

PINNs as numerical methods based on nonlinear approximation spaces

Victor Michel-Dansac*

Atelier “Fabrique ton PINN 2”, Paris, 20 October 2026

*MACARON project-team, Université de Strasbourg, CNRS, Inria, IRMA, France



- 1. Approximation [DeVore, 1998, Acta Numerica] 1
 - 1.1. Linear approximation spaces 3
 - 1.2. Nonlinear approximation spaces 8
- 2. Solving PDEs in nonlinear approximation spaces 16
 - 2.1. Elliptic problems 17
 - 2.2. Parameter-dependent elliptic problems 21
 - 2.3. Time-dependent problems 23
 - 2.4. Conclusion 26

1. Approximation

[DeVore, 1998, Acta Numerica]

Numerical analysis and supervised learning

Function approximation is paramount in several fields of applied mathematics. Examples include **supervised learning** and **numerical analysis**.

- **Numerical analysis**: Given a differential operator $\mathcal{D}$, we seek a function u such that $\mathcal{D}[u] = f$.
- **Supervised learning**: Given a set of data $(x_k, y_k)_{k=1}^M$, we seek a function u such that for all $k \in \llbracket 1, M \rrbracket$, $u(x_k) = y_k$.

In both cases, the sought solution u is in some **infinite-dimensional Hilbert space** $\mathcal{H}$, say $\mathcal{H} = L^2$.

Main idea: reduce this problem to a finite-dimensional one by approximating u in a **finite-dimensional function space** $V_N \subset \mathcal{H}$.

Plan of the presentation

In this talk, we will:

1. First, approximate a function $u \in \mathcal{H}$ **that we already know**, to introduce the notion of **linear approximation spaces** V_N and **nonlinear approximation spaces** $\mathcal{V}_N$;
2. Then, approximate an unknown function $u \in \mathcal{H}$ **that is the solution to a PDE** $\mathcal{D}[u] = f$, using the aforementioned spaces, leading to a **description of PINNs**.

1.1. Linear approximation spaces

Linear approximation spaces: definition

Definition (Linear approximation space): Let $\Omega \subset \mathbb{R}^d$ be a domain and $\mathcal{H} = L^2(\Omega)$ be a Hilbert space of functions defined on Ω . A linear approximation space V_N of dimension N is defined by

$$V_N = \text{Span}(\varphi_1, \dots, \varphi_N) = \left\{ u_\theta \in \mathcal{H} \text{ such that } \exists \theta \in \Theta, \forall x \in \Omega, u_\theta(x) = \sum_{j=1}^N \theta_j \varphi_j(x) \right\},$$

where $\varphi = (\varphi_j)_{j=1}^N \in \mathcal{H}^N$ are the basis functions of V_N and $\theta = (\theta_j)_{j=1}^N \in \Theta \subset \mathbb{R}^N$ are the parameters to be determined. The parameters θ are called **degrees of freedom** (dofs).

The choice of the basis functions φ completely defines the approximation space V_N . For instance:

- the **finite volume method** uses indicator functions of the mesh cells Ω_j : $\varphi_j = \mathbb{1}_{\Omega_j}$;
- the **finite element method** uses piecewise polynomials, e.g., hat functions $\varphi_j = \max(0, 1 - |x - x_j| / h_j)$.

Linear approximation spaces: definition

Definition (Linear approximation space): Let $\Omega \subset \mathbb{R}^d$ be a domain and $\mathcal{H} = L^2(\Omega)$ be a Hilbert space of functions defined on Ω . A linear approximation space V_N of dimension N is defined by

$$V_N = \text{Span}(\varphi_1, \dots, \varphi_N) = \left\{ u_\theta \in \mathcal{H} \text{ such that } \exists \theta \in \Theta, \forall x \in \Omega, u_\theta(x) = \sum_{j=1}^N \theta_j \varphi_j(x) \right\},$$

where $\varphi = (\varphi_j)_{j=1}^N \in \mathcal{H}^N$ are the basis functions of V_N and $\theta = (\theta_j)_{j=1}^N \in \Theta \subset \mathbb{R}^N$ are the parameters to be determined. The parameters θ are called **degrees of freedom** (dofs).

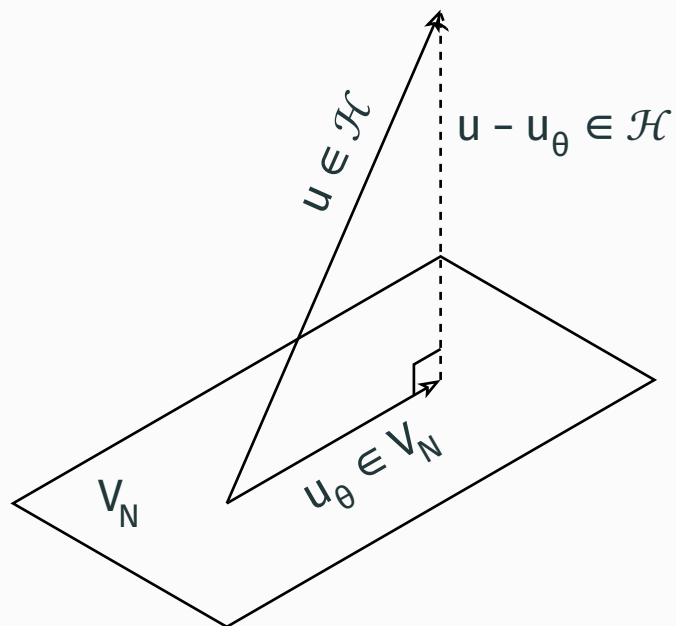
The choice of the basis functions φ completely defines the approximation space V_N . For instance:

- the **finite volume method** uses indicator functions of the mesh cells Ω_j : $\varphi_j = \mathbb{1}_{\Omega_j}$;
- the **finite element method** uses piecewise polynomials, e.g., hat functions $\varphi_j = \max(0, 1 - |x - x_j| / h_j)$.

Goal: Let $u \in \mathcal{H}$. Given a **finite-dimensional subspace** $V_N \subset \mathcal{H}$, we seek to approximate u by a function $u_\theta \in V_N$.

↪ How to find the dofs θ ?

Linear approximation spaces: orthogonal projection (Galerkin method)



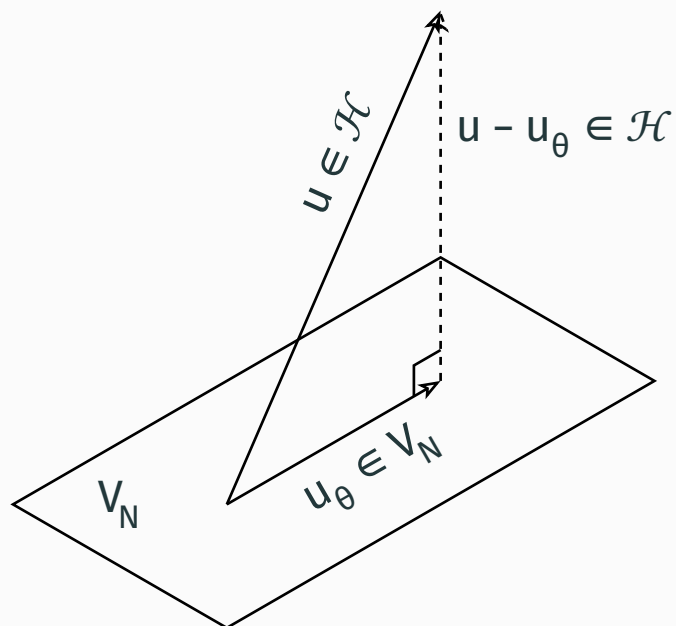
Finding u_θ amounts to determining the dofs θ such that the distance in $\mathcal{H}$ between u and u_θ is minimized:

$$\theta \in \operatorname{argmin}_{\vartheta \in \Theta} \mathcal{J}(\vartheta) = \operatorname{argmin}_{\vartheta \in \Theta} \frac{1}{2} \|u - u_\vartheta\|_{\mathcal{H}}^2$$

$$\theta \in \operatorname{argmin}_{\vartheta \in \Theta} \frac{1}{2} \int_{\Omega} |u(x) - u_\vartheta(x)|^2 dx.$$

We thus seek $u_\theta \in V_N$ such that, for all $v \in V_N$, $\langle u - u_\theta, v \rangle_{\mathcal{H}} = 0$. This amounts to an orthogonal projection of $u \in \mathcal{H}$ onto V_N .

