Machine Learning: Mathematical Theory and Scientific Applications

Weinan E

Joint work with:

Jiequn Han, Arnulf Jentzen, Chao Ma, Zheng Ma,
Han Wang, Qingcan Wang, Lei Wu, Linfeng Zhang, Yajun Zhou

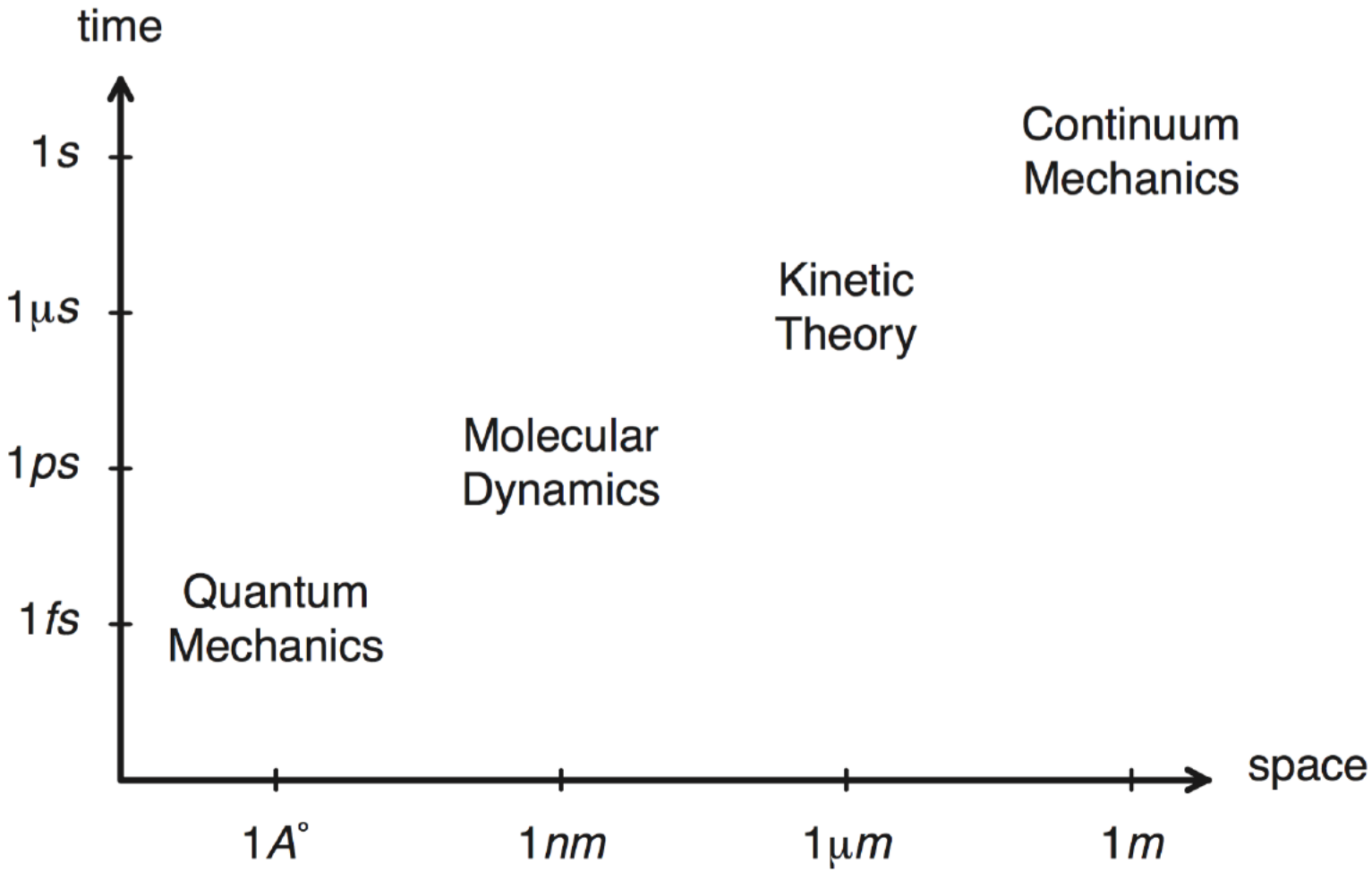
Roberto Car, Wissam A. Saidi

Outline

Outline

PDEs and fundamental laws of physics

- Navier-Stokes equations
- Boltzmann equation
- Schrödinger equation
- .....



Dirac's claim (1929)

"The underlying physical laws necessary for the mathematical theory of a large part of physics and the whole of chemistry are thus completely known, and the difficulty is only that the exact application of these laws leads to equations much too complicated to be soluble. "

For most practical applications, the difficulty is not in the fundamental laws of physics, but in the mathematics.

Outline

Numerical methods: 50's-80's

- finite difference
- finite element
- spectral methods
- .....

These have completely changed the way we do science, and to an even greater extend, engineering.

- gas dynamics
- structural analysis
- radar, sonar, optics
- control of flight vehicles, satellites
- .....

If the finite difference method was invented today, the shock wave that it will generate would be just as strong as the one generated by deep learning.

