Deep Learning meets Parametric Partial Differential Equations

Gitta Kutyniok

(Technische Universität Berlin and University of Tromsø)

Banach Center – Oberwolfach Graduate Seminar:
Mathematics of Deep Learning
Polish Academy of Sciences, Będlewo, November 17 – 23, 2019

Why Parametric PDEs?

Parameter dependent families of PDEs arise in basically any branch of science and engineering.

Some Exemplary Problem Classes:

- Complex design problems
- Inverse problems
- Optimization tasks
- Uncertainty quantification
- ...

The number of parameters can be

- finite (physical properties such as domain geometry, ...)
- infinite (modeling of random stochastic diffusion field, ...)

Parametric Map:

$$\mathcal{Y} \ni y \;\mapsto\; u_y \in \mathcal{H} \quad \text{such that} \quad \mathcal{L}(u_y, y) = f_y.$$

Example: The Parametric Poisson Equation

For $f : \Omega \subset \mathbb{R}^d \to \mathbb{R}$, consider the parametric Poisson Equation

$$\begin{cases} \nabla(a \cdot \nabla u) = f, \text{ in } \Omega, \\ u = 0, \quad \text{on } \partial\Omega. \end{cases}, \quad a \in \mathcal{A} \subset \{g : \Omega \to \mathbb{R}, \text{ bounded}\}.$$

If $\mathcal{A}$ is compact, there exist functions $(g_i)_{i=1}^{\infty}$ such that for every $a \in \mathcal{A}$ there exist $(y_i)_{i=1}^{\infty} \subset \mathbb{R}$ with

$$a = \sum_{i=1}^{\infty} y_i g_i.$$

Example: The Parametric Poisson Equation

For $f : \Omega \subset \mathbb{R}^d \to \mathbb{R}$, consider the parametric Poisson Equation

$$\begin{cases} \nabla(a \cdot \nabla u) = f, \text{ in } \Omega, \\ u = 0, \quad \text{on } \partial\Omega. \end{cases}, \quad a \in \mathcal{A} \subset \{g : \Omega \to \mathbb{R}, \text{ bounded}\}.$$

If $\mathcal{A}$ is compact, there exist functions $(g_i)_{i=1}^{\infty}$ such that for every $a \in \mathcal{A}$ there exist $(y_i)_{i=1}^{\infty} \subset \mathbb{R}$ with

$$a = \sum_{i=1}^{\infty} y_i g_i.$$

We restrict ourselves to the case that

$$a = \sum_{i=1}^{p} y_i g_i$$

for some $p \in \mathbb{N}$ which is potentially very large.

Parametric Partial Differential Equations

Our Setting: We will consider parameter-dependent equations of the form

$$b_y\left(u_y, v\right) = f_y(v), \quad \text{for all } y \in \mathcal{Y},\ v \in \mathcal{H},$$

where

(i) $\mathcal{Y} \subseteq \mathbb{R}^p$ (p large) is the *compact parameter set*,

(ii) $\mathcal{H}$ is a Hilbert space,

(ii) $b_y : \mathcal{H} \times \mathcal{H} \to \mathbb{R}$ is a *symmetric, uniformally coercive, and uniformally continuous bilinear form*,

(iv) $f_y \in \mathcal{H}^*$ is the *uniformly bounded, parameter-dependent right-hand side*,

(v) $u_y \in \mathcal{H}$ is the *solution*.

We assume the *solution manifold*

$$S(\mathcal{Y}) := \{u_y : y \in \mathcal{Y}\}$$

to be compact in $\mathcal{H}$.

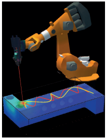
Multi-Query Situation

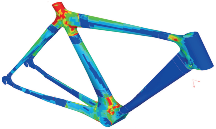
Many applications require solving the parametric PDE multiple times for different parameters:

$$\mathbb{R}^p \supset \mathcal{Y} \ni y = (y_1, \ldots, y_p) \quad \mapsto \quad u_y \in \mathcal{H}$$

Examples:

- Design optimization
- Optimal control
- Routine analysis
- Uncertainty quantification
- Inverse problems





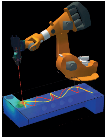
Multi-Query Situation

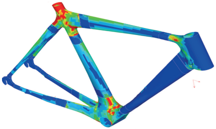
Many applications require solving the parametric PDE multiple times for different parameters:

$$\mathbb{R}^p \supset \mathcal{Y} \ni y = (y_1, \ldots, y_p) \quad \mapsto \quad u_y \in \mathcal{H}$$

Examples:

- Design optimization
- Optimal control
- Routine analysis
- Uncertainty quantification
- Inverse problems





Curse of Dimensionality:

Computational cost often much too high!

High-Fidelity Approximations

Galerkin Approach: Instead of $b_y(u_y, v) = f_y(v)$, we solve

$$b_y\left(u_y^h, v\right) = f_y(v) \qquad \text{for all } v \in U^h,$$

where $U^h \subset \mathcal{H}$ with $D := \dim\left(U^h\right) < \infty$ is the *high-fidelity discretization* and $u_y^h \in U^h$ is the solution.