Linear approximation spaces: orthogonal projection (Galerkin method)



Finding u_θ amounts to determining the dofs θ such that the distance in $\mathcal{H}$ between u and u_θ is minimized:

$$\theta \in \operatorname{argmin}_{\vartheta \in \Theta} \mathcal{J}(\vartheta) = \operatorname{argmin}_{\vartheta \in \Theta} \frac{1}{2} \|u - u_\vartheta\|_{\mathcal{H}}^2$$

$$\theta \in \operatorname{argmin}_{\vartheta \in \Theta} \frac{1}{2} \int_{\Omega} |u(x) - u_\vartheta(x)|^2 dx.$$

We thus seek $u_\theta \in V_N$ such that, for all $v \in V_N$, $\langle u - u_\theta, v \rangle_{\mathcal{H}} = 0$. This amounts to an orthogonal projection of $u \in \mathcal{H}$ onto V_N .

This leads to a linear system $\mathcal{M}\theta = U$, where the matrix $\mathcal{M}$ and right-hand side U are given by

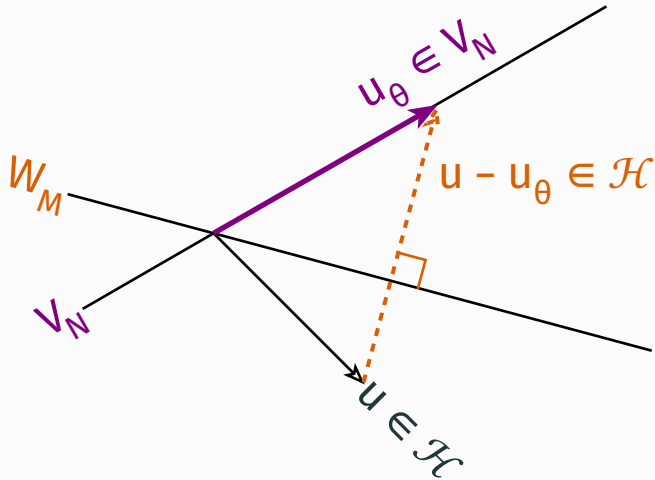
$$\forall i \in \llbracket 1, N \rrbracket, \quad \forall j \in \llbracket 1, N \rrbracket, \quad \mathcal{M}_{ij} = \langle \varphi_i, \varphi_j \rangle_{\mathcal{H}} \quad \text{and} \quad U_i = \langle u, \varphi_i \rangle_{\mathcal{H}}.$$

Linear approximation spaces: oblique projection (Petrov-Galerkin method)

To find $u_\theta \in V_N$, we compute the dofs θ such that the residual is orthogonal to a test space $W_M = \text{Span}(\psi_1, \dots, \psi_M)$:

$$u_\theta \in V_N \quad \text{s.t.} \quad \langle u - u_\theta, \psi_k \rangle_{\mathcal{H}} = 0, \quad \forall k \in \llbracket 1, M \rrbracket.$$

We seek $u_\theta \in V_N$ such that $u - u_\theta$ is orthogonal to W_M .
 If $V_N = W_M$, we recover the **orthogonal projection**.
 If $V_N \neq W_M$, this is an **oblique projection**.

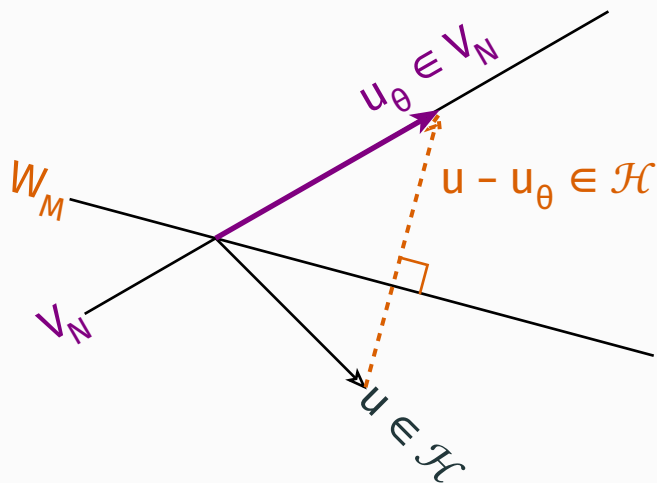


Linear approximation spaces: oblique projection (Petrov-Galerkin method)

To find $u_\theta \in V_N$, we compute the dofs θ such that the residual is orthogonal to a test space $W_M = \text{Span}(\psi_1, \dots, \psi_M)$:

$$u_\theta \in V_N \quad \text{s.t.} \quad \langle u - u_\theta, \psi_k \rangle_{\mathcal{H}} = 0, \quad \forall k \in \llbracket 1, M \rrbracket.$$

We seek $u_\theta \in V_N$ such that $u - u_\theta$ is orthogonal to W_M .
 If $V_N = W_M$, we recover the **orthogonal projection**.
 If $V_N \neq W_M$, this is an **oblique projection**.



Therefore, we seek $\theta = (\theta_j)_{j=1}^N$ such that

$$\forall k \in \llbracket 1, M \rrbracket, \quad \sum_{j=1}^N \theta_j \langle \varphi_j, \psi_k \rangle_{\mathcal{H}} = \langle u, \psi_k \rangle_{\mathcal{H}}.$$

This yields the linear system $K\theta = U$, where:

$$K_{kj} = \langle \varphi_j, \psi_k \rangle_{\mathcal{H}}, \quad U_k = \langle u, \psi_k \rangle_{\mathcal{H}}.$$

If $N \neq M$, we solve this rectangular linear system in a least-squares way:

$$\theta \in \underset{\vartheta \in \Theta}{\operatorname{argmin}} \quad \|K\vartheta - U\|_2^2.$$

Degrees of freedom for collocation methods

An important example of a Petrov-Galerkin method is the **collocation method**. In this case, we introduce a set of **M collocation points** $(x_k)_{k \in \llbracket 1, M \rrbracket}$. They can be sampled uniformly, randomly, ...

Then, the **test space** is made of Dirac deltas at those points: $W_M = \text{Span}(\delta_{x_1}, \dots, \delta_{x_M})$.

Degrees of freedom for collocation methods

An important example of a Petrov-Galerkin method is the **collocation method**. In this case, we introduce a set of **M collocation points** $(x_k)_{k \in \llbracket 1, M \rrbracket}$. They can be sampled uniformly, randomly, ...

Then, the **test space** is made of Dirac deltas at those points: $W_M = \text{Span}(\delta_{x_1}, \dots, \delta_{x_M})$.

With a slight abuse of notation, $\langle u, \delta_x \rangle$ denotes the pointwise evaluation $u(x)$. Recall that we seek

$$u_\theta \in V_N \quad \text{s.t.} \quad \langle u - u_\theta, \psi_k \rangle = 0, \quad \forall k \in \llbracket 1, M \rrbracket.$$

Computing this inner product, we get, for all $k \in \llbracket 1, M \rrbracket$, that u_θ satisfies

$$\langle u - u_\theta, \delta_{x_k} \rangle = \int_{\Omega} (u(x) - u_\theta(x)) \delta_{x_k}(x) dx = u(x_k) - \sum_{j=1}^N \theta_j \varphi_j(x_k) = 0.$$

Degrees of freedom for collocation methods

An important example of a Petrov-Galerkin method is the **collocation method**. In this case, we introduce a set of **M collocation points** $(x_k)_{k \in \llbracket 1, M \rrbracket}$. They can be sampled uniformly, randomly, ...

Then, the **test space** is made of Dirac deltas at those points: $W_M = \text{Span}(\delta_{x_1}, \dots, \delta_{x_M})$.

With a slight abuse of notation, $\langle u, \delta_x \rangle$ denotes the pointwise evaluation $u(x)$. Recall that we seek

$$u_\theta \in V_N \quad \text{s.t.} \quad \langle u - u_\theta, \psi_k \rangle = 0, \quad \forall k \in \llbracket 1, M \rrbracket.$$

Computing this inner product, we get, for all $k \in \llbracket 1, M \rrbracket$, that u_θ satisfies

$$\langle u - u_\theta, \delta_{x_k} \rangle = \int_{\Omega} (u(x) - u_\theta(x)) \delta_{x_k}(x) dx = u(x_k) - \sum_{j=1}^N \theta_j \varphi_j(x_k) = 0.$$

This is not reachable in general, and so we **minimize the residual at the collocation points**:

$$\theta \in \operatorname{argmin}_{\theta \in \Theta} \sum_{k=1}^M \left| u(x_k) - \sum_{j=1}^N \theta_j \varphi_j(x_k) \right|^2.$$

This is nothing but a linear least-squares fit of the function u at the collocation points $(x_k)_{k \in \llbracket 1, M \rrbracket}$!

