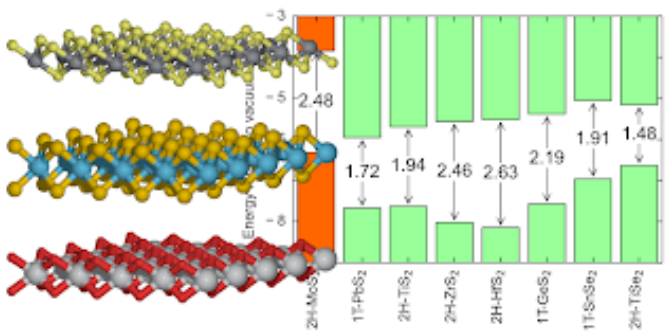
Deep Learning and Computations of PDEs

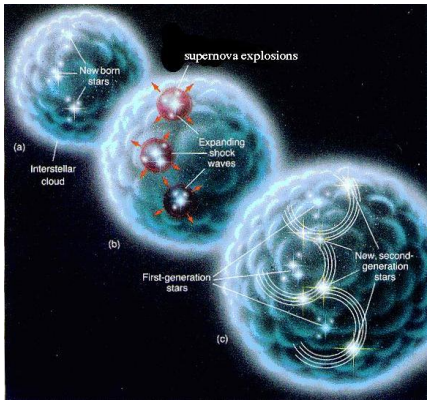
Siddhartha Mishra

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

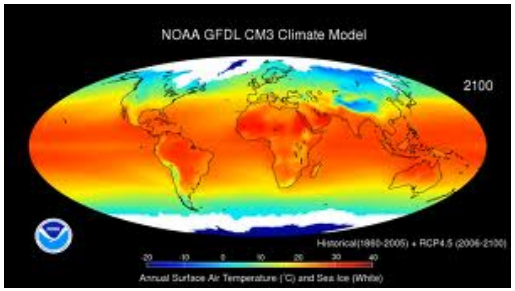
Partial Differential Equations (PDEs)



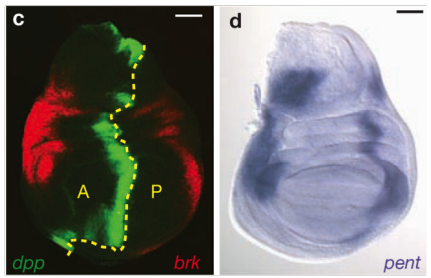
Atoms



Stars

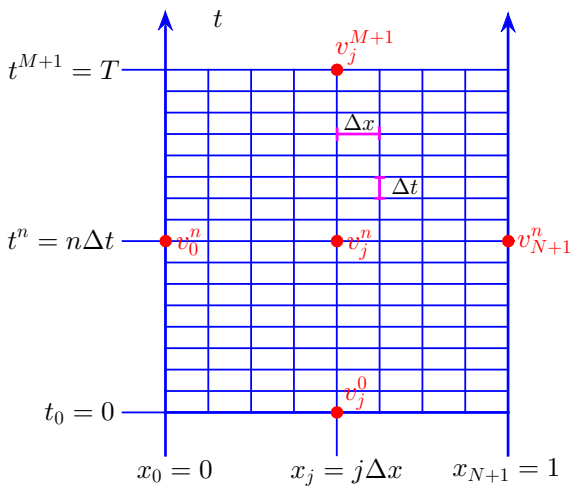


Climate

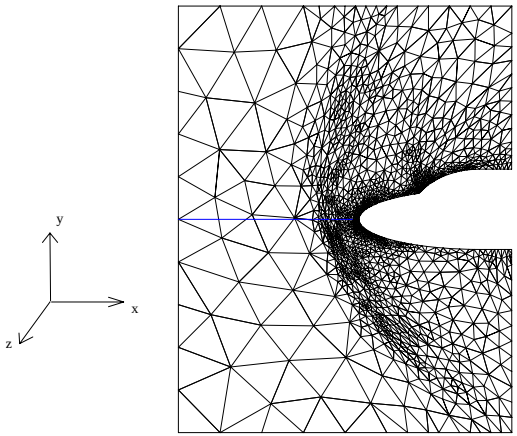


Fly wing

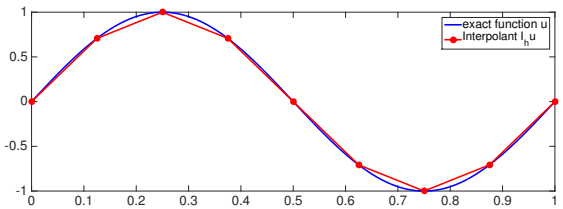
Numerical Methods



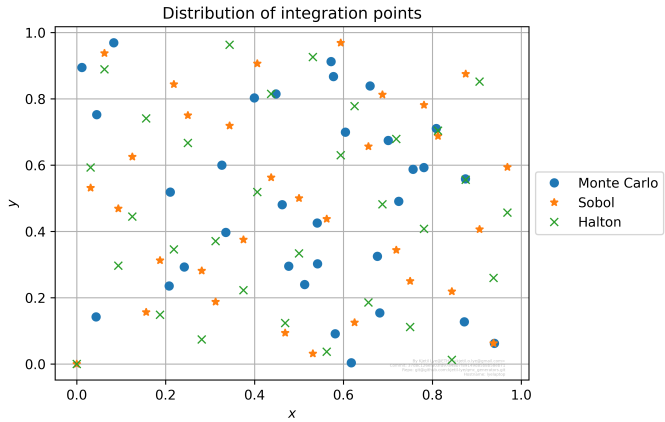
Finite Difference



Finite Volume



Finite Element

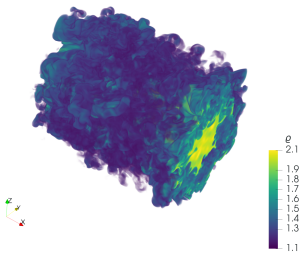


Collocation

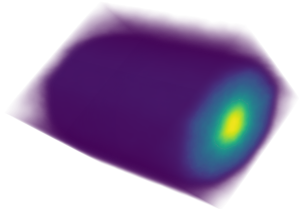
Numerical methods for PDEs: Issues

- Too expensive or infeasible in High Dimensions: $d > 4$
- Explicitly high-dimensional PDEs:
  - Boltzmann Eqn (7), Radiative Transfer Eqns (7).
  - Black-Scholes, Schrödinger ($10^2 - 10^3$)
- Implicitly High-dimensional Parametric PDEs.

