Deep Learning in Scientific Computing 2023: Lecture 4

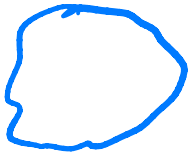
Siddhartha Mishra

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

What you learnt so far

- What is Deep Learning ?
  - Deep Neural Networks.
  - Large labelled datasets.
  - Gradient descent training algorithms.
  - Ability to generalize.
- The next several lectures: Use of Deep Learning for solving PDEs

What are PDEs ?



$D \subset \mathbb{R}^n$

$u : D \to \mathbb{R}$

u or $u(t,x)$

$u_t, u_{tt}, \ldots$

$\nabla u = (u_{x_1}, u_{x_2}, \ldots, u_{x_{n-1}})$

$\nabla^2 u, \ldots$

The general paradigm is : given a blueprint, to find the corresponding recipe. Much of the activity of science is an application of that paradigm : given the description of some natural phenomena, to find the differential equations for processes that will produce the phenomena.

- A prize quote from Herbert Simon (Nobel prize in Economics, 1978)
- Given a scientific problem, find a Partial Differential Equation (PDE) describing it

(PDEs)

$$F(x, t, u, \nabla u, u_t, \nabla^2 u, u_{tt}, \ldots) = 0$$

find u.

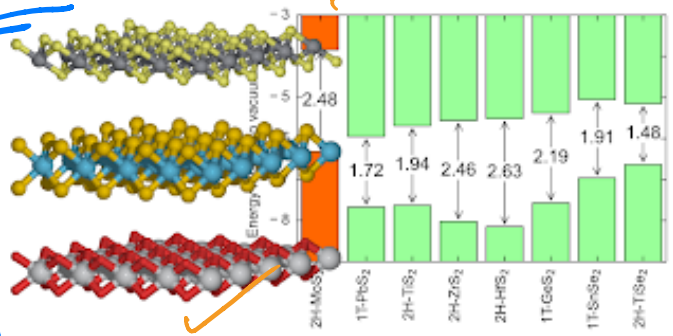
Partial Differential Equations (PDEs)

ψ - wave-function

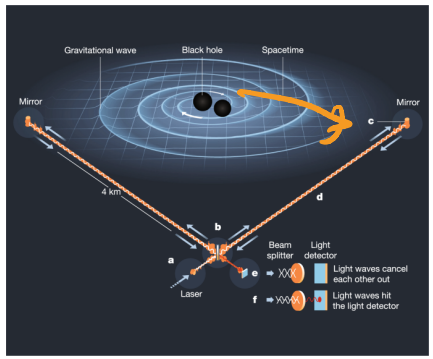
Schrödinger - Electronic structure.

$i\psi_t = H\psi$

$-\Delta\psi + V(x,t)$

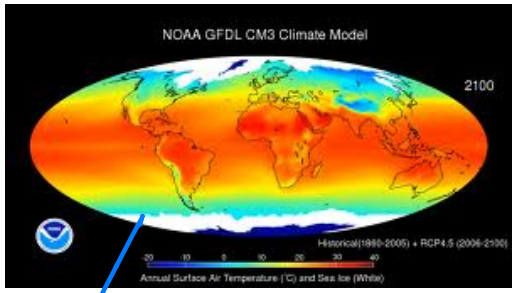


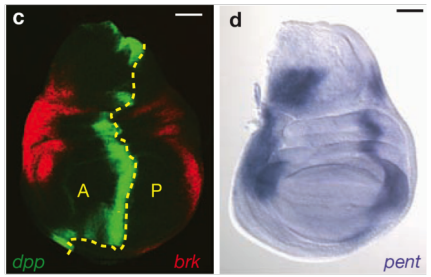
Atoms



- Einstein's Equations

Black Holes





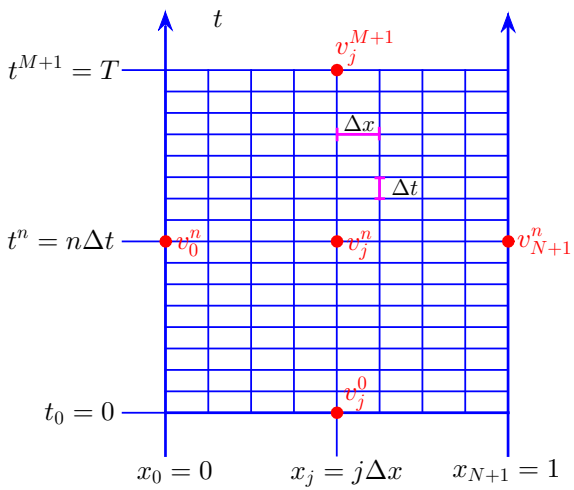
Climate

Fly wing

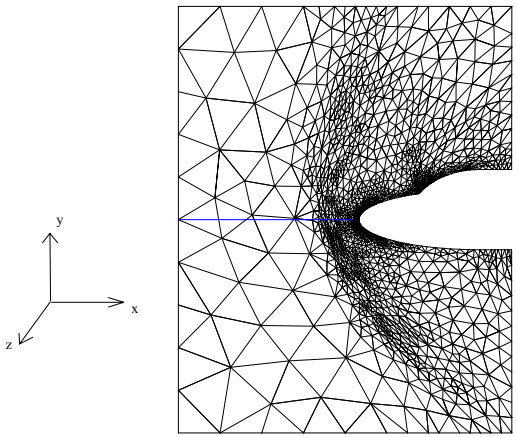
Euler or Navier-Stokes Eqns.