Many difficult problems remain

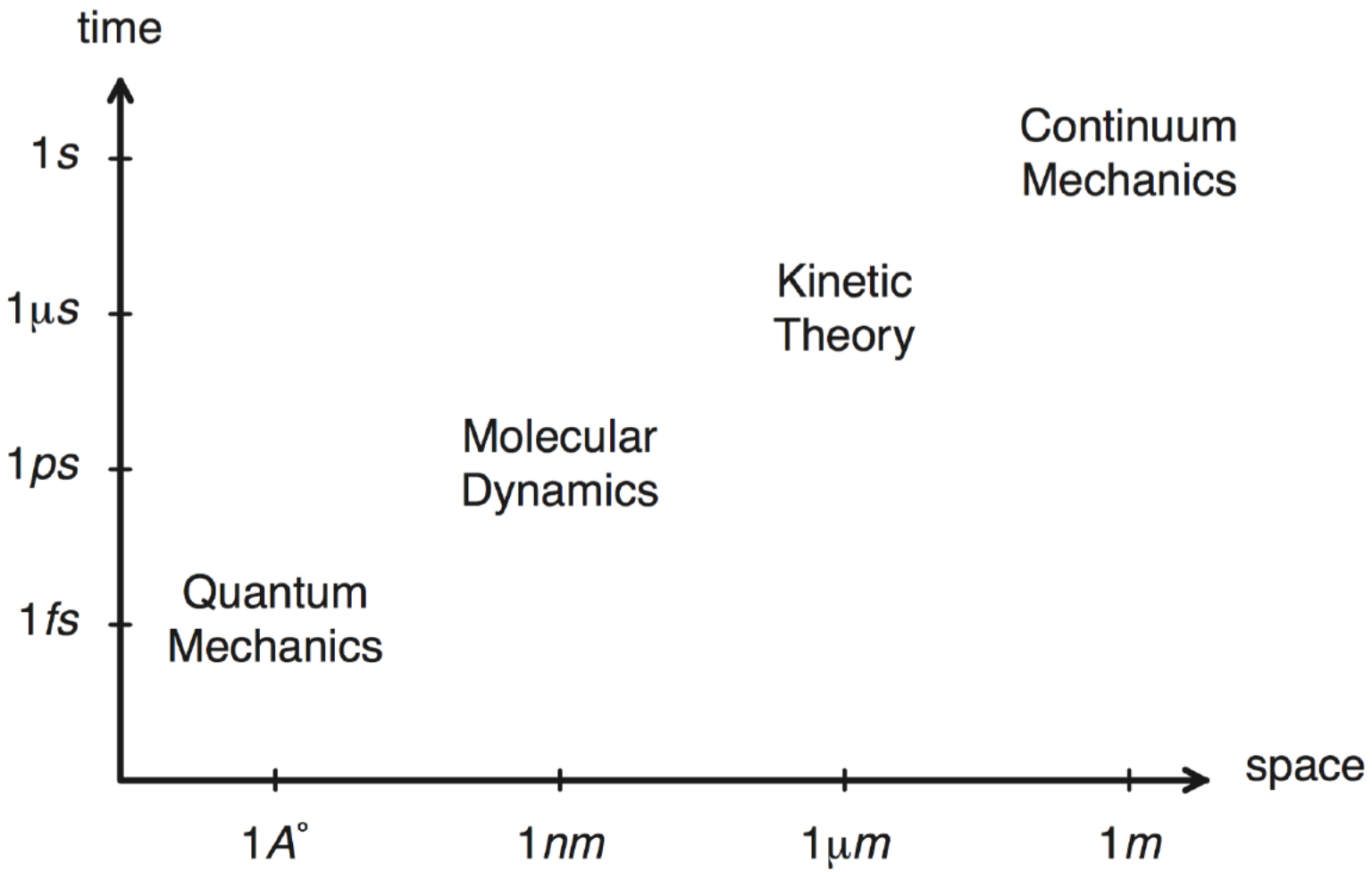
- many-body problems (classical and quantum, in molecular science)
- quantum control
- first principle-based drug and materials design
- protein folding
- turbulence, weather forecasting
- transitional flows in gas dynamics
- polymeric fluids
- plasticity
- control problems in high dimensions
- .....

Common feature of these problems: Dependence on many variables.

Curse of dimensionality: As the dimension grows, the complexity (or computational cost) grows exponentially.

Outline

Multiscale, multi-physics modeling: 90's till now



- works well when the macro- and micro-scales are very well separated
- not very effective when there are no separation of scales (e.g. turbulence problem)

Status summary

- Solved: low dimensional problems (few dependent variables)
- Unsolved: high dimensional problems (many dependent variables)

Machine learning, particularly deep learning, seems to be a powerful tool for high dimensional problems.

Outline

Example 1: Molecular dynamics

Traditional dilemma: accuracy vs cost.

$$E = E(\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_i, ..., \boldsymbol{x}_N),$$

$$m_i \frac{d^2 \boldsymbol{x}_i}{dt^2} = \boldsymbol{F}_i = -\nabla_{\boldsymbol{x}_i} E.$$

Two ways to calculate E and $\boldsymbol{F}$:

- Computing the inter-atomic forces on the fly using QM, e.g. the Car-Parrinello MD. Accurate but expensive:

$$E = \langle \Psi_0 | H_e^{KS} | \Psi_0 \rangle, \ \mu \ddot{\phi}_i = H_e^{KS} \phi_i + \sum_j \Lambda_{ij} \phi_j.$$

- Empirical potentials: efficient but unreliable. The Lennard-Jones potential:

$$V_{ij} = 4\epsilon[(\frac{\sigma}{r_{ij}})^{12} - (\frac{\sigma}{r_{ij}})^{6}], \ E = \frac{1}{2} \sum_{i \neq j} V_{ij}.$$

How can we represent (approximate) a function of many variables?

New paradigm:

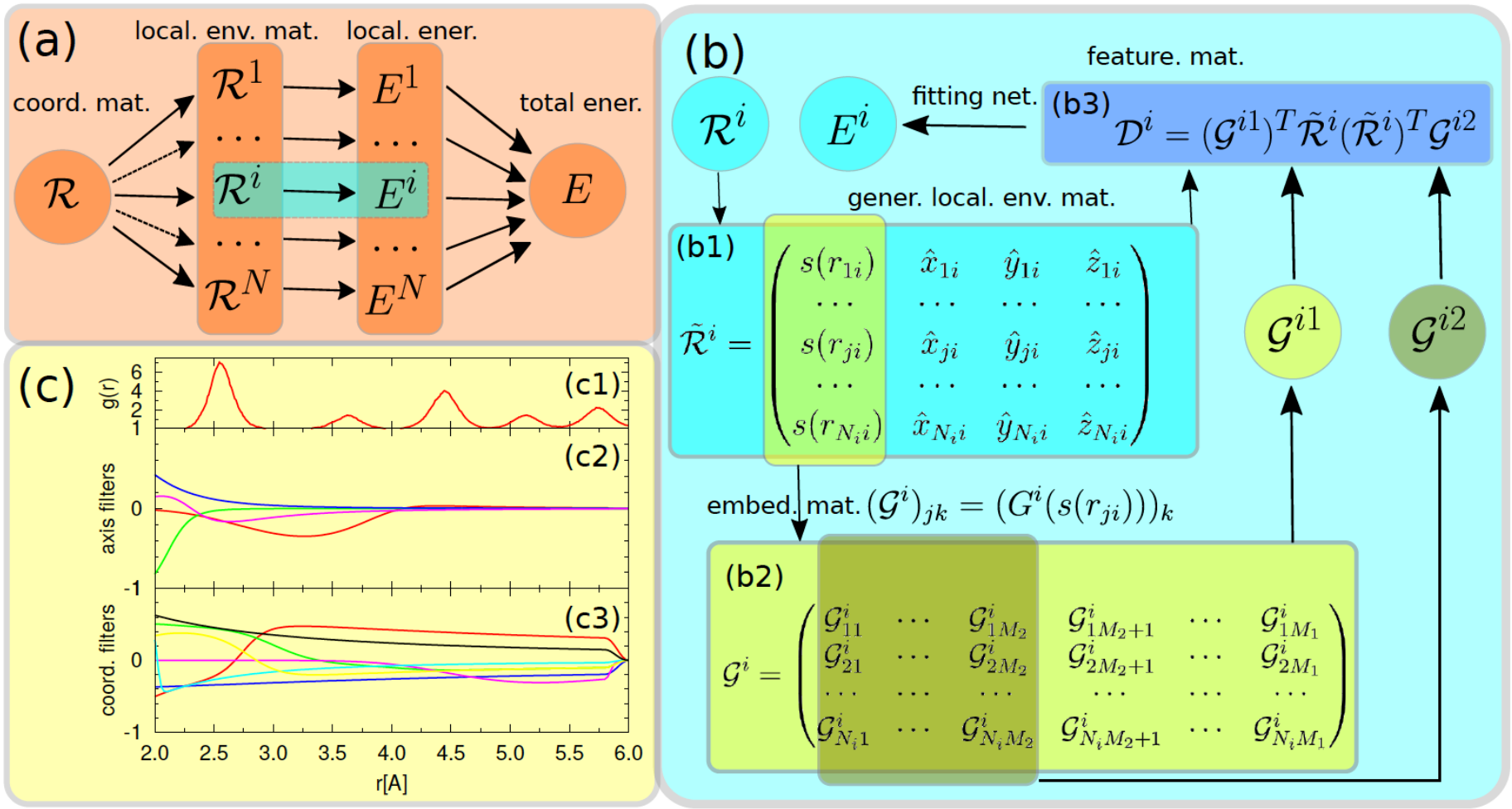
- quantum mechanics model – data generator
- machine learning – parametrize (represent) the model
- molecular dynamics – simulator

