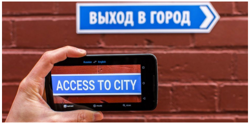
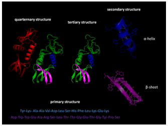
Deep Learning and Computations of PDEs

Siddhartha Mishra

Computational and Applied Mathematics Laboratory (CamLab)
Seminar for Applied Mathematics (SAM), D-MATH (and),
ETH AI Center,
ETH Zürich, Switzerland.

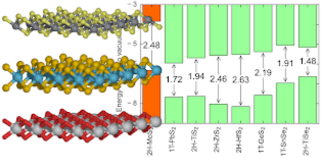
The age of Machine Intelligence



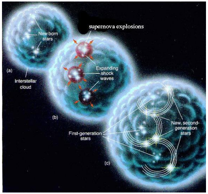


- Deep Neural Networks + Big Data
- Can Deep Learning impact Scientific Computing ?
- Particularly for Models described by PDEs

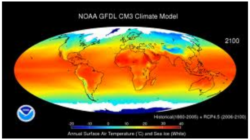
Partial Differential Equations (PDEs)



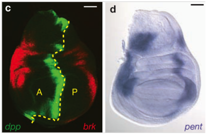
Atoms



Stars

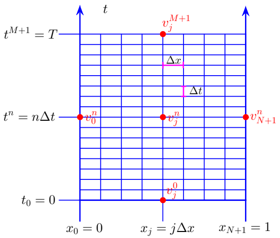


Climate

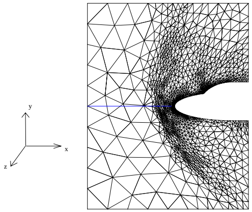


Fly wing

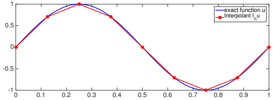
Traditional Numerical Methods



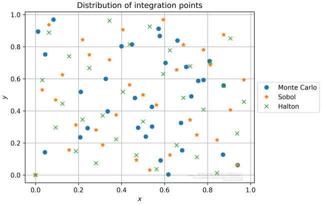
Finite Difference



Finite Volume



Finite Element



Collocation

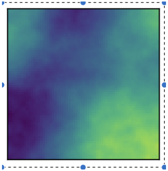
- Different flavors of Runge-Kutta for Time Integration

What do Numerical Methods do ?

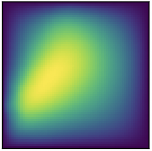
- Example 1: Consider Darcy PDEs:

$$-\mathrm{div}(a\nabla u) = f,$$

- Quantities of interest are:
  - u is temperature or pressure.
  - a is conductance or permeability.
  - f is the source.



a



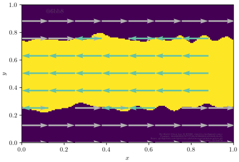
u

- FEM approximates the solution Operator $\mathcal{G} : a \mapsto \mathcal{G}a = u$.

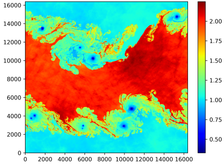
What do Numerical Methods do ?

- Example 2: Consider the Compressible Euler equations:

$$
\begin{aligned}
\rho_t + \mathrm{div}(\rho \mathrm{v}) &= 0, \\
(\rho \mathrm{v})_t + \mathrm{div}\left(\rho \mathrm{v} \otimes \mathrm{v} + p\mathrm{I}\right) &= 0, \\
E_t + \mathrm{div}\left((E + p)\mathrm{v}\right) &= 0., \\
u(x,0) = (\rho, \mathrm{v}, E)(x,0) &= a(x).
\end{aligned}
$$



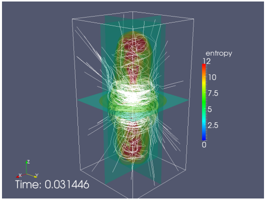
Initial Condition



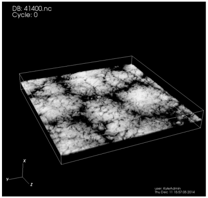
Solution at time T

- FVM approximates the solution Operator $\mathcal{G} : a \mapsto \mathcal{G}a = u$.
- Numerical methods approximates Operators !!

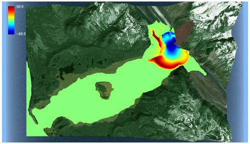
Numerical Methods are very Successful



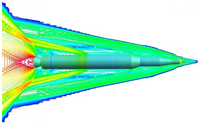
Supernovas



Clouds

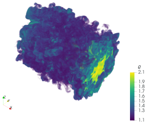


Tsunamis

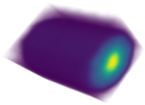


Rockets

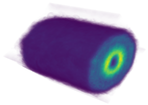
Issues with Traditional Numerical Methods



Sample



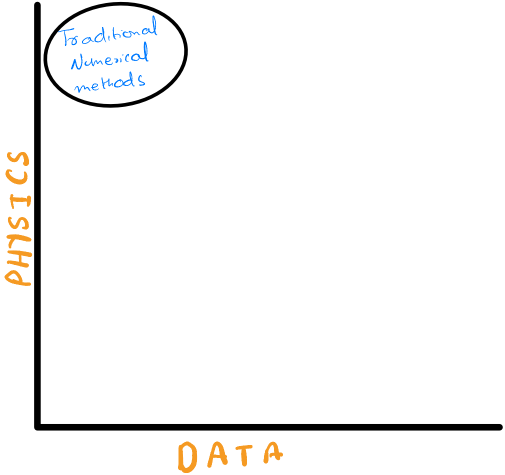
Mean



Variance

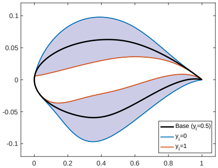
- Computational Cost !!: PDE solvers are very expensive.
- Especially for Many-Query Problems:
- UQ, Optimal Design, Inverse Problems.
  - Simulation of Compressible Flow with ALSVINN
  - 300 NH for a 1024^3 run on PIZ DAINT (15th in Top500)
  - Ensemble simulation costs $\approx$ 1 Mil USD
- Trad. PDE solvers are Data Agnostic (see SM, 2018).
- AIM: Combine Data + Physics to approximate PDEs.