Reaction-Diffusion Equations

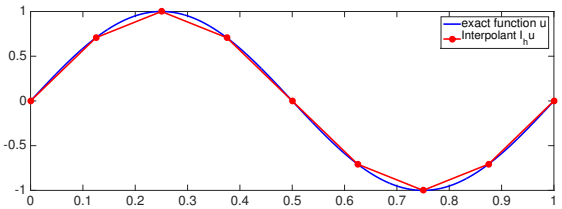
Traditional Numerical Methods



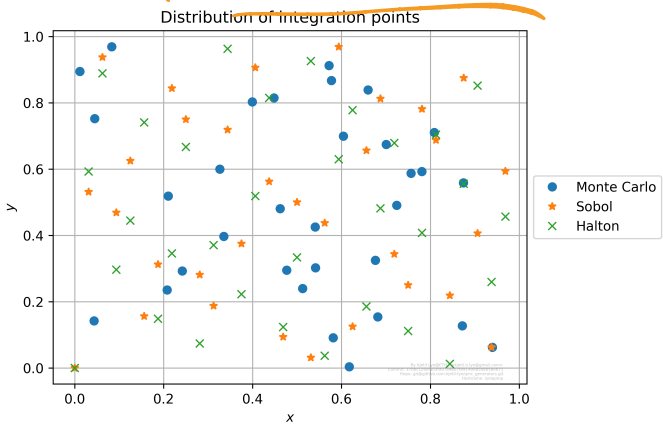
Finite Difference



Finite Volume



Finite Element



Collocation

- Different flavors of Runge-Kutta for Time Integration

Finite Difference Schemes (FDS) for the Heat Equation

1-D: $D = [0,1] \times [0,T]$ ~ Fourier ~ 1800:

u — Temperature

$u(x,t)$

$u_t = u_{xx}$

$u(0,t) = u(1,t) = 0$ — BCs

$u(x,0) = \bar{u}(x)$ — ICs

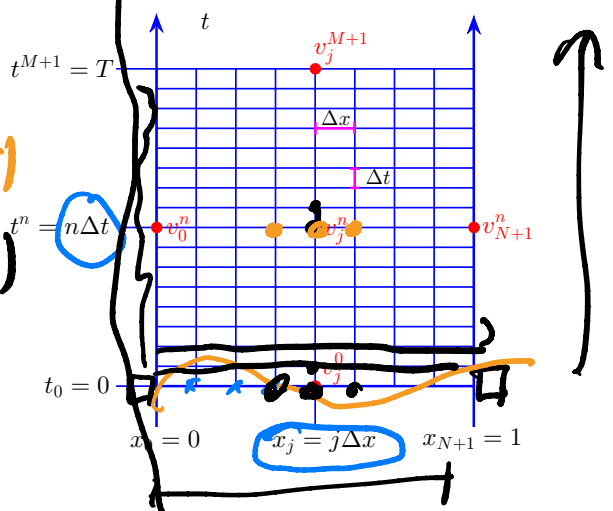
Inputs — functions — $a(x)$

Outputs — functions — $u(x,t)$

$u(x_j, t^n) \approx v_j^n$

Input — $\{v_j^0\}$

Output — v_j^n, $\forall\ 1 \le n \le M$



$D \approx \{x_j, t^n\}$

Δx - mesh size

Δt - time step

$$v_j^{n+1} = \left(1 - \frac{2\Delta t}{\Delta x^2}\right) v_j^n + \frac{\Delta t}{\Delta x^2}\left(v_{j-1}^n + v_{j+1}^n\right)$$

$$u_t \approx \frac{v_j^{n+1} - v_j^n}{\Delta t}, \quad u_{xx} \approx \frac{v_{j+1}^n - 2v_j^n + v_{j-1}^n}{\Delta x^2}$$

$u_t = u_{xx} \qquad u(x,t)$

$$u_j^{n+1} = \left(1 - \frac{2\Delta t}{\Delta x^2}\right) u_j^n + \frac{\Delta t}{\Delta x^2}\left(u_{j-1}^n + u_{j+1}^n\right)$$

Error: $\Delta x \sum_j |u(x_j, t^n) - u_j^n| = \varepsilon^n, \quad \varepsilon^{\Delta x} = \max_n \varepsilon^n$

