1.5 Solving Parametric PDEs with Neural Networks

Martin Eigel and Janina Schütte

Deutsche Forschungsgemeinschaft

Fig. 1: *Deutsche Forschungsgemeinschaft*

FoDL priority program

Fig. 2: *SPP 2298 Theoretical Foundations of Deep Learning*

Deep learning has emerged as a versatile numerical tool in many application areas, recently extending its reach beyond natural language processing, image recognition and generation also into the realms of solving *partial differential equations* (PDEs). PDEs are mathematical models for a wide range of physical phenomena used in science and engineering, ranging from heat conduction, electrostatics, and (quantum) mechanics to fluid dynamics. Parametric PDEs (pPDEs) generalize the concept by adding (possibly infinitely many) parameters describing the data of the models. The significance of solving pPDEs lies in their crucial role for practical problems where uncertainties or a large set of data realizations have to be taken into account. Understanding the impact of varying model parameters, determined by prescribed probability distributions, is essential for predicting outcome statistics and for making reliable simulation-driven decisions. Deep learning offers a modern approach to tackle the high complexity of pPDEs. By training deep neural networks on appropriate data sets, these models learn intricate relationships between parameters and the corresponding system behavior. This expedites the solution process and, therefore, enables one to observe different states of the system under the influence of very many different parameters and efficiently evaluate statistical properties.

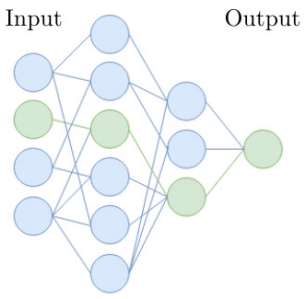

Fig. 3: *Visualization of a generic NN architecture*

Deep learning. Deep learning evolves around training *neural networks* (NNs), a class of function representations loosely inspired by the interconnections of neurons in a human brain. There are two main challenges when utilizing NNs for a problem at hand. First, the *architecture* of the NN has to be decided upon. This includes the number of neurons and the assumed connections between them. Second, the weights in the network, i.e., the strengths of the connections, need to be *learned*. The depth in deep learning comes from the way neurons are organized in an NN. It corresponds to the number of layers that an input passes through, as depicted in Figure 3. To solve pPDEs, it is possible to employ a special network architecture, called *convolutional neural networks* (CNNs), which are typically used for image data. It can be proven that the number weights and their values can be chosen such that the parameter-to-solution map is approximated arbitrarily well. To find these weights, the network has to be trained with many data points, consisting of pairs of parameters and corresponding solutions of the pPDE at hand. Since computing high resolution solutions with classical algorithms for given parameters as training data is expensive, we proposed a multilevel decomposition of the solutions, which works efficiently with a combination of a large set of computationally inexpensive coarse resolution data and a small set of expensive high resolution data.

Deep learning framework

Different deep learning methods have been proposed to tackle the task of finding a functional representation of the parameter-to-solution map. This can, e.g., be based on a reduced basis method [5], deep operator networks [6], Fourier neural operators [7], and physics-informed NNs (PINNs) [8]. Despite their large expressivity, these architectures have difficulties to live up to the

promising theoretical results in practical experiments, which is largely due to the challenging training process. In this article, a neural network architecture based on CNNs is constructed that can be trained efficiently in practice, reaching state-of-the-art accuracy for a class of pPDEs. Moreover, theoretical guarantees can be shown for the existence of CNNs that are able to approximate the solution operator arbitrarily well.

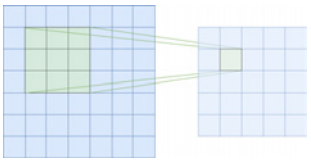

Fig. 4: *Visualization of the application of a kernel in a CNN architecture*

CNNs are a special class of neural networks designed for tasks involving visual data such as image recognition and computer vision. Applying the action of a network on an image involves the application of local kernels. Here, a kernel (equivalent to a small image) is multiplied to regions of the input image and summed over to yield an entry of the output image as illustrated in Figure 4. Furthermore, a nonlinear activation function, which decides how important a neuron is, such as the rectified linear unit function, or pooling operators are usually applied. An efficient architecture of CNNs is the U-Net, originally developed for biomedical image segmentation. It gets its name from the shape of its visualization as shown in Figure 5. By first contracting the input resolution to capture context of the input image and then expanding it to allow for localization, U-Nets are able to capture fine-scale as well as coarse-scale features of an image. Skip connections incorporate earlier computed results in later steps on each resolution level of the images, which gives the architecture the power to combine the information of different scales.

