Natural Gradients and Kernel Methods for PINNs

HACE Workshop — HPC/AI Hybridization, Toulouse

Nilo Schwencke

LIP – ENS Lyon • INRIA Lyon • CNRS

Formerly (PhD) : LISN – Paris-Saclay University • INRIA Saclay • CNRS

May 26, 2026

Outline

Setting the stage

PDE problem

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

PINNs loss

$$\mathcal{L}(u) := \int_{\Omega} \|D[u] - f\|^2 + \int_{\partial\Omega} \|B[u] - g\|^2$$

PINNs key idea (Lagaris et al.1998; Raissi et al.2019)

- Model u with a neural network u_θ
- Use autodiff to compute D and B
- Minimize the sampled loss:

$$\widehat{\ell}_{D,B}(\theta) := \frac{1}{2S_D} \sum_{i=1}^{S_D} \left(D[u_\theta](x_i^D) - f(x_i^D) \right)^2 + \frac{1}{2S_B} \sum_{i=1}^{S_B} \left(B[u_\theta](x_i^B) - g(x_i^B) \right)^2 .$$

The core difficulty

PINNs: key idea

Optimization is hard.

Usual optimizers (Adam, L-BFGS) often stagnate far from the true solution. Why?

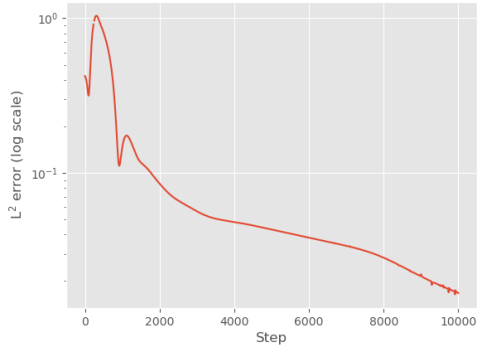


Figure: L^2-error of a PINN optimized with Adam on the 2D Laplace equation.

Three root causes

- **Ill-conditioned loss**: gradient imbalance between PDE and boundary terms
- **Spectral bias**: low frequencies learned first; high-frequency or localized features missed
- **No preconditioner**: standard optimizers are agnostic to the geometry of the problem

Intuition from Fourier

$$S_N : (\alpha_k) \in \mathbb{C}^{[\![-N,N]\!]} \mapsto \sum_{k=-N}^{N} \alpha_k e^{2i\pi kx}.$$

S_N singular values are all 1. **BUT:**

$\Delta[S_N]$ spectrum is $\{4\pi^2 k^2 : 1 \leqslant k \leqslant N\}$

Δ strongly impact the spectral conditionning.

The optimization problem

Our approach

Interpret PINNs optimization through the lens of **functional geometry** and **kernel methods**.

Parametric model

$$u : \begin{cases} \mathbb{R}^P & \to & \mathcal{H} \\ \theta & \mapsto & u_\theta \end{cases} ; \mathcal{H} \text{ Hilbert space}$$

- $\mathcal{M} := \operatorname{Im} u = \{u_\theta \,:\, \theta \in \mathbb{R}^P\}$
- $T_\theta \mathcal{M} := \operatorname{Im} \mathrm{d}u_\theta = \operatorname{Span}(\partial_p u_\theta)$

This naturally leads to the **natural gradient**.

Natural gradient for PINNs

Classical quadratic regression problem, with batch (x_i):

$$\widehat{\ell}(u) := \frac{1}{2S} \sum_{i=1}^{S} \left(u(x_i) - f(x_i)\right)^2 .$$

In the population limit:

$$\widehat{\ell}(\theta) \overset{S\to\infty}{\to} \mathcal{L}(u_\theta) ; \quad \mathcal{L}(u) := \frac{1}{2} \|u - f\|^2_{\mathrm{L}^2(\Omega)}$$

This yields the Fréchet derivative:

$$\mathrm{d}\mathcal{L}|_u(h) = \langle \underbrace{u - f}_{\nabla \mathcal{L}_{|u}}, h \rangle_{\mathrm{L}^2(\Omega)},$$

and thus the gradient flow:

$$\begin{cases} u_0 \in \mathrm{L}^2(\Omega) \\ \dot{u}_t = -\nabla \mathcal{L}_{|u_t} = f - u_t \end{cases} .$$

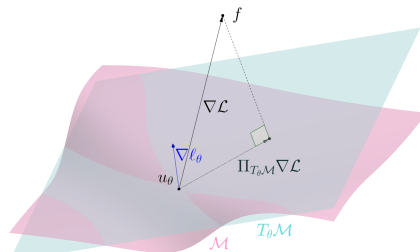
Solution: $u_t = f - e^{-t}(f - u_0)$.

Natural gradient in functional space

But u_θ lives on the manifold $\mathcal{M}$.
$\Rightarrow$ Project the functional gradient onto the tangent space $T_\theta \mathcal{M}$.