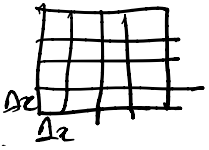
$$\varepsilon^{\Delta x} \sim O(\Delta t + \Delta x^2)$$

$$\Delta t \sim O(\Delta x^2)$$

Stability:

$$\varepsilon^{\Delta x} \sim O(\Delta x^2)$$

$u_t = u_{xx} + u_{yy}$ — FDS:



$$\#pt \sim \frac{1}{\Delta x^2}$$

d-dimensions:

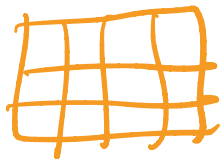
$$u_t = \Delta u; \quad \Delta u = \sum_{j=1}^{d} u_{x_j x_j}$$

$$\#(pts) \sim \frac{1}{\Delta x^d} \cdot \frac{1}{\Delta t} \sim \frac{1}{\Delta x^d} \frac{1}{\Delta x^2}$$

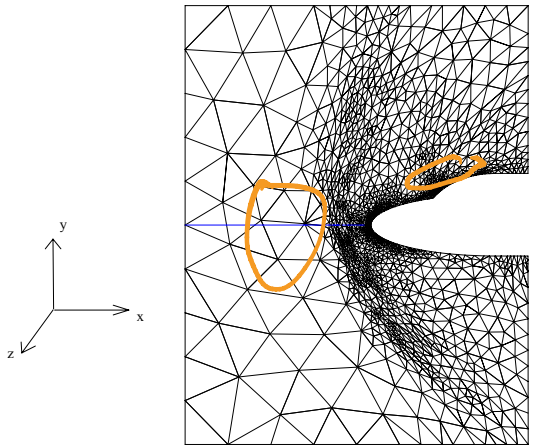
Computational work $\sim \frac{1}{\Delta x^{(d+2)}}$

Work grows exponentially with dimension

Issues with Numerical Simulations



- High-quality Grid Generation:



- Multiscale, MultiPhysics problems.
- High-Dimensional Problems

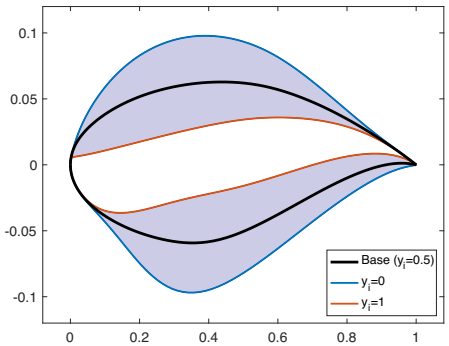
PDEs in high dimensions: $d \geq 4$

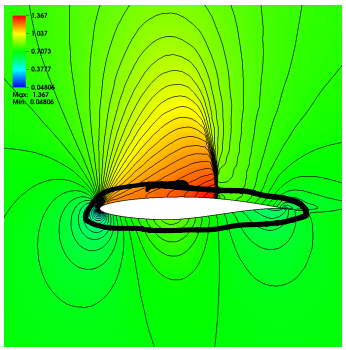
- Boltzmann Equation ($d = 7$)
- Radiative Transfer Equation ($d \geq 5$)
- Computational Finance: Black-Scholes ($d >> 1$)
- Computatational Chemistry: Schrödinger ($d >> 1$).

Many Query Problems: Optimal Design

→ Multiple calls to the PDE solver

- Example: Flow past airfoils (Euler or Navier-Stokes)
- Observables: Lift, Drag
- Uncertain parameters: Incident Mach Number, Angle of Attack, Pressure, Shape defects.
- Design parameters: Shape via Hicks-Henne functions. ~ 50 para





