2 for all networks.

**Definition 2.7** ([62, Definition 2.2 p. 6])**.** *For $i \in \{1,2\}$, let $L^i \in \mathbb{N}$, $N_0^i, N_1^i, \dots, N_{L^i}^i \in \mathbb{N}$ and*

$$\Phi^i = ((A_1^i, \boldsymbol{b}_1^i), \dots, (A_{L^i}^i, \boldsymbol{b}_{L^i}^i)) \in \mathcal{N}_{L^i}^{N_0^i, N_1^i, \dots, N_{L^i}^i}$$

*be arbitrary, but such that $N_{L^2}^2 = N_0^1$. We define the* concatenation $\Phi^1 \bullet \Phi^2 \in \mathcal{N}_{L^1+L^2-1}^{N_0^2, N_1^2, \dots, N_{L^2-1}^2, N_1^1, \dots, N_{L^1}^1}$ *as*

$$\Phi^1 \bullet \Phi^2 := \Big((A_1^2, \boldsymbol{b}_1^2), \dots, (A_{L^2-1}^2, \boldsymbol{b}_{L^2-1}^2), (A_1^1 A_{L^2}^2, A_1^1 \boldsymbol{b}_{L^2}^2 + \boldsymbol{b}_1^1), (A_2^1, \boldsymbol{b}_2^1), \dots, (A_{L^1}^1, \boldsymbol{b}_{L^1}^1)\Big). \tag{2.13}$$

**Lemma 2.8** ([62, p. 6, below Definition 2.2])**.** *For $\Phi^1$ and $\Phi^2$ as in Definition 2.7, it holds that*

$$R_\sigma(\Phi^1 \bullet \Phi^2) = R_\sigma(\Phi^1) \circ R_\sigma(\Phi^2).$$

*Proof.* It follows from Equations (2.3)–(2.4) that

$$\begin{aligned} \forall \boldsymbol{x} \in \mathbb{R}^{N_{L^2-1}^2} : \big(R_{\sigma,1}(\Phi^1) \circ R_{\sigma,L^2}(\Phi^2)\big)(\boldsymbol{x}) &= \sigma^*\big(A_1^1(A_{L^2}^2 \boldsymbol{x} + \boldsymbol{b}_{L^2}^2) + \boldsymbol{b}_1^1\big) \\ &= \sigma^*\big(A_1^1 A_{L^2}^2 \boldsymbol{x} + A_1^1 \boldsymbol{b}_{L^2}^2 + \boldsymbol{b}_1^1\big) \\ &= R_{\sigma,L^2}(\Phi^1 \bullet \Phi^2)(\boldsymbol{x}). \end{aligned}$$

The lemma now follows from the fact that

$$\begin{aligned} \forall l \in \{1, \dots, L^2-1\} &: R_{\sigma,l}(\Phi^1 \bullet \Phi^2) = R_{\sigma,l}(\Phi^2), \\ \forall l \in \{2, \dots, L^1\} &: R_{\sigma,L^2-1+l}(\Phi^1 \bullet \Phi^2) = R_{\sigma,l}(\Phi^1). \end{aligned}$$

□

More properties of concatenations are given in Remark A.1. In Remark A.2, we discuss the concatenation of a network with an identity network from Definition 2.4 in order to increase the depth of that network without changing its realisation. We use such extensions for the definition of parallelisations, see Definitions 2.10 and 2.11 below.

For $\Phi^1$ and $\Phi^2$ as in Definition 2.7, Equation (A.4) from Remark A.1 provides an upper bound on $\mathcal{M}(\Phi^1 \bullet \Phi^2)$ containing the term $\|A_1^1\|_{\ell^0} \|A_{L^2}^2\|_{\ell^0}$. In certain situations that term may be very large. Therefore, we now define a form of concatenation that has an upper bound on the number of non-zero coefficients that does not contain such products, see Equation (A.14) from Remark A.3.

**Definition 2.9** ([62, Definition 2.5 p. 7], see also [24, Setting 5.2 pp. 17–18])**.** *For $i \in \{1,2\}$, let $L^i \in \mathbb{N}$, $N_0^i, N_1^i, \dots, N_{L^i}^i \in \mathbb{N}$ and*

$$\Phi^i = ((A_1^i, \boldsymbol{b}_1^i), \dots, (A_{L^i}^i, \boldsymbol{b}_{L^i}^i)) \in \mathcal{N}_{L^i}^{N_0^i, N_1^i, \dots, N_{L^i}^i}$$

*be arbitrary, but such that $N_{L^2}^2 = N_0^1$. We define the* sparse concatenation *of $\Phi^1$ and $\Phi^2$ as*

$$\Phi^1 \odot \Phi^2 := \Phi^1 \bullet \Phi_{N_{L^2}^2,2}^{\mathrm{Id}} \bullet \Phi^2 \in \mathcal{N}_{L^1+L^2}^{N_0^2, N_1^2, \dots, N_{L^2-1}^2, 2N_{L^2}^2, N_1^1, \dots, N_{L^1}^1}. \tag{2.14}$$

Remark A.3 discusses some properties of sparse concatenations.

We now introduce parallelisations of multiple subnetworks, i.e. networks that in parallel implement the realisations of those subnetworks. For the parallelisation of subnetworks that have unequal depth, we use identity networks (Definition 2.4) to extend the subnetworks of less than maximum depth. Such extensions are discussed in Remark A.2. As in [62, Remark 2.8 p. 7], we have chosen to extend the inputs of subnetworks of less than maximum depth. Equivalently, parallelisations can be defined using extensions of the outputs of those subnetworks, cf. [24, Setting 5.2 pp. 17–18].

In Definition 2.10 below, we define the parallelisation of subnetworks that share the same inputs. As discussed in Remark A.5, the fact that the same inputs are shared by all subnetworks can be exploited to reduce the number of ReLUs needed to implement the extension of the inputs.

Besides that, parallelisations of subnetworks that do not share the same inputs are defined in Definition 2.11 below.

**Definition 2.10** ([62, Definition 2.7 and Remark 2.8 p. 7], cf. [24, Setting 5.2 pp. 17–18])**.** *Let $N, N_0 \in \mathbb{N}$. For $i \in \{1, \dots, N\}$, let $L^i \in \mathbb{N}$, $N_1^i, \dots, N_{L^i}^i \in \mathbb{N}$ and*

$$\Phi^i = ((A_1^i, \boldsymbol{b}_1^i), \dots, (A_{L^i}^i, \boldsymbol{b}_{L^i}^i)) \in \mathcal{N}_{L^i}^{N_0, N_1^i, \dots, N_{L^i}^i}$$

*be arbitrary, but such that $\{\Phi^i\}_{i\in\{1,\dots,N\}}$ all share the same input in $\mathbb{R}^{N_0}$.*

*Let $L^{\max} := \max_{i\in\{1,\dots,N\}} L^i$. For $i \in \{1, \dots, N\}$, let $\forall l \in \{1, \dots, L^{\max}\} : \tilde{N}_l^i \in \mathbb{N}$ be such that*

$$\Phi^i \bullet \Phi^{\mathrm{Id}}_{N_0, L^{\max}-L^i+1} \in \mathcal{N}_{L^{\max}}^{N_0, \tilde{N}_1^i, \dots, \tilde{N}_{L^{\max}}^i}$$

*and let $\{\tilde{A}_l^i\}_{i\in\{1,\dots,N\}, l\in\{1,\dots,L^{\max}\}}$ and $\{\tilde{\boldsymbol{b}}_l^i\}_{i\in\{1,\dots,N\}, l\in\{1,\dots,L^{\max}\}}$ be such that*

$$\Phi^i \bullet \Phi^{\mathrm{Id}}_{N_0, L^{\max}-L^i+1} = ((\tilde{A}_1^i, \tilde{\boldsymbol{b}}_1^i), \dots, (\tilde{A}_{L^{\max}}^i, \tilde{\boldsymbol{b}}_{L^{\max}}^i)). \tag{2.15}$$

*For $l \in \{1, \dots, L^{\max}\}$, let $\tilde{N}_l^{\mathrm{tot}} := \sum_{i\in\{1,\dots,N\}} \tilde{N}_l^i$.*

*Then, the* parallelisation for shared inputs Shared-Parallel$(\Phi^1, \dots, \Phi^N) \in \mathcal{N}_{L^{\max}}^{N_0, \tilde{N}_1^{\mathrm{tot}}, \dots, \tilde{N}_{L^{\max}}^{\mathrm{tot}}}$ *is defined as*

$$\text{Shared-Parallel}(\Phi^1, \dots, \Phi^N) := \left( \left( \begin{pmatrix} \tilde{A}_1^1 \\ \vdots \\ \tilde{A}_1^N \end{pmatrix}, \begin{pmatrix} \tilde{\boldsymbol{b}}_1^1 \\ \vdots \\ \tilde{\boldsymbol{b}}_1^N \end{pmatrix} \right), \right.$$

$$\left. \left( \begin{pmatrix} \tilde{A}_2^1 & & \\ & \ddots & \\ & & \tilde{A}_2^N \end{pmatrix}, \begin{pmatrix} \tilde{\boldsymbol{b}}_2^1 \\ \vdots \\ \tilde{\boldsymbol{b}}_2^N \end{pmatrix} \right), \dots, \left( \begin{pmatrix} \tilde{A}_{L^{\max}}^1 & & \\ & \ddots & \\ & & \tilde{A}_{L^{\max}}^N \end{pmatrix}, \begin{pmatrix} \tilde{\boldsymbol{b}}_{L^{\max}}^1 \\ \vdots \\ \tilde{\boldsymbol{b}}_{L^{\max}}^N \end{pmatrix} \right) \right). \tag{2.16}$$

Properties of parallelisations for shared inputs are given in Remark A.4. In short, if all subnetworks have equal depth, then the parallelisation for shared inputs also has that depth. In addition, in that case, its size and its number of non-zero coefficients equal the sum of the sizes resp. the number of non-zero coefficients of the subnetworks. If the subnetworks are not of equal depth, then the depth of the parallelisation for shared inputs equals the maximum depth, whereas its size and its number of non-zero coefficients are larger than the sums of those quantities for the subnetworks, as the identity networks add extra ReLUs.

A potentially smaller network with the same realisation is proposed in Remark A.5.

We now define the parallelisation of subnetworks that do not share the same inputs.

**Definition 2.11** ([24, Setting 5.2 pp. 17–18], cf. [62, Definition 2.7 and Remark 2.8 p. 7])**.** *For $N \in \mathbb{N}$ and for $i \in \{1, \dots, N\}$, let $L^i \in \mathbb{N}$, $N_0^i, N_1^i, \dots, N_{L^i}^i \in \mathbb{N}$ and*

$$\Phi^i = ((A_1^i, \boldsymbol{b}_1^i), \dots, (A_{L^i}^i, \boldsymbol{b}_{L^i}^i)) \in \mathcal{N}_{L^i}^{N_0^i, N_1^i, \dots, N_{L^i}^i}$$

*be arbitrary.*

*Let $L^{\max} := \max_{i \in \{1,\dots,N\}} L^i$. For $i \in \{1, \dots, N\}$, let $\forall l \in \{0, 1, \dots, L^{\max}\} : \tilde{N}_l^i \in \mathbb{N}$ be such that*

$$\Phi^i \bullet \Phi^{\mathrm{Id}}_{N_0^i, L^{\max} - L^i + 1} \in \mathcal{N}_{L^{\max}}^{\tilde{N}_0^i, \tilde{N}_1^i, \dots, \tilde{N}_{L^{\max}}^i}$$

*and let $\{\tilde{A}_l^i\}_{i \in \{1,\dots,N\}, l \in \{1,\dots,L^{\max}\}}$ and $\{\tilde{\boldsymbol{b}}_l^i\}_{i \in \{1,\dots,N\}, l \in \{1,\dots,L^{\max}\}}$ be such that*

$$\Phi^i \bullet \Phi^{\mathrm{Id}}_{N_0^i, L^{\max} - L^i + 1} = ((\tilde{A}_1^i, \tilde{\boldsymbol{b}}_1^i), \dots, (\tilde{A}_{L^{\max}}^i, \tilde{\boldsymbol{b}}_{L^{\max}}^i)). \tag{2.17}$$

*For $l \in \{0, 1, \dots, L^{\max}\}$, let $\tilde{N}_l^{\mathrm{tot}} := \sum_{i \in \{1,\dots,N\}} \tilde{N}_l^i$.*

*Then, the* parallelisation $\mathrm{Parallel}(\Phi^1, \dots, \Phi^N) \in \mathcal{N}_{L^{\max}}^{\tilde{N}_0^{\mathrm{tot}}, \tilde{N}_1^{\mathrm{tot}}, \dots, \tilde{N}_{L^{\max}}^{\mathrm{tot}}}$ *is defined as*

$$\mathrm{Parallel}(\Phi^1, \dots, \Phi^N) := \left( \left( \begin{pmatrix} \tilde{A}_1^1 & & \\ & \ddots & \\ & & \tilde{A}_1^N \end{pmatrix}, \begin{pmatrix} \tilde{\boldsymbol{b}}_1^1 \\ \vdots \\ \tilde{\boldsymbol{b}}_1^N \end{pmatrix} \right), \right.$$
$$\left. \left( \begin{pmatrix} \tilde{A}_2^1 & & \\ & \ddots & \\ & & \tilde{A}_2^N \end{pmatrix}, \begin{pmatrix} \tilde{\boldsymbol{b}}_2^1 \\ \vdots \\ \tilde{\boldsymbol{b}}_2^N \end{pmatrix} \right), \dots, \left( \begin{pmatrix} \tilde{A}_{L^{\max}}^1 & & \\ & \ddots & \\ & & \tilde{A}_{L^{\max}}^N \end{pmatrix}, \begin{pmatrix} \tilde{\boldsymbol{b}}_{L^{\max}}^1 \\ \vdots \\ \tilde{\boldsymbol{b}}_{L^{\max}}^N \end{pmatrix} \right) \right). \tag{2.18}$$

Properties of parallelisations of subnetworks that do not share inputs are given in Remark A.6. In short, as for parallelisations for shared inputs, the depth of a parallelisation of subnetworks that do not share inputs equals the maximum of the depths of the subnetworks. The size and the number of non-zero coefficients equal the sums of the respective quantities for the subnetworks if all subnetworks are of equal depth. The size and the number of non-zero coefficients exceed those sums if the subnetworks do not have equal depth.

## 2.2 ReLU DNN approximation of products

In this section and in the next section we discuss the approximation of products by deep ReLU networks satisfying Assumption 2.1. We use Notation 2.2 and follow the discussion in [75, Section 3.2 pp. 9–14], which is based on [89, Section 3.1 pp. 105–106]. The main result of this section is the following proposition on the approximation of products of two factors by ReLU networks, adapted from [75, Proposition 3.1 p. 11]. The main difference with respect to that result is that we show that, under the assumptions made below, the image of $[-M,M]^2$ under $R_\sigma(\tilde{\times})$ is contained in $[-M^2,M^2]$.

**Proposition 2.12** ([75, Proposition 3.1 p. 11] and [89, Proposition 3 p. 106]). *For $M>0$ and $0<\delta<\min\{M,M^2\}$, there exists a ReLU network $\tilde{\times}$ with two input values whose realisation satisfies $R_\sigma(\tilde{\times})\big|_{[-M,M]^2} : [-M,M]^2 \to [-M^2,M^2]$ and*

$$\sup_{a,b\in[-M,M]} |a\cdot b - R_\sigma(\tilde{\times})(a,b)| \le \delta, \tag{2.19}$$

$$\operatorname*{ess\,sup}_{(a,b)\in[-M,M]^2} \max\left\{\left|b-\frac{d}{da}R_\sigma(\tilde{\times})(a,b)\right|,\left|a-\frac{d}{db}R_\sigma(\tilde{\times})(a,b)\right|\right\} \le \delta. \tag{2.20}$$

*Here $\frac{d}{da}R_\sigma(\tilde{\times})(a,b)$ and $\frac{d}{db}R_\sigma(\tilde{\times})(a,b)$ denote weak derivatives. For all $b\in[-M,M]$, there exists a finite set $\mathcal{N}_b\subset(-M,M)$ such that $a\mapsto R_\sigma(\tilde{\times})(a,b)$ is strongly differentiable at all $a\in(-M,M)\backslash\mathcal{N}_b$. In addition, $\tilde{\times}$ has the* zero-in-zero-out property, *i.e.* $[a=0\vee b=0]\Rightarrow R_\sigma(\tilde{\times})(a,b)=0$.

*In total, the network comprises $\mathcal{O}(\log(M)+\log(1/\delta))$ layers, ReLUs and non-zero coefficients.*

The main idea behind the proposition is that squares can be approximated efficiently by deep ReLU networks ([89, Proposition 2 p. 105]), which is exploited by using a polarisation argument to write the product $[-M,M]^2\ni(a,b)\mapsto ab\in\mathbb{R}$ as a linear combination of squares. We use that for all $a,b\in[-M,M]$

$$ab = 2M^2\left[\left(\frac{|a+b|}{2M}\right)^2-\left(\frac{|a|}{2M}\right)^2-\left(\frac{|b|}{2M}\right)^2\right]. \tag{2.21}$$

Note that $\frac{|a+b|}{2M},\frac{|a|}{2M},\frac{|b|}{2M}\in[0,1]$. As in [89, Proposition 2 p. 105], for $m\in\mathbb{N}$, we approximate $[0,1]\to[0,1]:x\mapsto x^2$ by continuous, piecewise linear interpolation on the equispaced partition $\mathcal{T}^1_{2^m}$ of $[0,1]$ into $2^m$ intervals of size $2^{-m}$. The nodes of $\mathcal{T}^1_{2^m}$ are $\{x^1_k := k2^{-m} : k\in\{0,\dots,2^m\}\}$. The continuous, piecewise linear interpolant of $x\mapsto x^2$ on $\mathcal{T}^1_{2^m}$ is denoted by $f_m$. In Step 3 of the proof of Proposition 2.12, we construct a ReLU DNN that implements $f_m$.

Note that $f_m:[0,1]\to[0,1]$, $0\mapsto0$, $\frac12\mapsto\frac14$, $1\mapsto1$, that by convexity of $x\mapsto x^2$ $\forall x\in[0,1]: f_m(x)\ge x^2$, that $f_m$ is strictly increasing on $[0,1]$ and that

$$\|x^2-f_m(x)\|_{L^\infty[0,1]} = 2^{-2m-2}, \tag{2.22}$$

$$\begin{aligned}\|2x - f_m'(x)\|_{L^\infty[0,1]} &= \operatorname*{ess\,sup}_{x\in[0,1]} |2x - f_m'(x)| \\ &= \sup_{j\in\{1,\dots,2^m\}} \sup_{x\in[x_{j-1}^1, x_j^1]} \left|2x - f_m'(x)\right| \\ &= \sup_{j\in\{1,\dots,2^m\}} \sup_{x\in[x_{j-1}^1, x_j^1]} \left|2x - \tfrac{(x_j^1)^2-(x_{j-1}^1)^2}{x_j^1 - x_{j-1}^1}\right| \\ &= \sup_{j\in\{1,\dots,2^m\}} \sup_{x\in[x_{j-1}^1, x_j^1]} \left|2x - (x_j^1 + x_{j-1}^1)\right| \\ &= 2^{-m}. \end{aligned} \tag{2.23}$$

The properties of $f_m$ are discussed in more detail in Step 5 of the proof of Proposition 2.12 in Appendix B.2.

The function $f_m$ can be constructed efficiently from the standard hat function

$$g(x) = \begin{cases} 2x & 0 \le x \le \frac{1}{2}, \\ 2(1-x) & \frac{1}{2} < x \le 1, \\ 0 & x < 0 \text{ and } x > 1 \end{cases} \tag{2.24}$$

$$= 2\sigma(x) - 4\sigma(x - \tfrac{1}{2}) + 2\sigma(x-1), \qquad x \in \mathbb{R} \tag{2.25}$$

and *sawtooth functions* that are constructed as powers of $g$ under function composition:

$$\forall m \in \mathbb{N}: \quad g_m := g^{\circ m} = g \circ \cdots \circ g \tag{2.26}$$

(see also [77, Lemma 2.4 p. 4]). Note that for $x \in [0,1]$ we can leave out the last term in Equation (2.25), because $\sigma(x-1) \equiv 0$ for $x \le 1$. In terms of these sawtooth functions, the interpolants $\{f_m\}_{m\in\mathbb{N}}$ satisfy the following recurrence:

$$\begin{aligned} f_0(x) &= x = \sigma(x), && x \in [0,1], \\ \forall m \in \mathbb{N}: \quad f_m(x) &= f_{m-1}(x) - 2^{-2m} g_m(x), && x \in [0,1]. \end{aligned} \tag{2.27}$$

We now use this recurrence to prove Proposition 2.12.

*Proof of Proposition 2.12.* The proof follows [75, pp. 10–11], which is based on [89, Section 3.1 pp. 105–106], and consists of five steps. In the first step, given for all $m \in \mathbb{N}$ a ReLU network $F_m \in \mathcal{R}$ implementing $f_m$, an approximation $\tilde{\times}_m$ of the product $[-M,M]^2 \ni (a,b) \mapsto ab$ is constructed. It is shown that Equations (2.19) and (2.20) hold for appropriately chosen $m$. We define $\tilde{\times} := \tilde{\times}_m$ for an appropriate value of $m$. In the second step, it is discussed that for each $b \in [-M,M]$ we have strong differentiability of $a \mapsto R_\sigma(\tilde{\times})(a,b)$ at all but finitely many $a \in [-M,M]$ and that $R_\sigma(\tilde{\times})$ has the zero-in-zero-out property.

The last three steps are given in Appendix B.2. In the third step, for each $m \in \mathbb{N}$ a ReLU network $F_m$ that implements $f_m$ is constructed. In the fourth step, bounds on the depth, the size and the number of non-zero coefficients of $\tilde{\times}$ are given. In the fifth step, it is shown that $\forall (a,b) \in [-M,M]^2 : |R_\sigma(\tilde{\times})(a,b)| \le M^2$.

*Step 1.* Assume that for all $m \in \mathbb{N}$ a network $F_m \in \mathcal{R}$ that implements $f_m$ is given. For $m \in \mathbb{N}$ and $M > 0$, based on Equation (2.21), we define $\tilde{\times}_m \in \mathcal{R}$ such that it implements

$$[-M,M]^2 \ni (a,b) \mapsto 2M^2 f_m\left(\frac{|a+b|}{2M}\right) - 2M^2 f_m\left(\frac{|a|}{2M}\right) - 2M^2 f_m\left(\frac{|b|}{2M}\right). \tag{2.28}$$

The structure of $\tilde{\times}_m$ is depicted in Figure 2.1.

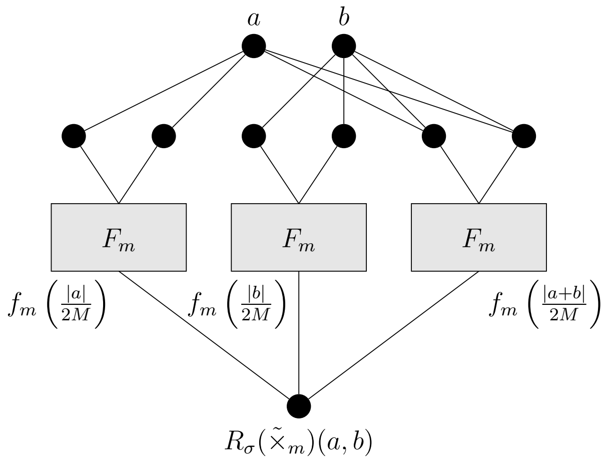

$a$
$b$
$F_m$
$F_m$
$F_m$
$f_m\left(\frac{|a|}{2M}\right)$
$f_m\left(\frac{|b|}{2M}\right)$
$f_m\left(\frac{|a+b|}{2M}\right)$
$R_\sigma(\tilde{\times}_m)(a,b)$

Figure 2.1: Structure of $\tilde{\times}_m$.

We define

$$\tilde{\times}_m := \left(\begin{pmatrix} -2M^2 & -2M^2 & 2M^2 \end{pmatrix}, \mathbf{0}\right) \bullet \mathrm{Parallel}(F_m, F_m, F_m) \bullet \Phi_{\tilde{\times}\mathrm{in}}, \tag{2.29}$$

where

$$\Phi_{\tilde{\times}\mathrm{in}} := \left(\left(\begin{pmatrix} 1 & \\ -1 & \\ & 1 \\ & -1 \\ 1 & 1 \\ -1 & -1 \end{pmatrix}, \mathbf{0}\right), \left(\frac{1}{2M}\begin{pmatrix} 1 & 1 & & & & \\ & & 1 & 1 & & \\ & & & & 1 & 1 \end{pmatrix}, \mathbf{0}\right)\right), \tag{2.30}$$

$$R_\sigma(\Phi_{\tilde{\times}\mathrm{in}}) : \mathbb{R}^2 \to \mathbb{R}^3 : (a,b) \mapsto \left(\tfrac{|a|}{2M}, \tfrac{|b|}{2M}, \tfrac{|a+b|}{2M}\right). \tag{2.31}$$

Equation (2.31) follows directly from Equations (2.3)–(2.4). Equation (2.28) follows from Equation (2.29), Lemma 2.8 and Equations (2.31) and (A.23).

We now determine a value of $m \in \mathbb{N}$ for which Equations (2.19)–(2.20) hold for $\tilde{\times}_m$.

From Equation (2.21), we see that $\tilde{\times}_m$ approximates the product $ab$ with absolute error less than $\delta$, as in Equation (2.19), if the squares are approximated with absolute error less than $\delta/(6M^2)$. Using Equation (2.22), we find that Equation (2.19) is satisfied for $\tilde{\times}_m$ when $2^{-2m-2} \leq \delta/(6M^2)$, which is satisfied when

$$2m \geq 2\log_2 M + \log_2(1/\delta) + 1. \tag{2.32}$$

We next derive a sufficient condition on $m$ such that Equation (2.20) holds for $\tilde{\times}_m$. Using Equation (2.23) twice, we find for all $(a,b) \in [0,M]^2$ satisfying $\frac{a}{2M}, \frac{b}{2M}, \frac{a+b}{2M} \notin \{x^1_j : j \in \{0,\ldots,2^m\}\}$ that

$$\left| b - \frac{d}{da} R_\sigma(\tilde{\times}_m)(a,b) \right| = M \left| \tfrac{2a+2b}{2M} - \tfrac{2a}{2M} - \left(f'_m\left(\tfrac{a+b}{2M}\right) - f'_m\left(\tfrac{a}{2M}\right)\right) \right| \leq 2M2^{-m}, \tag{2.33}$$

which is bounded by $\delta$ for

$$m = \lceil \log_2(2M) + \log(1/\delta) \rceil. \tag{2.34}$$

For that value of $m$, using the symmetry in $a$ and $b$, it follows from Equation (2.33) that Equation (2.20) is satisfied for $\tilde{\times}_m$ when the essential supremum is restricted to $(a,b) \in [0,M]^2$, because $\frac{a}{2M}, \frac{b}{2M}, \frac{a+b}{2M} \notin \{x_j^1 : j \in \{0,\dots,2^m\}\}$ holds for almost every $(a,b) \in [0,M]^2$. The analogous result holds when the essential supremum is restricted to $(a,b) \in [-M,0] \times [0,M]$, $(a,b) \in [0,M] \times [-M,0]$ or $(a,b) \in [-M,0]^2$. This finishes the proof of Equation (2.20) for $\tilde{\times}_m$ with $m$ as in Equation (2.34). For that value of $m$, Equation (2.32) is also satisfied, i.e. we have shown Equations (2.19)–(2.20) for $\tilde{\times}_m$. Therefore, we define

$$\tilde{\times} := \tilde{\times}_{\lceil \log_2(2M) + \log(1/\delta) \rceil}. \tag{2.35}$$

*Step 2.* For $b \in [-M,M]$, the claimed differentiability of $a \mapsto R_\sigma(\tilde{\times})(a,b)$ follows from Lemma 2.3 using that $a \mapsto R_\sigma(\tilde{\times})(a,b)$ can be implemented by

$$\tilde{\times} \bullet \left( \left( \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ b \end{pmatrix} \right) \right).$$

More explicitly, the piecewise linearity of $f_m$ on $[x_{k-1}^1, x_k^1]$ for $k \in \{1,\dots,2^m\}$ shows that

$$\mathcal{N}_b = \left\{ a \in (-M,M) : \left\{ \left|\tfrac{a}{2M}\right|, \left|\tfrac{a+b}{2M}\right| \right\} \cap \{x_j^1 : j \in \{0,\dots,2^m\}\} \neq \emptyset \right\}.$$

The zero-in-zero-out property ([89, Proposition 3(b) p. 106]) directly follows from Equation (2.28) and the fact that $f_m$ has the zero-in-zero-out property, which follows from the fact that the interpolant $f_m$ of $x \mapsto x^2$ is exact in the node $x_0^1 = 0$. This finishes Step 2 of the proof.

Steps 3–5 are given in Appendix B.2. □

**Remark 2.13** (Cf. [24, Lemma 6.2 p. 20])**.** *Note that for $\delta \geq M^2$ Equation (2.32) holds for all $m \in \mathbb{N}$ and that for $\delta \geq M$ the right-hand side of Equation (2.34) is bounded from above by $m$ for all $m \in \mathbb{N}$.*

*In addition, for $\tilde{\times} := \left( \begin{pmatrix} 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 \end{pmatrix} \right)$, Equation (2.19) holds when $\delta \geq M^2$ and Equation (2.20) holds when $\delta \geq M$. The other statements of the proposition also hold for this trivial network.*

## 2.3 ReLU DNN approximation of products of multiple factors

Products of more than two factors can be approximated by a binary tree-structured network of $\tilde{\times}_m$-subnetworks defined in Equation (2.29). The following proposition is [75, Proposition 3.3 p. 11] adjusted to Proposition 2.12.

**Proposition 2.14** ([75, Proposition 3.3 p. 11])**.** *For all $\delta_{\tilde{\prod}} \in (0,1)$ and all $n \in \mathbb{N}_{\geq 2}$, there exists a ReLU network $\tilde{\prod}^n \in \mathcal{R}$ with $n$ input values such that*

$$R_\sigma\left(\tilde{\prod}^n\right)\Big|_{[-1,1]^n} : [-1,1]^n \to [-1,1]$$

*and such that*

$$\sup_{(x_1,\dots,x_n)\in[-1,1]^n} \left| \prod_{j=1}^{n} x_j - R_\sigma\left(\tilde{\prod}^n\right)(x_1,\dots,x_n) \right| \le \delta_{\tilde{\prod}}. \tag{2.36}$$

*The depth of* $\tilde{\prod}^n$ *is of the order* $\mathcal{O}(\log(n)\log(n/\delta_{\tilde{\prod}}))$*, whereas its size and the number of non-zero coefficients are of the order* $\mathcal{O}(n\log(n/\delta_{\tilde{\prod}}))$*.*

*Proof.* This proof mainly follows [75, pp. 12–13]. For convenience, we extend the input such that the number of input values is a power of 2: let $\tilde{n} := \min\{2^k : k \in \mathbb{N}, 2^k \ge n\}$ and define $\forall n < j \le \tilde{n} : x_j := 1$. Note that $\tilde{n} < 2n$.

The rest of this proof is divided into three steps. First, for $m \in \mathbb{N}$, we construct the network $\tilde{\prod}^n_m$ as a binary tree-structured network of $\tilde{\times}_m$-subnetworks from Proposition 2.12. In Step 2, it will be shown that for each $\tilde{\times}_m$-subnetwork absolute accuracy $\delta = \delta_{\tilde{\prod}}/\tilde{n}$ suffices. In addition, a corresponding value of $m$ will be given. Finally, Step 3 in Appendix B.3 gives bounds on the depth, the size and the number of non-zero coefficients of the network in terms of $n$ and $\delta_{\tilde{\prod}}$. In addition, Appendix B.3 contains Remark B.2, which describes two ways to implement the auxiliary input values $x_j : n < j \le \tilde{n}$.

*Step 1.* We approximate the product of multiple real numbers of absolute value at most 1 by a binary tree-structured network $\tilde{\prod}^n_m$ consisting of $\tilde{\times}_m$-networks from Proposition 2.12 for $m \in \mathbb{N}$ to be determined later. The binary tree contains $\tilde{n} - 1$ $\tilde{\times}_m$-subnetworks and $\log_2(\tilde{n})$ layers of such subnetworks, e.g. for $n = \tilde{n} = 4$ we have

$$R_\sigma\left(\tilde{\prod}^n_m\right)(x_1,x_2,x_3,x_4) = R_\sigma(\tilde{\times}_m)\Big(R_\sigma(\tilde{\times}_m)(x_1,x_2), R_\sigma(\tilde{\times}_m)(x_3,x_4)\Big), \quad \boldsymbol{x} \in [-1,1]^4, \tag{2.37}$$

as depicted in Figure 2.2.

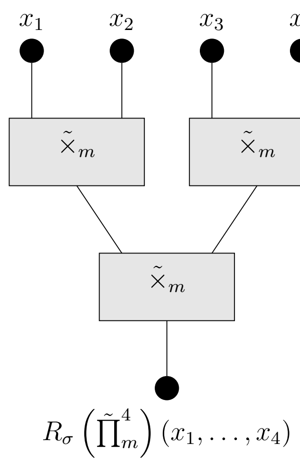


Figure 2.2: Structure of $\tilde{\prod}^4_m$: a binary tree-structured network of $\tilde{\times}_m$-subnetworks.