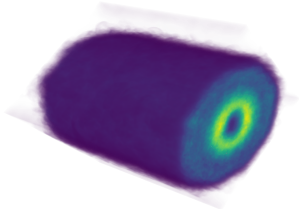
Uncertainty Quantification



Sample



Mean



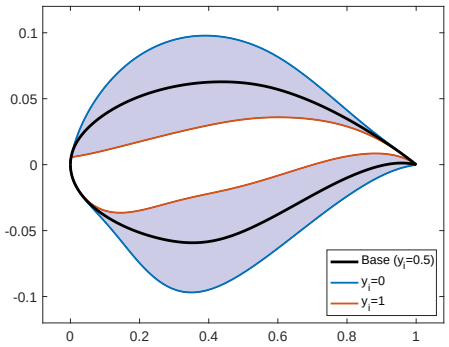
Variance

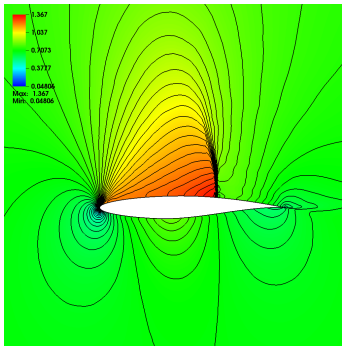
- Simulation of Compressible Flow with ALSVINN
- 280 NH for a 1024^3 run on PIZ DAINT (12th in Top500)

- Ensemble simulation costs 700 K USD !!!

Abstraction: Many Query Problems

- Another example: Design of airfoils
- Flow modeled with Euler or Navier-Stokes
- Design parameters: Shape via Hicks-Henne functions.





- Multiple calls to PDE solvers is very expensive.

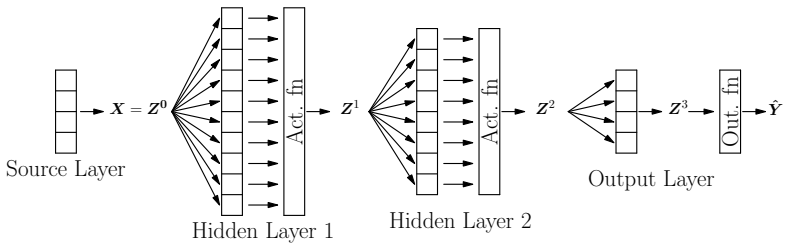
High-dimensional PDEs: Many Query Problems

- PDEs of Parametric form:

$$\mathcal{D}_{x,t}(\mathrm{u}(x,t,y)) = \mathrm{f}(y), \quad (t,x) \in \mathbb{R}_+ \times \mathbb{R}^d, \ y \in Y \subset \mathbb{R}^{\bar{d}}, \ \bar{d} >> 1.$$

- UQ: Y parametrizes probability space.
- Optimal control and design: Y parametrizes design space.
- Large number of Evaluations of:
  - Solution fields $\mathrm{u}(t,x,y)$
  - Observables $\mathcal{L}(y) = \int_{\mathbb{R}_+} \int_D \Psi(x,t) g(\mathrm{u}(x,t,y)) dx dt$
- Approximate Fields or observables with deep neural networks

Supervised learning with Deep Neural networks



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

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

Supervised learning for high-d Parametric PDEs

- Can we find DNN such that $\|\mathcal{L}^* - \mathcal{L}\| \sim \mathcal{O}(\epsilon)$?
- YES: Universal Approximation Property of DNNs $\Rightarrow$:
- Given any Continuous (measurable) $\mathcal{L}$, exists a $\hat{\mathcal{L}}$:

$$\|\mathcal{L} - \hat{\mathcal{L}}\| < \epsilon$$

- If $\mathcal{L} \in W^{s,p}$, $\exists$ DNN $\hat{\mathcal{L}}$ with M parameters (Yarotsky):

$$\|\mathcal{L} - \hat{\mathcal{L}}\|_p \sim \mathcal{O}\left(M^{-\frac{s}{\bar{d}}}\right).$$

- But in Scientific Computing (often):
- $\mathcal{L}$ is not be very Regular and $\bar{d} >> 1$
- If $\mathcal{L} \in W^{1,\infty}$, $\bar{d} = 6$: 1% error, need network of size 10^{12} !!
- Curse of dimensionality: DNN Size $M \sim \epsilon^{-\frac{\bar{d}}{s}}$

Refined Error Estimates

- Error $\mathcal{E} := \|\mathcal{L} - \mathcal{L}^*\|_p$ Decomposition: $\mathcal{E} \leq \mathcal{E}_{app} + \mathcal{E}_{gen} + \mathcal{E}_{opt}$.
- Approximation error $\mathcal{E}_{app} = \|\mathcal{L} - \hat{\mathcal{L}}\|_p$,
  - $\hat{\mathcal{L}}$ is best approximation of $\mathcal{L}$ in $\mathcal{NN}(M)$.
  - One can prove that $\mathcal{E}_{app} \sim \mathcal{O}\left(\bar{d}^{\sigma} M^{-\eta}\right)$ for,
  - Linear Elliptic PDEs: (Schwab, Kutyniok et al).
  - Semi-linear Parabolic PDEs: (E, Jentzen et al).
  - Nonlinear Hyperbolic PDEs: (DeRyck, SM, 2021).
- Optimization Error $\mathcal{E}_{opt} \sim$ Computable Training error:
- Generalization Error

$$\mathcal{E}_{gen}(\theta) := \|\mathcal{L} - \mathcal{L}^*_\theta\|_p^p - \frac{1}{N}\sum_i |\mathcal{L}^*_\theta(y_i) - \mathcal{L}(y_i)|^p$$

- Using Concentration inequalities + Covering number bounds:

$$\mathcal{E}_{gen} \sim \frac{C\left(M, \log(\|W\|)\right)\log(\sqrt{N})}{\sqrt{N}}$$