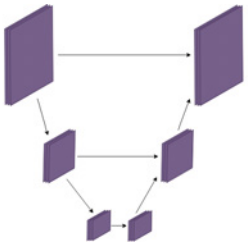

Fig. 5: *Visualization of a U-Net architecture*

When applied to solving pPDEs on a uniform grid, we have shown that the U-Net is able to approximate one step in a classical multigrid solver as used in finite element (FE) simulations; see [1]. Multigrid solvers leverage a hierarchical discretization of the problem to achieve fast convergence, combining coarse-scale solutions with smoothing and fine-scale corrections. When multiplying a $0-1$ mask to each image in the U-Net, we have shown that successive subspace algorithms can be approximated by the NN architecture; see [2]. This is required when the pPDE should be solved on an adaptively refined grid instead of a uniform grid, leading to more efficient approximations of the solution operator.

Parametric partial differential equations

There exist well-developed numerical methods to solve partial differential equations, specifically FE and finite volume methods. These methods can be extended to accommodate the setting of pPDEs. Prominent examples are, e.g., adaptive stochastic Galerkin FEM [4] or the variational Monte Carlo method [3], which are based on a polynomial chaos expansion and low-rank tensor approximations. The newly introduced NN-based methods are sample based and can be applied to data generated with a large class of linear and nonlinear pPDEs. In the analysis, the focus lies on the parametric (stationary diffusion) *Darcy problem*, which is also used as a benchmark problem in the numerical experiments and in many papers on uncertainty quantification. We define the problem in the following: Let $D \subset \mathbb{R}^d$ be a sufficiently smooth physical domain, here $\bar{D} = [0,1]^2$ is considered. Let $\Gamma \subset \mathbb{R}^{\mathbb{N}}$ be a possibly countable infinite-dimensional parameter space and $f : D \to \mathbb{R}$. We aim to approximate the map $u : \Gamma \times D \to \mathbb{R}$, which satisfies

$$\begin{cases} \nabla_x \cdot (\kappa(y,x)\nabla_x u(y,x)) = f(x) & \text{for } x \in D, \\ u(x) = 0 & \text{for } x \in \partial D \end{cases} \tag{1}$$

for the coefficient field $\kappa : \Gamma \times D \to \mathbb{R}$ and with derivatives with respect to the variable $x \in D$. The dependence of the parameter field on the parameter vector y is often characterized by a truncated *Karhunen–Loéve expansion* (KLE) for some $p \in \mathbb{N}$ given by

$$\kappa(y, x) = a_0(x) + \sum_{k=1}^{p} y_k a_k(x),$$

with $a_k : D \to \mathbb{R}$ for $k = 0, \dots, p$. We consider two problem settings, determined by the structure of κ. First, the *cookie problem* is defined for $\Gamma = [0, 1]^p$, $y \in \Gamma$ with uniformly distributed $y_k \sim U[0, 1]$ for $k = 1, \dots, p$, $a_0 = 0.1$, and $a_k = \chi_{D_k}$ indicator functions equal to 1 in D_k and 0 otherwise, where D_k are disks with fixed centers. Second, the *log-normal Gaussian coefficient* is defined for $\Gamma = \mathbb{R}^p$, normally distributed $y_k \sim \mathcal{N}(0, 1)$, and $\kappa(y, x) = \exp(\tilde{\kappa}(y, x))$, where $\tilde{\kappa}$ is defined by the KLE with $a_0 = 0$ and $\|a_k\|_\infty = 0.1k^{-2}$ for $k = 1, \dots, p$. A visualization of the cookie parameters, the log-normal coefficients, and the corresponding FE solutions can be seen in Figure 6 in the top and bottom row, respectively.