For $m \in \mathbb{N}$ and $k \in \{1, \ldots, \log_2(\tilde{n})\}$, we define

$$\mathbb{L}_m^k := \text{Parallel}\left(\tilde{\times}_m, \ldots, \tilde{\times}_m\right), \tag{2.38}$$

$$\tilde{\prod}_m^n := \mathbb{L}_m^1 \odot \ldots \odot \mathbb{L}_m^{\log_2 \tilde{n}}, \tag{2.39}$$

where $\mathbb{L}_m^k$ is the parallelisation of $2^{k-1}$ $\tilde{\times}_m$-subnetworks and has $2^k$ inputs. Because we are only interested in inputs in $[-1, 1]$, it suffices to take maximum input value $M = 1$ for each $\tilde{\times}_m$-subnetwork: Proposition 2.12 shows that $\forall (a, b) \in [-1, 1]^2 : R_\sigma(\tilde{\times}_m)(a, b) \in [-1, 1]$, which inductively shows that for all $\tilde{\times}_m$-subnetworks in the binary tree the input and the output are contained in $[-1, 1]$. In particular, $R_\sigma\left(\tilde{\prod}_m^n\right)([-1, 1]^n) \subset [-1, 1]$. Moreover, for each $\tilde{\times}_m$-subnetwork, we take absolute accuracy $\delta := \delta_{\tilde{\Pi}}/\tilde{n}$ in the sense of Equation (2.19). In Step 2 below, we will show that with this choice Equation (2.36) holds.

It follows from Equations (2.38) and (A.23) that for $(b_1, \ldots, b_{2^k}) \in [-1, 1]^{2^k}$

$$R_\sigma(\mathbb{L}_m^k)(b_1, \ldots, b_{2^k}) = \left(R_\sigma(\tilde{\times}_m)(b_1, b_2), \ldots, R_\sigma(\tilde{\times}_m)(b_{2^k-1}, b_{2^k})\right), \tag{2.40}$$

which together with Equation (2.39) and Remark A.3 determines the realisation of $\tilde{\prod}_m^n$.

*Step 2.* We now show that Equation (2.36) holds if Equation (2.19) is satisfied for each $\tilde{\times}_m$-subnetwork with absolute accuracy $\delta = \delta_{\tilde{\Pi}}/\tilde{n}$. If $\tilde{n} = 2$ this is trivially true, if $\tilde{n} > 2$ we use the following claim.

**Claim.**

$$\forall k \in \mathbb{N}, (b_1, \ldots, b_{2^k}) \in [-1, 1]^{2^k} : \left| \prod_{j=1}^{2^k} b_j - R_\sigma\left(\tilde{\prod}_m^{2^k}\right)(b_1, \ldots, b_{2^k}) \right| \leq (2^k - 1)\delta.$$

Using the claim for $k = \log_2(\tilde{n})$, we find

$$\begin{aligned} \left| \prod_{j=1}^{n} x_j - R_\sigma\left(\tilde{\prod}_m^n\right)(x_1, \ldots, x_n) \right| &\overset{\text{def}}{=} \left| \prod_{j=1}^{\tilde{n}} x_j - R_\sigma\left(\tilde{\prod}_m^{\tilde{n}}\right)(x_1, \ldots, x_{\tilde{n}}) \right| \\ &\leq (\tilde{n} - 1)\delta \leq \tfrac{\tilde{n}-1}{\tilde{n}}\delta_{\tilde{\Pi}} < \delta_{\tilde{\Pi}}, \end{aligned}$$

as desired.

We now prove the claim by induction over $k$. For $k = 1$ we have $|R_\sigma(\tilde{\times}_m)(b_1, b_2) - b_1 b_2| \leq \delta = (2^1 - 1)\delta$. For the induction step, we use that

$$\begin{aligned} &R_\sigma\left(\tilde{\prod}_m^{2^k}\right)(b_1, \ldots, b_{2^k}) \\ &= R_\sigma(\tilde{\times}_m)\left(R_\sigma\left(\tilde{\prod}_m^{2^{k-1}}\right)(b_1, \ldots, b_{2^{k-1}}), R_\sigma\left(\tilde{\prod}_m^{2^{k-1}}\right)(b_{2^{k-1}+1}, \ldots, b_{2^k})\right) \end{aligned}$$

and find

$$\begin{aligned}
&\left|\prod_{j=1}^{2^k} b_j - R_\sigma\left(\tilde{\prod}_m^{2^k}\right)(b_1,\dots,b_{2^k})\right| \\
&\le \left|\prod_{j=1}^{2^{k-1}} b_j\right| \left|\prod_{j'=2^{k-1}+1}^{2^k} b_{j'} - R_\sigma\left(\tilde{\prod}_m^{2^{k-1}}\right)(b_{2^{k-1}+1},\dots,b_{2^k})\right| \\
&\quad + \left|\prod_{j=1}^{2^{k-1}} b_j - R_\sigma\left(\tilde{\prod}_m^{2^{k-1}}\right)(b_1,\dots,b_{2^{k-1}})\right| \left|R_\sigma\left(\tilde{\prod}_m^{2^{k-1}}\right)(b_{2^{k-1}+1},\dots,b_{2^k})\right| \\
&\quad + \left|R_\sigma\left(\tilde{\prod}_m^{2^{k-1}}\right)(b_1,\dots,b_{2^{k-1}}) \cdot R_\sigma\left(\tilde{\prod}_m^{2^{k-1}}\right)(b_{2^{k-1}+1},\dots,b_{2^k})\right. \\
&\quad \left. - R_\sigma(\tilde{\times}_m)\left(R_\sigma\left(\tilde{\prod}_m^{2^{k-1}}\right)(b_1,\dots,b_{2^{k-1}}), R_\sigma\left(\tilde{\prod}_m^{2^{k-1}}\right)(b_{2^{k-1}+1},\dots,b_{2^k})\right)\right| \\
&\le (2^{k-1}-1)\delta + (2^{k-1}-1)\delta + \delta = (2^k-1)\delta,
\end{aligned}$$

where the first and the second of the three terms have been estimated using the induction hypothesis for $k-1$ and using $\left|\prod_{j=1}^{2^{k-1}} b_j\right| \le 1$ resp.

$$\left|R_\sigma\left(\tilde{\prod}_m^{2^{k-1}}\right)(b_{2^{k-1}+1},\dots,b_{2^k})\right| \le 1,$$

and where the third of the three terms has been estimated using Equation (2.19). This finishes the proof of the claim and thereby the proof of Equation (2.36).

Because the $\tilde{\times}_m$-subnetworks do not need to satisfy Equation (2.20), we may choose a smaller value for the parameter $m$ than in Equation (2.35). More precisely, for $M = 1$ and $\delta = \delta_{\tilde{\prod}}/\tilde{n}$, according to Equation (2.32), $m = \lceil \frac{1}{2}\log(\tilde{n}/\delta_{\tilde{\prod}}) + \frac{1}{2}\rceil$ implies Equation (2.19) with $\delta = \delta_{\tilde{\prod}}/\tilde{n}$. As shown in the beginning of Step 2 of this proof, Equation (2.36) holds for $\tilde{\prod}_m^n$ if the maximum input value of each $\tilde{\times}_m$-subnetwork equals $M = 1$ and if Equation (2.19) holds with $\delta = \delta_{\tilde{\prod}}/\tilde{n}$ for each $\tilde{\times}_m$-subnetwork. Therefore,

$$\tilde{\prod}^n := \tilde{\prod}^n_{\lceil \frac{1}{2}\log(\tilde{n}/\delta_{\tilde{\prod}}) + \frac{1}{2}\rceil} \tag{2.41}$$

satisfies the properties stated in the proposition. This finishes Step 2 of the proof.

Step 3 is given in Appendix B.3. □

**Remark 2.15.** *By Remark 2.13, for $\delta_{\tilde{\prod}} \ge 1$, the statements of the proposition hold for*

$$\tilde{\prod}^n := \left(\begin{pmatrix} 0 & \cdots & 0 \end{pmatrix}, \begin{pmatrix} 0 \end{pmatrix}\right), \quad n \in \mathbb{N}.$$

**Remark 2.16** (Cf. [75, Remarks 3.4 and 3.5 p. 13])**.** *In Proposition 2.14, products were approximated by $\tilde{\times}_m$-subnetworks from Proposition 2.12 organised in a binary tree. Instead, products of multiple factors could be approximated by the consecutive composition* *of $\tilde{\times}_m$-subnetworks, cf. [89, Equation (14) p. 107]. For all $n \in \mathbb{N}_{\geq 2}$ and $\delta_{\hat{\prod}} \in (0,1)$, let $m = \lceil \frac{1}{2}\log(n/\delta_{\hat{\prod}}) + \frac{1}{2} \rceil$. Then, we can recursively define $\hat{\prod}^2 := \tilde{\times}_m$ and for $n > 2$*

$$\hat{\prod}^n := \tilde{\times}_m \bullet \operatorname{Parallel}(\hat{\prod}^{n-1}, \Phi^{\mathrm{Id}}_{1,1}).$$

*Note that in the parallelisation the subnetwork $\Phi^{\mathrm{Id}}_{1,1}$ gets extended. It holds that for all $y_1, \ldots, y_n \in [-1,1]$*

$$R_\sigma\left(\hat{\prod}^n\right)(y_1, \ldots, y_n) = R_\sigma(\tilde{\times}_m)\left(R_\sigma\left(\hat{\prod}^{n-1}\right)(y_1, \ldots, y_{n-1}), y_n\right).$$

*By inductively applying Equation (2.19), it can be shown that the analogy of Equation (2.36) holds for $\hat{\prod}^n$ if the $\tilde{\times}_m$-networks have pointwise accuracy $\delta = \delta_{\hat{\prod}}/n$.*

*This approximation of products is advantageous for the approximation of polynomials in one variable. If $y_1 = \ldots = y_n =: y \in [-1,1]$, then the approximation of $y^n$ as $\hat{\prod}^n(y, \ldots, y)$ obtains all powers $\{y^j\}_{j \in \{1,\ldots,n\}}$ as partial results, which allows the approximation of polynomials in one variable $y \in [-1,1]$ by a simple extension of the $\hat{\prod}^n$-network.*

*For identical inputs $y$, the $\hat{\prod}^n$-networks defined above are not efficient in terms of size. The stacked parallelisations include extensions of the input. Because all the inputs equal $y$, there are multiple parallel identity networks that all compute $y$. Using a construction similar to that in Remark A.5, $R_\sigma\left(\tilde{\prod}^n\right)(y, \ldots, y)$ can be implemented by smaller networks, having depth, size and number of non-zero coefficients of the order $\mathcal{O}(n \log(n/\delta_{\hat{\prod}}))$. Extensions of such networks implementing polynomials in one variable $y \in [-1,1]$ have the same order of depth, size and number of non-zero coefficients, while their absolute error is bounded from above by $\delta_{\hat{\prod}}$ times the sum of the coefficients of the approximated polynomial.*

*These ideas can be used more generally for the approximation of polynomials of one variable $y \in [-M, M]$ and polynomials of multiple such variables, which in turn can be used for the efficient approximation of smooth functions. This approach is followed in [89, Theorem 1 p. 106] for the ReLU approximation of $\{f \in W^{n,\infty}((0,1)^d) : \|f\|_{W^{n,\infty}((0,1)^d)} \leq 1\}$ for $n, d \in \mathbb{N}$. Similar results for networks with ReLUs and BSUs are [48, Theorems 4 p. 5 and 9 p. 7 and Corollary 10 p. 7–8].*

Let $\mathcal{F}$ be as defined in Section 1.4.2. Given for each $n \in \mathbb{N}$ a finite index set $\Lambda_n \subset \mathcal{F}$ of size $|\Lambda_n| = n$, the following lemma describes ReLU networks $\{f_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu} \in \Lambda_n}$ that approximate the monomials $\boldsymbol{y}^{\boldsymbol{\nu}} : \boldsymbol{y} \in [-1,1]^{\mathbb{N}}, \boldsymbol{\nu} \in \Lambda_n$. The networks are defined such that a weighted sum of their errors is small.

**Lemma 2.17** ([75, Lemma 3.8 p. 14]). *Given $0 < p_b < 1$ and a sequence $(b_{\boldsymbol{\nu}})_{\boldsymbol{\nu} \in \mathcal{F}} \subset \mathbb{R}_{>0}$ satisfying $(b_{\boldsymbol{\nu}})_{\boldsymbol{\nu} \in \mathcal{F}} \in \ell^{p_b}(\mathcal{F})$, as well as for each $n \in \mathbb{N}$ a downward closed index set $\Lambda_n \subset \mathcal{F}$ satisfying $|\Lambda_n| = n$, $\sup_{\boldsymbol{\nu} \in \Lambda_n} |\boldsymbol{\nu}|_1 = \mathcal{O}(\log n)$ and $\bigcup_{\boldsymbol{\nu} \in \Lambda_n} \operatorname{supp} \boldsymbol{\nu} \subset \{1, \ldots, n\}$.*

*Then, with $U := [-1,1]^{\mathbb{N}}$ as in Section 1.4.3, for each $n \in \mathbb{N}$, there exist ReLU networks $\{f_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu} \in \Lambda_n}$ having inputs $y_1, \ldots, y_n \in [-1,1]$ such that for all $\boldsymbol{\nu} \in \Lambda_n$ the realisation $R_\sigma(f_{\boldsymbol{\nu}})$*

*is constant in* $\{y_j : j \in \{1,\dots,n\} \setminus \operatorname{supp} \boldsymbol{\nu}\}$ *and such that* $\{f_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\Lambda_n}$ *satisfy*

$$\sup_{\boldsymbol{y}\in U} \sum_{\boldsymbol{\nu}\in\Lambda_n} b_{\boldsymbol{\nu}} |\boldsymbol{y}^{\boldsymbol{\nu}} - R_\sigma(f_{\boldsymbol{\nu}})(y_1,\dots,y_n)| \le n^{-1/p_b+1}, \tag{2.42}$$

$$\sup_{\boldsymbol{\nu}\in\Lambda_n} \sup_{\boldsymbol{y}\in U} |R_\sigma(f_{\boldsymbol{\nu}})(y_1,\dots,y_n)| \le 1. \tag{2.43}$$

*In addition,*

$$\max_{\boldsymbol{\nu}\in\Lambda_n} \operatorname{depth}(f_{\boldsymbol{\nu}}) = \mathcal{O}(\log(n)\log\log(n)), \tag{2.44}$$

$$\sum_{\boldsymbol{\nu}\in\Lambda_n} \operatorname{size}(f_{\boldsymbol{\nu}}) = \mathcal{O}(n\log(n)\log\log(n)), \tag{2.45}$$

$$\sum_{\boldsymbol{\nu}\in\Lambda_n} \mathcal{M}(f_{\boldsymbol{\nu}}) = \mathcal{O}(n\log(n)\log\log(n)). \tag{2.46}$$

*Proof.* This proof follows [75, p. 15] and consists of three steps. In the first step, for all $n \in \mathbb{N}$, we define the networks $\{f_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\Lambda_n}$. In Step 2, we prove Equations (2.42)–(2.43), while Equations (2.44)–(2.46) are proved in Step 3.

*Step 1.* Let $n \in \mathbb{N}$. For all $\boldsymbol{\nu} \in \Lambda_n$, it holds that $\boldsymbol{y}^{\boldsymbol{\nu}}$ is the product of $|\boldsymbol{\nu}|_1$ numbers. For such $\boldsymbol{\nu}$, by assumption, it holds that $|\boldsymbol{\nu}|_1 \le \sup_{\boldsymbol{\nu}'\in\Lambda_n} |\boldsymbol{\nu}'|_1 = \mathcal{O}(\log n)$. We will approximate $\{\boldsymbol{y}^{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\Lambda_n}$ with networks from Proposition 2.14. Note that in this lemma $n = |\Lambda_n|$, which differs from $n$ as used in Proposition 2.14. There, $n$ denotes the number of factors. In this lemma, for $\boldsymbol{\nu} \in \Lambda_n$, the number of factors in the product $\boldsymbol{y}^{\boldsymbol{\nu}}$ equals $|\boldsymbol{\nu}|_1$.

For $\boldsymbol{\nu} \in \Lambda_n$ satisfying $|\boldsymbol{\nu}|_1 \in \mathbb{N}_{\ge 2}$ and for $i \in \{1,\dots,|\boldsymbol{\nu}|_1\}$, we define $j_{i;\boldsymbol{\nu}} \in \{1,\dots,n\}$ such that for all $\boldsymbol{y} \in U$ it holds that $\boldsymbol{y}^{\boldsymbol{\nu}} = \prod_{i=1}^{|\boldsymbol{\nu}|_1} y_{j_{i;\boldsymbol{\nu}}}$. For such $\boldsymbol{\nu}$, we define the matrix $\mathrm{pr}_{n;\boldsymbol{\nu}} \in \mathbb{R}^{|\boldsymbol{\nu}|_1\times n}$ and the network $f_{\boldsymbol{\nu}} \in \mathcal{R}$ as

$$(\mathrm{pr}_{n;\boldsymbol{\nu}})_{i,j} := \begin{cases} 1 & i \le |\boldsymbol{\nu}|_1, j = j_{i;\boldsymbol{\nu}}, \\ 0 & \text{else}, \end{cases}$$

$$f_{\boldsymbol{\nu}} := \tilde{\prod}^{|\boldsymbol{\nu}|_1} \bullet (\mathrm{pr}_{n;\boldsymbol{\nu}}, \mathbf{0}), \tag{2.47}$$

$$R_\sigma(f_{\boldsymbol{\nu}})(y_1,\dots,y_n) = R_\sigma\left(\tilde{\prod}^{|\boldsymbol{\nu}|_1}\right)(y_{j_{1;\boldsymbol{\nu}}},\dots,y_{j_{|\boldsymbol{\nu}|_1;\boldsymbol{\nu}}}), \tag{2.48}$$

with $\tilde{\prod}^{|\boldsymbol{\nu}|_1}$ as constructed in Proposition 2.14 and where Equation (2.48) follows from Equation (2.47) and Lemma 2.8. For each $\boldsymbol{\nu} \in \Lambda_n$, we define $\delta_{\boldsymbol{\nu}} := \min\{1, b_{\boldsymbol{\nu}}^{-1} n^{-1/p_b}\}$. For $\boldsymbol{\nu} \in \Lambda_n$ satisfying $|\boldsymbol{\nu}|_1 \ge 2$, we take $\delta_{\tilde{\prod}} = \delta_{\boldsymbol{\nu}}$ as accuracy of $\tilde{\prod}^{|\boldsymbol{\nu}|_1}$ in the sense of Equation (2.36). Note that $f_{\boldsymbol{\nu}}$ hence explicitly depends on $n$. In addition, for $\boldsymbol{\nu} = \mathbf{0}$, we define

$$f_{\mathbf{0}} := \left(\begin{pmatrix} 0 & \cdots & 0 \end{pmatrix}, \mathbf{1}\right),$$

which has constant output 1. For $j \in \{1,\dots,n\}$, we define

$$f_{\boldsymbol{e}_j} := \left(\begin{pmatrix} 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \end{pmatrix}, \mathbf{0}\right),$$

which implements $(y_1,\dots,y_n) \mapsto y_j$. For $\boldsymbol{\nu} \in \Lambda_n$ satisfying $|\boldsymbol{\nu}|_1 \in \{0,1\}$, it holds that $\forall \boldsymbol{y} \in U : \boldsymbol{y}^{\boldsymbol{\nu}} = R_\sigma(f_{\boldsymbol{\nu}})(y_1,\dots,y_n)$, hence those $f_{\boldsymbol{\nu}}$ do not contribute to Equation (2.42) and they satisfy Equation (2.43).

It can be seen from Equation (2.48) that for $n \in \mathbb{N}$ and $\boldsymbol{\nu} \in \Lambda_n$ the realisation $R_\sigma(f_{\boldsymbol{\nu}})$ is constant in $\{y_j : j \in \{1, \ldots, n\} \setminus \operatorname{supp} \boldsymbol{\nu}\}$.

*Step 2.* By Equation (2.36), for all $n \in \mathbb{N}$, it holds that

$$\begin{aligned}
\sup_{\boldsymbol{y} \in U} \sum_{\boldsymbol{\nu} \in \Lambda_n} b_{\boldsymbol{\nu}} |\boldsymbol{y}^{\boldsymbol{\nu}} - R_\sigma(f_{\boldsymbol{\nu}})(y_1, \ldots, y_n)| &\overset{(2.36)}{\leq} \sup_{\boldsymbol{y} \in U} \sum_{\boldsymbol{\nu} \in \Lambda_n} b_{\boldsymbol{\nu}} \min\{1, b_{\boldsymbol{\nu}}^{-1} n^{-1/p_b}\} \\
&\leq \sum_{\boldsymbol{\nu} \in \Lambda_n} b_{\boldsymbol{\nu}} b_{\boldsymbol{\nu}}^{-1} n^{-1/p_b} = n^{-1/p_b + 1},
\end{aligned}$$

which shows Equation (2.42). Equation (2.43) directly follows from Proposition 2.14. This finishes Step 2 of the proof.

*Step 3.* Let $n \in \mathbb{N}$. We now show Equations (2.44)–(2.46), giving bounds on the maximum depth and the sum of the sizes of $\{f_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu} \in \Lambda_n}$, as well as on the total number of non-zero coefficients.

First of all, $\operatorname{depth}(f_{\boldsymbol{0}}) = 1$, $\operatorname{size}(f_{\boldsymbol{0}}) = 0$ and $\mathcal{M}(f_{\boldsymbol{0}}) = 1$. Similarly, for $j \in \{1, \ldots, n\}$, it holds that $\operatorname{depth}(f_{\boldsymbol{e}_j}) = 1$, $\operatorname{size}(f_{\boldsymbol{e}_j}) = 0$ and $\mathcal{M}(f_{\boldsymbol{e}_j}) = 1$.

By Equations (2.47), (A.1), (A.2) and (A.3), it follows that for $\boldsymbol{\nu} \in \Lambda_n$ satisfying $|\boldsymbol{\nu}|_1 \geq 2$ it holds that $\operatorname{depth}(f_{\boldsymbol{\nu}}) = \operatorname{depth}\left(\tilde{\prod}^{|\boldsymbol{\nu}|_1}\right)$, $\operatorname{size}(f_{\boldsymbol{\nu}}) = \operatorname{size}\left(\tilde{\prod}^{|\boldsymbol{\nu}|_1}\right)$ and $\mathcal{M}(f_{\boldsymbol{\nu}}) = \mathcal{M}\left(\tilde{\prod}^{|\boldsymbol{\nu}|_1}\right)$.

Using $(b_{\boldsymbol{\nu}})_{\boldsymbol{\nu} \in \mathcal{F}} \in \ell^{p_b}(\mathcal{F}) \hookrightarrow \ell^\infty(\mathcal{F})$, we find $\max_{\boldsymbol{\nu} \in \Lambda_n} 1/\delta_{\boldsymbol{\nu}} = \max_{\boldsymbol{\nu} \in \Lambda_n} \max\{1, b_{\boldsymbol{\nu}} n^{1/p_b}\} = \mathcal{O}(n^{1/p_b})$. This fact and the assumption that $\max_{\boldsymbol{\nu} \in \Lambda_n} |\boldsymbol{\nu}|_1 = \mathcal{O}(\log n)$ show Equation (2.44), because Proposition 2.14 implies that $\operatorname{depth}(f_{\boldsymbol{\nu}}) = \mathcal{O}(\log(|\boldsymbol{\nu}|_1) \log(|\boldsymbol{\nu}|_1/\delta_{\boldsymbol{\nu}}))$.

By Proposition 2.14, we have $\forall \boldsymbol{\nu} \in \Lambda_n : \operatorname{size}(f_{\boldsymbol{\nu}}) \leq C(1 + |\boldsymbol{\nu}|_1 \log(|\boldsymbol{\nu}|_1/\delta_{\boldsymbol{\nu}}))$. Hence, we find (at (*) using that for $x \geq 1$ it holds that $\log(x) \leq x - 1$)

$$\begin{aligned}
\sum_{\boldsymbol{\nu} \in \Lambda_n} \operatorname{size}(f_{\boldsymbol{\nu}}) &\leq \sum_{\boldsymbol{\nu} \in \Lambda_n} C(1 + |\boldsymbol{\nu}|_1 \log(|\boldsymbol{\nu}|_1/\delta_{\boldsymbol{\nu}})) \\
&\leq C \sum_{\boldsymbol{\nu} \in \Lambda_n} (1 + \log(n) \log\log(n)) + C \sum_{\boldsymbol{\nu} \in \Lambda_n} (1 + \log(n) \log(\max\{1, b_{\boldsymbol{\nu}} n^{1/p_b}\})) \\
&\leq C(1 + n \log(n) \log\log(n)) + C \log(n) \sum_{\boldsymbol{\nu} \in \Lambda_n} p_b^{-1} \log(\max\{1, b_{\boldsymbol{\nu}}^{p_b} n\}) \\
&\overset{(*)}{\leq} C(1 + n \log(n) \log\log(n)) + C \log(n) \sum_{\boldsymbol{\nu} \in \Lambda_n} p_b^{-1} b_{\boldsymbol{\nu}}^{p_b} n \\
&\leq C(1 + n \log(n) \log\log(n)) + C n \log(n) p_b^{-1} \|b_{\boldsymbol{\nu}}\|_{\ell^{p_b}(\mathcal{F})}^{p_b} \\
&= \mathcal{O}(n \log(n) \log\log(n)),
\end{aligned}$$

which shows Equation (2.45). Equation (2.46) follows by the same argument. This finishes the proof of Lemma 2.17. $\square$

## 3 ReLU DNN approximation in $\boldsymbol{H^2 \cap H_0^1((0,1)^2)}$

In this section, for $D = (0,1)^2$, we discuss the approximation of functions in $H^2 \cap H_0^1(D)$ by ReLU networks. This approximation is done in two steps: First, functions in $H^2 \cap H_0^1(D)$ are approximated by continuous, piecewise bilinear interpolants. Then, the interpolants are approximated by ReLU networks.

We use an approach similar to that in [54]. There, more regular functions were approximated, namely functions in the *Korobov space*

$$X_0^{2,p}(D) = \{v \in L^p(D) : v|_{\partial D} = 0 \text{ and } D^{\boldsymbol{\alpha}} v \in L^p, |\boldsymbol{\alpha}|_\infty \leq 2\}$$

for any $p \in [2, \infty]$ ([54, Equation (2.10) p. 6], we use a subscript 0 to denote the choice of boundary conditions). In [54], the regularity of functions from Korobov spaces allows to use of *sparse grid* approximations as defined in [12, Section 3.2 pp. 162–182, in particular Equation (3.61) p. 172], using the error estimate from [12, Theorem 3.8 p. 176] for functions in $X_0^{2,p}(D)$. Because we want to approximate functions in $H^2 \cap H_0^1(D)$, we cannot use that error estimate. Therefore, we use a *full grid* approximation, i.e. we use the approximation spaces $\{V_n^{(\infty)}\}_{n \in \mathbb{N}}$ defined in Equation (3.4) below.

The main result of this section is that for $N \in \mathbb{N}$ large enough and for any function in $H^2 \cap H_0^1(D)$ there exists a ReLU network with size at most $N$, having depth of the order $\mathcal{O}(\log N)$ and approximating the function with $H^1$-error of the order $\mathcal{O}(N^{-1/2}(\log N)^{1/2})$ (Proposition 3.5).

### 3.1 Continuous, piecewise bilinear interpolation in $\boldsymbol{H^2 \cap H_0^1((0,1)^2)}$

We start by introducing the continuous, piecewise bilinear functions we will use as interpolants and the corresponding notation. A more general introduction to the used interpolants can be found in [26, Sections 1.2–1.5 pp. 19–67].

In this section, we use the equispaced rectangular partition $\mathcal{T}^{1,2}_{2^n,2^n}$ of $D$ into squares of size $2^{-n} \times 2^{-n}$. It can be constructed from two equispaced one-dimensional partitions $\mathcal{T}^1_{2^n}$ and $\mathcal{T}^2_{2^n}$, which both divide the interval $(0,1)$ into $2^n$ disjoint intervals of length $2^{-n}$. The grid $\mathcal{T}^1_{2^n}$ has grid points $\{x^1_{k_1} := k_1 2^{-n} : k_1 \in \{0, \dots, 2^n\}\}$ and the grid $\mathcal{T}^2_{2^n}$ has grid points $\{x^2_{k_2} := k_2 2^{-n} : k_2 \in \{0, \dots, 2^n\}\}$. The partition $\mathcal{T}^{1,2}_{2^n,2^n}$ has grid points $\{\boldsymbol{x}^{1,2}_{\boldsymbol{k}} := (x^1_{k_1}, x^2_{k_2}) : \boldsymbol{k} = (k_1, k_2) \in (\mathbb{N}_0)^2_{\leq 2^n}\}$ and grid lines $\{\{\boldsymbol{x} \in [0,1]^2 : x^i = x^i_{k_i}\} : i \in \{1,2\}, k_i \in \{0, \dots, 2^n\}\}$.

For the interpolation of functions in $H^2 \cap H_0^1(D)$, we will use continuous, piecewise bilinear functions defined on $\mathcal{T}^{1,2}_{2^n,2^n}$. Because functions in $H^2 \cap H_0^1(D)$ vanish at the boundary, we restrict ourselves to continuous, piecewise bilinear interpolants that vanish at the boundary. The vector space of such functions will be denoted by $V_n^{(\infty)}(D)$, it will be defined below.

Basis functions for $V_n^{(\infty)}(D)$ are constructed as tensor products of univariate continuous, piecewise linear functions, namely so-called *hat functions* on $\mathcal{T}^1_{2^n}$ and $\mathcal{T}^2_{2^n}$. For $i \in \{1,2\}$

and $k_i \in \{1, \dots, 2^n - 1\}$, they are defined as

$$\varphi^i_{k_i}(x_i) := \begin{cases} 2^n(x_i - x^i_{k_i-1}) & x^i_{k_i-1} \le x_i \le x^i_{k_i}, \\ 2^n(x^i_{k_i+1} - x_i) & x^i_{k_i} < x_i \le x^i_{k_i+1}, \\ 0 & x_i < x^i_{k_i-1} \text{ and } x_i > x^i_{k_i+1} \end{cases} \tag{3.1}$$

$$= \sigma\big(1 - 2^n\sigma(x^i_{k_i} - x_i) - 2^n\sigma(x_i - x^i_{k_i})\big), \qquad x_i \in \mathbb{R}. \tag{3.2}$$

We note that $\forall i \in \{1,2\}, k_i \in \{1, \dots, 2^n - 1\} : \varphi^i_{k_i} \in H^1_0(D)$ (cf. [11, Section 8.2 Examples pp. 202–203]).

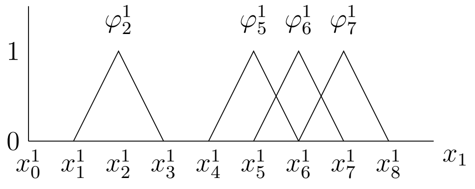


Figure 3.1: Examples of univariate hat functions.

For $\boldsymbol{k} \in \mathbb{N}^2_{\le 2^n-1}$, we define

$$\varphi^{1,2}_{\boldsymbol{k}}(\boldsymbol{x}) := \varphi^1_{k_1}(x_1) \cdot \varphi^2_{k_2}(x_2), \quad \boldsymbol{x} \in D, \tag{3.3}$$

$$S^{1,2}_{\boldsymbol{k}} := \operatorname{supp} \varphi^{1,2}_{\boldsymbol{k}},$$

$$V^{(\infty)}_n(D) := \operatorname{span}\{\varphi^{1,2}_{\boldsymbol{k}} : \boldsymbol{k} \in \mathbb{N}^2_{\le 2^n-1}\}. \tag{3.4}$$

The functions $\{\varphi^{1,2}_{\boldsymbol{k}}\}_{\boldsymbol{k}\in\mathbb{N}^2_{\le 2^n-1}}$ are *continuous, piecewise bilinear hat functions*. Note that $V^{(\infty)}_n(D) \subset H^1_0(D)$, cf. [12, pp. 157–158 between Equations (3.15) and (3.17)].

Note that for $\boldsymbol{k}, \boldsymbol{k}' \in \mathbb{N}^2_{\le 2^n-1}$

$$\varphi^{1,2}_{\boldsymbol{k}}(\boldsymbol{x}^{1,2}_{\boldsymbol{k}'}) = \begin{cases} 1 & \boldsymbol{k} = \boldsymbol{k}', \\ 0 & \text{else}, \end{cases} \tag{3.5}$$

hence $\{\varphi^{1,2}_{\boldsymbol{k}}\}_{\boldsymbol{k}\in\mathbb{N}^2_{\le 2^n-1}}$ form a basis of $V^{(\infty)}_n(D)$ and $\dim V^{(\infty)}_n(D) = (2^n - 1)^2 = \mathcal{O}(2^{2n})$. In what follows, we will often omit $D$ in notation and simply write $V^{(\infty)}_n$.