Cea's Lemma: u_y^h is (up to a constant) a best approximation of u_y by elements in U^h.

High-Fidelity Approximations

Galerkin Approach: Instead of $b_y(u_y, v) = f_y(v)$, we solve

$$b_y\left(u_y^h, v\right) = f_y(v) \qquad \text{for all } v \in U^h,$$

where $U^h \subset \mathcal{H}$ with $D := \dim\left(U^h\right) < \infty$ is the *high-fidelity discretization* and $u_y^h \in U^h$ is the solution.

Cea's Lemma: u_y^h is (up to a constant) a best approximation of u_y by elements in U^h.

Galerkin Solution: Let $(\varphi_i)_{i=1}^D$ be a basis for U^h. Then u_y^h satisfies

$$u_y^h = \sum_{i=1}^{D} (\mathbf{u}_y^h)_i \varphi_i \qquad \text{with} \qquad \mathbf{u}_y^h := \left(\mathbf{B}_y^h\right)^{-1} \mathbf{f}_y^h \in \mathbb{R}^D,$$

where $\mathbf{B}_y^h := (b_y(\varphi_j, \varphi_i))_{i,j=1}^D$ and $\mathbf{f}_y^h := (f_y(\varphi_i))_{i=1}^D$.

What about Deep Neural Networks?

Parametric Map:

$$\mathcal{Y} \ni y \mapsto \mathbf{u}_y^{\mathrm{h}} \in \mathbb{R}^D \quad \text{such that} \quad b_y\left(u_y^h, v\right) = f_y(v)\ \forall v \in U^h.$$

Can a Neural Network Approximate the Parametric Map?

What about Deep Neural Networks?

Parametric Map:

$$\mathcal{Y} \ni y \mapsto \mathbf{u}_y^{\mathrm{h}} \in \mathbb{R}^D \quad \text{such that} \quad b_y\left(u_y^h, v\right) = f_y(v)\ \forall v \in U^h.$$

Can a Neural Network Approximate the Parametric Map?

Advantages:

- After training, extremely rapid computation of the map.
- Flexible, universal approach.

Questions: Let $\varepsilon > 0$.

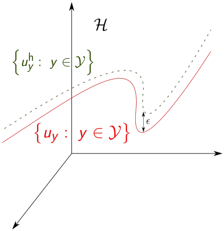
(1) Does there exist a neural network Φ such that

$$\|\Phi - \mathbf{u}_y^{\mathrm{h}}\| \leq \varepsilon \qquad \text{for all } y \in \mathcal{Y}?$$

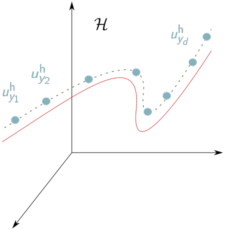
(2) How does the complexity of Φ depend on p and D?

Reduced Basis Method: Key Ideas

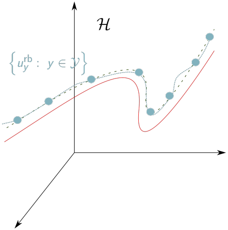
High-Fidelity Discretization:



Key Idea:



Offline (slow):
Compute snap shots



Online (fast):
Compute solutions for new parameters

Reduced Basis Method: Details

Assumption: For all $\varepsilon > \varepsilon_0$, there exists $U^{\mathrm{rb}} \subset \mathcal{H}$, $d(\varepsilon) \coloneqq \dim\left(U^{\mathrm{rb}}\right) \ll D$ such that

$$\sup_{y \in \mathcal{Y}} \inf_{w \in U^{\mathrm{rb}}} \|u_y - w\|_{\mathcal{H}} \leq \varepsilon.$$

$\rightsquigarrow$ Optimality through *Kolmogorov N-width*!

Reduced Basis Method: Details

Assumption: For all $\varepsilon > \varepsilon_0$, there exists $U^{\mathrm{rb}} \subset \mathcal{H}$, $d(\varepsilon) := \dim\left(U^{\mathrm{rb}}\right) \ll D$ such that

$$\sup_{y \in \mathcal{Y}} \inf_{w \in U^{\mathrm{rb}}} \|u_y - w\|_{\mathcal{H}} \leq \varepsilon.$$

$\rightsquigarrow$ Optimality through *Kolmogorov N-width*!

Transfer to Reduced Basis:

- Let $U^{\mathrm{rb}} := \mathrm{span}\left(\psi_i\right)_{i=1}^{d(\varepsilon)}$ with $\left(\psi_i\right)_{i=1}^{d(\varepsilon)} = \left(\sum_{j=1}^{D} \mathbf{V}_{j,i}\varphi_j\right)_{i=1}^{d(\varepsilon)}$.