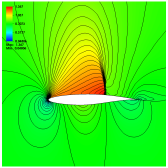
Data vs. Physics



Setup

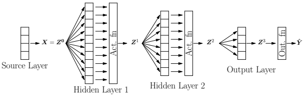
- X, Y are Banach spaces and $\mu \in \mathrm{Prob}(X)$
- Abstract PDE: $\mathcal{D}_a(u) = f$
- Solution Operator: $\mathcal{G} : X \mapsto Y$ with $\mathcal{G}(a, f) = u$
- Simplified Setting: $\dim(\mathrm{Supp}(\mu)) = d_y < \infty$
- Corresponds to Parametrized PDEs with finite parameters.
- Find Soln $u(t, x, y)$ or Observable $\mathcal{L}(y)$ for $y \in Y \subset \mathbb{R}^{d_y}$.





- Approximate Fields or observables with deep neural networks

Supervised learning of target $\mathcal{L}$ with Deep Neural networks



- $\mathcal{L}^*(z) = \sigma_o \odot C_K \odot \sigma \odot C_{K-1} \ldots\ldots\ldots \odot \sigma \odot C_2 \odot \sigma \odot C_1(z)$.
- At the k-th Hidden layer: $z^{k+1} := \sigma(C_k z^k) = \sigma(W_k z^k + B_k)$
- Tuning Parameters: $\theta = \{W_k, B_k\} \in \Theta$,
- σ: scalar Activation function: ReLU, Tanh
- Random Training set: $\mathcal{S} = \{z_i\}_{i=1}^N \in Z$, with i.i.d z_i
- Use SGD (ADAM) to find $\mathcal{L} \approx \mathcal{L}^* = \mathcal{L}^*_{\theta^*}$

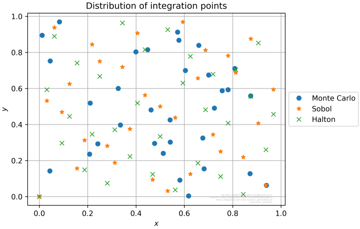
$$\theta^* := \arg\min_{\theta \in \Theta} \sum_{i=1}^{N} |\mathcal{L}(z_i) - \mathcal{L}^*_\theta(z_i)|^p,$$

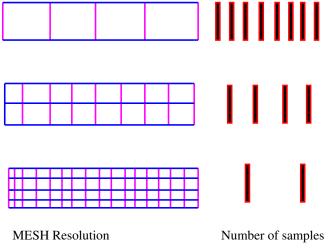
Can this work ?

- For any Measurable $\mathcal{L} : Z \subset \mathbb{R}^{d_z} \mapsto \mathbb{R}^m$.
- Universal Approximation: $\exists$ DNN $\mathcal{L}^*$ s.t $\|\mathcal{L} - \mathcal{L}^*\| < \epsilon$
- Total Error: $\mathcal{E} = \|\mathcal{L} - \mathcal{L}^*\|_p \leq \mathcal{E}_{app} + \mathcal{E}_{gen} + \mathcal{E}_{train}$.
- If $\mathcal{L} \in W^{s,p}(Z)$ and $\mathcal{E}_{app} \sim \epsilon \Rightarrow \mathrm{Size}(\mathcal{L}^*) \sim \epsilon^{-\frac{d_y}{s}}$
- Curse of dimensionality
- $\mathcal{E}_{app} = \|\mathcal{L} - \hat{\mathcal{L}}\|_p \sim \mathcal{O}\left(d_z^{\beta} M^{-\eta}\right)$ with $\beta \lesssim 1, 2$
  - Linear Elliptic PDEs: (Schwab, Kutyniok et al).
  - Semi-linear Parabolic PDEs: (E, Jentzen,Grohs et al).
  - Nonlinear Hyperbolic PDEs: (DeRyck, SM, 2022).
- Statistical Learning Theory $\Rightarrow \mathcal{E}_{gen} \sim \frac{C(M)\log(N)}{\sqrt{N}}$
- Challenge in Scientific Computing: learn in a data poor regime
- Have to stick to Moderate Data i.e, $10^2 - 10^3$ samples.