Issues (different from usual AI applications):

- preserving physical symmetries (translation, rotation, permutation)
- getting the "optimal data set"

Deep potential

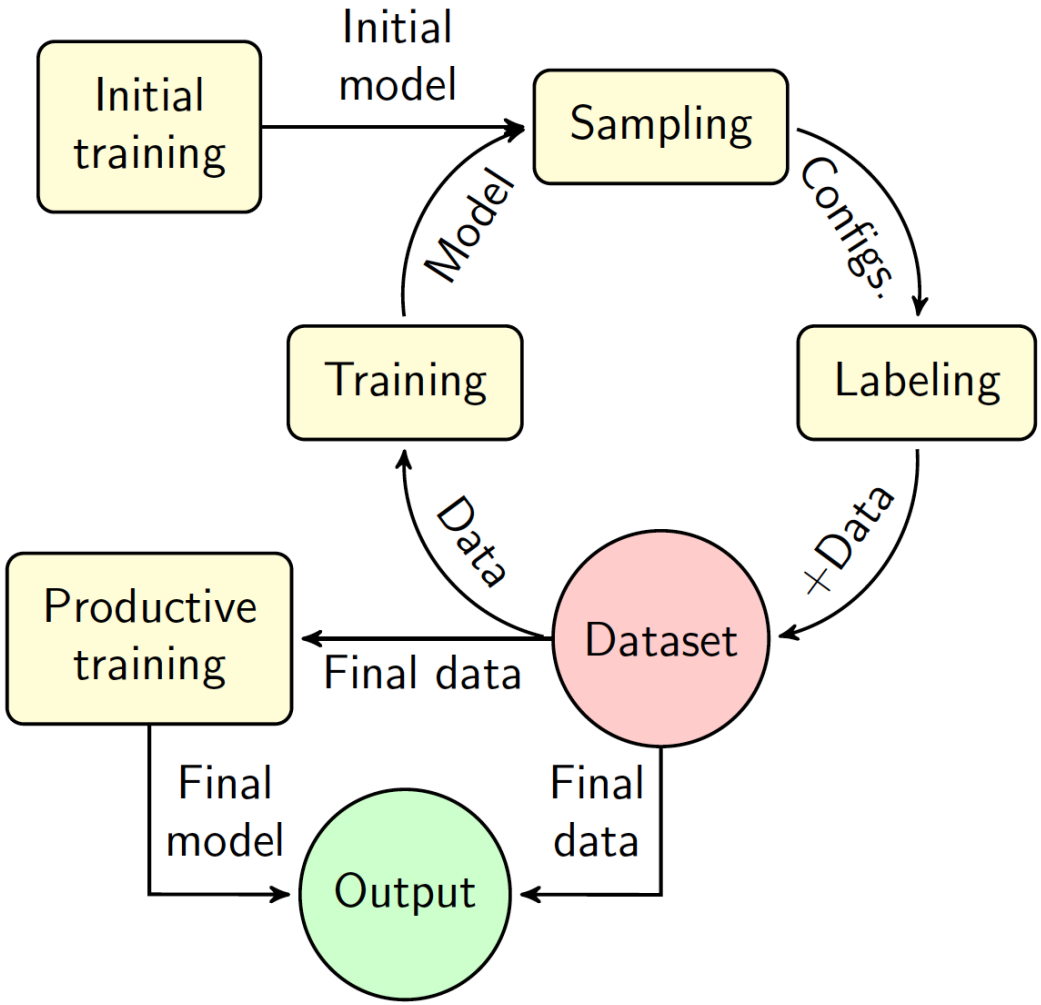
The whole sub-network consists of an encoding net $\mathcal{D}^i(\mathcal{R}^i)$ and a fitting net $E^i(\mathcal{D}^i)$.



(Rotation: $\tilde{\mathcal{R}}^i(\tilde{\mathcal{R}}^i)^T$, permutation: $(\mathcal{G}^{i1})^T\tilde{\mathcal{R}}^i$ and $(\tilde{\mathcal{R}}^i)^T\mathcal{G}^{i2}$.)

DeepPot-SE (arxiv: 1805.09003, NIPS 2018), see also Behler and Parrinello, PRL 2007.

DP-GEN: active learning for uniformly accurate model



Indicator: $\epsilon = \max_i \sqrt{\langle \| \boldsymbol{f}_i - \bar{\boldsymbol{f}}_i \|^2 \rangle}$, $\bar{\boldsymbol{f}}_i = \langle \boldsymbol{f}_i \rangle$ "Active Learning of Uniformly Accurate Inter-atomic Potentials for Materials Simulation." arXiv:1810.11890 (2018).