Reduced Basis Method: Details

Assumption: For all $\varepsilon > \varepsilon_0$, there exists $U^{\mathrm{rb}} \subset \mathcal{H}$, $d(\varepsilon) \coloneqq \dim\left(U^{\mathrm{rb}}\right) \ll D$ such that

$$\sup_{y \in \mathcal{Y}} \inf_{w \in U^{\mathrm{rb}}} \|u_y - w\|_{\mathcal{H}} \leq \varepsilon.$$

$\rightsquigarrow$ Optimality through *Kolmogorov N-width*!

Transfer to Reduced Basis:

- Let $U^{\mathrm{rb}} := \mathrm{span}\,(\psi_i)_{i=1}^{d(\varepsilon)}$ with $(\psi_i)_{i=1}^{d(\varepsilon)} = \left(\sum_{j=1}^{D} \mathbf{V}_{j,i}\varphi_j\right)_{i=1}^{d(\varepsilon)}$.
- Set $\mathbf{B}_y^{\mathrm{rb}} := (b_y(\psi_j, \psi_i))_{i,j=1}^{d(\varepsilon)} = \mathbf{V}^T \mathbf{B}_y^{\mathrm{h}} \mathbf{V} \in \mathbb{R}^{d(\varepsilon) \times d(\varepsilon)}$.
- Set $\mathbf{f}_y^{\mathrm{rb}} := (f_y(\psi_i))_{i=1}^{d(\varepsilon)} = \mathbf{V}^T \mathbf{f}_y^{\mathrm{h}} \in \mathbb{R}^{d(\varepsilon)}$.

Reduced Basis Method: Details

Assumption: For all $\varepsilon > \varepsilon_0$, there exists $U^{\mathrm{rb}} \subset \mathcal{H}$, $d(\varepsilon) \coloneqq \dim\left(U^{\mathrm{rb}}\right) \ll D$ such that

$$\sup_{y \in \mathcal{Y}} \inf_{w \in U^{\mathrm{rb}}} \|u_y - w\|_{\mathcal{H}} \leq \varepsilon.$$

$\rightsquigarrow$ Optimality through *Kolmogorov N-width*!

Transfer to Reduced Basis:

- Let $U^{\mathrm{rb}} := \operatorname{span}(\psi_i)_{i=1}^{d(\varepsilon)}$ with $(\psi_i)_{i=1}^{d(\varepsilon)} = \left(\sum_{j=1}^{D} \mathbf{V}_{j,i}\varphi_j\right)_{i=1}^{d(\varepsilon)}$.
- Set $\mathbf{B}_y^{\mathrm{rb}} := (b_y(\psi_j, \psi_i))_{i,j=1}^{d(\varepsilon)} = \mathbf{V}^T \mathbf{B}_y^{\mathrm{h}} \mathbf{V} \in \mathbb{R}^{d(\varepsilon) \times d(\varepsilon)}$.
- Set $\mathbf{f}_y^{\mathrm{rb}} := (f_y(\psi_i))_{i=1}^{d(\varepsilon)} = \mathbf{V}^T \mathbf{f}_y^{\mathrm{h}} \in \mathbb{R}^{d(\varepsilon)}$.

Galerkin Solution: $(\sup_{y \in \mathcal{Y}} \|u_y - u_y^{\mathrm{rb}}\|_{\mathcal{H}} \leq C\varepsilon)$

$$u_y^{\mathrm{rb}} =$$

Reduced Basis Method: Details

Assumption: For all $\varepsilon > \varepsilon_0$, there exists $U^{\mathrm{rb}} \subset \mathcal{H}$, $d(\varepsilon) \coloneqq \dim\left(U^{\mathrm{rb}}\right) \ll D$ such that

$$\sup_{y \in \mathcal{Y}} \inf_{w \in U^{\mathrm{rb}}} \|u_y - w\|_{\mathcal{H}} \leq \varepsilon.$$

$\rightsquigarrow$ Optimality through *Kolmogorov N-width*!

Transfer to Reduced Basis:

- Let $U^{\mathrm{rb}} := \mathrm{span}\,(\psi_i)_{i=1}^{d(\varepsilon)}$ with $(\psi_i)_{i=1}^{d(\varepsilon)} = \left(\sum_{j=1}^{D} \mathbf{V}_{j,i}\varphi_j\right)_{i=1}^{d(\varepsilon)}$.
- Set $\mathbf{B}_y^{\mathrm{rb}} := (b_y(\psi_j, \psi_i))_{i,j=1}^{d(\varepsilon)} = \mathbf{V}^T \mathbf{B}_y^{\mathrm{h}} \mathbf{V} \in \mathbb{R}^{d(\varepsilon) \times d(\varepsilon)}$.
- Set $\mathbf{f}_y^{\mathrm{rb}} := (f_y(\psi_i))_{i=1}^{d(\varepsilon)} = \mathbf{V}^T \mathbf{f}_y^{\mathrm{h}} \in \mathbb{R}^{d(\varepsilon)}$.