— Compressible Eules eqs

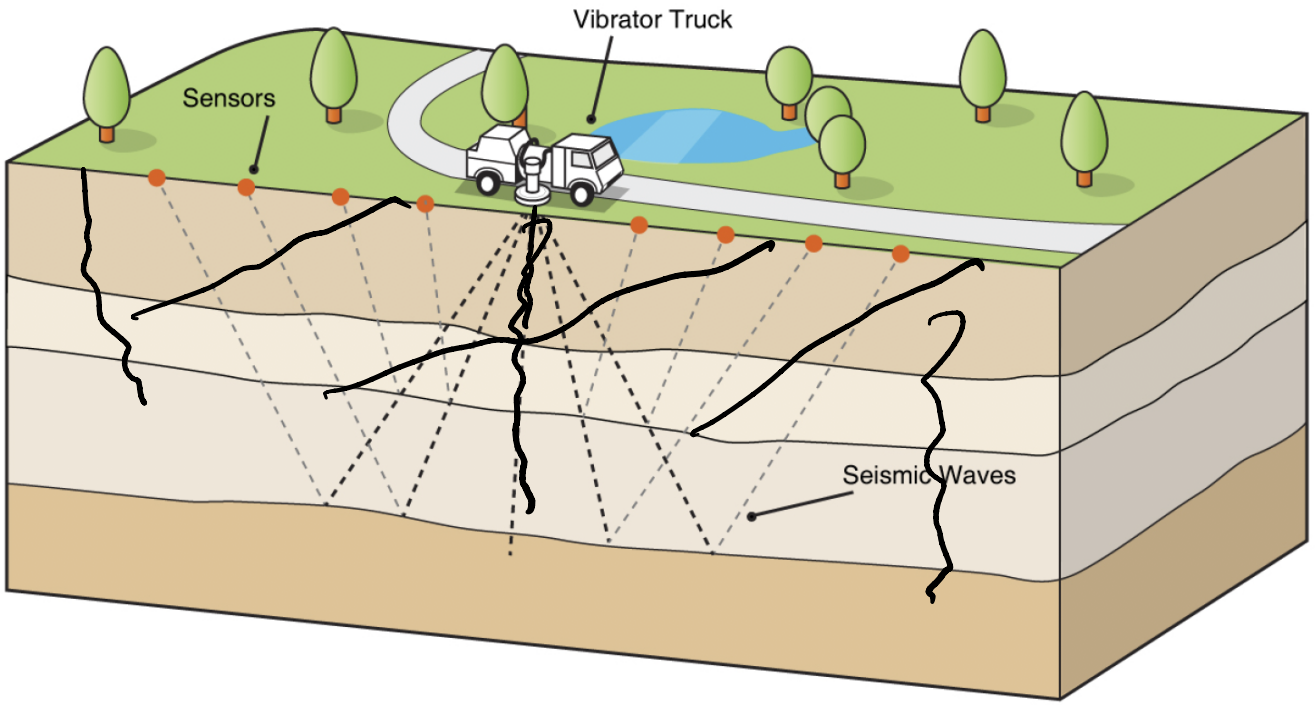
~ 20-30 ms

Many query Problems: Inverse Problem

Wave Equation:

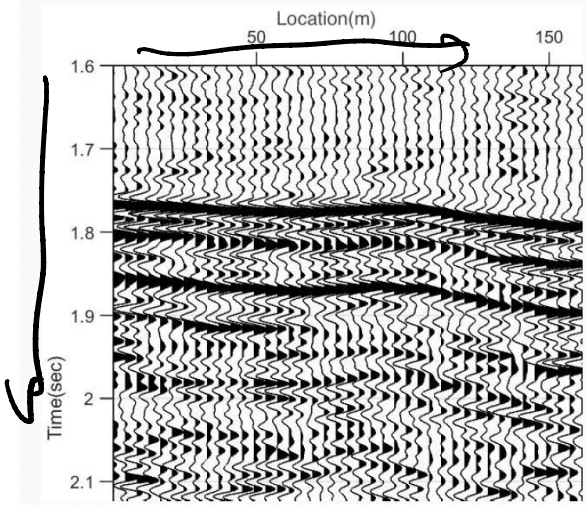
$$u_{tt} - \frac{1}{c^2(x)} \Delta u = 0$$

