
Operator Learning

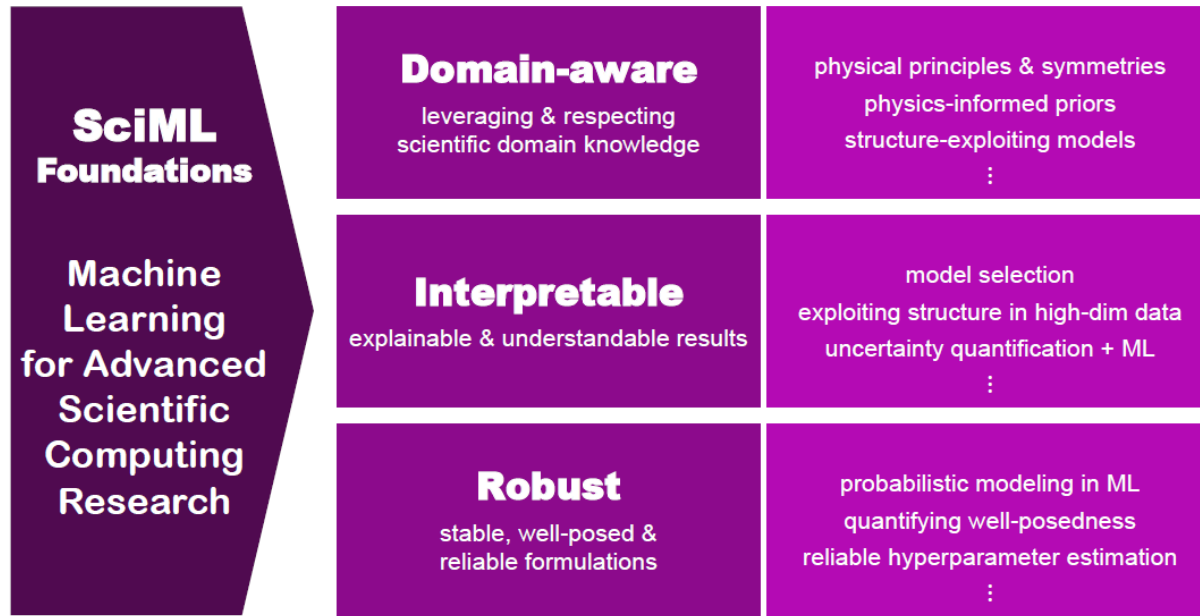
Learning solution operator

Nicolas Boullé

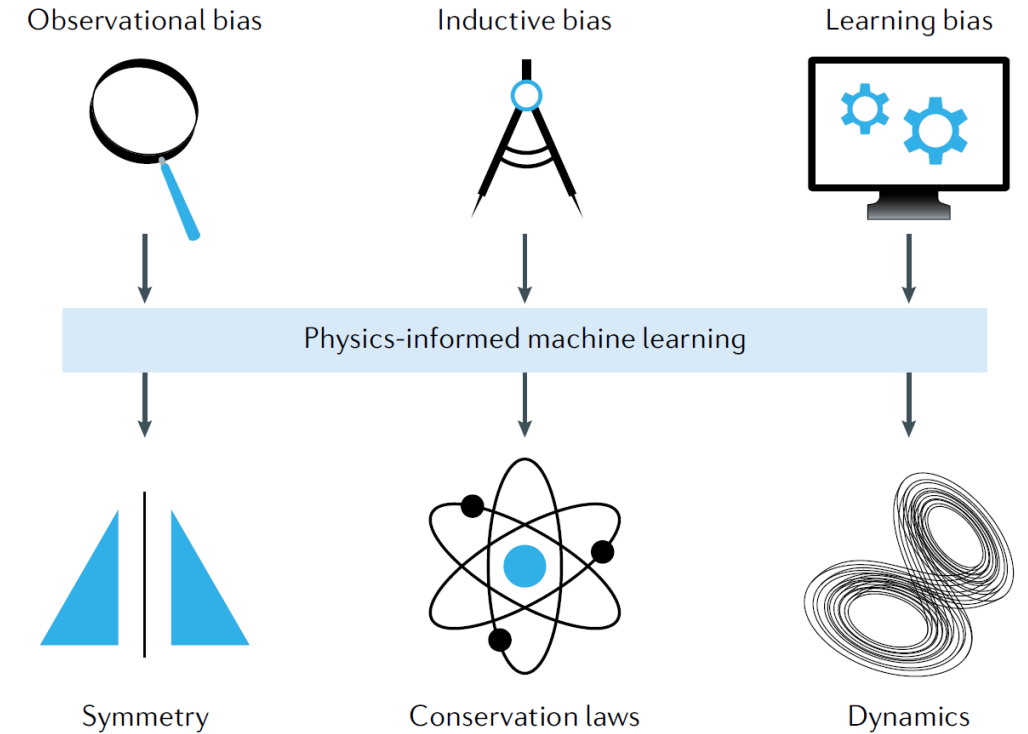
n.boulle@imperial.ac.uk

IMPERIAL

Scientific machine learning



[Baker et al., DOE Tech. Report, 2019]

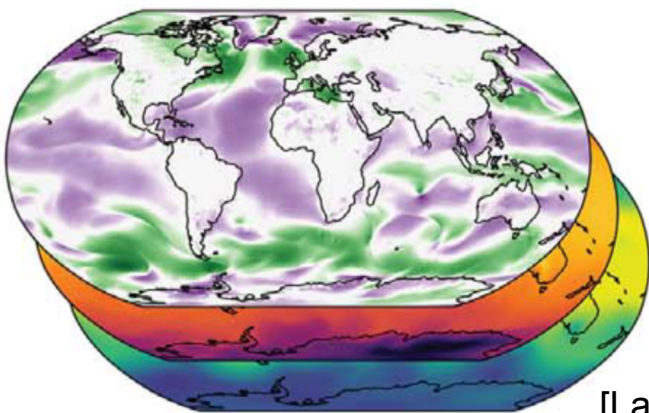


[Karniadakis et al., 2021]

Combining numerical analysis and machine learning for scientific discoveries.

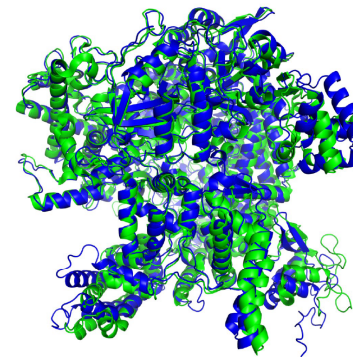
Recent applications of SciML

Weather forecasting



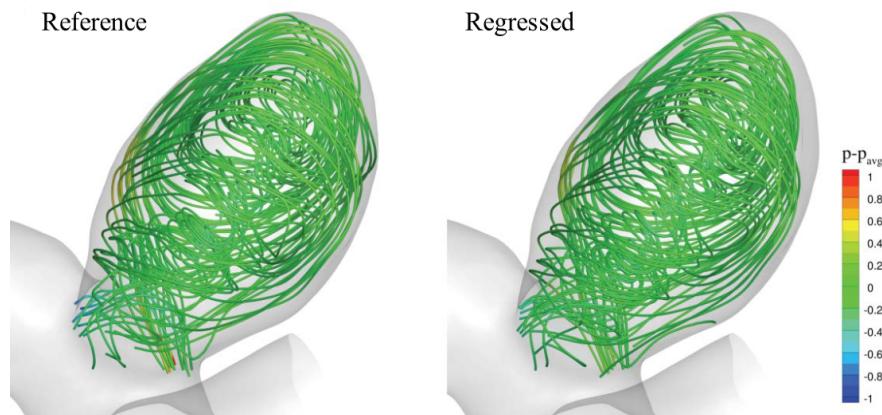
[Lam et al., Science, 2023]

Protein folding



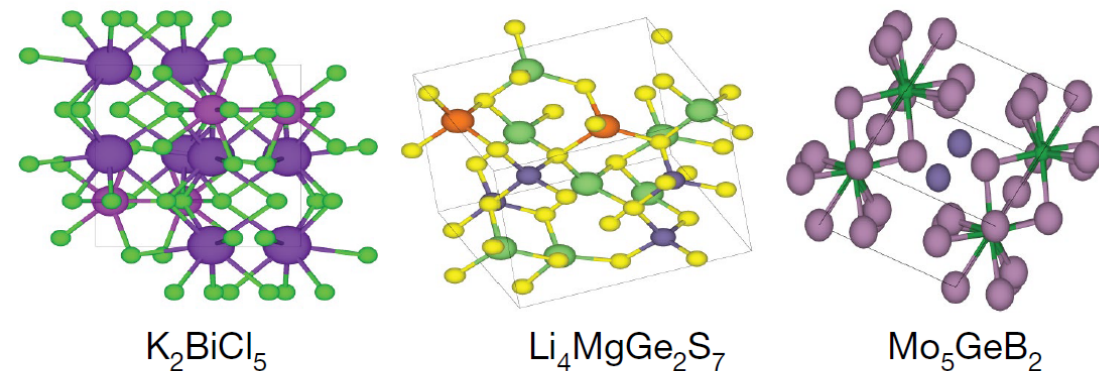
[Jumper et al., Nature, 2021]

Numerical simulations



[Raissi, Yazdani, Karniadakis, Science, 2020]

Materials discovery



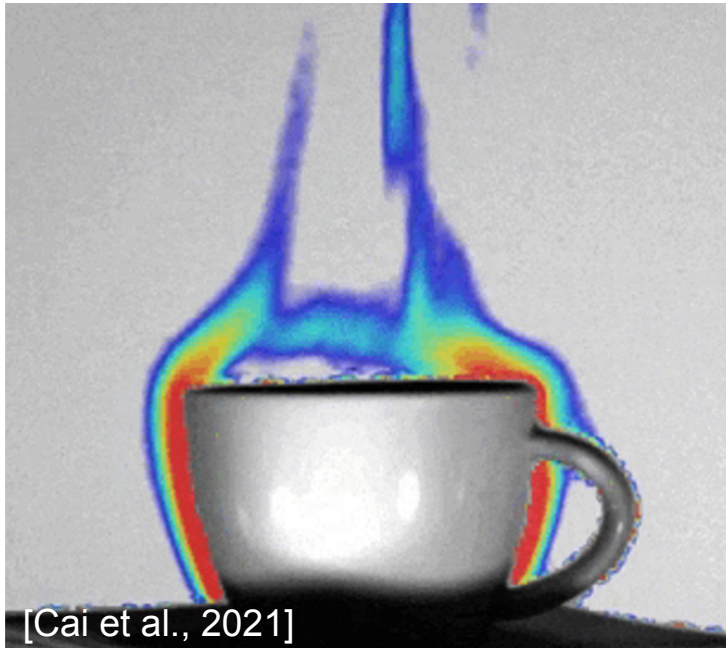
[Merchant et al., Nature, 2023]

Three areas to rule them all

Physics-informed machine learning

George Em Karniadakis^{1,2,3}, Ioannis G. Kevrekidis^{5,4}, Lu Lu⁵, Paris Perdikaris⁶, Sifan Wang⁷ and Liu Yang¹

Abstract | Despite great progress in simulating multiphysics problems using the numerical discretization of partial differential equations (PDEs), one still cannot seamlessly incorporate noisy data into existing algorithms, mesh generation remains complex, and high-dimensional problems governed by parameterized PDEs cannot be tackled. Moreover, solving inverse problems with hidden physics is often prohibitively expensive and requires different formulations and elaborate computer codes. Machine learning has emerged as a promising alternative, but training deep neural networks requires big data, not always available for scientific problems. Instead, such networks can be trained from additional information obtained by enforcing the physical laws (for example, at random points in the continuous space-time domain). Such physics-informed learning integrates (noisy) data and mathematical models, and implements them through neural networks or other kernel-based regression networks. Moreover, it may be possible to design specialized network architectures that automatically satisfy some of the physical invariants for better accuracy, faster training and improved generalization. Here, we review some of the prevailing trends in embedding physics into machine learning, present some of the current capabilities and limitations and discuss diverse applications of physics-informed learning both for forward and inverse problems, including discovering hidden physics and tackling high-dimensional problems.