Well-trained Networks

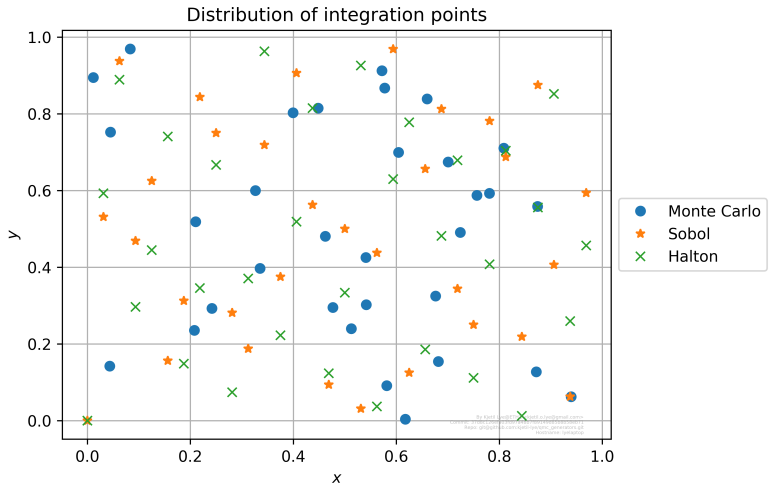
- Assume we can find DNN such that $\mathcal{E}_{app}, \mathcal{E}_T << 1$
- Still Overall Error behaves as

$$\mathcal{E} \sim \frac{C(M)}{N^{\alpha}}, \quad \alpha \leq \frac{1}{2}.$$

- If $C(M) \sim \mathcal{O}(1)$, error of 1% requires 10^4 training samples !!
- Challenge: learn maps of low regularity in a data poor regime
- Contrast with Big Data successes of machine learning.

A Trick: Lye, SM, Ray, 2020

- Use Low discrepancy sequences $\{y_i\}_{i=1}^N \in Y$ as Training Set



- These sequences are Equidistributed (better spread out).
- Examples: Sobol, Halton, Owen, Niederreiter ++
- Basis of Quasi-Monte Carlo (QMC) integration.

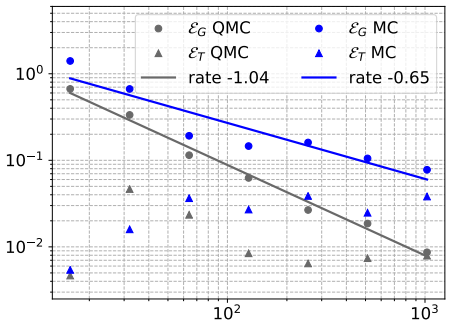
Training on Low-Discrepancy Sequences

- For $\mathcal{L}$ with Bounded Hardy-Krause variation and smooth σ.
- Generalization Error for Sobol sequences: (SM, Rusch, 2020),

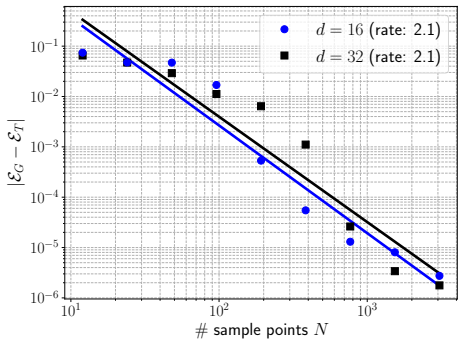
$$\mathcal{E} \leq \mathcal{E}_T + C\left(V_{HK}(\mathcal{L}), V_{HK}(\mathcal{L}^*)\right) \frac{(\log N)^d}{N},$$

- For Holomorphic $\mathcal{L}$ (Elliptic and Parabolic PDEs)
- Error for EPL pts (SM, Longo, Schwab, Rusch, 2020)

$$\mathcal{E} \leq \mathcal{E}_T + \frac{C(\mathcal{L}, \mathcal{L}^*)}{N^2},$$



Sobol

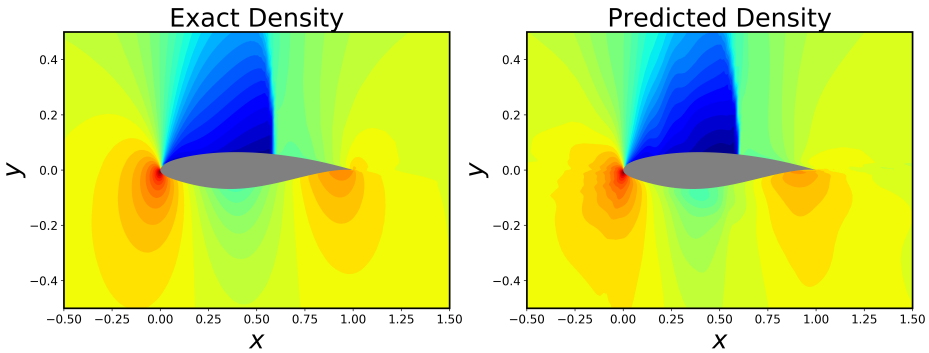


EPL

Prediction

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

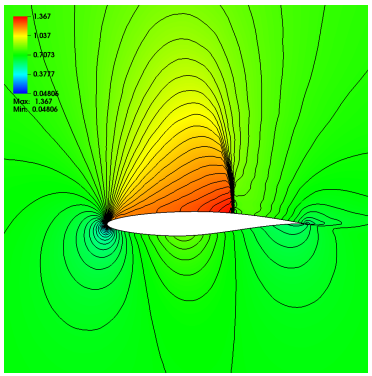
	Run time (1 sample)	Training	Evaluation	Error
Lift	2400 s	700 s	10^{-5} s	0.78%
Drag	2400 s	840 s	10^{-5} s	1.87%
Field	2400 s	1 hr	0.2 s	1.9%