Galerkin Solution: $\left(\sup_{y \in \mathcal{Y}} \|u_y - u_y^{\mathrm{rb}}\|_{\mathcal{H}} \leq C\varepsilon\right)$

$$u_y^{\mathrm{rb}} = \sum_{i=1}^{d(\varepsilon)} \left(\mathbf{u}_y^{\mathrm{rb}}\right)_i \psi_i =$$

Reduced Basis Method: Details

Assumption: For all $\varepsilon > \varepsilon_0$, there exists $U^{\mathrm{rb}} \subset \mathcal{H}$, $d(\varepsilon) := \dim\left(U^{\mathrm{rb}}\right) \ll D$ such that

$$\sup_{y \in \mathcal{Y}} \inf_{w \in U^{\mathrm{rb}}} \|u_y - w\|_{\mathcal{H}} \leq \varepsilon.$$

$\rightsquigarrow$ Optimality through *Kolmogorov N-width*!

Transfer to Reduced Basis:

- Let $U^{\mathrm{rb}} := \operatorname{span}(\psi_i)_{i=1}^{d(\varepsilon)}$ with $(\psi_i)_{i=1}^{d(\varepsilon)} = \left(\sum_{j=1}^{D} \mathbf{V}_{j,i}\varphi_j\right)_{i=1}^{d(\varepsilon)}$.
- Set $\mathbf{B}_y^{\mathrm{rb}} := (b_y(\psi_j, \psi_i))_{i,j=1}^{d(\varepsilon)} = \mathbf{V}^T \mathbf{B}_y^{\mathrm{h}} \mathbf{V} \in \mathbb{R}^{d(\varepsilon) \times d(\varepsilon)}$.
- Set $\mathbf{f}_y^{\mathrm{rb}} := (f_y(\psi_i))_{i=1}^{d(\varepsilon)} = \mathbf{V}^T \mathbf{f}_y^{\mathrm{h}} \in \mathbb{R}^{d(\varepsilon)}$.

Galerkin Solution: $(\sup_{y \in \mathcal{Y}} \|u_y - u_y^{\mathrm{rb}}\|_{\mathcal{H}} \leq C\varepsilon)$

$$u_y^{\mathrm{rb}} = \sum_{i=1}^{d(\varepsilon)} \left(\mathbf{u}_y^{\mathrm{rb}}\right)_i \psi_i = \sum_{j=1}^{D} \left(\mathbf{V}\mathbf{u}_y^{\mathrm{rb}}\right)_j \varphi_j =$$

Reduced Basis Method: Details

Assumption: For all $\varepsilon > \varepsilon_0$, there exists $U^{\mathrm{rb}} \subset \mathcal{H}$, $d(\varepsilon) := \dim\left(U^{\mathrm{rb}}\right) \ll D$ such that

$$\sup_{y \in \mathcal{Y}} \inf_{w \in U^{\mathrm{rb}}} \|u_y - w\|_{\mathcal{H}} \leq \varepsilon.$$

$\rightsquigarrow$ Optimality through *Kolmogorov N-width*!

Transfer to Reduced Basis:

- Let $U^{\mathrm{rb}} := \mathrm{span}\,(\psi_i)_{i=1}^{d(\varepsilon)}$ with $(\psi_i)_{i=1}^{d(\varepsilon)} = \left(\sum_{j=1}^{D} \mathbf{V}_{j,i}\varphi_j\right)_{i=1}^{d(\varepsilon)}$.
- Set $\mathbf{B}_y^{\mathrm{rb}} := (b_y(\psi_j, \psi_i))_{i,j=1}^{d(\varepsilon)} = \mathbf{V}^T \mathbf{B}_y^{\mathrm{h}} \mathbf{V} \in \mathbb{R}^{d(\varepsilon) \times d(\varepsilon)}$.
- Set $\mathbf{f}_y^{\mathrm{rb}} := (f_y(\psi_i))_{i=1}^{d(\varepsilon)} = \mathbf{V}^T \mathbf{f}_y^{\mathrm{h}} \in \mathbb{R}^{d(\varepsilon)}$.

Galerkin Solution: $\left(\sup_{y \in \mathcal{Y}} \|u_y - u_y^{\mathrm{rb}}\|_{\mathcal{H}} \leq C\varepsilon\right)$

$$u_y^{\mathrm{rb}} = \sum_{i=1}^{d(\varepsilon)} \left(\mathbf{u}_y^{\mathrm{rb}}\right)_i \psi_i = \sum_{j=1}^{D} \left(\mathbf{V}\mathbf{u}_y^{\mathrm{rb}}\right)_j \varphi_j = \sum_{j=1}^{D} \left(\mathbf{V}(\mathbf{B}_y^{\mathrm{rb}})^{-1}\mathbf{V}^T \mathbf{f}_y^{\mathrm{h}}\right)_j \varphi_j.$$

Deep Learning Approaches to Parametric PDEs

Solving Parametric PDEs with Neural Networks:

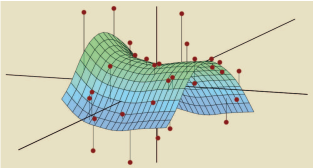
- K. Lee, K. Carlberg; 2018
  Learn a parametrization of $S(\mathcal{Y})$ represented by neural networks.
- J.S. Hesthaven, S. Ubbiali; 2018
  Find reduced basis and then train neural networks to predict coefficients of solution in that basis.
- Schwab, Zech; 2018
  Assume that there is a reduced basis of polynomial chaos functions. These and the coefficients can be efficiently represented by neural networks.