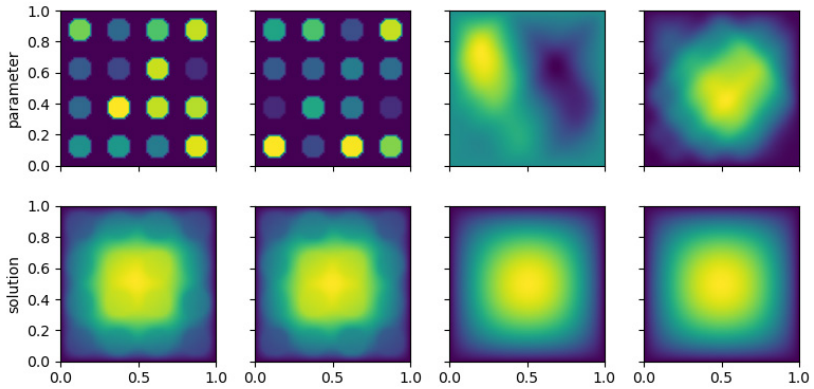

Fig. 6: *Realizations of parameter fields for the cookie problem (first two columns), the log-normal coefficient (last two columns), and the corresponding solutions of the parametric Darcy problem*

Solving parametric PDEs with neural networks

Multilevel decomposition. Training large NNs with many parameters is a key challenge in deep learning due to the nonlinearity and nonconvexity of the model class. Many aspects of the learning algorithm, such as the chosen optimizer, the learning rate, the batch sizes, and several more have to be considered. To circumvent the handling of large intractable NNs, we propose a multilevel decomposition of the data as depicted in Figure 7 and described in the following.

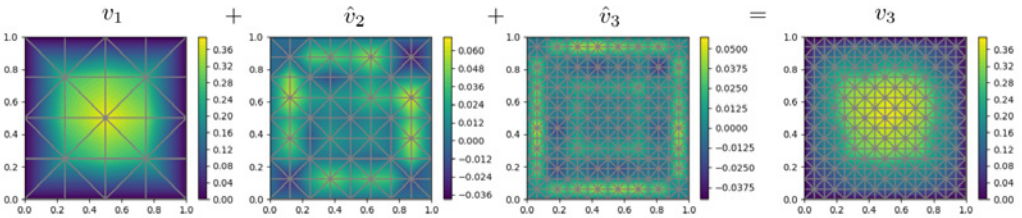

Fig. 7: *Visualization of the multilevel decomposition*

Let $V_1 \subset H_0^1(D)$ be a conforming piecewise linear (P_1) FE space on a uniform grid and let V_ℓ be the space on the uniformly refined grid of $V_{\ell-1}$ for $\ell = 2, \dots, L$, where $L \in \mathbb{N}$ is the number of levels. For the Galerkin projection v_ℓ of the solution of the PDE (1) for a fixed parameter $y \in \Gamma$ onto the space V_ℓ, let $\hat{v}_\ell := v_\ell - v_{\ell-1}$ be the correction of $v_{\ell-1}$ to the solution on the finer grid for $\ell = 2, \dots, L$. Then, individual CNNs can be trained to approximate the solution on the coarse grid v_1 and the corrections on the finer grids $\hat{v}_\ell$ for $\ell = 2, \dots L$. Due to solution smoothness, the entries of the corrections decrease exponentially over the levels. This yields a rapid decrease of importance and, therefore, of required accuracy in the fine grid-corrections, which can be translated to a reduced number of parameters in the CNN and a small number of expensive fine-grid training samples. In contrast, on coarse grids the accuracy has to be high, but only few FE coefficients have to be approximated, by which the network sizes can be controlled. Therefore, the decomposition leads to two advantages. On the one hand, the CNNs to be trained do not consist of a large amount of parameters. On the other hand, a small set of expensive-to-compute solutions on a fine grid suffices for training.

Network architecture. We derived a CNN architecture that can provably approximate the solution of the parametric Darcy problem in the conforming P_1 FE function space V_L on a uniformly refined square grid in the following sense: Denote by f_L and $\kappa_L(y)$ the coefficients of the interpolation of f and $\kappa(y, \cdot)$ in V_L, respectively. Assume that $\kappa_L(y)$ is uniformly bounded for all $y \in \Gamma$. We have shown that there exists a constant $C > 0$ such that for any $\varepsilon > 0$ there exists a CNN $\Psi : \mathbb{R}^{2\times \dim V_L} \to \mathbb{R}^{\dim V_L}$ with the number of parameters bounded by $CL\log(\varepsilon^{-1}) + CL^2$ such that