A bag of tricks from Numerical Analysis

- Train on Low discrepancy sequences: SM, Rusch
- With Sobol pts: $\mathcal{E} \leq \mathcal{E}_T + C\left(V_{HK}(\mathcal{L}), V_{HK}(\mathcal{L}^*)\right)\frac{(\log N)^d}{N}$
- Multi-level Training: Lye, SM, Molinaro

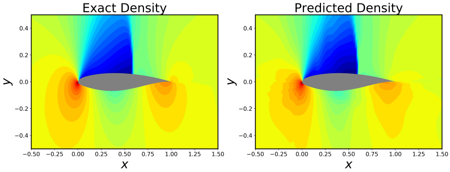




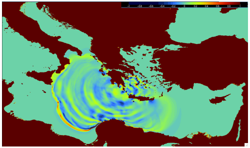
Prediction

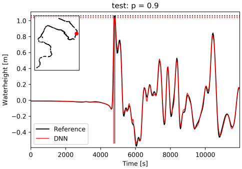
- Given Hicks-Henne parameter: Predict Drag, Lift, Flow
- DNN with $10^3 - 10^4$ parameters and 128 training samples :

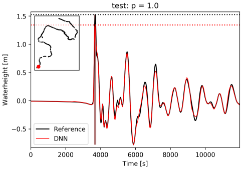
	Run time (1 sample)	Training	Evaluation	Error
Lift	2400 s	700 s	10^{-5} s	0.7%
Drag	2400 s	840 s	10^{-5} s	1.8%
Field	2400 s	1 hr	0.2 s	1.7%



Tsunami TimeSeries Predictions





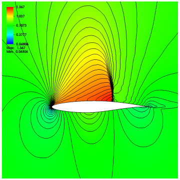


- Modeled by one-layer Shallow water equations
- Okada Model: Seafloor deformation $\Rightarrow$ Initial conditions.
- State of the art Prediction + UQ $\approx$ 60 mins.
- DNN Prediction in < 1 sec (Grosheintz-Laval et al 2022)

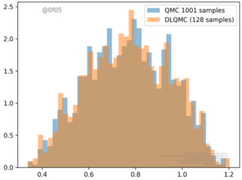
Uncertainty Quantification (UQ)

- DL-UQ algorithm of Lye,SM,Ray, 2020 is

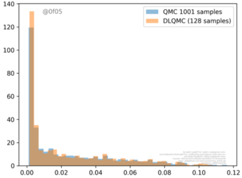
$$\mathcal{L}\#\mu \approx \frac{1}{M}\sum_{i=1}^{M}\delta_{\mathcal{L}(y_i)} \approx \frac{1}{M}\sum_{i=1}^{M}\delta_{\mathcal{L}^*(y_i)}$$



Sample



Lift PDF

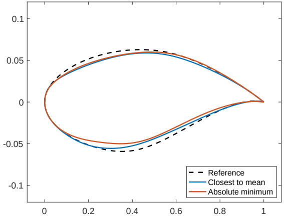


Drag PDF

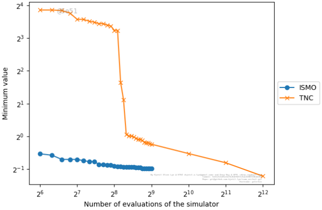
Observable	Speedup (MC)	Speedup (QMC)
Lift	246.02	6.64
Drag	179.54	8.56

- Another factor of $3-5$ gain with Multi-level Training

PDE constrained optimization



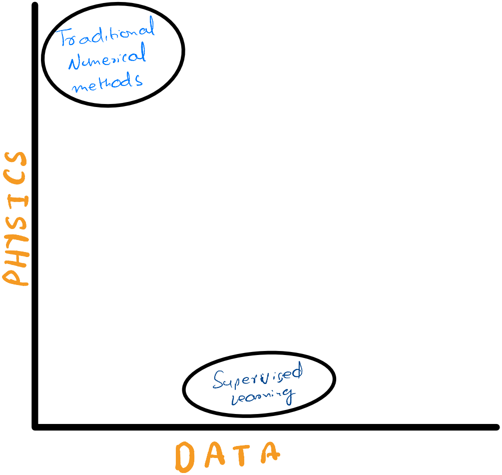
Shapes



ISMO vs. TNC

- Change airfoil shape to Minimize Drag for constant Lift.
- Active learning ISMO: Lye, SM, Ray, Praveen, 2020.
- $>50\%$ Drag reduction at near constant lift.
- Variant works for Bayesian Inversion (Grosheintz Laval et al, 2022).

Data vs. Physics



Caveats

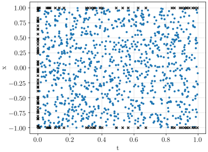
- Supervised Learning worked with Moderate Data.
- Availability of PDE solvers with reasonable computational cost for 1 solve.
- Many situations with NO such solvers:
  - Domains with complex geometries (Difficult Grid Generation)
  - Multi-scale, Multi-physics systems.
  - PDEs with High spatial dimensions:
    - Boltzmann Equation ($d = 7$)
    - Radiative transport Equation ($d \geq 5$)
    - Computational Finance: Black-Scholes ($d >> 1$)
    - Computatational Chemistry: Schrödinger ($d >> 1$).
    - Multi-agent systems.
- Need Unsupervised Learning in this setting.

Physics Informed Neural Networks

- Variants of PINNs stem from Dissanayake, Phan-Thien, 1994.
- Also in Lagaris et al, mid 1990s.
- Reintroduced by Raissi, Perdikaris, Karniadakis, 2017.
- Extensively developed by Karniadakis and collaborators.
- 100s of papers on PINNs already.
- Our Aim: Elucidate possible mechanisms to explain why PINNs work

Heat Eqn: $u_t = u_{xx}$ with 0-BC and $u(x,0) = \bar{u}(x)$ IC

- Training Set: $\mathcal{S} = \mathcal{S}_{int} \cup \mathcal{S}_{tb} \cup \mathcal{S}_{sb}$ Randomly chosen.