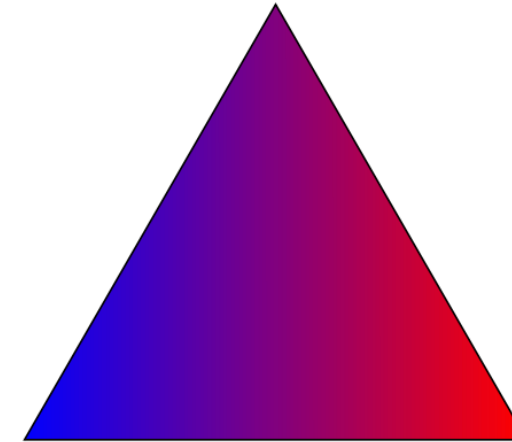


Inverse Problem



Forward Problem

PDE Discovery



PDE Solvers

Operator Learning



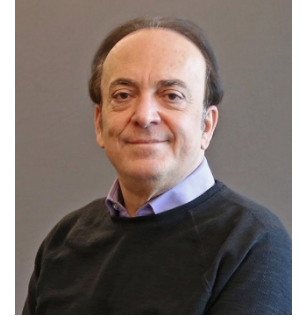
Small data
Lots of physics

Some data
Some physics

Lots of data
No physics

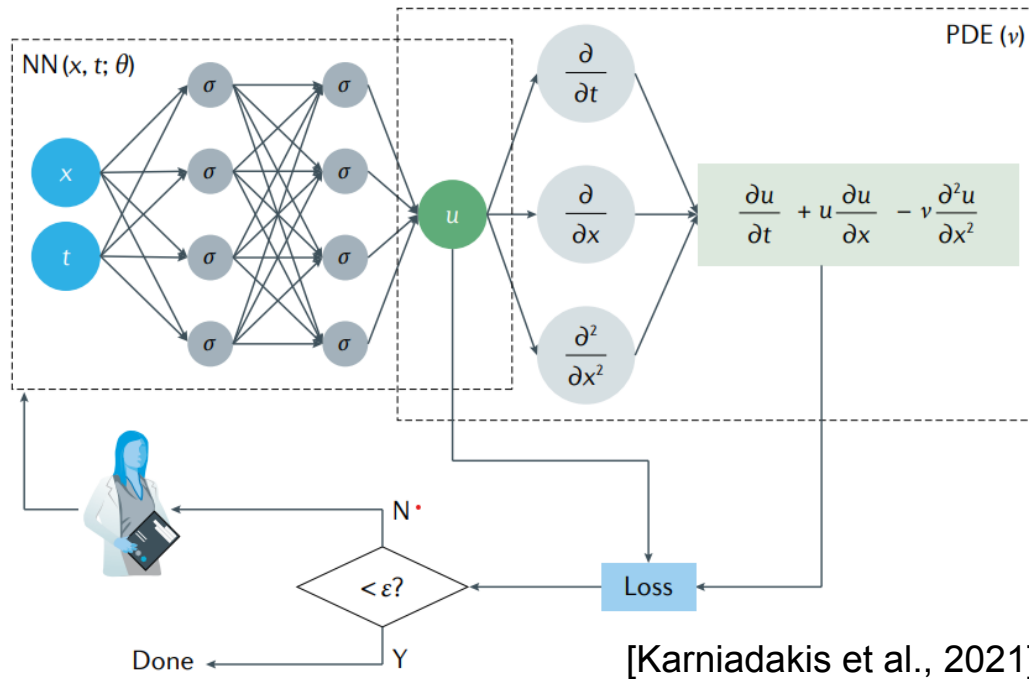
PDE solvers and SciML

Aim: Approximate the solution to a given PDE by a neural network



George Karniadakis

Physics-informed neural networks (**PINNs**)



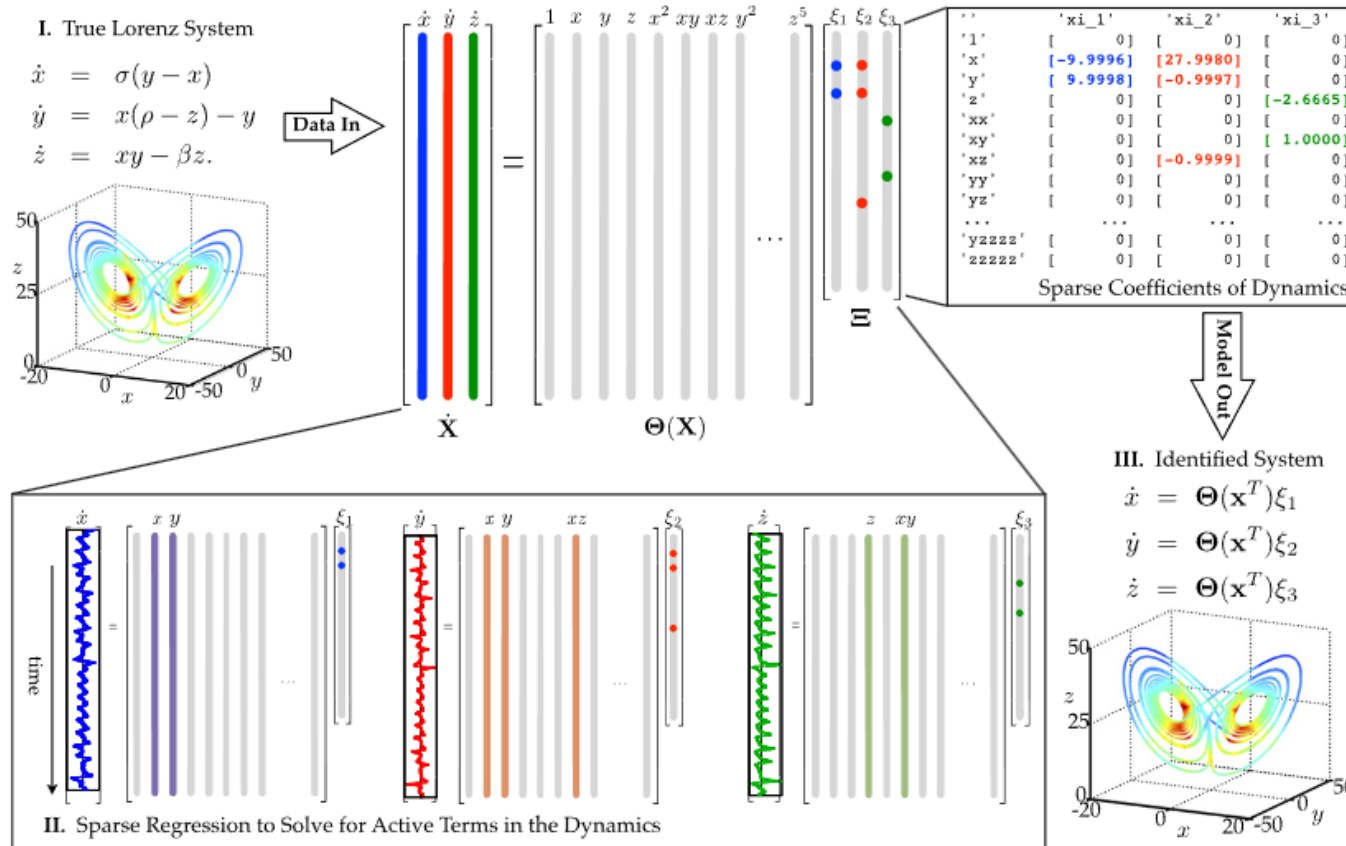
Advantages

- Easy to implement
- Can be embedded into downstream applications
- Works better for high-dimensional problems?

Related works: [Han et al, 2018], [Raissi et al., 2018], [Sirignano, Spiliopoulos, 2018], [Karniadakis et al., 2021], [Lu et al., 2021], [Cai et al., 2022]

PDE Discovery and SciML

Sparse Identification of Nonlinear Dynamics (SINDy)



[Brunton et al, 2016]



Steven Brunton



Nathan Kutz

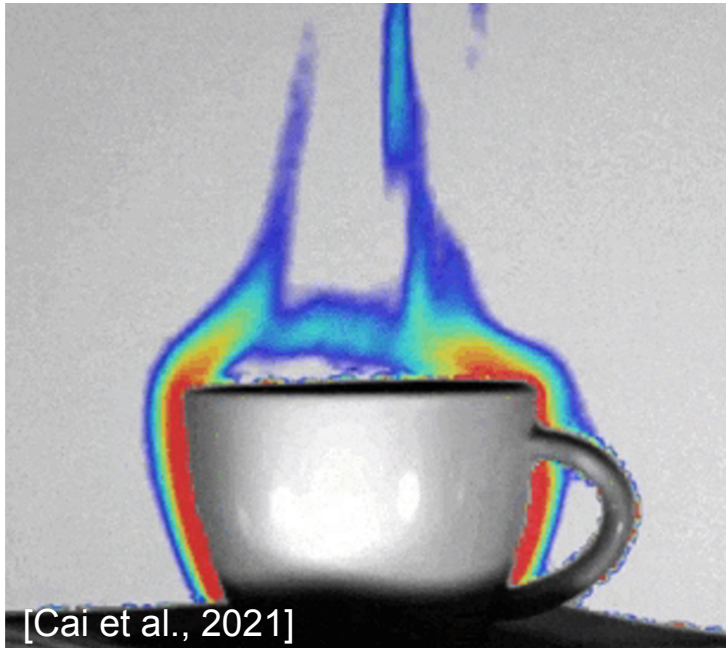
Related works: [Brunto et al., 2016], [Rudy et al., 2017], [Champion et al, 2019], [Udrescu, Tegmark, 2020], [Udrescu et al., 2020], [Liu, Tegmark, 2021]

Operator learning

Physics-informed machine learning

George Em Karniadakis^{1,2,3}, Ioannis G. Kevrekidis^{5,4}, Lu Lu⁵, Paris Perdikaris⁶, Sifan Wang⁷ and Liu Yang⁶

Abstract | Despite great progress in simulating multiphysics problems using the numerical discretization of partial differential equations (PDEs), one still cannot seamlessly incorporate noisy data into existing algorithms, mesh generation remains complex, and high-dimensional problems governed by parameterized PDEs cannot be tackled. Moreover, solving inverse problems with hidden physics is often prohibitively expensive and requires different formulations and elaborate computer codes. Machine learning has emerged as a promising alternative, but training deep neural networks requires big data, not always available for scientific problems. Instead, such networks can be trained from additional information obtained by enforcing the physical laws (for example, at random points in the continuous space-time domain). Such physics-informed learning integrates (noisy) data and mathematical models, and implements them through neural networks or other kernel-based regression networks. Moreover, it may be possible to design specialized network architectures that automatically satisfy some of the physical invariants for better accuracy, faster training and improved generalization. Here, we review some of the prevailing trends in embedding physics into machine learning, present some of the current capabilities and limitations and discuss diverse applications of physics-informed learning both for forward and inverse problems, including discovering hidden physics and tackling high-dimensional problems.

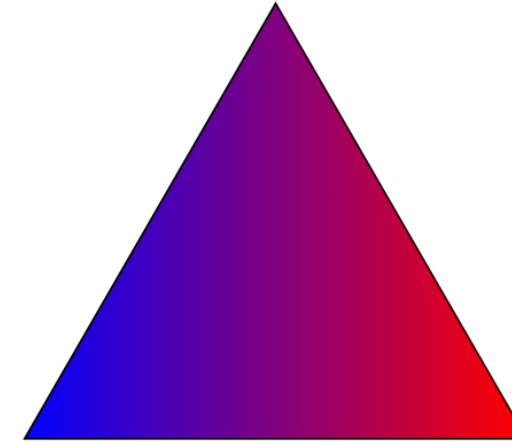


Inverse Problem



Forward Problem

PDE Discovery



PDE Solvers

Operator Learning



Small data
Lots of physics

Some data
Some physics

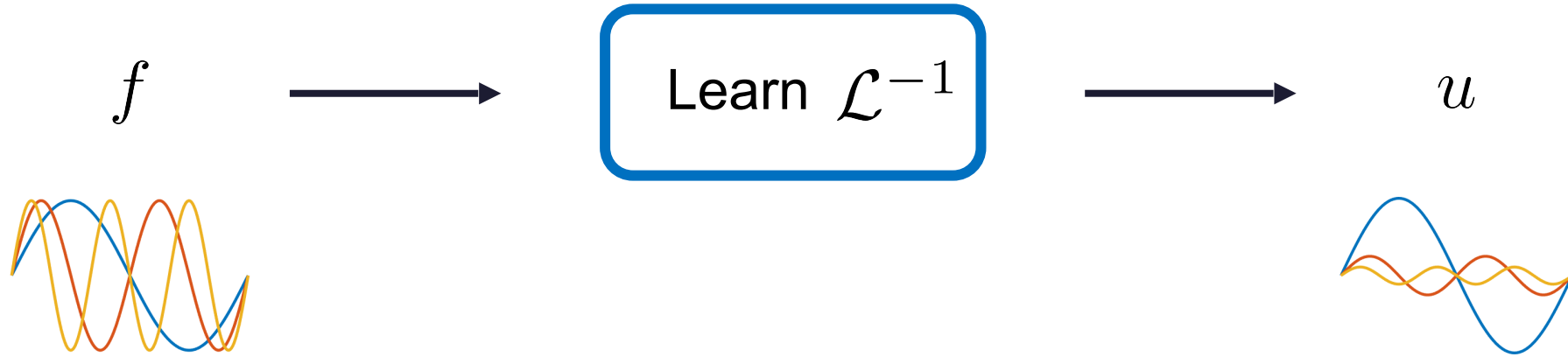
Lots of data
No physics

Recent surveys:

- B., Townsend, “A Mathematical Guide to Operator Learning”, 2024.
- Kovachki, Lanthaler, Stuart, “Operator Learning: Algorithms and Analysis”, 2024.