Advantages and drawbacks of linear methods

Advantage (Convergence): Linear function approximation methods generally satisfy, for some $p > 0$,

$$\|u - u_\theta\|_{\mathcal{H}} = \mathcal{O}\left(N^{-\frac{p}{d}}\right)$$

This is a great convergence result! As the number N of dofs increases, the error quantifiably decreases.

Advantages and drawbacks of linear methods

Advantage (Convergence): Linear function approximation methods generally satisfy, for some $p > 0$,

$$\|u - u_\theta\|_{\mathcal{H}} = \mathcal{O}\left(N^{-\frac{p}{d}}\right)$$

This is a great convergence result! As the number N of dofs increases, the error quantifiably decreases.

Drawback (Curse of dimensionality): To reach a given error target ε , one requires roughly

$$N = \mathcal{O}\left(\varepsilon^{-\frac{d}{p}}\right)$$

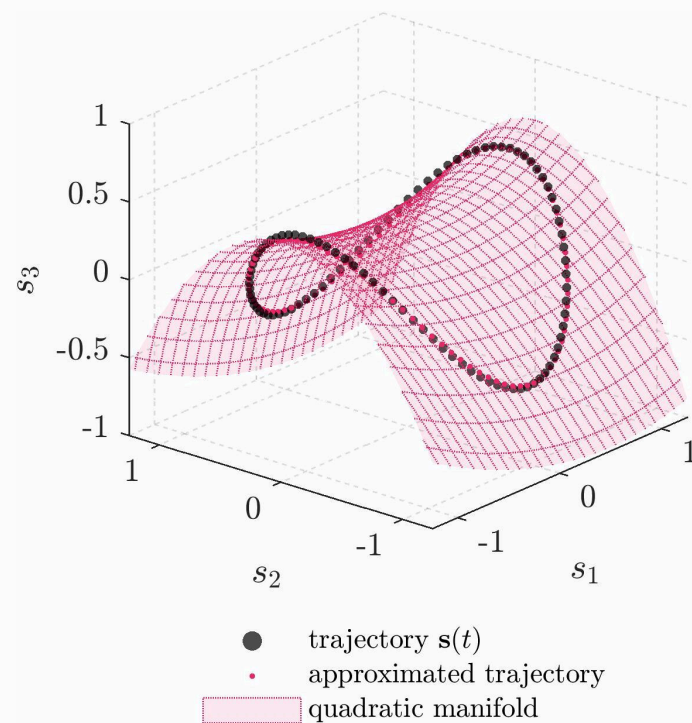
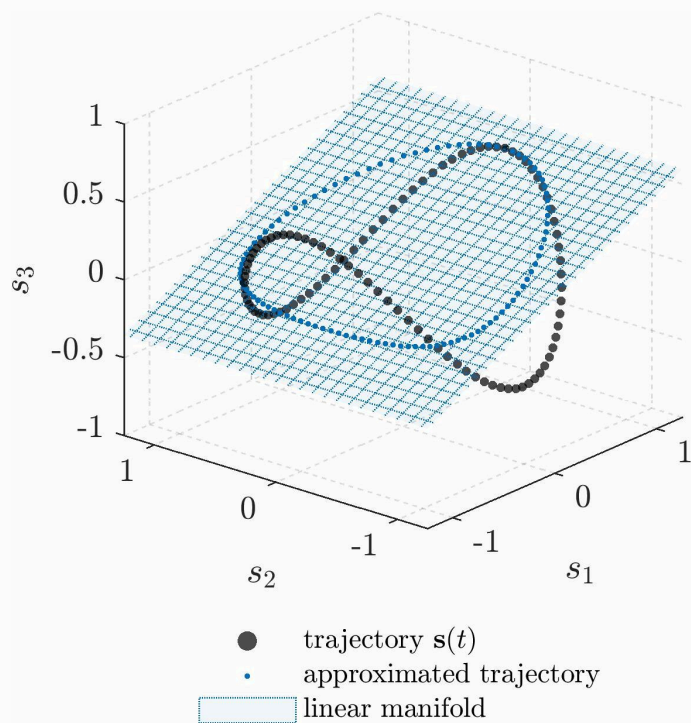
dofs for a d -dimensional domain Ω .

This is a disheartening result, as the number of dofs increases exponentially with the dimension of the domain.

1.2. Nonlinear approximation spaces

Linear vs nonlinear approximation spaces

Nonlinear numerical methods rely on replacing the linear subspace $\mathcal{V}_N \subset \mathcal{H}$ with a nonlinear subspace $\mathcal{V}_N \subset \mathcal{H}$.



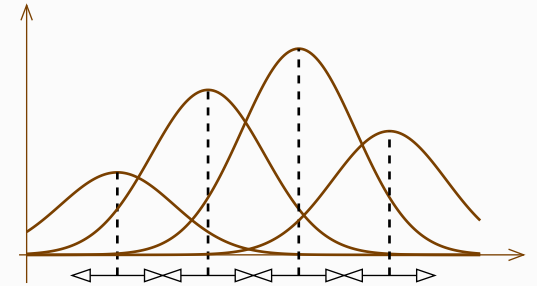
Of course, this only makes sense if the solution u is actually in a low-dimensional manifold $\mathcal{V} \subset \mathcal{H}$.

figures from Geelen et al, 2022

Example: linear vs nonlinear RBFs

For **linear Gaussian RBFs**, the centers $c_j \in \Omega$ and the variance σ^2 of the Gaussian basis functions are fixed:

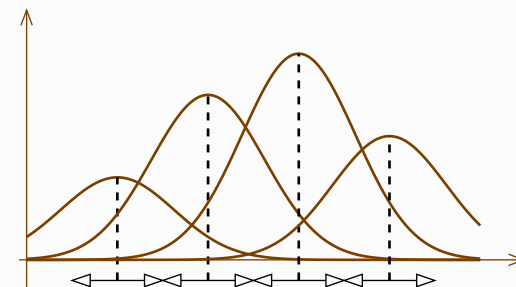
$$u_\theta(x) = \sum_{j=1}^N \theta_j \exp\left(-\frac{|x - c_j|^2}{2\sigma^2}\right).$$



Example: linear vs nonlinear RBFs

For **linear Gaussian RBFs**, the centers $c_j \in \Omega$ and the variance σ^2 of the Gaussian basis functions are fixed:

$$u_{\theta}(x) = \sum_{j=1}^N \theta_j \exp\left(-\frac{|x - c_j|^2}{2\sigma^2}\right).$$

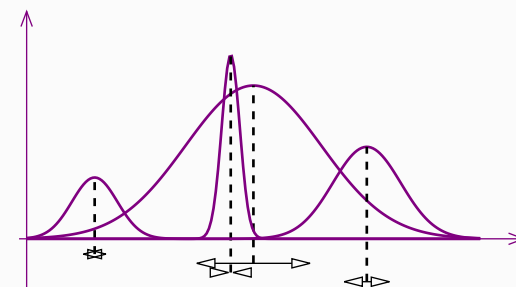


For **nonlinear Gaussian RBFs**, the centers and variances are also dofs:

$$u_{\alpha, c, \sigma}(x) = \sum_{j=1}^{\frac{N}{d+2}} \alpha_j \exp\left(-\frac{|x - c_j|^2}{2\sigma_j^2}\right).$$