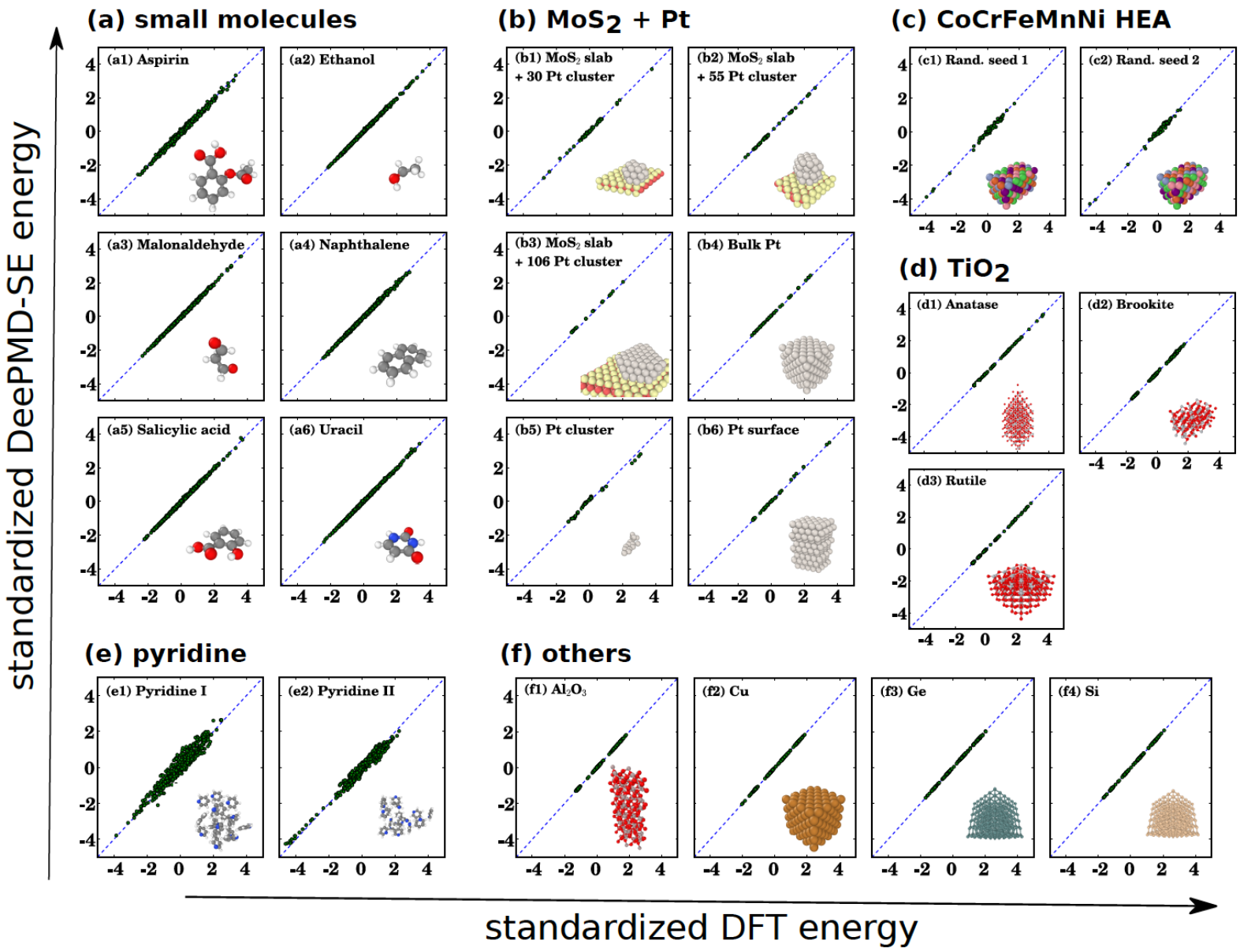
In addition, initialize the exploration with a variety of different initial configurations.

Systems			Al		Mg		Al-Mg alloy	
Type	Lattice	#atom	#Confs	#Data	#Confs	#Data	#Confs	#Data
Bulk	FCC	32	15,174,000	1,326	15,174,000	860	39,266,460	7,313
	HCP	16	15,174,000	908	15,174,000	760	18,999,900	2,461
	Diamond	16	5,058,000	1,026	5,058,000	543	5,451,300	2,607
	SC	8	5,058,000	713	5,058,000	234	2,543,940	667
Surface	FCC (100)	12	3,270,960	728	3,270,960	251	62,203,680	1,131
	FCC (110)	16[a],20[b]	3,270,960	838	3,270,960	353	10,744,2720	2,435
	FCC (111)	12	3,270,960	544	3,270,960	230	62,203,680	1,160
	HCP (0001)	12	3,270,960	39	3,270,960	109	62,203,680	176
	HCP $(10\bar{1}0)$	12	3,270,960	74	3,270,960	167	62,203,680	203
	HCP $(11\bar{2}0)$	16[a],20[b]	3,270,960	293	3,270,960	182	107,442,720	501
sum			60,089,760	6,489	60,089,760	3,689	529,961,760	18,654

[a]Pure Al
[b]Mg and Al-Mg alloy

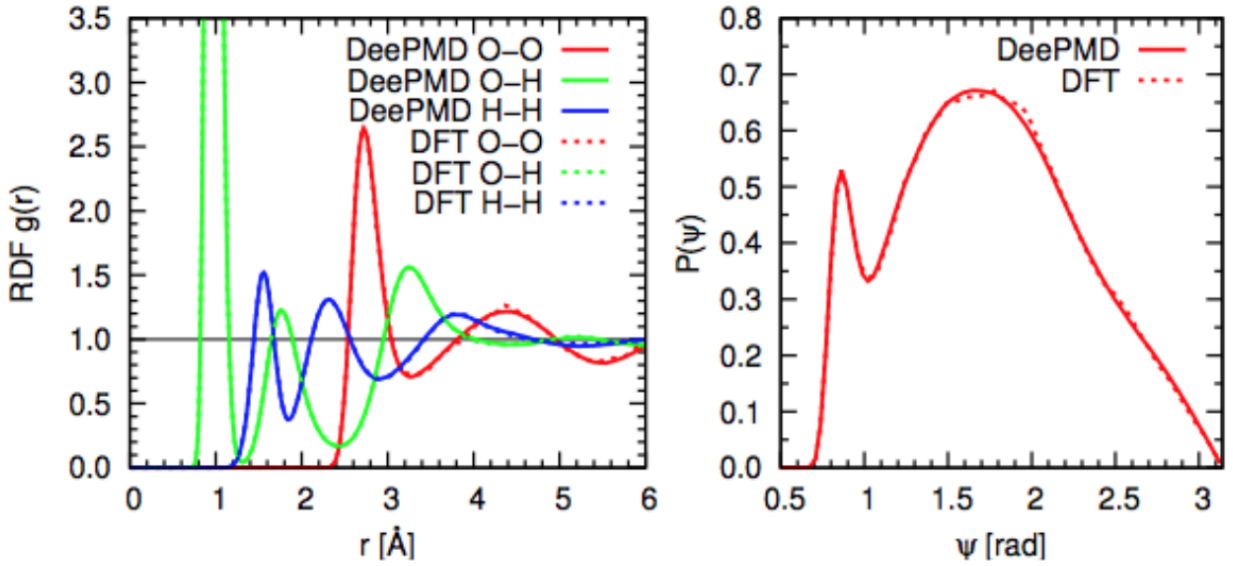
$\sim$0.005% configurations explored by DeePMD are selected for labeling.

Case 1: accuracy is comparable to the accuracy of the data

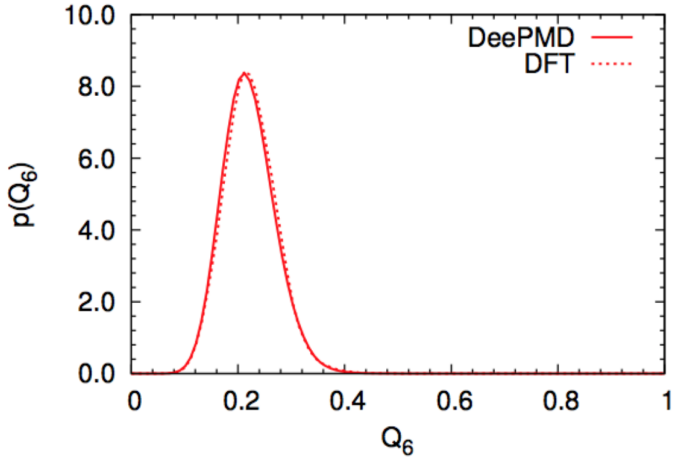


Case 2: structural information of DFT water

Radial and angular distribution function of liquid water (PI-AIMD):