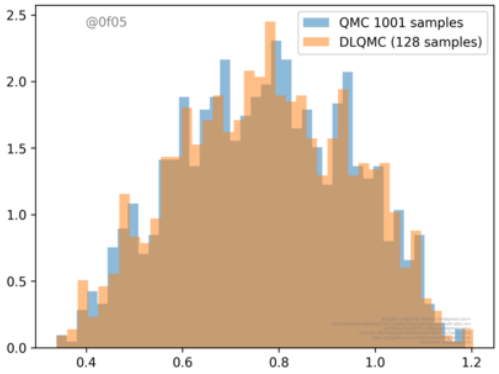
Forward UQ

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

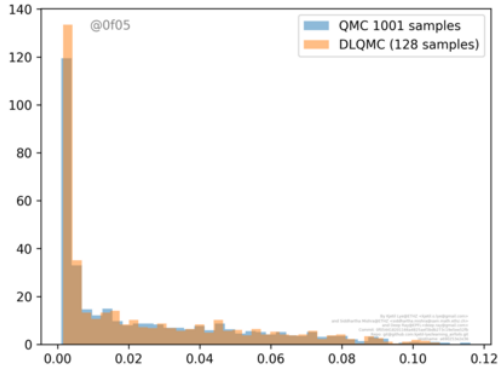
$$\mathcal{L}\#\mu \approx \frac{1}{M}\sum_{i=1}^{M}\delta_{\mathcal{L}(y_i)} \approx \frac{1}{M}\sum_{i=1}^{M}\delta_{\mathcal{L}^*(y_i)}$$



Sample



Lift PDF



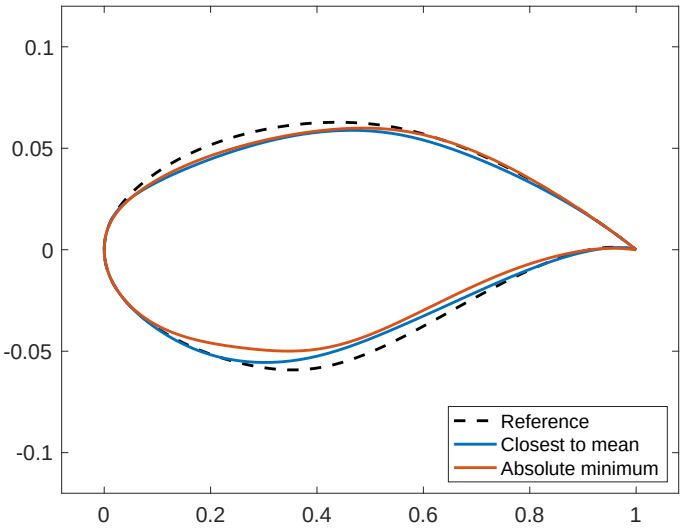
Drag PDF

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

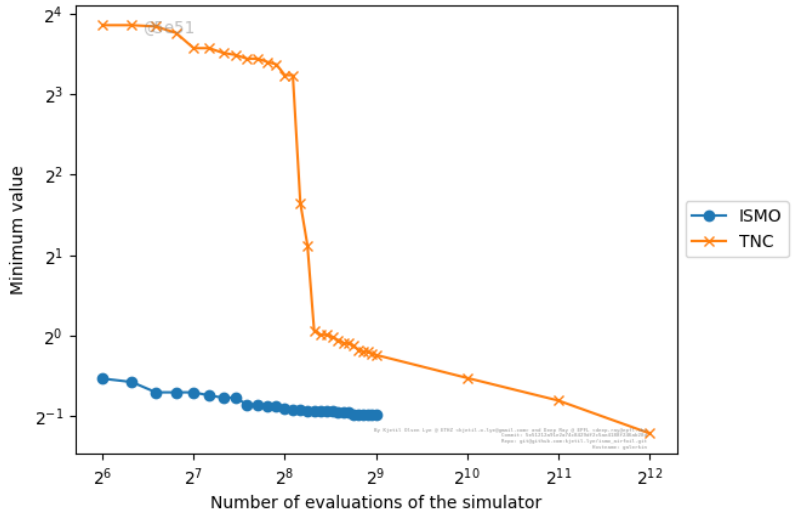
- Another factor of $3-5$ gain with Multi-level Training

PDE constrained Optimization

- Active learning ISMO algorithm (Lye,SM,Ray,Praveen, 2020).