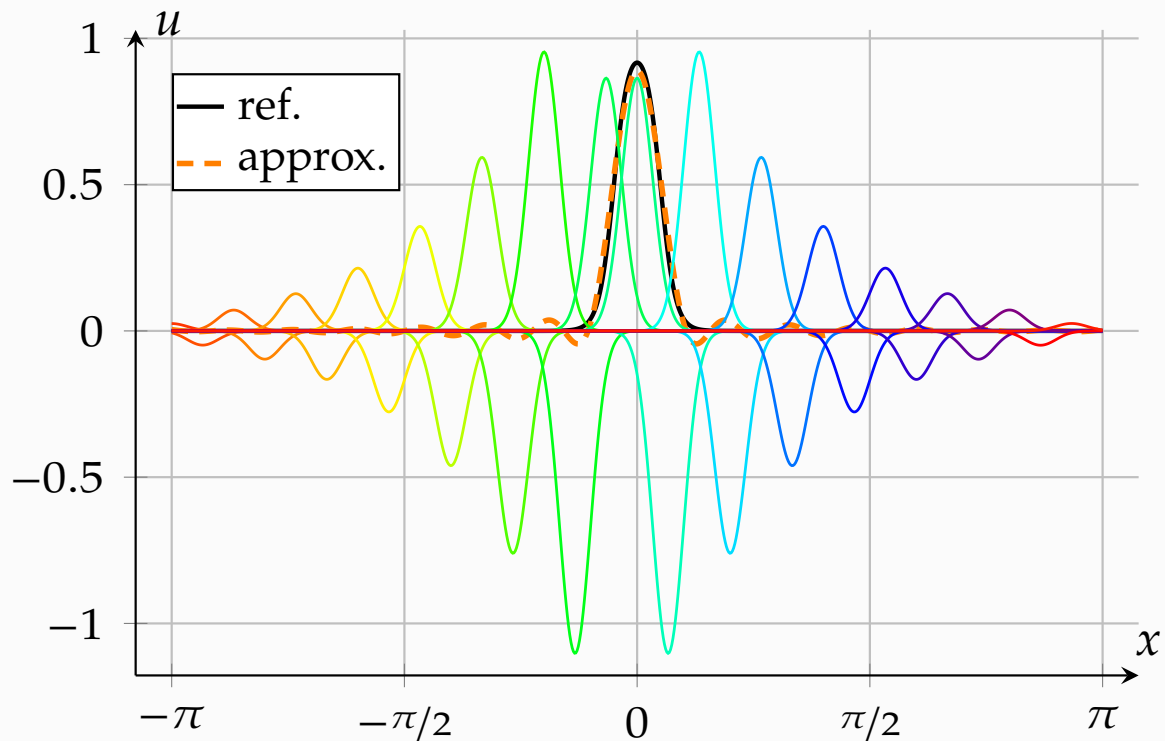
We define a vector $\theta = (\alpha_j, c_j, \sigma_j)_{j=1}^{\frac{N}{d+2}} \in \mathbb{R}^N$ and a nonlinear function φ such that

$$u_{\theta}(x) = \varphi(x; \theta) = \sum_{j=1}^{\frac{N}{d+2}} \alpha_j \exp\left(-\frac{|x - c_j|^2}{2\sigma_j^2}\right).$$

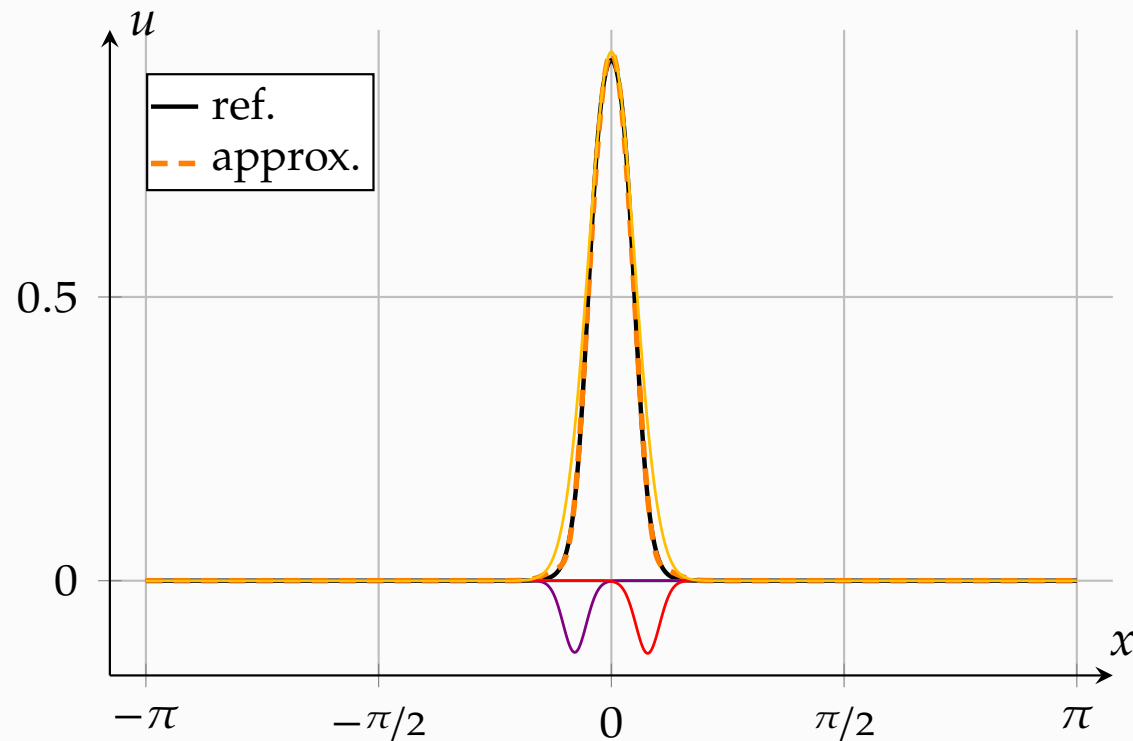


Numerical experiment: approximating a localized function

linear space, 30 dofs



nonlinear space, 9 dofs



Very few nonlinear RBFs yield a better approximation than many more linear RBFs!

Nonlinear approximation spaces

Definition (Nonlinear approximation space): Let $\varphi : \Omega \times \mathbb{R}^N \rightarrow \mathbb{R}$ be a nonlinear function. The nonlinear approximation space $\mathcal{V}_N \subset \mathcal{H}$ of dimension N , based on φ , is defined by

$$\mathcal{V}_N = \text{Im}(\theta \mapsto \varphi(\cdot; \theta)) = \{u_\theta \in \mathcal{H} \text{ such that } \exists \theta \in \Theta, \forall x \in \Omega, u_\theta(x) = \varphi(x; \theta)\},$$

where $\theta = (\theta_j)_{j=1}^N \in \Theta \subset \mathbb{R}^N$ remain the **degrees of freedom** to be determined.

Nonlinear approximation spaces

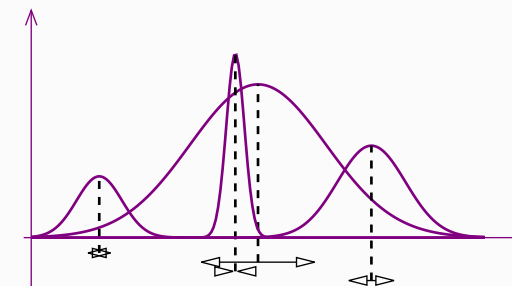
Definition (Nonlinear approximation space): Let $\varphi : \Omega \times \mathbb{R}^N \rightarrow \mathbb{R}$ be a nonlinear function. The nonlinear approximation space $\mathcal{V}_N \subset \mathcal{H}$ of dimension N , based on φ , is defined by

$$\mathcal{V}_N = \text{Im}(\theta \mapsto \varphi(\cdot; \theta)) = \{u_\theta \in \mathcal{H} \text{ such that } \exists \theta \in \Theta, \forall x \in \Omega, u_\theta(x) = \varphi(x; \theta)\},$$

where $\theta = (\theta_j)_{j=1}^N \in \Theta \subset \mathbb{R}^N$ remain the **degrees of freedom** to be determined.

- The choice of the nonlinear function $\varphi : \Omega \times \mathbb{R}^N \rightarrow \mathbb{R}$ completely defines the approximation space $\mathcal{V}_N$.
- Example: for **nonlinear Gaussian Radial Basis Functions (RBFs)**, the nonlinear function reads:

$$\varphi(x; \theta) = \varphi(x; (\alpha, c, \sigma)) = \sum_{j=1}^N \alpha_j \exp\left(-\frac{|x - c_j|^2}{2\sigma_j^2}\right).$$



- Example: for **MLPs**, the nonlinear function is $\varphi(x; \theta) = \mathcal{N}(x; \theta)$, with $\mathcal{N}$ a neural network with parameters θ .

Relationship between linear and nonlinear approximation spaces

Linear spaces read

$$V_N = \left\{ u_\theta \in \mathcal{H} \quad \text{such that} \quad \exists \theta \in \Theta, \forall x \in \Omega, u_\theta(x) = \sum_{j=1}^N \theta_j \varphi_j^{\text{linear}}(x) \right\},$$

while nonlinear spaces may be rewritten, with $N = N_1 + N_2$, as

$$\mathcal{V}_N = \left\{ u_{\alpha,\beta} \in \mathcal{H} \quad \text{such that} \quad \exists \alpha \in \mathbb{R}^{N_1}, \exists \beta \in \mathbb{R}^{N_2}, \forall x \in \Omega, u_{\alpha,\beta}(x) = \sum_{j=1}^{N_1} \alpha_j \varphi_j^{\text{nonlinear}}(x; \beta) \right\}.$$

Hence, they are **not linear spaces** as the functions φ_j are **not a fixed basis**, but rather they **depend on the target function u through the dofs β** .