For $u \in H^2 \cap H^1_0(D)$, using $H^2(D) \hookrightarrow C^0(D)$, we define the following continuous, piecewise bilinear interpolant of $u$:

$$V^{(\infty)}_n \ni \mathbb{I}_{2^n} u : \boldsymbol{x} \mapsto \sum_{\boldsymbol{k}\in\mathbb{N}^2_{\le 2^n-1}} u(\boldsymbol{x}^{1,2}_{\boldsymbol{k}})\varphi^{1,2}_{\boldsymbol{k}}(\boldsymbol{x}), \quad \boldsymbol{x} \in D. \tag{3.6}$$

This interpolant is a *nodal interpolant*, because the weights of $\varphi^{1,2}_{\boldsymbol{k}}$ are simply the function values of $u$ in the *nodes* $\boldsymbol{x}^{1,2}_{\boldsymbol{k}}$. It follows from Equation (3.6) that on each element $T$ of the partition $\mathcal{T}^{1,2}_{2^n,2^n}$ $\mathbb{I}_{2^n}u$ bilinearly interpolates the function values of $u$ in the vertices of $T$, which are nodes of the partition. Hence, it holds that

$$\|\mathbb{I}_{2^n} u\|_{L^\infty(D)} \le \|u\|_{L^\infty(D)} \le C|u|_{H^2(D)}, \tag{3.7}$$

where $|u|_{H^2(D)} := \|\nabla^2 u\|_{L^2(D)}$ denotes the $H^2$-seminorm. In addition, we have the following approximation result:

**Proposition 3.1** ([26, Corollaries 1.110 pp. 61–62 and 1.109 p. 61])**.** *There exists a $C > 0$ such that for all $n \in \mathbb{N}$ and all $u \in H^2 \cap H^1_0(D)$*

$$\|u - \mathbb{I}_{2^n} u\|_{H^j(D)} \le C 2^{-n(2-j)} |u|_{H^2(D)}, \quad j \in \{0, 1\}. \tag{3.8}$$

## 3.2 ReLU DNN approximation of continuous, piecewise bilinear functions on $(0,1)^2$

We fix $n \in \mathbb{N}$ and $w \in V_n^{(\infty)}$ for $V_n^{(\infty)}$ as defined in Equation (3.4) and approximate $w$ by a ReLU network $\tilde{w}$. Throughout this section and the next, we use Assumption 2.1 and Notation 2.2 for all discussed ReLU networks.

Based on Equation (3.2), for $i \in \{1,2\}$ and $k_i \in \{1, \dots, 2^n - 1\}$, we define the network $\tilde{\varphi}^i_{k_i}$ as

$$\tilde{\varphi}^i_{k_i} := \left( \left( \begin{pmatrix} -1 \\ 1 \end{pmatrix}, \begin{pmatrix} x^i_{k_i} \\ -x^i_{k_i} \end{pmatrix} \right), \left( \begin{pmatrix} -2^n & -2^n \end{pmatrix}, \mathbf{1} \right), (\mathrm{Id}_{\mathbb{R}}, \mathbf{0}) \right), \tag{3.9}$$

it exactly implements $\varphi^i_{k_i}$. This network is analogous to $\Phi^2$ in Equation (2.12) from Example 2.6. Alternatively, univariate hat functions can be implemented analogous to $\Phi^1$ from the same example (Equation (2.11)).

For $\boldsymbol{k} \in \mathbb{N}^2_{\le 2^n - 1}$, we use the ReLU network $\tilde{\times}$ of Proposition 2.12 to approximate $\varphi^{1,2}_{\boldsymbol{k}}$:

$$\tilde{\varphi}^{1,2}_{\boldsymbol{k}} := \tilde{\times} \bullet \mathrm{Parallel}(\tilde{\varphi}^1_{k_1}, \tilde{\varphi}^2_{k_2}), \tag{3.10}$$

$$R_\sigma(\tilde{\varphi}^{1,2}_{\boldsymbol{k}})(\boldsymbol{x}) = R_\sigma(\tilde{\times})\big(\varphi^1_{k_1}(x_1), \varphi^2_{k_2}(x_2)\big), \quad \boldsymbol{x} \in D. \tag{3.11}$$

Because $\forall i \in \{1,2\}, k_i \in \{1, \dots, 2^n - 1\} : \|\varphi^i_{k_i}\|_{L^\infty((0,1))} = 1$, we can take $M = 1$ as maximum input value of $\tilde{\times}$ for all $\boldsymbol{k} \in \mathbb{N}^2_{\le 2^n - 1}$. Similarly, we take the same pointwise accuracy $\delta \in (0,1)$ of $\tilde{\times}$ for all $\boldsymbol{k} \in \mathbb{N}^2_{\le 2^n - 1}$, in the sense of Equations (2.19)–(2.20). As a result, all $\tilde{\times}$-subnetworks are identical.

We use the following notation:

$$\tilde{V}_n^{(\infty)}(D) := \mathrm{span}\{R_\sigma(\tilde{\varphi}^{1,2}_{\boldsymbol{k}}) : \boldsymbol{k} \in \mathbb{N}^2_{\le 2^n - 1}\}. \tag{3.12}$$

Based on Equation (3.5), we now construct a network $\tilde{w}$ such that

$$R_\sigma(\tilde{w}) = \sum_{\boldsymbol{k} \in \mathbb{N}^2_{\le 2^n - 1}} w(\boldsymbol{x}^{1,2}_{\boldsymbol{k}}) R_\sigma(\tilde{\varphi}^{1,2}_{\boldsymbol{k}}) \in \tilde{V}_n^{(\infty)}. \tag{3.13}$$

We first construct the parallel implementation of univariate hat functions:

$$P := \mathrm{Parallel}(\text{Shared-Parallel}(\tilde{\varphi}^1_1, \dots, \tilde{\varphi}^1_{2^n-1}), \text{Shared-Parallel}(\tilde{\varphi}^2_1, \dots, \tilde{\varphi}^2_{2^n-1})), \tag{3.14}$$

$$R_\sigma(P)(\boldsymbol{x}) \overset{(2.16),(2.18)}{=} \big(\varphi^1_1(x_1), \dots, \varphi^1_{2^n-1}(x_1), \varphi^2_1(x_2), \dots, \varphi^2_{2^n-1}(x_2)\big), \quad \boldsymbol{x} \in D. \tag{3.15}$$

In order to define the parallel ReLU approximation of bivariate hat functions, we now define the matrix $\mathrm{pr} \in \mathbb{R}^{2(2^n-1)^2 \times 2(2^n-1)}$ to connect the outputs of $P$ to the inputs of the following $\tilde{\times}$-networks:

$$(\mathrm{pr})_{i,j} := \begin{cases} 1 & 1 \le j \le 2^n - 1, i \in 2\mathbb{Z}+1, (j-1)(2^n-1) < \frac{i+1}{2} \le j(2^n-1), \\ 1 & 2^n \le j \le 2(2^n-1), i \in 2(2^n-1)\mathbb{Z} + 2(j-(2^n-1)), \\ 0 & \text{else.} \end{cases} \tag{3.16}$$

It holds that

$$\begin{aligned} R_\sigma((\mathrm{pr}, \mathbf{0}) \bullet P) = (&\varphi_1^1(x_1), \varphi_1^2(x_2), \varphi_1^1(x_1), \varphi_2^2(x_2), \ldots, \\ &\varphi_{2^n-1}^1(x_1), \varphi_{2^n-2}^2(x_2), \varphi_{2^n-1}^1(x_1), \varphi_{2^n-1}^2(x_2)). \end{aligned} \tag{3.17}$$

Now, we define the matrix $\mathbb{W} \in \mathbb{R}^{1 \times (2^n-1)^2}$ and the network $\tilde{w}$ as

$$\begin{aligned} &\forall \boldsymbol{k} \in \mathbb{N}^2_{\le 2^n-1} : \mathbb{W}_{1,(k_1-1)(2^n-1)+k_2} := w(\boldsymbol{x}_{\boldsymbol{k}}^{1,2}), \\ &\tilde{w} := (\mathbb{W}, \mathbf{0}) \bullet \mathrm{Parallel}(\tilde{\times}, \ldots, \tilde{\times}) \bullet (\mathrm{pr}, \mathbf{0}) \bullet P, \end{aligned} \tag{3.18}$$

where $\mathrm{Parallel}(\tilde{\times}, \ldots, \tilde{\times})$ is the parallelisation of the $(2^n-1)^2$ $\tilde{\times}$-subnetworks of $\{\tilde{\varphi}_{\boldsymbol{k}}^{1,2}\}_{\boldsymbol{k} \in \mathbb{N}^2_{\le 2^n-1}}$, which all have maximum input value $M = 1$ and the same pointwise accuracy $\delta \in (0,1)$. Equation (3.13) follows from Equation (3.18), Lemma 2.8 and Equations (3.17) and (A.23).

The structure of $\tilde{w}$ is depicted in Figure 3.2 below.

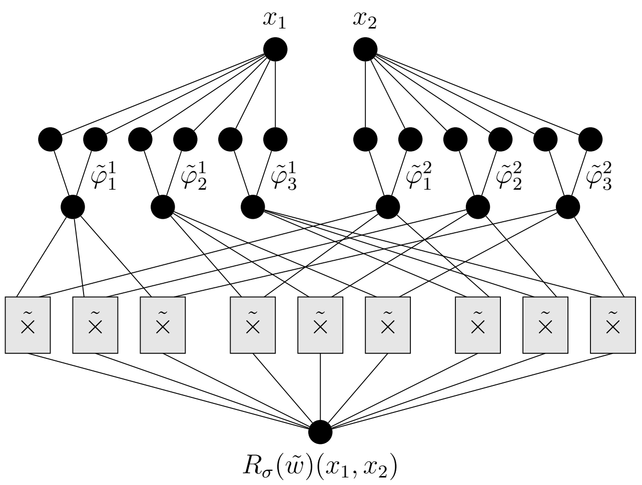


Figure 3.2: ReLU approximation $\tilde{w}$ of $w \in V_2^{(\infty)}$.

**Lemma 3.2.** *For each $\delta \in (0,1)$, each $n \in \mathbb{N}$ and each $w \in V_n^{(\infty)}$, the network $\tilde{w}$ defined in Equation (3.18) satisfies* $\mathrm{depth}(\tilde{w}) = \mathcal{O}(\log(1/\delta))$, $\mathrm{size}(\tilde{w}) = \mathcal{O}(2^{2n}\log(1/\delta))$ *and* $\mathcal{M}(\tilde{w}) = \mathcal{O}(2^{2n}\log(1/\delta))$. *Moreover, the constants implied in these bounds are independent of $w$.*

The proof of the lemma is given in Appendix B.4.

The following lemma bounds the $L^2$- and $L^\infty$-error of the approximation.

**Lemma 3.3.** *For each $\delta \in (0,1)$, each $n \in \mathbb{N}$ and each $w \in V_n^{(\infty)}$, there exists a ReLU network $\tilde{w}$ with depth of the order $\mathcal{O}(\log(1/\delta))$ and size and number of non-zero coefficients of the order $\mathcal{O}(2^{2n}\log(1/\delta))$, with implied constants independent of $w$, satisfying*

$$\|w - R_\sigma(\tilde{w})\|_{L^2(D)} \leq \|w - R_\sigma(\tilde{w})\|_{L^\infty(D)} \leq 4\delta\|w\|_{L^\infty(D)} \tag{3.19}$$

*and hence*

$$\delta \leq \varepsilon/4 \quad \Rightarrow \quad \|w - R_\sigma(\tilde{w})\|_{L^2(D)} \leq \varepsilon\|w\|_{L^\infty(D)}. \tag{3.20}$$

*Proof.* We take $\tilde{w}$ as defined in Equation (3.18) and we use Lemma 3.2. It remains to show Equation (3.19), from which Equation (3.20) directly follows.

We note that at each $\boldsymbol{x} \in D$ at most four functions of the form $\varphi_{\boldsymbol{k}}^{1,2}$ do not vanish, because for at most two values of $k_1 \in \{1, \dots, 2^n - 1\}$ it holds that $\varphi_{k_1}^1(x_1) \neq 0$ and for at most two values of $k_2 \in \{1, \dots, 2^n - 1\}$ it holds that $\varphi_{k_2}^2(x_2) \neq 0$. It follows from the zero-in-zero-out property of $\tilde{\times}$ that at most four functions of the form $R_\sigma(\tilde{\varphi}_{\boldsymbol{k}}^{1,2})$ do not vanish at $\boldsymbol{x}$. Combined with Equations (3.13) and (2.19), this shows the second inequality in Equation (3.19). The first inequality is true for all functions in $L^\infty((0,1)^2)$. $\square$

We next estimate the $H^1$-error of the approximation.

**Lemma 3.4.** *For each $\delta \in (0,1)$, each $n \in \mathbb{N}$ and each $w \in V_n^{(\infty)}$, there exists a ReLU network $\tilde{w}$ with depth of the order $\mathcal{O}(\log(1/\delta))$ and size and number of non-zero coefficients of the order $\mathcal{O}(2^{2n}\log(1/\delta))$, with implied constants independent of $w$, satisfying Equations (3.19)–(3.20) and*

$$\delta \leq \varepsilon 2^{-n-3} \quad \Rightarrow \quad \|w - R_\sigma(\tilde{w})\|_{H^1(D)} \leq \varepsilon\|w\|_{L^\infty(D)}. \tag{3.21}$$

*Proof.* By Lemma 3.3, it only remains to show Equation (3.21).

At first, we fix $\boldsymbol{k} \in \mathbb{N}^2_{\leq 2^n-1}$ and $T \in \mathcal{T}^{1,2}_{2^n,2^n}$ such that $T \subset S_{\boldsymbol{k}}^{1,2}$. Using the chain rule and Equation (2.20), we find

$$\begin{aligned}
&\left\| \frac{d}{dx_1}\varphi_{\boldsymbol{k}}^{1,2} - \frac{d}{dx_1}R_\sigma(\tilde{\varphi}_{\boldsymbol{k}}^{1,2}) \right\|_{L^\infty(T)} \\
&= \left\| \frac{d}{dx_1}\big(\varphi_{k_1}^1(x_1)\varphi_{k_2}^2(x_2)\big) - \frac{d}{dx_1}\big(R_\sigma(\tilde{\times})(\varphi_{k_1}^1(x_1), \varphi_{k_2}^2(x_2))\big) \right\|_{L^\infty(T)} \\
&= \left\| \varphi_{k_2}^2(x_2)\frac{d}{dx_1}\big(\varphi_{k_1}^1(x_1)\big) - \frac{d}{da}\bigg|_{a=\varphi_{k_1}^1(x_1)}\big(R_\sigma(\tilde{\times})(a, \varphi_{k_2}^2(x_2))\big)\frac{d}{dx_1}\big(\varphi_{k_1}^1(x_1)\big) \right\|_{L^\infty(T)} \\
&\overset{(2.20)}{\leq} \delta \left\| \frac{d}{dx_1}\varphi_{k_1}^1 \right\|_{L^\infty(T)} = \delta 2^n,
\end{aligned}$$

where all derivatives are weak derivatives. Hence,

$$\left\| \frac{d}{dx_1}\varphi_{\boldsymbol{k}}^{1,2} - \frac{d}{dx_1}R_\sigma(\tilde{\varphi}_{\boldsymbol{k}}^{1,2}) \right\|^2_{L^2(T)} \leq \delta^2 2^{2n} 2^{-2n} = \delta^2.$$

Combined with the analogous statement for $\frac{d}{dx_2}$, we have

$$\left\|\nabla \varphi_{\boldsymbol{k}}^{1,2} - \nabla R_\sigma(\tilde{\varphi}_{\boldsymbol{k}}^{1,2})\right\|_{L^2(T)}^2 \leq 2\delta^2.$$

Using the general fact $\forall m \in \mathbb{N} : \|\sum_{j=1}^m f_j\|^2 \leq m \sum_{j=1}^m \|f_j\|^2$ and the fact that at each $\boldsymbol{x} \in D$ at most four functions of the form $R_\sigma(\tilde{\varphi}_{\boldsymbol{k}}^{1,2})$ do not vanish, we get

$$\begin{aligned}
\|\nabla w - \nabla R_\sigma(\tilde{w})\|_{L^2(D)}^2 &\leq \|w\|_{L^\infty(D)}^2 \sum_{T\in\mathcal{T}_{2^n,2^n}^{1,2}} \left\| \sum_{\boldsymbol{k}:S_{\boldsymbol{k}}^{1,2}\cap\mathring{T}\neq\emptyset} \nabla\varphi_{\boldsymbol{k}}^{1,2} - \sum_{\boldsymbol{k}:S_{\boldsymbol{k}}^{1,2}\cap\mathring{T}\neq\emptyset} \nabla R_\sigma(\tilde{\varphi}_{\boldsymbol{k}}^{1,2})\right\|_{L^2(T)}^2 \\
&\leq \|w\|_{L^\infty(D)}^2 \sum_{T\in\mathcal{T}_{2^n,2^n}^{1,2}} 4 \sum_{\boldsymbol{k}:S_{\boldsymbol{k}}^{1,2}\cap\mathring{T}\neq\emptyset} \left\|\nabla\varphi_{\boldsymbol{k}}^{1,2} - \nabla R_\sigma(\tilde{\varphi}_{\boldsymbol{k}}^{1,2})\right\|_{L^2(T)}^2 \\
&\leq \|w\|_{L^\infty(D)}^2 2^{2n} 4^2 2\delta^2,
\end{aligned}$$

where $\mathring{T}$ denotes the interior of $T$. Combining this with the $L^2$-error estimate in Equation (3.19), we get

$$\|w - R_\sigma(\tilde{w})\|_{H^1(D)}^2 \leq 4^2\delta^2\|w\|_{L^\infty(D)}^2(2^{2n+1}+1) \leq 2^{2n+6}\delta^2\|w\|_{L^\infty(D)}^2$$

and hence

$$\delta \leq \varepsilon 2^{-n-3} \quad \Rightarrow \quad \|w - R_\sigma(\tilde{w})\|_{H^1(D)} \leq \varepsilon\|w\|_{L^\infty(D)}.$$

□

## 3.3 Convergence rate of a family of ReLU DNN approximations of functions in $H^2 \cap H_0^1((0,1)^2)$ in terms of the network size

We now combine the results of Sections 3.1 and 3.2. As in the previous section, all considered ReLU networks satisfy Assumption 2.1 and we use Notation 2.2.

Equation (3.8) from Proposition 3.1 shows that there exists a $C > 0$ such that for each $n \in \mathbb{N}$ and each $u \in H^2 \cap H_0^1(D)$

$$\|u - \mathbb{I}_{2^n}u\|_{H^j(D)} \leq C2^{-n(2-j)}|u|_{H^2(D)}, \quad j \in \{0,1\}.$$

For each $n \in \mathbb{N}$, we now approximate $\mathbb{I}_{2^n}u$ by a ReLU network $\widetilde{\mathbb{I}_{2^n}u}$ as in Lemma 3.4, for $\delta \in (0,1)$ to be determined next. Taking $\varepsilon_{L^2} := 2^{-2n}$ in Equation (3.20) and $\varepsilon_{H^1} := 2^{-n}$ in Equation (3.21), together with Equations (3.7) and (3.8), we find that for $j \in \{0,1\}$ it holds that

$$\begin{aligned}
\|u - R_\sigma\big(\widetilde{\mathbb{I}_{2^n}u}\big)\|_{H^j(D)} &\leq \|u - \mathbb{I}_{2^n}u\|_{H^j(D)} + \|\mathbb{I}_{2^n}u - R_\sigma\big(\widetilde{\mathbb{I}_{2^n}u}\big)\|_{H^j(D)} \\
&\overset{(3.8),(3.20),(3.21)}{\leq} C2^{-n(2-j)}|u|_{H^2(D)} + 2^{-n(2-j)}\|\mathbb{I}_{2^n}u\|_{L^\infty(D)} \\
&\overset{(3.7)}{\leq} C2^{-n(2-j)}|u|_{H^2(D)}.
\end{aligned}$$

The choices for $\varepsilon_{L^2}$ and $\varepsilon_{H^1}$ require $\delta$ to satisfy $\delta \leq 2^{-2n}/4$ and $\delta \leq 2^{-n}2^{-n-3}$, i.e. $\delta = 2^{-2n-3} = \mathcal{O}(2^{-2n})$ suffices. For that choice of $\delta$, $\widetilde{\mathbb{I}_{2^n} u}$ has depth of the order $\mathcal{O}(\log(1/\delta)) = \mathcal{O}(n)$ and size and number of non-zero coefficients of the order $\mathcal{O}(2^{2n}\log(1/\delta)) = \mathcal{O}(2^{2n}n)$.

For all $n \in \mathbb{N}$, denoting size$\big(\widetilde{\mathbb{I}_{2^n} u}\big) =: N$, we have for $j \in \{0,1\}$ and $C > 0$ independent of $u$ and $N$

$$\|u - R_\sigma\big(\widetilde{\mathbb{I}_{2^n} u}\big)\|_{H^j(D)} \leq C N^{-(1-j/2)}(\log N)^{1-j/2}|u|_{H^2(D)}.$$

In Lemma 3.4, the size of $\tilde{w}$ is bounded in terms of $n$ and $\delta$, but independent of $w \in V_n^{(\infty)}$. As a result, size$(\widetilde{\mathbb{I}_{2^n} u})$ is bounded from above by a function of $n$, independent of $u \in H^2 \cap H_0^1(D)$ (however, the error of the approximation does depend on $u$). Hence, we can define $m_0 \in \mathbb{N}_{>1}$ to be that upper bound for $n = 1$.

For general $N \in \mathbb{N}_{\geq m_0}$ and $n = \max\{n_* : \text{size}(\widetilde{\mathbb{I}_{2^{n_*}} u}) \leq N\}$, we define $\tilde{I}_N u := \widetilde{\mathbb{I}_{2^n} u}$ satisfying $R_\sigma\big(\widetilde{\mathbb{I}_{2^n} u}\big) \in \tilde{V}_n^{(\infty)}$. For the same $N$ and $n$, we define $I_N u := \mathbb{I}_{2^n} u \in V_n^{(\infty)}$.

Note that $\forall n \in \mathbb{N} : \delta = 2^{-2n-3} \leq 1/4$. Hence, by Equation (3.19),

$$\|I_N u - R_\sigma(\tilde{I}_N u)\|_{L^\infty(D)} \leq \|I_N u\|_{L^\infty(D)}.$$

The results of this section are summarised by the following proposition:

**Proposition 3.5.** *There exists a $C > 0$ such that for each $N \in \mathbb{N}_{\geq m_0}$ and each $u \in H^2 \cap H_0^1(D)$ there exists a ReLU network $\tilde{I}_N u$ of size at most $N$ satisfying*

$$\|u - R_\sigma(\tilde{I}_N u)\|_{H^j(D)} \leq C N^{-(1-j/2)}(\log N)^{1-j/2}|u|_{H^2(D)}, \tag{3.22}$$

$$\|I_N u - R_\sigma(\tilde{I}_N u)\|_{L^\infty(D)} \leq \|I_N u\|_{L^\infty(D)}. \tag{3.23}$$

*Moreover,* depth$(\tilde{I}_N u) = \mathcal{O}(\log N)$ *and* $\mathcal{M}(\tilde{I}_N u) = \mathcal{O}(N)$. *For these two bounds the implied constants are independent of $u$.*

# 4 Properties of the Taylor gpc approximation of the solution map of a parametric diffusion equation

In this section, we discuss properties of the solutions of a class of parametric elliptic PDEs. More precisely, we study the elliptic diffusion equation

$$-\operatorname{div}(a(\boldsymbol{y})\nabla u) = f, \qquad u|_{\partial D} = 0 \tag{4.1}$$

on the domain $D = (0,1)^2$ for given $f \in H^{-1}(D)$, arbitrary parameters $\boldsymbol{y} \in U = [-1,1]^{\mathbb{N}}$ (see Section 1.4.3), given parameter-dependent scalar diffusion coefficient $a(\boldsymbol{y}) \in L^\infty(D)$ (see Section 1.4.4) and unknown $u$. We consider the case in which all functions are real valued.

We will often use the following weak form of Equation (4.1). For $V := H_0^1(D)$ and for $V' = H^{-1}(D)$ denoting the dual of $V$, it reads

$$A_{\boldsymbol{y}}(u,v) = \langle f, v\rangle_{V',V}, \quad \forall v \in V, \tag{4.2}$$

where

$$A_{\boldsymbol{y}}(u,v) := \int_D a(\boldsymbol{y})\nabla u \cdot \nabla v, \quad u, v \in V \tag{4.3}$$

and where $\langle\cdot,\cdot\rangle_{V',V}$ is the duality pairing. The boundary conditions are imposed by the choice $V = H_0^1(D)$.

Section 4.1 discusses Equation (4.2) for fixed $\boldsymbol{y} \in U$. Section 4.2 discusses properties of the parametric diffusion equation and the corresponding solution map $u : U \to V$. It introduces the Taylor gpc expansion of the solution map with Taylor gpc coefficients in $V$. It proposes $n$-term truncations of the Taylor gpc expansion as Taylor gpc approximations. For $\mathcal{F}$ as defined in Section 1.4.2, it shows that the weighted $\ell^2(\mathcal{F})$-summability of the $V$-norms of the Taylor gpc coefficients leads to $\ell^p(\mathcal{F})$-summability of the $V$-norms for some $0 < p < 1$ and to an upper bound on the approximation error of the Taylor gpc approximations in $V$-norm. Section 4.3 gives sufficient conditions for the required weighted $\ell^2(\mathcal{F})$-summability of the $V$-norms of the Taylor gpc coefficients. For more regular data, which imply $u : U \to H^2 \cap H_0^1(D)$, Section 4.4 shows a similar summability result for the $H^2$-norms of the Taylor gpc coefficients, from which the $\ell^p(\mathcal{F})$-summability of the $H^2$-norms of the Taylor gpc coefficients follows for some $0 < p < 1$ that is in general larger than the $p$ for the summability of the $V$-norms.

## 4.1 Properties of the non-parametric diffusion equation

In this section, we study Equation (4.2) for fixed $\boldsymbol{y} \in U$. We simply write $a$ for the diffusion coefficient $a(\boldsymbol{y})$ and $A$ for the bilinear form $A_{\boldsymbol{y}}$.

We recall the following result stating the well-posedness of Equation (4.2). It is a specific case of the Lax-Milgram lemma.

**Lemma 4.1** (Special case of [26, Lemma 2.2 p. 83], cf. [26, Sections A.2.3 and A.2.4 pp. 472–474] and [74, Theorem C.20 p. 450])**.** *Consider*

$$\int_D a\nabla u \cdot \nabla v = \langle f, v\rangle_{V',V}, \quad \forall v \in V \tag{4.4}$$

*for given $a \in L^\infty(D)$ and $f \in V'$. Assume that there exists a constant $a_{\min} > 0$ such that*

$$0 < a_{\min} \leq \operatorname*{ess\,inf}_{\boldsymbol{x} \in D} a(\boldsymbol{x}). \tag{4.5}$$

*Then, there exists a unique solution $u \in V$ of Equation (4.4) that satisfies*

$$\|u\|_V \leq \tfrac{1}{a_{\min}} \|f\|_{V'}. \tag{4.6}$$

*Moreover, if in addition $\exists a_{\max} < \infty$ such that*

$$\operatorname*{ess\,sup}_{\boldsymbol{x} \in D} a(\boldsymbol{x}) \leq a_{\max} < \infty, \tag{4.7}$$

*then the bounded linear operator $\mathcal{A} \in \mathcal{L}(V, V')$ defined by*

$$\forall u, v \in V : \langle \mathcal{A} u, v \rangle_{V', V} = A(u, v) = \int_D a \nabla u \cdot \nabla v \tag{4.8}$$

*is boundedly invertible.*

In the parametric context, for $\boldsymbol{y} \in U$, we will denote $\mathcal{A}$ by $\mathcal{A}_{\boldsymbol{y}}$.

In applications, the following *variational formulation* of the PDE in Equation (4.4) is very useful.

**Proposition 4.2** (Special case of [26, Proposition 2.4 p. 84], see also [26, Remark 2.5 p. 84])**.** *Under the assumptions of Lemma 4.1, the solution $u \in V$ of Equation (4.4) is uniquely determined by*

$$u = \operatorname*{arg\,min}_{v \in V} J(v), \qquad J(v) := \tfrac{1}{2} A(v, v) - \langle f, v \rangle_{V', V}, \quad v \in V. \tag{4.9}$$

## 4.2 Properties of the parametric diffusion equation

We now consider the parametric problem, i.e. Equation (4.2) with parametric scalar diffusion coefficient $a : U \ni \boldsymbol{y} \mapsto a(\boldsymbol{y}) \in L^\infty(D)$. In order for Lemma 4.1 to hold for all $\boldsymbol{y} \in U$, we assume $a(\boldsymbol{y})$ to satisfy the following *uniform ellipticity condition*: There exist real numbers $0 < a_{\min} \leq a_{\max} < \infty$ such that

$$\forall \boldsymbol{y} \in U : a_{\min} \leq \operatorname*{ess\,inf}_{\boldsymbol{x} \in D} a(\boldsymbol{x}, \boldsymbol{y}) \leq \operatorname*{ess\,sup}_{\boldsymbol{x} \in D} a(\boldsymbol{x}, \boldsymbol{y}) \leq a_{\max}. \tag{4.10}$$

Under this assumption, for each $\boldsymbol{y} \in U$, there exists a unique solution $u \in V$ of Equation (4.2), which we denote by $u(\boldsymbol{y}) \in V$. Note that it satisfies Equation (4.6) for $a_{\min}$ independent of $\boldsymbol{y} \in U$. The map $u : U \to V : \boldsymbol{y} \mapsto u(\boldsymbol{y})$ is the solution map of the parametric PDE in Equation (4.2) (see Section 1.1.2).

In the rest of Section 4, we discuss several results stated in [3], which give properties of the solution map under the assumption that the diffusion coefficient is *affinely parametrised*, i.e.

$$a : U \to L^\infty(D) : \boldsymbol{y} \mapsto \bar{a} + \sum_{j\in\mathbb{N}} y_j \psi_j \tag{4.11}$$

for $\bar{a} \in L^\infty(D)$ and $\forall j \in \mathbb{N} : \psi_j \in L^\infty(D)$.

For this parametrisation of $a$, the uniform ellipticity condition is satisfied if

$$\sum_{j\in\mathbb{N}} |\psi_j(\boldsymbol{x})| \le \bar{a}(\boldsymbol{x}) - a_{\min} \qquad \text{for a.e. } \boldsymbol{x} \in D \tag{4.12}$$

or, equivalently, if

$$0 < \operatorname*{ess\,inf}_{\boldsymbol{x}\in D} \bar{a}(\boldsymbol{x}) =: \bar{a}_{\min} \tag{4.13}$$

and

$$\theta := \left\| \frac{\sum_{j\in\mathbb{N}} |\psi_j|}{\bar{a}} \right\|_{L^\infty(D)} < 1. \tag{4.14}$$

The fact that Equation (4.12) implies Equation (4.14) can be seen from

$$\left\| \frac{\sum_{j\in\mathbb{N}} |\psi_j|}{\bar{a}} \right\|_{L^\infty(D)} \le \left\| \frac{\bar{a} - a_{\min}}{\bar{a}} \right\|_{L^\infty(D)} \le 1 - \frac{a_{\min}}{\|\bar{a}\|_{L^\infty(D)}} < 1$$

(e.g. [4, Equation (2.1) p. 325]). The other implication is similar to [17, Equations (2.2)–(2.4) p. 621]:

$$\begin{aligned} \sum_{j\in\mathbb{N}} |\psi_j(\boldsymbol{x})| &= \frac{\sum_{j\in\mathbb{N}} |\psi_j(\boldsymbol{x})|}{\bar{a}(\boldsymbol{x})} \bar{a}(\boldsymbol{x}) \le \theta \bar{a}(\boldsymbol{x}) \\ &\le \bar{a}(\boldsymbol{x}) - (1-\theta)\bar{a}_{\min} =: \bar{a}(\boldsymbol{x}) - a_{\min}, \qquad \text{for a.e. } \boldsymbol{x} \in D. \end{aligned}$$

For $\mathcal{F}$ the index set defined in Section 1.4.2, by [17, Theorem 4.2 p. 625], for all $\boldsymbol{\nu} \in \mathcal{F}$, the partial derivative $\partial^{\boldsymbol{\nu}} u(\boldsymbol{y}) \in V$ exists at all $\boldsymbol{y} \in U$. In general, the assumptions on $\{\psi_j\}_{j\in\mathbb{N}}$ made in [17] are stronger than ours. However, those assumptions are not used in the proof of [17, Theorem 4.2], we may hence use that result.

In order to approximate the solution map, we note that it has the following Taylor gpc expansion (see Section 1.1.2):

$$\forall \boldsymbol{y} \in U : u(\boldsymbol{y}) = \sum_{\boldsymbol{\nu}\in\mathcal{F}} t_{\boldsymbol{\nu}} \boldsymbol{y}^{\boldsymbol{\nu}}, \qquad t_{\boldsymbol{\nu}} := \tfrac{1}{\boldsymbol{\nu}!} \partial^{\boldsymbol{\nu}} u|_{\boldsymbol{y}=0} \in V. \tag{4.15}$$

The functions $t_{\boldsymbol{\nu}} \in V$ are the Taylor gpc coefficients of the solution map. The unconditional convergence of this sum directly follows from $\forall \boldsymbol{y} \in U, \boldsymbol{\nu} \in \mathcal{F} : |\boldsymbol{y}^{\boldsymbol{\nu}}| \le 1$ and the $\ell^1(\mathcal{F})$-summability of $(\|t_{\boldsymbol{\nu}}\|_V)_{\boldsymbol{\nu}\in\mathcal{F}}$, implied by the $\ell^p(\mathcal{F})$-summability of that sequence for some $0 < p < 1$, which we will show in Lemma 4.3 below under the assumptions made there.