$$\|\Psi(\kappa_L, f_L) - v_L^c(y)\|_{H^1(D)} \leq \varepsilon \|f\|_*,$$

where $v_L^c(y)$ denotes the coefficients $v_L(y)$ with parameter field $\kappa_L(y, \cdot)$. The architecture of the considered CNN is depicted in Figure 8. Here, the first two yellow images depict the input containing the coefficients of the interpolated parameter field and the right-hand side in V_L. The orange outputs of the network correspond to the solution and corrections of the solution $v_1, \hat{v}_2, \dots, \hat{v}_L$. The different colored U-Nets have been shown to approximate multigrid solvers in the spaces $V_1, \dots, V_L$.

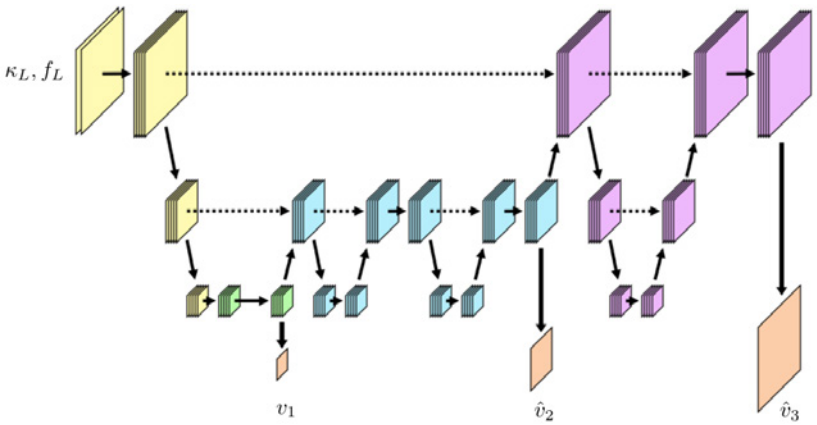

Fig. 8: *Example architecture of our multilevel CNN approximating the solution on $L = 3$ levels*

Numerical results. In the experiments, the overall error $\mathcal{E}^{\text{ref}}$ and the network error $\mathcal{E}^{\text{net}}$ are considered separately, where $\mathcal{E}^{\text{ref}}$ is the average $H^1(D)$ distance of the network output to a reference solution on a twice-uniformly refined grid, and $\mathcal{E}^{\text{net}}$ is the average $H^1(D)$ error of the network output to the Galerkin solution on the same grid.

problem	parameter dimension p	$\mathcal{E}^{\text{ref}}$	$\mathcal{E}^{\text{net}}$
cookie	16	7.09e−2 ± 1.87e−5	9.41e−4 ± 1.12e−4
	64	9.73e−2 ± 1.16e−5	1.85e−3 ± 1.38e−4
log-normal	10	5.33e−3 ± 2.93e−6	2.15e−4 ± 6.18e−5
	200	5.34e−3 ± 3.03e−6	3.00e−4 ± 1.43e−5

Table 1: *Multilevel CNN evaluated on test sets to solve the parametric Darcy problem*

In the experiments, it is evident that the network error is at least one magnitude smaller than the reference error, which implies that the reduction of the overall error can only be achieved by considering finer resolutions or better-suited FE discretization spaces. Uniformly refining the spaces quickly leads to an infeasible computational complexity as the number of parameters grows exponentially in the number of levels.

Adaptive refinement. Efficient discretization spaces V_h can be built iteratively to control the overall error $\mathcal{E} = \|u(y,\cdot) - u_h(y,\cdot)\|_{H^1_0(D)}$ for any $y \in \Gamma$, where u is the solution of (1) in $H^1_0(D)$ and u_h is its Galerkin projection onto V_h. The space is built in an adaptive way by starting with a coarse space V_1 and repeating the procedure:

$$\text{Solve on current space} \to \text{Estimate } \mathcal{E} \text{ locally} \to \text{Mark large error regions} \to \text{Refine marked regions.}$$