+ BCs



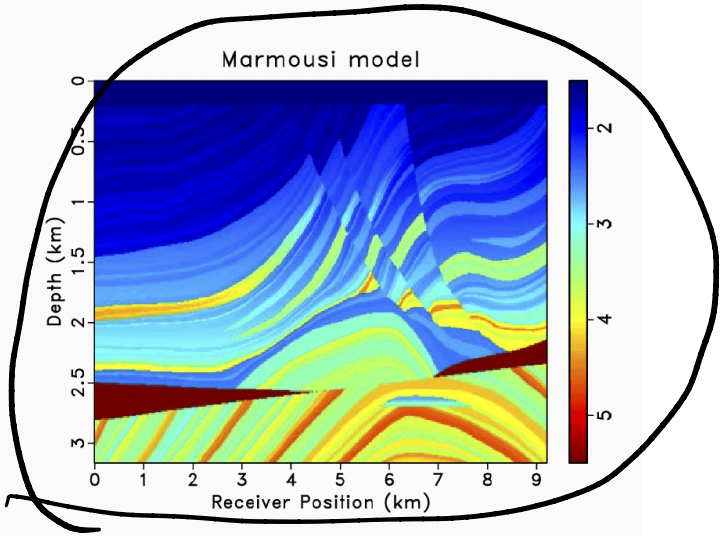
Inverse Problem for Wave Equation

Optimization



Waveform measurements from receivers at the surface

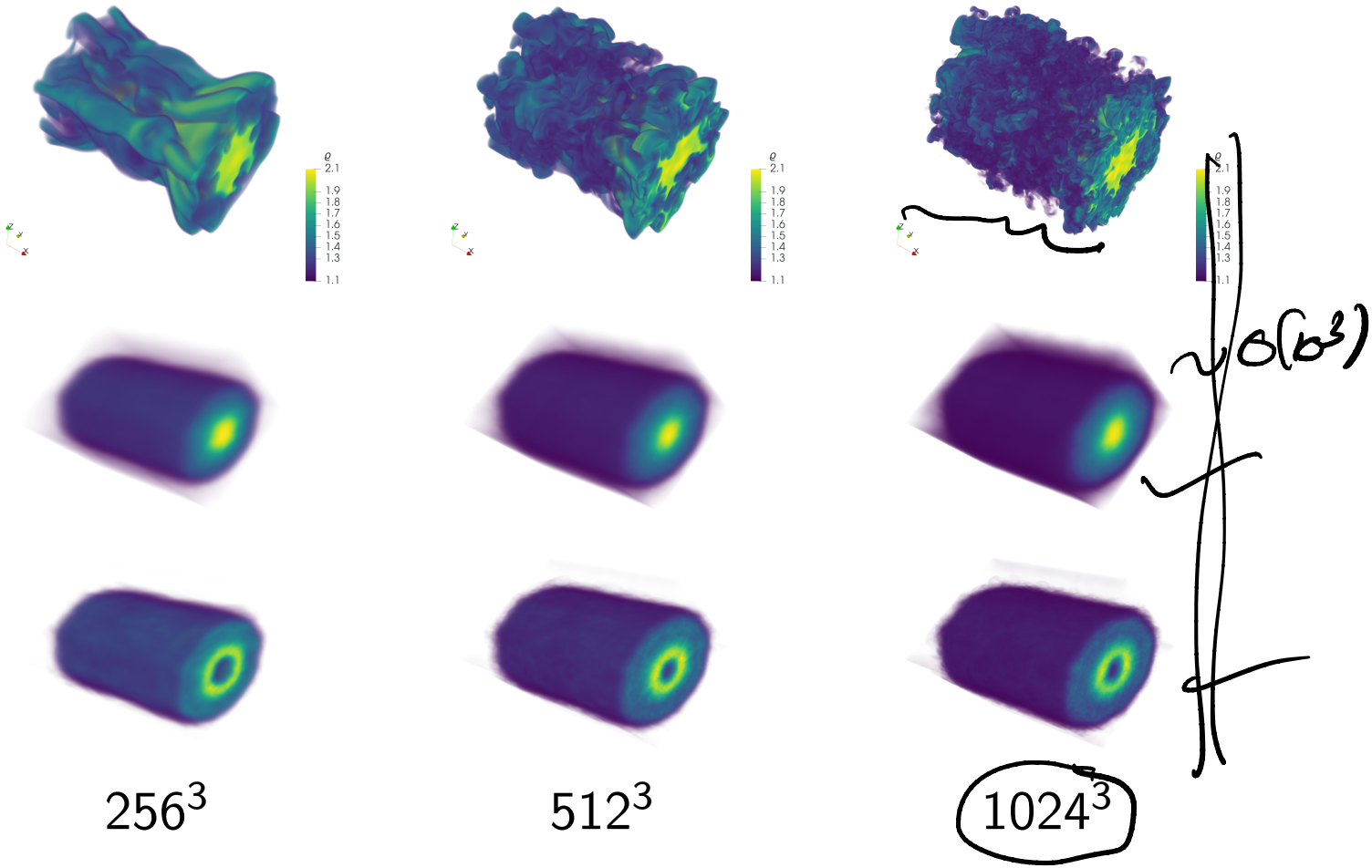
Invert



Subsurface properties (i.e., wave velocity or material density)

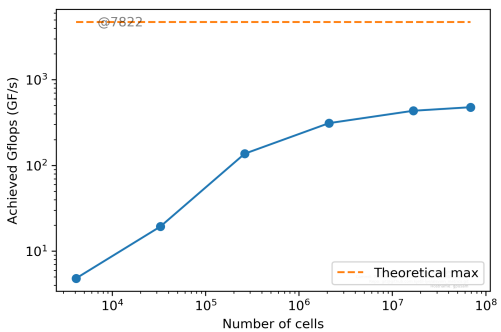
Many Query Problems: UQ

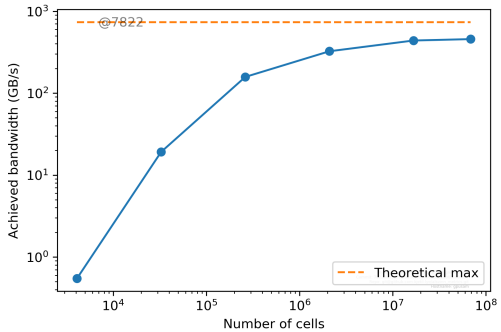
→ Uncertainty Quantification

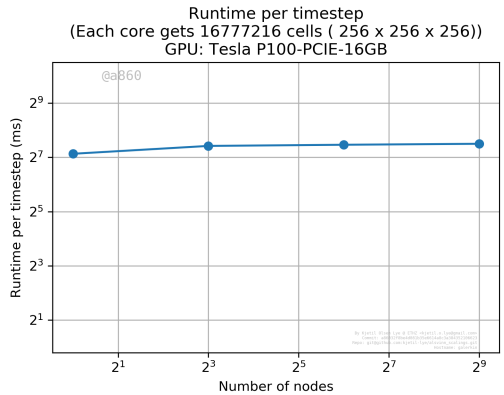


A little secret: Computational cost

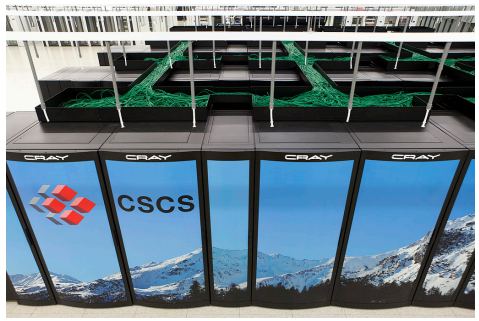
- ALSVINN: A custom-made highly efficient multi-GPU code







- 280 NH for a 1024^3 run on PIZ DAINT (15th in Top500)



- Ensemble simulation costs 1 Mil CHF !!!