The Natural Gradient is then (Amari and Douglas1998):

$$\theta_{t+1} \leftarrow \theta_t - \eta \, \mathrm{d}u^\dagger_{\theta_t} \left(\Pi^{\perp}_{T_{\theta_t}\mathcal{M}} \nabla \mathcal{L}_{|u_{\theta_t}} \right),$$



Key remark

The only difference between the losses:

$$\widehat{\ell}_{D,B}(\theta) := \frac{1}{2S_D}\sum_{i=1}^{S_D}\left(D[u_\theta](x_i^D) - f(x_i^D)\right)^2 + \frac{1}{2S_B}\sum_{i=1}^{S_B}\left(B[u_\theta](x_i^B) - g(x_i^B)\right)^2,$$

and $\widehat{\ell}(u) := \frac{1}{2S}\sum_{i=1}^{S}\left(u(x_i) - f(x_i)\right)^2$ is the use of the operators D and B.

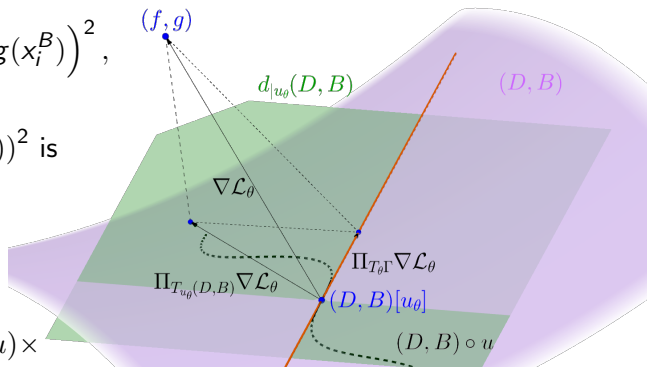
Proposition

PINNs are a quadradic regression problem with model: $(D, B) \circ u$:

$$\begin{cases} \mathbb{R}^P & \to & \mathcal{H} & \to & L^2(\Omega \to \mathbb{R}, \mu) \times \\ & & & & L^2(\partial\Omega \to \mathbb{R}, \sigma) \\ \theta & \mapsto & u_\theta & \mapsto & (D[u_\theta], B[u_\theta]) \end{cases}$$

Natural Gradient of PINNs

Figure: Illustration of PINNs Natural Gradient



Theorem

Let $D : \mathcal{H} \to L^2(\Omega \to \mathbb{R}, \mu)$ be a linear differential operator and $u : \mathbb{R}^P \to \mathcal{H}$ a parametric model. Then $for\ all\ \theta \in \mathbb{R}^P$, the generalized Green's function of D on $T_\theta\mathcal{M} = \mathrm{Im}\, du_\theta$ is given by: $for\ all\ x, y \in \Omega$

$$g_{T_\theta\mathcal{M}}(x, y) := \sum_{1 \leqslant p,q \leqslant P} \partial_p u_\theta(x)\, G^\dagger_{p,q} \partial_q D[u_\theta](y),$$

with: $for\ all\ 1 \leqslant p, q \leqslant P$

$$G_{pq} := \langle \partial_p D[u_\theta]\,,\, \partial_q D[u_\theta] \rangle_{L^2(\Omega \to \mathbb{R}, \mu)}\,.$$

In particular, the natural gradient of PINNs can be rewritten:

$$\theta_{t+1} \leftarrow \theta_t - \eta\, du^\dagger_{\theta_t} \left(x \in \Omega \mapsto \int_\Omega g_{T_{\theta_t}\mathcal{M}}(x, y) \nabla \mathcal{L}_{\theta_t}(y) \mu(dy) \right).$$

Comparison with Galerkin methods

Galerkin / FEM

Fix a finite-dimensional space $\mathcal{H}_n = \mathrm{Span}(\varphi_i)_{i=1}^n$.

- **Basis functions**: φ_i
- **Stiffness matrix**: $K_{ij} = \langle D\varphi_i \,,\, D\varphi_j \rangle_{\mathrm{L}^2}$
- **Right-hand side**: $b_i = \langle \varphi_i \,,\, f \rangle_{\mathrm{L}^2}$
- **Solve**: $K\alpha = b$

Natural gradient of PINNs

At each step, $T_\theta \mathcal{M} = \mathrm{Span}(\partial_p u_\theta)_{p=1}^P$.

- **"Basis functions"**: $\partial_p u_\theta$ (change every step!)
- **Gram matrix**: $G_{\theta_{pq}} = \langle \partial_p D[u_\theta] \,,\, \partial_q D[u_\theta] \rangle_{\mathrm{L}^2}$
- **Right-hand side**: $\nabla \ell(\theta)$
- **Update**: $\theta_{t+1} = \theta_t - \eta \, G_\theta^\dagger \nabla \ell(\theta)$

Key difference with FEM

In FEM, K is assembled *once* and is sparse/well-conditioned for elliptic PDEs.
In PINNs, G_θ is **dense**, $P \times P$, changes at *every* step, costs $\mathcal{O}(P^3)$ to invert, and is **severely ill-conditioned**, because neural network feature maps have **exponentially decaying singular values** (König and Richter1984) .

Problem 1: Cost

Naive natural gradient requires:

1. Form $G_\theta \in \mathbb{R}^{P\times P}$: $\mathcal{O}(P^2)$
2. Invert G_θ: $\mathcal{O}(P^3)$

That is the cost of a full FEM solve **at every gradient step**.

$\Rightarrow$ **ANaGRAM** (Schwencke and Furtlehner2025) : reduce to $\mathcal{O}(\min(PS^2, P^2S))$ via a kernel reformulation.