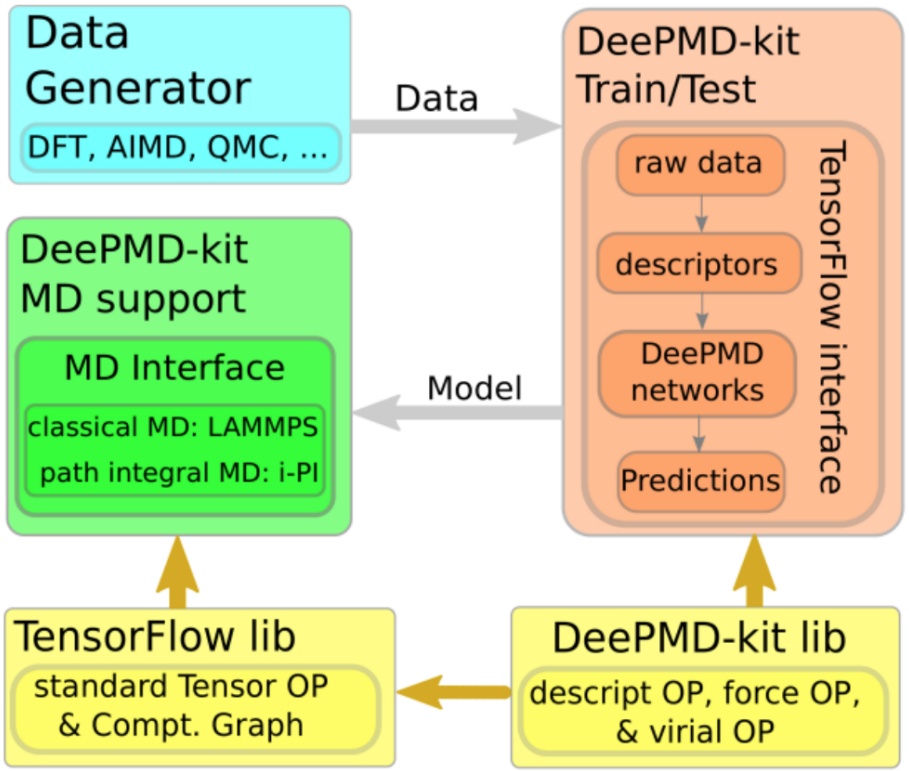


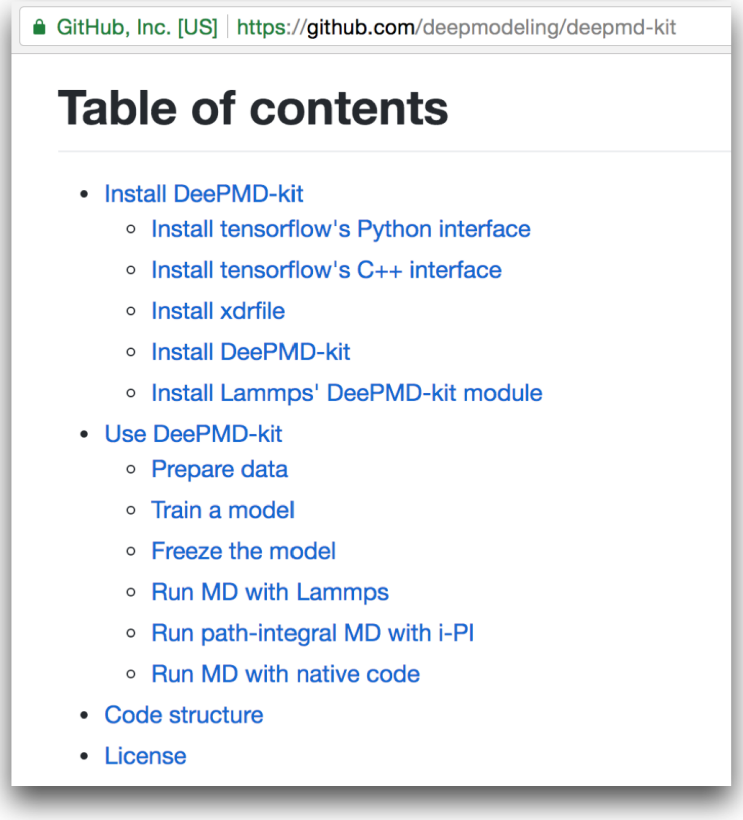
Distribution of the Steinhardt order parameter $\bar{Q}_6$:



DeePMD-kit

Towards realization of a general platform for ML-based PES modeling.





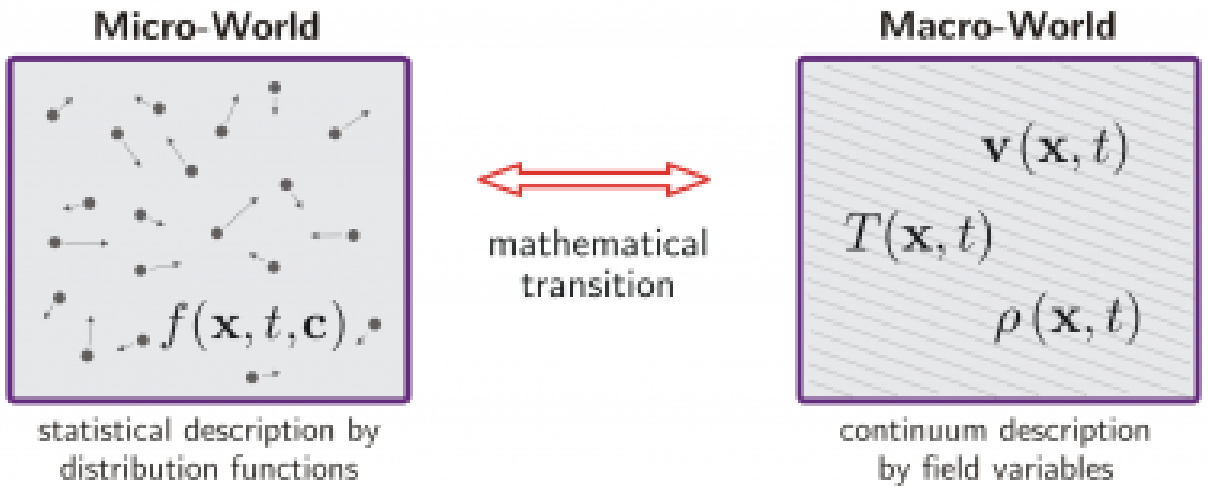
- interfacing state-of-the-art deep learning and MD packages: TensorFlow, LAMMPS, i-PI;
- parallelization: MPI/GPU support.