The resulting meshes for the cookie problem are visualized in Figure 9. A CNN architecture can be derived that approximates every step of the above iteration. Let $\mathcal{F} : \mathbb{R}^{\sum_{\ell=1}^{L} \dim V_\ell} \to V_L$ map coefficients to the corresponding FE function. Then, there exists a constant $C > 0$ such that for any $\varepsilon > 0$, number of iterations $K \in \mathbb{N}$, and local refinements $L \in \mathbb{N}$, there exists a CNN $\Psi : \mathbb{R}^{2\times \dim V_L} \to \mathbb{R}^{\sum_{\ell=1}^{L} \dim V_\ell}$ with at most $CLK \log(\varepsilon^{-1})$ parameters such that

$$\|u(y,\cdot) - \mathcal{F}(\Psi(\kappa_L(y), f_L))\|_{H^1(D)} \le \|u(y,\cdot) - u_h(y,\cdot)\|_{H^1(D)} + \varepsilon.$$

Conclusions and outlook

Convolutional neural networks are an efficient tool to solve pPDEs and are amenable to a thorough mathematical analysis. Theoretically, small approximation errors can be achieved with network sizes growing only logarithmically with the inverse of the required error bound. Numerically, the multilevel decomposition of the data allows for efficient training of small networks and with only few expensive and many cheap data samples. Solving a pPDE for a given parameter with the trained neural network only takes one forward pass of the network, which can be evaluated quickly to obtain statistical estimates of the solution. Interesting directions for future research are the application of this network architecture to more challenging (nonlinear, instationary) PDEs and using it in the context of statistical inverse problems.

References

[1] C. Heiss, I. Gühring, M. Eigel, *Multilevel CNNs for parametric PDEs*, J. Mach. Learn. Res., **24** (2024), pp. 373/1–373/42.

[2] M. Eigel, J. Schütte, *Adaptive neural networks for parametric PDEs*, WIAS Preprint, in preparation.

[3] M. Eigel, R. Schneider, P. Trunschke, S. Wolf, *Variational Monte Carlo–bridging concepts of machine learning and high-dimensional partial differential equations*, Adv. Comput. Math., **45** (2019), pp. 2503–2532.

[4] M. Eigel, C. J. Gittelson, C. Schwab, E. Zander, *Adaptive stochastic Galerkin FEM*, Comput. Methods Appl. Mech. Engrg., **270** (2014), pp. 247–269.

[5] G. Kutyniok, P. Petersen, M. Raslan, R. Schneider, *A theoretical analysis of deep neural networks and parametric PDEs*, Constr. Approx., **55** (2022), pp. 73–125.

[6] C. Marcati, C. Schwab, *Exponential convergence of deep operator networks for elliptic partial differential equations*, SIAM J. Numer. Anal., **61** (2022), pp. 1513–1545.

[7] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, A. Anandkumar, *Fourier neural operator for parametric partial differential equations*, arxiv Preprint no. 2010.08895, 2021.

[8] M. Raissi, P. Perdikaris, P.E. Karniadakis, *Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations*, J. Comput. Phys., **378** (2019), pp. 686–707.

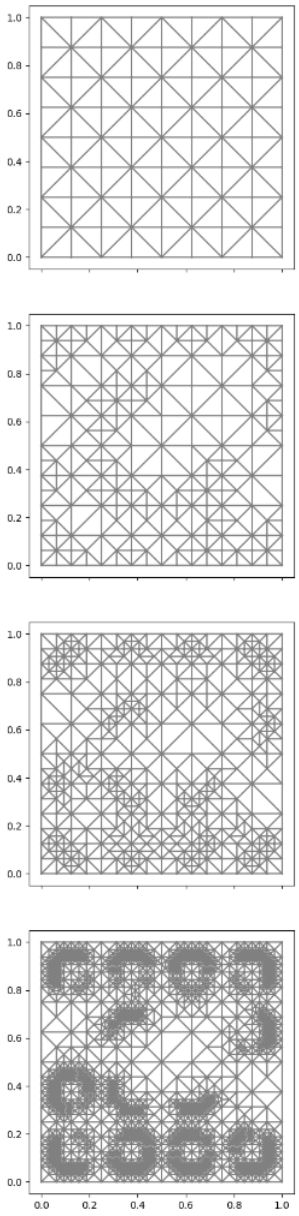

Fig. 9: *Adaptively refined meshes for the cookie problem with* $p = 16$ *inclusions*