- Deep Neural networks : $(x,t) \mapsto u_\theta(x,t)$, $\theta \in \Theta$.
- Temporal boundary residual: $\mathcal{R}_{tb,\theta} = u_\theta(\cdot, 0) - \bar{u}$
- Spatial boundary residual: $\mathcal{R}_{sb,\theta} = u_\theta|_{\partial D}$.
- Interior PDE Residual: $\mathcal{R}_{int,\theta} = \partial_t u_\theta - \partial_{xx} u_\theta$
- Evaluate PDE Residual by Automatic Differentiation
- Loss function:

$$J = \frac{1}{N_{tb}} \sum_{n=1}^{N_{tb}} |\mathcal{R}_{tb,\theta}(x_n)|^2 + \frac{1}{N_{sb}} \sum_{n=1}^{N_{sb}} |\mathcal{R}_{sb,\theta}(x_n, t_n)|^2 + \frac{1}{N_{int}} \sum_{n=1}^{N_{int}} |\mathcal{R}_{int,\theta}|^2.$$

Why PINNs are great ?: I

- Very easy to Code !!
- A few lines in PyTorch or TensorFlow

```
def compute_res(self, network, x_f_train):
    x_f_train.requires_grad = True
    u = network(x_f_train).reshape(-1, )
    grad_u = torch.autograd.grad(u, x_f_train, grad_outputs=torch.ones(x_f_train.shape[0], ).to(self.device), create_graph=True)[0]
    grad_u_t = grad_u[:, 0]
    grad_u_x = grad_u[:, 1]
    grad_u_xx = torch.autograd.grad(grad_u_x, x_f_train, grad_outputs=torch.ones(x_f_train.shape[0], ).to(self.device), create_graph=True)[0][:, 1

    residual = grad_u_t - self.v * grad_u_xx
    return residual
```

Why PINNs are great ?: II

- Sound theoretical basis for PINNs (DeRyck, SM, Molinaro)
- Regularity of PDEs $\Rightarrow$ Small Residuals
- Coercivity of PDEs $\Rightarrow$ Small Total Error
- Quadrature bounds $\Rightarrow$ Small Generalization Error
- Bounds available for both Forward and Inverse Problems.
- Also explain limitations of PINNs.

Kolmogorov PDEs

- Linear Parabolic PDEs of form:

$$\partial_t u = \sum_{i=1}^{d} \mu_i(x)\partial_{x_i} u + \frac{1}{2}\sum_{i,j,k=1}^{d} \sigma_{ik}(x)\sigma_{kj}(x)\partial_{x_i x_j} u,$$

$$u|_{\partial D\times(0,T)} = \Psi(x,t), \quad u(x,0) = \varphi(x)$$

- μ, σ are Affine
- Examples:
  - Heat Equation: $\mu = 0$, $\sigma = ID$
  - Black-Scholes Equation for Option Pricing:
  - Interest rate μ, Stock Volatilities β and correlations ρ

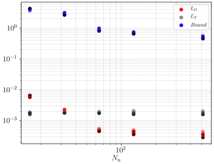
$$u_t = \sum_{i,j=1}^{d} \beta_i\beta_j\rho_{ij}x_i x_j u_{x_i x_j} + \sum_{j=1}^{d} \mu x_j u_{x_j}$$

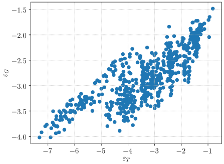
- Note that $d >> 1$ (Very high-dimensional)

Error Bounds: De Ryck, SM, 2021.

- $\exists$ Tanh PINN $\hat{u}$ of size $\mathcal{O}(\epsilon^{-\alpha(d)})$: $\mathcal{E}_{G,T}(\hat{\theta}) \sim \epsilon$,
- Uses Dynkin's formula to overcome curse of dimensionality.
- Stability of PDE: $\|u - u_\theta\|_2 \leq C \left(\|\mathcal{R}_{int,\theta}\| + \|\mathcal{R}_{sb,\theta}\|^{\frac{1}{2}} \right)$
- Use Hoeffding's inequality + Lipschitz bounds on u_θ:

$$\mathcal{E}_G^2(\theta) \sim \mathcal{O}\left(\mathcal{E}_T^2(\theta) + \frac{C\left(M, \log(\|W\|)\right) \log(\sqrt{N})}{\sqrt{N}} \right)$$





Numerical Results: (SM, Molinaro, Tanios, 2021)

- Heat Equation:

Dimension	Training Error	Total error
20	0.006	0.79%
50	0.006	1.5%
100	0.004	2.6%

- Black-Scholes type PDE with Uncorrelated Noise:

Dimension	Training Error	Total error
20	0.0016	1.0%
50	0.0031	1.5%
100	0.0031	1.8%

- Heston option-pricing PDE

Dimension	Training Error	Total error
20	0.0064	1.0%
50	0.0037	1.3%
100	0.0032	1.4%

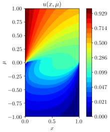
Radiative Transfer Equations

- $2d + 1$-dim Integro-Differential PDE for Intensity

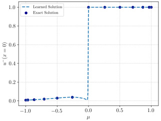
$$\frac{1}{c}u_t + \omega \cdot \nabla u + \left(k(x,\nu) + \sigma(x,\nu)\right) u \\ - \frac{\sigma(x,\nu)}{s_d} \int\limits_{R_+} \int\limits_{S} \Phi(\omega,\omega',\nu,\nu') u d\omega' d\nu' = f(x,t,n,\nu).$$

- High-dimensional, non-local, mixed-type, multiphysics
- PINNs applied and bound derived in SM, Molinaro 2021.