Given for each $n \in \mathbb{N}$ an index set $\Lambda_n \subset \mathcal{F}$ satisfying $|\Lambda_n| = n$, we can approximate $u$ by

$$u_n : U \to V : \boldsymbol{y} \mapsto \sum_{\boldsymbol{\nu}\in\Lambda_n} \boldsymbol{y}^{\boldsymbol{\nu}} t_{\boldsymbol{\nu}}, \tag{4.16}$$

which is an *n-term truncation* of the Taylor gpc expansion, a Taylor gpc approximation (see [21, especially Section 2 pp. 56–60] for a general introduction to $n$-term approximations). The approximation error is bounded from above by

$$\|u - u_n\|_{L^\infty(U,V)} := \sup_{\boldsymbol{y}\in U} \|u(\boldsymbol{y}) - u_n(\boldsymbol{y})\|_V \leq \sum_{\boldsymbol{\nu}\in\Lambda_n^c} \|t_{\boldsymbol{\nu}}\|_V. \tag{4.17}$$

This bound is minimised, when for each $n \in \mathbb{N}$ the index set $\Lambda_n \subset \mathcal{F}$ contains $n$ elements $\boldsymbol{\nu} \in \mathcal{F}$ for which $\|t_{\boldsymbol{\nu}}\|_V$ are largest. Note that such sets $\Lambda_n$ need not be unique. Approximations of $u$ using this choice of index sets $\{\Lambda_n\}_{n\in\mathbb{N}}$ are called *best n-term approximations* of $u$. In general, $(\|t_{\boldsymbol{\nu}}\|_V)_{\boldsymbol{\nu}\in\mathcal{F}}$ are unknown, hence the best $n$-term approximation cannot be implemented directly. Instead, we construct $\{\Lambda_n\}_{n\in\mathbb{N}}$ in Lemma 4.3 below, for $b_{\boldsymbol{\nu}} = \|t_{\boldsymbol{\nu}}\|_V$. The construction in the lemma is based on the weighted $\ell^2(\mathcal{F})$-summability of $(\|t_{\boldsymbol{\nu}}\|_V)_{\boldsymbol{\nu}\in\mathcal{F}}$ and gives an upper bound on $\sum_{\boldsymbol{\nu}\in\Lambda_n^c} \|t_{\boldsymbol{\nu}}\|_V$ for $\{\Lambda_n\}_{n\in\mathbb{N}}$ as constructed there. Independent of the choice of $\{\Lambda_n\}_{n\in\mathbb{N}}$, it is shown that the assumed weighted $\ell^2(\mathcal{F})$-summability implies $(\|t_{\boldsymbol{\nu}}\|_V)_{\boldsymbol{\nu}\in\mathcal{F}} \in \ell^p(\mathcal{F})$ for some $0 < p < 1$.

**Lemma 4.3** (Special case of [75, Lemma 2.8 p. 7], see also [4, Corollary 2.3 p. 328])**.** *For $0 < p < 1$ and a decreasing sequence $\boldsymbol{\beta} = (\beta_j)_{j\in\mathbb{N}} \subset (0,1)$ satisfying $\boldsymbol{\beta} \in \ell^q(\mathbb{N})$ for $q = \frac{2p}{2-p}$, assume that for some non-negative sequence $(b_{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}}$*

$$(\boldsymbol{\beta}^{-\boldsymbol{\nu}} b_{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}} \in \ell^2(\mathcal{F}). \tag{4.18}$$

*Then, $(b_{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}} \in \ell^p(\mathcal{F})$ and for each $n \in \mathbb{N}$ there exists a downward closed index set $\Lambda_n \subset \mathcal{F}$ of size $|\Lambda_n| = n$ such that*

$$\sum_{\boldsymbol{\nu}\in\Lambda_n^c} b_{\boldsymbol{\nu}} \leq Cn^{-1/p+1} \tag{4.19}$$

*and*

$$\max_{\boldsymbol{\nu}\in\Lambda_n} |\boldsymbol{\nu}|_1 = \mathcal{O}(\log n) \tag{4.20}$$

*and such that for each $j \in \mathbb{N}$: $\boldsymbol{e}_j \in \Lambda_n$ implies $\boldsymbol{e}_i \in \Lambda_n$ for $i \leq j$.*

The proof of Lemma 4.3 is given in Appendix B.5.

When its requirements are satisfied for $b_{\boldsymbol{\nu}} = \|t_{\boldsymbol{\nu}}\|_V$, the lemma first of all shows that $(\|t_{\boldsymbol{\nu}}\|_V)_{\boldsymbol{\nu}\in\mathcal{F}} \in \ell^p(\mathcal{F}) \hookrightarrow \ell^1(\mathcal{F})$, i.e. the Taylor gpc expansion in Equation (4.15) converges unconditionally. In addition, we get the following bound on the $V$-error of $u_n$ defined in Equation (4.16):

$$\|u - u_n\|_{L^\infty(U,V)} \leq \sum_{\boldsymbol{\nu}\in\Lambda_n^c} \|t_{\boldsymbol{\nu}}\|_V \leq Cn^{-1/p+1}. \tag{4.21}$$

Hence, for fast convergence of $u_n$, it suffices to show $\ell^2$-summability of $(\boldsymbol{\beta}^{-\boldsymbol{\nu}}\|t_{\boldsymbol{\nu}}\|_V)_{\boldsymbol{\nu}\in\mathcal{F}}$ for any $\boldsymbol{\beta} \in \ell^q(\mathbb{N})$ satisfying the requirements of the lemma for small $q$. The next section shows sufficient conditions for that summability to hold.

## 4.3 Weighted summability of the $H^1$-norms of the Taylor gpc coefficients of the solution map

The desired summability is shown in [4, Section 2 pp. 325–329], we now recall the results obtained there and their proofs.

**Proposition 4.4** ([4, Theorem 2.2 p. 327]). *Let $f \in V'$ and $a : U \to L^\infty(D)$ be given such that $a$ satisfies Equation (4.11) with $\bar{a} \in L^\infty(D)$ satisfying Equation (4.13) and with $\forall j \in \mathbb{N} : \psi_j \in L^\infty(D)$. Let $\{t_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\mathcal{F}}$ be as in Equation (4.15). Assume that for $\boldsymbol{\beta} = (\beta_j)_{j\in\mathbb{N}} \subset \mathbb{R}_{>0}$ we have the weighted uniform ellipticity condition*

$$\theta := \left\| \frac{\sum_{j\in\mathbb{N}} \beta_j^{-1} |\psi_j|}{\bar{a}} \right\|_{L^\infty(D)} < 1. \tag{4.22}$$

*Then, $(\boldsymbol{\beta}^{-\boldsymbol{\nu}} \|t_{\boldsymbol{\nu}}\|_V)_{\boldsymbol{\nu}\in\mathcal{F}} \in \ell^2(\mathcal{F})$.*

**Remark 4.5** ([4, Theorem 2.2 p. 327]). *For $\boldsymbol{\beta} \subset \mathbb{R}_{>0}$, Equations (4.11), (4.15) and (4.22) are equivalent to those equations for $\tilde{y}_j := \beta_j y_j$ instead of $y_j$, $\tilde{\psi}_j := \beta_j^{-1}\psi_j$ instead of $\psi_j$, $\tilde{t}_{\boldsymbol{\nu}} := \boldsymbol{\beta}^{-\boldsymbol{\nu}} t_{\boldsymbol{\nu}}$ instead of $t_{\boldsymbol{\nu}}$ and $\tilde{\beta}_j := 1$ instead of $\beta_j$. This means that we can assume $\boldsymbol{\beta} = \mathbf{1}$ without loss of generality. The parameter set $\tilde{U} = \times_{j\in\mathbb{N}}[-\beta_j, \beta_j]$ from which $\tilde{\boldsymbol{y}}$ are taken is compact as well, see Section 1.4.3.*

*Proof of Proposition 4.4.* We follow [4, Section 2 pp. 325–329] and [3, pp. 2163–2164]. By Remark 4.5, we may without loss of generality assume that $\boldsymbol{\beta} = \mathbf{1}$.

For $\boldsymbol{\nu} \in \mathcal{F}$, we define

$$d_{\boldsymbol{\nu}} := \int_D \bar{a} |\nabla t_{\boldsymbol{\nu}}|^2, \qquad d_{\boldsymbol{\nu},j} := \int_D |\psi_j| |\nabla t_{\boldsymbol{\nu}}|^2 \tag{4.23}$$

and note that, by Poincaré's inequality, for some $C > 0$ independent of $t_{\boldsymbol{\nu}}$

$$C \|t_{\boldsymbol{\nu}}\|_V^2 \le d_{\boldsymbol{\nu}} \le \|\bar{a}\|_{L^\infty(D)} \|t_{\boldsymbol{\nu}}\|_V^2. \tag{4.24}$$

Equation (4.22) with $\boldsymbol{\beta} = \mathbf{1}$ implies that for all $\boldsymbol{\nu} \in \mathcal{F}$

$$\sum_{j\in\mathbb{N}} d_{\boldsymbol{\nu},j} \le \theta d_{\boldsymbol{\nu}}. \tag{4.25}$$

Differentiation of Equation (4.2) for $u = u(\boldsymbol{y})$ by applying $\frac{1}{\boldsymbol{\nu}!}\partial^{\boldsymbol{\nu}}|_{\boldsymbol{y}=0}$ gives

$$\int_D \bar{a} \nabla t_{\boldsymbol{\nu}} \cdot \nabla v = - \sum_{j\in\operatorname{supp}\boldsymbol{\nu}} \int_D \psi_j \nabla t_{\boldsymbol{\nu}-\boldsymbol{e}_j} \cdot \nabla v, \quad \forall v \in V. \tag{4.26}$$

Using $v = t_{\boldsymbol{\nu}} \in V$ as test function and using Young's inequality, we get

$$\begin{aligned} d_{\boldsymbol{\nu}} = \int_D \bar{a} |\nabla t_{\boldsymbol{\nu}}|^2 &\le \sum_{j\in\operatorname{supp}\boldsymbol{\nu}} \int_D |\psi_j| |\nabla t_{\boldsymbol{\nu}-\boldsymbol{e}_j}| |\nabla t_{\boldsymbol{\nu}}| \\ &\le \sum_{j\in\operatorname{supp}\boldsymbol{\nu}} \int_D |\psi_j| (\tfrac{1}{2}|\nabla t_{\boldsymbol{\nu}-\boldsymbol{e}_j}|^2 + \tfrac{1}{2}|\nabla t_{\boldsymbol{\nu}}|^2) \\ &= \sum_{j\in\operatorname{supp}\boldsymbol{\nu}} \tfrac{1}{2}(d_{\boldsymbol{\nu}-\boldsymbol{e}_j,j} + d_{\boldsymbol{\nu},j}) \end{aligned} \tag{4.27}$$

and, by using Equation (4.25) twice, for all $k \in \mathbb{N}$

$$(1-\tfrac{\theta}{2})d_{\boldsymbol{\nu}} \overset{(4.25) \text{ and } (4.27)}{\leq} \tfrac{1}{2} \sum_{j \in \operatorname{supp} \boldsymbol{\nu}} d_{\boldsymbol{\nu}-\boldsymbol{e}_j, j},$$

$$(1-\tfrac{\theta}{2}) \sum_{|\boldsymbol{\nu}|_1 = k} d_{\boldsymbol{\nu}} \leq \tfrac{1}{2} \sum_{|\boldsymbol{\nu}|_1 = k} \sum_{j \in \operatorname{supp} \boldsymbol{\nu}} d_{\boldsymbol{\nu}-\boldsymbol{e}_j, j} = \tfrac{1}{2} \sum_{|\boldsymbol{\nu}|_1 = k-1} \sum_{j \in \mathbb{N}} d_{\boldsymbol{\nu}, j} \overset{(4.25)}{\leq} \tfrac{1}{2} \sum_{|\boldsymbol{\nu}|_1 = k-1} \theta d_{\boldsymbol{\nu}},$$

$$\sum_{|\boldsymbol{\nu}|_1 = k} d_{\boldsymbol{\nu}} \leq \tfrac{\theta}{2-\theta} \sum_{|\boldsymbol{\nu}|_1 = k-1} d_{\boldsymbol{\nu}}. \tag{4.28}$$

Writing $\kappa := \frac{\theta}{2-\theta}$, it follows from Equation (4.28) that

$$\sum_{\boldsymbol{\nu} \in \mathcal{F}} d_{\boldsymbol{\nu}} \leq \tfrac{1}{1-\kappa} d_{\mathbf{0}} < \infty. \tag{4.29}$$

It now follows from Equation (4.24) that

$$\sum_{\boldsymbol{\nu} \in \mathcal{F}} \|t_{\boldsymbol{\nu}}\|_V^2 < \infty. \tag{4.30}$$

This finishes the proof. □

## 4.4 Weighted summability of the $H^2$-norms of the Taylor gpc coefficients of the solution map

Under the assumption of more regular data, we show that the $H^2$-norms of the Taylor gpc coefficients of the solution map are weighted $\ell^2(\mathcal{F})$-summable, from which by Lemma 4.3 $\ell^p(\mathcal{F})$-summability follows for some $0 < p < 1$.

More precisely, let $f \in L^2(D)$ and assume that $a : U \to W^{1,\infty}(D)$ satisfies Equation (4.11) with $\bar{a} \in W^{1,\infty}(D)$ satisfying Equation (4.13) and with $\forall j \in \mathbb{N} : \psi_j \in W^{1,\infty}(D)$. By [33, Theorem 3.2.1.2 p. 147], the solution of Equation (4.2) is a strong solution of Equation (4.1), i.e. $u \in H^2(D)$. In [18, Section 5.1 pp. 39–40] the same is shown, as well as $\forall \boldsymbol{\nu} \in \mathcal{F} : t_{\boldsymbol{\nu}} \in H^2(D)$. For this situation, we have [3, Theorem 4.1 p. 2162], which is similar to Proposition 4.4. We now recall the theorem and its proof.

**Proposition 4.6** ([3, Theorem 4.1 p. 2162])**.** *Let $f \in L^2(D)$ and $a : U \to W^{1,\infty}(D)$ be given such that a satisfies Equation (4.11) with $\bar{a} \in W^{1,\infty}(D)$ satisfying Equation (4.13) and with $\forall j \in \mathbb{N} : \psi_j \in W^{1,\infty}(D)$. Let $\{t_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu} \in \mathcal{F}}$ be as in Equation (4.15). Assume that for $\boldsymbol{\beta} = (\beta_j)_{j \in \mathbb{N}} \subset \mathbb{R}_{>0}$*

$$\theta := \left\| \frac{\sum_{j \in \mathbb{N}} \beta_j^{-1} |\psi_j|}{\bar{a}} \right\|_{L^\infty(D)} < 1, \qquad \left\| \sum_{j \in \mathbb{N}} \beta_j^{-1} |\nabla \psi_j| \right\|_{L^\infty(D)} < \infty. \tag{4.31}$$

*Then, $(\boldsymbol{\beta}^{-\boldsymbol{\nu}} \|t_{\boldsymbol{\nu}}\|_{H^2})_{\boldsymbol{\nu} \in \mathcal{F}} \in \ell^2(\mathcal{F})$.*

**Remark 4.7** ([3, Proof Theorem 4.1 p. 2163], cf. [4, Theorem 2.2 p. 327])**.** *Analogous to Remark 4.5, for $\boldsymbol{\beta} \subset \mathbb{R}_{>0}$, Equations (4.11), (4.15) and (4.31) are equivalent to those* *equations for $\tilde{y}_j := \beta_j y_j$ instead of $y_j$, $\tilde{\psi}_j := \beta_j^{-1}\psi_j$ instead of $\psi_j$, $\tilde{t}_{\boldsymbol{\nu}} := \boldsymbol{\beta}^{-\boldsymbol{\nu}} t_{\boldsymbol{\nu}}$ instead of $t_{\boldsymbol{\nu}}$ and $\tilde{\beta}_j := 1$ instead of $\beta_j$, i.e. we can again assume $\boldsymbol{\beta} = \mathbf{1}$ without loss of generality. As before, the parameter set $\tilde{U} = \times_{j\in\mathbb{N}}[-\beta_j, \beta_j]$ from which $\tilde{\boldsymbol{y}}$ are taken is compact.*

*Proof of Proposition 4.6.* We follow [3, pp. 2163–2165]. By Remark 4.7, we may without loss of generality assume that $\boldsymbol{\beta} = \mathbf{1}$.

For $\boldsymbol{\nu} \in \mathcal{F}$ and $n \in \mathbb{N}_0$, we define

$$c_{\boldsymbol{\nu}} := \int_D \bar{a}|\Delta t_{\boldsymbol{\nu}}|^2, \qquad C_n := \sum_{|\boldsymbol{\nu}|_1 = n} c_{\boldsymbol{\nu}}, \qquad D_n := \sum_{|\boldsymbol{\nu}|_1 = n} d_{\boldsymbol{\nu}} \tag{4.32}$$

and note that for some $C > 0$ independent of $t_{\boldsymbol{\nu}}$

$$C\|t_{\boldsymbol{\nu}}\|_{H^2}^2 \le c_{\boldsymbol{\nu}} \le \|\bar{a}\|_{L^\infty(D)}\|t_{\boldsymbol{\nu}}\|_{H^2}^2 \tag{4.33}$$

(cf. [33, Theorem 3.1.3.1 p. 142 and the proof of Theorem 3.2.1.2 pp. 147–149]).

Differentiating Equation (4.1) for $u = u(\boldsymbol{y})$ by applying $\frac{1}{\boldsymbol{\nu}!}\partial^{\boldsymbol{\nu}}|_{\boldsymbol{y}=0}$, we find

$$-\bar{a}\Delta t_{\boldsymbol{\nu}} = \nabla\bar{a}\cdot\nabla t_{\boldsymbol{\nu}} + \sum_{j\in\operatorname{supp}\boldsymbol{\nu}} (\psi_j \Delta t_{\boldsymbol{\nu}-\boldsymbol{e}_j} + \nabla\psi_j\cdot\nabla t_{\boldsymbol{\nu}-\boldsymbol{e}_j}), \tag{4.34}$$

which again shows that $\forall\boldsymbol{\nu}\in\mathcal{F} : \Delta t_{\boldsymbol{\nu}} \in L^2(D)$. We can integrate Equation (4.34) against $\Delta t_{\boldsymbol{\nu}}$ and find, using Young's inequality for arbitrary $\varepsilon > 0$,

$$\begin{aligned} c_{\boldsymbol{\nu}} = \int_D \bar{a}|\Delta t_{\boldsymbol{\nu}}|^2 \le{}& \varepsilon\int_D |\nabla\bar{a}||\Delta t_{\boldsymbol{\nu}}|^2 + \tfrac{1}{4\varepsilon}\int_D |\nabla\bar{a}||\nabla t_{\boldsymbol{\nu}}|^2 \\ &+ \tfrac{1}{2}\sum_{j\in\operatorname{supp}\boldsymbol{\nu}}\int_D |\psi_j||\Delta t_{\boldsymbol{\nu}}|^2 + \tfrac{1}{2}\sum_{j\in\operatorname{supp}\boldsymbol{\nu}}\int_D |\psi_j||\Delta t_{\boldsymbol{\nu}-\boldsymbol{e}_j}|^2 \\ &+ \varepsilon\sum_{j\in\operatorname{supp}\boldsymbol{\nu}}\int_D |\nabla\psi_j||\Delta t_{\boldsymbol{\nu}}|^2 + \tfrac{1}{4\varepsilon}\sum_{j\in\operatorname{supp}\boldsymbol{\nu}}\int_D |\nabla\psi_j||\nabla t_{\boldsymbol{\nu}-\boldsymbol{e}_j}|^2. \end{aligned} \tag{4.35}$$

The sum of the first, the third and the fifth term is bounded by

$$\tfrac{\varepsilon}{\bar{a}_{\min}}\left\| |\nabla\bar{a}| + \sum_{j\in\operatorname{supp}\boldsymbol{\nu}} |\nabla\psi_j| \right\|_{L^\infty(D)} c_{\boldsymbol{\nu}} + \tfrac{\theta}{2}c_{\boldsymbol{\nu}} \le (\tfrac{\theta}{2} + B\varepsilon)c_{\boldsymbol{\nu}},$$

where $B := \frac{1}{\bar{a}_{\min}}\left\| |\nabla\bar{a}| + \sum_{j\in\mathbb{N}} |\nabla\psi_j| \right\|_{L^\infty(D)} < \infty$. Regarding the second and the sixth term in Equation (4.35),

$$\begin{aligned} &\sum_{|\boldsymbol{\nu}|_1=n}\left(\tfrac{1}{4\varepsilon}\int_D |\nabla\bar{a}||\nabla t_{\boldsymbol{\nu}}|^2 + \tfrac{1}{4\varepsilon}\sum_{j\in\operatorname{supp}\boldsymbol{\nu}}\int_D |\nabla\psi_j||\nabla t_{\boldsymbol{\nu}-\boldsymbol{e}_j}|^2\right) \\ \le{}& \tfrac{B}{4\varepsilon}D_n + \tfrac{1}{4\varepsilon}\sum_{|\boldsymbol{\nu}|_1=n-1}\sum_{j\in\mathbb{N}}\int_D |\nabla\psi_j||\nabla t_{\boldsymbol{\nu}}|^2 \le \tfrac{B}{4\varepsilon}(D_n + D_{n-1}). \end{aligned}$$

Regarding the fourth term in Equation (4.35),

$$\sum_{|\boldsymbol{\nu}|_1=n}\tfrac{1}{2}\sum_{j\in\operatorname{supp}\boldsymbol{\nu}}\int_D |\psi_j||\Delta t_{\boldsymbol{\nu}-\boldsymbol{e}_j}|^2 \le \tfrac{\theta}{2}C_{n-1}.$$

Summing over $\{\boldsymbol{\nu} \in \mathcal{F} : |\boldsymbol{\nu}|_1 = n\}$ in Equation (4.35), we get

$$C_n \leq (\tfrac{\theta}{2} + \varepsilon B)C_n + \tfrac{\theta}{2}C_{n-1} + \tfrac{B}{4\varepsilon}(D_n + D_{n-1}).$$

For $\varepsilon > 0$ such that $\frac{\theta}{2} + \varepsilon B < \frac{1}{2}$, we have

$$\tau := \frac{\theta}{2 - \theta - 2\varepsilon B} < 1,\ A := \frac{B}{2\varepsilon(2 - \theta - 2\varepsilon B)}, \qquad C_n \leq \tau C_{n-1} + A(D_n + D_{n-1}). \quad (4.36)$$

We recall from the proof of Proposition 4.4 that $\forall n \in \mathbb{N} : D_n \leq \kappa D_{n-1}$ with $\kappa = \frac{\theta}{2-\theta}$. Taking $\delta$ such that $\kappa \leq \tau < \delta < 1$, it follows from Equation (4.36) that

$$\begin{aligned} C_n &\overset{(4.36)}{\leq} AD_0(1 + \kappa^{-1})\kappa^n + \tau C_{n-1} \\ &\overset{(4.36)}{\leq} AD_0(1 + \kappa^{-1})(\kappa^n + \tau\kappa^{n-1}) + \tau^2 C_{n-2} \\ &\overset{(4.36)}{\leq} AD_0(1 + \kappa^{-1}) \left( \sum_{j=1}^{n} (\kappa/\tau)^j \right) \tau^n + \tau^n C_0 \\ &\leq \left[AD_0(1 + \kappa^{-1})(1 - \kappa/\delta)^{-1} + C_0\right] \delta^n. \end{aligned}$$

Summing over $n \in \mathbb{N}_0$, we find

$$\sum_{\boldsymbol{\nu} \in \mathcal{F}} c_{\boldsymbol{\nu}} = \sum_{n \in \mathbb{N}_0} C_n \leq \left[AD_0(1 + \kappa^{-1})(1 - \kappa/\delta)^{-1} + C_0\right](1 - \delta)^{-1} < \infty.$$

Together with Equation (4.33), this finishes the proof. □

**Remark 4.8.** *Note that Propositions 4.3 and 4.6 show that* $(\|t_{\boldsymbol{\nu}}\|_{H^2(D)})_{\boldsymbol{\nu} \in \mathcal{F}} \in \ell^p(\mathcal{F})$, *independent of any choice of index sets* $\{\Lambda_n\}_{n \in \mathbb{N}}$.

**Remark 4.9** ([3, Remark 2.2 p. 2156])**.** *We note that in Lemma 4.3 the required summability* $(\boldsymbol{\beta}^{-\boldsymbol{\nu}} b_{\boldsymbol{\nu}})_{\boldsymbol{\nu} \in \mathcal{F}} \in \ell^2(\mathcal{F})$ *poses stronger restrictions on* $\boldsymbol{\beta}$ *when* $b_{\boldsymbol{\nu}} = \|t_{\boldsymbol{\nu}}\|_{H^2}$ *than when* $b_{\boldsymbol{\nu}} := \|t_{\boldsymbol{\nu}}\|_V$. *Hence, the best possible value of* $q$ *for the summability of the* $H^2$*-norms will be at least as large as the best possible value of* $q$ *for the summability of the* $V$*-norms. This implies that the same holds for the values of* $p$, *i.e. the summability of the* $V$*-norms is at least as good as the summability of the* $H^2$*-norms, as expected.*

**Remark 4.10.** *In the derivation of Equation (4.21), we never explicitly used that the norms of* $u - u_n$ *and* $t_{\boldsymbol{\nu}}$ *were calculated with respect to the* $V$*-norm. That is, by exactly the same arguments as used there, we find that under the conditions of Proposition 4.6*

$$\|u - u_n\|_{L^\infty(U, H^2(D))} \leq \sum_{\boldsymbol{\nu} \in \Lambda_n^c} \|t_{\boldsymbol{\nu}}\|_{H^2(D)} \leq Cn^{-1/p+1}. \quad (4.37)$$

*By Remark 4.9,* $p$ *for the summability of the* $V$*-norms is at least as small as* $p$ *for the summability of the* $H^2$*-norms, i.e. the convergence rate in Equation (4.21) is at least as good as the convergence rate in Equation (4.37).*

*Note that the index sets* $\{\Lambda_n\}_{n \in \mathbb{N}}$ *constructed in Lemma 4.3 for* $\forall \boldsymbol{\nu} \in \mathcal{F} : b_{\boldsymbol{\nu}} = \|t_{\boldsymbol{\nu}}\|_{H^2}$ *may differ from those constructed in Lemma 4.3 for* $\forall \boldsymbol{\nu} \in \mathcal{F} : b_{\boldsymbol{\nu}} = \|t_{\boldsymbol{\nu}}\|_V$. *In Section 5, we will choose* $\{\Lambda_n\}_{n \in \mathbb{N}}$ *based on the weighted* $\ell^2(\mathcal{F})$*-summability of* $(\|t_{\boldsymbol{\nu}}\|_V)_{\boldsymbol{\nu} \in \mathcal{F}}$. *Hence, in Section 5, Equation (4.37) does not necessarily hold.*

## 5 ReLU DNN approximation of the solution map of a parametric diffusion equation

We now use the obtained results to show Theorem 5.1 below concerning the approximation of the solution map $\boldsymbol{y} \mapsto u(\boldsymbol{y})$ of Equation (4.1) under the assumptions made in Section 4.4. It is a generalisation of [75, Theorem 4.8 p. 22] stated in Section 1.3, for $D = (0,1)^2$ instead of $D = (0,1)$ used in [75]. The main difference is that on $D = (0,1)$ the continuous, piecewise linear interpolants of $\{t_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\mathcal{F}}$ could be implemented exactly by ReLU networks, cf. [75, Lemma 4.5 p. 20]. For the continuous, piecewise bilinear interpolants introduced in Section 3.1 that is not possible, as ReLU networks can only implement continuous, piecewise linear functions, see [55, Proposition 4 p. 6], Lemma 2.3 and Remark B.1. Instead, we use the ReLU approximations of continuous, piecewise bilinear interpolants discussed in Sections 3.2 and 3.3.

Throughout this section, we will write $D = (0,1)^2$, $V = H_0^1(D)$ and

$$X := H^2 \cap H_0^1(D). \tag{5.1}$$

**Theorem 5.1** (Generalisation of [75, Theorem 4.8 p. 22])**.** *We consider Equation (4.1) for given $f \in L^2(D)$ and $a : U \to W^{1,\infty}(D)$ satisfying Equation (4.11) with $\bar{a} \in W^{1,\infty}(D)$ satisfying Equation (4.13) and $\forall j \in \mathbb{N} : \psi_j \in W^{1,\infty}(D)$. We assume that Equation (4.22) holds with a decreasing sequence $\boldsymbol{\beta}_V \in (0,1)^{\mathbb{N}}$ satisfying $\boldsymbol{\beta}_V \in \ell^{q_V}(\mathbb{N})$ for $q_V = (\frac{1}{p_V} - \frac{1}{2})^{-1}$ for some $0 < p_V < 1$ and that Equation (4.31) holds with a decreasing sequence $\boldsymbol{\beta}_X \in (0,1)^{\mathbb{N}}$ satisfying $\boldsymbol{\beta}_X \in \ell^{q_X}(\mathbb{N})$ for $q_X = (\frac{1}{p_X} - \frac{1}{2})^{-1}$ for some $0 < p_V \leq p_X < 1$.*

*Then, for each $n \in \mathbb{N}$, there exists a ReLU network $\tilde{u}_n \in \mathcal{R}$ of size $\mathcal{N}_n^* \geq n$ approximating the solution map $u : U \to X : \boldsymbol{y} \mapsto u(\boldsymbol{y})$ of the parametric PDE in Equation (4.1), having $n+2$ inputs, denoted by $x_1, x_2, y_1, \ldots, y_n$, such that $\tilde{u}_n$ satisfies*

$$\sup_{\boldsymbol{y}\in U} \|u(\cdot,\cdot,\boldsymbol{y}) - R_\sigma(\tilde{u}_n)(\cdot,\cdot,y_1,\ldots,y_n)\|_V = \mathcal{O}\left((\mathcal{N}_n^*)^{-r^*}\right) \tag{5.2}$$

*for any $r^*$ that satisfies, for arbitrary $0 < \gamma < \frac{1}{2}$,*

$$0 < r^* < r := \gamma \min\left\{1, \frac{1/p_V - 1}{\gamma + 1/p_V - 1/p_X}\right\}. \tag{5.3}$$

*In addition,*

$$\operatorname{depth}(\tilde{u}_n) = \mathcal{O}(\log(\mathcal{N}_n^*) \log\log(\mathcal{N}_n^*))$$

*and the number of non-zero coefficients is of the order $\mathcal{O}(\mathcal{N}_n^*)$.*

In what follows, we will often suppress the inputs $x_1$ and $x_2$ of $\tilde{u}_n$ in notation and write $R_\sigma(\tilde{u}_n)(y_1,\ldots,y_n) \in V$.

**Remark 5.2** (Cf. [3, p. 2160])**.** *Note that the convergence rate $r^*$ holds for parametric PDEs with countably many parameters, i.e. in that sense $\{\tilde{u}_n\}_{n\in\mathbb{N}}$ do not suffer from the curse of dimensionality.*

*In addition, note that $r \leq \gamma < \frac{1}{2}$ and that $r = \gamma$ if $1/p_X - \gamma \geq 1$, which is equivalent to $p_X \leq (1+\gamma)^{-1}$ and to $1/p_X - 1 \geq \gamma$. As we will see in the proof, in this case, the rate $r = \gamma$ is determined by the convergence rate of the spatial ReLU approximations of $(t_{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}}$. Conversely, we have $r < \gamma$ if $1/p_X - 1 < \gamma$. In addition, in Remark 4.9 it is argued that the assumption $p_V \leq p_X$ poses no loss of generality. The assumption implies that $r \leq 1/p_V - 1$. The value $1/p_V - 1$ is the convergence rate of the best $n$-term approximation of $u$, see Equation (4.17) and Remark B.3. That is, $r$ is bounded from above by $\gamma$ and $1/p_V - 1$, which are the convergence rate of the ReLU approximations of the Taylor gpc coefficients and the best $n$-term convergence rate of $u_n$.*

*If the requirements of the theorem hold for some $p_X \in (0,1)$, we can always take $p_V$ equal to $p_X$ (see Remark 4.9). Interestingly, if $p_V < p_X$ we get a better lower bound on the convergence rate than if $p_V = p_X$: If $1/p_X - 1 < \gamma$, then it holds that $0 < \frac{1/p_V - 1}{\gamma + 1/p_V - 1/p_X} < 1$, hence $0 < (1/p_V - 1/p_X) < \min\{1/p_V - 1, \gamma + 1/p_V - 1/p_X\}$ implies that*

$$\gamma\frac{1/p_V - 1}{\gamma + 1/p_V - 1/p_X} > \gamma\frac{1/p_V - 1 - (1/p_V - 1/p_X)}{\gamma + 1/p_V - 1/p_X - (1/p_V - 1/p_X)} = 1/p_X - 1,$$

*where $1/p_X - 1$ is the lower bound on the convergence rate if we take $p_V = p_X$.*

For the proof of the theorem, we recall from Sections 4.2, 4.3 and 4.4 that $(\boldsymbol{\beta}_V^{-\boldsymbol{\nu}}\|t_{\boldsymbol{\nu}}\|_V)_{\boldsymbol{\nu}\in\mathcal{F}}, (\boldsymbol{\beta}_X^{-\boldsymbol{\nu}}\|t_{\boldsymbol{\nu}}\|_X)_{\boldsymbol{\nu}\in\mathcal{F}} \in \ell^2(\mathcal{F})$, that $(\|t_{\boldsymbol{\nu}}\|_V)_{\boldsymbol{\nu}\in\mathcal{F}} \in \ell^{p_V}(\mathcal{F})$ and that $(\|t_{\boldsymbol{\nu}}\|_X)_{\boldsymbol{\nu}\in\mathcal{F}} \in \ell^{p_X}(\mathcal{F})$. In addition, Equation (4.31) and $\boldsymbol{\beta}_X \in (0,1)^{\mathbb{N}}$ imply that $\sup_{\boldsymbol{y}\in U}\|a(\boldsymbol{y})\|_{W^{1,\infty}(D)} < \infty$.

We need the following lemma, which is adapted from [75, Lemma 4.7 p. 21].