Degrees of freedom for nonlinear collocation methods

We determine the dofs in a Petrov-Galerkin way, using a **collocation method**. In this case, we introduce a set of **M collocation points** $(x_k)_{k \in \llbracket 1, M \rrbracket}$. They can be sampled uniformly, randomly, ...

Then, the **test space** is made of Dirac deltas at those points: $W_M = \text{Span}(\delta_{x_1}, \dots, \delta_{x_M})$.

Degrees of freedom for nonlinear collocation methods

We determine the dofs in a Petrov-Galerkin way, using a **collocation method**. In this case, we introduce a set of **M collocation points** $(x_k)_{k \in \llbracket 1, M \rrbracket}$. They can be sampled uniformly, randomly, ...

Then, the **test space** is made of Dirac deltas at those points: $W_M = \text{Span}(\delta_{x_1}, \dots, \delta_{x_M})$.

Recall that we seek

$$u_\theta \in \mathcal{V}_N \quad \text{s.t.} \quad \langle u - u_\theta, \psi_k \rangle = 0, \quad \forall k \in \llbracket 1, M \rrbracket.$$

Computing this inner product, we get, for all $k \in \llbracket 1, M \rrbracket$, that u_θ satisfies

$$\langle u - u_\theta, \delta_{x_k} \rangle = \int_{\Omega} (u(x) - u_\theta(x)) \delta_{x_k}(x) dx = u(x_k) - \varphi(x_k; \theta) = 0.$$

Degrees of freedom for nonlinear collocation methods

We determine the dofs in a Petrov-Galerkin way, using a **collocation method**. In this case, we introduce a set of **M collocation points** $(x_k)_{k \in \llbracket 1, M \rrbracket}$. They can be sampled uniformly, randomly, ...

Then, the **test space** is made of Dirac deltas at those points: $W_M = \text{Span}(\delta_{x_1}, \dots, \delta_{x_M})$.

Recall that we seek

$$u_\theta \in \mathcal{V}_N \quad \text{s.t.} \quad \langle u - u_\theta, \psi_k \rangle = 0, \quad \forall k \in \llbracket 1, M \rrbracket.$$

Computing this inner product, we get, for all $k \in \llbracket 1, M \rrbracket$, that u_θ satisfies

$$\langle u - u_\theta, \delta_{x_k} \rangle = \int_{\Omega} (u(x) - u_\theta(x)) \delta_{x_k}(x) dx = u(x_k) - \varphi(x_k; \theta) = 0.$$

This is not reachable in general, and so we **minimize the residual at the collocation points**:

$$\theta \in \underset{\theta \in \Theta}{\operatorname{argmin}} \sum_{k=1}^M |u(x_k) - \varphi(x_k; \theta)|^2.$$

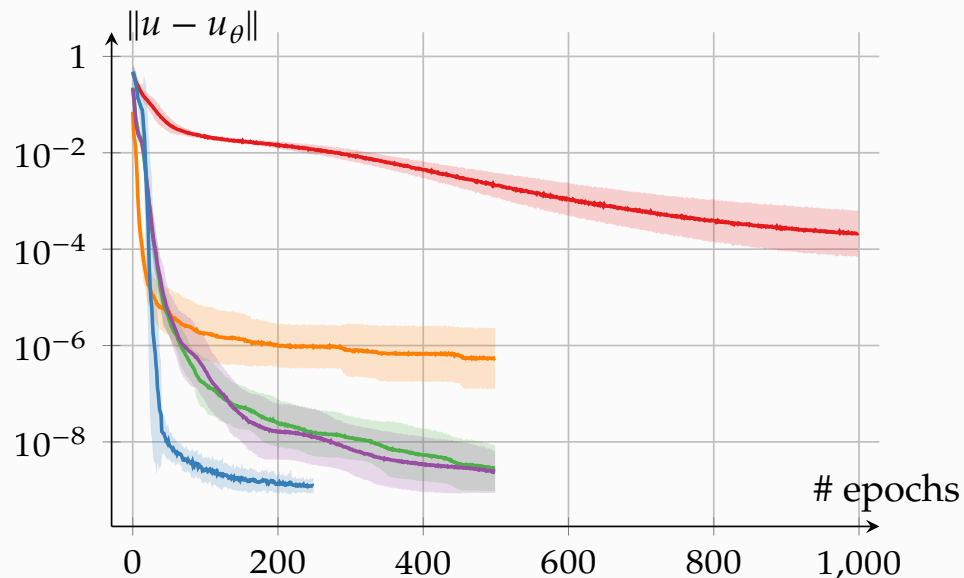
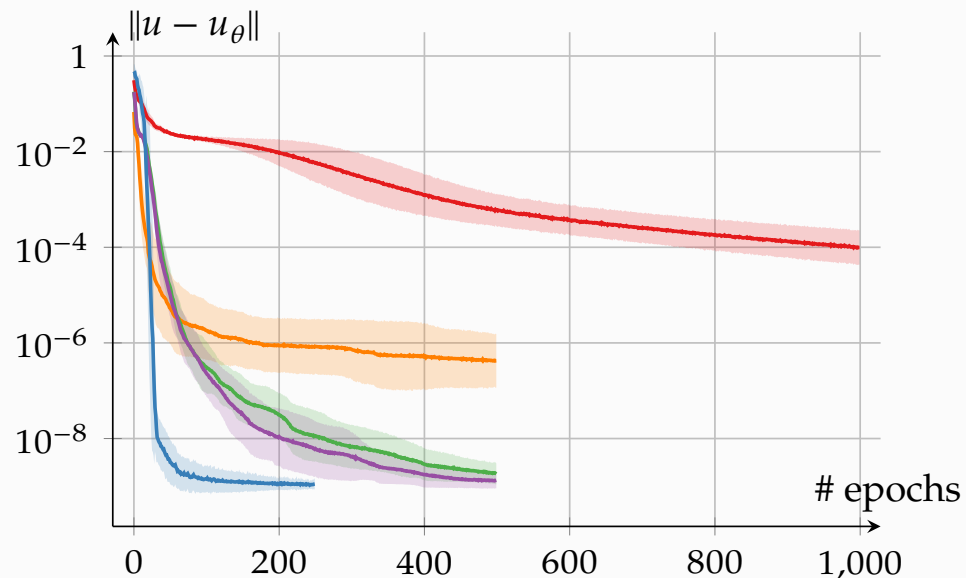
This is nothing but a nonlinear least-squares fit of the function u at the collocation points $(x_k)_{k \in \llbracket 1, M \rrbracket}$!

Determining the degrees of freedom

This collocation method has the following advantages and drawbacks:

- 😊 **No mesh is required**; we need to “sample” the domain to obtain the collocation points.
 - 😊 **High dimensions** become reachable, even with a low number of dofs.
 - 🤖 We obtain a **non-quadratic and non-convex minimization problem** in θ ! This does not lead to a linear system.
- ~> We require a **very good optimization algorithm** to solve this problem.

Comparison of nonlinear optimizers (for a small number N of dofs)

(a) approximation with 60 dofs(b) approximation with 192 dofs

— Adam — LBFGS — ssBFGS — ssBroyden — ENG + SGD

Conclusion: Even on a simple problem, optimization efficiency is crucial for obtaining good accuracy.

2. Solving PDEs in nonlinear approximation spaces

Plan of the section

In this section, we will tackle the approximation of PDE solutions in linear and nonlinear spaces.

1. First, we tackle **elliptic PDEs**, as they already contain most of the required concepts to understand the main ideas of the method.
2. Then, we consider **parameter-dependent PDEs**, which depend on some physical parameters. The parameters are taken into account in a **monolithic** way, i.e., they appear as additional dimensions of the problem.
3. Finally, we move on to **time-dependent PDEs**, and distinguish between **monolithic** and **time-discrete** approaches.