Problem 2: Ill-conditioning

Unlike FEM stiffness matrices, G_θ is severely ill-conditioned:

- Singular values of $\widehat{\phi}_\theta$ (the Jacobian of $(D, B) \circ u_\theta$) decay **exponentially** (König and Richter1984).
- Naive pseudoinverse amplifies noise catastrophically
- A regularization is unavoidable ; but which one?

$\Rightarrow$ **AMStraMGRAM** (Schwencke et al.2025): principled adaptive cutoff based on the RCE.

ANaGRAM: reframing natural gradient as a kernel method

Intuition

$$\widehat{\ell}(u) := \frac{1}{2S} \sum_{i=1}^{S} \left(u(x_i) - f(x_i)\right)^2 .$$

$$\frac{\mathrm{d}u_{\theta(t)}}{\mathrm{d}t}(x) = -\frac{1}{S} \sum_{i=1}^{S} \left(u_\theta(x_i) - f(x_i)\right) \delta_{x_i}$$

Gradient descent (Jacot et al.2018)

$$\frac{\mathrm{d}u_{\theta(t)}}{\mathrm{d}t}(x) = -\frac{1}{S} \sum_{i=1}^{S} \mathrm{NTK}_\theta(\mathrm{x}, \mathrm{x_i}) \left(\mathrm{u}_\theta(\mathrm{x_i}) - \mathrm{f}(\mathrm{x_i})\right)$$

$$\text{with: } \mathrm{NTK}_\theta(\mathrm{x}, \mathrm{y}) := \sum_{\mathrm{p}=1}^{\mathrm{P}} \partial_{\mathrm{p}} \mathrm{u}_\theta(\mathrm{x}) \partial_{\mathrm{p}} \mathrm{u}_\theta(\mathrm{y})^\top .$$

Natural Gradient (Rudner et al.2019)

$$\frac{\mathrm{d}u_{\theta(t)}}{\mathrm{d}t}(x) = -\frac{1}{S} \sum_{i=1}^{S} \mathrm{NNTK}_\theta(\mathrm{x}, \mathrm{x_i}) \left(\mathrm{u}_\theta(\mathrm{x_i}) - \mathrm{f}(\mathrm{x_i})\right)$$

$$\text{with: } \mathrm{NNTK}_\theta(\mathrm{x}, \mathrm{y}) := \sum_{1 \leqslant \mathrm{p}, \mathrm{q} \leqslant \mathrm{P}} \partial_{\mathrm{p}} \mathrm{u}_\theta(\mathrm{x}) \mathrm{G}^\dagger_{\theta_{\mathrm{p,q}}} \partial_{\mathrm{q}} \mathrm{u}_\theta(\mathrm{y})^\top ,$$

$$G_{\theta_{p,q}} = \left\langle \partial_p u_\theta \, , \, \partial_q u_\theta \right\rangle .$$

Neural Tangent Kernel (NTK)

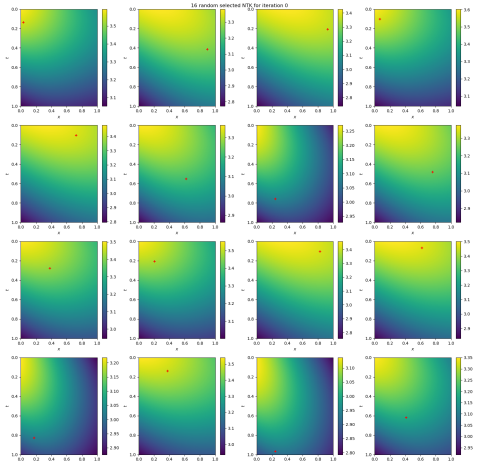


Figure: NTK for the Heat equation.

Reading: the plot represents the function $NTK_{\theta_0}(\cdot, x_i)$; the red cross marks the point x_i.

Key Observation

The empirical dynamics takes place in:

$$\widehat{T}_\theta \mathcal{M} := \mathrm{Span}\left((N)NTK_\theta(x_i, \cdot) \,:\, (x_i)_{1 \leqslant i \leqslant N}\right).$$

We can define the empirical Natural Gradient:

$$\theta_{t+1} = \theta_t - \eta \, \mathrm{d}u^\dagger_{\theta_t}\left(\Pi^\perp_{\widehat{T}_{\theta_t}\mathcal{M}} \nabla \mathcal{L}_{|u_{\theta_t}}\right).$$

Theorem (ANaGRAM)

Under mild assumptions:

$$du^\dagger_{\theta_t}\left(\Pi^\perp_{\widehat{T}_{\theta_t}\mathcal{M}} \nabla \mathcal{L}_{|u_{\theta_t}}\right) \simeq \widehat{\phi}^\dagger_{\theta_t} \widehat{\nabla \mathcal{L}}_{\theta_t},$$

with: for all $1 \leqslant p \leqslant P, 1 \leqslant i \leqslant S$

- $\widehat{\phi}_{\theta_t i,p} := \partial_p u_{\theta_t}(x_i)$
- $\widehat{\nabla \mathcal{L}}_{\theta_t i} := \nabla \mathcal{L}_{|u_{\theta_t}}(x_i)$

Key fact

$\hat{\phi}^\dagger_{\theta_t}$ **can be computed with a SVD, with complexity** $\mathrm{O}\big(\min(PS^2, P^2S)\big)$.