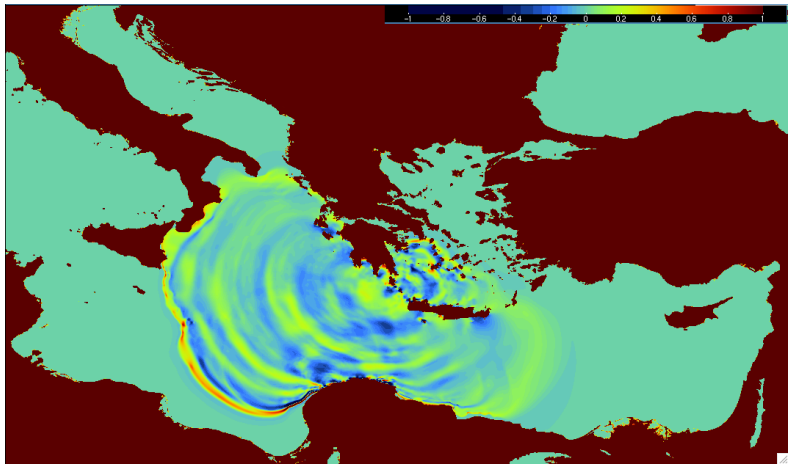
Shapes



ISMO vs. TNC

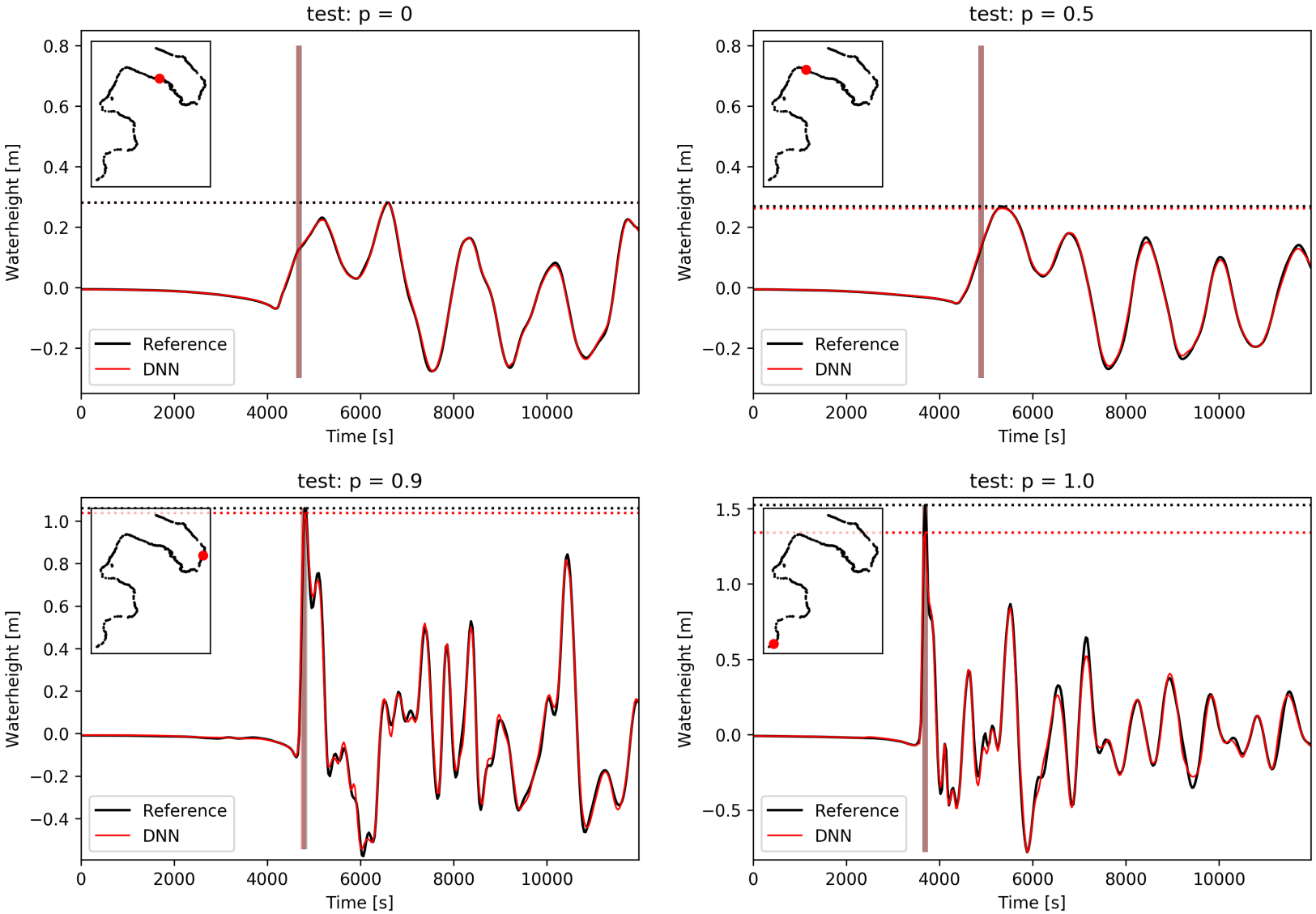
- Change airfoil shape to Minimize Drag for constant Lift.
- $>$ 50% Drag reduction at near constant lift.

Tsunami in the Mediterranean sea



- Modeled by one-layer Shallow water equations
- Okada Model: Seafloor deformation $\Rightarrow$ Initial conditions.
- State of the art Prediction + UQ $\approx$ 60 mins.

Time Series Prediction with DNNs



Recall: High-dimensional parametric PDEs

- PDEs of Parametric form:

$$\mathcal{D}_{x,t}(\mathrm{u}(x,t,y)) = \mathrm{f}(y), \quad (t,x) \in \mathbb{R}_+ \times \mathbb{R}^d, \ y \in Y \subset \mathbb{R}^{\bar{d}}, \ \bar{d} >> 1.$$

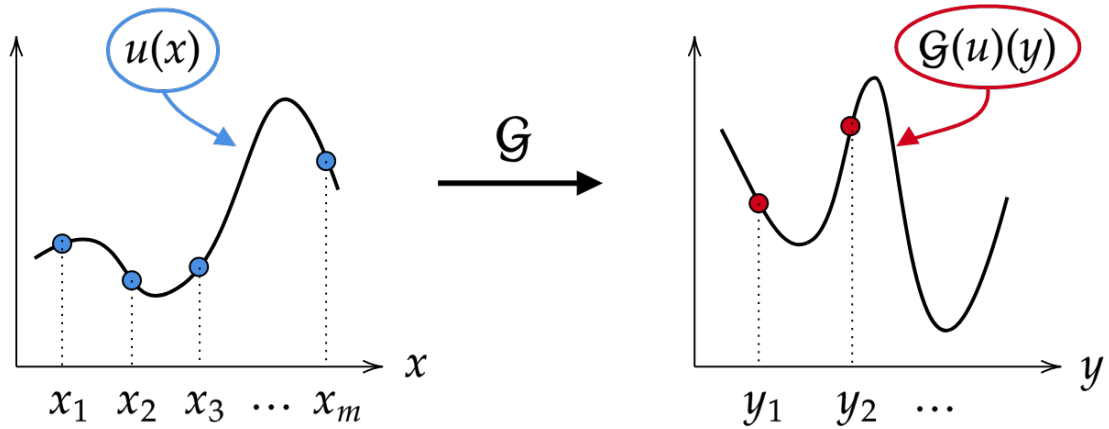
- UQ: Y parametrizes probability space.
- Optimal control and design: Y parametrizes design space.

DL for Many-Query Problems: Further Issues

- Approach so far needs knowledge of underlying PDEs
- Knowledge of underlying Measure for Parametrization
- Convergent numerical discretization for Data generation.
- In Reality (often):
  - Missing or Incomplete physics.
  - Data acquired from experiments or observations.
  - Only able to Sample from measure μ
- Consequences:
  - Difficult to design good parameterizations.
  - No possibly of Out of distribution evaluations.
  - Resolution dependent surrogates.
- Need Operator Learning