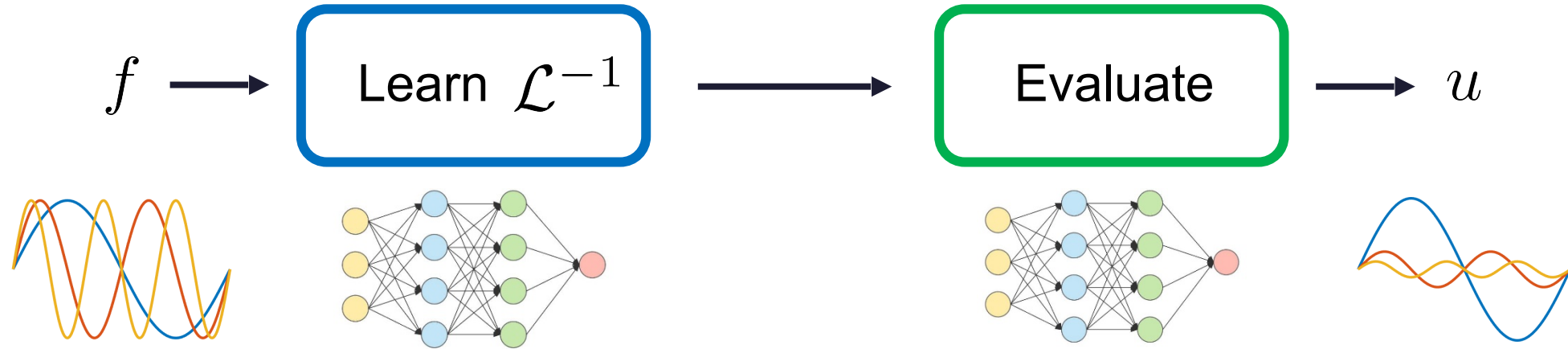
Introduction to operator learning

Aim: Approximate **solution operators** of unknown PDEs $\mathcal{L}(u) = f$



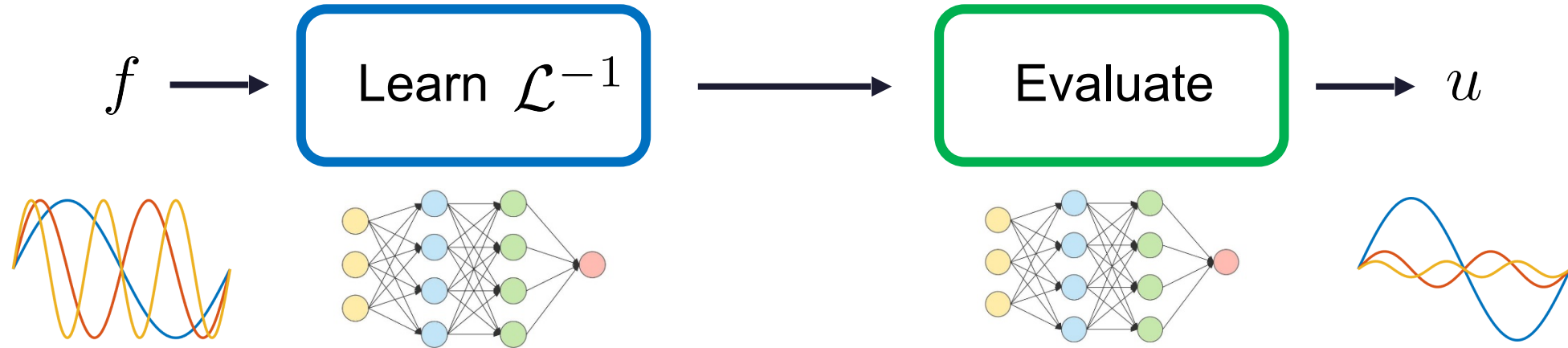
Introduction to operator learning

Aim: Approximate **solution operators** of unknown PDEs $\mathcal{L}(u) = f$

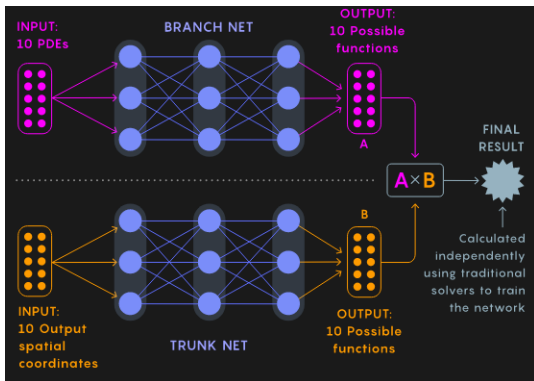


Introduction to operator learning

Aim: Approximate **solution operators** of unknown PDEs $\mathcal{L}(u) = f$

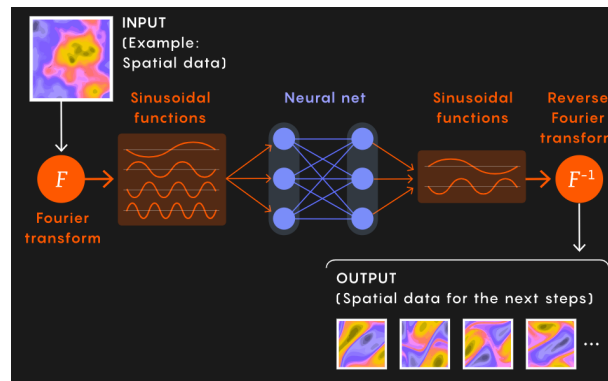


DeepONet



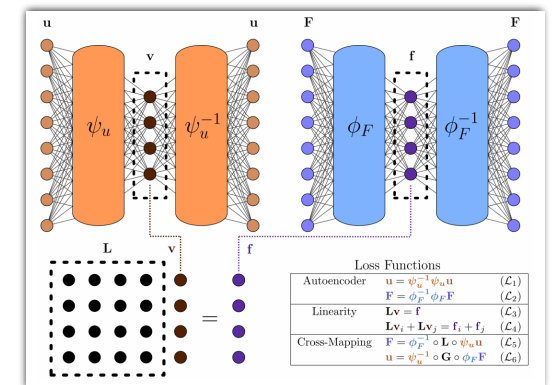
[Quanta Magazine; Lu et al, 2021]

Fourier Neural Operator



[Quanta Magazine; Li et al, 2020]

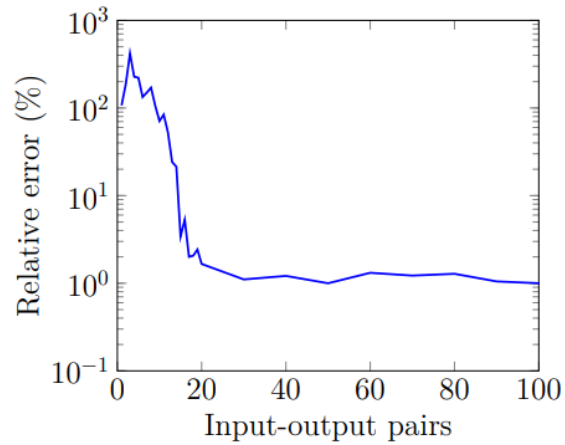
DeepGreen



[Gin et al., 2020]

Main challenges in operator learning

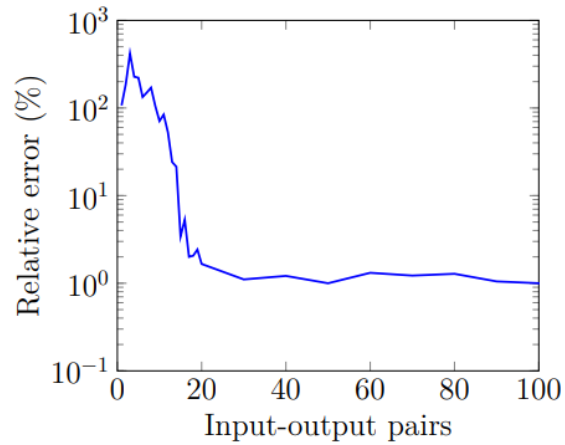
1. Theoretical results



- Type and number of training data
- Performance guarantees
- Neural network architectures
- Noise robustness

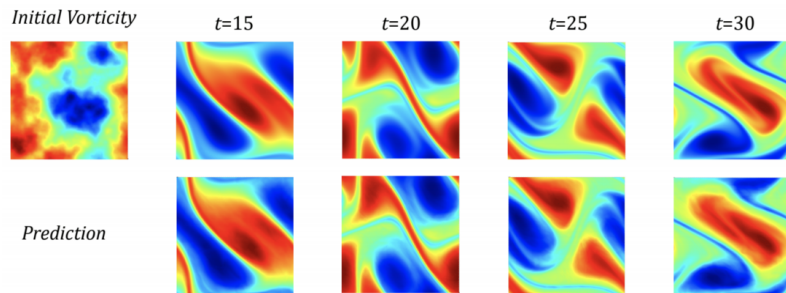
Main challenges in operator learning

1. Theoretical results



- Type and number of training data
- Performance guarantees
- Neural network architectures
- Noise robustness

2. Interpretability of the model



[Li et al, 2020]

- Dominant modes
- Symmetries
- Conservation laws
- Singularities

Operator learning and approximation theory

Density result for neural operators:

Theorem 1 (Universal Approximation Theorem for Operator).

Suppose that σ is a continuous non-polynomial function, X is a Banach space, $K_1 \subset X$, $K_2 \subset \mathbb{R}^d$ are two compact sets in X and $\mathbb{R}^d$, respectively, V is a compact set in $C(K_1)$, G is a nonlinear continuous operator, which maps V into $C(K_2)$. Then for any $\epsilon > 0$, there are positive integers n, p and m , constants $c_i^k, \xi_{ij}^k, \theta_i^k, \zeta_k \in \mathbb{R}, w_k \in \mathbb{R}^d, x_j \in K_1, i = 1, \dots, n, k = 1, \dots, p$ and $j = 1, \dots, m$, such that

$$\left| G(u)(y) - \underbrace{\sum_{k=1}^p \sum_{i=1}^n c_i^k \sigma \left(\sum_{j=1}^m \xi_{ij}^k u(x_j) + \theta_i^k \right)}_{\text{branch}} \underbrace{\sigma(w_k \cdot y + \zeta_k)}_{\text{trunk}} \right| < \epsilon$$

[Chen, Chen, 1995], [Lu et al., 2021]

Neural network results: [Cybenko, 1989], [Hornik et al., 1989], [Hornik, 1991], [Maioro, Pinkus, 1999], [Lu et al., 2017], [Kidger et al., 2020]

Operator learning and approximation theory

Entropy / asymptotic result for neural operators:

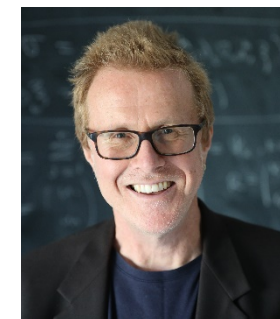
Theorem 2.11 (Curse of Parametric Complexity). Let $K \subset \mathcal{X}$ be a compact set in an infinite-dimensional Banach space $\mathcal{X}$. Assume that K contains an infinite-dimensional hypercube Q_α for some $\alpha > 1$. Then for any $r \in \mathbb{N}$ and $\delta > 0$, there exists $\bar{\epsilon} > 0$ and an r -times Fréchet differentiable functional $\mathcal{S}^\dagger : K \subset \mathcal{X} \rightarrow \mathbb{R}$, such that approximation to accuracy $\epsilon \leq \bar{\epsilon}$ by a functional $\mathcal{S}_\epsilon$ of neural network-type,

$$(2.10) \quad \sup_{u \in K} |\mathcal{S}^\dagger(u) - \mathcal{S}_\epsilon(u)| \leq \epsilon,$$

implies complexity bound $\text{cmplx}(\mathcal{S}_\epsilon) \geq \exp(c\epsilon^{-1/(\alpha+1+\delta)r})$; here $c, \bar{\epsilon} > 0$ are constants depending only on α, δ and r .
[Lanthaler, Stuart, 2024] $\diamond$



Samuel Lanthaler



Andrew Stuart

Recent surveys:

- DeVore, Hanin, Petrova, “Neural network approximation”, 2021.
- Kovachki, Lanthaler, Stuart, “Operator Learning: Algorithms and Analysis”, 2024.

Related works: [Telgarsky, 2016], [Yarotsky, 2017], [Lanthaler et al., 2022], [Kovachki et al., 2023], [Lanthaler et al., 2023], [Lanthaler, Stuart, 2024]

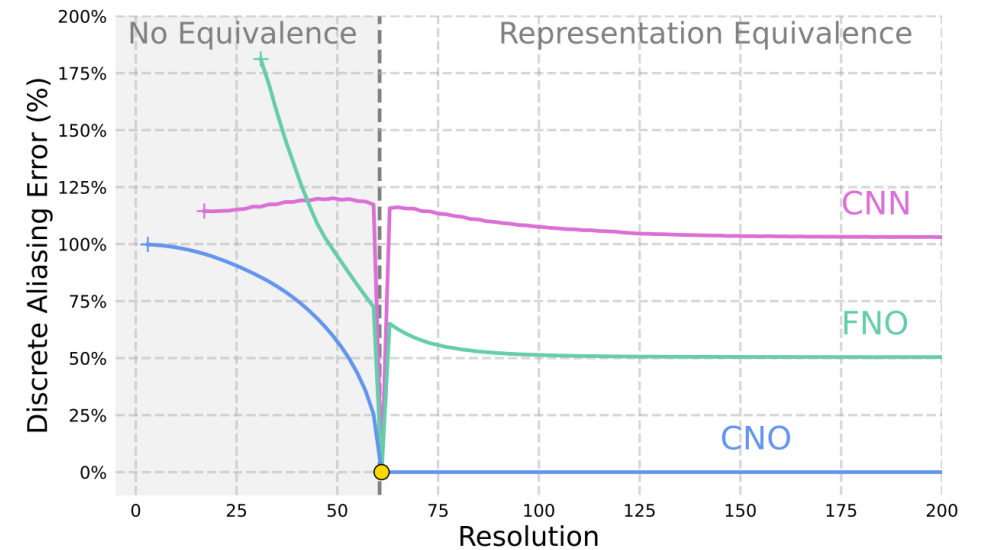
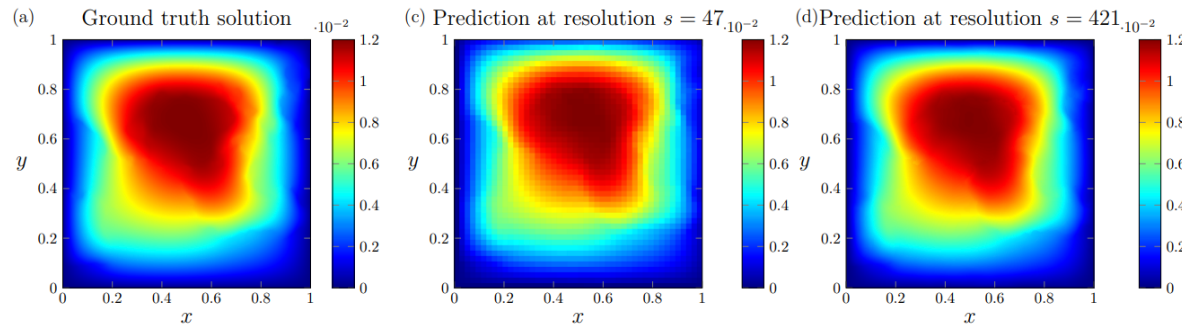
Operator learning and discretization errors

Resolution invariance / super-resolution:

Neural operators can be trained at low-resolution and evaluated at high-resolution



Sid Mishra



[Bartolucci et al., 2023]

Related works: [Li et al., 2020], [Bartolucci et al., 2023], [Kovachki et al., 2023], [Raonic et al., 2023]

Operator learning and numerical linear algebra

Neural operators can be defined as compositions of integral operators and nonlinear activation functions:

$$u_{i+1}(x) = \sigma \left(\int_{\Omega_i} K^{(i)}(x, y) u_i(y) \, dy + b_i(x) \right)$$

Operator learning and numerical linear algebra

Neural operators can be defined as compositions of integral operators and nonlinear activation functions:

$$u_{i+1}(x) = \sigma \left(\int_{\Omega_i} K^{(i)}(x, y) u_i(y) \, dy + b_i(x) \right)$$

Properties of the unknown PDE and neural operator architectures are related to structured matrix recovery problems:

Property of the PDE	Neural operator	Matrix structure
—	DeepONet	Low rank
Periodic BCs & const. coeffs	FNO	Circulant
Localized behavior	Graph neural operator	Banded
Elliptic / Parabolic	Multipole GNO	Hierarchical low rank

Operator learning and numerical linear algebra

Class of PDE $\longrightarrow$ Solution operator property $\longrightarrow$ Matrix structure

Operator learning **motivates numerical linear algebra research** on matrix recovery problems and infinite-dimensional NLA.