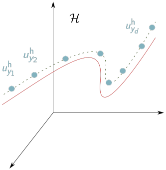
Our Theoretical Analysis

Statistical Learning Problem = Parametric Problem?

Comparison/Similarities:

Statistical Learning Problem	Parametric Problem
Learn $f : X \to Y$	
Distribution on $X \times Y$	
Loss function $\mathcal{L}\colon Y \times Y \to \mathbb{R}^+$	
Training data $(\mathbf{x}_i, \mathbf{y}_i)_{i=1}^N$	
Training phase $\sum_{i=1}^N \mathcal{L}(f(\mathbf{x}_i), \mathbf{y}_i)$	

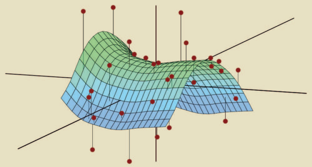


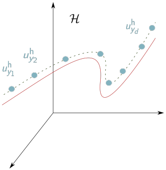


Statistical Learning Problem = Parametric Problem?

Comparison/Similarities:

Statistical Learning Problem	Parametric Problem
Learn $f: X \to Y$	Learn $\mathcal{Y} \ni y \mapsto u_y \in H$
Distribution on $X \times Y$	
Loss function $\mathcal{L}: Y \times Y \to \mathbb{R}^+$	
Training data $(\mathbf{x}_i, \mathbf{y}_i)_{i=1}^N$	
Training phase $\sum_{i=1}^N \mathcal{L}(f(\mathbf{x}_i), \mathbf{y}_i)$	

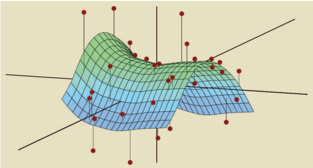


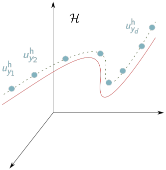


Statistical Learning Problem = Parametric Problem?

Comparison/Similarities:

Statistical Learning Problem	Parametric Problem
Learn $f : X \to Y$	Learn $\mathcal{Y} \ni y \mapsto u_y \in H$
Distribution on $X \times Y$	PDE
Loss function $\mathcal{L}\colon Y \times Y \to \mathbb{R}^+$	
Training data $(\mathbf{x}_i, \mathbf{y}_i)_{i=1}^N$	
Training phase $\sum_{i=1}^N \mathcal{L}(f(\mathbf{x}_i), \mathbf{y}_i)$	

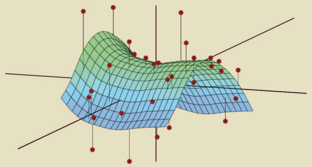


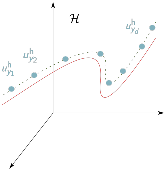


Statistical Learning Problem = Parametric Problem?

Comparison/Similarities:

Statistical Learning Problem	Parametric Problem
Learn $f : X \to Y$	Learn $\mathcal{Y} \ni y \mapsto u_y \in H$
Distribution on $X \times Y$	PDE
Loss function $\mathcal{L}$: $Y \times Y \to \mathbb{R}^+$	Metric on state space
Training data $(\mathbf{x}_i, \mathbf{y}_i)_{i=1}^N$	
Training phase $\sum_{i=1}^N \mathcal{L}(f(\mathbf{x}_i), \mathbf{y}_i)$	

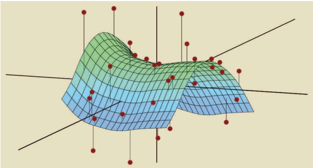


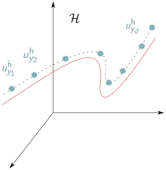


Statistical Learning Problem = Parametric Problem?

Comparison/Similarities:

Statistical Learning Problem	Parametric Problem
Learn $f : X \to Y$	Learn $\mathcal{Y} \ni y \mapsto u_y \in H$
Distribution on $X \times Y$	PDE
Loss function $\mathcal{L}\colon Y \times Y \to \mathbb{R}^+$	Metric on state space
Training data $(\mathbf{x}_i, \mathbf{y}_i)_{i=1}^N$	Snapshots
Training phase $\sum_{i=1}^N \mathcal{L}(f(\mathbf{x}_i), \mathbf{y}_i)$	

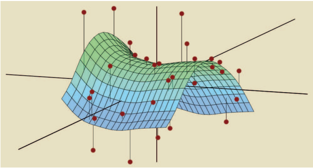


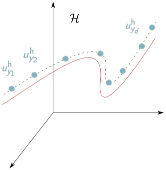


Statistical Learning Problem = Parametric Problem?