2.1. Elliptic problems

Linear methods for elliptic problems

Let $\mathcal{D}$ be a differential operator. We consider the **elliptic problem** (disregarding boundary conditions):

$$\forall x \in \Omega, \quad \mathcal{D}[u](x) = f(x).$$

Similarly to the function approximation task, we define a **linear approximation space** V_N :

$$V_N = \left\{ f_\theta := \mathcal{D}[u_\theta] \in \mathcal{H} \quad \text{such that} \quad \exists \theta \in \Theta, \forall x \in \Omega, u_\theta(x) = \sum_{j=1}^N \theta_j \varphi_j(x) \right\}.$$

Linear methods for elliptic problems

Let $\mathcal{D}$ be a differential operator. We consider the **elliptic problem** (disregarding boundary conditions):

$$\forall x \in \Omega, \quad \mathcal{D}[u](x) = f(x).$$

Similarly to the function approximation task, we define a **linear approximation space** V_N :

$$V_N = \left\{ f_\theta := \mathcal{D}[u_\theta] \in \mathcal{H} \quad \text{such that} \quad \exists \theta \in \Theta, \forall x \in \Omega, u_\theta(x) = \sum_{j=1}^N \theta_j \varphi_j(x) \right\}.$$

We then solve the **oblique projection**:

$$\text{find } f_\theta \in V_N \quad \text{s.t.} \quad \forall w \in W_M, \quad \langle f - f_\theta, w \rangle_{\mathcal{H}} = 0,$$

$$\text{or equivalently, find } \theta \in \Theta \quad \text{s.t.} \quad \forall w \in W_M, \quad \langle f - \mathcal{D}[u_\theta], w \rangle_{\mathcal{H}} = 0.$$

Linear methods for elliptic problems

Let $\mathcal{D}$ be a differential operator. We consider the **elliptic problem** (disregarding boundary conditions):

$$\forall x \in \Omega, \quad \mathcal{D}[u](x) = f(x).$$

Similarly to the function approximation task, we define a **linear approximation space** V_N :

$$V_N = \left\{ f_\theta := \mathcal{D}[u_\theta] \in \mathcal{H} \quad \text{such that} \quad \exists \theta \in \Theta, \forall x \in \Omega, u_\theta(x) = \sum_{j=1}^N \theta_j \varphi_j(x) \right\}.$$

We then solve the **oblique projection**:

$$\text{find } f_\theta \in V_N \quad \text{s.t.} \quad \forall w \in W_M, \quad \langle f - f_\theta, w \rangle_{\mathcal{H}} = 0,$$

$$\text{or equivalently, find } \theta \in \Theta \quad \text{s.t.} \quad \forall w \in W_M, \quad \langle f - \mathcal{D}[u_\theta], w \rangle_{\mathcal{H}} = 0.$$

If $W_M = \text{Span}(\varphi_1, \dots, \varphi_N)$, we get the **Galerkin method**:

$$\forall i \in \llbracket 1, N \rrbracket, \quad \left\langle \mathcal{D}\left[\sum_{j=1}^N \theta_j \varphi_j\right], \varphi_i \right\rangle_{\mathcal{H}} = \langle f, \varphi_i \rangle_{\mathcal{H}}.$$

If $W_M = \text{Span}(\delta_{x_1}, \dots, \delta_{x_M})$, we get the **collocation method**:

$$\theta \in \operatorname{argmin}_{\theta \in \Theta} \sum_{k=1}^M \left| \mathcal{D}\left[\sum_{j=1}^N \theta_j \varphi_j\right](x_k) - f(x_k) \right|^2.$$

Nonlinear methods for elliptic problems

Like in the linear case, we define a **nonlinear approximation space** $\mathcal{V}_N$:

$$\mathcal{V}_N = \{f_\theta := \mathcal{D}[u_\theta] \in \mathcal{H} \quad \text{such that} \quad \exists \theta \in \Theta, \forall x \in \Omega, u_\theta(x) = \varphi(x; \theta)\}.$$

We use a **collocation method**, with **M collocation points** $(x_k)_{k \in \llbracket 1, M \rrbracket}$ and the **test space** $W_M = \text{Span}(\delta_{x_1}, \dots, \delta_{x_M})$.

Nonlinear methods for elliptic problems

Like in the linear case, we define a **nonlinear approximation space** $\mathcal{V}_N$:

$$\mathcal{V}_N = \{f_\theta := \mathcal{D}[u_\theta] \in \mathcal{H} \text{ such that } \exists \theta \in \Theta, \forall x \in \Omega, u_\theta(x) = \varphi(x; \theta)\}.$$

We use a **collocation method**, with **M collocation points** $(x_k)_{k \in \llbracket 1, M \rrbracket}$ and the **test space** $W_M = \text{Span}(\delta_{x_1}, \dots, \delta_{x_M})$.

Recall that we seek

$$f_\theta \in \mathcal{V}_N \text{ s.t. } \langle f - f_\theta, \psi_k \rangle = 0, \quad \forall k \in \llbracket 1, M \rrbracket.$$

Computing this inner product, we get, for all $k \in \llbracket 1, M \rrbracket$, that u_θ satisfies

$$\langle f - f_\theta, \delta_{x_k} \rangle = \langle f - \mathcal{D}[u_\theta], \delta_{x_k} \rangle = \int_{\Omega} (f(x) - \mathcal{D}[u_\theta](x)) \delta_{x_k}(x) dx = f(x_k) - \mathcal{D}[u_\theta](x_k) = 0.$$

Nonlinear methods for elliptic problems

Like in the linear case, we define a **nonlinear approximation space** $\mathcal{V}_N$:

$$\mathcal{V}_N = \{f_\theta := \mathcal{D}[u_\theta] \in \mathcal{H} \text{ such that } \exists \theta \in \Theta, \forall x \in \Omega, u_\theta(x) = \varphi(x; \theta)\}.$$

We use a **collocation method**, with **M collocation points** $(x_k)_{k \in \llbracket 1, M \rrbracket}$ and the **test space** $W_M = \text{Span}(\delta_{x_1}, \dots, \delta_{x_M})$.

Recall that we seek

$$f_\theta \in \mathcal{V}_N \text{ s.t. } \langle f - f_\theta, \psi_k \rangle = 0, \quad \forall k \in \llbracket 1, M \rrbracket.$$

Computing this inner product, we get, for all $k \in \llbracket 1, M \rrbracket$, that u_θ satisfies

$$\langle f - f_\theta, \delta_{x_k} \rangle = \langle f - \mathcal{D}[u_\theta], \delta_{x_k} \rangle = \int_{\Omega} (f(x) - \mathcal{D}[u_\theta](x)) \delta_{x_k}(x) dx = f(x_k) - \mathcal{D}[u_\theta](x_k) = 0.$$

This is not reachable in general, and so we **minimize the residual at the collocation points**:

$$\theta \in \underset{\vartheta \in \Theta}{\operatorname{argmin}} \sum_{k=1}^M |f(x_k) - \mathcal{D}[u_\vartheta](x_k)|^2.$$

We recover **physics-informed neural networks** (PINNs, [Raissi et al., 2019])!

Boundary conditions: multi-objective loss functions

In general, the PDE comes with a set of boundary conditions, and so the BVP reads

$$\begin{cases} \mathcal{D}[u] = f & \text{in } \Omega, \\ \mathcal{B}[u] = 0 & \text{on } \partial\Omega. \end{cases}$$

So the minimization problem has to include both objectives:

$$\mathcal{D}[u_\theta] \approx f \quad \text{and} \quad \mathcal{B}[u_\theta] \approx 0.$$

The main way of handling these multi-objective problems is to use a weighted sum of the objectives, i.e.,

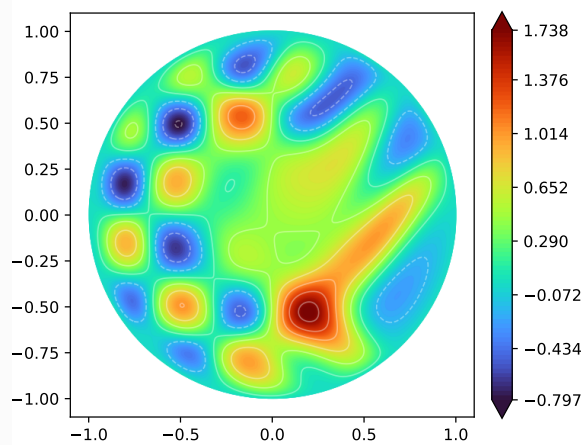
$$\theta \in \operatorname{argmin}_{\theta \in \Theta} \left\{ \sum_{k=1}^M |\mathcal{D}[u_\theta](x_k) - f(x_k)|^2 + \lambda \sum_{k=1}^{M_b} |\mathcal{B}[u_\theta](x_k^b)|^2 \right\},$$

with a **well-chosen coefficient** λ to balance the two objectives.