Structure	$\#x \mapsto Ax$	$\#x \mapsto A^\top x$
diagonal	1	-
block- k diagonal	k	0
tridiagonal	3	0
symm. tridiag.	2	-
rank- k	$k + p$	k
symm. rank- k	$k + p$	-
circulant	1	0
Toeplitz or Hankel	2	0
symmetric	N	-
orthogonal	N	0
Toeplitz-like	$2p + 4$	$2p + 4$



Diana Halikias

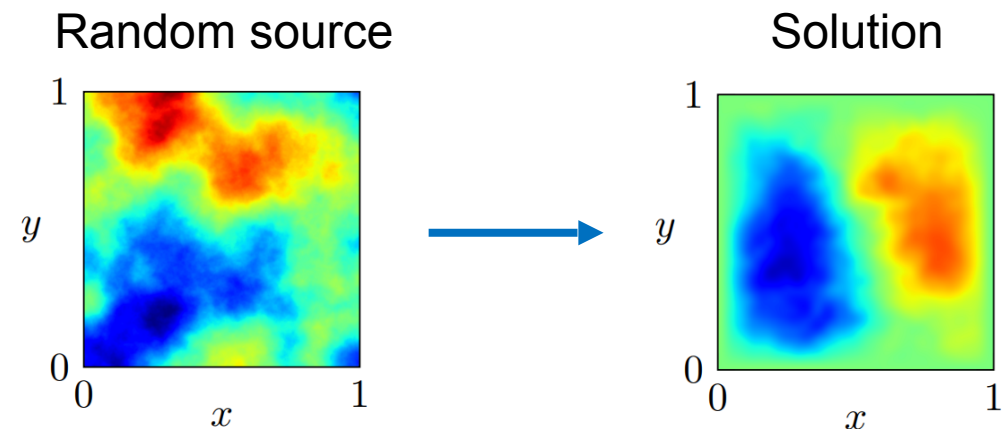
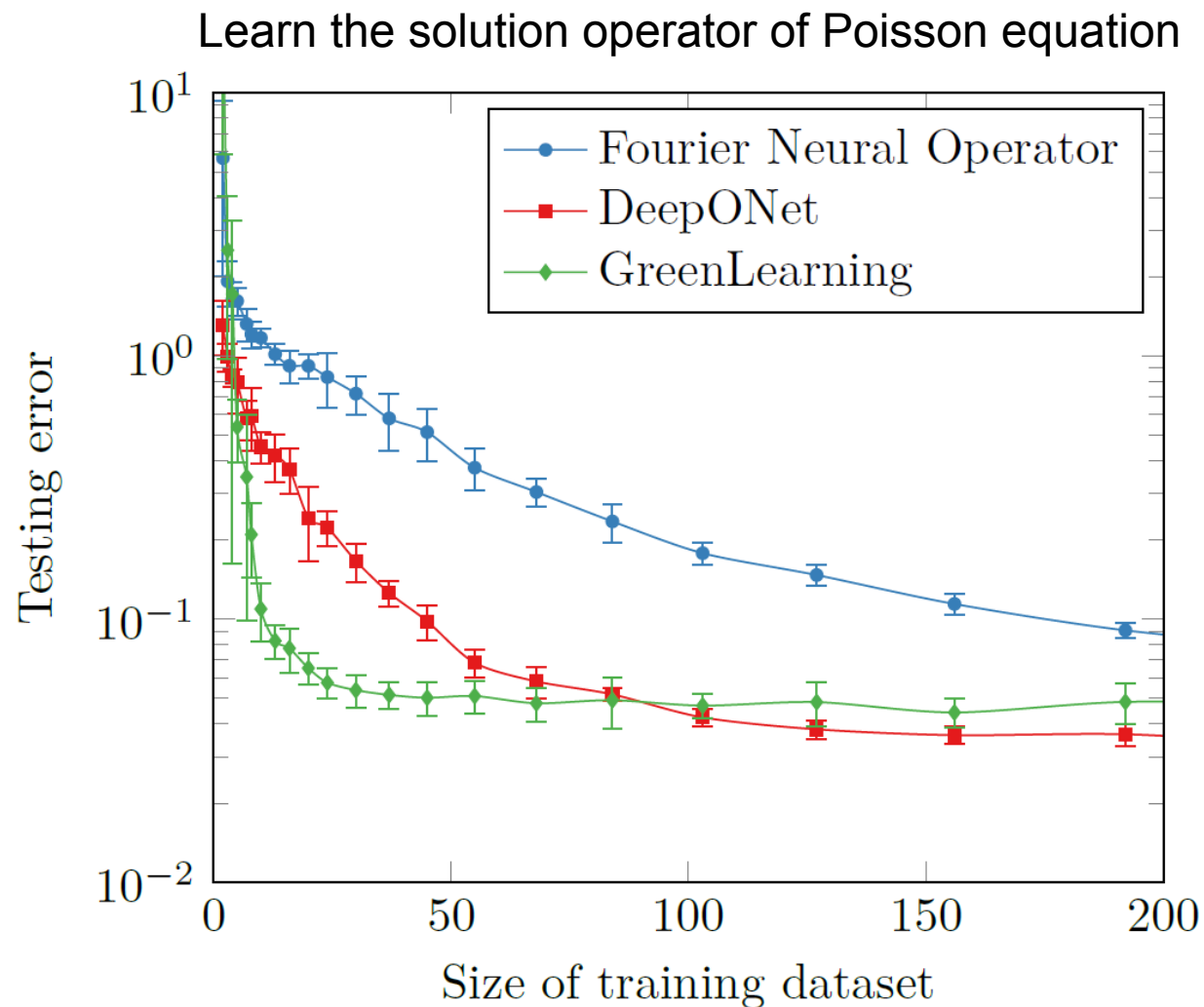


Alex Townsend

[Halikias, Townsend, 2024]

Related works: [Lin et al., 2011], [Gittens, Mahoney, 2016], [Martinsson, 2016], [Tropp et al., 2017], [Nakatsukasa, 2020], [Sun et al., 2021], [Persson et al., 2022], [Cortinovis, Kressner, 2022], [B., Halikias, Otto, Townsend, 2024], [Amsel et al., 2024], ...

Operator learning and sample complexity



How much training data is needed to learn a PDE?

Related works: [B., Townsend, 2022], [B. et al., 2022], [Chen et al., 2023], [de Hoop et al., 2021], [Lu et al., 2021], [Schäfer, Owhadi, 2021], [Schäfer et al., 2017]

Green's functions

Linear differential equation:

$$\begin{array}{l} \mathcal{L}u = f \\ u|_{\partial D} = 0 \end{array} \longrightarrow u(x) = \int_D G(x, y) f(y) \, dy$$

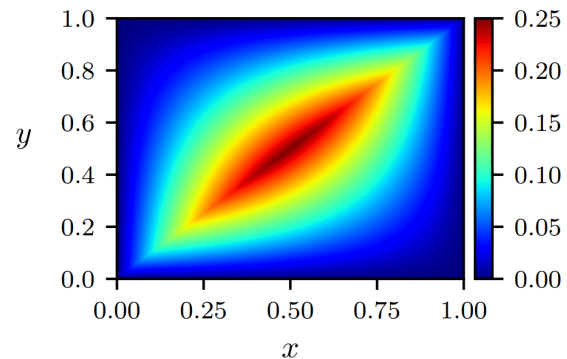
Green's functions

Linear differential equation:

$$\begin{array}{l} \mathcal{L}u = f \\ u|_{\partial D} = 0 \end{array} \longrightarrow u(x) = \int_D G(x, y) f(y) \, dy$$

Poisson equation

$$\begin{array}{l} -\nabla^2 u = f \\ u(0) = u(1) = 0 \end{array}$$



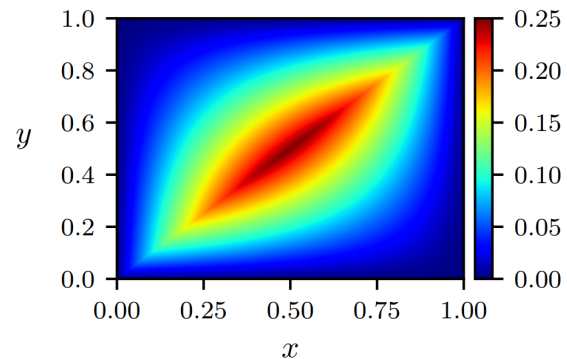
Green's functions

Linear differential equation:

$$\begin{array}{l} \mathcal{L}u = f \\ u|_{\partial D} = 0 \end{array} \longrightarrow u(x) = \int_D G(x, y) f(y) \, dy$$

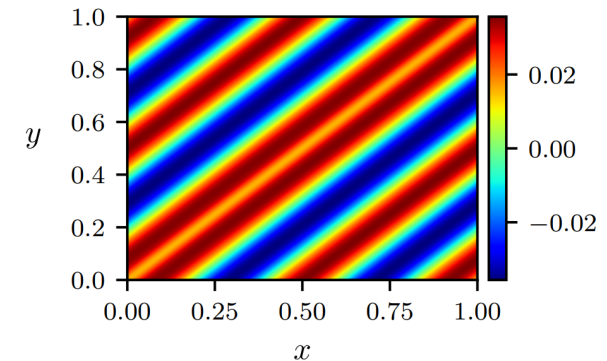
Poisson equation

$$\begin{array}{l} -\nabla^2 u = f \\ u(0) = u(1) = 0 \end{array}$$



Helmholtz equation

$$\begin{array}{l} \nabla^2 u + k^2 u = f \\ u(0) = u(1) \end{array}$$



Learning Green's functions of elliptic PDEs

Elliptic PDEs in 3D of the form:

$$\mathcal{L}u := -\nabla \cdot (A(x)\nabla u) = f \quad \longrightarrow \quad u(x) = \int_D G(x, y) f(y) \, dy$$

Theorem (B., Halikias, Townsend, 2023).

*There is a randomized algorithm that achieves **exponential convergence** for learning the Green's function with respect to the number of pairs (f_j, u_j) with high probability of success.*

The proof combines core techniques in numerical linear algebra and generalizes them to **infinite dimensions**: randomized SVD + hierarchical matrices.

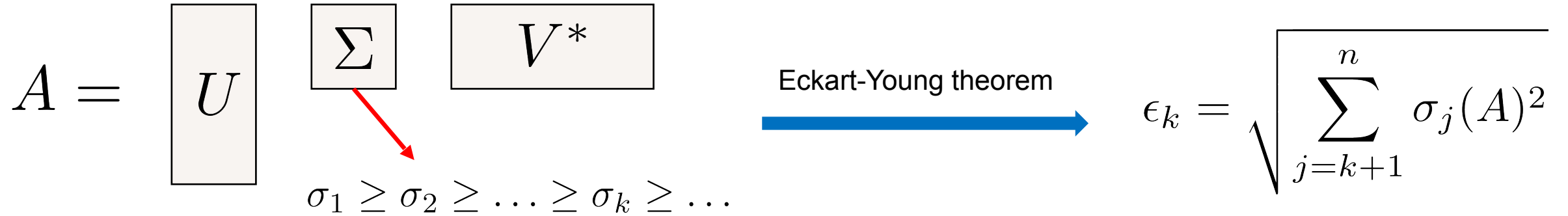
Randomized singular value decomposition

Best rank k approximation

$$A = \begin{bmatrix} U \end{bmatrix} \begin{bmatrix} \Sigma \end{bmatrix} \begin{bmatrix} V^* \end{bmatrix}$$

$\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_k \geq \dots$

Eckart-Young theorem

$$\epsilon_k = \sqrt{\sum_{j=k+1}^n \sigma_j(A)^2}$$
The diagram illustrates the process of finding the best rank k approximation of a matrix A using its singular value decomposition. It shows the decomposition A = UΣV*, where U and V* are orthogonal matrices and Σ is a diagonal matrix of singular values. A red arrow points from the Σ matrix to the sequence of singular values σ1 ≥ σ2 ≥ ... ≥ σk ≥ A blue arrow labeled "Eckart-Young theorem" points from this sequence to the formula for the approximation error εk, which is the square root of the sum of the squares of the singular values from the (k+1)th to the nth.

Randomized singular value decomposition

Best rank k approximation

$$A = \begin{bmatrix} U \end{bmatrix} \begin{bmatrix} \Sigma \end{bmatrix} \begin{bmatrix} V^* \end{bmatrix} \xrightarrow{\text{Eckart-Young theorem}} \epsilon_k = \sqrt{\sum_{j=k+1}^n \sigma_j(A)^2}$$

$\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_k \geq \dots$

Algorithm 1 Randomized singular value decomposition.

- 1: Draw a random matrix $X \in \mathbb{R}^{N \times k}$ with i.i.d. standard Gaussian entries.
 - 2: Perform k queries with A : $Y = AX$.
 - 3: Compute the QR factorization of $Y = QR$.
 - 4: Perform k queries with $A^\top$: $Z = A^\top Q$.
 - 5: Return $A \approx QZ^\top$.
-

Randomized singular value decomposition

Best rank k approximation

$$A = \begin{bmatrix} U \end{bmatrix} \begin{bmatrix} \Sigma \end{bmatrix} \begin{bmatrix} V^* \end{bmatrix}$$

$\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_k \geq \dots$

Eckart-Young theorem

$$\epsilon_k = \sqrt{\sum_{j=k+1}^n \sigma_j(A)^2}$$

Theorem (Halko, Martinsson, Tropp, 2011).