Issue with traditional PDE solvers

Very high:

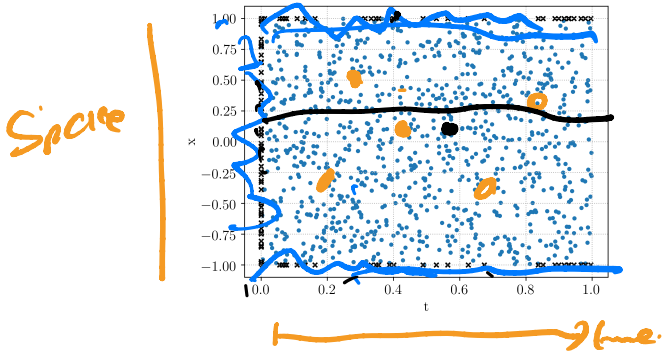
- Computational Cost (Many Query Problems)
- Feasibility (High-dimensional Problems)
- Systems with Incomplete Physics but lots of data.
- Most of the rest of the course deals with finding alternative strategies based on deep learning.

Physics Informed Neural Networks (PINNs)

Qn: Can one forget Traditional numerical methods and solve PDEs only with NNs ?

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 / Backprop.
- Loss function:

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

Physics Informed Neural Networks

↳ A different (PDE-informed) loss function.

- Variants of PINNs stem from Dissanayake, Phan-Thien, 1994.
- Also in Lagaris et al, mid 1990s. ~ 1996-97
- Reintroduced by Raissi, Perdikaris, Karniadakis, 2017.
- Extensively developed by Karniadakis and collaborators.
- 100 – 1000s of papers on PINNs already.

— In principle "meshless"

— Easy to Implement

— Scalable: — dimensions: — different PDEs

Burgers Equation

$u_t + uu_x = \nu u_{xx}$

$(x,t) \longmapsto u_\theta(x,t)$

$(x,t) \longmapsto \partial_x u_\theta(x,t)$

PDE forward problem

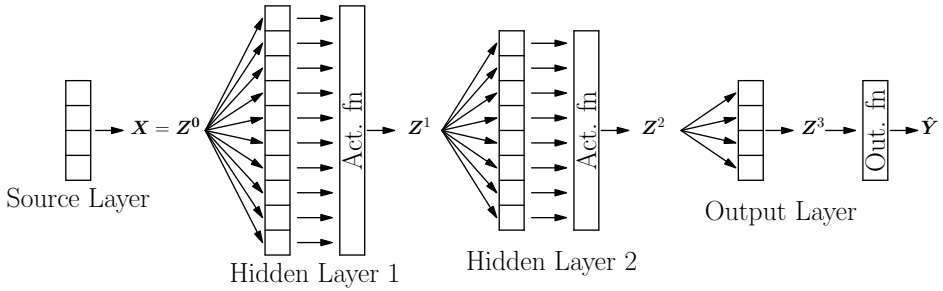
- Let X, Y be Banach spaces with $Y = L^p(\mathbb{D}; \mathbb{R}^m)$, with $1 \leq p < \infty$
- $X^* \subset X, Y^* \subset Y$
- $\mathbb{D} = D$ or $\mathbb{D} = D \times (0, T)$, with $D \subset \mathbb{R}^d$
- Abstract PDE is

$$\mathcal{D}(\mathrm{u}) = \mathrm{f},$$

Heat Equation
Navier Stokes
Eulers
$\mathcal{D}(u) = \mathcal{D}(u, \nabla u, \nabla^2 u, \ldots)$

- $\mathcal{D} : X^* \mapsto Y^*$ is the Differential operator .
- $\mathrm{f} \in Y^*$ is the input.
- Boundary (Initial) conditions are implicit.