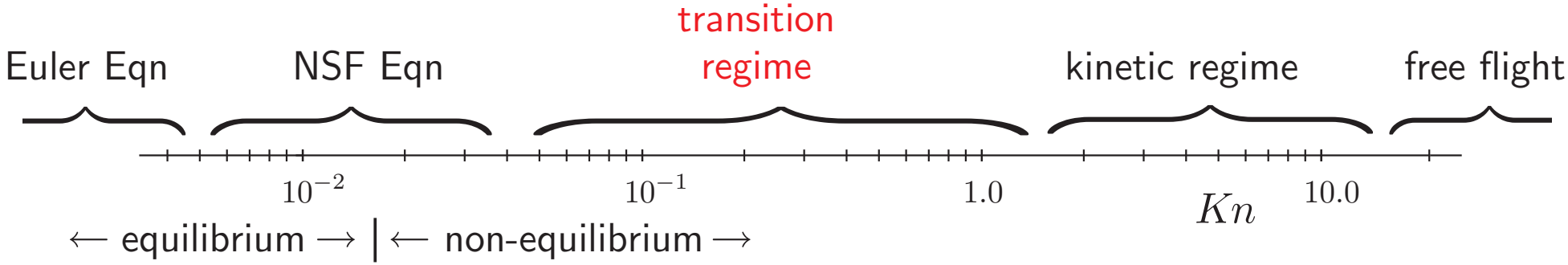
Comp. Phys. Comm., 2018: 0010-4655 (https://github.com/deepmodeling/deepmd-kit))

1. physical/chemical problems
   - understanding water (phase diagram of water, including reactive regions; phase transition: ice to water, ionic liquid to super-ionic ice; nuclear quantum effect: collective tunneling, isotope effect; reactive event: dissociation and recombination; water surface and water/TiO2 interface; spectra: infra-red; Raman; X-ray Absorption; exotic properties: dielectric constant; density anomaly, etc.
   - physical understanding of different systems that require long-time large-scale simulation with high degrees of model fidelity (high-pressure iron: fractional defect; phase boundary; high-pressure hydrogen: exotic phases)
   - catalysis (Pt cluster on MoS2 surface; CO molecules on gold surface, etc.)
2. materials science problems
   - battery materials (diffusion of lithium in LGePS, LSGeSiPS, etc.; diffusion of Se in Cu2Se alloy)
   - high entropy/high temperature alloy (CoCrFeMnNi alloy; Ni-based alloy)
3. organic chemistry/bio problems
   - crystal structure prediction of molecular crystals;
   - protein-ligand interaction;
   - protein folding.

Example 2: Modeling gas dynamics

$$Kn = \frac{\text{mean free path}}{\text{macroscopic length}}$$





Boltzmann Equation

One-particle density function $f(\boldsymbol{x}, \boldsymbol{v}, t)$

$$\partial_t f + \boldsymbol{v} \cdot \nabla_{\boldsymbol{x}} f = \frac{1}{\varepsilon} Q(f), \quad \boldsymbol{v} \in \mathbb{R}^3, \quad \boldsymbol{x} \in \Omega \subset \mathbb{R}^3,$$

$\varepsilon =$ Knudsen number and Q is the collision operator.

Macroscopic state variables: ρ, $\boldsymbol{u}$ and T (density, bulk velocity and temperature)

$$\rho = \int f \, \mathrm{d}\boldsymbol{v}, \quad \boldsymbol{u} = \frac{1}{\rho} \int f \boldsymbol{v} \, \mathrm{d}\boldsymbol{v}, \quad T = \frac{1}{3\rho} \int f |\boldsymbol{v} - \boldsymbol{u}|^2 \, \mathrm{d}\boldsymbol{v}.$$

When $\varepsilon \ll 1$, Boltzmann can be approximated by Euler:

$$\partial_t \boldsymbol{U} + \nabla_{\boldsymbol{x}} \cdot \boldsymbol{F}(\boldsymbol{U}) = 0,$$

with $p = \rho T$, $E = \frac{1}{2}\rho \boldsymbol{u}^2 + \frac{3}{2}\rho T$,

$$\boldsymbol{U} = (\rho, \rho \boldsymbol{u}, E)^T$$

$$\boldsymbol{F}(\boldsymbol{U}) = (\rho \boldsymbol{u}, \rho \boldsymbol{u} \otimes \boldsymbol{u} + pI, (E + p)\boldsymbol{u})^T$$

Machine learning-based moment method

Objective: construct an uniformly accurate (generalized) moment model using machine learning.

Step 1: Learn the Moments through Autoencoder
Find an encoder Ψ that maps $f(\cdot, \boldsymbol{v})$ to generalized moments $\boldsymbol{W} \in \mathbb{R}^M$ and a decoder Φ that recovers the original f from $\boldsymbol{U}, \boldsymbol{W}$

$$\boldsymbol{W} = \Psi(f) = \int \mathbf{w} f \, \mathrm{d}\boldsymbol{v}, \quad \Phi(\boldsymbol{U}, \boldsymbol{W})(\boldsymbol{v}) = h(\boldsymbol{v}; \boldsymbol{U}, \boldsymbol{W}).$$

The goal is essentially to find optimal $\mathbf{w}$ and h parametrized by neural networks through minimizing

$$\mathop{\mathbb{E}}_{f \sim \mathcal{D}} \|f - \Phi(\Psi(f))\|^2 + \lambda_\eta (\eta(f) - h_\eta(\boldsymbol{U}, \boldsymbol{W}))^2.$$

$\eta(f)$ denotes entropy.

Step 2: Learn the Fluxes and Source Terms in the PDE

Recall the general conservative form of the moment system

$$\begin{cases} \partial_t \boldsymbol{U} + \nabla_{\boldsymbol{x}} \cdot \boldsymbol{F}(\boldsymbol{U}, \boldsymbol{W}; \varepsilon) = 0, \\ \partial_t \boldsymbol{W} + \nabla_{\boldsymbol{x}} \cdot \boldsymbol{G}(\boldsymbol{U}, \boldsymbol{W}; \varepsilon) = \mathbb{R}(\boldsymbol{U}, \boldsymbol{W}; \varepsilon). \end{cases}$$

Rewrite it into (variance reduction)

$$\begin{cases} \partial_t \boldsymbol{U} + \nabla_{\boldsymbol{x}} \cdot [\boldsymbol{F}_0(\boldsymbol{U}) + \tilde{\boldsymbol{F}}(\boldsymbol{U}, \boldsymbol{W}; \varepsilon)] = 0, \\ \partial_t \boldsymbol{W} + \nabla_{\boldsymbol{x}} \cdot [\boldsymbol{G}_0(\boldsymbol{U}) + \tilde{\boldsymbol{G}}(\boldsymbol{U}, \boldsymbol{W}; \varepsilon)] = \mathbb{R}(\boldsymbol{U}, \boldsymbol{W}; \varepsilon). \end{cases}$$

$\boldsymbol{F}_0(\boldsymbol{U}), \boldsymbol{G}_0(\boldsymbol{U})$ are the fluxes of the moments $\boldsymbol{U}, \boldsymbol{W}$ under the Maxwellian distribution.

Our goal is to obtain ML models for $\tilde{\boldsymbol{F}}, \tilde{\boldsymbol{G}}, \mathbb{R}$ from the original kinetic equation.

Issues: (1) physical symmetries (e.g. Galilean invariance); (2) data generation (active learning algorithm); (3) locality vs. non-locality of the model

Numerical results for transitional flows

ε varies from 10^{-3} to 10 in the domain. $\boldsymbol{W} \in \mathbb{R}^6$.