We can construct an approximation A_k of A from $k+5$ *random* input vectors such that

$$\mathbb{P} \left[\|A - A_k\|_F \leq (1 + 15\sqrt{k+5})\epsilon_k \right] \geq 0.999$$

Randomized singular value decomposition

$$A = \begin{bmatrix} U \end{bmatrix} \begin{bmatrix} \Sigma \end{bmatrix} \begin{bmatrix} V^* \end{bmatrix}$$

$\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_k \geq \dots$

Eckart-Young theorem

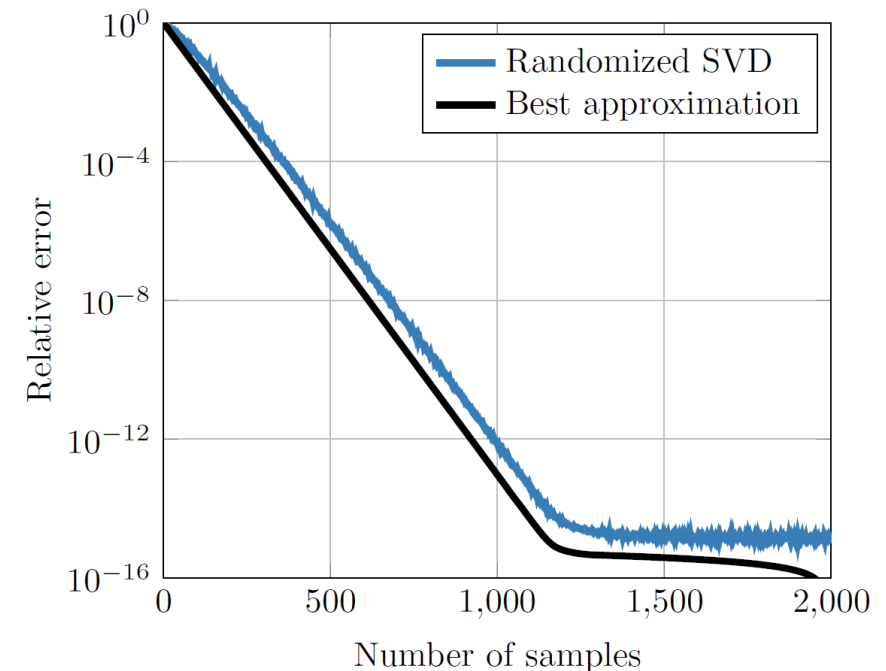
Best rank k approximation

$$\epsilon_k = \sqrt{\sum_{j=k+1}^n \sigma_j(A)^2}$$

Theorem (Halko, Martinsson, Tropp, 2011).

We can construct an approximation A_k of A from $k+5$ *random* input vectors such that

$$\mathbb{P} \left[\|A - A_k\|_F \leq (1 + 15\sqrt{k+5})\epsilon_k \right] \geq 0.999$$



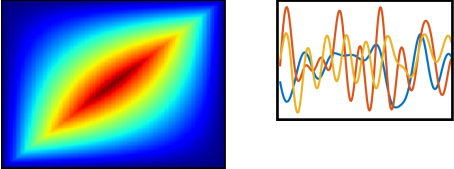
Randomized SVD in infinite-dimensions

$$A \times F$$

A	0.7 0.1
	-0.4 0.2
	0.8 0.5

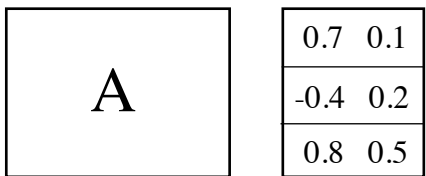
[B., Townsend, 2022]



$$\int_D G(x, y) f(y) \, dy$$
Two small plots are shown below the integral. The left plot is a heatmap representing a kernel function G(x, y), showing a bright, elongated elliptical region of high intensity (red/yellow) against a dark blue background. The right plot is a line graph showing two oscillating functions, one in orange and one in blue, representing f(y).

Randomized SVD in infinite-dimensions

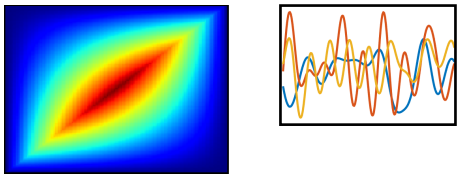
$$A \times F$$



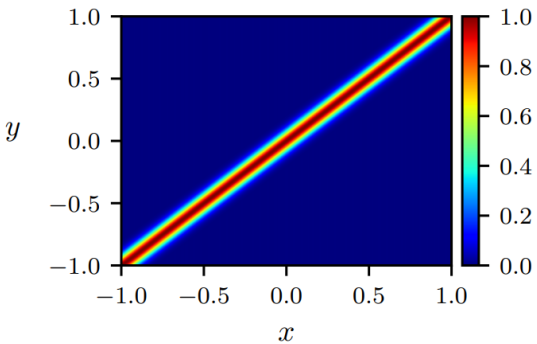
[B., Townsend, 2022]



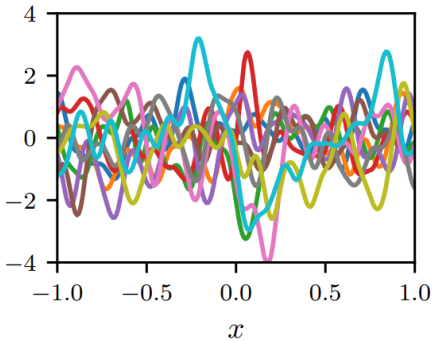
$$\int_D G(x, y) f(y) \, dy$$



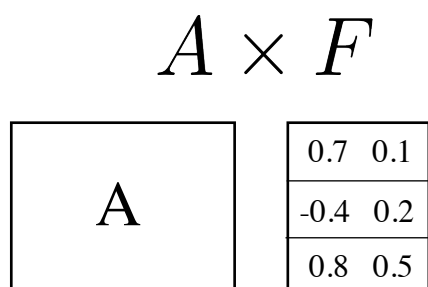
Covariance
kernel



Random
functions



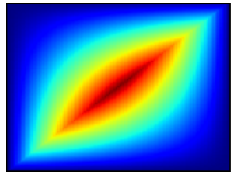
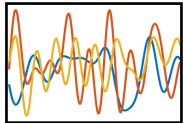
Randomized SVD in infinite-dimensions



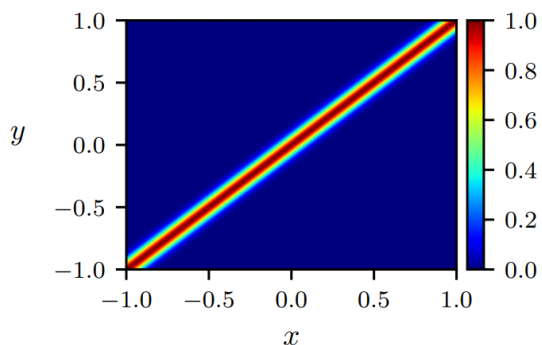
[B., Townsend, 2022]



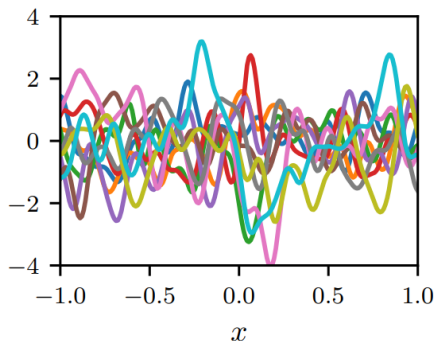
$$\int_D G(x, y) f(y) dy$$

Covariance
kernel



Random
functions



Theorem (B., Townsend, 2022).

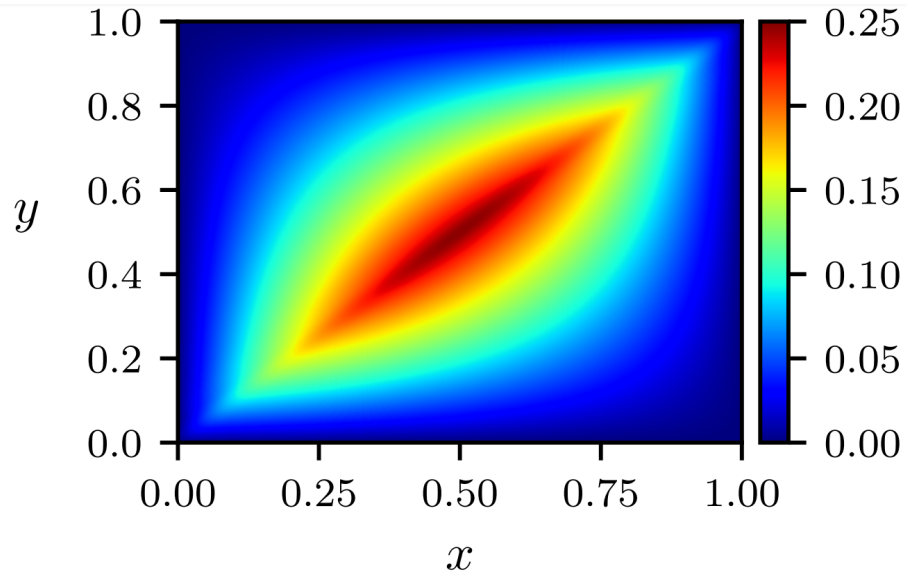
We can construct an approximation G_k of G from $k+5$ *random input functions* f such that

$$\mathbb{P} \left[\|G - G_k\|_{L^2} \leq \mathcal{O} \left(\sqrt{k^2 / \gamma_k} \right) \epsilon_k \right] \geq 0.999$$

ϵ_k contains the tail of the singular values of G .

Randomized SVD for Green's functions

Green's function



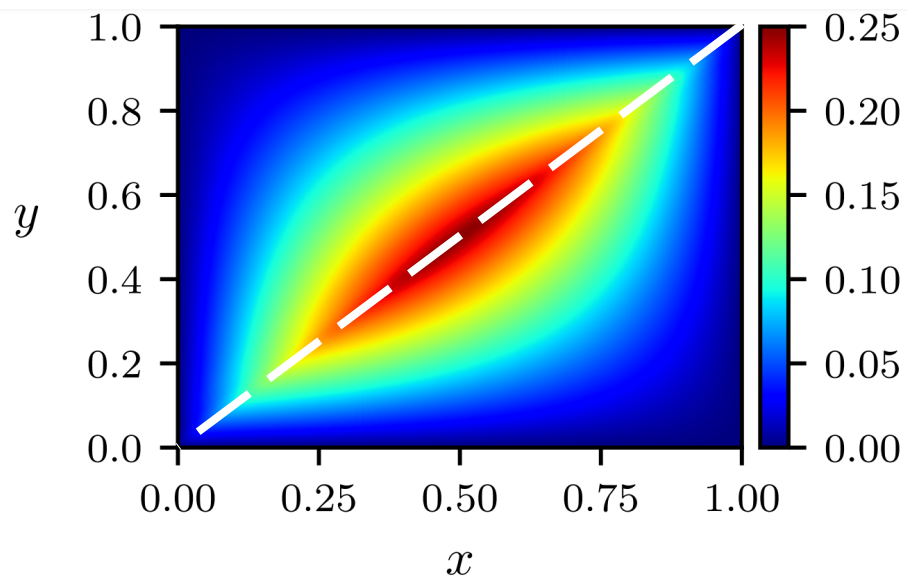
Theorem (B., Townsend, 2022).

We can construct an approximation G_k of G from $k+5$ random input functions f such that

$$\mathbb{P} \left[\|G - G_k\|_{L^2} \leq \mathcal{O} \left(\sqrt{k^2 / \gamma_k} \right) \epsilon_k \right] \geq 0.999$$

Randomized SVD for Green's functions

Green's function



Theorem (B., Townsend, 2022).

We can construct an approximation G_k of G from $k+5$ random input functions f such that

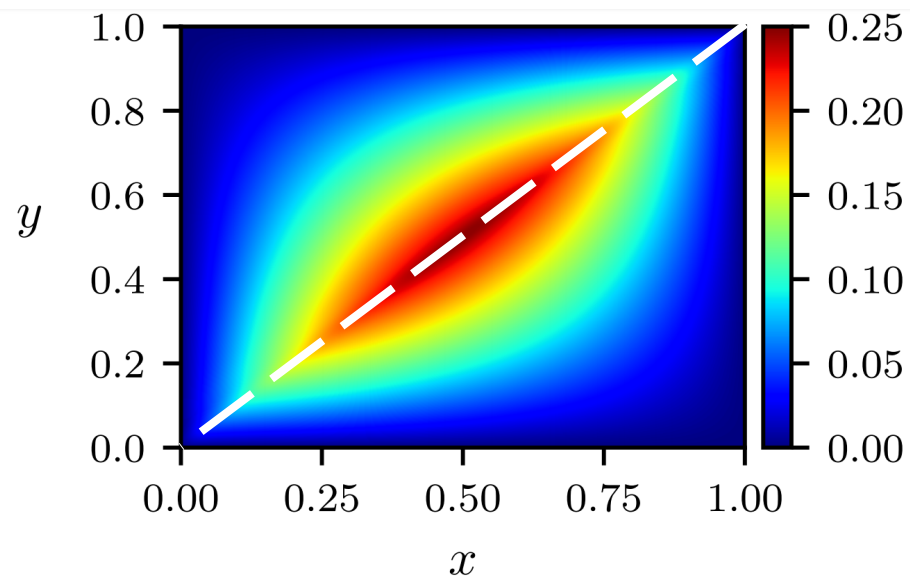
$$\mathbb{P} \left[\|G - G_k\|_{L^2} \leq \mathcal{O} \left(\sqrt{k^2 / \gamma_k} \right) \epsilon_k \right] \geq 0.999$$

Problem:

The Green's functions are not smooth near the diagonal.