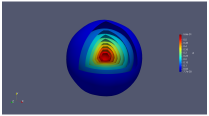
Numerical Results



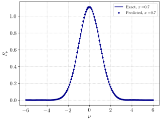
2-D, Intensity



2-D, Boundary



6-D, Inc. Radiation

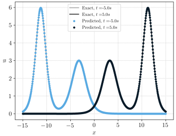


6-D, Radial flux

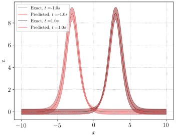
Dimension	Network Size	Error	Training Time
2	24×8	0.3%	57 min
6	20×8	2.1%	66 min

PINNs for nonlinear Dispersive PDEs

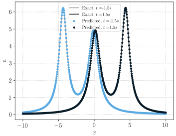
- Bounds + Numex in Bai, Koley, SM, Molinaro, 2021.



KdV



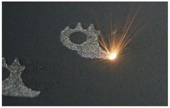
6-d param KdV



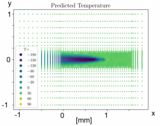
Benjamin-Ono

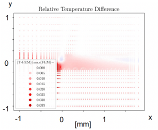
PDE	Network Size	Error
KdV	32×4	0.1%
6-D param KdV	24×8	0.5%
Benjamin-Ono	20×4	0.7%

Industrial Case Study: Joint with EMPA


www.all3dp.com

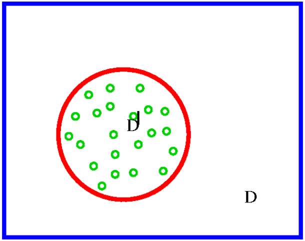

www.alphaform.de





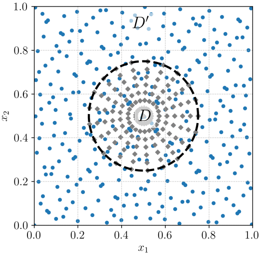
- Multi-scale thermal Simulation of Powder bed Additive Manufacturing
- Modeled with Parameterized Diffusion Equation
- Replace FEM simulations with PINNs
- FEM simulation time (31.6) CPU hrs vs PINN training time (0.4) hrs.

Inverse Problem: Data Assimilation.



- Abstract PDE is $\mathcal{D}(\mathrm{u}) = \mathrm{f}$ in $\mathbb{D}$
- Measurements: $\mathcal{L}(\mathrm{u}) = \mathrm{g}$ in $\mathbb{D}'$
- Goal: Reconstruct u given f, g.
- $\mathrm{u}_\theta : \mathbb{D} \mapsto \mathbb{R}^m$ is a PINN
- PDE Residual: $\mathcal{R} := \mathcal{R}_\theta(y) = \mathcal{D}\left(\mathrm{u}_\theta(y)\right) - \mathrm{f}(y), \;\; y \in \mathbb{D}$
- Data Residual: $\mathcal{R}_d := \mathcal{R}_{d,\theta}(z) = \mathcal{L}\left(\mathrm{u}_\theta(z)\right) - \mathrm{g}(z), \;\; z \in \mathbb{D}'$.
- PINNs are minimizers of $\int_{\mathbb{D}} |\mathcal{R}_\theta(y)|^{p_y} dy + \int_{\mathbb{D}'} |\mathcal{R}_{d,\theta}(z)|^{p_z} dz$

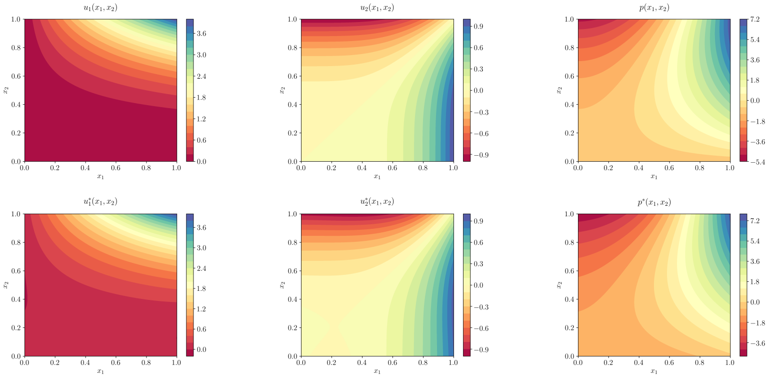
Stokes Equations in SM, Molinaro, 2021



- PDE: $\Delta u + \nabla p = f, \quad \mathrm{div}\ u = 0, \quad u|_{\mathbb{D}'} = g \quad \text{in } \mathbb{D}' \subset \mathbb{D}$
- Coercivity by Carleman estimates (Uhlmann et al)
- Error Estimate:

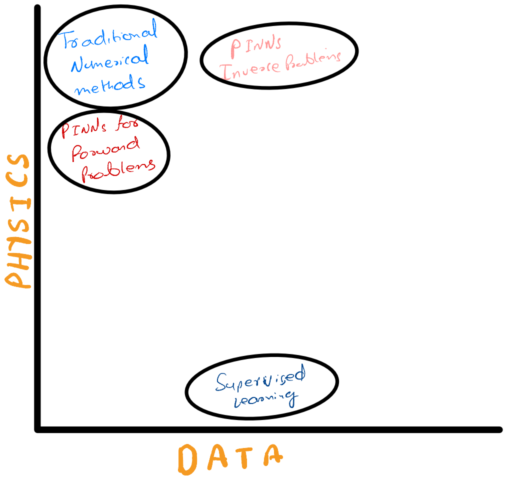
$$\|u - u^*\|_{L^2(B_R)} \sim \mathcal{O}\left(\mathcal{E}_{p,T}^{1-\tau}(1 + \mathcal{E}_{d,T}^{\tau}) + N^{-(1-\tau)\alpha}(1 + N_d^{-\alpha_d \tau})\right)$$