Operator Learning

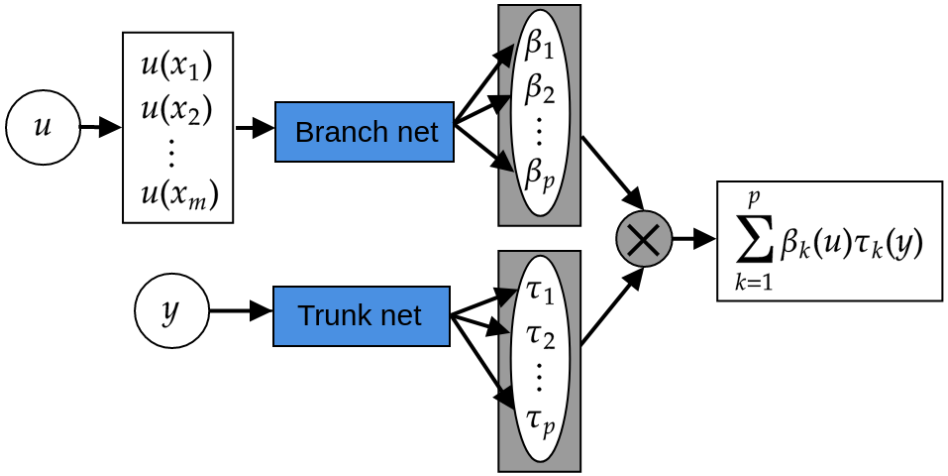
- Given abstract PDE $\mathcal{D}\mathrm{v} = \mathrm{u}$ with differential operator $\mathcal{D}$
- Underlying Solution Operator: $\mathcal{G}(\mathrm{u}) = \mathrm{v}$



- Task: Find a Surrogate (based on DNNs) $\mathcal{G}^* \approx \mathcal{G}$ from data.
- We discuss a recently proposed paradigm: DeepOnets

DeepOnets

- Operator Networks of Chen,Chen 1995.
- DeepOnets proposed by Karniadakis, Lu et al, 2020.



- DeepOnet:

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

- Very successful in numerical experiments (Karniadakis ++)

Why do DeepOnets work ?

- Universal Approximation Theorem of Chen,Chen 1995.
- For any Continuous operator $\mathcal{G} : K \subset C(D) \mapsto C(U)$, with Compact K: $\exists$ DeepOnet $\hat{\mathcal{N}}$ such that:

$$\sup_{u \in K} \sup_{y \in U} \left| \mathcal{G}(u)(y) - \hat{\mathcal{N}}(u)(y) \right| < \epsilon.$$

- In practice, K is NOT compact, $\mathcal{G}$ can be discontinuous
- and no explicit information on:
  - Number m and type of Sensor points.
  - Number p and architecture of Branch,Trunk nets (DNNs)
- Underlying learning task is ∞-dimensional $\Rightarrow$

$$\text{Size}(\text{DeepOnet}) \sim \mathcal{O}\left(e^{\frac{1}{\epsilon}}\right)$$

- Operator Approximation can be marred by Curse of dimensionality !!

DeepOnet Theory: Lanthaler,SM, Karniadakis, 2021

- DeepOnet $\mathcal{N} \approx \mathcal{G}$ is trained by minimizing Loss:

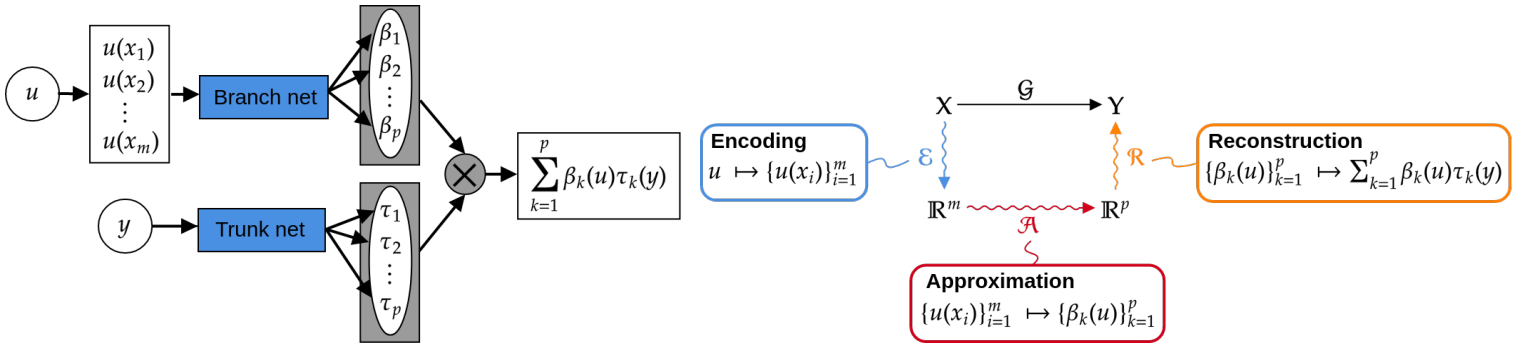
$$\mathcal{J} := \sum_{i,j} |\mathcal{G}(u_i)(y_j) - \mathcal{N}(u_i)(y_j)|^2, \quad u_i \sim \mu \in \mathrm{Prob}(X).$$

- Suggests the following definition of Generalization Error:

$$\hat{\mathcal{E}} := \left(\int_X \int_U |\mathcal{G}(u)(y)) - \mathcal{N}(u)(y)|^2 \, dy d\mu(u) \right)^{\frac{1}{2}}$$

- (Universal Approximation) Thm: For any $\mu \in Prob(C(D))$ and any $\mathcal{G} : L^2 \mapsto L^2$ and $\epsilon > 0$, $\exists \mathcal{N}$ (DeepOnet): $\hat{\mathcal{E}} < \epsilon$
- Key role of Input Measure μ

DeepOnet Decomposition



- Natural Error Decomposition:

$$\mathcal{E} \lesssim \mathcal{E}_{\mathcal{E}} + \mathcal{E}_{\mathcal{A}} + \mathcal{E}_{\mathcal{R}}.$$

- Goal: Control each error component !!

Bounds on Reconstruction Error

- Defined by: $\mathcal{E}_{\mathcal{R}}^2 = \int_Y \|v - \mathcal{P}_{Im(\mathcal{R})}(v)\|^2 d\mathcal{G}_{\#\mu}(v)$.
- PCA on Output space $\Rightarrow$

$$\sqrt{\sum_{\ell>p} \lambda_\ell} \leq \mathcal{E}_{\mathcal{R}} \leq \mathcal{E} \leq \mathcal{E}_{\mathcal{E}} + \mathcal{E}_{\mathcal{A}} + \sqrt{\sum_{\ell>p} \lambda_\ell} + C \max_{1\leq k\leq p} \|\tau_k - \varphi_k\|_{L^2}$$

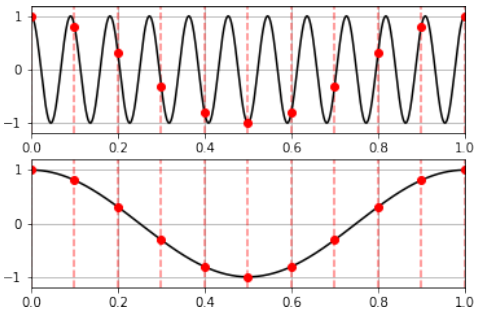
- With (λ_k, φ_k): Eigensystem of Covariance Operator

$$\Gamma := \int_Y (v \otimes v)\, d\mathcal{G}_{\#\mu}(v)$$

- Trunk Nets have to approximate Eigenfunctions.
- Reveals Fundamental Limitations of DeepOnets !!