Randomized SVD for Green's functions

Green's function



Theorem (B., Townsend, 2022).

We can construct an approximation G_k of G from $k+5$ random input functions f such that

$$\mathbb{P} \left[\|G - G_k\|_{L^2} \leq \mathcal{O} \left(\sqrt{k^2 / \gamma_k} \right) \epsilon_k \right] \geq 0.999$$

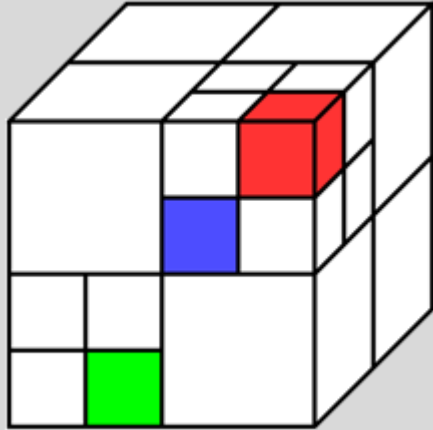
Problem:

The Green's functions are not smooth near the diagonal.



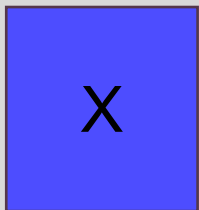
ϵ_k decays very slowly with k

Low-rank structure

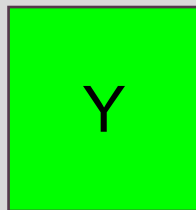


Low-rank structure on well separated domains.

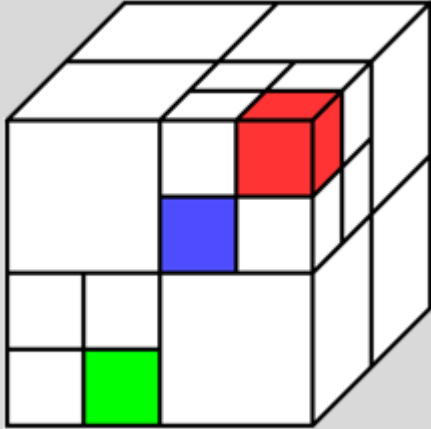
[Bebendorf, Hackbush, 2003]



$\times$

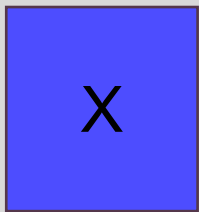


Low-rank structure

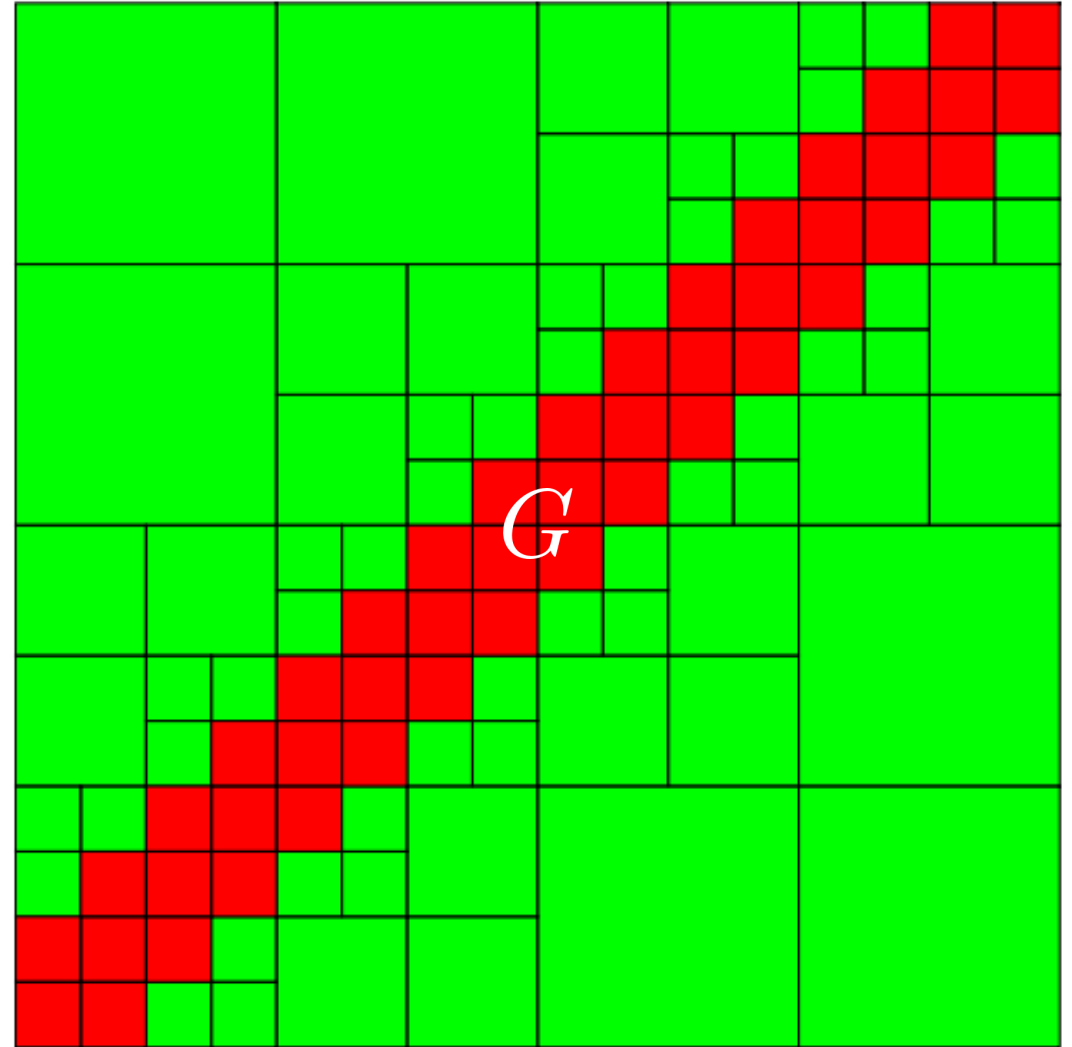
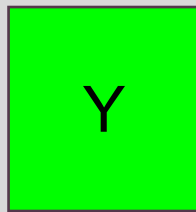


Low-rank structure on well separated domains.

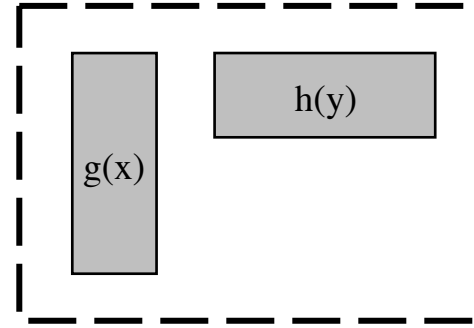
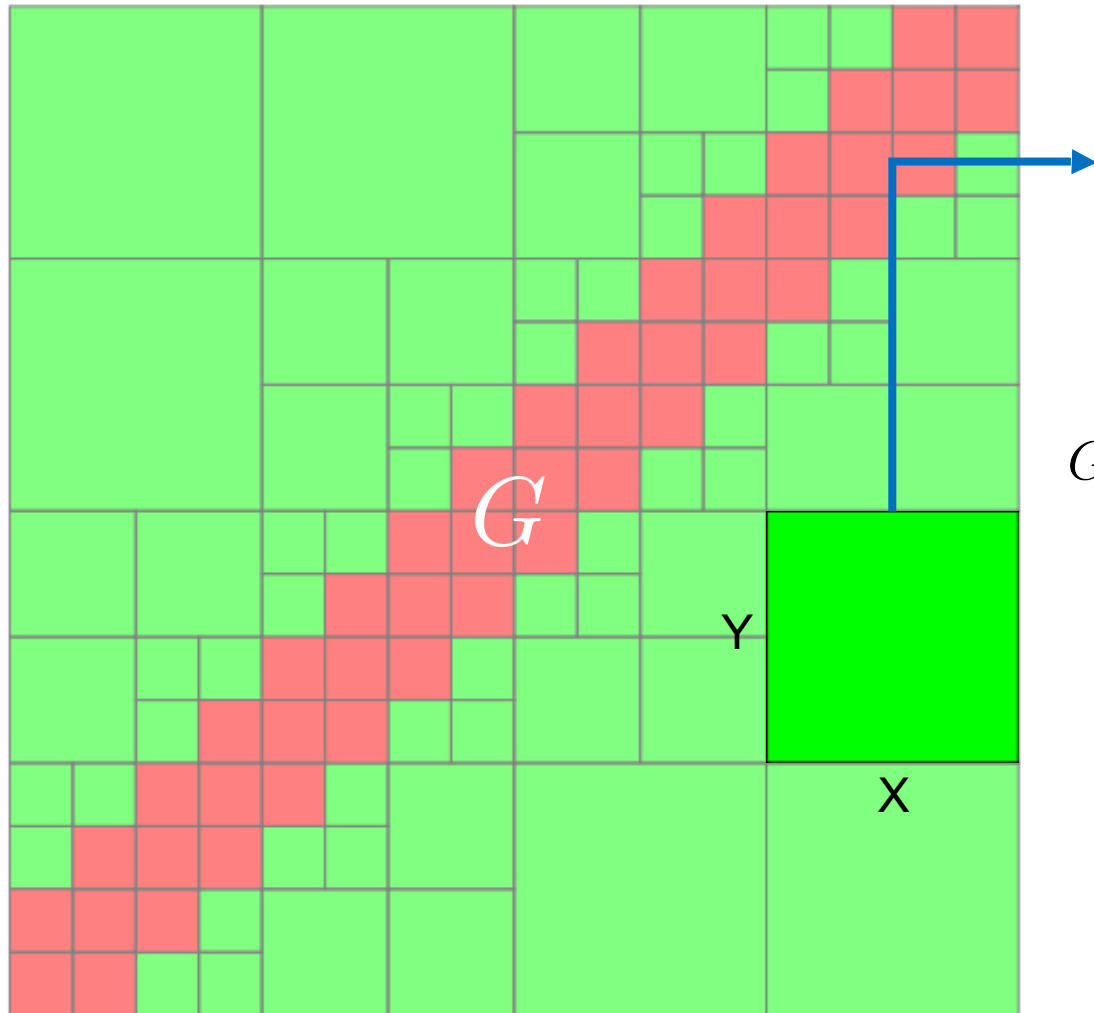
[Bebendorf, Hackbush, 2003]