**Lemma 5.3** ([75, Lemma 4.7 p. 21]). *Let $m_0$ be as defined in Section 3.3, assume the conditions of Theorem 5.1 and let $\{\Lambda_n\}_{n\in\mathbb{N}}$ be as in Lemma 4.3 with $\forall\boldsymbol{\nu}\in\mathcal{F} : b_{\boldsymbol{\nu}} = \|t_{\boldsymbol{\nu}}\|_V$.*

*Then, there exists a $C > 0$ such that for all $n \in \mathbb{N}$ there exists a sequence $(m_{n;\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\Lambda_n} \in \mathbb{N}_{\geq m_0}^{\Lambda_n}$ such that with $\mathcal{N}_n := \sum_{\boldsymbol{\nu}\in\Lambda_n} m_{n;\boldsymbol{\nu}} \geq n$ we have*

$$\sum_{\boldsymbol{\nu}\in\Lambda_n}\|t_{\boldsymbol{\nu}}\|_X m_{n;\boldsymbol{\nu}}^{-\gamma} + \sum_{\boldsymbol{\nu}\in\Lambda_n^c}\|t_{\boldsymbol{\nu}}\|_V \leq Cn^{-1/p_V+1} \leq C\mathcal{N}_n^{-r} \tag{5.4}$$

*for $r$ as defined in Equation (5.3) for arbitrary $0 < \gamma < \frac{1}{2}$.*

The proof of this lemma is given in Appendix B.6. The main difference with respect to [75, Lemma 4.7 p. 21] is that we explicitly choose $m_{n;\boldsymbol{\nu}} \geq m_0$ for all $n \in \mathbb{N}$ and $\boldsymbol{\nu} \in \Lambda_n$.

*Proof of Theorem 5.1.* We closely follow [75, pp. 22–25]. Let $n \in \mathbb{N}$. Throughout this proof, let the index set $\Lambda_n$ be as in Lemma 4.3 with $\forall\boldsymbol{\nu}\in\mathcal{F} : b_{\boldsymbol{\nu}} = \|t_{\boldsymbol{\nu}}\|_V$.

The proof consists of five steps. In the first step, we estimate the errors of ReLU approximations of $\{t_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\Lambda_n}$ and $\{\boldsymbol{y} \mapsto \boldsymbol{y}^{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\Lambda_n}$. In the second step, we define the network $\tilde{u}_n$, which is an approximation of $u_n$ as defined in Equation (4.16). In the third step, its error is estimated. In the fourth step, which is stated in Appendix B.7, bounds on the depth, the size and the number of non-zero coefficients are calculated. In the fifth step, stated in this section, a lower bound on the convergence rate of $\tilde{u}_n$ in terms of the network size is given.

*Step 1.* Let $m_0$ be as defined in Section 3.3 and let $\{m_{n;\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\Lambda_n}$ be as given by Lemma 5.3. Using that $\forall\boldsymbol{\nu}\in\Lambda_n : m_{n;\boldsymbol{\nu}} \geq m_0$, we get from Proposition 3.5 that for any $\gamma < \frac{1}{2}$ it holds that $\forall\boldsymbol{\nu}\in\Lambda_n : \|t_{\boldsymbol{\nu}} - R_\sigma(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}})\|_V \leq C\|t_{\boldsymbol{\nu}}\|_X m_{n;\boldsymbol{\nu}}^{-\gamma}$, where $C>0$ is independent of $t_{\boldsymbol{\nu}}$, $\boldsymbol{\nu}$ and $n$. Hence, by Lemma 5.3,

$$\sum_{\boldsymbol{\nu}\in\Lambda_n} \|t_{\boldsymbol{\nu}} - R_\sigma(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}})\|_V \overset{(3.21)}{\leq} C \sum_{\boldsymbol{\nu}\in\Lambda_n} \|t_{\boldsymbol{\nu}}\|_X m_{n;\boldsymbol{\nu}}^{-\gamma} \overset{(5.4)}{\leq} Cn^{-1/p_V+1}. \tag{5.5}$$

Moreover, by Lemma 4.3, we have $\bigcup_{\boldsymbol{\nu}\in\Lambda_n}\{y_j : j\in\operatorname{supp}\boldsymbol{\nu}\} \subset \{y_1,\dots,y_n\}$. This allows us to use Lemma 2.17 with $\forall\boldsymbol{\nu}\in\mathcal{F} : b_{\boldsymbol{\nu}} = \|t_{\boldsymbol{\nu}}\|_V$ and $p_b = p_V$, which shows that there exist ReLU networks $\{f_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\Lambda_n}$, all having inputs $y_1,\dots,y_n$, such that

$$\begin{aligned}
&\sup_{\boldsymbol{y}\in U} \left\| \sum_{\boldsymbol{\nu}\in\Lambda_n} t_{\boldsymbol{\nu}} \boldsymbol{y}^{\boldsymbol{\nu}} - \sum_{\boldsymbol{\nu}\in\Lambda_n} t_{\boldsymbol{\nu}} R_\sigma(f_{\boldsymbol{\nu}})(y_1,\dots,y_n) \right\|_V \\
&\leq \sup_{\boldsymbol{y}\in U} \sum_{\boldsymbol{\nu}\in\Lambda_n} \|t_{\boldsymbol{\nu}}\|_V |\boldsymbol{y}^{\boldsymbol{\nu}} - R_\sigma(f_{\boldsymbol{\nu}})(y_1,\dots,y_n)| \overset{(2.42)}{\leq} n^{-1/p_V+1}
\end{aligned} \tag{5.6}$$

and such that Equations (2.43)–(2.46) hold.

*Step 2.* With $X$ as defined in Equation (5.1), we use boundedness of the embedding $X \hookrightarrow L^\infty(D)$ and define

$$\begin{aligned}
M :=& \max\left\{1, 2\sup_{\boldsymbol{\nu}\in\mathcal{F}} \|t_{\boldsymbol{\nu}}\|_{L^\infty(D)}\right\} \\
\leq& \max\left\{1, 2\sup_{\boldsymbol{\nu}\in\mathcal{F}} C\|t_{\boldsymbol{\nu}}\|_X\right\} \\
\overset{(*)}{\leq}& \max\left\{1, 2C\left(\sum_{\boldsymbol{\nu}\in\mathcal{F}} (\boldsymbol{\beta}_X^{-\boldsymbol{\nu}} \|t_{\boldsymbol{\nu}}\|_X)^2\right)^{1/2}\right\} \overset{\text{Proposition 4.6}}{<} \infty,
\end{aligned}$$

where at (*) we used $\forall\boldsymbol{\nu}\in\mathcal{F} : \boldsymbol{\beta}_X^{-\boldsymbol{\nu}} \geq 1$. By Equation (2.43), we have $\sup_{\boldsymbol{y}\in U,\boldsymbol{\nu}\in\Lambda_n} |R_\sigma(f_{\boldsymbol{\nu}})((y_j)_{j\in\operatorname{supp}\boldsymbol{\nu}})| \leq M$. Using Equations (3.7) and (3.23), we find

$$\begin{aligned}
\|R_\sigma(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}})\|_{L^\infty(D)} &\leq \|I_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}}\|_{L^\infty(D)} + \|I_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}} - R_\sigma(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}})\|_{L^\infty(D)} \\
&\overset{(3.23)}{\leq} 2\|I_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}}\|_{L^\infty(D)} \\
&\overset{(3.7)}{\leq} 2\|t_{\boldsymbol{\nu}}\|_{L^\infty(D)} \\
&\leq M.
\end{aligned}$$

We will now define $\tilde{u}_n \in \mathcal{R}$ such that

$$R_\sigma(\tilde{u}_n)(x_1,x_2,y_1,\dots,y_n) = \sum_{\boldsymbol{\nu}\in\Lambda_n} R_\sigma(\tilde{\times})\big(R_\sigma(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}})(\boldsymbol{x}), R_\sigma(f_{\boldsymbol{\nu}})(y_1,\dots,y_n)\big), \qquad \boldsymbol{x}\in D, \boldsymbol{y}\in U. \tag{5.7}$$

We first define the matrix $\mathrm{pr}_{\tilde{u}_n} \in \mathbb{R}^{2n\times 2n}$ to connect the output of $\{\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\Lambda_n} \cup \{f_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\Lambda_n}$ to the input of the $\tilde{\times}$-networks.

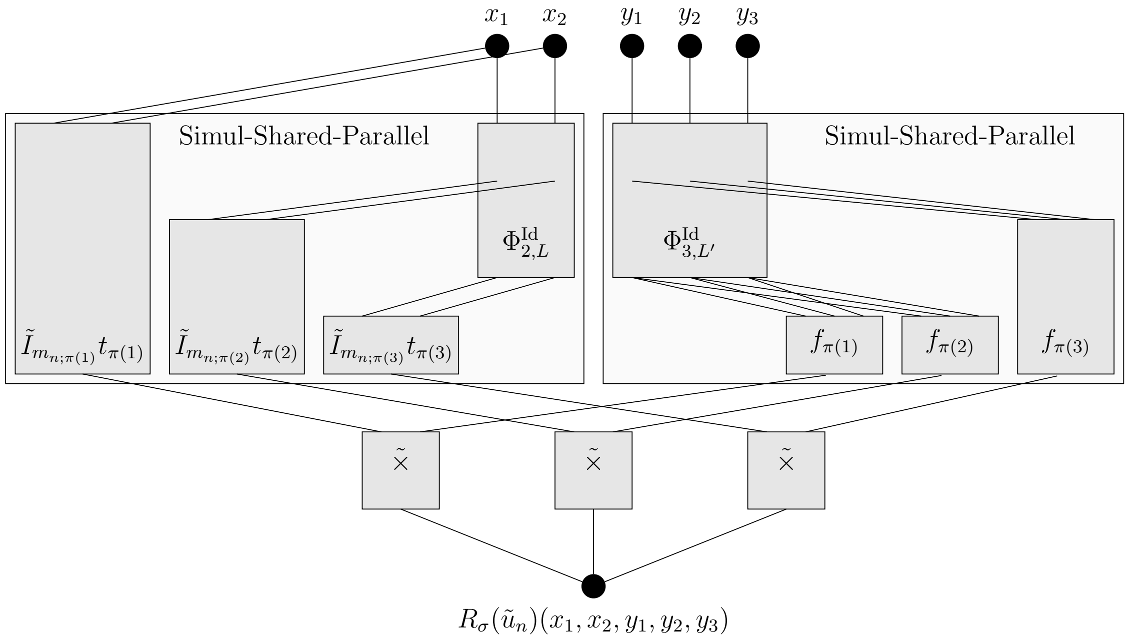
$$(\mathrm{pr}_{\tilde{u}_n})_{i,j} := \begin{cases} 1 & 1 \le j \le n, i = 2j-1, \\ 1 & n+1 \le j \le 2n, i = 2(j-n), \\ 0 & \text{else.} \end{cases}$$

For a bijection $\pi : \{1, \dots, n\} \to \Lambda_n$ and for Simul-Shared-Parallel as constructed in Remark A.5, we define

$$\begin{aligned} P_{\tilde{u}_n} &:= \text{Parallel}\,\big(\text{Simul-Shared-Parallel}(\tilde{I}_{m_{n;\pi(1)}} t_{\pi(1)}, \dots, \tilde{I}_{m_{n;\pi(n)}} t_{\pi(n)}), \\ &\qquad\qquad \text{Simul-Shared-Parallel}(f_{\pi(1)}, \dots, f_{\pi(n)})\big), \\ \tilde{u}_n &:= \Big(\begin{pmatrix} 1 & \cdots & 1 \end{pmatrix}, \mathbf{0}\Big) \bullet \text{Parallel}(\tilde{\times}, \dots, \tilde{\times}) \bullet (\mathrm{pr}_{\tilde{u}_n}, \mathbf{0}) \odot P_{\tilde{u}_n}, \end{aligned} \tag{5.8}$$

where the $\tilde{\times}$-subnetworks are as in Proposition 2.12 with maximum input size $M$ and accuracy $\delta_n := n^{-1/p_V}$ in the sense of Equations (2.19)–(2.20), i.e. all those $\tilde{\times}$-subnetworks are identical. It follows from the estimates in the beginning of Step 2 of this proof that maximum input size $M$ suffices.

The structure of the network is depicted in Figure 5.1.



$x_1$ $x_2$ $y_1$ $y_2$ $y_3$
Simul-Shared-Parallel
Simul-Shared-Parallel
$\Phi^{\mathrm{Id}}_{2,L}$
$\Phi^{\mathrm{Id}}_{3,L'}$
$\tilde{I}_{m_{n;\pi(1)}} t_{\pi(1)}$
$\tilde{I}_{m_{n;\pi(2)}} t_{\pi(2)}$
$\tilde{I}_{m_{n;\pi(3)}} t_{\pi(3)}$
$f_{\pi(1)}$
$f_{\pi(2)}$
$f_{\pi(3)}$
$\tilde{\times}$
$\tilde{\times}$
$\tilde{\times}$
$R_\sigma(\tilde{u}_n)(x_1, x_2, y_1, y_2, y_3)$

Figure 5.1: Structure of $\tilde{u}_3$; the sizes of the subnetworks $\{\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\Lambda_3} \cup \{f_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\Lambda_3} \cup \{\Phi^{\mathrm{Id}}_{2,L}, \Phi^{\mathrm{Id}}_{3,L'}\}$ depend on the approximated function $u$, this figure depicts a possible situation. By construction of Simul-Shared-Parallel in Remark A.5, there is only one identity network per Simul-Shared-Parallel-subnetwork. Moreover, some networks take the output of a hidden layer of one of the identity networks as input.

Equation (5.7) follows from Equation (5.8), Remark A.3, Lemma 2.8 and Equations (A.23) and (A.15) (see Remark A.5 for the fact that Simul-Shared-Parallel also satisfies Equation (A.15)).

For $\boldsymbol{\nu} \in \Lambda_n$, according to Lemma 2.17, the realisation $R_\sigma(f_{\boldsymbol{\nu}})$ is constant in $\{y_j : j \in \{1, \dots, n\} \setminus \operatorname{supp} \boldsymbol{\nu}\}$. Hence, $R_\sigma(\tilde{u}_n)$ is constant in $\{y_j : j \in \{1, \dots, n\} \setminus \bigcup_{\boldsymbol{\nu} \in \Lambda_n} \operatorname{supp} \boldsymbol{\nu}\}$, i.e. those inputs could be left out.

*Step 3.* We begin the error estimate by estimating the $H^1$-error made by the $\tilde{\times}$-subnetworks appearing in Equation (5.8).

By Lemma 2.3, the functions $R_\sigma(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}})$ and $R_\sigma(\tilde{u}_n)$ are continuous and piecewise linear on a finite partition of their domain. Combined with Proposition 2.12, it shows that for all $\boldsymbol{y} \in U$ and $n \in \mathbb{N}$ at almost all $\boldsymbol{x} \in D$ those functions are strongly differentiable w.r.t. $x_1$ and $x_2$, hence so is the expression

$$\sum_{\boldsymbol{\nu} \in \Lambda_n} R_\sigma(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}})(\boldsymbol{x}) R_\sigma(f_{\boldsymbol{\nu}})(y_1, \dots, y_n) - \sum_{\boldsymbol{\nu} \in \Lambda_n} R_\sigma(\tilde{\times})\big(R_\sigma(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}})(\boldsymbol{x}), R_\sigma(f_{\boldsymbol{\nu}})(y_1, \dots, y_n)\big).$$

As a result, Equation (2.20) leads to the following estimate. For $\boldsymbol{y} \in U$ and $\boldsymbol{\nu} \in \mathcal{F}$, with $\frac{d}{dx_1}$ denoting weak derivatives that are strong at almost every $\boldsymbol{x} \in D$, it holds that

$$\begin{aligned}
&\bigg| \frac{d}{dx_1} \Big( R_\sigma(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}})(\boldsymbol{x}) R_\sigma(f_{\boldsymbol{\nu}})(y_1, \dots, y_n) \Big) \\
&\quad - \frac{d}{dx_1} \Big( R_\sigma(\tilde{\times})\big(R_\sigma(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}})(\boldsymbol{x}), R_\sigma(f_{\boldsymbol{\nu}})(y_1, \dots, y_n)\big) \Big) \bigg| \\
\leq &\bigg| R_\sigma(f_{\boldsymbol{\nu}})(y_1, \dots, y_n) \frac{d}{dx_1} \Big( R_\sigma(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}})(\boldsymbol{x}) \Big) \\
&\quad - \frac{d}{da}\bigg|_{a = R_\sigma(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}})(\boldsymbol{x})} \Big( R_\sigma(\tilde{\times})\big(a, R_\sigma(f_{\boldsymbol{\nu}})(y_1, \dots, y_n)\big) \Big) \frac{d}{dx_1} \Big( R_\sigma(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}})(\boldsymbol{x}) \Big) \bigg| \\
\overset{(2.20)}{\leq} &\ \delta_n \left| \frac{d}{dx_1} R_\sigma(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}})(\boldsymbol{x}) \right|, \quad \boldsymbol{x} \in D,
\end{aligned}$$

$$\begin{aligned}
&\bigg\| \frac{d}{dx_1} \Big( R_\sigma(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}})(\boldsymbol{x}) R_\sigma(f_{\boldsymbol{\nu}})(y_1, \dots, y_n) \Big) \\
&\quad - \frac{d}{dx_1} \Big( R_\sigma(\tilde{\times})\big(R_\sigma(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}})(\boldsymbol{x}), R_\sigma(f_{\boldsymbol{\nu}})(y_1, \dots, y_n)\big) \Big) \bigg\|_{L^2(D)} \\
\leq &\,\delta_n \left\| \frac{d}{dx_1} R_\sigma(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}})(\boldsymbol{x}) \right\|_{L^2(D)}.
\end{aligned}$$

Using the analogous statement for $\frac{d}{dx_2}$ and Equation (2.19), it follows that

$$\begin{aligned}
&\Big\| R_\sigma(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}}) R_\sigma(f_{\boldsymbol{\nu}})(y_1, \dots, y_n) - R_\sigma(\tilde{\times})\big(R_\sigma(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}}), R_\sigma(f_{\boldsymbol{\nu}})(y_1, \dots, y_n)\big) \Big\|_V \\
\overset{(2.19)}{\leq} &\ \delta_n (1 + \|R_\sigma(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}})\|_V),
\end{aligned}$$

$$
\begin{aligned}
&\sup_{\boldsymbol{y}\in U}\Bigg\|\sum_{\boldsymbol{\nu}\in\Lambda_n} R_\sigma(\tilde{I}_{m_n;\boldsymbol{\nu}}t_{\boldsymbol{\nu}})R_\sigma(f_{\boldsymbol{\nu}})(y_1,\dots,y_n)\\
&\quad-\sum_{\boldsymbol{\nu}\in\Lambda_n} R_\sigma(\tilde{\times})\big(R_\sigma(\tilde{I}_{m_n;\boldsymbol{\nu}}t_{\boldsymbol{\nu}}),R_\sigma(f_{\boldsymbol{\nu}})(y_1,\dots,y_n)\big)\Bigg\|_V\\
&\le\sup_{\boldsymbol{y}\in U}\sum_{\boldsymbol{\nu}\in\Lambda_n}\Big\|R_\sigma(\tilde{I}_{m_n;\boldsymbol{\nu}}t_{\boldsymbol{\nu}})R_\sigma(f_{\boldsymbol{\nu}})(y_1,\dots,y_n)\\
&\quad-R_\sigma(\tilde{\times})\big(R_\sigma(\tilde{I}_{m_n;\boldsymbol{\nu}}t_{\boldsymbol{\nu}}),R_\sigma(f_{\boldsymbol{\nu}})(y_1,\dots,y_n)\big)\Big\|_V\\
&\le\sup_{\boldsymbol{y}\in U}|\Lambda_n|\delta_n(1+\sup_{\boldsymbol{\nu}\in\mathcal{F}}\|R_\sigma(\tilde{I}_{m_n;\boldsymbol{\nu}}t_{\boldsymbol{\nu}})\|_V)\\
&\overset{(*)}{\le} Cn\delta_n=Cn^{-1/p_V+1}.
\end{aligned}\tag{5.9}
$$

At (*) we have used that Equation (5.5) implies the following uniform bound for all $n\in\mathbb{N}$ and $\boldsymbol{\nu}\in\Lambda_n$:

$$
\begin{aligned}
\|R_\sigma(\tilde{I}_{m_n;\boldsymbol{\nu}}t_{\boldsymbol{\nu}})\|_V&\le\|t_{\boldsymbol{\nu}}\|_V+\|t_{\boldsymbol{\nu}}-R_\sigma(\tilde{I}_{m_n;\boldsymbol{\nu}}t_{\boldsymbol{\nu}})\|_V\\
&\overset{(5.5)}{\le}\|(\|t_{\boldsymbol{\nu}}\|_V)_{\boldsymbol{\nu}\in\mathcal{F}}\|_{\ell^2(\mathcal{F})}+Cn^{-1/p_V+1}\le\|(\|t_{\boldsymbol{\nu}}\|_V)_{\boldsymbol{\nu}\in\mathcal{F}}\|_{\ell^2(\mathcal{F})}+C<\infty.
\end{aligned}
$$

All together, we have the following bound on the $H^1$-error:

$$
\begin{aligned}
\|u-R_\sigma(\tilde{u}_n)\|_{L^\infty(U,V)}&=\sup_{\boldsymbol{y}\in U}\|u(\boldsymbol{y})-R_\sigma(\tilde{u}_n)(y_1,\dots,y_n)\|_V\\
&\le\sup_{\boldsymbol{y}\in U}\Bigg\|\sum_{\boldsymbol{\nu}\in\mathcal{F}}t_{\boldsymbol{\nu}}\boldsymbol{y}^{\boldsymbol{\nu}}-\sum_{\boldsymbol{\nu}\in\Lambda_n}t_{\boldsymbol{\nu}}\boldsymbol{y}^{\boldsymbol{\nu}}\Bigg\|_V\\
&\quad+\sup_{\boldsymbol{y}\in U}\Bigg\|\sum_{\boldsymbol{\nu}\in\Lambda_n}t_{\boldsymbol{\nu}}\boldsymbol{y}^{\boldsymbol{\nu}}-\sum_{\boldsymbol{\nu}\in\Lambda_n}t_{\boldsymbol{\nu}}R_\sigma(f_{\boldsymbol{\nu}})(y_1,\dots,y_n)\Bigg\|_V\\
&\quad+\sup_{\boldsymbol{y}\in U}\Bigg\|\sum_{\boldsymbol{\nu}\in\Lambda_n}t_{\boldsymbol{\nu}}R_\sigma(f_{\boldsymbol{\nu}})(y_1,\dots,y_n)\\
&\quad-\sum_{\boldsymbol{\nu}\in\Lambda_n}R_\sigma(\tilde{I}_{m_n;\boldsymbol{\nu}}t_{\boldsymbol{\nu}})R_\sigma(f_{\boldsymbol{\nu}})(y_1,\dots,y_n)\Bigg\|_V\\
&\quad+\sup_{\boldsymbol{y}\in U}\Bigg\|\sum_{\boldsymbol{\nu}\in\Lambda_n}R_\sigma(\tilde{I}_{m_n;\boldsymbol{\nu}}t_{\boldsymbol{\nu}})R_\sigma(f_{\boldsymbol{\nu}})(y_1,\dots,y_n)\\
&\quad-\sum_{\boldsymbol{\nu}\in\Lambda_n}R_\sigma(\tilde{\times})\big(R_\sigma(\tilde{I}_{m_n;\boldsymbol{\nu}}t_{\boldsymbol{\nu}}),R_\sigma(f_{\boldsymbol{\nu}})(y_1,\dots,y_n)\big)\Bigg\|_V\\
&\le Cn^{-1/p_V+1}+n^{-1/p_V+1}+Cn^{-1/p_V+1}+Cn^{-1/p_V+1}\\
&\overset{(5.4)}{=}\mathcal{O}(\mathcal{N}_n^{-r}).
\end{aligned}\tag{5.10}
$$

The first of four terms has been estimated using Equations (4.17) and (5.4), the second using Equation (5.6), the third using and Equations (2.43) and (5.5) and the fourth using Equation (5.9). The last step follows from Equation (5.4). This finishes Step 3 of this proof.

Step 4 is given in Appendix B.7 and gives bounds on the depth, the size and the number of non-zero coefficients of the network.

*Step 5.* In total, the network has depth of the order $\mathcal{O}(\log(\mathcal{N}_n)\log\log(\mathcal{N}_n))$ and size and number of non-zero coefficients of the order $\mathcal{O}(\mathcal{N}_n\log(\mathcal{N}_n)\log\log(\mathcal{N}_n))$. Denoting the network size by $\mathcal{N}_n^*$ and comparing $\mathcal{N}_n^*$ with Equation (5.10), we find that the proposed network $\tilde{u}_n$ achieves

$$\sup_{\boldsymbol{y}\in U}\|u(\boldsymbol{y})-R_\sigma(\tilde{u}_n)(y_1,\dots,y_n)\|_V=\mathcal{O}\left((\mathcal{N}_n^*)^{-r^*}\right)$$

for any $r^*<r$. $\square$

**Remark 5.4.** *The last of four terms in Equation (5.10) is of the order $\mathcal{O}(n\delta_n)$. Taking $\hat{\delta}_n:=\mathcal{N}_n^{-r-1}$ instead of $\delta_n=n^{-1/p_V}$, as in [75], the resulting network is slightly smaller, but we cannot improve the current bound, namely that the size is of the order $\mathcal{O}(\mathcal{N}_n+n\log(n)\log\log(n))$. For $\hat{\delta}_n$, the error bound in Equation (5.10) is still of the order $\mathcal{O}(\mathcal{N}_n^{-r})$, hence taking $\hat{\delta}_n$ instead of $\delta_n$ does not affect the convergence rate in terms of the network size.*

*For $\hat{\delta}_n$ instead of $\delta_n$, the depth and the number of non-zero coefficients are also of the same order as before.*

# 6 Discussion and directions for further research

We first discuss Theorem 5.1 in Section 6.1. We then discuss possible generalisations of the theorem. Section 6.2 contains two alternatives for the ReLU approximation of gpc coefficients discussed in Section 3. The first alternative is a more efficient ReLU approximation of the full grid interpolants introduced in Section 3.1, the second alternative involves the ReLU approximation of functions with corner singularities. Section 6.3 discusses the generalisation of the spatial domain to hypercubes and general two-dimensional polygons. Section 6.4 discusses alternatives for the parametric PDE theory in Section 4, especially an alternative that allows PDE data with corner singularities.

## 6.1 Discussion of Theorem 5.1

Theorem 5.1 shows a lower bound on the convergence rate of the family of ReLU networks $\{\tilde{u}_n\}_{n\in\mathbb{N}}$ defined in Equation (5.8) in terms of the network size. The lower bound has been discussed in Remark 5.2. As the number of non-zero coefficients is of the same order as the network size, the same bound holds for the convergence rate in terms of the number of non-zero coefficients. The bound should be seen as a benchmark for the performance of training algorithms, i.e. it describes how the upper bound on the error decreases with an increase in network size. However, it does not bound the amount of computational work needed to reach a certain accuracy through training or how this amount of work depends on the required accuracy.

We note that in [75] and in this thesis the $H^1$-error is estimated, whereas most other ReLU DNN approximation results are w.r.t the $L^\infty$-norm (e.g. the results in [89, 48, 54, 24]). Denoting $D = (0,1)^2$, we expect that our results also hold w.r.t. the $L^\infty(D)$-norm instead of the $H^1(D)$-norm. Instead of Proposition 4.4 for $H^1_0(D) =: V$, we can use Proposition 4.6 for $H^2 \cap H^1_0(D) =: X$ and use boundedness of the embedding $X \hookrightarrow L^\infty(D)$. Then, the summability exponent $p_{L^\infty} \in (0,1)$ replacing $p_V$ equals $p_X$, which means that, if the statement analogous to Theorem 5.1 holds, the lower bound on the convergence rate w.r.t. the $L^\infty(D)$-norm equals $\min\{\gamma, 1/p_X - 1\}$ (see Remark 5.2).

In addition, we note that the proof of the convergence rate bound is mostly constructive: the architecture is constructed explicitly, assuming that the Taylor gpc coefficients $\{t_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\mathcal{F}}$ are known. For DNN training, it is not very important that $\{t_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\mathcal{F}}$ are in general unknown, as the coefficients resulting from training are not expected to approximate those of $\tilde{u}_n$ constructed in this thesis.

Even if $\{t_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\mathcal{F}}$ are unknown, it would be interesting to see whether an architecture similar to that of the networks $\{\tilde{u}_n\}_{n\in\mathbb{N}}$ performs well. In addition, it remains to be studied whether the coefficients of $\{\tilde{u}_n\}_{n\in\mathbb{N}}$ that are independent of $\{t_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\mathcal{F}}$ can used to improve the results of training, e.g. by using some of them as initial values for the training process.

## 6.2 Alternatives for the ReLU DNN approximation of Taylor gpc coefficients

### 6.2.1 More efficient ReLU DNN approximation of the full grid continuous, piecewise bilinear interpolants introduced in Section 3.1

We note that the full grid approximation of functions in $H^2 \cap H^1_0(D)$ for $D = (0,1)^2$ used in Section 3.1 has a sum-product structure similar to that of the networks studied in [19]. Inspired by the hierarchical network structure used there, we could approximate the full grid interpolants given by Proposition 3.1 more hierarchically, namely by a network $\tilde{w}^{\text{hier}}$ satisfying

$$R_\sigma(\tilde{w}^{\text{hier}})(\boldsymbol{x}) = \sum_{k_1=1}^{2^n-1} R_\sigma(\tilde{\times}) \left( \varphi^1_{k_1}(x_1), \sum_{k_2=1}^{2^n-1} w(\boldsymbol{x}^{1,2}_{\boldsymbol{k}}) \varphi^2_{k_2}(x_2) \right), \quad \boldsymbol{x} \in D.$$

We can define the matrix $\text{pr}_{\text{hier}} \in \mathbb{R}^{2(2^n-1)\times 2(2^n-1)}$ and the network $\tilde{w}^{\text{hier}}$ as

$$(\text{pr}_{\text{hier}})_{i,j} := \begin{cases} 1 & 1 \le j \le 2^n - 1, i = 2j-1, \\ w\big(\boldsymbol{x}^{1,2}_{(i/2, j-(2^n-1))}\big) & 2^n \le j \le 2(2^n-1), i \in 2\mathbb{Z}, \\ 0 & \text{else}, \end{cases}$$

$$\tilde{w}^{\text{hier}} := \Big( \begin{pmatrix} 1 & \cdots & 1 \end{pmatrix}, \boldsymbol{0} \Big) \bullet \text{Parallel}(\tilde{\times}, \dots, \tilde{\times}) \bullet (\text{pr}_{\text{hier}}, \boldsymbol{0}) \bullet P,$$

where $P$ is as defined in Equation (3.14) and where $\text{Parallel}(\tilde{\times}, \dots, \tilde{\times})$ is the parallelisation of $2^n - 1 = \mathcal{O}(2^n)$ $\tilde{\times}$-subnetworks, instead of $(2^n-1)^2 = \mathcal{O}(2^{2n})$ subnetworks in Section 3.2. If we take the maximum input value $M \ge 1$ and the accuracy $0 < \delta < M \le M^2$ equal for all those $\tilde{\times}$-subnetworks, then it can be shown that $\text{depth}(\tilde{w}^{\text{hier}}) = \mathcal{O}(\log(M/\delta))$, $\text{size}(\tilde{w}^{\text{hier}}) = \mathcal{O}(2^n \log(M/\delta))$ and $\mathcal{M}(\tilde{w}^{\text{hier}}) = \mathcal{O}(2^{2n} + 2^n \log(M/\delta))$.

Note that the network size is much smaller than before. In fact, it is of smaller order than the the number of degrees of freedom of the full grid approximation it approximates. That number equals $(2^n-1)^2$, because the full grid approximation is determined by the function values $\{w(\boldsymbol{x}^{1,2}_{\boldsymbol{k}})\}_{\boldsymbol{k}\in\mathbb{N}^2_{\le 2^n-1}}$. The network size can be smaller than the number of degrees of freedom, because there can be many non-zero coefficients associated to a single ReLU. For that reason, in general, the convergence rate of a family of ReLU approximations in terms of the network size can be better than the convergence rate of traditional numerical methods in terms of the number of degrees of freedom, even if the ReLU approximations simply implement such a traditional numerical method. However, it is unknown whether training algorithms can profit from this advantage. Moreover, the advantage cannot be arbitrarily large, as the number of coefficients of a network $\Phi$ satisfying Assumption 2.1 (including vanishing coefficients) equals

$$\sum_{l=1}^{\text{depth}(\Phi)} (N_{l-1}+1) N_l \le \text{depth}(\Phi) \big( N_0 + \text{size}(\Phi) + N_{\text{depth}(\Phi)} \big)^2.$$

Because the function values $\{w(\boldsymbol{x}^{1,2}_{\boldsymbol{k}})\}_{\boldsymbol{k}\in\mathbb{N}^2_{\le 2^n-1}}$ appear as coefficients, the number of non-zero coefficients of $\tilde{w}^{\text{hier}}$ is bounded from below by $(2^n-1)^2$, hence the number of non-zero coefficients is nearly of the same order as in Lemma 3.2. This shows that in this situation the number of non-zero coefficients is a better measure for the number of degrees of freedom than the number of ReLUs.

The $H^1$-error of $\tilde{w}^{\text{hier}}$ is expected to be of the same order as that of $\tilde{w}$, except for an increase in error if $M > 1$, which does not affect the asymptotic behaviour for $\delta \downarrow 0$. This implies that the convergence rate with respect to the $H^1$-norm in terms of the network size is at least $\gamma$ for any $\gamma < 1$, which is considerably better than for $\tilde{w}$. It also follows that the lower bound on the convergence rate in terms of the number of non-zero coefficients is $\gamma/2$ for any $\gamma < 1$.

Moreover, it is expected that the increased convergence rate of the ReLU approximations of functions in $H^2 \cap H_0^1((0,1)^2)$ directly translates into an increased convergence rate in Theorem 5.1, i.e. that Equations (5.2) and (5.3) hold for arbitrary $\gamma < 1$. In terms of the number of non-zero coefficients, it is expected that those equations hold with $\gamma$ replaced by $\gamma/2$ for any $\gamma < 1$.