Corollary

There exists P *points* $(\hat{x}_i)$ *such that:*

$$\Pi^\perp_{\widehat{T}_\theta \mathcal{M}} \nabla \mathcal{L}_{u_\theta} = \Pi^\perp_{T_\theta \mathcal{M}} \nabla \mathcal{L}_{|u_\theta}.$$

Key Observation

The empirical dynamics takes place in:

$$\widehat{T}_\theta\mathcal{M} := \mathrm{Span}\left((N)NTK_\theta(x_i,\cdot) \,:\, (x_i)_{1\leqslant i\leqslant N}\right).$$

We can define the empirical Natural Gradient:

$$\theta_{t+1} = \theta_t - \eta\, \mathrm{d}u^\dagger_{\theta_t}\left(\Pi^\perp_{\widehat{T}_{\theta_t}\mathcal{M}}\nabla\mathcal{L}_{|u_{\theta_t}}\right).$$

Byproduct

Yields an optimal criterion for (x_i) choice:

$$(x_i)^\star = \underset{(x_i)\in\Omega^S}{\mathrm{argmin}} \left\|\Pi^\perp_{\widehat{T}^{(x_i)}_{\theta,K}\mathcal{M}}\nabla\mathcal{L}_{|u_{\theta_t}} - \nabla\mathcal{L}_{|u_{\theta_t}}\right\|_{\mathrm{L}^2(\Omega)}$$

Theorem (ANaGRAM)

Under mild assumptions:

$$du^\dagger_{\theta_t}\left(\Pi^\perp_{\widehat{T}_{\theta_t}\mathcal{M}}\nabla\mathcal{L}_{|u_{\theta_t}}\right) \simeq \widehat{\phi}^\dagger_{\theta_t}\widehat{\nabla\mathcal{L}}_{\theta_t},$$

with: for all $1\leqslant p\leqslant P, 1\leqslant i\leqslant S$

- $\widehat{\phi}_{\theta_t\, i,p} := \partial_p u_{\theta_t}(x_i)$
- $\widehat{\nabla\mathcal{L}}_{\theta_t\, i} := \nabla\mathcal{L}_{|u_{\theta_t}}(x_i)$

Key fact

$\hat{\phi}^\dagger_{\theta_t}$ **can be computed with a SVD, with complexity** $\mathrm{O}\big(\min(\mathrm{PS}^2, \mathrm{P}^2\mathrm{S})\big)$.

Corollary

There exists P *points* $(\hat{x}_i)$ *such that:*

$$\Pi^\perp_{\widehat{T}_\theta\mathcal{M}}\nabla\mathcal{L}_{u_\theta} = \Pi^\perp_{T_\theta\mathcal{M}}\nabla\mathcal{L}_{|u_\theta}.$$

1+1 D Heat equation

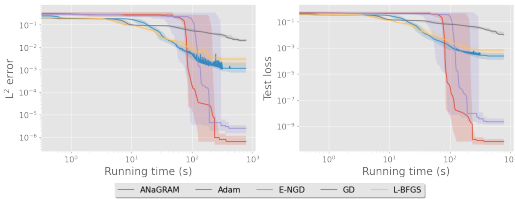


Figure: Performance comparison w.r.t running time for Heat equation in 1+1 D:

$$\begin{cases} \partial_t u - \frac{1}{4}\partial_{xx} u = 0 & \text{in } [0,1]^2 \\ u = 0 & \text{on } [0,1] \times \{0,1\} \\ u = sin(\pi x) & \text{on } \{0\} \times [0,1] \end{cases}$$

1+1 D Allen-Cahn equation

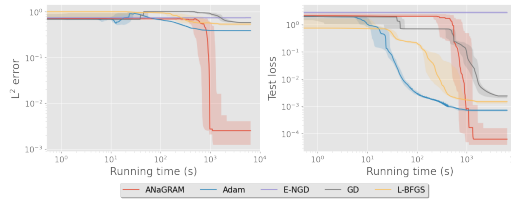


Figure: Performance comparison w.r.t running time for Allen-Cahn equation in 1+1 D:

$$\begin{cases} \partial_t u - 10^{-3}\,\partial_{xx} u = 5(u - u^3) & \text{in } \Omega = [0,1] \times [-1,1] \\ u = -1 & \text{on } \partial\Omega_b = [0,1] \times \{-1,1\} \\ u(0,x) = x^2 \cos(\pi x) & \text{on } \partial\Omega_0 = \{0\} \times [-1,1] \end{cases}$$

Note: E-NGD refers to Müller and Zeinhofer2023).

Intuition

Computing the Gram matrix is only useful when you need to separate signal from noise:

- When $\nabla \mathcal{L}_{u_{\theta_t}} \notin T_{\theta_t}\mathcal{M}$
- To accurately project from $T_\theta \mathcal{M}$ onto $\widehat{T}_\theta \mathcal{M}$

ANaGRAM = exact NTK interpolation

Consider kernel interpolation of $\nabla \mathcal{L}$ with the NTK kernel:

$$\widetilde{\nabla \mathcal{L}}_\theta(x) = \sum_{i,j} NTK(x, x_i) \widehat{K}^\dagger_{i,j} \widehat{\nabla \mathcal{L}}_j;$$

$$K_{i,j} := NTK(x_i, x_j)$$

Then the ANaGRAM update exactly follows this direction.