Deep Neural networks



- $u^*(y) = \sigma_o \odot C_K \odot \sigma \odot C_{K-1} \ldots\ldots\ldots \odot \sigma \odot C_2 \odot \sigma \odot C_1(y).$
- At the k-th Hidden layer: $y^{k+1} := \sigma(C_k y^k) = \sigma(W^k y^k + B^k)$
- Parameters: $\theta = \{W_k, B_k\} \in \Theta$.
- Scalar Activation function σ
- Sigmoid, Tanh

PINNs for the PDE

- For Parameters $\theta \in \Theta$, $u_\theta : \mathbb{D} \mapsto \mathbb{R}^m$ is a DNN, with $u_\theta \in X^*$
- Aim: Find $\theta \in \Theta$ such that $u_\theta \approx u$ (in suitable sense).
- Compute PDE Residual by Automatic Differentiation:

$$\mathcal{R} := \mathcal{R}_\theta(y) = \mathcal{D}(u_\theta(y)) - f(y), \; y \in \mathbb{D} \quad \mathcal{R}_\theta \in Y^*, \quad \forall \theta \in \Theta$$

- PINNs are minimizers of $\|\mathcal{R}_\theta\|_Y^p \sim \int_{\mathbb{D}} |\mathcal{R}_\theta(y)|^p \, dy$
- Replace Integral by Quadrature !
- Let $\mathcal{S} = \{y_i\}_{1 \le i \le N}$ be quadrature points in $\mathbb{D}$, with weights w_i
- PINN for approximating PDE is defined as $u^* = u_{\theta^*}$ such that

$$\theta^* = \arg\min_{\theta \in \Theta} \sum_{i=1}^{N} w_i |\mathcal{R}_\theta(y_i)|^p$$

- Minimize Very high-d Non-Convex loss with ADAM,L-BFGS

Do PINNs work ?

- Multi-D Heat Equation
- PINN with Depth 4, Width 20, Interior training points 2^{16}, Boundary points 2^{15}

Dimension	Training Error	Generalization error
1	2.8×10^{-5}	0.0035%
5	0.0002	0.016%
10	0.0003	0.03%
20	0.006	0.79%
50	0.006	1.5%
100	0.004	2.6%

- No Curse of dimensionality !!

When and Why do PINNs work for a PDE $\mathcal{D}(u) = f$?

- PDE solution u, DNN u_θ with parameters $\theta \in \Theta$
- AIM is to ensure small Total Error:

$$\mathcal{E}(\theta) := \|u - u_\theta\|_p$$

- PINNs may not have access to samples from Exact Solution u
- On the other hand, PINNs minimize PDE Residual:

$$\mathcal{E}_G(\theta) = \|\mathcal{R}_\theta\|_p = \|\mathcal{D}(u_\theta) - f\|_p$$

- In practice, we only have access to Training Error:

$$\mathcal{E}_T(\theta) = \left(\sum_{i=1}^{N} w_i |\mathcal{R}_\theta(y_i)|^p \right)^{\frac{1}{p}}$$

Key Theoretical Questions

- Is the PDE Residual small in the class of Neural Networks that approximate the exact solution u ? i.e.

$$\text{Does } \exists \hat{\theta}, \tilde{\theta} \in \Theta, \quad \mathcal{E}_G(\hat{\theta}), \mathcal{E}_T(\tilde{\theta}) < \epsilon, \text{ and } \mathcal{E} \sim \mathcal{O}(\epsilon) \text{ ?.}$$

- Does small PINN Residual $\Rightarrow$ small Total Error ? i.e.,
- Can we derive a bound of the form:

$$\mathcal{E}(\theta) \leq C\mathcal{E}_G(\theta), \quad \forall \theta \in \Theta$$

- Does small Training Loss $\Rightarrow$ small PINN Residual ? i.e.,
- Can we derive a bound of the form ?

$$\mathcal{E}_G(\theta) \leq \overline{C}\left(\mathcal{E}_T(\theta), N\right) \sim o\left(N^{-1}\right) \quad \forall \theta \in \Theta$$

On the smallness of PDE Residuals in the class of Neural Networks

- For sufficiently smooth u solving $\mathcal{D}(\mathrm{u}) = \mathrm{f}$ observe that

$$\mathcal{E}_G(\theta) = \|\mathcal{D}(\mathrm{u}_\theta) - \mathrm{f}\|_p = \|\mathcal{D}(\mathrm{u}_\theta) - \mathcal{D}(\mathrm{u})\|_p \leq C(\mathrm{u}, \mathrm{u}_\theta) \|\mathrm{u} - \mathrm{u}_\theta\|_{W^{s,p}}$$

- Here s depends on the number of derivatives in the Differential Operator $\mathcal{D}$.
- Novel DNN approximation results in high-order Sobolev spaces:
- For example in (DeRyck,Lanthaler,SM, 2021) can be used to conclude:

$$\exists \hat{\theta} \in \Theta, \quad \|\mathrm{u} - \mathrm{u}_{\hat{\theta}}\|_{W^{s,p}} < \epsilon$$

- smoothness of u $\Rightarrow$ small PINN Residuals.

