High-dim PDEs

$d = \dim$

$d = 4$ medium
$d = 10^4, 10^5$ high.

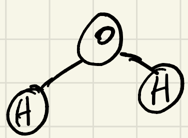
Schrödinger Equ

$$i\hbar \frac{\partial}{\partial t}\Psi = H\Psi$$

Ψ ← wave function

$\Psi(x,t) \in \mathbb{C}$ x = state

$|\Psi(x,t)|^2$ = probability that system is in state x at time t



pos, mom of each atom

n atoms $x \in \mathbb{R}^{6n}$

– HJB (Hamilton–Jacobi–Bellman)

$$\frac{\partial V}{\partial t} + \min_u \left(\frac{\partial V}{\partial x} \cdot F(x,u) + C(x,u) \right) = 0$$

$V(x,t)$ = value fcn = cost to go in state x

$$= \min_u \left[\int_t^T C(x(t), u(t))\, dt + D(x(T)) \right]$$

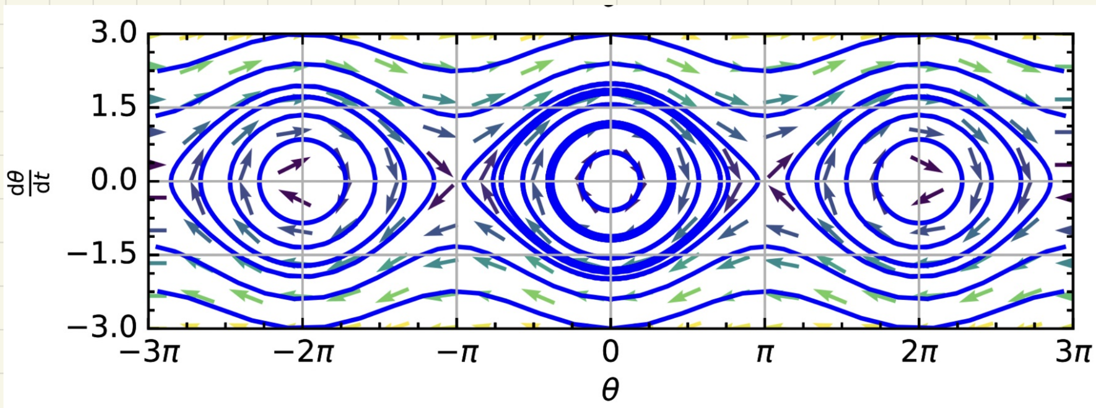
with $\dot{x} = F(x,u)$

x = system state
u = control input
C = cost
D = terminal cost

(x, y, v, θ) $\quad x \in \mathbb{R}^4$

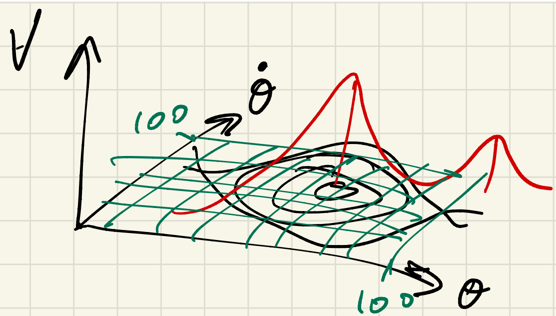
How does cost scale with d $\qquad \Psi(x,t) \quad x \in \mathbb{R}^d$

$V(x,t) \quad t \in \mathbb{R}$



$\ddot{\theta} = -\sin\theta + u$

$C = |u|^2$



cost $\sim 100^2 = 10^4$

$\sim 100^d$

car $\sim 100^4 = 10^8$

drone $\sim 100^9 = 10^{18}$

curse of dimensionality : cost $\sim \exp(d)$

solving high-dim PDEs with PINNs

SDGD - stochastic dimension gradient descent

(Hu et al, 2023)

Idea: $\nabla^2 u(x) = f(x) \qquad x \in \Omega \subset \mathbb{R}^d$

$$\sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2} u_\theta(x) = f(x)$$

$$l_\theta(x) = \frac{1}{2}\left(\sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2} u_\theta(x) - f(x)\right)^2$$

$$\nabla_\theta l_\theta(x) = \left(\sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2} u_\theta(x) - f(x)\right)\left(\sum_{i=1}^{d} \nabla_\theta \frac{\partial^2}{\partial x_i^2} u_\theta(x)\right)$$

$$L = \sum_j l_\theta(x^j) \qquad \nabla_\theta L = \sum_j \nabla_\theta l_\theta(x^j)$$

choose subsets $I, J \subset \{1, \dots, d\}$

$$\nabla_\theta l_\theta(x) \approx \left(\frac{d}{|J|} \sum_{i \in J} \frac{\partial^2}{\partial x_i^2} u_\theta(x) - f(x) \right) \left(\frac{d}{|I|} \sum_{i \in I} \nabla_\theta \frac{\partial^2}{\partial x_i^2} u_\theta(x) \right)$$