2-D Results



N	Training Error	$\mathcal{E}(u)$	$\mathcal{E}(p)$
20^2	0.0007	2.3%	5.6%
40^2	0.0004	1.7%	4.0%
80^2	0.0004	1.5%	3.5%

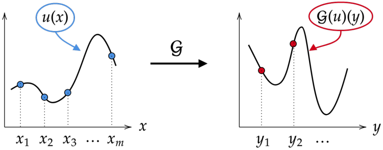
Data vs. Physics



Further Caveats

- Recall Setting:
  - Abstract PDE: $\mathcal{D}_a(u) = f$
  - Solution Operator: $\mathcal{G} : X \mapsto Y$ with $\mathcal{G}(a, f) = u$
- Approach so far needed:
  - Parametrizations as $\dim(\mathrm{Supp}(\mu)) = d_y < \infty$
  - Complete knowledge of underlying physics.
- In Reality (often):
  - Missing physics + Data acquired from observations.
  - Sample from measure $\mu \Rightarrow$ No good Parametrizations
  - Resolution dependent data and models
- Possible Solution by Operator Learning i.e., Learn Operators from Data

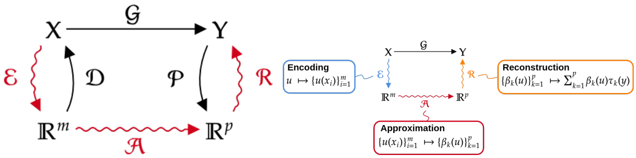
Operator Learning



- Underlying Solution Operator: $\mathcal{G}(a) = u$ for PDE $\mathcal{D}u = a$
- Task: Find a Surrogate (based on DNNs) $\mathcal{G}^* \approx \mathcal{G}$ from data.
- Challenge: DNNs are finite dimensional but we need infinite-dimensional objects here.

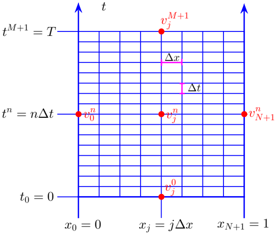
Revisiting Numerical Methods

- Can be reinterpreted in the following abstract Paradigm:

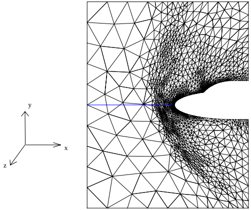


Scheme	Encoder	Approximator	Reconstructor
Finite Difference	Point values	Scheme	Poly. Interpolant
Finite Element	Node Values	Scheme	Galerkin Basis
Finite Volume	Cell Averages	Scheme	Poly. Interpolant
Spectral	Fourier Coeffs.	Scheme	Fourier Interpolant

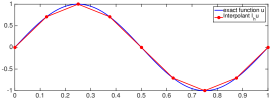
Traditional Numerical Methods



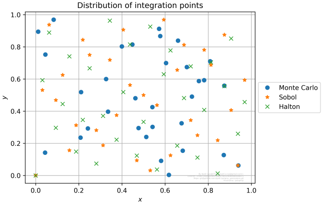
Finite Difference



Finite Volume



Finite Element



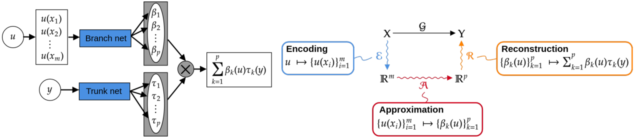
Collocation

- Different flavors of Runge-Kutta for Time Integration

Operator Networks

- First proposed by Chen,Chen 1995.
- DeepONets proposed by Karniadakis, Lu et al, 2020:

$$\mathcal{N}(a)(y) = \tau_0(y) + \sum_{k=1}^{} \beta_k(a)\tau_k(y) \approx \mathcal{G}(a)(y)$$



Architecture	Encoder	Approximator	Reconstructor
DeepOnet	Sensor Evals.	DNNs	DNNs
PCA-Net	Input PCA	DNNs	Output PCA

- PCA-net of Bhattacharya, Stuart et al, 2020.

Neural Operators

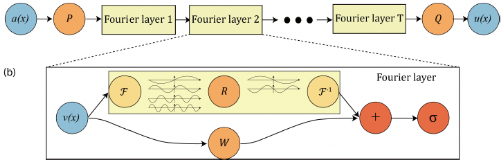
- Formalized in Kovachki et al, 2021.
- Recall: DNNs are $\mathcal{L}_\theta = \sigma_K \odot \sigma_{K-1} \odot \ldots\ldots \sigma_1$
- Single hidden layer: $\sigma_k(y) = \sigma(A_k y + B_k)$
- Neural Operators generalize DNNs to ∞-dimensions:
- NO: $\mathcal{N}_\theta = \mathcal{N}_L \odot \mathcal{N}_{L-1} \odot \ldots\ldots \mathcal{N}_1$
- Single hidden layer;

$$(\mathcal{N}_\ell v)(x) = \sigma \left(A_\ell v(x) + B_\ell(x) + \int\limits_D K_\ell(x,y) v(y) dy \right)$$

- Kernel Integral Operators are general Linear operators.
- Learning Parameters in A_ℓ, B_ℓ, K_ℓ
- Different Kernels $\Rightarrow$ Low-Rank NOs, Graph NOs, Multipole NOs, ...

Fourier Neural Operators

- FNO proposed in Li et al, 2020.