### 6.2.2 Anisotropic sparse grid-based ReLU DNN approximation

For $D = (0,1)^2$, one of the drawbacks of using full grid approximations proposed in Section 3.1 for the approximation of Taylor gpc coefficients is that it requires Taylor gpc coefficients to be in $H^2(D)$ to get the convergence rates of Proposition 3.1.

More generally, the approximation of functions that may have corner singularities, which functions in $H^2(D)$ do not have, is studied in [56]. It studies the approximation of such functions by tensor products of univariate continuous, piecewise linear wavelets. Such wavelets can be implemented efficiently by ReLU networks using a construction analogous to that in Sections 3.2–3.3.

We now shortly discuss how we expect that special cases of results shown in [56] can be used for the generalisation of the results in Section 3. The results from [56] mentioned here hold in much more generality, see [56, especially Section 1 pp. 63–65].

For $s \in \mathbb{N}_0$, $\zeta \in \mathbb{R}$ and $r_D$ a smooth, positive, real-valued function on $D$ that close to each vertex of $D$ equals the distance to that vertex, the *Kondratiev space* $\mathcal{K}^s_\zeta(D)$ is defined by

$$\|u\|^2_{\mathcal{K}^s_\zeta(D)} := \sum_{|\boldsymbol{\alpha}|_1 \leq s} \|r_D^{|\boldsymbol{\alpha}|_1 - \zeta} \partial^{\boldsymbol{\alpha}} u\|^2_{L^2(D)},$$
$$\mathcal{K}^s_\zeta(D) := \{u : D \to \mathbb{C} \text{ measurable and } \|u\|_{\mathcal{K}^s_\zeta(D)} < \infty\}$$

([13, Equation (2) p. 2] and [90, Equation (4.4) p. 28]). We expect that for $1 < \zeta$ small enough it can be shown that functions in $\mathcal{K}^4_\zeta(D)$ can be approximated by an anisotropic sparse grid of tensor products of univariate continuous, piecewise linear wavelets, namely using [56, Theorem 1 p. 74], [56, Equation (13) p. 71] and [90, Equation (4.20) p. 33] for $p = 1$ and $\frac{3}{4} < \gamma < 1$. Denoting by $N_w$ the number of such bivariate continuous, piecewise bilinear wavelets, we expect that the $H^1$-error of the approximation can be shown to be of the order $\mathcal{O}(N_w^{-1}(\log N_w)^2)$. We expect that these continuous, piecewise bilinear wavelets can be implemented efficiently by ReLU networks, i.e. that the $H^1$-error of the resulting ReLU approximations of functions in $\mathcal{K}^4_\zeta(D)$ is of the order $\mathcal{O}(N^{-1}(\log N)^k)$ for some

$k \in \mathbb{R}_{>0}$ and where $N$ denotes the network size. Note that this is better than the rate $\mathcal{O}(N^{-1/2}(\log N)^{1/2})$ achieved by the full grid approximation in Proposition 3.5.

In Example 6.1 below, we will use the ReLU approximation of functions in $\mathcal{K}^4_\zeta(D)$ discussed here to propose a generalisation of Theorem 5.1 that allows the data of the parametric PDE to have corner singularities.

## 6.3 Generalisations of the choice of domain

### 6.3.1 Generalisation of Theorem 5.1 to $(0,1)^d$ for $d > 2$

We can consider the extension of Theorem 5.1 to a domain $(0,1)^d$ of dimension $d > 2$. The results of Section 3.1 directly generalise to piecewise multilinear interpolation on such domains for any $d \in \mathbb{N}$, but the required number of $d$-variate continuous, piecewise $d$-linear hat functions grows exponentially in the dimension $d$: it is of the order $\mathcal{O}((2^n)^d)$ when the domain is partitioned according to a tensor product of partitions, one for each spatial coordinate, each consisting of $2^n$ elements. This is called *the curse of dimensionality.*

Another issue is that for $d \geq 4$ the space $H^2((0,1)^d)$ does not embed into $L^\infty((0,1)^d)$, hence some of the currently used error bounds do not generalise to $d \geq 4$. In addition, for $d \geq 3$, we need error estimates on the derivatives of ReLU approximations of products of $d$ numbers. They are needed to generalise the results of Section 3.2. All in all, the extension to $(0,1)^3$ is expected to hold. For extensions to $(0,1)^d$ more issues arise for $d > 3$. Independent of that, the curse of dimensionality means that such extensions are not of practical importance for $d$ much larger than 3.

### 6.3.2 Generalisation of Theorem 5.1 to general polygonal domains

Another way to generalise the domain is to consider the elliptic diffusion equation on a general two-dimensional polygon $D$. We discuss two approaches.

We first assume that a family of gpc approximations of the solution map is given and that we have shown a lower bound on its convergence rate. For parameters $\boldsymbol{y} \in [-1,1]^{\mathbb{N}} = U$ we denote the solution map by $u : U \to H^1_0(D) : \boldsymbol{y} \mapsto u(\boldsymbol{y})$. We note that $D$ can be partitioned into a finite number of triangles ([26, Section 7.3.3 pp. 349–354]) and that each triangle can be divided into three quadrilaterals, e.g. along the line segments that connect the midpoints of its edges with its barycentre. For each such quadrilateral $K$, there exists a bilinear (hence smooth) transformation $F_K$ that maps $\hat{K} := (0,1)^2$ to $K$. Using $F_K$, the spatially restricted solution map $u|_K(\boldsymbol{y})$ can be pulled back to $\hat{K}$. We expect that the obtained results on ReLU approximations can be applied to the map $u|_K(\boldsymbol{y}) \circ F_K$ on $\hat{K}$, resulting in ReLU approximations of that map that each take $\boldsymbol{x} \in \hat{K}$ and finitely many of the parameters $\boldsymbol{y} \in U$ as input. We can now pull back by a ReLU approximation of the map $F_K^{-1}$, which gives ReLU approximations of the solution map $u|_K(\boldsymbol{y})$ on $K$. In this procedure, we exploit the fact that the rational map $F_K^{-1}$ can efficiently be approximated by ReLU networks ([78, Theorem 1.1 p. 1]) and that the composition of realisations of ReLU networks can simply be implemented by the concatenation of the networks.

This can be done for each such quadrilateral $K$. It then needs to be shown that the resulting approximations of $u$ are continuous across the edges of the quadrilaterals and that they can each be implemented globally by one ReLU network. In addition, for each $K$, the error bounds on the ReLU approximations of $u|_K(\boldsymbol{y}) \circ F_K$ on $\hat{K}$ have to be transferred to error bounds on the ReLU approximations of $u|_K(\boldsymbol{y})$ on $K$. Finally, the error bounds achieved for all quadrilaterals $K$ have to be combined into a global error estimate for the approximations of $u(\boldsymbol{y})$ on all of $D$.

Another approach is to show that $u|_K(\boldsymbol{y}) \circ F_K$ satisfies a parametric elliptic PDE on $\hat{K} = (0,1)^2$. If we can generalise the results of Section 4 to that PDE on $\hat{K}$ and if we can generalise Theorem 5.1 to that setting as well, then for each quadrilateral $K$ we can pull back the resulting ReLU approximations of the solution map to $K$ by a ReLU approximation of $F_K^{-1}$ and proceed as in the other approach, i.e. by combining the ReLU approximations on quadrilaterals into networks that approximate $u$ globally and by combining the error estimates on the quadrilaterals into a global error estimate.

The advantage of the second approach is that it suffices to show the sparsity of gpc expansions on $(0,1)^2$, whereas in the first approach we need to show it on $D$. On the other hand, the parametric PDE that the pull-back of $u|_K(\boldsymbol{y}) \circ F_K$ satisfies could be of a different type than that on $D$: the pull-back of a diffusion equation with a scalar diffusion coefficient need not be a diffusion equation with a scalar diffusion coefficient.

## 6.4 Generalisation of the parametric PDE theory in Section 4

The parametric PDE theory discussed in Section 4 can be generalised in several ways. We could consider more general elliptic PDEs in the current $L^2$-based Hilbert space setting, but could also, even more generally, use $L^q$-based theory for $1 < q < 2$. For any generalisation, it has to be shown that the equation is well-posed uniformly w.r.t. the parameters and that the solution has a sparse gpc expansion.

For well-posedness, we need to generalise Lemma 4.1. It is a special case of the Lax-Milgram lemma, which only gives sufficient conditions for well-posedness of the PDE and which is restricted to Hilbert spaces. Instead, we could use the more general Banach-Nečas-Babuška theorem (*BNB theorem*, [26, Theorem 2.6 p. 85]), which gives conditions that are both necessary and sufficient for well-posedness. Moreover, it is not restricted to Hilbert spaces. It holds in the more general context of reflexive Banach spaces, which includes $L^q$-based settings for $1 < q < 2$.

It then remains a point of further investigation whether Propositions 4.4 and 4.6 can be extended, based on the isomorphism property that holds under the assumptions of the BNB-theorem, i.e. that the differential operator in the PDE is boundedly invertible. The reason why the proofs given in Section 4 do not generalise directly is that in Equations (4.27) and (4.35) $t_{\boldsymbol{\nu}}$ was used as a test function, which is not possible in the $L^q$-based setting with $1 < q < 2$.

Examples of more general parametric PDEs that have a sparse gpc expansion were given in [14]. Sparsity was shown for solution maps of a class of linear elliptic PDEs, a class of linear parabolic PDEs, a nonlinear elliptic PDE and a PDE with parameter-dependent domain. These four cases were covered by showing that the solution maps are $(\boldsymbol{b}, \varepsilon)$-holomorphic (see e.g. [14, Definition 2.1 p. 407] and [75, Definition 2.1 p. 4]), which implies sparsity of the gpc expansions. We note that ReLU approximations of $(\boldsymbol{b}, \varepsilon)$-holomorphic maps are studied in [75, especially Theorem 2.7 p. 6].

The choice for an affine parametrisation of the diffusion coefficient (Equation (4.11)) and a Taylor gpc expansion (Equation (4.15)) are not restrictive. Results similar to those discussed in Section 4 for different expansions have been shown in e.g. [3, Sections 5 and 6 pp. 2169–2176].

Based on [13], we now discuss a generalisation of the parametric PDE theory that allows PDE data with corner singularities of the type discussed in Section 6.2.2. We propose a generalisation of Theorem 5.1 that combines the ideas discussed in Section 6.2.2 for the ReLU approximation of gpc coefficients with the parametric PDE theory introduced in this section.

**Example 6.1** (Generalisation of the parametric PDE theory that allows PDE data with corner singularities)**.** *The generalisation of the parametric PDE theory is based on [13, Theorem 1.1 p. 3], which studies a class of non-parametric elliptic PDEs that includes Equation (4.1) with non-parametric diffusion coefficient, i.e.*

$$-\operatorname{div}(a\nabla u) = f, \qquad u|_{\partial D} = 0. \tag{6.1}$$

*The results in [13] hold for general curvilinear polygonal domains, for simplicity we here take $D = (0,1)^2$. In addition, the results in [13] hold for more general elliptic PDEs and more general boundary conditions.*

*For $s \in \mathbb{N}_0$ and for $r_D$ as in Section 6.2.2, let the space $\mathcal{W}^{s,\infty}(D)$ be defined by*

$$\|a\|_{\mathcal{W}^{s,\infty}(D)} := \sup_{|\boldsymbol{\alpha}|_1 \le s} \|r_D^{|\boldsymbol{\alpha}|_1} D^{\boldsymbol{\alpha}} u\|_{L^\infty(D)},$$

$$\mathcal{W}^{s,\infty}(D) := \{a : D \to \mathbb{C} \text{ measurable and } r_D^{|\boldsymbol{\alpha}|_1} D^{\boldsymbol{\alpha}} a \in L^\infty(D), |\boldsymbol{\alpha}|_1 \le s\}$$

*([13, Equation (5) p. 3] and [90, Equation (4.5) p. 28]). For $\zeta > 1$ small enough, by [13, Theorem 1.1 p. 3], $a \in \mathcal{W}^{3,\infty}(D)$ and $f \in \mathcal{K}^2_{\zeta-2}(D)$ imply that the solution $u$ of Equation (6.1) is contained in $\mathcal{K}^4_\zeta(D)$, where $\mathcal{K}^2_{\zeta-2}(D)$ and $\mathcal{K}^4_\zeta(D)$ are Kondratiev spaces defined in Section 6.2.2. In particular, the solution may also have corner singularities. We note that $\zeta$ and $\zeta - 2$ in our notation correspond to $a + 1$ and $a - 1$ in [13].*

*In [13, Theorem 1.1 p. 3], it is also shown that Equation (6.1) is well posed and that the solution $u$ depends analytically on the data $a$ and $f$. Stronger still, for uniformly elliptic parametric data, these results hold uniformly in the parameters.*

*For uniformly elliptic affinely parametrised data, the parameter-to-data map is analytic. We could use that to show analyticity of the solution map of Equation (4.1). In [18], analyticity of the solution map of Equation (4.1) with respect to the $V$-norm ([18, Section 2 pp. 20–26]) was used to show sparsity of the Taylor gpc expansion of the solution map ([18, Theorem 1.2 p. 17]). That theorem shows the $\ell^p(\mathcal{F})$-summability of the $V$-norms of the Taylor gpc coefficients for some $0 < p < 1$, which is also shown in Lemma 4.3 and Proposition 4.4. For $\forall \boldsymbol{\nu} \in \mathcal{F} : b_{\boldsymbol{\nu}} := \|t_{\boldsymbol{\nu}}\|_V$, Lemma 4.3 shows more than the result of [18, Theorem 1.2 p. 17]. We expect that [75, Theorem 2.7 p. 6] can be used to show the analogy of Lemma 4.3 for that case.*

*We expect that similar arguments can be used to show sparsity of the Taylor gpc expansion with respect to the $\mathcal{K}^4_\zeta(D)$-norm, which generalises Remark 4.8. That is, we have proposed generalisations for the main results in Section 4. We expect that they can be used to generalise Theorem 5.1, using the ReLU approximation of functions in $\mathcal{K}^4_\zeta(D)$ proposed in Section 6.2.2 for the approximation of the Taylor gpc coefficients.*

## A Properties of concatenations and parallelisations

**Remark A.1** (Cf. [62, Definition 2.2 p. 6]). *For $\Phi^1$ and $\Phi^2$ as in Definition 2.7, it follows from Equation (2.13) that*

$$\begin{aligned}\operatorname{depth}(\Phi^1 \bullet \Phi^2) &= \operatorname{depth}(\Phi^1) + \operatorname{depth}(\Phi^2) - 1, && \text{(A.1)}\\ \operatorname{size}(\Phi^1 \bullet \Phi^2) &= \operatorname{size}(\Phi^1) + \operatorname{size}(\Phi^2), && \text{(A.2)}\\ \mathcal{M}(\Phi^1 \bullet \Phi^2) &= \mathcal{M}(\Phi^1) + \mathcal{M}(\Phi^2) - \|A^2_{L^2}\|_{\ell^0} - \|A^1_1\|_{\ell^0} + \|A^1_1 A^2_{L^2}\|_{\ell^0}\\ &\quad - \|\boldsymbol{b}^2_{L^2}\|_{\ell^0} - \|\boldsymbol{b}^1_1\|_{\ell^0} + \|A^1_1 \boldsymbol{b}^2_{L^2} + \boldsymbol{b}^1_1\|_{\ell^0} && \text{(A.3)}\\ &\le \sum_{l=2}^{L^1} \|A^1_l\|_{\ell^0} + \sum_{l=1}^{L^2-1} \|A^2_l\|_{\ell^0} + \|A^1_1\|_{\ell^0}\|A^2_{L^2}\|_{\ell^0} + N^1_1. && \text{(A.4)}\end{aligned}$$

**Remark A.2** (Extensions, cf. [24, Setting 5.2 pp. 17–18]). *For arbitrary $L \in \mathbb{N}$, $N_0, N_1, \dots, N_L \in \mathbb{N}$ and*

$$\Phi = ((A_1, \boldsymbol{b}_1), \dots, (A_L, \boldsymbol{b}_L)) \in \mathcal{N}_L^{N_0, N_1, \dots, N_L},$$

*it follows from Lemma 2.8 and Remarks A.1 and 2.5 that for $L' \in \mathbb{N}$*

$$\begin{aligned}R_\sigma(\Phi^{\mathrm{Id}}_{N_L,L'} \bullet \Phi) &= R_\sigma(\Phi) = R_\sigma(\Phi \bullet \Phi^{\mathrm{Id}}_{N_0,L'}), && \text{(A.5)}\\ \operatorname{depth}(\Phi^{\mathrm{Id}}_{N_L,L'} \bullet \Phi) &= L + L' - 1 = \operatorname{depth}(\Phi \bullet \Phi^{\mathrm{Id}}_{N_0,L'}), && \text{(A.6)}\\ \operatorname{size}(\Phi^{\mathrm{Id}}_{N_L,L'} \bullet \Phi) &= \operatorname{size}(\Phi) + 2(L'-1)N_L, && \text{(A.7)}\\ \operatorname{size}(\Phi \bullet \Phi^{\mathrm{Id}}_{N_0,L'}) &= \operatorname{size}(\Phi) + 2(L'-1)N_0, && \text{(A.8)}\\ \mathcal{M}(\Phi^{\mathrm{Id}}_{N_L,L'} \bullet \Phi) &\overset{(A.3)}{=} 2L'N_L + \mathcal{M}(\Phi) - \|A_L\|_{\ell^0} - 2N_L + (1 + \mathbb{1}[L' > 1])\|A_L\|_{\ell^0}\\ &\quad - \|\boldsymbol{b}_L\|_{\ell^0} - 0 + (1 + \mathbb{1}[L' > 1])\|\boldsymbol{b}_L\|_{\ell^0}\\ &= \mathcal{M}(\Phi) + 2(L'-1)N_L + \mathbb{1}[L' > 1](\|A_L\|_{\ell^0} + \|\boldsymbol{b}_L\|_{\ell^0}), && \text{(A.9)}\\ \mathcal{M}(\Phi \bullet \Phi^{\mathrm{Id}}_{N_0,L'}) &\overset{(A.3)}{=} \mathcal{M}(\Phi) + 2L'N_0 - 2N_0 - \|A_1\|_{\ell^0} + (1 + \mathbb{1}[L' > 1])\|A_1\|_{\ell^0}\\ &\quad - 0 - \|\boldsymbol{b}_1\|_{\ell^0} + \|\boldsymbol{b}_1\|_{\ell^0}\\ &= \mathcal{M}(\Phi) + 2(L'-1)N_0 + \mathbb{1}[L' > 1]\|A_1\|_{\ell^0}. && \text{(A.10)}\end{aligned}$$

*In particular, we note that $\Phi^{\mathrm{Id}}_{N_L,1} \bullet \Phi = \Phi = \Phi \bullet \Phi^{\mathrm{Id}}_{N_0,1}$.*

**Remark A.3** (Cf. [24, Lemma 5.3 p. 18] and [62, Remark 2.6 p. 7]). *For $\Phi^1$ and $\Phi^2$ as in Definition 2.9, it follows from Equation (2.14) that*

$$\begin{aligned}\Phi^1 \odot \Phi^2 = \Bigg( &(A^2_1, \boldsymbol{b}^2_1), \dots, (A^2_{L^2-1}, \boldsymbol{b}^2_{L^2-1}), \left( \begin{pmatrix} A^2_{L^2} \\ -A^2_{L^2} \end{pmatrix}, \begin{pmatrix} \boldsymbol{b}^2_{L^2} \\ -\boldsymbol{b}^2_{L^2} \end{pmatrix} \right),\\ &\left( \begin{pmatrix} A^1_1 & -A^1_1 \end{pmatrix}, \boldsymbol{b}^1_1 \right), (A^1_2, \boldsymbol{b}^1_2), \dots, (A^1_{L^1}, \boldsymbol{b}^1_{L^1}) \Bigg), && \text{(A.11)}\end{aligned}$$

*from which it follows that*

$$\begin{aligned}\operatorname{depth}(\Phi^1 \odot \Phi^2) &= \operatorname{depth}(\Phi^1) + \operatorname{depth}(\Phi^2), && \text{(A.12)}\\ \operatorname{size}(\Phi^1 \odot \Phi^2) &= \operatorname{size}(\Phi^1) + 2N^2_{L^2} + \operatorname{size}(\Phi^2), && \text{(A.13)}\\ \mathcal{M}(\Phi^1 \odot \Phi^2) &= \mathcal{M}(\Phi^1) + \mathcal{M}(\Phi^2) + \|A^2_{L^2}\|_{\ell^0} + \|\boldsymbol{b}^2_{L^2}\|_{\ell^0} + \|A^1_1\|_{\ell^0}. && \text{(A.14)}\end{aligned}$$

*It follows from Lemma 2.8 and Remark 2.5 that* $R_\sigma(\Phi^1 \odot \Phi^2) = R_\sigma(\Phi^1) \circ R_\sigma(\Phi^2)$*.*

**Remark A.4** (Cf. [62, p. 7 below Definition 2.7, including Remark 2.8] and [24, Lemma 5.4 p. 19])**.** *We consider the situation of Definition 2.10. By Equation (A.5), we find*

$$\begin{aligned} R_\sigma(\text{Shared-Parallel}(\Phi^1, \dots, \Phi^N)) : \mathbb{R}^{N_0} \to \mathbb{R}^{\tilde{N}^{\text{tot}}_{L^{\max}}} :& \\ \boldsymbol{x} \mapsto ((R_\sigma(\Phi^1)(\boldsymbol{x}))^\top, \dots, (R_\sigma(\Phi^N)(\boldsymbol{x}))^\top)^\top.& \end{aligned} \tag{A.15}$$

*In addition, using Remark A.2, we find*

$$\text{depth}(\text{Shared-Parallel}(\Phi^1, \dots, \Phi^N)) = L^{\max} = \max_{i \in \{1,\dots,N\}} L^i, \tag{A.16}$$

$$\forall i \in \{1, \dots, N\} : \tilde{N}^i_l = \begin{cases} 2N_0 & l \in \{1, \dots, L^{\max} - L^i\}, \\ N^i_{-(L^{\max}-L^i)+l} & l \in \{L^{\max} - L^i + 1, \dots, L^{\max}\}, \end{cases} \tag{A.17}$$

$$\begin{aligned} \text{size}(\text{Shared-Parallel}(\Phi^1, \dots, \Phi^N)) &\overset{(A.8)}{=} \sum_{i=1}^N \big( \text{size}(\Phi^i) + 2N_0(L^{\max} - L^i) \big) \\ &= \sum_{i=1}^N \text{size}(\Phi^i) + 2N_0 \left( NL^{\max} - \sum_{i=1}^N L^i \right), \end{aligned} \tag{A.18}$$

$$\begin{aligned} \mathcal{M}(\text{Shared-Parallel}(\Phi^1, \dots, \Phi^N)) &\overset{(A.10)}{=} \sum_{i=1}^N \big( \mathcal{M}(\Phi^i) + 2N_0(L^{\max} - L^i) \\ &\quad + \mathbb{1}[L^{\max} > L^i] \|A^i_1\|_{\ell^0} \big) \\ &= \sum_{i=1}^N \mathcal{M}(\Phi^i) + 2N_0 \left( NL^{\max} - \sum_{i=1}^N L^i \right) \\ &\quad + \sum_{i=1}^N \mathbb{1}[L^{\max} > L^i] \|A^i_1\|_{\ell^0}. \end{aligned} \tag{A.19}$$

**Remark A.5.** *Note that for all* $l \in \{1, \dots, L^{\max}\}$ *the weight matrices* $\{\tilde{A}^i_l\}_{i: L^{\max}-L^i > l-1}$ *coincide and equal* $\text{Id}_{R^{2N_0}}$ *and that the bias vectors* $\{\tilde{\boldsymbol{b}}^i_l\}_{i: L^{\max}-L^i > l-1}$ *vanish. This fact can be used to construct the* parallelisation for shared inputs with simultaneous implementation of the identity operator Simul-Shared-Parallel$(\Phi^1, \dots, \Phi^N)$*, which satisfies Equation (A.15), but has smaller size and number of non-zero coefficients than* Shared-Parallel$(\Phi^1, \dots, \Phi^N)$*.*

*Without loss of generality, we may assume that* $L^1 = \min_{i \in \{1,\dots,N\}} L^i$*, which allows us to* *define*

$$
\begin{aligned}
&\text{Pre-Simul-Shared-Parallel}(\Phi^1, \ldots, \Phi^N) := \\
&\left( \left( \begin{pmatrix} \tilde{A}_1^1 \\ \mathbb{1}[L^{\max} - L^2 = 0]\tilde{A}_1^2 \\ \vdots \\ \mathbb{1}[L^{\max} - L^N = 0]\tilde{A}_1^N \end{pmatrix}, \begin{pmatrix} \tilde{\boldsymbol{b}}_1^1 \\ \mathbb{1}[L^{\max} - L^2 = 0]\tilde{\boldsymbol{b}}_1^2 \\ \vdots \\ \mathbb{1}[L^{\max} - L^N = 0]\tilde{\boldsymbol{b}}_1^N \end{pmatrix} \right), \right. \\
&\left( \begin{pmatrix} \tilde{A}_2^1 & & & \\ \mathbb{1}[L^{\max} - L^2 = 1]\tilde{A}_2^2 & \mathbb{1}[L^{\max} - L^2 < 1]\tilde{A}_2^2 & & \\ \vdots & & \ddots & \\ \mathbb{1}[L^{\max} - L^N = 1]\tilde{A}_2^N & & & \mathbb{1}[L^{\max} - L^N < 1]\tilde{A}_2^N \end{pmatrix}, \right. \\
&\left. \begin{pmatrix} \tilde{\boldsymbol{b}}_2^1 \\ \mathbb{1}[L^{\max} - L^2 \leq 1]\tilde{\boldsymbol{b}}_2^2 \\ \vdots \\ \mathbb{1}[L^{\max} - L^N \leq 1]\tilde{\boldsymbol{b}}_2^N \end{pmatrix} \right), \\
&\left( \begin{pmatrix} \tilde{A}_3^1 & & & \\ \mathbb{1}[L^{\max} - L^2 = 2]\tilde{A}_3^2 & \mathbb{1}[L^{\max} - L^2 < 2]\tilde{A}_3^2 & & \\ \vdots & & \ddots & \\ \mathbb{1}[L^{\max} - L^N = 2]\tilde{A}_3^N & & & \mathbb{1}[L^{\max} - L^N < 2]\tilde{A}_3^N \end{pmatrix}, \right. \\
&\left. \left. \begin{pmatrix} \tilde{\boldsymbol{b}}_3^1 \\ \mathbb{1}[L^{\max} - L^2 \leq 2]\tilde{\boldsymbol{b}}_3^2 \\ \vdots \\ \mathbb{1}[L^{\max} - L^N \leq 2]\tilde{\boldsymbol{b}}_3^N \end{pmatrix} \right), \ldots, \left( \begin{pmatrix} \tilde{A}_{L^{\max}}^1 & & & \\ & \tilde{A}_{L^{\max}}^2 & & \\ & & \ddots & \\ & & & \tilde{A}_{L^{\max}}^N \end{pmatrix}, \begin{pmatrix} \tilde{\boldsymbol{b}}_{L^{\max}}^1 \\ \tilde{\boldsymbol{b}}_{L^{\max}}^2 \\ \vdots \\ \tilde{\boldsymbol{b}}_{L^{\max}}^N \end{pmatrix} \right) \right),
\end{aligned}
\tag{A.20}
$$

*which satisfies Equations (A.15)–(A.16) and*

$$
\begin{aligned}
\mathcal{M}(\text{Pre-Simul-Shared-Parallel}(\Phi^1, \ldots, \Phi^N)) = &\sum_{i=1}^N \mathcal{M}(\Phi^i) + \sum_{i=1}^N \mathbb{1}[L^{\max} > L^i] \|A_1^i\|_{\ell^0} \\
&+ 2N_0 \left( L^{\max} - \min_{i \in \{1,\ldots,N\}} L^i \right).
\end{aligned}
\tag{A.21}
$$

*Let* $\{\mathbb{A}_l\}_{l \in \{1,\ldots,L^{\max}\}}$, $\{\mathbb{b}_l\}_{l \in \{1,\ldots,L^{\max}\}}$ *be such that* $\text{Pre-Simul-Shared-Parallel}(\Phi^1, \ldots, \Phi^N) = ((\mathbb{A}_1, \mathbb{b}_1), \ldots, (\mathbb{A}_{L^{\max}}, \mathbb{b}_{L^{\max}}))$. *For* $i \in \{2, \ldots, N\}$ *and* $l \in \{1, \ldots, L^{\max}\}$ *that satisfy* $L^{\max} - L^i > l - 1$, *the* $i$*'th "row" of* $\mathbb{A}_l$, *the* $i$*'th "row" of* $\mathbb{b}_l$ *and the* $i+1$*'st "column" of* $\mathbb{A}_{l+1}$ *vanish identically. For such* $i$ *and* $l$, *changing the sizes of* $\mathbb{A}_l$, $\mathbb{b}_l$ *and* $\mathbb{A}_{l+1}$ *by removing "row"* $i$ *from* $\mathbb{A}_l$ *and* $\mathbb{b}_l$ *and "column"* $i+1$ *from* $\mathbb{A}_{l+1}$ *gives a network* $\text{Simul-Shared-Parallel}(\Phi^1, \ldots, \Phi^N)$ *that satisfies Equations (A.15), (A.16), (A.21) and*

$$
\text{size}(\text{Simul-Shared-Parallel}(\Phi^1, \ldots, \Phi^N)) = \sum_{i=1}^N \text{size}(\Phi^i) + 2N_0 \left( L^{\max} - \min_{i \in \{1,\ldots,N\}} L^i \right).
\tag{A.22}
$$

**Remark A.6** (Cf. [24, Lemma 5.4 p. 19] and [62, p. 7 below Definition 2.7, including Remark 2.8])**.** *Analogous to Remark A.4, assuming the situation of Definition 2.11, it holds that*

$$\begin{aligned} R_\sigma(\mathrm{Parallel}(\Phi^1, \dots, \Phi^N)) &: \mathbb{R}^{\tilde{N}_0^{\mathrm{tot}}} \to \mathbb{R}^{\tilde{N}_{L^{\max}}^{\mathrm{tot}}} : \\ \boldsymbol{x} \mapsto ((R_\sigma(\Phi^1)((x_1, \dots, x_{\tilde{N}_0^1})))^\top, &\dots, (R_\sigma(\Phi^N)((x_{\tilde{N}_0^{\mathrm{tot}} - \tilde{N}_0^N + 1}, \dots, x_{\tilde{N}_0^{\mathrm{tot}}})))^\top)^\top. \end{aligned} \tag{A.23}$$

*In addition, using Remark A.2, we find*

$$\mathrm{depth}(\mathrm{Parallel}(\Phi^1, \dots, \Phi^N)) = L^{\max} = \max_{i \in \{1, \dots, N\}} L^i, \tag{A.24}$$

$$\forall i \in \{1, \dots, N\} : \tilde{N}_l^i = \begin{cases} 2N_0^i & l \in \{1, \dots, L^{\max} - L^i\}, \\ N^i_{-(L^{\max} - L^i) + l} & l \in \{L^{\max} - L^i + 1, \dots, L^{\max}\}, \end{cases} \tag{A.25}$$

$$\mathrm{size}(\mathrm{Parallel}(\Phi^1, \dots, \Phi^N)) \overset{(A.8)}{=} \sum_{i=1}^N \big( \mathrm{size}(\Phi^i) + 2N_0^i(L^{\max} - L^i) \big), \tag{A.26}$$

$$\begin{aligned} \mathcal{M}(\mathrm{Parallel}(\Phi^1, \dots, \Phi^N)) \overset{(A.10)}{=} \sum_{i=1}^N \big( &\mathcal{M}(\Phi^i) + 2N_0^i(L^{\max} - L^i) \\ &+ \mathbb{1}[L^{\max} > L^i] \|A_1^i\|_{\ell^0} \big). \end{aligned} \tag{A.27}$$

# B Proofs

## B.1 Proof of Lemma 2.3

*Proof.* The idea of this proof is that of [60, Lemma 1 pp. 3–4]. The lemma follows from Equation (2.6) and the following two observations:

**First observation.** *Let $g : \mathbb{R}^{N_0} \to \mathbb{R}$ be a linear combination of $K \in \mathbb{N}$ continuous, piecewise linear functions $\{g_i\}_{i\in\{1,\dots,K\}} : \mathbb{R}^{N_0} \to \mathbb{R}$ that have the property of the lemma for families of sets $\{\mathcal{T}_i\}_{i\in\{1,\dots,K\}}$ satisfying $\forall i \in \{1,\dots,K\} : \mathcal{T}_i = \{T_{i;j_i}\}_{j_i\in\{1,\dots,J_i\}}$. Then, $g$ has the property of the lemma.*

*Proof of the first observation.* Let

$$\forall (j_1,\dots,j_K) \in \bigtimes_{i=1}^{K} \{1,\dots,J_i\} : T_{(j_1,\dots,j_K)} := \bigcap_{i=1}^{K} T_{i;j_i},$$

$$\mathcal{T} := \left\{T_{(j_1,\dots,j_K)}\right\}_{(j_1,\dots,j_K)\in\times_{i=1}^{K}\{1,\dots,J_i\}}$$

and observe that $\mathcal{T}$ is a finite family of subsets of $\mathbb{R}^{N_0}$. Let $J \in \mathbb{N}$ be such that there exists a bijection $B : \{1,\dots,J\} \to \times_{i=1}^{K}\{1,\dots,J_i\}$. For $j \in \{1,\dots,J\}$, let $T_j \in \mathcal{T}$ be defined by $T_j := T_{B(j)} \in \mathcal{T}$. $\mathcal{T}$ satisfies $\mathbb{R}^{N_0} = \bigcup_{j\in\{1,\dots,J\}} \overline{T_j}$ and all elements of $\mathcal{T}$ are finite intersections of open convex sets, hence open and convex. Note that, by construction of $\mathcal{T}$, the functions $\{g_i\}_{i\in\{1,\dots,K\}} : \mathbb{R}^{N_0} \to \mathbb{R}$ are linear on each element $T \in \mathcal{T}$, hence $g$ is linear on each element $T \in \mathcal{T}$. In addition, as a finite sum of continuous functions, $g$ is continuous. This finishes the proof of the first observation.