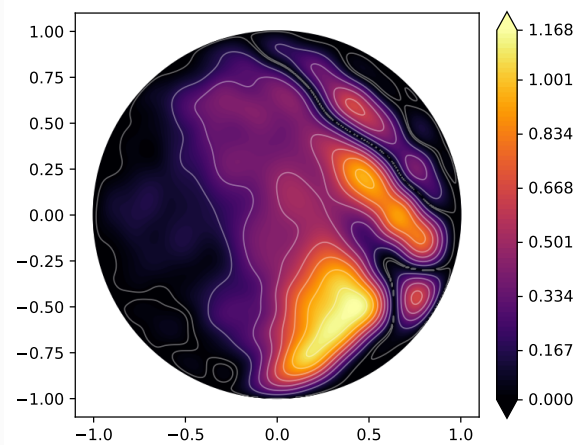
↪ This quickly becomes **very hard to tune for complex, multi-domain and multi-BC PDEs!**

Numerical experiment: Poisson problem in a disk

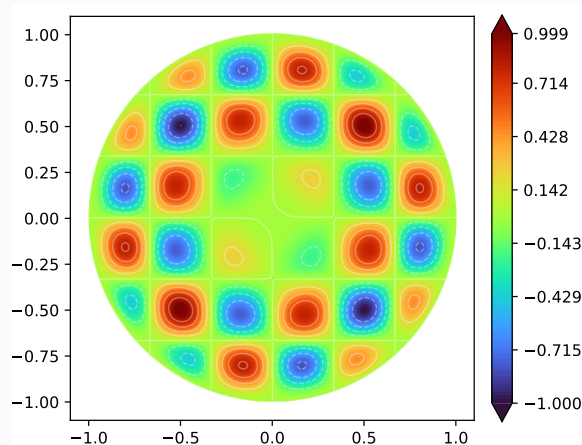
(a) approximation, Adam optimizer



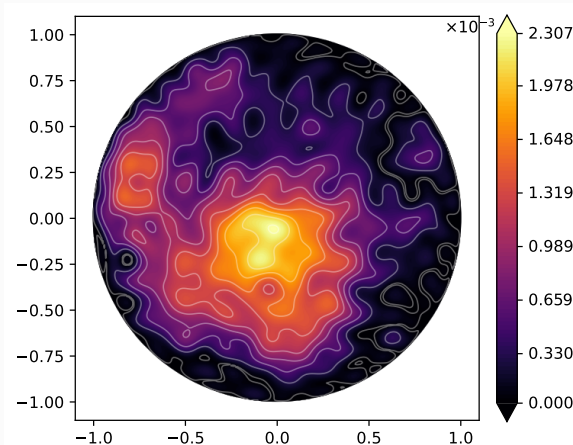
(b) error, Adam optimizer



(c) approximation, natural gradient optimizer



(d) error, natural gradient optimizer



Approximation of the solution to the Poisson problem

$$\begin{cases} -\Delta u = f, & \text{in } \Omega \\ u = 0, & \text{on } \partial\Omega, \end{cases}$$

where Ω is the unit disk.

We use an MLP with 360 dofs; this is not a lot for such an oscillatory solution.

Top: using a first-order optimizer

Bottom: using a second-order optimizer

2.2. Parameter-dependent elliptic problems

Parameter-dependent PDEs

We can also consider parameter-dependent PDEs, where the solution depends on some parameters $\mu \in \mathbb{M} \subset \mathbb{R}^p$:

$$\forall x \in \Omega, \quad \forall \mu \in \mathbb{M}, \quad \mathcal{D}[u](x, \mu) = f(x, \mu).$$

We emphasize that **u is treated as a function of the joint variable $(x, \mu) \in \Omega \times \mathbb{M}$** . The current approach is **monolithic**: the parameters are seen as additional coordinates in the input space.

Parameter-dependent PDEs

We can also consider parameter-dependent PDEs, where the solution depends on some parameters $\mu \in \mathbb{M} \subset \mathbb{R}^p$:

$$\forall x \in \Omega, \quad \forall \mu \in \mathbb{M}, \quad \mathcal{D}[u](x, \mu) = f(x, \mu).$$

We emphasize that **u is treated as a function of the joint variable $(x, \mu) \in \Omega \times \mathbb{M}$** . The current approach is **monolithic**: the parameters are seen as additional coordinates in the input space.

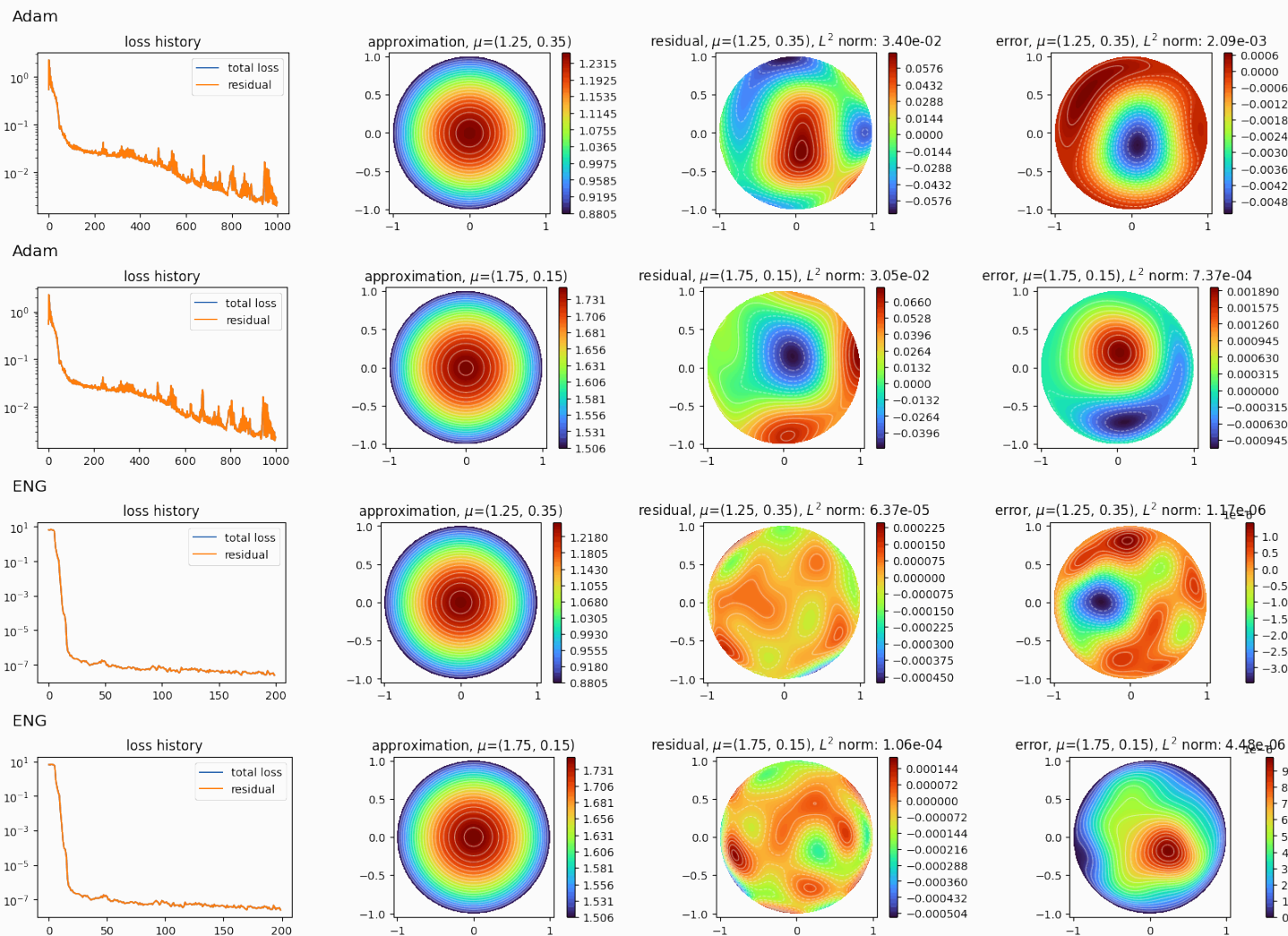
We can then define a parametric nonlinear approximation space

$$\mathcal{V}_N = \{f_\theta := \mathcal{D}[u_\theta] \in L^2(\Omega \times \mathbb{M}) \quad \text{such that} \quad \exists \theta \in \Theta, \forall x \in \Omega, \forall \mu \in \mathbb{M}, u_\theta(x, \mu) = \varphi(x, \mu; \theta)\}.$$

The projection is treated in the same way as before, but now the collocation points are defined in $\Omega \times \mathbb{M}$:

$$\theta \in \operatorname{argmin}_{\theta \in \Theta} \sum_{k=1}^M |\mathcal{D}[u_\theta](x_k, \mu_k) - f(x_k, \mu_k)|^2.$$

Result on a parametric Poisson problem in a disk



- parametric Poisson problem:

$$-\mu_1 \Delta u(x, y, \mu_1, \mu_2) = f(x, y, \mu_1, \mu_2)$$

2D (space) + 2D (parameters) = 4D problem, solved with 960 dofs!
- At the end of the optimization, we obtain a vector $\theta \in \Theta$, and so a function $u_\theta : \Omega \times \mathbb{M} \rightarrow \mathbb{R}$. We can evaluate this function for any $x \in \Omega$ and $\mu \in \mathbb{M}$.
- Once again, it is essential to have a good optimization strategy to obtain good accuracy.