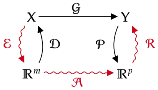
- Translation invariant Kernel $K(x, y) = K(x - y)$
- Use Fourier and Inverse Fourier Transform to define the KIO:

$$\int_D K_\ell(x, y) v(y) dy = \mathcal{F}^{-1}(\mathcal{F}(K)\mathcal{F}(v))(x)$$

- Parametrize Kernel in Fourier space.
- Fast implementation through FFT
- Can also be viewed in the earlier paradigm as:

Architecture	Encoder	Approximator	Reconstructor
FNO	Grid Evals	DNNs	Fourier Basis

Why do ONets/NOs work ?: Theory



- DeepOnets : Lanthaler, SM, Karniadakis, 2022.
- FNOs: Kovachki, Lanthaler, SM, 2022.
- PCA-net: Lanthaler, SM, Stuart, 2022.
- Universal Approximation Thm: For $\mu \in Prob(L^2(D))$ and any measurable $\mathcal{G} : H^r \mapsto H^s$ and $\epsilon > 0$, $\exists \mathcal{N}$ (Onet): $\hat{\mathcal{E}} < \epsilon$
- Upper (and lower) bounds via $\mathcal{E}_{\mathcal{R}} \leq \mathcal{E} \leq C(\mathcal{E}_{\mathcal{E}} + \mathcal{E}_{\mathcal{A}} + \mathcal{E}_{\mathcal{R}})$.
- $\mathcal{E}_{\mathcal{E},\mathcal{R}}$ decay as spectrum of Covariance operator of μ and $\mathcal{G}\#_{\mu}$.
- For $G = \mathcal{P} \circ \mathcal{G} \circ \mathcal{D} \in C^k(\mathbb{R}^m, \mathbb{R}^p) \Rightarrow \text{size}\ (\mathcal{N}) \sim \mathcal{O}\left(\epsilon^{-\frac{m(\epsilon)}{k}}\right)$.
- Curse of Dimensionality (CoD) for DeepOnets !!!

On CoD for Operator Networks

- DeepOnet, FNO, PCA-net
- Break the Curse of Dimensionality !!
- For operators $\mathcal{G}$ corresponding to many PDEs:

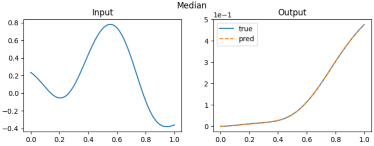
PDE	Operator	Complexity
$-u'' = \sin(u) + f$	$\mathcal{G} : f \to u(T)$	$M \sim \mathcal{O}(\epsilon^{-\eta})\ \eta \approx 0$
$-\mathrm{div}(a\nabla u) = f$	$\mathcal{G} : a \to u$	$M \sim \mathcal{O}(\epsilon^{-\eta})\ \eta \approx 0$
$u_t = \Delta u + f(u)$	$\mathcal{G} : u_0 \to u(T)$	$M \sim \mathcal{O}(\epsilon^{-2(d+1)})$
$u_t + \mathrm{div}(f(u)) = 0$	$\mathcal{G} : u_0 \to u(T)$	$M \sim \mathcal{O}(\epsilon^{-\alpha(d+1)})$
$u_t + u \cdot \nabla u + \nabla p = \nu \Delta u$	$\mathcal{G} : u_0 \to u(T)$	$M \sim \mathcal{O}(\epsilon^{-(d+1)})$

- Case by Case basis: Holomorphy, Emulation of Numerical Schemes etc...
- Unified framework in DeRyck, SM, 2022.

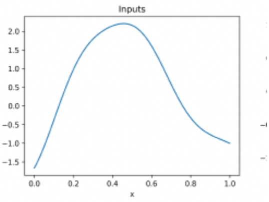
Numerical Results: Forced Pendulum

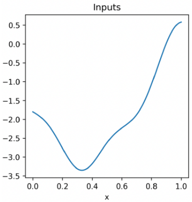
- Forced Pendulum: $-u'' = \sin(u) + f$ with Operator $\mathcal{G} : f \to u$
- Measure μ is Law of a Gaussian Random Field (GRF) with correlation length scale $\ell = 0.2$
- Comparison of test errors with 200 training samples:

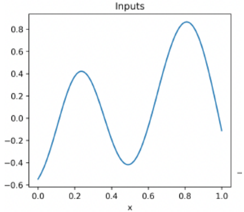
Architecture	Error
DeepONet	0.35%
PCA-Net	0.21%
FNO	0.23%