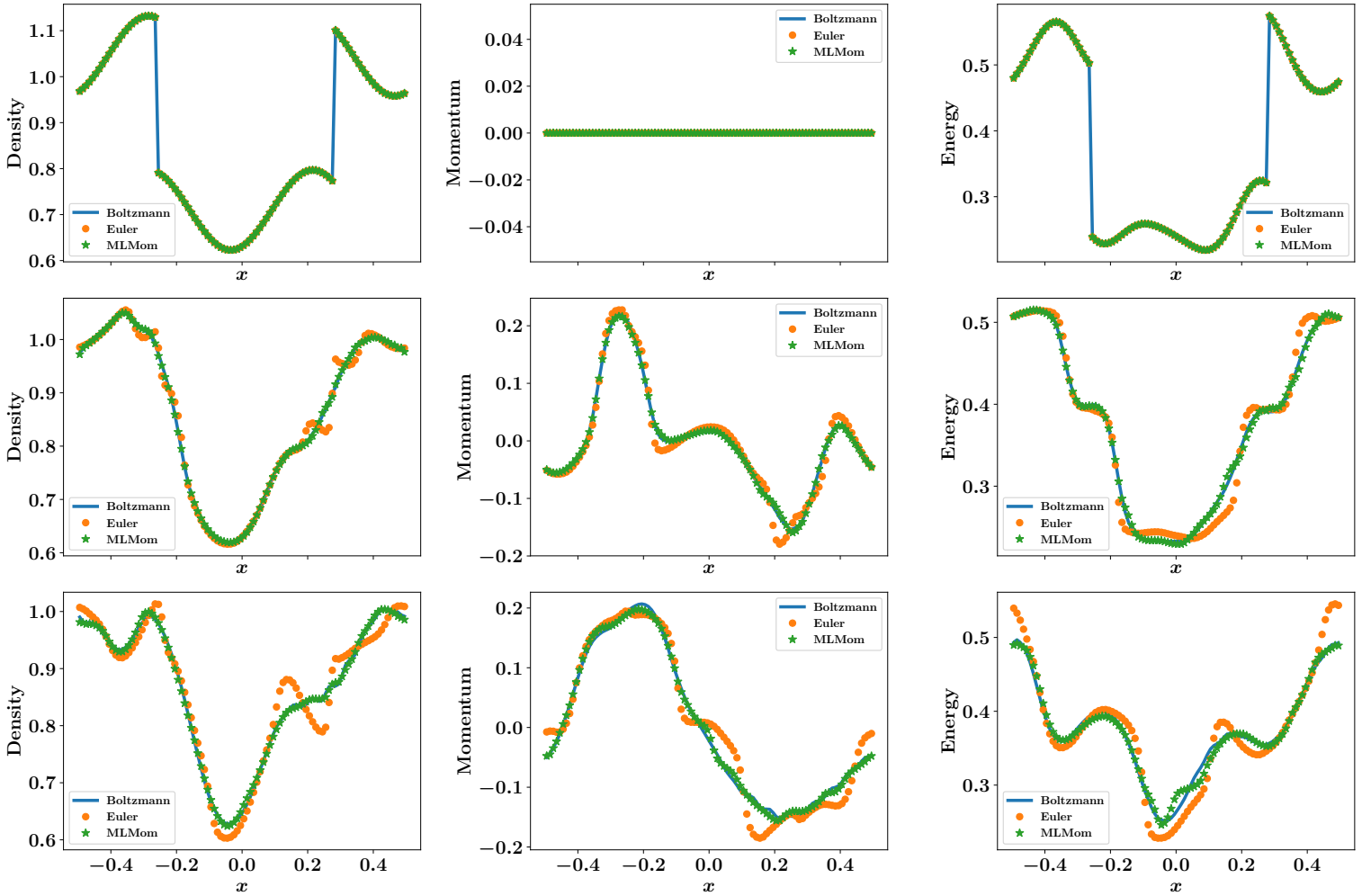
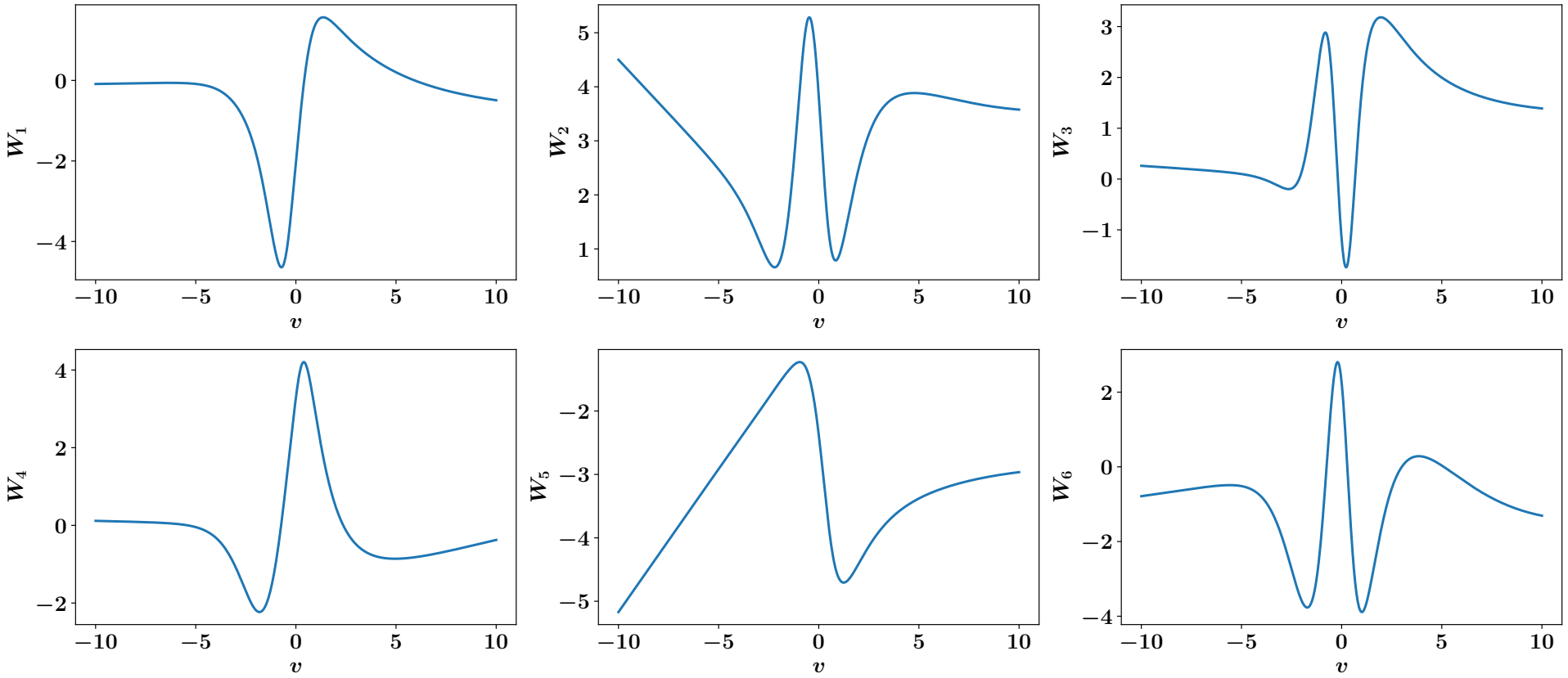


Figure: Profiles of $\rho, \rho\boldsymbol{u}, E$ (from left to right) at $t = 0, 0.05, 0.1$ (from top to bottom)

Numerical results

Learned functions $\mathbf{w}(v)$ as generalized moments



Possible applications

- transitional flow in gas dynamics (e.g. reentry of space-craft)
- plasma dynamics (e.g. tokomak)
- complex fluids (colloids, polymer fluids, etc)

Most importantly, these kinds of ideas provide a very promising way of doing multiscale modeling in the absence of scale separation.