Insights on the approximation theorem

Why switch from NNTK to NTK?

- Computational convenience.
- **More importantly**: G_θ is so ill-conditioned that its pseudoinverse is **numerically meaningless** and so is the NNTK.

ANaGRAM

- **"Basis functions"**: $NTK_\theta(x_i, \cdot)$ (change every step!)
- **"Gram matrix"**: $\widehat{K}_{\theta_{ij}} = NTK_\theta(x_i, x_j)$
- **Update**: $\theta_{t+1} = \theta_t - \eta \, \widehat{\phi}^\dagger_{\theta_t} \widehat{\nabla \mathcal{L}}_{\theta_t}$

Key point: the NTK **adapts** as θ evolves.

AMStraMGRAM: adaptive cutoff regularization

In-Depth Empirical Analysis of *Cutoff* Regularization in *ANaGRAM*

SVD pseudoinverse details

$$\widehat{\phi}_\theta = \widehat{V}\,\widehat{\Delta}\,\widehat{U}^\top; \qquad \widehat{\phi}^\dagger_{\theta_t} = \widehat{U}\widehat{\Delta}^\dagger\widehat{V}^\top.$$

Singular values

In practice, we apply a *cutoff*:

$$\widehat{\Delta}^{\dagger_\alpha} := \begin{cases} \widehat{\Delta}_i^{-1} & \text{if } \widehat{\Delta}_i \geqslant \alpha \\ 0 & \text{otherwise} \end{cases},$$

with $\alpha > 0$ the cutoff level. Thus:

$$\widehat{\phi}^{\dagger_\alpha}_\theta \widehat{\nabla\mathcal{L}}_\theta = \sum_{i=1}^{r_\alpha} \widehat{U}_i \widehat{\Delta}_i^{-1} \widehat{V}_i^\top \widehat{\nabla\mathcal{L}}_\theta,$$

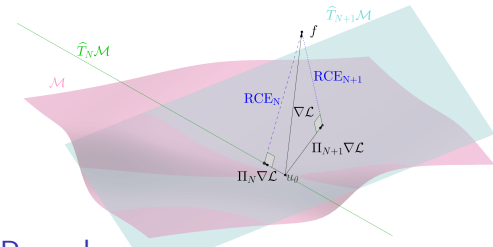
with $r_\alpha := \#\{i : \widehat{\Delta}_i \geqslant \alpha\} \leqslant \min(P, S)$.

Reconstruction Error (RCE)

$$\mathrm{RCE}_n = \frac{1}{\sqrt{S}} \left\| \widehat{\nabla\mathcal{L}}_\theta - \sum_{i=1}^{n} \widehat{V}_i \widehat{V}_i^\top \widehat{\nabla\mathcal{L}}_\theta \right\|_{\mathbb{R}^S}$$

Intuition on RCE

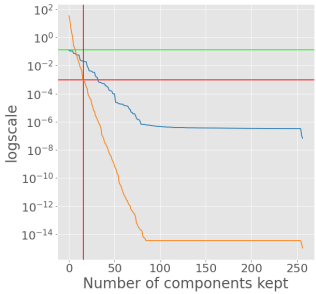
RCE_N quantifies the part of $\nabla\mathcal{L}_\theta$ orthogonal to the top N components of $\widehat{T}_\theta\mathcal{M}$.



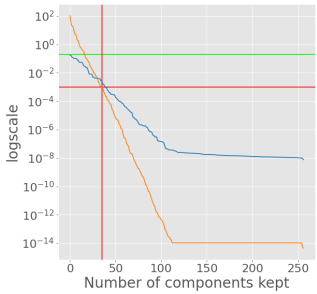
Remark

- $\lim_{S\to\infty} \widehat{T}_\theta\mathcal{M} = T_\theta\mathcal{M}$
- $\mathrm{RCE}_0^2 = \frac{1}{S}\left\|\widehat{\nabla\mathcal{L}_\theta}\right\|^2_{\mathbb{R}^S} = \widehat{\ell}(\theta)$

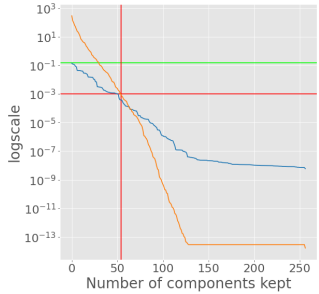
Empirical insights on the *cutoff* impact in *ANaGRAM*



(a) Iteration 0: intersection point between singular values and RCE lies before cutoff.

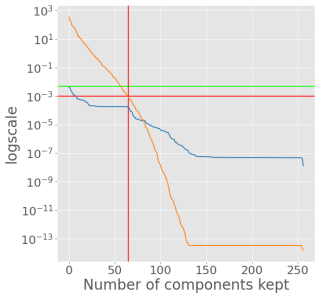


(b) Iteration 40: intersection point shifts rightward toward cutoff.

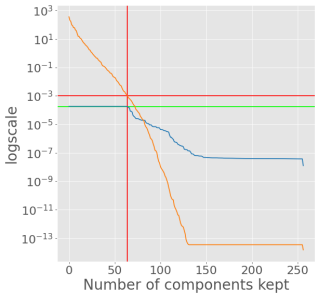


(c) Iteration 90: intersection point passes the cutoff threshold.