Comparison/Similarities:

Statistical Learning Problem	Parametric Problem
Learn $f : X \to Y$	Learn $\mathcal{Y} \ni y \mapsto u_y \in H$
Distribution on $X \times Y$	PDE
Loss function $\mathcal{L}$: $Y \times Y \to \mathbb{R}^+$	Metric on state space
Training data $(\mathbf{x}_i, \mathbf{y}_i)_{i=1}^N$	Snapshots
Training phase $\sum_{i=1}^N \mathcal{L}(f(\mathbf{x}_i), \mathbf{y}_i)$	Offline phase





Our Results: Discrete Version

Theorem (K, Petersen, Raslan, Schneider; 2019):
We assume the following:

- For all $\varepsilon > 0$, there exists $d(\varepsilon) \ll D$, $\mathbf{V} \in \mathbb{R}^{D\times d(\varepsilon)}$, such that for all $y \in \mathcal{Y}$ there exists $\mathbf{B}_y^{\mathrm{rb}} \in \mathbb{R}^{d(\varepsilon)\times d(\varepsilon)}$ with

$$\|\mathbf{V}(\mathbf{B}_y^{\mathrm{rb}})^{-1}\mathbf{V}^T\mathbf{f}_y^{\mathrm{h}} - \mathbf{u}_y^{\mathrm{h}}\| \leq \varepsilon.$$

- There exist ReLU neural networks Φ^B and Φ^f of size $O(\mathrm{poly}(p)d(\varepsilon)^2\mathrm{polylog}(\varepsilon))$ such that, for all $y \in \mathcal{Y}$,

$$\|\Phi^B - \mathbf{B}_y^{\mathrm{rb}}\| \leq \varepsilon \quad \text{and} \quad \|\Phi^f - \mathbf{f}_y^{\mathrm{rb}}\| \leq \varepsilon.$$

Then there exists a ReLU neural network Φ of size $O(d(\varepsilon)^3\mathrm{polylog}(\varepsilon) + D + \mathrm{poly}(p)d(\varepsilon)^2\mathrm{polylog}(\varepsilon))$ such that

$$\|\Phi - \mathbf{u}_y^{\mathrm{h}}\| \leq \varepsilon \qquad \text{for all } y \in \mathcal{Y}.$$

Our Results: Continuous Version

Theorem (K, Petersen, Raslan, Schneider; 2019):

Let $(\psi_i)_{i=1}^{d(\varepsilon)}$ denote the reduced basis. We assume in addition the following:

- There exist ReLU neural networks $(\Phi_i)_{i=1}^{d(\varepsilon)}$ of size $O(\text{polylog}(\varepsilon))$ such that $\|\Phi_i - \psi_i\|_{\mathcal{H}} \leq \varepsilon$ for all $i = 1, \ldots, d(\varepsilon)$.

Then there exists a ReLU neural network Φ of size $O(d(\varepsilon)^3 \text{polylog}(\varepsilon) + \text{poly}(p) d(\varepsilon)^2 \text{polylog}(\varepsilon))$ such that

$$\|\Phi - u_y\|_{\mathcal{H}} \leq \varepsilon \qquad \text{for all } y \in \mathcal{Y}.$$

Our Results: Continuous Version

Theorem (K, Petersen, Raslan, Schneider; 2019):

Let $(\psi_i)_{i=1}^{d(\varepsilon)}$ denote the reduced basis. We assume in addition the following:

- There exist ReLU neural networks $(\Phi_i)_{i=1}^{d(\varepsilon)}$ of size $O(\text{polylog}(\varepsilon))$ such that $\|\Phi_i - \psi_i\|_{\mathcal{H}} \leq \varepsilon$ for all $i = 1, \ldots, d(\varepsilon)$.

Then there exists a ReLU neural network Φ of size $O(d(\varepsilon)^3 \text{polylog}(\varepsilon) + \text{poly}(p) d(\varepsilon)^2 \text{polylog}(\varepsilon))$ such that

$$\|\Phi - u_y\|_{\mathcal{H}} \leq \varepsilon \qquad \text{for all } y \in \mathcal{Y}.$$

Remark: The hypotheses are fulfilled, for example, by

- Diffusion equations,
- Linear elasticity equations.

Possible Impact

Theoretical Foundation:

- Theoretical underpinning for the empirical success of neural networks for parametric problems.

$\rightsquigarrow$ *Can the link between deep learning techniques for parametric PDE problems with approximation theory be further exploited?*

Possible Impact

Theoretical Foundation:

- Theoretical underpinning for the empirical success of neural networks for parametric problems.