On bounds on total error in terms of Residuals

- Sufficient Conditions of SM, Molinaro, 2021:
- Coercivity of the PDE $\mathcal{D}\mathrm{u} = f$: for any $\mathrm{u}, \bar{\mathrm{u}} \in X^*$:

$$\|\mathrm{u} - \bar{\mathrm{u}}\|_p \leq C_{pde}(\bar{\mathrm{u}}, \mathrm{u})\|\mathcal{D}(\bar{\mathrm{u}}) - \mathcal{D}(\mathrm{u})\|_p$$

- Coercivity $\Rightarrow$ Bounds in terms of Residuals as,

$$\begin{aligned}
\mathcal{E}(\theta) &= \|\mathrm{u}_\theta - \mathrm{u}\|_p, \\
&\leq C_{pde}(\mathrm{u}, \mathrm{u}_\theta)\|\mathcal{D}(\mathrm{u}_\theta) - \mathcal{D}(\mathrm{u})\|_p \quad (\text{Coercivity}), \\
&\leq C_{pde}(\mathrm{u}, \mathrm{u}_\theta)\|\mathcal{D}(\mathrm{u}_\theta) - f\|_p \quad \text{as } \mathcal{D}(\mathrm{u}) = f, \\
&\leq C_{pde}(\mathrm{u}, \mathrm{u}_\theta)\mathcal{E}_G(\theta) \quad (\text{Definition of } \mathcal{E}_G)
\end{aligned}$$

- Training Error $\mathcal{E}_T$ is Quadrature Approximation of $\mathcal{E}_G$:

$$\mathcal{E}_G \leq \mathcal{E}_T + C_{quad}(u_{\theta^*})^{\frac{1}{p}} N^{-\frac{\alpha}{p}} \quad \text{quadrature error},$$

A Strategy for PINN Error Bounds SM, Molinaro, 2021

- Use smoothness of exact solution to u of PDE $\mathcal{D}(\mathrm{u}) = f$
- And DNN approximation results in high-order Sobolev spaces to show that:

$$\exists \theta \in \Theta : \quad \mathcal{E}_G(\theta), \mathcal{E}_T(\theta) \leq \underline{C}(\mathrm{u}, \mathrm{u}_\theta) \|\mathrm{u} - \mathrm{u}_\theta\|_{W^{s,p}}.$$

- Use Coercivity of a given PDE to show that

$$\|\mathrm{u} - \mathrm{u}_\theta\|_p \leq \overline{C}(\mathrm{u}, \mathrm{u}_\theta) \mathcal{E}_G(\theta), \quad \forall \theta \in \Theta.$$

- Use Quadrature bounds to show that,

$$\mathcal{E}_G \leq \mathcal{E}_T + C_{quad}(u_{\theta^*})^{\frac{1}{p}} N^{-\frac{\alpha}{p}}$$

- Prove explicit growth bounds on the constants $\underline{C}, \overline{C}, C_{quad}$ in terms of Neural Network architecture and number of collocation points.

Ex: 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 in practice, $d >> 1$ (Very high-dimensional)

Error Bounds: De Ryck, SM, 2021.

- Carried out the afore-described strategy in this case.
- $\exists$ Tanh PINN $\hat{u}$ of size $\mathcal{O}(\epsilon^{-\alpha(d)})$ with $\alpha \sim \mathrm{poly}(d)$

$$\mathcal{E}_{G,T}(\hat{\theta}) \sim \epsilon,$$

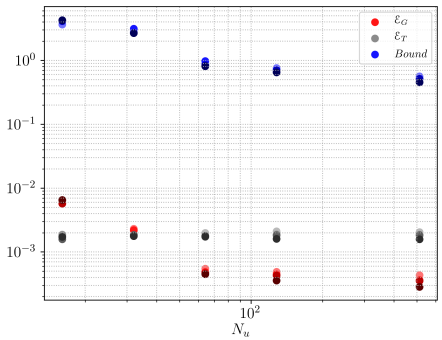
- Uses Dynkin's formula to overcome curse of dimensionality.
- Coercivity by Stability of this linear PDE:

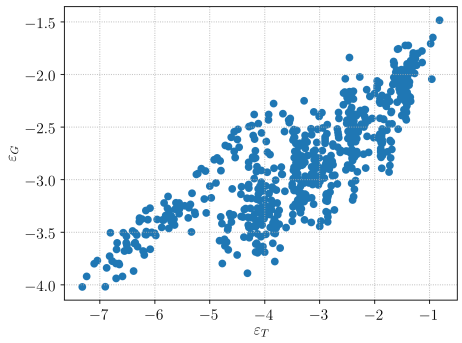
$$\|u - u_\theta\|_2 \leq C \left(\|\mathcal{R}_{int,\theta}\| + \|\mathcal{R}_{sb,\theta}\|^{\frac{1}{2}} \right)$$

- Use Hoeffding's inequality + Lipschitz bounds on u_θ to prove Quadrature bounds with M (No. of Neurons), W weights and N number of collocation pts.

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

Numerical Results





- A computable upper bound shows same asymptotic behavior with increasing N as the computed error.
- $\mathcal{E}_G$ and $\mathcal{E}_T$ are tightly correlated as small $\mathcal{E}_T \Rightarrow$ small $\mathcal{E}_G$.
- Useful for selecting hyperparameters.
- Results also verify that $\mathcal{E}_G \sim \sqrt{\mathcal{E}_T}$ as predicted by theory !!