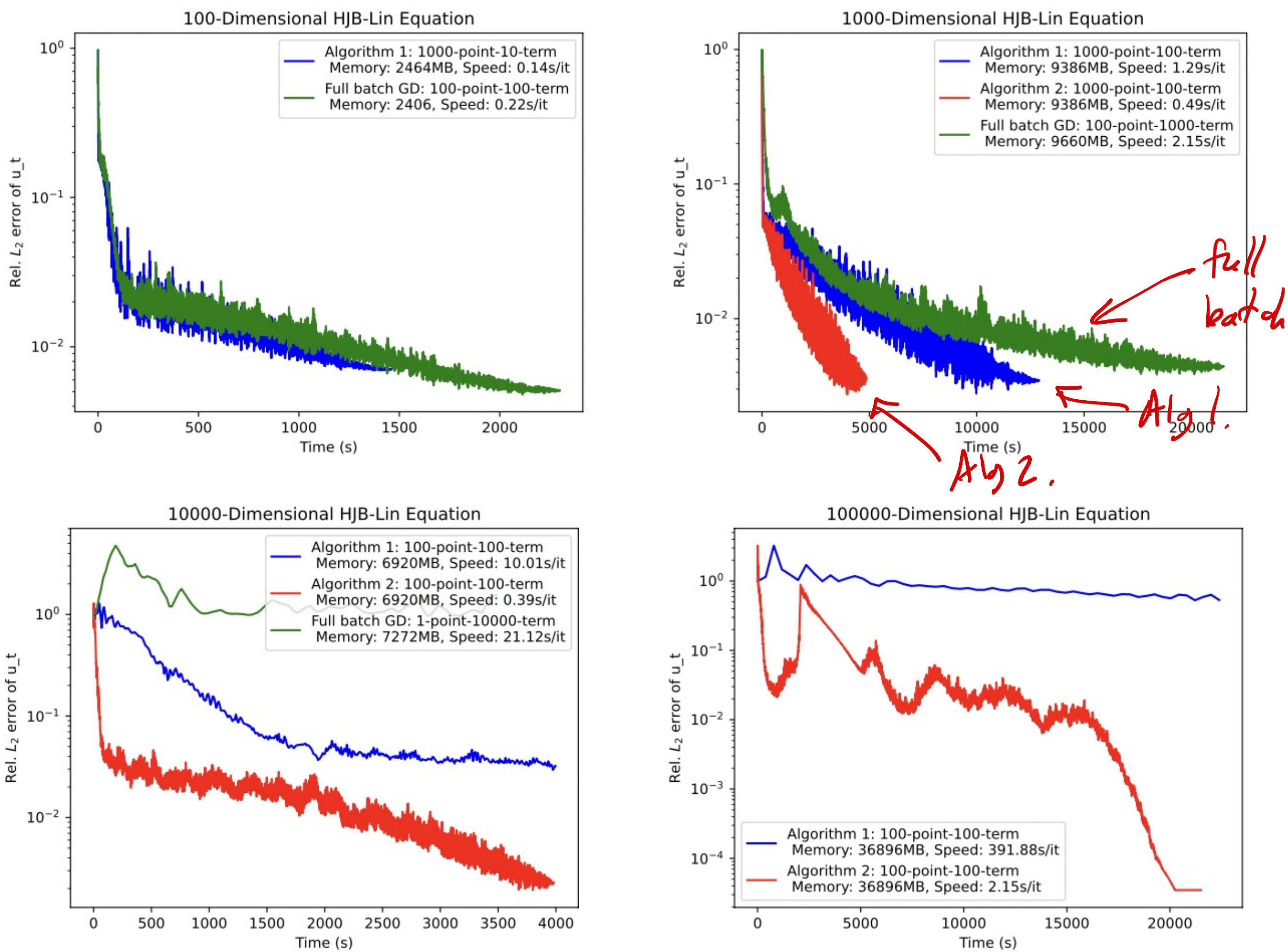
rescale

$$E[\nabla_{I,J,\theta}\, l_\theta(x)] = \nabla_\theta\, l_\theta(x)$$

Alg 1. $\rightarrow$ only subsample I

Alg 2. $\rightarrow$ both I, J

In PyTorch : compute $\sum_{i \in I} \frac{\partial^2}{\partial x_i^2} u_\theta(x) \rightarrow$ grad.


100-Dimensional HJB-Lin Equation
Algorithm 1: 1000-point-10-term
Memory: 2464MB, Speed: 0.14s/it
Full batch GD: 100-point-100-term
Memory: 2406, Speed: 0.22s/it
Rel. L_2 error of u_t
Time (s)
1000-Dimensional HJB-Lin Equation
Algorithm 1: 1000-point-100-term
Memory: 9386MB, Speed: 1.29s/it
Algorithm 2: 1000-point-100-term
Memory: 9386MB, Speed: 0.49s/it
Full batch GD: 100-point-1000-term
Memory: 9660MB, Speed: 2.15s/it
Rel. L_2 error of u_t
Time (s)
full batch
Alg 1.
Alg 2.
10000-Dimensional HJB-Lin Equation
Algorithm 1: 100-point-100-term
Memory: 6920MB, Speed: 10.01s/it
Algorithm 2: 100-point-100-term
Memory: 6920MB, Speed: 0.39s/it
Full batch GD: 1-point-10000-term
Memory: 7272MB, Speed: 21.12s/it
Rel. L_2 error of u_t
Time (s)
100000-Dimensional HJB-Lin Equation
Algorithm 1: 100-point-100-term
Memory: 36896MB, Speed: 391.88s/it
Algorithm 2: 100-point-100-term
Memory: 36896MB, Speed: 2.15s/it
Rel. L_2 error of u_t
Time (s)