Bounds on Encoding Error

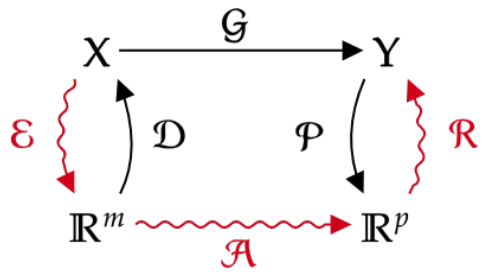
- $\mathcal{E}_{\mathcal{E}}^2 = \int_X \|u - \mathcal{P}_{\mathcal{E}}(u)\|^2 d\mu(u)$, Covariance: $\Gamma := \int_X (u \otimes u)\, du$
- PCA on Input space with $\mathcal{P}_{\mathcal{E}} : L^2(D) \mapsto \text{span}\,(\varphi_1, \ldots, \varphi_m)$
- Bound of the form: $\mathcal{E}_{\mathcal{E}} \sim \sqrt{\sum_{\ell > p} \lambda_\ell} + \mathcal{E}_{\text{aliasing}}$
- Aliasing errors due to pointwise sensing: $\{u(x_j)\}_{1 \leq j \leq m}$



- Choice of m Random sensor points is almost optimal:

$$\mathcal{E}_{\mathcal{E}} \leq \sqrt{\sum_{\ell > \frac{m}{\log(m)}} \lambda_\ell}.$$

Bounds on the Approximation Error



- Given by:

$$(\mathcal{E}_{\mathcal{A}})^2 = \int_{\mathbb{R}^m} \|\mathcal{A}(\mathrm{u}) - \underbrace{\mathcal{P} \circ \mathcal{G} \circ \mathcal{D}}_{=:G}(\mathrm{u})\|_{\ell^2}^2 \, d\left(\mathcal{E}_{\#}\mu\right)(\mathrm{u}).$$

- DNN approximation bounds for $G \in C^k(\mathbb{R}^m, \mathbb{R}^p)$:

$$\text{For } \mathcal{E}_{\mathcal{A}} \sim \mathcal{O}(\epsilon) \quad \Leftarrow \quad \text{size } (\mathcal{A}) \sim \mathcal{O}\left(\epsilon^{-\frac{m}{k}}\right)$$

- But for $\mathcal{E}_{\mathcal{E}} \sim \mathcal{O}(\epsilon)$, we need $m = m(\epsilon) \to \infty$
- $\Rightarrow$ Curse of Dimensionality (CoD) as

$$\text{size } (\mathcal{N}) \sim \mathcal{O}\left(\epsilon^{-\frac{m(\epsilon)}{k}}\right), \quad \text{for} \quad \mathcal{E} \sim \mathcal{O}(\epsilon).$$

- Exponential growth of DeepOnet size.

DeepOnets Break CoD I: By Holomorphy

- As size $(\mathcal{N}) \sim \mathcal{O}\left(\epsilon^{-\frac{m(\epsilon)}{k}}\right)$, No CoD if formally $k = \infty$
- Rigorous for Holomorphic maps: Schwab et al
- Thm: G is Holomorphic, then $\forall \delta$, size$(\mathcal{A}) \sim \mathcal{O}(\epsilon^{-\delta})$ for

$$\|G - \mathcal{A}\| \sim \mathcal{O}(\epsilon)$$

- Examples:

PDE	Operator	Complexity
$-u'' = \sin(u) + f$	$\mathcal{G} : f \to u$	$M \sim \mathcal{O}(\epsilon^{-\eta})$ $\eta \approx 0$
$-\mathrm{div}(a\nabla u) = f$	$\mathcal{G} : a \to u$	$M \sim \mathcal{O}(\epsilon^{-\eta})$ $\eta \approx 0$

DeepOnets Break CoD II: By Emulating Num. Methods

- Elliptic PDE $-\mathrm{div}(a\nabla u) = f$ with Operator $\mathcal{G} : f \to u$
- FEM with basis functions $\psi_{1,\ldots,p}$
  - Evaluate $f(x_j)$ at m Quadrature points.
  - Form Stiffness Matrix $A_{ij} = \int a\nabla\psi_i\nabla\psi_j dx$,
  - Load vector $b_j = \sum w_q f(x_q)\psi(x_q)$.
  - Invert to obtain $c = A^{-1}b$
  - FEM approximation is $u_m = \sum c_k\psi_k$
- DeepOnet $\mathcal{N}$ emulates FEM by
  - Sensor points: $\{x_q\}$ are quadrature points.
  - Branch net β is DNN $\approx$ Linear map: $\{f(x_q)\} \mapsto \{c_k\}$
  - Trunk Net τ approximates the FEM basis functions ψ.