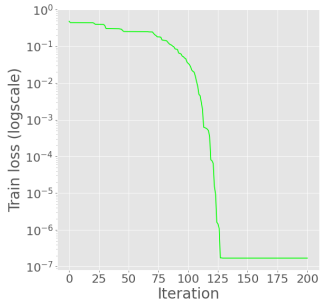
Empirical insights on the *cutoff* impact in *ANaGRAM*



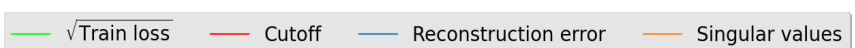
(d) Iteration 120. Beginning of *flattening*: RCE stabilizes at constant level before cutoff.

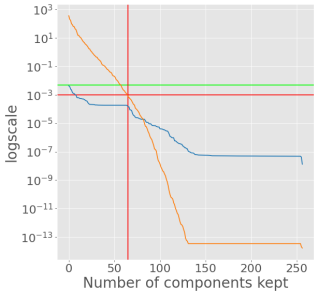


(e) Iteration 150: End of the *flattening phenomenon*. The train loss reaches the flattened part of the RCE.

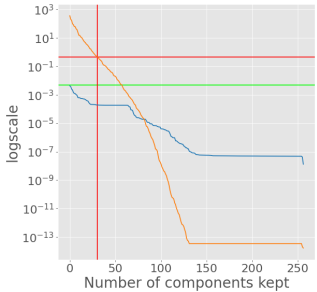


(f) Train loss dynamics.

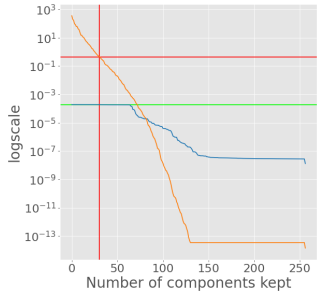




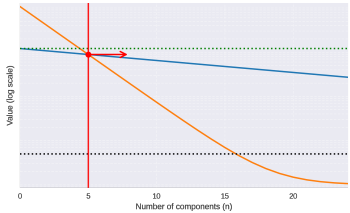
(g) Incomplete flattening of the RCE with a fixed cutoff at 10^{-3}.



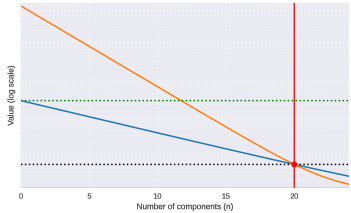
(h) New cutoff located roughly at the location of the "elbow" in the RCE curve.



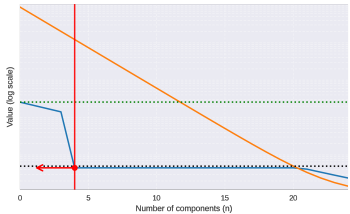
(i) Complete flattening after one natural gradient step with the new cutoff.



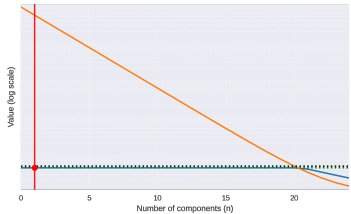
(j) Early iterations: cutoff set at the $\mathrm{RCE}-$singular values intersection.



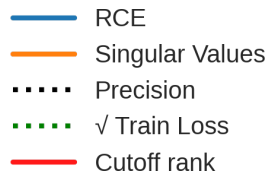
(k) The $\mathrm{RCE}-$singular values intersection drops below precision.



(l) Flattening phase: Cutoff is set at the $\mathrm{RCE}-$precision intersection.



(m) End of flattening: The RCE and loss stands at precision level.



Experiment	Train Loss		L_2 Error	
	AMStraMGRAM	ANaGRAM	AMStraMGRAM	ANaGRAM
Heat Equation	**6.29e-29 ± 6.78e-30**	8.56e-11 ± 7.05e-11	**2.32e-14 ± 1.14e-14**	1.28e-06 ± 1.75e-06
Laplace 2D	**1.46e-28 ± 1.87e-29**	4.27e-13 ± 4.66e-13	**2.24e-15 ± 2.52e-16**	3.49e-09 ± 3.58e-09
Laplace 5D	**2.04e-08 ± 1.16e-08**	6.37e-08 ± 7.01e-08	**2.12e-05 ± 8.15e-06**	4.00e-05 ± 2.93e-05
Allen–Cahn	**3.19e-11 ± 2.37e-11**	2.19e-04 ± 4.16e-04	**5.87e-05 ± 6.25e-06**	4.32e-03 ± 5.93e-03

Experiment	Train Loss		L_2 Error	
	AMStraMGRAM	SSBroyden *	AMStraMGRAM	SSBroyden *
Burgers (1+1 D)	**2.99e-12 ± 9.26e-13**	2.92e-10 ± 1.45e-10	**1.5e-06 ± 9.43e-7**	1.59e-06 ± 1.02e-6
Non-Linear Poisson	**8.51e-24 ± 2.24e-24**	3.03e-16 ± 3.82e-16	6.81e-10 ± 1.41e-09	**9.29e-12 ± 5.85e-12**
Allen–Cahn	3.19e-11 ± 2.37e-11	**6.42e-12 ± 5.52e-12**	5.87e-05 ± 6.25e-06	**3.94e-06 ± 1.72e-06**

* refers to the order two method of Urbán et al.2025), with adaptive sampling and hard constraint enforcement on boundary conditions.