$\times$

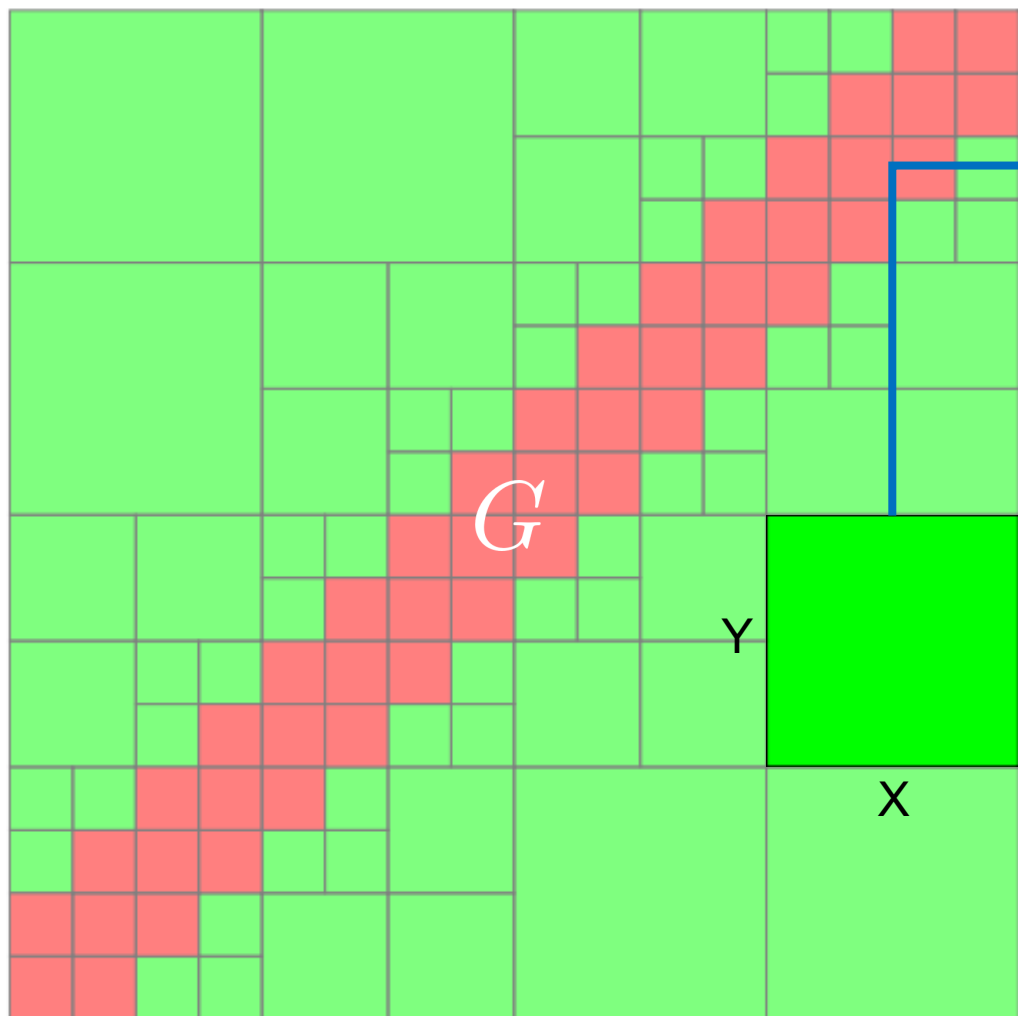


Hierarchical reconstruction



$$G(x, y) \approx \sum_{j=1}^k g_j(x) h_j(y)$$

Hierarchical reconstruction

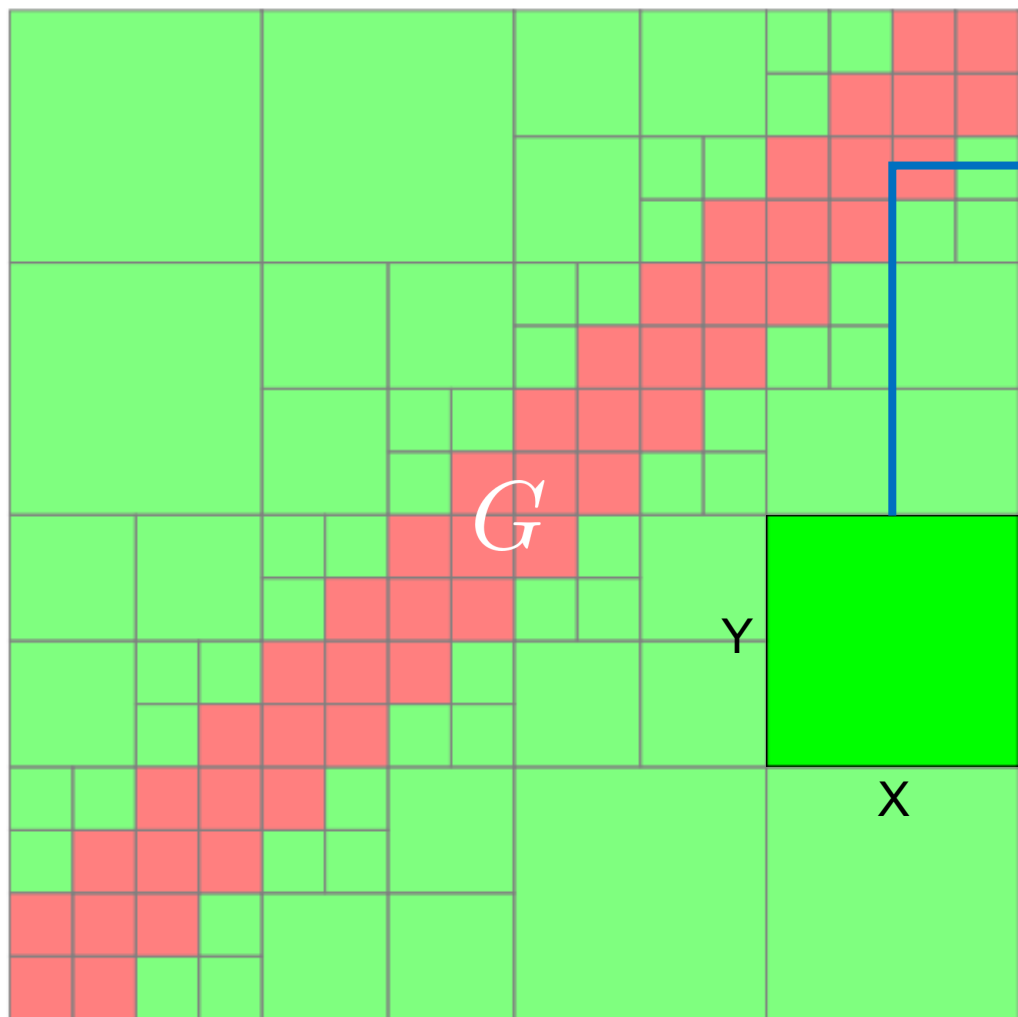


Randomized SVD on each subdomain

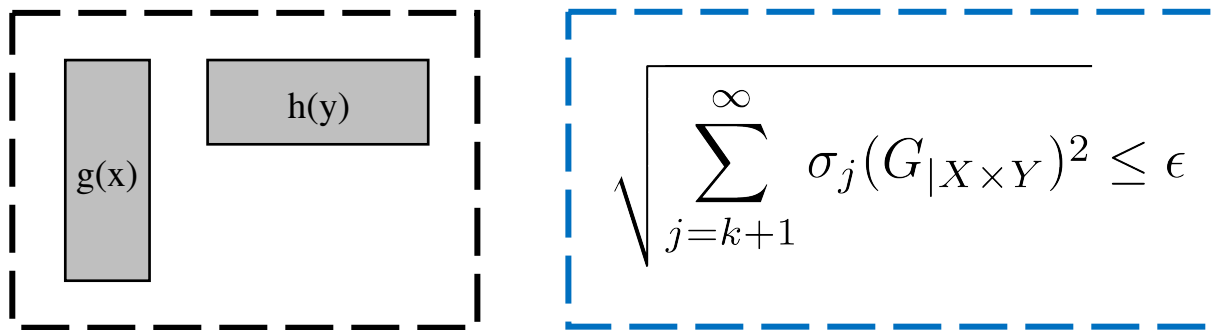
$$G(x, y) \approx \sum_{j=1}^k g_j(x) h_j(y)$$

$$\sqrt{\sum_{j=k+1}^{\infty} \sigma_j(G|_{X \times Y})^2} \leq \epsilon$$

Hierarchical reconstruction



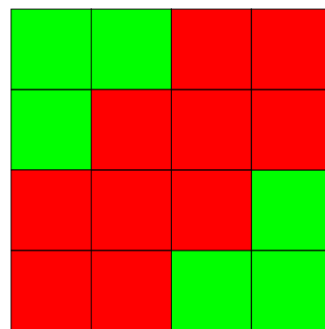
Randomized SVD on each subdomain



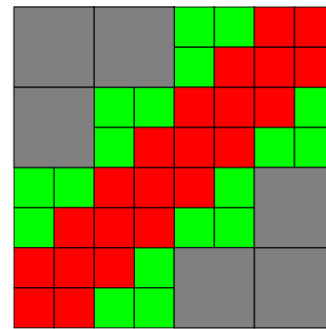
$$G(x, y) \approx \sum_{j=1}^k g_j(x) h_j(y)$$

Hierarchical structure

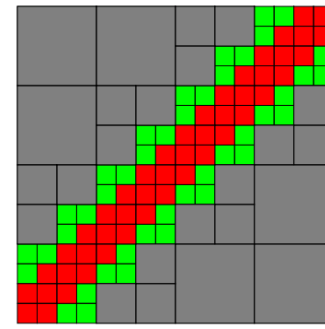
Level 2



Level 3



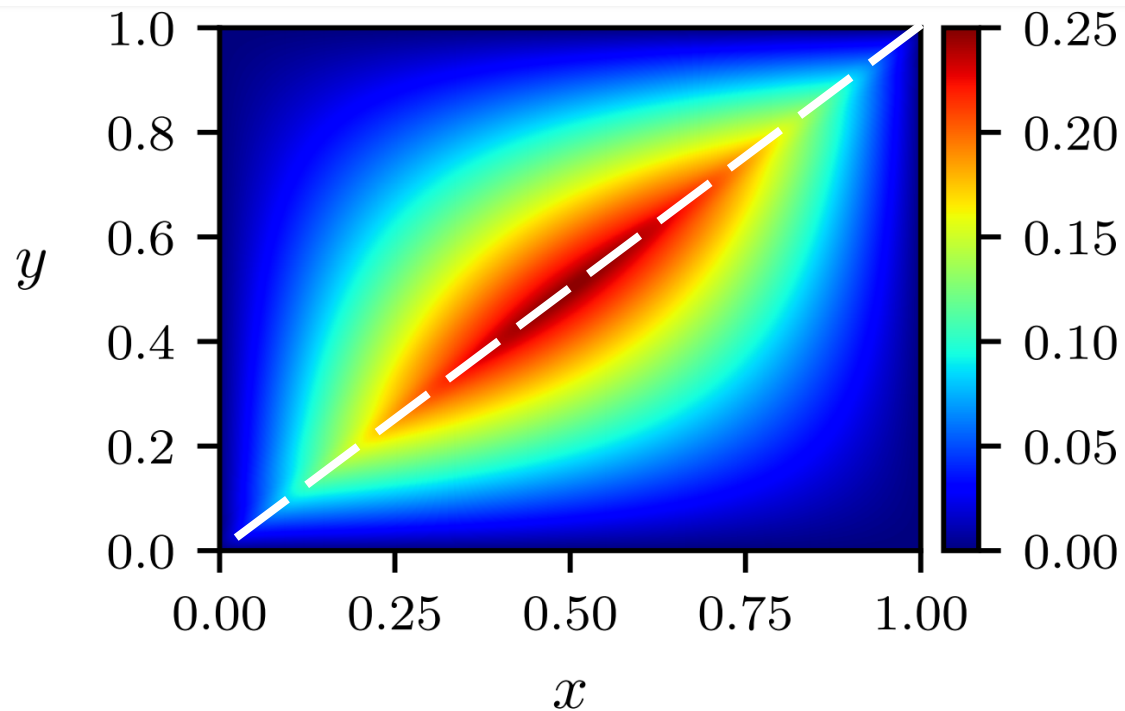
Level 4



Off-diagonal decay

Green's function of the Laplace operator:

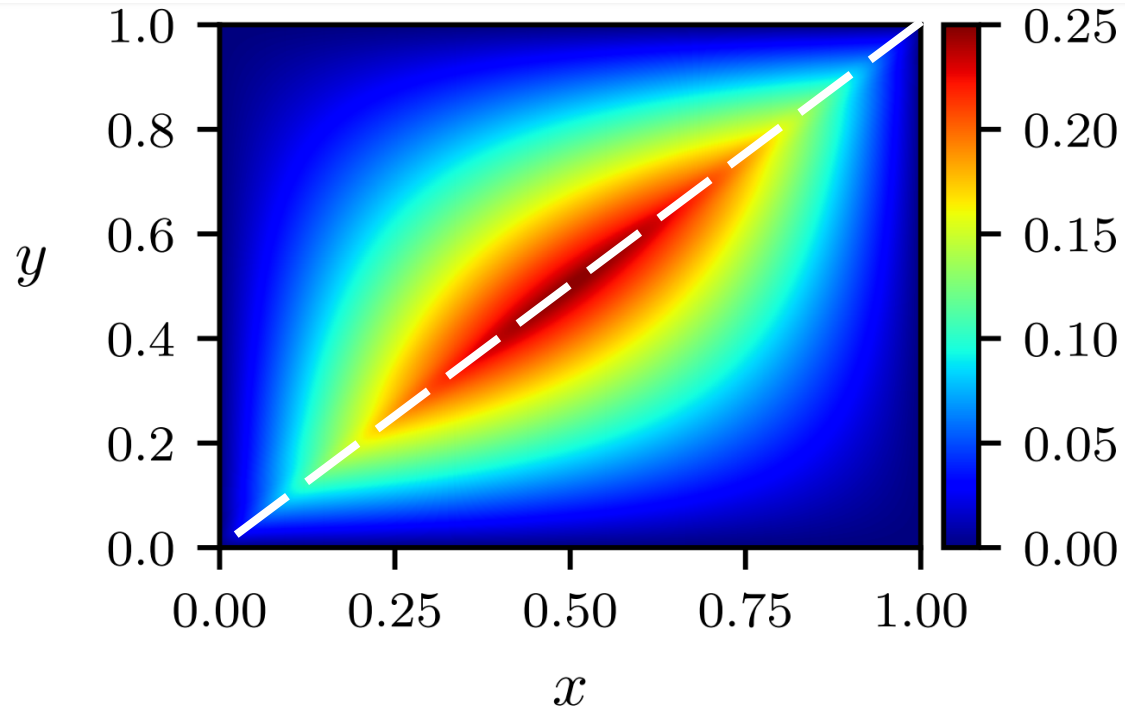
$$-\nabla^2 u = f$$



Off-diagonal decay

Green's function of the Laplace operator:

$$-\nabla^2 u = f$$



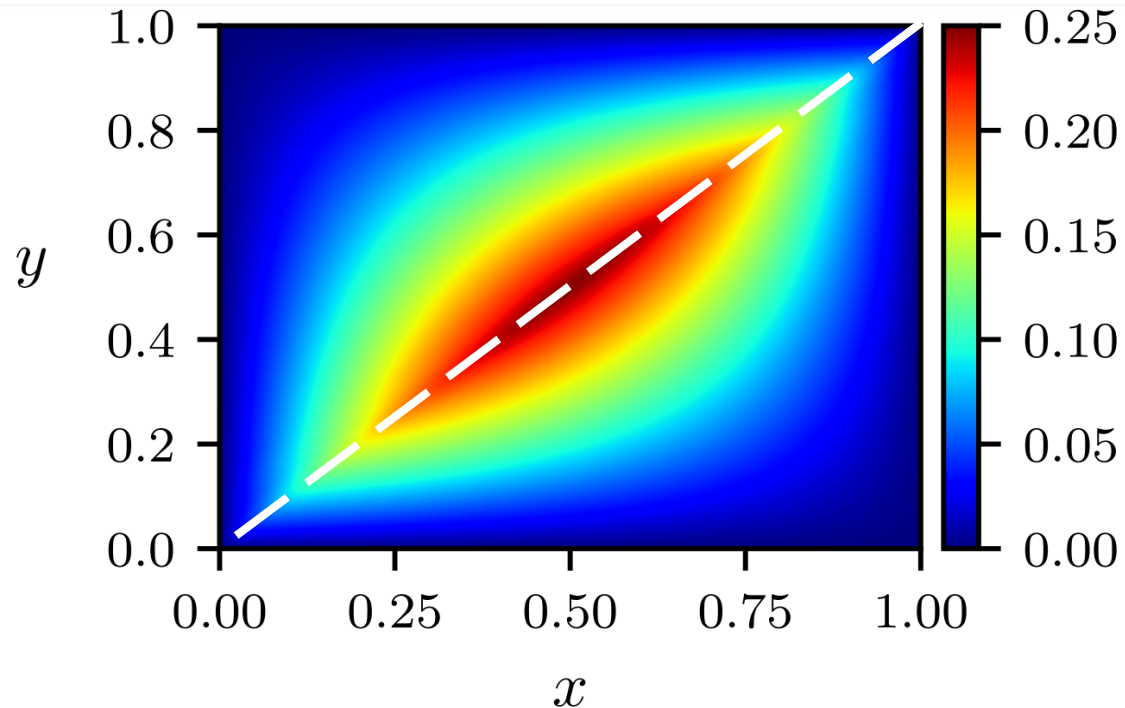
Green's functions are smooth and decay off the diagonal. [Grüter, Widman, 1982]

$$G(x, y) \leq \frac{1}{\|x - y\|}$$

Off-diagonal decay

Green's function of the Laplace operator:

$$-\nabla^2 u = f$$

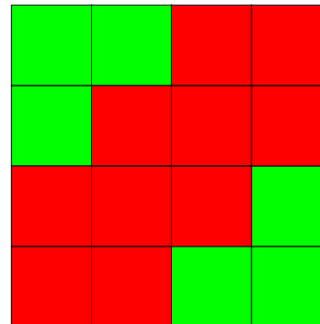


Green's functions are smooth and decay off the diagonal. [Grüter, Widman, 1982]

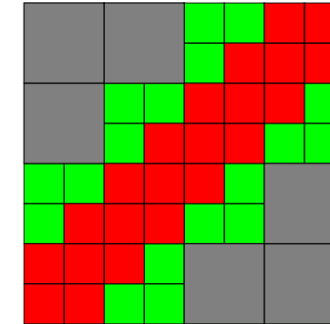
$$G(x, y) \leq \frac{1}{\|x - y\|}$$

Hierarchical structure

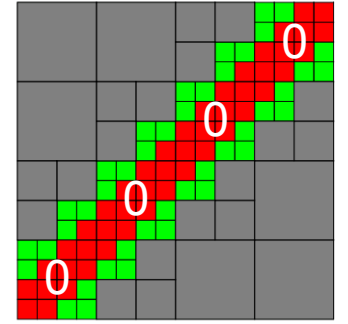
Level 2



Level 3



Level 4



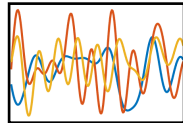
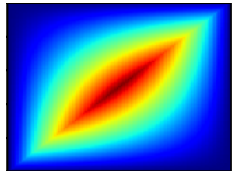
Summary of the result

Theorem (B., Halikias, Townsend, 2023).