**Second observation.** *For a continuous, piecewise linear function $g : \mathbb{R}^{N_0} \to \mathbb{R}$ that has the property of the lemma for a family of sets $\mathcal{T}$, $\sigma \circ g$ also has the property of the lemma.*

*Proof of the second observation.* On each element $T$ of $\mathcal{T}$, the function $g$ is linear. If $g$ does not vanish identically on $T$, $T_+ := \{g > 0\} \cap T$ and $T_- := \{g < 0\} \cap T$ are open, convex subsets of $\mathbb{R}^{N_0}$ satisfying $T \subset \overline{T_+} \cup \overline{T_-}$. The function $\sigma \circ g$ is linear on $T_+$ and $T_-$. If $g$ vanishes identically on $T$, then $T_+ := T$ and $T_- := \emptyset$ are open, convex subsets of $\mathbb{R}^{N_0}$ satisfying $T \subset \overline{T_+} \cup \overline{T_-}$, such that $\sigma \circ g$ is linear on $T_+$ and $T_-$. We have $\mathbb{R}^{N_0} = \bigcup_{T\in\mathcal{T}}(\overline{T_+} \cup \overline{T_-})$. In addition, $\sigma \circ g$ is the composition of two continuous functions, hence continuous. This finishes the proof of the second observation.

The lemma follows from the two observations, Equation (2.6) and the fact that $\forall i \in \{1,\dots,N_0\} : \mathbb{R}^{N_0} \ni \boldsymbol{x} \mapsto x_i$ trivially has the property of the lemma for $\mathcal{T} = \{\mathbb{R}^{N_0}\}$. To show the lemma, we consider a network $\Phi$ that implements $f$. We use the same notation as in Notation 2.2. Assuming that for some $l \in \{1,\dots,L-1\}$ for all $k \in \{1,\dots,N_{l-1}\}$ the map $\mathbb{R}^{N_0} \ni \boldsymbol{x} \mapsto z_k^{l-1}$ has the property of the lemma, the first observation shows that for all $j \in \{1,\dots,N_l\}$ the function $\mathbb{R}^{N_0} \ni \boldsymbol{x} \mapsto \sum_{k=1}^{N_{l-1}} A_{l;j,k} z_k^{l-1} + b_{l;j}$ has the property as well. The second observation now shows that the same holds for $\mathbb{R}^{N_0} \ni \boldsymbol{x} \mapsto \sigma\left(\sum_{k=1}^{N_{l-1}} A_{l;j,k} z_k^{l-1} + b_{l;j}\right) = z_j^l$ for all $j \in \{1,\dots,N_l\}$. For $L = 1$, for all $k \in \{1,\dots,N_0\}$, the map $\boldsymbol{x} \mapsto x_k = z_k^0$ has the property of the lemma. By induction it follows that for all $k \in \{1,\dots,N_{L-1}\}$ the map $\mathbb{R}^{N_0} \ni \boldsymbol{x} \mapsto z_k^{L-1}$ also has the property. The first observation now shows that the same holds for $\mathbb{R}^{N_0} \ni \boldsymbol{x} \mapsto \sum_{k=1}^{N_{L-1}} A_{L;1,k} z_k^{L-1} + b_{L;1} = f(\boldsymbol{x})$, which finishes the proof of the lemma. $\square$

**Remark B.1** ([55, Proposition 4 p. 6], cf. [60, Lemma 1 pp. 3–4])**.** *Note that the statement of the lemma also holds for $N_0, N \in \mathbb{N}$ and functions $f : \mathbb{R}^{N_0} \to \mathbb{R}^N$ that can be implemented by ReLU networks as described in Assumption 2.1 and Notation 2.2. Each component of the output of $f$ can be implemented by a ReLU network of the desired type: Assume that $\Phi \in \mathcal{R}$ is such that $R_\sigma(\Phi) = f$. For all $i \in \{1, \dots, N\}$, we define $\Phi^i := ((\boldsymbol{e}_i)^\top, \mathbf{0}))$, where we interpret $\boldsymbol{e}_i$ as an $N \times 1$-matrix. Then, $\forall i \in \{1, \dots, N\} : R_\sigma(\Phi^i \bullet \Phi) : \mathbb{R}^{N_0} \to \mathbb{R}$ implements the i'th component of $f$ and satisfies the requirements of the lemma. The claim now follows from [60, Lemma 1 pp. 3–4], which is based on an argument similar to that used to prove the first observation in the proof of Lemma 2.3.*

## B.2 Proof of Proposition 2.12, Steps 3–5

*Step 3.* For arbitrary $m \in \mathbb{N}$, we construct a ReLU network $F_m$ satisfying $R_\sigma(F_m)\big|_{[0,1]} = f_m$ by implementing Equations (2.25)–(2.27) for $x \in [0,1]$, a construction that was first given in [89, Proposition 2 and its proof pp. 105–106]. The network is depicted in Figure B.1.

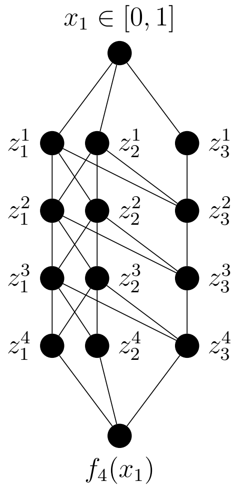


Figure B.1: Structure of $F_4$.

We define

$$A_l := \begin{pmatrix} 2 & -4 & 0 \\ 2 & -4 & 0 \\ -2^{-2(l-1)+1} & 2^{-2(l-1)+2} & 1 \end{pmatrix}, \quad l \in \{2,\dots,m\}, \qquad \boldsymbol{b} := \begin{pmatrix} 0 \\ -\frac{1}{2} \\ 0 \end{pmatrix}, \tag{B.1}$$

$$A_{m+1} := \begin{pmatrix} -2^{-2m+1} & 2^{-2m+2} & 1 \end{pmatrix}, \tag{B.2}$$

$$F_m := \left( \left( \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ -\frac{1}{2} \\ 0 \end{pmatrix} \right), (A_2, \boldsymbol{b}), (A_3, \boldsymbol{b}), \dots, (A_m, \boldsymbol{b}), (A_{m+1}, \boldsymbol{0}) \right). \tag{B.3}$$

For $l \in \{1,\dots,m\}$ and $x_1 \in [0,1]$, we define $\boldsymbol{z}^l := \big(R_{\sigma,l}(F_m) \circ \cdots \circ R_{\sigma,1}(F_m)\big)(x_1) \in \mathbb{R}^3$. It holds that

$$\boldsymbol{z}^1 = \big(\sigma(x_1), \sigma(x_1 - \tfrac{1}{2}), f_0(x_1)\big), \tag{B.4}$$

$$\boldsymbol{z}^l = \big(\sigma(g_{l-1}(x_1)), \sigma(g_{l-1}(x_1) - \tfrac{1}{2}), f_{l-1}(x_1)\big), \quad l \in \{2,\dots,m\}, \tag{B.5}$$

$$R_\sigma(F_m)(x_1) = -2^{-2m} g_m(x_1) + f_{m-1}(x_1) = f_m(x_1). \tag{B.6}$$

Note that $z_1^1 = z_3^1$.

*Step 4.* We bound the depth, the size and the number of non-zero coefficients of $\tilde{\times}$ in terms of $m = \lceil \log_2(2M) + \log(1/\delta) \rceil$ (Equation (2.34)). Before we do so, we calculate the second and the last weight matrix of $\tilde{\times}$. As a result of the concatenations and the parallelisation in Equation (2.29), they are defined as products of weight matrices of subnetworks. By Equations (2.29), (2.30), (2.18) and (B.3) we find that the second weight matrix of $\tilde{\times}$ equals

$$\frac{1}{2M} \begin{pmatrix} 1 & & \\ 1 & & \\ 1 & & \\ & 1 & \\ & 1 & \\ & 1 & \\ & & 1 \\ & & 1 \\ & & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 & & & & \\ & & 1 & 1 & & \\ & & & & 1 & 1 \end{pmatrix} = \frac{1}{2M} \begin{pmatrix} 1 & 1 & & & & \\ 1 & 1 & & & & \\ 1 & 1 & & & & \\ & & 1 & 1 & & \\ & & 1 & 1 & & \\ & & 1 & 1 & & \\ & & & & 1 & 1 \\ & & & & 1 & 1 \\ & & & & 1 & 1 \end{pmatrix}, \tag{B.7}$$

the second bias vector of $\tilde{\times}$ equals $(0, -\frac{1}{2}, 0, 0, -\frac{1}{2}, 0, 0, -\frac{1}{2}, 0)^\top$. By Equations (2.29), (2.18) and (B.2) we find that the last weight matrix equals

$$2M^2 \begin{pmatrix} -1 & -1 & 1 \end{pmatrix} \begin{pmatrix} A_{m+1} & & \\ & A_{m+1} & \\ & & A_{m+1} \end{pmatrix} = 2M^2 \begin{pmatrix} -A_{m+1} & -A_{m+1} & A_{m+1} \end{pmatrix}, \tag{B.8}$$

whereas the last bias vector vanishes.

In order to calculate depth($\tilde{\times}$), we note that depth($F_m$) $= m+1$. Hence, by Equations (2.29), (A.1), (2.30) and (A.24), we find that depth($\tilde{\times}$) $= m+2$.

From Equations (B.1)–(B.3), we see that for $N_0 = N_{m+1} = 1$ and $N_1 = \ldots = N_m = 3$ it holds that

$$F_m \in \mathcal{N}_{m+1}^{N_0,N_1,\ldots,N_{m+1}}.$$

Hence, by Equations (2.29), (2.30), (B.7), (A.25) and (B.8), we find

$$\tilde{\times} \in \mathcal{N}_{m+2}^{N'_0,N'_1,\ldots,N'_{m+2}}$$

for $N'_0 = 2$, $N'_1 = 6$, $N'_2 = \ldots = N'_{m+1} = 9$ and $N'_{m+2} = 1$. It follows that $\mathrm{size}(\tilde{\times}) = 9m+6$.

It follows from Equations (2.29), (2.30), (B.7), (A.27), (B.1) and (B.8) that $\mathcal{M}(\tilde{\times}) = 8 + 21 + (m-1)24 + 9 = 24m + 14$.

Substituting $m$ as in Equation (2.34), we find that $\mathrm{depth}(\tilde{\times})$, $\mathrm{size}(\tilde{\times})$ and $\mathcal{M}(\tilde{\times})$ are of the order $\mathcal{O}(\log(M) + \log(1/\delta))$.

*Step 5.* The aim of this step is to prove the claim that $\forall (a,b) \in [-M,M]^2 : |R_\sigma(\tilde{\times})(a,b)| \leq M^2$. Throughout this step, $m$ is as in Equation (2.34).

All properties of $f_m$ used here and in Section 2.2 directly follow the following explicit expression for $f_m$:

$$\begin{aligned}\forall k \in \{1,\ldots,2^m\}, x \in [x^1_{k-1}, x^1_k] : f_m(x) &= (x^1_{k-1})^2 + (x - x^1_{k-1}) \frac{(x^1_k)^2 - (x^1_{k-1})^2}{x^1_k - x^1_{k-1}} \\ &= 2^{-m}(2k-1)x - k(k-1)2^{-2m}.\end{aligned}$$

To prove the claim, we study the function $E_m : [0,1] \to \mathbb{R} : x \mapsto f_m(x) - x^2$. For $k \in \{1,\ldots,2^m\}$, on $[x^1_{k-1}, x^1_k]$, $E_m$ is given by $E_m(x) = (x - x^1_{k-1})(x^1_k - x)$, it vanishes in $x^1_{k-1}$ and $x^1_k$, it satisfies $E_m(\frac{x^1_{k-1}+x^1_k}{2}) = 2^{-2m-2}$ and on $[x^1_{k-1}, x^1_k]$ it is symmetric around $\frac{x^1_{k-1}+x^1_k}{2}$. Note that $E_m$ is $2^{-m}$-periodic, hence also symmetric around $\{x^1_k\}_{k\in\{1,\ldots,2^m-1\}}$.

We start by showing that $\forall (a,b) \in [0,M]^2 : R_\sigma(\tilde{\times})(a,b) \leq ab \leq M^2$. To that end, we use that $E_m \geq 0$, which implies that it suffices to show $\forall (a,b) \in [0,M]^2 : E_m(\frac{a+b}{2M}) \leq E_m(\frac{a}{2M}) + E_m(\frac{b}{2M})$. We define $x' := \frac{a}{2M}$ and $y' := \frac{b}{2M}$. The $2^{-m}$-periodicity of $E_m$ implies that it suffices to show the result for $x', y' \in [0, 2^{-m}]$. Let us first look at the case in which $x' + y' \leq 2^{-m}$ and assume by contradiction that $E_m(x'+y') > E_m(x') + E_m(y')$. That implies

$$\frac{E_m(x'+y') - E_m(y')}{x'} > \frac{E_m(x')}{x'},$$

which contradicts the strict concavity of $E_m|_{[0,2^{-m}]}$. For $x' + y' > 2^{-m}$ we can use the symmetry of $E_m$ in the point $2^{-m}$ and apply the same argument, but now for $x'' := 2^{-m} - x'$ and $y'' := 2^{-m} - y'$, using $E_m(x'') = E_m(x')$, $E_m(y'') = E_m(y')$, $E_m(x''+y'') = E_m(x'+y')$ and $0 \leq x'' + y'' < 2^{-m}$. This finishes the proof of $\forall (a,b) \in [0,M]^2 : R_\sigma(\tilde{\times})(a,b) \leq ab \leq M^2$.

For $(a,b) \in [0,M]^2$, Equation (2.19) together with $\delta < M^2$ shows that $R_\sigma(\tilde{\times})(a,b) \geq -M^2$.

Note that the case $(a,b) \in [-M,0]^2$ is completely analogous to $(a,b) \in [0,M]^2$, as $R_\sigma(\tilde{\times})$ only depends on $a, b$ through $|a|$, $|b|$ and $|a+b|$, see Equation (2.28). This shows that $|R_\sigma(\tilde{\times})(a,b)| \leq M^2$ for $(a,b) \in [-M,0]^2 \cup [0,M]^2$.

The fact that $R_\sigma(\tilde{\times})(a,b) \leq M^2$ for $(a,b) \in [-M,M]^2$ with $ab < 0$ follows from Equation (2.19) and $\delta < M^2$. Lastly, for $(a,b) \in [-M,M]^2$ with $ab < 0$, the fact that $R_\sigma(\tilde{\times})(a,b) \geq -M^2$ directly follows from Equation (2.28): the first term is non-negative and for the two other terms we know that $f_m(\frac{|a|}{2M}), f_m(\frac{|b|}{2M}) \in [0, \frac{1}{4}]$, because $f_m$ is strictly increasing, $\frac{|a|}{2M}, \frac{|b|}{2M} \leq \frac{1}{2}$ and $f_m(\frac{1}{2}) = \frac{1}{4}$. This finishes the proof that $\forall (a,b) \in [-M,M]^2 : |R_\sigma(\tilde{\times})(a,b)| \leq M^2$ and thereby the proof of Proposition 2.12. □

## B.3 Proof of Proposition 2.14, Step 3

*Step 3.* We now give bounds on the depth, the size and the number of non-zero coefficients in the network. As shown in Step 4 of the proof of Proposition 2.12 in Appendix 2.12, for $m = \lceil \frac{1}{2} \log(\tilde{n}/\delta_{\tilde{\Pi}}) + \frac{1}{2} \rceil$, each $\tilde{\times}_m$-subnetwork has depth $m+2$, size $9m+6$ and not more than $24m+14$ non-zero coefficients. By Remarks A.3 and A.6, we have

$$\begin{aligned}
\operatorname{depth}\left(\tilde{\prod}^n\right) &\overset{(A.12),(A.24)}{\leq} \log_2(\tilde{n})(m+2) = \mathcal{O}(\log(n)\log(n/\delta_{\tilde{\Pi}})),\\
\operatorname{size}\left(\tilde{\prod}^n\right) &\overset{(A.13),(A.26)}{\leq} \tilde{n}(9m+6) + 2\tilde{n} = \mathcal{O}(n\log(n/\delta_{\tilde{\Pi}})),\\
\mathcal{M}\left(\tilde{\prod}^n\right) &\overset{(A.14),(A.27),(2.30),(B.8)}{\leq} \tilde{n}(24m+14) + (9+8)\tilde{n} = \mathcal{O}(n\log(n/\delta_{\tilde{\Pi}})).
\end{aligned}$$

This finishes the proof of Proposition 2.14. □

**Remark B.2** (Cf. [24, Proof Proposition 6.3 pp. 20–23], especially [24, Equation (112) p. 21])**.** *Note that we can implement the auxiliary input values $x_j, n < j \leq \tilde{n}$ by biases in the first computational layer, keeping the rest of the network as it would be for $n = \tilde{n}$, in particular not affecting the number of non-zero coefficients.*

*Note that $\forall x \in [-M,M] : R_\sigma(\tilde{\times}_m)(x,1) = R_\sigma(\tilde{\times}_m)(1,x) = x$ when $\log_2(2M) \in (\mathbb{N}_0)_{\leq m}$. As a result, the auxiliary input values do not introduce errors. To show this, we use properties of the function $E_m(y) = f_m(y) - y^2$ on $[0,1]$ that are discussed in Step 5 of the proof of Proposition 2.12 in Appendix B.2.*

*For $x \in [0,M]$, it follows from the exactness of $f_m$ in the nodes of $\mathcal{T}^1_{2^m}$ and the $2^{-m}$-periodicity of $E_m$ on $[0,1]$ that*

$$|x - R_\sigma(\tilde{\times}_m)(1,x)| = E_m(\tfrac{|1+x|}{2M}) - E_m(\tfrac{1}{2M}) - E_m(\tfrac{|x|}{2M}) = E_m(\tfrac{|x|}{2M} + \tfrac{1}{2M}) - 0 - E_m(\tfrac{|x|}{2M}) = 0.$$

*For $x \in [-M,0]$, there are two cases: when $x < -1$, we have by the same arguments as before that*

$$|x - R_\sigma(\tilde{\times}_m)(1,x)| = E_m(\tfrac{|1+x|}{2M}) - E_m(\tfrac{1}{2M}) - E_m(\tfrac{|x|}{2M}) = E_m(\tfrac{|x|}{2M} - \tfrac{1}{2M}) - 0 - E_m(\tfrac{|x|}{2M}) = 0.$$

*When $-1 \leq x < 0$, then the exactness of $f_m$ in the nodes of $\mathcal{T}^1_{2^m}$ and the symmetry of $E_m$ around $x = \frac{1}{4M}$ imply*

$$|x - R_\sigma(\tilde{\times}_m)(1,x)| = E_m(\tfrac{|1+x|}{2M}) - E_m(\tfrac{1}{2M}) - E_m(\tfrac{|x|}{2M}) = E_m(\tfrac{1}{2M} - \tfrac{|x|}{2M}) - 0 - E_m(\tfrac{|x|}{2M}) = 0.$$

*Instead, we could leave out all $\tilde{\times}_m$-subnetworks that do not depend on the actual input values $x_j, 1 \leq j \leq n$ and replace $\tilde{\times}_m$-subnetworks that depend on actual input values through one of their inputs by identity networks as defined in Definition 2.4.*

## B.4 Proof of Lemma 3.2

*Proof.* It follows from Equations (3.14), (A.26), (A.18) and (3.9) that the number of ReLUs in the first two hidden layers is $2(2^n-1)(2+1)=\mathcal{O}(2^n)$ and from Equations (3.14), (A.27), (A.19) and (3.9) that the number of non-zero coefficients in those layers is $2(2^n-1)(4+3)=\mathcal{O}(2^n)$.

By Equations (3.18), (3.14) and (3.9), the rest of $\tilde{w}$ equals

$$(\mathbb{W},\mathbf{0})\bullet \operatorname{Parallel}(\tilde{\times},\dots,\tilde{\times})\bullet(\mathrm{pr},\mathbf{0})\bullet(\mathrm{Id}_{\mathbb{R}^{2(2^n-1)}},\mathbf{0}), \tag{B.9}$$

which includes $(2^n-1)^2$ parallel $\tilde{\times}$-subnetworks. It follows from Equations (A.1), (A.24) and Proposition 2.12 that the depth is of the order $\mathcal{O}(\log(1/\delta))$. Equations (A.2), (A.26) and Proposition 2.12 show that the number of ReLUs equals $(2^n-1)^2\operatorname{size}(\tilde{\times})=\mathcal{O}(2^{2n}\log(1/\delta))$.

In order to bound the number of non-zero coefficients of these layers, we first compute the third and the last weight matrix of $\tilde{w}$. For

$$A:=\begin{pmatrix}1 & \\ -1 & \\ & 1\\ & -1\\ 1 & 1\\ -1 & -1\end{pmatrix},\qquad \mathbb{A}:=\begin{pmatrix}A & & \\ & \ddots & \\ & & A\end{pmatrix},$$

by Equations (B.9), (2.13), (2.18) and (2.30), the third weight matrix of $\tilde{w}$ equals $\mathbb{A}\,\mathrm{pr}\,\mathrm{Id}_{\mathbb{R}^{2(2^n-1)}}$, which has at most $8(2^n-1)^2$ non-zero components. $8(2^n-1)^2$ also bounds the number of non-zero coefficients in the first computational layer of $\operatorname{Parallel}(\tilde{\times},\dots,\tilde{\times})$, having weight matrix $\mathbb{A}$. For

$$\mathbb{B}:=\begin{pmatrix}B & & \\ & \ddots & \\ & & B\end{pmatrix}\in\mathbb{R}^{1\times 9(2^n-1)^2},\qquad B:=2M^2\begin{pmatrix}-A_{m+1} & -A_{m+1} & A_{m+1}\end{pmatrix},$$

it follows from Equations (B.9), (2.13), (2.18) and (B.8) that the last weight matrix of $\tilde{w}$ equals $\mathbb{W}\mathbb{B}$, which has at most $9(2^n-1)^2$ non-zero components. This number also bounds the number of non-zero coefficients in the last layer of $\operatorname{Parallel}(\tilde{\times},\dots,\tilde{\times})$, which has weight matrix $\mathbb{B}$. As a result, we find that the number of non-zero coefficients of the network in Equation (B.9) is at most $(2^n-1)^2\mathcal{M}(\tilde{\times})$, which by Proposition 2.12 is of the order $\mathcal{O}(2^{2n}\log(1/\delta))$.

The lemma now follows by adding up the depth, the size and the number of non-zero coefficients of the first two computational layers and those quantities for the rest of the network.

The independence of the implied constants of $w$ follows from the fact that all the bounds are explicit functions of $n$, $\delta$, $\operatorname{depth}(\tilde{\times})$, $\operatorname{size}(\tilde{\times})$ and $\mathcal{M}(\tilde{\times})$. By Proposition 2.12, the latter three are bounded in terms of $\delta$. $\square$

## B.5 Proof of Lemma 4.3

*Proof.* This proof follows [75, pp. 7–8], see also [4, p. 328].

Hölder's inequality with $1 = \frac{p}{2} + \frac{2-p}{2}$ gives

$$\sum_{\boldsymbol{\nu}\in\mathcal{F}} b_{\boldsymbol{\nu}}^p = \sum_{\boldsymbol{\nu}\in\mathcal{F}} (b_{\boldsymbol{\nu}}\boldsymbol{\beta}^{-\boldsymbol{\nu}})^p (\boldsymbol{\beta}^{\boldsymbol{\nu}})^p \leq \left(\sum_{\boldsymbol{\nu}\in\mathcal{F}} \boldsymbol{\beta}^{-2\boldsymbol{\nu}} b_{\boldsymbol{\nu}}^2\right)^{p/2} \left(\sum_{\boldsymbol{\nu}\in\mathcal{F}} \boldsymbol{\beta}^{q\boldsymbol{\nu}}\right)^{(2-p)/2}.$$

In addition, using $\boldsymbol{\beta} \in (0,1)^{\mathbb{N}}$, we find

$$\sum_{\boldsymbol{\nu}\in\mathcal{F}} \boldsymbol{\beta}^{q\boldsymbol{\nu}} = \prod_{j\in\mathbb{N}} \left(\sum_{k\in\mathbb{N}_0} \beta_j^{qk}\right) = \prod_{j\in\mathbb{N}} (1-\beta_j^q)^{-1},$$

which converges because $\boldsymbol{\beta} \in \ell^q(\mathbb{N})$, i.e. we have shown $(\boldsymbol{\beta}^{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}} \in \ell^q(\mathcal{F})$ and $(b_{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}} \in \ell^p(\mathcal{F})$.

Next, we fix some $0 < q_\bullet < q$ and define for all $\boldsymbol{\nu} \in \mathcal{F}$

$$\alpha_{\boldsymbol{\nu}} := \begin{cases} j^{-1/q_\bullet} & \boldsymbol{\nu} = \boldsymbol{e}_j, \\ 0 & \text{else,} \end{cases} \qquad \zeta_{\boldsymbol{\nu}} := \max\{\boldsymbol{\beta}^{\boldsymbol{\nu}}, \alpha_{\boldsymbol{\nu}}\}$$

and note that $\boldsymbol{\zeta} \in \ell^q(\mathcal{F}) \hookrightarrow \ell^2(\mathcal{F})$, because $q < 2$. In addition, we note that

$$\sum_{\boldsymbol{\nu}\in\mathcal{F}} (\zeta_{\boldsymbol{\nu}}^{-1} b_{\boldsymbol{\nu}})^2 \leq \sum_{\boldsymbol{\nu}\in\mathcal{F}} (\boldsymbol{\beta}^{-\boldsymbol{\nu}} b_{\boldsymbol{\nu}})^2 < \infty. \tag{B.10}$$

Because $(\alpha_{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}}$ and $(\boldsymbol{\beta}^{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}}$ are decreasing with respect to the ordering on $\mathcal{F}$, $(\zeta_{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}}$ is decreasing as well, hence we can define a bijection $\pi : \mathbb{N} \to \mathcal{F}$ such that $(\zeta_{\pi(j)})_{j\in\mathbb{N}}$ is decreasing and such that $\Lambda_n := \pi(\mathbb{N}_{\leq n})$ is downward closed for all $n \in \mathbb{N}$. Because $(\alpha_{\boldsymbol{e}_j})_{j\in\mathbb{N}}$ and $\boldsymbol{\beta}$ are decreasing, so is $(\boldsymbol{\beta}^{\boldsymbol{e}_j})_{j\in\mathbb{N}}$ and hence $(\zeta_{\boldsymbol{e}_j})_{j\in\mathbb{N}}$, i.e. $\zeta_{\boldsymbol{e}_i} \geq \zeta_{\boldsymbol{e}_j}$ for $i \leq j$, hence $\pi$ can be chosen such that for all $n \in \mathbb{N}$ and for each $j \in \mathbb{N}$: $\boldsymbol{e}_j \in \Lambda_n$ implies $\boldsymbol{e}_i \in \Lambda_n$ for all $i \leq j$.

We next show Equation (4.19). From Hölder's inequality, we get

$$\sum_{\boldsymbol{\nu}\in\Lambda_n^c} b_{\boldsymbol{\nu}} \leq \|(\zeta_{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\Lambda_n^c}\|_{\ell^2(\Lambda_n^c)} \|(\zeta_{\boldsymbol{\nu}}^{-1} b_{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\Lambda_n^c}\|_{\ell^2(\Lambda_n^c)} < \infty. \tag{B.11}$$

By [91, Lemma 2.9 p. 10], $\boldsymbol{\zeta} \in \ell^q(\mathcal{F})$ implies that there exists a $C > 0$ such that $\forall j \in \mathbb{N}$ : $\zeta_{\pi(j)} \leq Cj^{-1/q}$. We find

$$\|(\zeta_{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\Lambda_n^c}\|_{\ell^2(\Lambda_n^c)} \leq \left(C \sum_{j>n} j^{-2/q}\right)^{1/2} \leq C(n^{1-2/q})^{1/2} \leq Cn^{-1/p+1} \tag{B.12}$$

(cf. [21, proof of Equation (2.18) pp. 59–60]), which together with Equations (B.10) and (B.11) shows Equation (4.19).

It remains to show Equation (4.20). By definition of $(\zeta_{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}}$, we have

$$\min\{\zeta_{\boldsymbol{\nu}} : \boldsymbol{\nu} \in \Lambda_n\} \geq \zeta_{\boldsymbol{e}_n} \geq n^{-1/q_\bullet}. \tag{B.13}$$

Using $\beta_1 = \max_{j\in\mathbb{N}} \beta_j < 1$, we have

$$\sup\{\zeta_{\boldsymbol{\nu}} : \boldsymbol{\nu} \in \mathcal{F}, |\boldsymbol{\nu}|_1 = d\} \overset{d\geq 1}{=} \sup\{\boldsymbol{\beta}^{\boldsymbol{\nu}} : \boldsymbol{\nu} \in \mathcal{F}, |\boldsymbol{\nu}|_1 = d\} = (\beta_1)^d. \tag{B.14}$$

For all $n \in \mathbb{N}$, let $d_n \in \mathbb{N}_{\geq 2}$ be such that $n^{-1/q_\bullet} > (\beta_1)^{d_n}$. We then find

$$\max_{\boldsymbol{\nu}\in\Lambda_n} |\boldsymbol{\nu}|_1 < d_n \tag{B.15}$$

by Equations (B.13) and (B.14). Hence, using that $x \mapsto \log(x)/\log(\beta_1)$ is the inverse of $d \mapsto (\beta_1)^d$, it follows from Equation (B.15) that

$$\max_{\boldsymbol{\nu}\in\Lambda_n} |\boldsymbol{\nu}|_1 \leq \log(n^{-1/q_\bullet})/\log(\beta_1) = \mathcal{O}(\log n).$$

This finishes the proof of Lemma 4.3. □

**Remark B.3.** *Note that, a fortiori, Equation (4.19) holds if we define $\{\Lambda_n\}_{n\in\mathbb{N}}$ based on $(\boldsymbol{\beta}^{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}}$ instead of $(\zeta_{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}}$.*

*The sequence $(\zeta_{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}}$ was used instead of $(\boldsymbol{\beta}^{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}}$, because it allowed us to construct $\{\Lambda_n\}_{n\in\mathbb{N}}$ such that Equation (4.20) holds and such that for all $n \in \mathbb{N}$ and for each $j \in \mathbb{N}$: $\boldsymbol{e}_j \in \Lambda_n$ implies $\boldsymbol{e}_i \in \Lambda_n$ for $i \leq j$. As Equation (4.18) is the only information we have about the size of $\{b_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\mathcal{F}}$, using $(\zeta_{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}}$ instead of $(\boldsymbol{\beta}^{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}}$ for the definition of $\{\Lambda_n\}_{n\in\mathbb{N}}$ in general increases $\sum_{\boldsymbol{\nu}\in\Lambda_n^c} b_{\boldsymbol{\nu}}$.*

*However, by the argument in [21, proof of Equation (2.18) pp. 59–60], it follows that for a bijection $\pi : \mathbb{N} \to \mathcal{F}$ for which $(b_{\pi(j)})_{j\in\mathbb{N}}$ is decreasing*

$$\sum_{j>n} b_{\pi(j)} \overset{def}{=} \|(b_{\pi(j)})_{j\in\mathbb{N}_{>n}}\|_{\ell^1(\mathbb{N}_{>n})} \leq Cn^{1-1/p} \tag{B.16}$$

*is equivalent to*

$$\forall j \in \mathbb{N} : b_{\pi(j)} \leq Cj^{-1/p} \tag{B.17}$$

*(one of the implications is similar to Equation (B.12), but for $\ell^1(\Lambda_n^c)$ instead of $\ell^2(\Lambda_n^c)$). Equation (B.17) follows from $(b_{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}} \in \ell^p(\mathcal{F})$ by [91, Lemma 2.9 p. 10] (conversely, Equation (B.17) implies $(b_{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}} \in \ell^{p'}(\mathcal{F})$ for all $p' > p$). That is, $1/p - 1$ is the* best $n$-term convergence rate *of the sequence $(b_{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}} \in \ell^p(\mathcal{F})$ (for the discussion of best $n$-term approximations in comparison to linear approximations, we refer to [21, especially Section 2 pp. 56–60]). Note that we achieved the best $n$-term convergence rate with $\Lambda_n$ constructed in terms of $(\zeta_{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}}$. Hence, we find that the use of $(\zeta_{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}}$ instead of $(\boldsymbol{\beta}^{\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\mathcal{F}}$ does not decrease the bound on the convergence rate.*

## B.6 Proof of Lemma 5.3

*Proof.* This proof mainly follows [75, pp. 21–22], which contains results similar to those derived in [3, Section 3] and [31, Section 2].