Overfitting

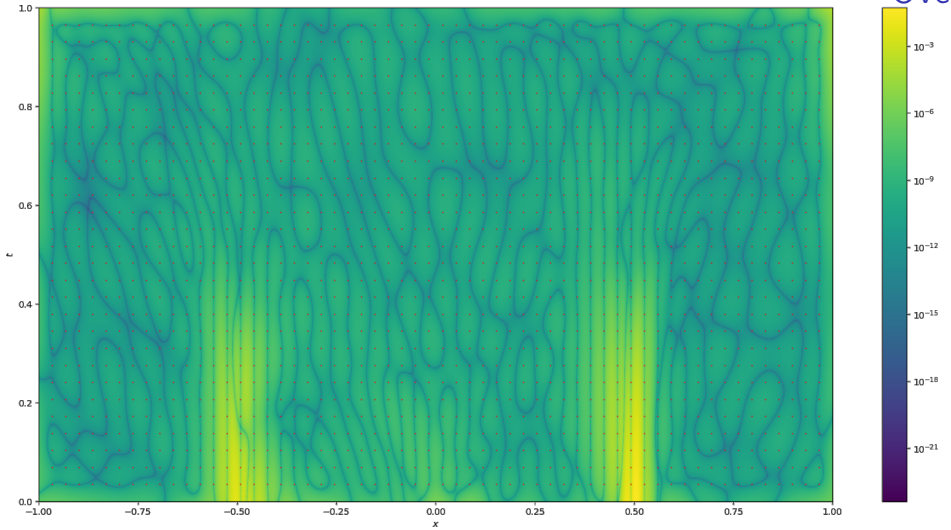


Figure: Overfitting on Allen–Cahn: residual lines align with sampling lines.

Overfitting

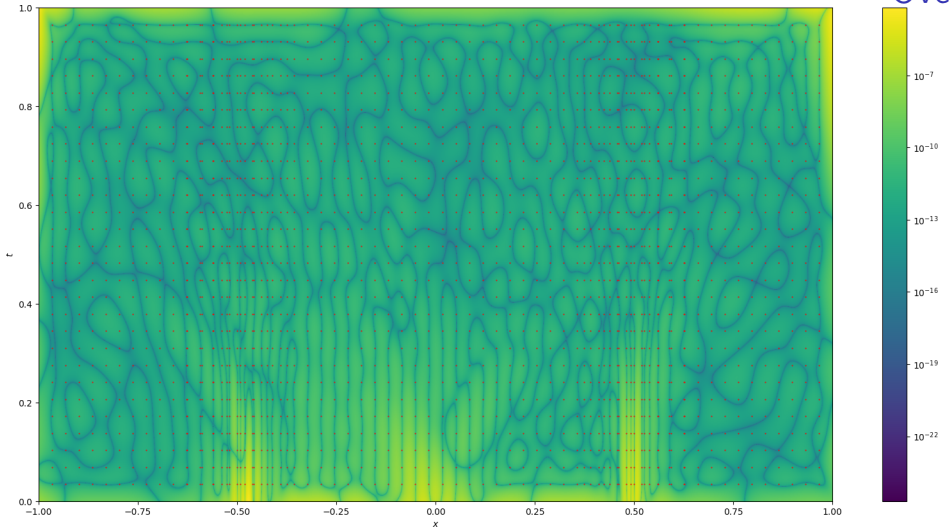


Figure: Overfitting on Allen–Cahn: densifying the sampling mitigates overfitting.

Conclusion and perspectives

Three take-aways

1. **Natural gradient = discrete Green's function** on the evolving tangent space $T_\theta\mathcal{M}$ — connects PINNs optimization to Galerkin theory.
2. **ANaGRAM** = exact kernel interpolation with the operator-NTK. Cost $\mathcal{O}(\min(PS^2, P^2S))$ via SVD of the Jacobian. No Gram matrix inversion.
3. **AMStraMGRAM** = principled adaptive cutoff tracking the RCE–singular value intersection. Reaches **machine-precision accuracy** on standard benchmarks.

Publications

- Schwencke and Furtlehner2025): ANaGRAM — ICLR 2025
- Schwencke, Rousselot, Shilova, Furtlehner (2025): AMStraMGRAM — arXiv preprint

Perspectives

- Extension of the cutoff adaptation strategy
- Proof of the connection to lazy training
- Optimal adaptive collocation points
- Design of a multiscale kernel strategy
- Extension to operator learning

ScimBa — Scientific Machine Learning in Python/PyTorch

- Open-source library for hybrid SciML / PDE solvers
- Supports PINNs, Deep Ritz, neural Galerkin, neural semi-Lagrangian
- Multiple approximation spaces: neural nets, kernel methods, Fourier
- Complex geometries via level-set techniques
- All natural gradient preconditioners implemented:
  `pinn_preconditioners.anagram_ng`
  `pinn_preconditioners.energy_ng`
  `pinn_preconditioners.nystrom_ng`
- AMD ROCm support (HPC-ready)

Get started

`pip install scimba` | `uv add scimba`

`https://www.scimba.org` | GitLab: `gitlab.com/scimba/scimba`

Thank you for your attention !