*There is a randomized algorithm that achieves **exponential convergence** for learning the Green's function with respect to the number of pairs (f_j, u_j) with high probability of success.*

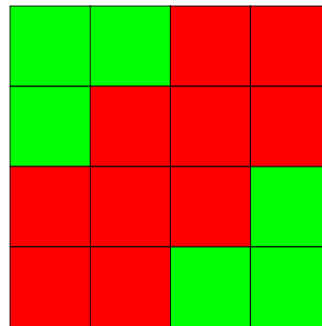
Randomized SVD

$$\int_D G(x, y) f(y) dy$$

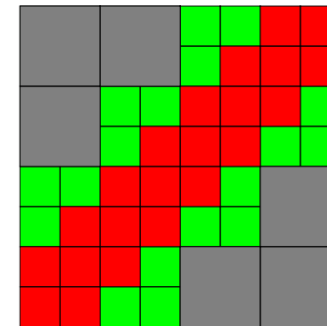


Hierarchical structure

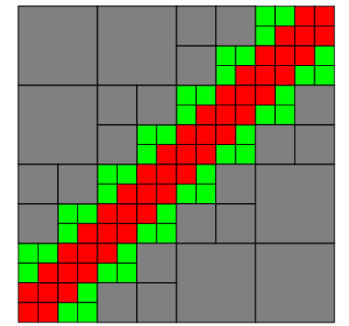
Level 2



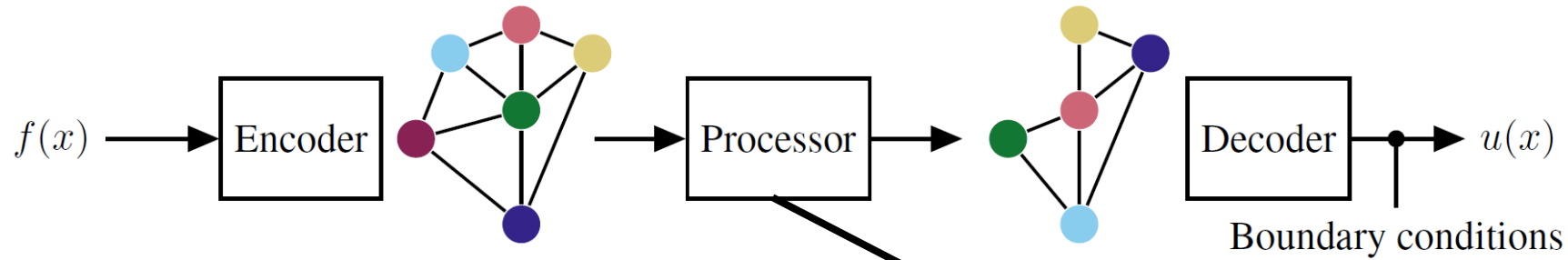
Level 3



Level 4

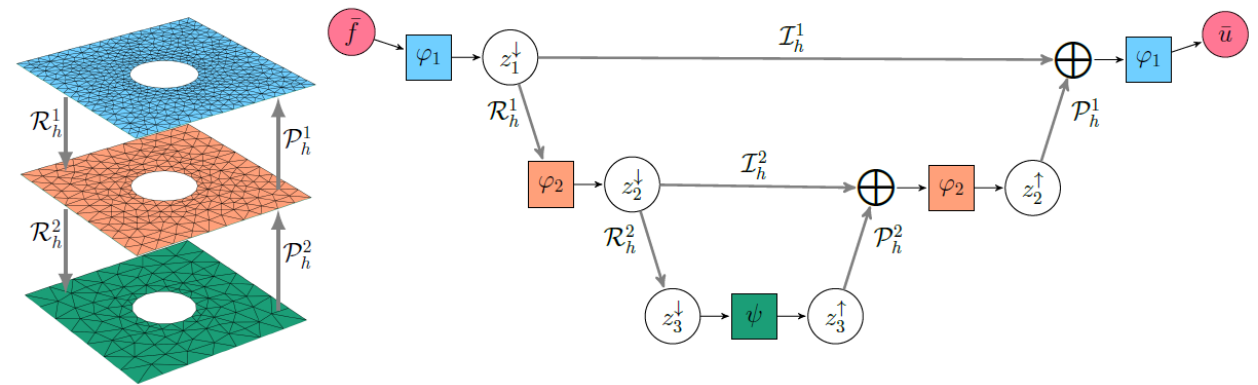


Towards multi-scale architectures



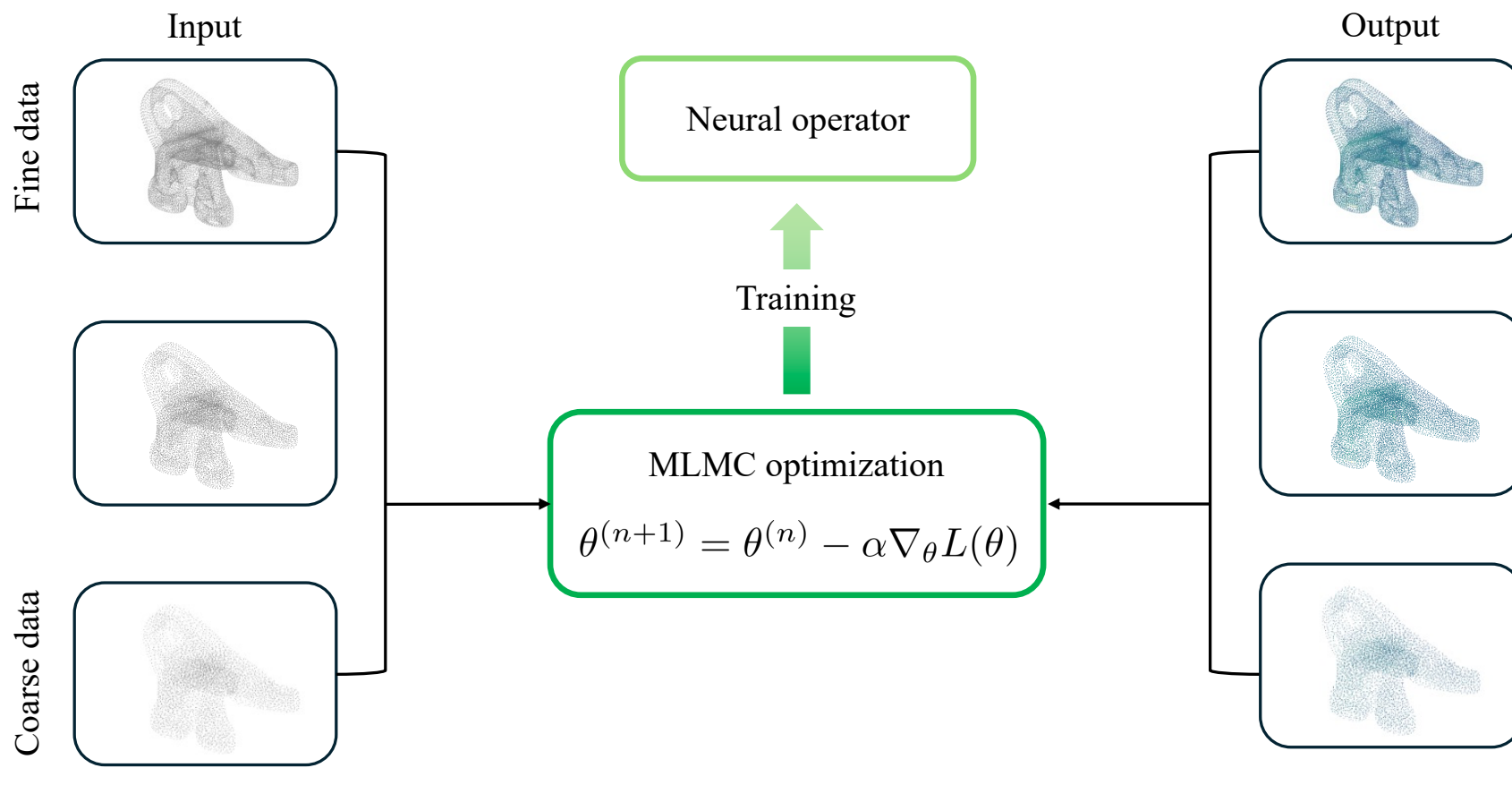
Nacime Bouziani

- Finite element encoder / decoder
- Structure-preserving discretization
- Multigrid-based processor



Towards multi-level training

Multi-Level Monte-Carlo approach for training neural operators



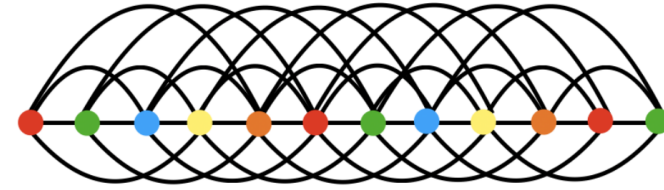
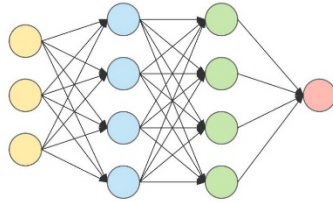
James Rowbottom



Stefania Fresca

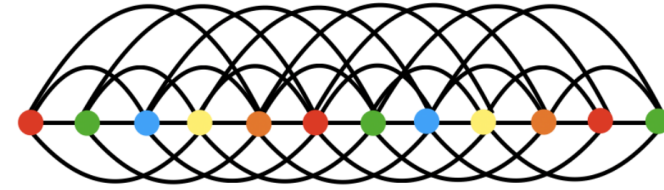
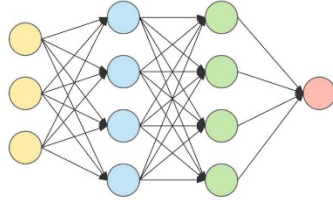
Conclusions

1. Choice of neural operator architectures



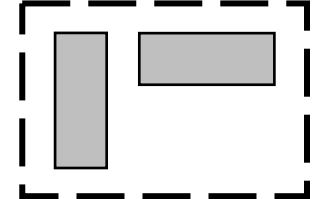
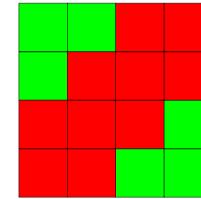
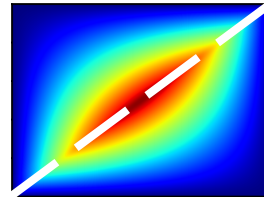
Conclusions

1. Choice of neural operator architectures



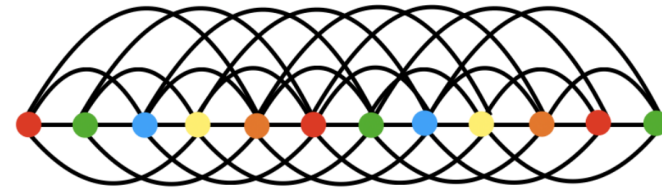
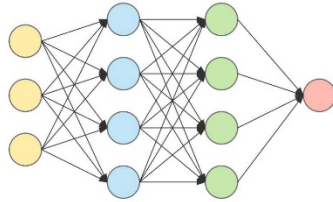
2. Theoretical understanding

$$\mathcal{L}u = -\nabla \cdot (A(x)\nabla u)$$



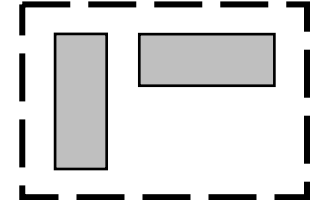
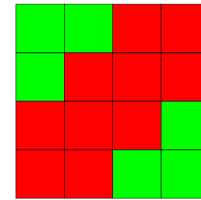
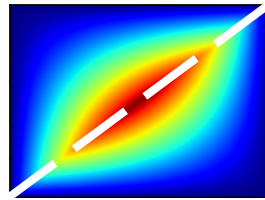
Conclusions

1. Choice of neural operator architectures



2. Theoretical understanding

$$\mathcal{L}u = -\nabla \cdot (A(x)\nabla u)$$



3. Interpretability in operator learning