$\rightsquigarrow$ *Can the link between deep learning techniques for parametric PDE problems with approximation theory be further exploited?*

Understanding the Circumvention of the Curse of Dimensionality:

- We showed that those neural networks circumvent the curse of dimensionality and essentially only depend on the size of the reduced basis.

$\rightsquigarrow$ *What general structural components are required to avoid the curse?*

Possible Impact

Theoretical Foundation:

- Theoretical underpinning for the empirical success of neural networks for parametric problems.

$\rightsquigarrow$ *Can the link between deep learning techniques for parametric PDE problems with approximation theory be further exploited?*

Understanding the Circumvention of the Curse of Dimensionality:

- We showed that those neural networks circumvent the curse of dimensionality and essentially only depend on the size of the reduced basis.

$\rightsquigarrow$ *What general structural components are required to avoid the curse?*

Identifying Suitable Architectures:

- Neural networks of sufficient depth and size are able to yield very efficient approximations.

$\rightsquigarrow$ *How do they perform for stochastic gradient descent?*

Key Idea of the Proof

Main Task: Approximate $\mathbf{V}(\mathbf{B}_y^{\mathrm{rb}})^{-1}\mathbf{V}^T\mathbf{f}_y^{\mathrm{h}}$ by a ReLU neural network and control its size!

Key Idea of the Proof

Main Task: Approximate $\mathbf{V}(\mathbf{B}_y^{\mathrm{rb}})^{-1}\mathbf{V}^T\mathbf{f}_y^{\mathrm{h}}$ by a ReLU neural network and control its size!

Step 1 (Scalar Multiplication from Yarotsky; 2017):
For $g(x) := \min\{2x, 2-2x\}$ and $g_s := g \circ \ldots \circ g$ (s times), we have

$$x^2 = \lim_{n\to\infty} x - \sum_{s=1}^{n} \frac{g_s(x)}{2^{2s}} \quad \text{for all } x \in [0,1].$$

Key Idea of the Proof

Main Task: Approximate $\mathbf{V}(\mathbf{B}_y^{\mathrm{rb}})^{-1}\mathbf{V}^T\mathbf{f}_y^{\mathrm{h}}$ by a ReLU neural network and control its size!

Step 1 (Scalar Multiplication from Yarotsky; 2017):
For $g(x) := \min\{2x, 2-2x\}$ and $g_s := g \circ \ldots \circ g$ (s times), we have

$$x^2 = \lim_{n\to\infty} x - \sum_{s=1}^{n} \frac{g_s(x)}{2^{2s}} \quad \text{for all } x \in [0,1].$$

Also, g can be represented by a neural network due to

$$g(x) = 2\rho(x) - 4\rho(x - \frac{1}{2}) + 2\rho(x-2) \quad \text{for all } x \in [0,1].$$

Key Idea of the Proof

Main Task: Approximate $\mathbf{V}(\mathbf{B}_y^{\mathrm{rb}})^{-1}\mathbf{V}^T\mathbf{f}_y^{\mathrm{h}}$ by a ReLU neural network and control its size!

Step 1 (Scalar Multiplication from Yarotsky; 2017):
For $g(x) := \min\{2x, 2-2x\}$ and $g_s := g \circ \ldots \circ g$ (s times), we have

$$x^2 = \lim_{n\to\infty} x - \sum_{s=1}^{n} \frac{g_s(x)}{2^{2s}} \quad \text{for all } x \in [0,1].$$

Also, g can be represented by a neural network due to

$$g(x) = 2\rho(x) - 4\rho(x - \frac{1}{2}) + 2\rho(x-2) \quad \text{for all } x \in [0,1].$$

Moreover,

$$xz = 1/4((x+z)^2 - (x-z)^2) \quad \text{for all } x, z \in \mathbb{R}.$$

$\Longrightarrow$ *Scalar multiplication on $[-1,1]^2$ can be ε-approximated by a neural network of size $\mathcal{O}(\log_2(1/\varepsilon))$.*

Key Idea of the Proof

Step 2 (Multiplication):

A matrix multiplication of two matrices of size $d \times d$ can be performed by d^3 scalar multiplications.

$\Longrightarrow$ *Matrix multiplication can be ε-approximated by a neural network of size $\mathcal{O}(d(\varepsilon)^3 \log_2(1/\varepsilon))$.*

Key Idea of the Proof

Step 2 (Multiplication):

A matrix multiplication of two matrices of size $d \times d$ can be performed by d^3 scalar multiplications.