Amari, S.-I. and S. C. Douglas (1998): "Why Natural Gradient?" in *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181)*, IEEE, vol. 2, 1213–1216.

Jacot, A., F. Gabriel, and C. Hongler (2018): "Neural Tangent Kernel: Convergence and Generalization in Neural Networks," *Advances in neural information processing systems*, 31.

König, H. and S. Richter (1984): "Eigenvalues of Integral Operators Defined by Analytic Kernels," *Mathematische Nachrichten*, 119, 141–155.

Lagaris, I. E., A. Likas, and D. I. Fotiadis (1998): "Artificial Neural Networks for Solving Ordinary and Partial Differential Equations," *IEEE transactions on neural networks*, 9, 987–1000.

Müller, J. and M. Zeinhofer (2023): "Achieving High Accuracy with PINNs via Energy Natural Gradient Descent," in *International Conference on Machine Learning*, PMLR, 25471–25485.

Raissi, M., P. Perdikaris, and G. Karniadakis (2019): "Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations," *Journal of Computational Physics*, 378, 686–707.

Rudner, T. G., F. Wenzel, Y. W. Teh, and Y. Gal (2019): "The Natural Neural Tangent Kernel: Neural Network Training Dynamics under Natural Gradient Descent," in *4th Workshop on Bayesian Deep Learning (NeurIPS 2019)*.

Schwencke, N. and C. Furtlehner (2025): "ANaGRAM: A Natural Gradient Relative to Adapted Model for Efficient PINNs Learning," in *The Thirteenth International Conference on Learning Representations*.

Schwencke, N., C. Rousselot, A. Shilova, and C. Furtlehner (2025): "AMStramGRAM: Adaptive Multi-Cutoff Strategy Modification for ANaGRAM," in *arXiv Preprint*, arXiv Preprint.

Urbán, J. F., P. Stefanou, and J. A. Pons (2025): "Unveiling the Optimization Process of Physics Informed Neural Networks: How Accurate and Competitive Can PINNs Be?" *Journal of Computational Physics*, 523, 113656.

Backup: NTK vs NNTK at initialization and end of training

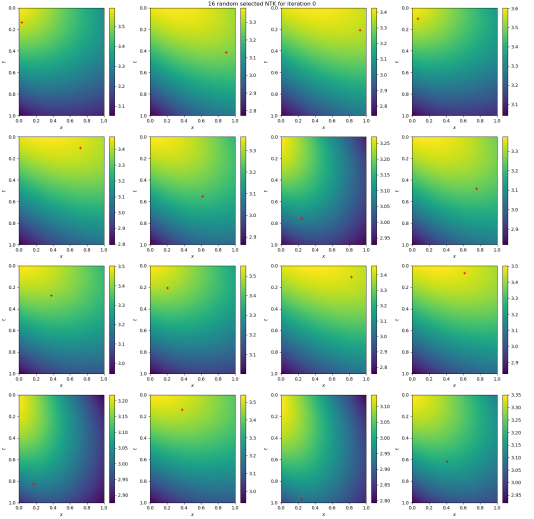


Figure: NTK at initialization

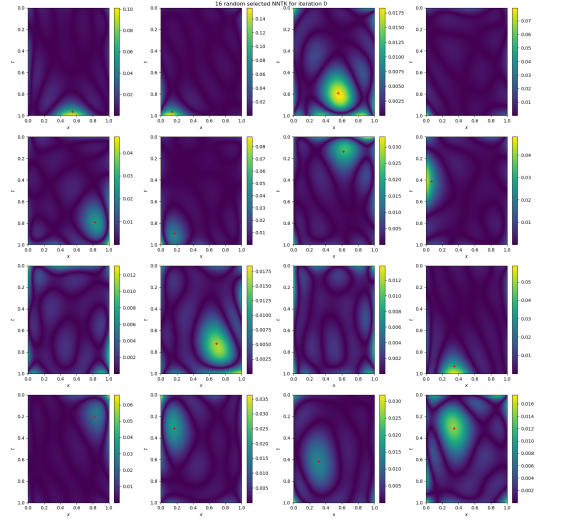


Figure: NNTK at initialization

Backup: ANaGRAM complements — 2D and 5D Laplace

2 D Laplace equation

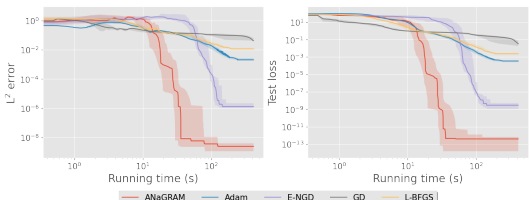


Figure: Performance comparison w.r.t running time, Laplace 2D.

5 D Laplace equation

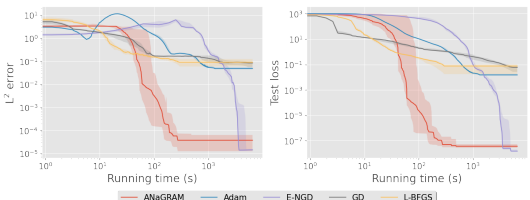


Figure: Performance comparison w.r.t running time, Laplace 5D.