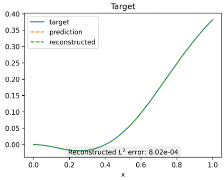
Forced Pendulum

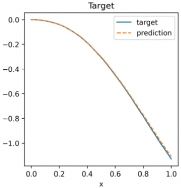


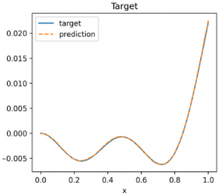




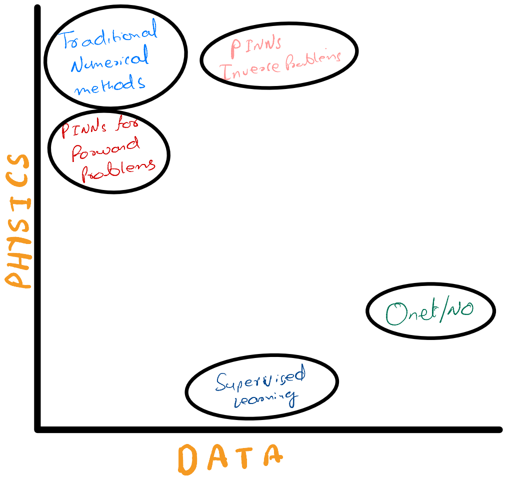
DeepONet PCA-net FNO





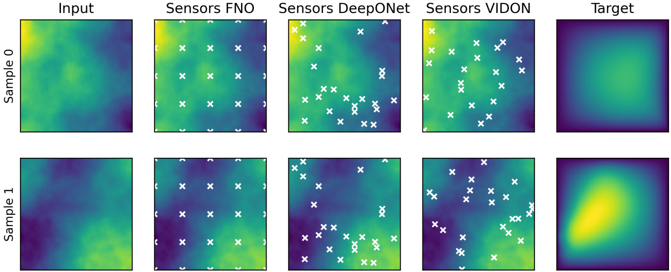


Data vs. Physics

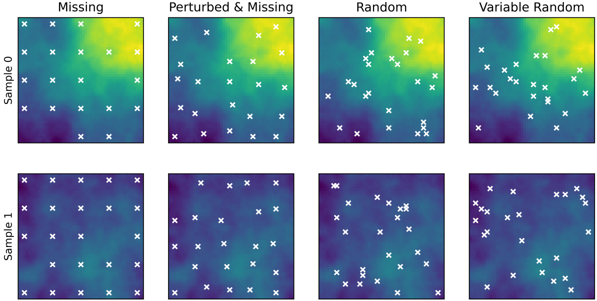


Onets/NOs: Issues

- DeepONets/FNOs can only handle Rigid Inputs !!
- Same number of sensors over samples at the same locations.



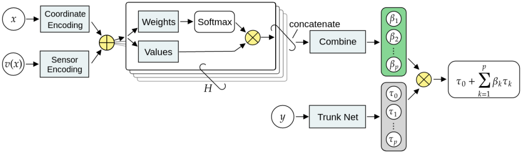
In Reality



- Encoded inputs to DeepONets/FNO/PCA are not Permutation Invariant !!
- Inputs: Vectors of fixed length rather than functions at arbitrary points.

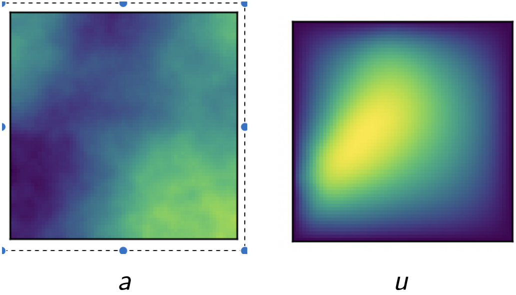
Variable Input Deep Operator Networks

- VIDON proposed in Prasthofer, DeRyck,SM, 2022.



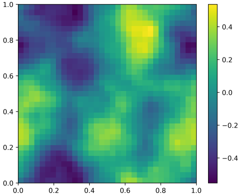
- Inspired by Deep Sets and Transformers
- Allows variable sensor locations and numbers across samples.
- Size is linear in number of sensors.
- Similar theoretical results as DeepONets/FNOs.

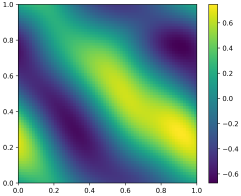
Results for Darcy Flow



Configuration	# (Sensors)	**FNO**	**DeepONet**	**VIDON**
Regular Grid	51×51	0.76%	1.48%	1.29%
Irregular Grid	$51^2 = 2601$	-	1.52%	1.48%
Missing Data	$[2081, 2601]$	-	-	1.77%
Perturbed Grid	[2341,2861]	-	-	1.68%
Random Locations	2601	-	-	2.58%
Variable Random	[2341,2861]	-	-	2.55%

Results for 2-D Navier-Stokes

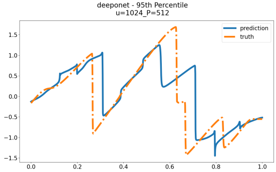




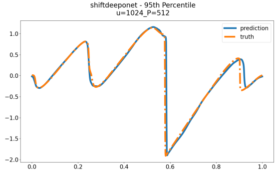
Configuration	# (Sensors)	**FNO**	**DeepONet**	**VIDON**
Regular Grid	33×33	3.49%	4.20%	5.22%
Irregular Grid	$33^2 = 1089$	-	4.33%	5.45%
Missing Data	$[871, 1089]$	-	-	5.64%
Perturbed Grid	[980, 1198]	-	-	5.34%
Random Locations	1089	-	-	8.35%
Variable Random	[980, 1198]	-	-	8.28%

Onets/NOs: Issues

- Affine Reconstructors $\Rightarrow$ Slow decay of eigenvalues of Covariance operator of $\mathcal{G}\#_\mu$.
- Particularly for Transport dominated problems



DeepOnet



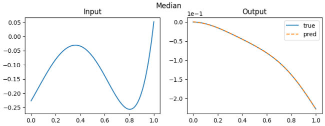
ShiftDeepOnet

- Nonlinear reconstructions with ShiftDeepOnets of Hadorn, Lanthaler, SM, 2022.

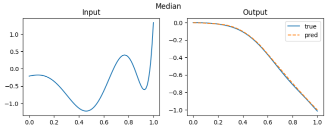
$$\mathcal{N}(a)(y) = \sum_{k=1} \beta_k(a)\tau_k(\gamma_k(a)y + \lambda_k(a)) \approx \mathcal{G}(a)(y)$$

ONets/NOs: Issues

- Out of Distribution Evaluation:



Cheb 5 (Error: 0.43%)



Cheb 10 (Error: 3.78%)

- Use Casuality to improve Generalization.
  - Limited Data.
  - PINNs + Onets/NOs $\Rightarrow$ PIONs/PINOs.
  - Theory in DeRyck, SM, 2022.

Data vs. Physics