$\Longrightarrow$ *Matrix multiplication can be ε-approximated by a neural network of size $\mathcal{O}(d(\varepsilon)^3 \log_2(1/\varepsilon))$.*

Step 3 (Inversion):

- Neural networks can approximate matrix polynomials.
- Neural networks can the inversion operator $\mathbf{A} \mapsto \mathbf{A}^{-1}$ using

$$\sum_{s=0}^{m} \mathbf{A}^s \longrightarrow (\mathbf{Id}_{\mathbb{R}^d} - \mathbf{A})^{-1} \quad \text{as } m \to \infty.$$

$\Longrightarrow$ *Matrix inversion can be ε-approximated by a neural network of size $\mathcal{O}(d(\varepsilon)^3 \log_2^q(1/\varepsilon))$ for a constant $q > 0$.*

Key Idea of the Proof

Step 4 (Discrete Parametric Map w.r.t Reduced Basis):

- Now use the assumptions on $\mathbf{B}_y^{\mathrm{rb}}$ and $\mathbf{f}_y^{\mathrm{rb}}$.

$\Longrightarrow$ *The map $y \mapsto (\mathbf{B}_y^{\mathrm{rb}})^{-1}\mathbf{f}_y^{\mathrm{rb}}$ can be ε-approximated by a neural network Φ^{rb} of size $\mathcal{O}(d(\varepsilon)^3 \log_2^q(1/\varepsilon) + poly(p) d(\varepsilon)^2 \log_2^q(1/\varepsilon))$.*

Key Idea of the Proof

Step 4 (Discrete Parametric Map w.r.t Reduced Basis):

- Now use the assumptions on $\mathbf{B}_y^{\mathrm{rb}}$ and $\mathbf{f}_y^{\mathrm{rb}}$.

$\Longrightarrow$ *The map* $y \mapsto (\mathbf{B}_y^{\mathrm{rb}})^{-1}\mathbf{f}_y^{\mathrm{rb}}$ *can be* ε*-approximated by a neural network* Φ^{rb} *of size* $\mathcal{O}(d(\varepsilon)^3 \log_2^q(1/\varepsilon) + \mathit{poly}(p) d(\varepsilon)^2 \log_2^q(1/\varepsilon))$.

For Theorem 1:

- Now use the assumption that every element from the reduced basis can be approximately represented in the high-fidelity basis.
- Consider then $\mathbf{V} \circ \Phi^{\mathrm{rb}}$.

$\Longrightarrow$ *The discrete parametric map can be* ε*-approximated by a neural network of size* $\mathcal{O}(d(\varepsilon)^3 \log_2^q(1/\varepsilon) + d(\varepsilon)D + \mathit{poly}(p) d(\varepsilon)^2 \log_2^q(1/\varepsilon))$.

Key Idea of the Proof

Step 4 (Discrete Parametric Map w.r.t Reduced Basis):

- Now use the assumptions on $\mathbf{B}_y^{\mathrm{rb}}$ and $\mathbf{f}_y^{\mathrm{rb}}$.

$\Longrightarrow$ *The map* $y \mapsto (\mathbf{B}_y^{\mathrm{rb}})^{-1}\mathbf{f}_y^{\mathrm{rb}}$ *can be* ε*-approximated by a neural network* Φ^{rb} *of size* $\mathcal{O}(d(\varepsilon)^3 \log_2^q(1/\varepsilon) + poly(p) d(\varepsilon)^2 \log_2^q(1/\varepsilon))$.

For Theorem 1:

- Now use the assumption that every element from the reduced basis can be approximately represented in the high-fidelity basis.
- Consider then $\mathbf{V} \circ \Phi^{\mathrm{rb}}$.

$\Longrightarrow$ *The discrete parametric map can be* ε*-approximated by a neural network of size* $\mathcal{O}(d(\varepsilon)^3 \log_2^q(1/\varepsilon) + d(\varepsilon)D + poly(p) d(\varepsilon)^2 \log_2^q(1/\varepsilon))$.

For Theorem 2:

- Now use the assumption that neural networks can approximate each element of the reduced basis.

$\Longrightarrow$ *The continuous parametric map can be* ε*-approximated by a neural network of size* $\mathcal{O}(d(\varepsilon)^3 \log_2^q(1/\varepsilon) + poly(p) d(\varepsilon)^2 \log_2^q(1/\varepsilon))$.

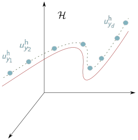
Conclusions

What to take Home...?

Deep Learning:

- Impressive performance *in combination with classical mathematical methods* (Inverse Problems, PDEs, ...).
- Theoretical foundation of neural networks almost entirely missing: *Expressivity, Learning, Generalization, and Explainability*.

Parametric PDEs:



- One key problem is the *curse of dimensionality*.
- The *reduced basis method* uses the low-dimensionality of the solution manifold.

A Theoretical Analysis:

- We derive *upper bounds on the complexity* of ReLU neural networks to approximate parametric maps.
- Those neural networks *do not suffer from the curse of dimensionality* and essentially only depend on the size of the reduced basis.
- We provide a *construction for such neural networks*.

THANK YOU!

References available at:

www.math.tu-berlin.de/∼kutyniok

Code available at:

www.ShearLab.org