2.3. Time-dependent problems

Time-dependent PDEs: monolithic PINNs

Moving on to a time-dependent PDE of the form

$$\partial_t u(t, x) + \mathcal{D}[u](t, x) = f(t, x), \quad \text{for } t \in [0, T] \text{ and } x \in \Omega,$$

PINNs [Raissi et al., 2019] still approximate the solution u by a neural network

$$u_\theta : (t, x) \mapsto \varphi(t, x; \theta),$$

whose dofs (trainable parameters) θ are global constants in time and space.

Time-dependent PDEs: monolithic PINNs

Moving on to a time-dependent PDE of the form

$$\partial_t u(t, x) + \mathcal{D}[u](t, x) = f(t, x), \quad \text{for } t \in [0, T] \text{ and } x \in \Omega,$$

PINNs [Raissi et al., 2019] still approximate the solution u by a neural network

$$u_\theta : (t, x) \mapsto \varphi(t, x; \theta),$$

whose **dofs (trainable parameters)** θ are **global constants** in time and space.

So, we define a nonlinear approximation space

$$\mathcal{V}_N = \{f_\theta := \partial_t u_\theta + \mathcal{D}[u_\theta] \in L^2([0, T] \times \Omega) \text{ such that } \exists \theta \in \Theta, \forall t \in [0, T], \forall x \in \Omega, u_\theta(t, x) = \varphi(t, x; \theta)\}.$$

The collocation points are now defined in $[0, T] \times \Omega$:

$$\theta \in \operatorname{argmin}_{\theta \in \Theta} \sum_{k=1}^M |\partial_t u_\theta(t_k, x_k) + \mathcal{D}[u_\theta](t_k, x_k) - f(t_k, x_k)|^2.$$

This leads to a **global space-time optimization problem**, which may be difficult to solve in practice.

Time-dependent PDEs: time-discrete PINNs

In the time-discrete case, the solution u is approximated by a neural network whose **dofs** θ **evolve in time**:

$$u_{\theta(t)} : x \mapsto \varphi(x; \theta(t)).$$

For each time $t \in [0, T]$, the dofs $\theta(t)$ should minimize the PDE residual

$$\int_{\Omega} \left| \partial_t u_{\theta(t)}(x) + \mathcal{D}[u_{\theta(t)}](x) - f(t, x) \right|^2 dx.$$

Time-dependent PDEs: time-discrete PINNs

In the time-discrete case, the solution u is approximated by a neural network whose **dofs** θ **evolve in time**:

$$u_{\theta(t)} : x \mapsto \varphi(x; \theta(t)).$$

For each time $t \in [0, T]$, the dofs $\theta(t)$ should minimize the PDE residual

$$\int_{\Omega} \left| \partial_t u_{\theta(t)}(x) + \mathcal{D}[u_{\theta(t)}](x) - f(t, x) \right|^2 dx.$$

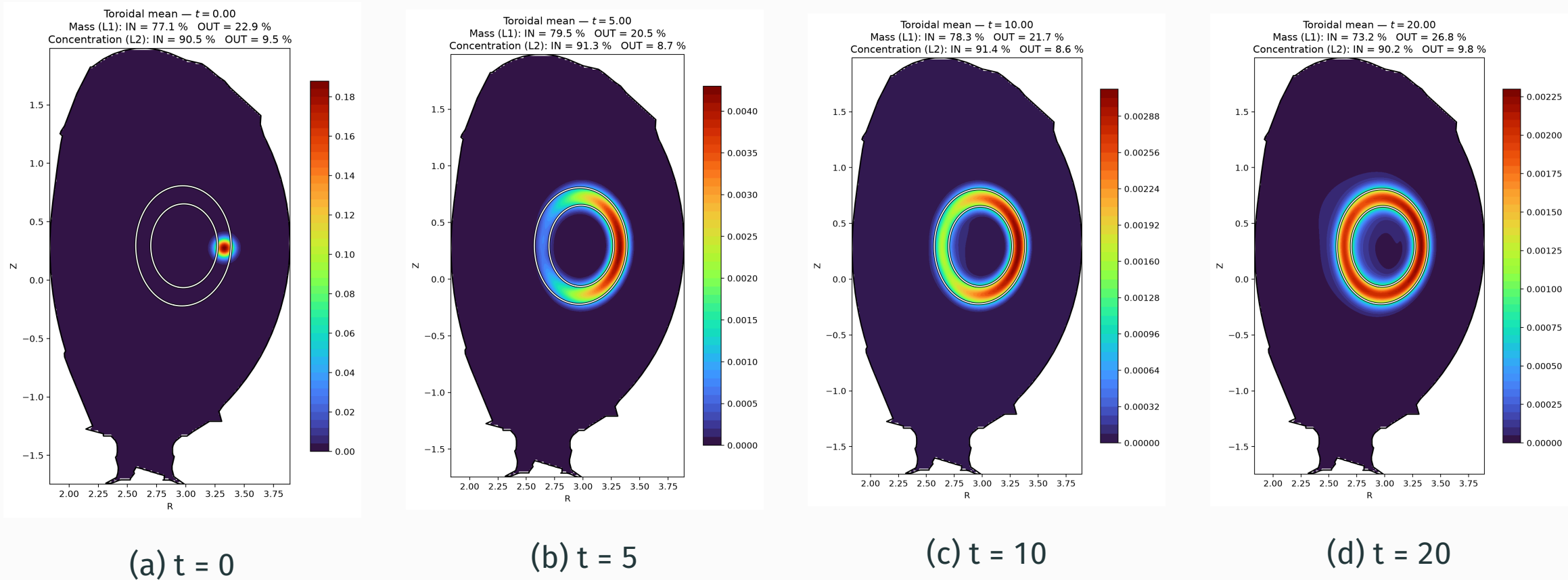
In **Discrete PINNs** [Stiasny et al., 2023], we first **discretize in time** and then **optimize the discrete system**.

We discretize the time domain $[0, T]$ as $(t^n)_n$, and we **seek the dofs** $\theta(t^{n+1})$ **at time** t^{n+1} **from those at time** t^n . Then, we discretize the residual in time (e.g., explicit Euler) and **solve an optimization problem** for $\theta(t^{n+1})$:

$$\theta(t^{n+1}) \in \operatorname{argmin}_{\theta \in \Theta} \int_{\Omega} \left| u_{\theta}(x) - u_{\theta(t^n)}(x) + \Delta t \mathcal{D}[u_{\theta(t^n)}](x) - f(t^n, x) \right|^2 dx.$$

The PDE is thus reduced to a **sequence of optimizations**.

Example: Anisotropic diffusion in the JET tokamak



Results by N. Pailliez: Anisotropic diffusion in the JET tokamak, solved with a time-discrete PINN. The graphs represent the toroidal mean of the full 3D solution (i.e., averaged over the toroidal angle).

2.4. Conclusion

Conclusion

Advantages:

- Very few dofs are required to reach good accuracy, even in high dimensions.
- Complex domains may be easier to deal with (no meshing required).

Drawbacks:

- The cost per dof is generally much higher than with linear methods.
- No good convergence results are available. In practice, adding dofs does not necessarily reduce the error.

Conclusion

Advantages:

- Very few dofs are required to reach good accuracy, even in high dimensions.
- Complex domains may be easier to deal with (no meshing required).

Drawbacks:

- The cost per dof is generally much higher than with linear methods.
- No good convergence results are available. In practice, adding dofs does not necessarily reduce the error.

Our team's main objectives are to improve:

- **the accuracy of nonlinear methods**, and
- **the error constant of linear methods** by hybridizing the two.

This is being implemented in **Scimba**,
a Python + PyTorch + JAX library
with SciML and classical methods for PDEs.

More information is available at:

<https://www.scimba.org>.



Thank you for your attention!