From Lemma 4.3, we know that

$$\sum_{\boldsymbol{\nu}\in\Lambda_n^c} \|t_{\boldsymbol{\nu}}\|_V \leq C n^{-1/p_V+1}. \tag{B.18}$$

Next, we aim to choose $(m_{n;\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\Lambda_n}$ such that $\mathcal{N}_n = \sum_{\boldsymbol{\nu}\in\Lambda_n} m_{n;\boldsymbol{\nu}}$ is minimal, under the constraint that

$$\sum_{\boldsymbol{\nu}\in\Lambda_n} \|t_{\boldsymbol{\nu}}\|_X m_{n;\boldsymbol{\nu}}^{-\gamma} \leq n^{-1/p_V+1}. \tag{B.19}$$

We solve this optimisation problem as a continuous optimisation problem, i.e. we calculate $(\tilde{m}_{n;\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\Lambda_n} \subset \mathbb{R}_{>0}$ using the Lagrange multiplier $\lambda$. For $F((\tilde{m}_{n;\boldsymbol{\nu}})_{\boldsymbol{\nu}\in\Lambda_n}, \lambda) := \sum_{\boldsymbol{\nu}\in\Lambda_n} \tilde{m}_{n;\boldsymbol{\nu}} + \lambda(\sum_{\boldsymbol{\nu}\in\Lambda_n} \|t_{\boldsymbol{\nu}}\|_X \tilde{m}_{n;\boldsymbol{\nu}}^{-\gamma} - n^{-1/p_V+1})$, setting $\nabla F = 0$ gives

$$\tilde{m}_{n;\boldsymbol{\nu}} = n^{(1/p_V-1)/\gamma} \|t_{\boldsymbol{\nu}}\|_X^{1/(1+\gamma)} \left( \sum_{\boldsymbol{\nu}\in\Lambda_n} \|t_{\boldsymbol{\nu}}\|_X^{1/(1+\gamma)} \right)^{1/\gamma}, \tag{B.20}$$

$$n^{-1/p_V+1} = \sum_{\boldsymbol{\nu}\in\Lambda_n} \|t_{\boldsymbol{\nu}}\|_X \tilde{m}_{n;\boldsymbol{\nu}}^{-\gamma}. \tag{B.21}$$

Therefore, we define

$$m_{n;\boldsymbol{\nu}} := \max\left\{ \lceil \tilde{m}_{n;\boldsymbol{\nu}} \rceil, m_0 \right\}. \tag{B.22}$$

Note that Equations (B.21) and (B.22) imply Equation (B.19), which together with Equation (B.18) shows the first inequality in Equation (5.4).

It remains to show the second inequality in Equation (5.4), i.e. that for some $C > 0$ for all $n \in \mathbb{N}$

$$n^{-1/p_V+1} \leq C\mathcal{N}_n^{-r}.$$

By Equation (B.22),

$$\mathcal{N}_n = \sum_{\boldsymbol{\nu}\in\Lambda_n} m_{n;\boldsymbol{\nu}} \leq n m_0 + n^{(1/p_V-1)/\gamma} \left( \sum_{\boldsymbol{\nu}\in\Lambda_n} \|t_{\boldsymbol{\nu}}\|_X^{1/(1+\gamma)} \right)^{(1+\gamma)/\gamma}.$$

Let us first assume that $1/p_X - 1 \geq \gamma$. Then, we have

$$\begin{aligned}
\mathcal{N}_n &\leq n m_0 + n^{(1/p_V-1)/\gamma} \|(\|t_{\boldsymbol{\nu}}\|_X)_{\boldsymbol{\nu}\in\mathcal{F}}\|_{\ell^{p_X}(\mathcal{F})}^{1/\gamma} \\
&\leq n^{(1/p_V-1)/\gamma} \left( m_0 + \|(\|t_{\boldsymbol{\nu}}\|_X)_{\boldsymbol{\nu}\in\mathcal{F}}\|_{\ell^{p_X}(\mathcal{F})}^{1/\gamma} \right) \\
&=: n^{(1/p_V-1)/\gamma} C^{1/\gamma}, \\
\Rightarrow \quad n^{-1/p_V+1} &\leq C\mathcal{N}_n^{-r},
\end{aligned}$$

where we have used $1/p_V - 1 \geq 1/p_X - 1 \geq \gamma$ in the second step and $r = \gamma$ in the last step.

For $1/p_X - 1 < \gamma$, which is equivalent to $p_X(1+\gamma) > 1$, we use Hölder's inequality to get

$$\begin{aligned}
\mathcal{N}_n &\leq n m_0 + n^{(1/p_V-1)/\gamma} \left( \left( \sum_{\boldsymbol{\nu}\in\Lambda_n} \|t_{\boldsymbol{\nu}}\|_X^{p_X} \right)^{1/(p_X(1+\gamma))} n^{1-1/(p_X(1+\gamma))} \right)^{(1+\gamma)/\gamma} \\
&\leq n m_0 + n^{(\gamma+1/p_V-1/p_X)/\gamma} \|(\|t_{\boldsymbol{\nu}}\|_X)_{\boldsymbol{\nu}\in\mathcal{F}}\|_{\ell^{p_X}(\mathcal{F})}^{1/\gamma} \\
&\leq n^{(\gamma+1/p_V-1/p_X)/\gamma} \left( m_0 + \|(\|t_{\boldsymbol{\nu}}\|_X)_{\boldsymbol{\nu}\in\mathcal{F}}\|_{\ell^{p_X}(\mathcal{F})}^{1/\gamma} \right) \\
&=: n^{(\gamma+1/p_V-1/p_X)/\gamma} C^{(\gamma+1/p_V-1/p_X)/(\gamma(1/p_V-1))}, \\
\Rightarrow \quad n^{-1/p_V+1} &\leq C \mathcal{N}_n^{-r},
\end{aligned}$$

where we have used $(\gamma + 1/p_V - 1/p_X)/\gamma \geq 1$ in the third step. This finishes the proof of Equation (5.4).

Finally, note that $\forall \boldsymbol{\nu} \in \Lambda_n : m_{n;\boldsymbol{\nu}} \geq m_0$ implies $\mathcal{N}_n \geq m_0 n \geq n$, which finishes the proof of Lemma 5.3. □

## B.7 Proof of Proposition 5.1, Step 4

*Step 4.* In this step, we bound the depth, the size and the number of non-zero coefficients of $\tilde{u}_n$. Before we do so, we note that Equation (5.4) implies that $\log \mathcal{N}_n = \mathcal{O}(\log n)$.

It holds that $\sum_{\boldsymbol{\nu}\in\Lambda_n} \operatorname{size}(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}}) \leq \sum_{\boldsymbol{\nu}\in\Lambda_n} m_{n;\boldsymbol{\nu}} = \mathcal{N}_n$ by definition of $\{\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\Lambda_n}$ in Proposition 3.5. By Proposition 3.5, we also find $\max_{\boldsymbol{\nu}\in\Lambda_n} \operatorname{depth}(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}}) = \mathcal{O}(\log \mathcal{N}_n) = \mathcal{O}(\log n)$ and $\sum_{\boldsymbol{\nu}\in\Lambda_n} \mathcal{M}(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}}) = \mathcal{O}(\mathcal{N}_n)$.

According to Lemma 2.17, we have

$$\begin{aligned}
\max_{\boldsymbol{\nu}\in\Lambda_n} \operatorname{depth}(f_{\boldsymbol{\nu}}) &= \mathcal{O}(\log(n) \log\log(n)), \\
\sum_{\boldsymbol{\nu}\in\Lambda_n} \operatorname{size}(f_{\boldsymbol{\nu}}) &= \mathcal{O}(n \log(n) \log\log(n)), \\
\sum_{\boldsymbol{\nu}\in\Lambda_n} \mathcal{M}(f_{\boldsymbol{\nu}}) &= \mathcal{O}(n \log(n) \log\log(n)).
\end{aligned}$$

By Remarks A.6 and A.5, the latter of which in particular showing that Equations (A.16) and (A.21) hold for Simul-Shared-Parallel, it holds that

$$\begin{aligned}
\operatorname{depth}(P_{\tilde{u}_n}) &\overset{(A.24),(A.16)}{=} \mathcal{O}(\log(n)\log\log(n)),\\
\operatorname{size}(P_{\tilde{u}_n}) &\overset{(A.26),(A.22)}{\leq} \sum_{\boldsymbol{\nu}\in\Lambda_n} \operatorname{size}(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}}) + 2\cdot 2\operatorname{depth}(P_{\tilde{u}_n})\\
&\qquad + \sum_{\boldsymbol{\nu}\in\Lambda_n} \operatorname{size}(f_{\boldsymbol{\nu}}) + 2n\operatorname{depth}(P_{\tilde{u}_n})\\
&= \mathcal{N}_n + \mathcal{O}(n\log(n)\log\log(n)),\\
\mathcal{M}(P_{\tilde{u}_n}) &\overset{(A.27),(A.21)}{\leq} \sum_{\boldsymbol{\nu}\in\Lambda_n} \mathcal{M}(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}}) + \sum_{\boldsymbol{\nu}\in\Lambda_n} \mathcal{M}(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}})\\
&\qquad + 2\cdot 2\operatorname{depth}(P_{\tilde{u}_n}) + \sum_{\boldsymbol{\nu}\in\Lambda_n} \mathcal{M}(f_{\boldsymbol{\nu}})\\
&\qquad + \sum_{\boldsymbol{\nu}\in\Lambda_n} \mathcal{M}(f_{\boldsymbol{\nu}}) + 2n\operatorname{depth}(P_{\tilde{u}_n})\\
&= \mathcal{O}(\mathcal{N}_n + n\log(n)\log\log(n)),
\end{aligned}$$

where in the second to last step the second and the fifth term bound the second term in Equation (A.21) for $\{\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\Lambda_n}$ resp. $\{f_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\Lambda_n}$.

Equation (A.14) implies that

$$\mathcal{M}(\tilde{u}_n) \leq 2\mathcal{M}\left(\left(\begin{pmatrix} 1 & \cdots & 1 \end{pmatrix}, \mathbf{0}\right) \bullet \operatorname{Parallel}(\tilde{\times},\ldots,\tilde{\times}) \bullet (\operatorname{pr}_{\tilde{u}_n}, \mathbf{0})\right) + 2\mathcal{M}(P_{\tilde{u}_n}) \quad \text{(B.23)}$$

([62, Remark 2.6 p. 7], see also [24, Lemma 5.3.(iii) p. 18]). In addition, for

$$A := \begin{pmatrix} 1 & \\ -1 & \\ & 1 \\ & -1 \\ 1 & 1 \\ -1 & -1 \end{pmatrix}, \qquad \mathbb{A} := \begin{pmatrix} A & & \\ & \ddots & \\ & & A \end{pmatrix},$$

by Equations (2.13), (2.18) and (2.30), the first weight matrix of

$$\left(\begin{pmatrix} 1 & \cdots & 1 \end{pmatrix}, \mathbf{0}\right) \bullet \operatorname{Parallel}(\tilde{\times},\ldots,\tilde{\times}) \bullet (\operatorname{pr}_{\tilde{u}_n}, \mathbf{0}) \quad \text{(B.24)}$$

equals $\mathbb{A}\operatorname{pr}_{\tilde{u}_n}$. Because, by construction, multiplication by $\operatorname{pr}_{\tilde{u}_n}$ only permutes the columns of $\mathbb{A}$, the number of non-zero components of $\mathbb{A}\operatorname{pr}_{\tilde{u}_n}$ equals that of $\mathbb{A}$. Besides that, by Equations (2.13), (2.18), (B.8) and (B.2), the last weight matrix of the network in Equation (B.24) is a $1\times 9n$ matrix, it has at most $9n$ non-zero components.

By Proposition 2.12, the $\tilde{\times}$-subnetworks appearing in Equation (5.8) all have depth, size and number of non-zero coefficients of the order $\mathcal{O}(\log(M)+\log(1/\delta_n)) = \mathcal{O}(\log n)$. Using

Equation (5.8) and Remarks A.1 and A.6, we find

$$\begin{aligned}
\operatorname{depth}(\tilde{u}_n) &\overset{(A.12),(A.1),(A.24)}{=} \mathcal{O}(\log(n)\log\log(n)) + \mathcal{O}(\log(1/\delta_n)) \\
&= \mathcal{O}(\log(n)\log\log(n)) = \mathcal{O}(\log(\mathcal{N}_n)\log\log(\mathcal{N}_n)), \\
\operatorname{size}(\tilde{u}_n) &\overset{(A.13),(A.2),(A.26)}{\leq} \mathcal{N}_n + \mathcal{O}(n\log(n)\log\log(n)) + 2(2n) + \mathcal{O}(n\log(1/\delta_n)) \\
&= \mathcal{N}_n + \mathcal{O}(n\log(n)\log\log(n)) = \mathcal{O}(\mathcal{N}_n\log(\mathcal{N}_n)\log\log(\mathcal{N}_n)), \\
\mathcal{M}(\tilde{u}_n) &\overset{(B.23),(A.27)}{\leq} 2\mathcal{O}(\mathcal{N}_n + n\log(n)\log\log(n)) + 2\mathcal{O}(n\log(1/\delta_n)) + 9n \\
&= \mathcal{O}(\mathcal{N}_n + n\log(n)\log\log(n)) = \mathcal{O}(\mathcal{N}_n\log(\mathcal{N}_n)\log\log(\mathcal{N}_n)).
\end{aligned}$$

This finishes Step 4 of the proof of Theorem 5.1. Step 5 in Section 5 gives the convergence rate of $\tilde{u}_n$ in terms of the network size. □

**Remark B.4.** *Note that* $\sum_{\boldsymbol{\nu}\in\Lambda_n} \operatorname{size}(\tilde{I}_{m_{n;\boldsymbol{\nu}}} t_{\boldsymbol{\nu}})$ *is the only term in the upper bound on* $\operatorname{size}(\tilde{u}_n)$ *that need not be of the order* $\mathcal{O}(n\log(n)\log\log(n))$.

*We can reduce the number of ReLUs needed for the implementation of these subnetworks by simultaneously approximating* $\{t_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\Lambda_n}$. *Hat functions only need to be evaluated once, namely the hat functions on the finest grid, used to implement* $\tilde{I}_{m_{n;\boldsymbol{\nu}'}} t_{\boldsymbol{\nu}'}$ *for* $\boldsymbol{\nu}'$ *that satisfies* $m_{n;\boldsymbol{\nu}'} = \max_{\boldsymbol{\nu}\in\Lambda_n} m_{n;\boldsymbol{\nu}}$. *The functions* $\{t_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\Lambda_n}$ *can be approximated by* $\{\tilde{I}_{m_{n;\boldsymbol{\nu}'}} t_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\Lambda_n}$, *which are all linear combinations of the same hat functions. Note that this involves at most* $\max_{\boldsymbol{\nu}\in\Lambda_n} m_{n;\boldsymbol{\nu}}$ *ReLUs, implying that the corresponding size of* $\tilde{u}_n$ *is of the order* $\mathcal{O}(\max_{\boldsymbol{\nu}\in\Lambda_n} m_{n;\boldsymbol{\nu}} + n\log(n)\log\log(n))$.

*The order of the depth is unaffected, while the number of non-zero coefficients is of the order* $\mathcal{O}(n\max_{\boldsymbol{\nu}\in\Lambda_n} m_{n;\boldsymbol{\nu}} + n\log(n)\log\log(n))$, *which is larger than for the case without simultaneous approximation of* $\{t_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu}\in\Lambda_n}$. *This increase in the number of non-zero coefficients can be overcome by using a hierarchical basis of hat functions.*

## References

[1] I. Babuška, F. Nobile, and R. Tempone. A stochastic collocation method for elliptic partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 45(3):1005–1034, 2007. https://doi.org/10.1137/050645142.

[2] I. Babuška, R. Tempone, and G. E. Zouraris. Galerkin finite element approximations of stochastic elliptic partial differential equations. *SIAM Journal on Numerical Analysis*, 42(2):800–825, 2004. https://doi.org/10.1137/S0036142902418680.

[3] M. Bachmayr, A. Cohen, D. Dũng, and C. Schwab. Fully discrete approximation of parametric and stochastic elliptic PDEs. *SIAM Journal on Numerical Analysis*, 55(5):2151–2186, 2017. https://doi.org/10.1137/17M111626X.

[4] M. Bachmayr, A. Cohen, and G. Migliorati. Sparse polynomial approximation of parametric elliptic PDEs. Part I: affine coefficients. *ESIAM: Mathematical Modelling and Numerical Analysis*, 51:321–339, 2017. https://doi.org/10.1051/m2an/2016045.

[5] A. R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945, 1993. https://doi.org/10.1109/18.256500.

[6] A. R. Barron. Approximation and estimation bounds for artificial neural networks. *Machine Learning*, 14(1):115–133, 1994. https://doi.org/10.1007/BF00993164.

[7] C. Beck, W. E, and A. Jentzen. Machine learning approximation algorithms for high-dimensional fully nonlinear partial differential equations and second-order backward stochastic differential equations. pages 1–56, 2017. https://arxiv.org/abs/1709.05963v1.

[8] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. *Advances in Neural Information Processing Systems*, 19:153–160, 2007. https://papers.nips.cc/paper/3048-greedy-layer-wise-training-of-deep-networks.

[9] H. Bölcskei, P. Grohs, G. Kutyniok, and P. Petersen. Memory-optimal neural network approximation. In *Wavelets and Sparsity MMXVII*, volume 10394 of *Proceedings of SPIE*, pages 1–12, 2017. https://doi.org/10.1117/12.2272490.

[10] H. Bölcskei, P. Grohs, G. Kutyniok, and P. Petersen. Optimal approximation with sparsely connected deep neural networks. pages 1–36, 2017. Revised 2018. http://arxiv.org/abs/1705.01714v4.

[11] H. Brezis. *Functional analysis, Sobolev spaces and partial differential equations*. Universitext. Springer, New York, NY, 2011. https://doi.org/10.1007/978-0-387-70914-7.

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

https://doi.org/10.1017/S0962492904000182.

[13] C. Băcuţă, H. Li, and V. Nistor. Differential operators on domains with conical points: precise uniform regularity estimates. pages 1–22, 2016. https://arxiv.org/abs/1605.07907v1.

[14] A. Chkifa, A. Cohen, and C. Schwab. Breaking the curse of dimensionality in sparse polynomial approximation of parametric PDEs. *Journal des Mathématiques Pures et Appliquées*, 103(2):400–428, 2015. https://doi.org/10.1016/j.matpur.2014.04.009.

[15] C.-K. Chui, X. Li, and H. N. Mhaskar. Neural networks for localized approximation. *Mathematics of Computation*, 63:607–623, 1994. https://doi.org/10.1090/S0025-5718-1994-1240656-2.

[16] D.-A. Clevert, T. Unterthiner, and S. Hochreiter. Fast and accurate deep network learning by exponential linear units (ELUs). In *International Conference on Learning Representations 2016*, pages 1–14, 2016. https://arxiv.org/abs/1511.07289v5.

[17] A. Cohen, R. Devore, and C. Schwab. Convergence rates of best $N$-term Galerkin approximations for a class of elliptic sPDEs. *Foundations of Computational Mathematics*, 10(6):615–646, 2010. https://doi.org/10.1007/s10208-010-9072-2.

[18] A. Cohen, R. DeVore, and C. Schwab. Analytic regularity and polynomial approximation of parametric and stochastic elliptic PDE's. *Analysis and Applications*, 9(1):11–47, 2011. https://doi.org/10.1142/S0219530511001728.

[19] N. Cohen, O. Sharir, and A. Shashua. On the expressive power of deep learning: a tensor analysis. In *29th Annual Conference on Learning Theory*, volume 49 of *Proceedings of Machine Learning Research*, pages 698–728, 2016. http://proceedings.mlr.press/v49/cohen16.html.

[20] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, 1989. https://doi.org/10.1007/BF02551274.

[21] R. A. DeVore. Nonlinear approximation. *Acta Numerica*, 7:51–150, 1998. https://doi.org/10.1017/S0962492900002816.

[22] W. E, J. Han, and A. Jentzen. Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations. *Communications in Mathematics and Statistics*, 5(4):349–380, 2017. https://doi.org/10.1007/s40304-017-0117-6.

[23] W. E and B. Yu. The Deep Ritz Method: A deep learning-based numerical algorithm for solving variational problems. *Communications in Mathematics and Statistics*, 6(1):1–12, 2018. https://doi.org/10.1007/S40304-018-0127-Z.

[24] D. Elbrächter, P. Grohs, A. Jentzen, and C. Schwab. DNN expression rate analysis

of high-dimensional PDEs: application to option pricing. Technical report, 2018. Technical report, version 27 August 2018.

[25] H. B. Enderton. *Elements of set theory.* Academic Press, 1977. https://doi.org/10.1016/C2009-0-22079-4.

[26] A. Ern and J.-L. Guermond. *Theory and practice of finite elements*, volume 159 of *Applied Mathematical Sciences.* Springer, New York, NY, 2004. https://doi.org/10.1007/978-1-4757-4355-5.

[27] M. Fujii, A. Takahashi, and M. Takahashi. Asymptotic expansion as prior knowledge in deep learning method for high dimensional BSDEs. 2017. https://arxiv.org/abs/1710.07030v2.

[28] K.-I. Funahashi. On the approximate realization of continuous mappings by neural networks. *Neural Networks*, 2(3):183–192, 1989. https://doi.org/10.1016/0893-6080(89)90003-8.

[29] R. G. Ghanem and P. D. Spanos. Spectral techniques for stochastic finite elements. *Archives of Computational Methods in Engineering*, 4(1):63–100, 1997. https://doi.org/10.1007/BF02818931.

[30] C. J. Gittelson. An adaptive stochastic galerkin method for random elliptic operators. *Mathematics of Computation*, 82(283):1515–1541, 2013. https://doi.org/10.1090/S0025-5718-2013-02654-3.

[31] C. J. Gittelson. Convergence rates of multilevel and sparse tensor approximations for a random elliptic PDE. *SIAM Journal on Numerical Analysis*, 51(4):2426–2447, 2013. https://doi.org/10.1137/110826539.

[32] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*, volume 1. MIT Press, 2016. http://www.deeplearningbook.org.

[33] P. Grisvard. *Elliptic problems in nonsmooth domains*, volume 24 of *Monographs and studies in mathematics.* Pitman Publishing Inc., Marshfield, Massachusetts, 1985.

[34] W. Hackbusch. *Tensor spaces and numerical tensor calculus*, volume 42 of *Springer Series in Computational Mathematics.* Springer, Berlin, Heidelberg, 2012. https://doi.org/10.1007/978-3-642-28027-6.

[35] W. Hackbusch and S. Kühn. A new scheme for the tensor representation. *Journal of Fourier Analysis and Applications*, 15(5):706–722, 2009. https://doi.org/10.1007/s00041-009-9094-9.

[36] J. Han, A. Jentzen, and W. E. Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Sciences of the United States of America*, 115(34):8505–8510, 2018. https://doi.org/10.1073/pnas.1718942115.

[37] G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006. https://doi.org/10.1162/neco.2006.18.7.1527.

[38] G. E. Hinton and T. Shallice. Lesioning an attractor network: Investigations of acquired dyslexia. *Psychological Review*, 98(1):74–95, 1991. https://doi.org/10.1037//0033-295X.98.1.74.

[39] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. https://doi.org/10.1016/0893-6080(89)90020-8.

[40] Y. Khoo, J. Lu, and L. Ying. Solving parametric PDE problems with artificial neural networks. pages 1–17, 2017. Revised 2018. https://arxiv.org/abs/1707.03351v3.

[41] V. Khrulkov, A. Novikov, and I. Oseledets. Expressive power of recurrent neural networks. In *International Conference on Learning Representations 2018*, pages 1–12, 2018. https://openreview.net/forum?id=S1WRibb0Z.

[42] D. P. Kingma and J. L. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations 2015*, pages 1–15, 2015. Revised 2017. https://arxiv.org/abs/1412.6980v9.

[43] I. E. Lagaris, A. C. Likas, and D. I. Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks*, 9(5):987–1000, 1998. https://doi.org/10.1109/72.712178.

[44] I. E. Lagaris, A. C. Likas, and D. G. Papageorgiou. Neural-network methods for boundary value problems with irregular boundaries. *IEEE Transactions on Neural Networks*, 11(5):1041–1049, 2000. https://doi.org/10.1109/72.870037.

[45] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015. https://doi.org/10.1038/nature14539.

[46] H. Lee and I. S. Kang. Neural algorithm for solving differential equations. *Journal of Computational Physics*, 91(1):110–131, 1990. https://doi.org/10.1016/0021-9991(90)90007-N.

[47] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6(6):861–867, 1993. https://doi.org/10.1016/S0893-6080(05)80131-5.

[48] S. Liang and R. Srikant. Why deep neural networks for function approximation? In *International Conference on Learning Representations 2017*, pages 1–17, 2017. https://openreview.net/forum?id=SkpSlKIel.

[49] A. Malek and R. Shekari Beidokhti. Numerical solution for high order differential equations using a hybrid neural network-optimization method. *Applied Mathematics and Computation*, 183(1):260–271, 2006. https://doi.org/10.1016/j.amc.2006.

05.068.

[50] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5(4):115–133, 1943. https://doi.org/10.1007/BF02478259.

[51] H. Mhaskar, Q. Liao, and T. Poggio. Learning functions: When is deep better than shallow. pages 1–12, 2016. https://arxiv.org/abs/1603.00988v4.

[52] H. N. Mhaskar. Neural networks for optimal approximation of smooth and analytic functions. *Neural Computation*, 8(1):164–177, 1996. https://doi.org/10.1162/neco.1996.8.1.164.

[53] H. N. Mhaskar and T. Poggio. Deep vs. shallow networks: An approximation theory perspective. *Analysis and Applications*, 14(6):829–848, 2016. https://doi.org/10.1142/S0219530516400042.

[54] H. Montanelli and Q. Du. Deep ReLU networks lessen the curse of dimensionality. pages 1–15, 2017. Revised 2018. https://arxiv.org/abs/1712.08688v3.

[55] G. F. Montufar, R. Pascanu, K. Cho, and Y. Bengio. On the number of linear regions of deep neural networks. *Advances in Neural Information Processing Systems*, 27:2924–2932, 2014. http://papers.nips.cc/paper/5422-on-the-number-of-linear-regions-of-deep-neural-networks.html.

[56] P.-A. Nitsche. Sparse approximation of singularity functions. *Constructive Approximation*, 21(1):63–81, 2004. https://doi.org/10.1007/s00365-004-0559-4.

[57] F. Nobile, R. Tempone, and C. G. Webster. An anisotropic sparse grid stochastic collocation method for partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 46(5):2411–2442, 2008. https://doi.org/10.1137/070680540.

[58] F. Nobile, R. Tempone, and C. G. Webster. A sparse grid stochastic collocation method for partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 46(5):2309–2345, 2008. https://doi.org/10.1137/060663660.

[59] I. V. Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011. https://doi.org/10.1137/090752286.

[60] R. Pascanu, G. Montúfar, and Y. Bengio. On the number of response regions of deep feedforward networks with piecewise linear activations. pages 1–17, 2013. Revised 2014. https://arxiv.org/abs/1312.6098v5.

[61] D. Perekrestenko, P. Grohs, D. Elbrächter, and H. Bölcskei. The universal approximation power of finite-width deep ReLU networks. pages 1–16, 2018. In review, 32nd Conference on Neural Information Processing Systems (NIPS 2018). https://arxiv.org/abs/1806.01528v1.

[62] P. Petersen and F. Voigtlaender. Optimal approximation of piecewise smooth functions using deep ReLU neural networks. pages 1–54, 2017. Revised 2018. `https://arxiv.org/abs/1709.05289v4`.

[63] A. Pinkus. Approximation theory of the MLP model in neural networks. *Acta Numerica*, 8:143–195, 1999. `https://doi.org/10.1017/S0962492900002919`.

[64] T. Poggio, H. Mhaskar, L. Rosasco, B. Miranda, and Q. Liao. When can deep—but not shallow—networks avoid the curse of dimensionality: A review. *International Journal of Automation and Computing*, 14(5):503–519, 2017. `https://doi.org/10.1007/s11633-017-1054-2`.

[65] M. Ranzato, C. Poultney, S. Chopra, and Y. L. LeCun. Efficient learning of sparse representations with an energy-based model. *Advances in Neural Information Processing Systems*, 19:1137–1144, 2007. `https://papers.nips.cc/paper/3112-efficient-learning-of-sparse-representations-with-an-energy-based-model`.

[66] S. J. Reddi, S. Kale, and S. Kumar. On the convergence of Adam and beyond. In *International Conference on Learning Representations 2018*, pages 1–23, 2018. `https://openreview.net/forum?id=ryQu7f-RZ`.

[67] D. Rolnick and M. Tegmark. The power of deeper networks for expressing natural functions. In *International Conference on Learning Representations 2018*, pages 1–14, 2018. `https://openreview.net/forum?id=SyProzZAW`.

[68] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958. `https://doi.org/10.1037/h0042519`.

[69] K. Rudd. *Solving partial differential equations using artificial neural networks*. PhD thesis, Duke University, 2013. `https://lisc.mae.cornell.edu/PastThesis/KeithRuddPhD.pdf`.

[70] S. Ruder. An overview of gradient descent optimization algorithms. pages 1–14, 2016. Revised 2017. `https://arxiv.org/abs/1609.04747v2`.

[71] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986. `https://doi.org/10.1038/323533a0`.

[72] V. Runde. *A taste of topology*. Universitext. Springer, New York, NY, 2008. `https://doi.org/10.1007/0-387-28387-0`.

[73] J. Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015. `http://dx.doi.org/10.1016/j.neunet.2014.09.003`.

[74] C. Schwab and C. J. Gittelson. Sparse tensor discretizations of high-dimensional parametric and stochastic PDEs. *Acta Numerica*, 20:291–467, 2011. `https://doi.org/10.1017/S0962492911000055`.

[75] C. Schwab and J. Zech. Deep learning in high dimension: neural network expression rates for generalized polynomial chaos expansions in UQ. Technical Report 2017-57, Seminar for Applied Mathematics, ETH Zürich, Switzerland, 2017. Revised 2018. To appear in Analysis and Applications (Singapore) (2018). `http://www.sam.math.ethz.ch/sam_reports/reports_final/reports2017/2017-57_rev2.pdf`.

[76] J. Sirignano and K. Spiliopoulos. DGM: A deep learning algorithm for solving partial differential equations. pages 1–31, 2017. Revised 2018. `https://arxiv.org/abs/1708.07469v4`.

[77] M. Telgarsky. Representation benefits of deep feedforward networks. pages 1–5, 2015. `https://arxiv.org/abs/1509.08101v2`.

[78] M. Telgarsky. Neural networks and rational functions. In *34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3387–3393, 2017. `http://proceedings.mlr.press/v70/telgarsky17a.html`.

[79] R. A. Todor and C. Schwab. Convergence rates for sparse chaos approximations of elliptic problems with stochastic coefficients. *IMA Journal of Numerical Analysis*, 27(2):232–261, 2007. `https://doi.org/10.1093/imanum/drl025`.

[80] A. Tychonoff. Über die topologische Erweiterung von Räumen. *Mathematische Annalen*, 102(1):544–561, 1930. `https://doi.org/10.1007/BF01782364`.

[81] P. Urysohn. Zum Metrisationsproblem. *Mathematische Annalen*, 94(1):309–315, 1925. `https://doi.org/10.1007/BF01208661`.

[82] B. Widrow. An adaptive "Adaline" neuron using chemical "memistors". Technical Report 1553-2, 1960. `http://www-isl.stanford.edu/~widrow/papers/t1960anadaptive.pdf`.

[83] B. Widrow. Adaptive sampled-data systems. In *IFAC Moscow Congress Record*, pages 1–6, 1960. `http://www-isl.stanford.edu/~widrow/papers/c1960adaptivesampled.pdf`.

[84] B. Widrow and M. E. Hoff. Adaptive switching circuits. In *Wescon Convention Record*, number 4, pages 96–104, 1960. `http://www-isl.stanford.edu/~widrow/papers/c1960adaptiveswitching.pdf`.

[85] N. Wiener. The homogeneous chaos. *American Journal of Mathematics*, 60(4):897–936, 1938. `https://doi.org/10.2307/2371268`.

[86] N. Wiener. *Cybernetics: Or control and communication in the animal and the machine.* Hermann, Paris, first edition, 1948.

[87] D. Xiu. *Numerical methods for stochastic computations: a spectral method approach.* Princeton University Press, Princeton, N.J., 2010. `https://ebookcentral.proquest.com/lib/ethz/detail.action?docID=557164`.

[88] D. Xiu and G. E. Karniadakis. The Wiener-Askey polynomial chaos for stochastic differential equations. *SIAM Journal on Scientific Computing*, 24(2):619–644, 2002. https://doi.org/10.1137/S1064827501387826.

[89] D. Yarotsky. Error bounds for approximations with deep ReLU networks. *Neural Networks*, 94:103–114, 2017. https://doi.org/10.1016/j.neunet.2017.07.002.

[90] J. Zech, D. Dũng, and C. Schwab. Multilevel approximation of parametric and stochastic PDEs. Technical Report 2018-05, Seminar for Applied Mathematics, ETH Zürich, Switzerland, 2018. http://www.sam.math.ethz.ch/sam_reports/reports_final/reports2018/2018-05_fp.pdf.

[91] J. Zech and C. Schwab. Convergence rates of high dimensional Smolyak quadrature. Technical Report 2017-27, Seminar for Applied Mathematics, ETH Zürich, Switzerland, 2017. http://www.sam.math.ethz.ch/sam_reports/reports_final/reports2017/2017-27_fp.pdf.

**ETH**

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

## Declaration of originality

---

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

**Title of work** (in block letters):

ReLU DNN expression of sparse gpc expansion in uncertainty quantification

**Authored by** (in block letters):
*For papers written by groups the names of all authors are required.*

| | |
|---|---|
| **Name(s):** | **First name(s):** |
| Opschoor | Joost |
| | |
| | |
| | |
| | |

With my signature I confirm that
- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

| | |
|---|---|
| **Place, date** | **Signature(s)** |
| Zurich, 3 September 2018 | |

*For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.*