Nonlinear Time-dependent PDEs

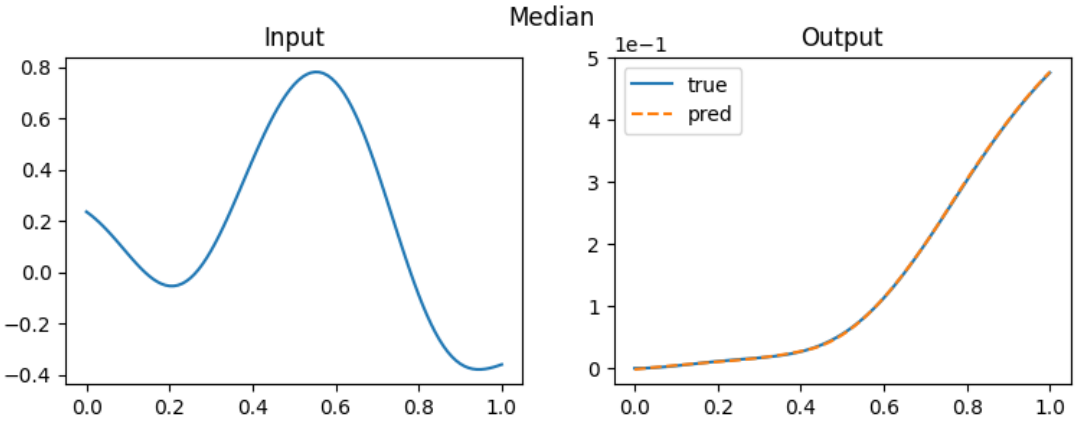
PDE	Operator	Complexity
$u_t = \Delta u + f(u)$	$\mathcal{G} : u_0 \to u(T)$	$M \sim \mathcal{O}(\epsilon^{-2(d+1)})$
$u_t + \mathrm{div}(f(u)) = 0$	$\mathcal{G} : u_0 \to u(T)$	$M \sim \mathcal{O}(\epsilon^{-\alpha(d+1)})$

- For Scalar conservation Law in 1-D:
- DeepOnet with

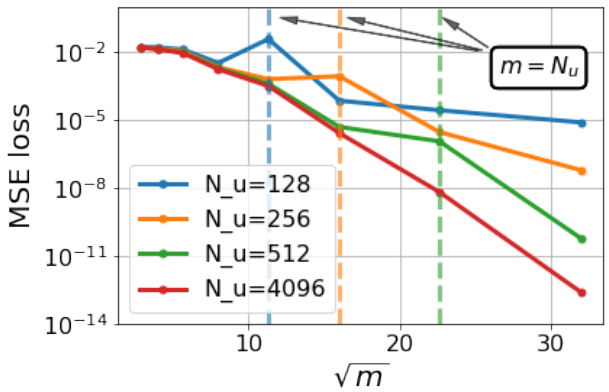
$$\text{Size } \tau \sim p, \quad \text{Size } \beta \sim p^{\frac{5}{2}},$$
$$\frac{1}{p} < \mathcal{E} \sim \frac{1}{p^{\alpha}}, \quad \alpha \approx 1.$$

Numerical Results

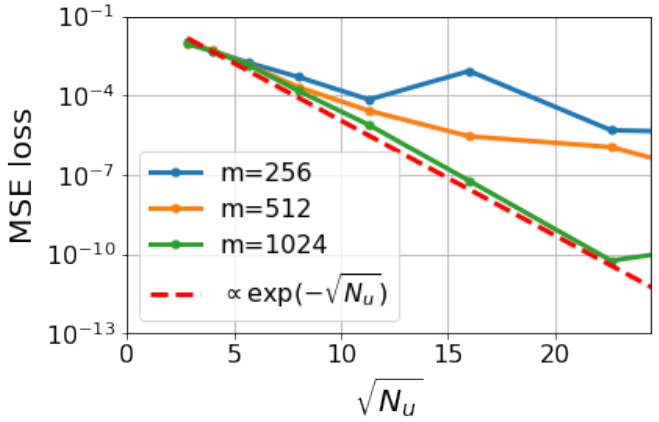
- Forced Pendulum: $-u'' = \sin(u) + f$ with Operator $\mathcal{G} : f \to u$
- Measure μ is Law of a Gaussian Random Field (GRF) with correlation length scale $\ell = 0.2$
- # Sensors $= 101$, Branch Net $= [20, 10]$, Trunk Net $= [40, 40, 10]$, #(Training Samples) $= 100$
- Mean Test Error of 0.71%



Further Numerical Results



Err vs. #(Sensors)



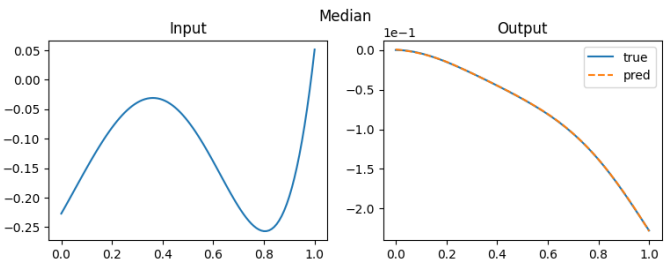
Err vs. #(Samples)

- Error decays exponentially wrt number of sensors (Predicted by Theory)
- Error decays exponentially wrt number of training samples !!
- Holds for small $N_u \Rightarrow$ Fast Training.
- We only have Generalization Error bound:

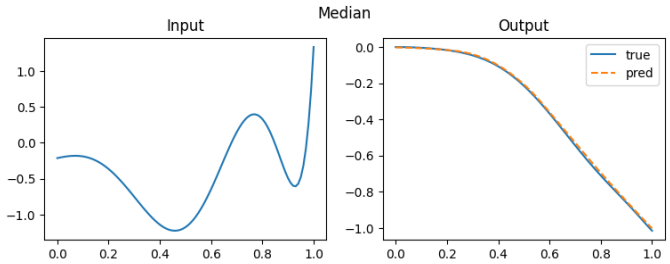
$$\mathcal{E}_{gen} \sim \frac{C\left(M, \log(\|W\|)\right) \log(\sqrt{N_u})}{\sqrt{N_u}}$$

Out of Distribution Evaluations

- Recall: Underlying measure is a GRF.
- Mean Error on Chebychev Polynomials of degree 5: 0.43%
- Mean Error on Chebychev Polynomials of degree 10: 3.78%



Cheb 5



Cheb 10

Deep learning and High-dimensional PDEs

- Many-Query Problems with Parametric PDEs
  - Training with Low-Discrepancy sequences
  - Multi-Fidelity training
  - Prediction, UQ, PDE constrained Optimization
- DeepOnets for Operator Learning.
- Other Paradigms: Fourier Neural Operators (FNOs):
  - Proposed in Li et al, 2021.
  - Analyzed in Lanthaler, Kovachki, SM. 2021
  - Universal Approximation for Operators
  - Breaks CoD for many PDEs
- PINNs for PDEs with high-dimensional state spaces.
- Successful application requires significant innovation over black box ML algorithms.