Example 3: Economics and finance

Nonlinear parabolic PDEs

$$\frac{\partial u}{\partial t} + \frac{1}{2}\Delta u + \nabla u \cdot \mu + f\big(u, \nabla u\big) = 0.$$

- Terminal condition: $u(T, x) = g(x)$.
- To fix ideas, we are interested in the solution at $t = 0$, $x = \xi$ for some vector $\xi \in \mathbb{R}^d$.

Example: Black-Scholes Equation with Default Risk:

$$f = -(1 - \delta)Q(u)u - Ru$$

Connection between PDE and BSDE

Backward stochastic differential equations (Pardoux and Peng 1992): Find an adapted process $\{(X_t, Y_t, Z_t)\}_{t\in[0,T]}$ such that

$$X_t = \xi + \int_0^t \mu(s, X_s)\, ds + dW_t$$

$$Y_t = g(X_T) + \int_t^T f(Y_s, Z_s)\, ds - \int_t^T (Z_s)^{\mathrm{T}}\, dW_s$$

Connection to the PDEs (nonlinear Feynman-Kac formula):

$$Y_t = u(t, X_t), \quad Z_t = \nabla u(t, X_t)$$

In other words, given the stochastic process satisfying

$$X_t = \xi + \int_0^t \mu(s, X_s)\, ds + W_t,$$

the solution of PDE satisfies the following SDE

$$u(t, X_t) - u(0, X_0) = -\int_0^t f(u(s, X_s), \nabla u(s, X_s))ds + \int_0^t \nabla u(s, X_s)dW_s.$$

Neural Network Approximation

- Key step: approximate the function $x \mapsto \nabla u(t, x)$ at each discretized time step $t = t_n$ by a feedforward neural network (a subnetwork)

$$\nabla u(t_n, X_{t_n}) \approx \nabla u(t_n, X_{t_n}|\theta_n)$$

  where θ_n denotes neural network parameters.

- Observation: after time discretization, we can stack all the subnetworks together to form a deep neural network (DNN) as a whole:

$$X_{t_{n+1}} - X_{t_n} \approx \mu(t_n, X_{t_n})\,\Delta t_n + \Delta W_n$$

$$u(t_{n+1}, X_{t_{n+1}}) - u(t_n, X_{t_n}) \approx -f(u(t_n, X_{t_n}), \nabla u(t_n, X_{t_n}))\Delta t_n + \nabla u(t_n, X_{t_n})\Delta W_n.$$

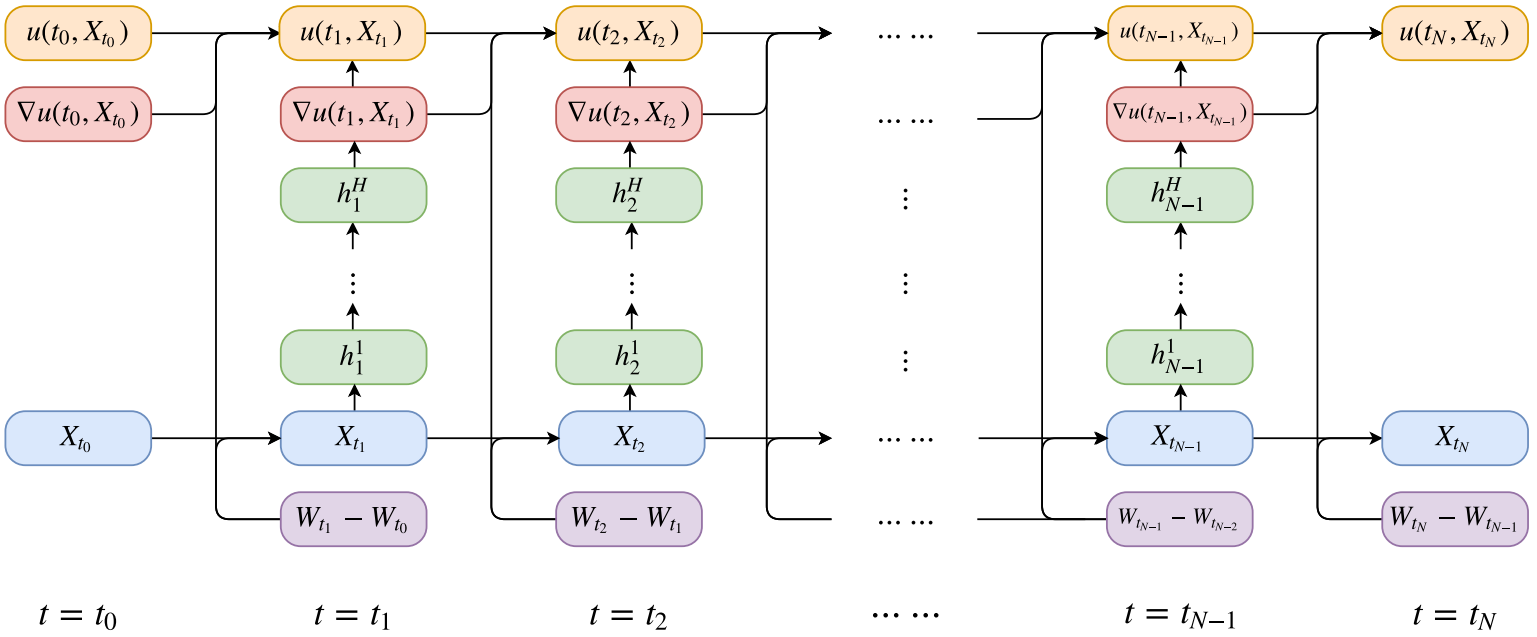


Figure: Each column corresponds to a subnetwork at time $t = t_n$

$$L(\theta) = \mathbb{E}\Big[\big|g(X_{t_N}) - \hat{u}\big(\{X_{t_n}\}_{0\le n\le N}, \{W_{t_n}\}_{0\le n\le N}\big)\big|^2\Big].$$

Open-source code on https://github.com/frankhan91/DeepBSDE

LQG (linear quadratic Gaussian) Example for d=100

$$dX_t = 2\sqrt{\lambda}\, m_t\, dt + \sqrt{2}\, dW_t,$$

Cost functional: $J(\{m_t\}_{0\le t\le T}) = \mathbb{E}\big[\int_0^T \|m_t\|_2^2\, dt + g(X_T)\big]$.

HJB equation:

$$\frac{\partial u}{\partial t} + \Delta u - \lambda \|\nabla u\|_2^2 = 0$$

$$u(t,x) = -\frac{1}{\lambda}\ln\Big(\mathbb{E}\Big[\exp\Big(-\lambda g(x+\sqrt{2}W_{T-t})\Big)\Big]\Big).$$

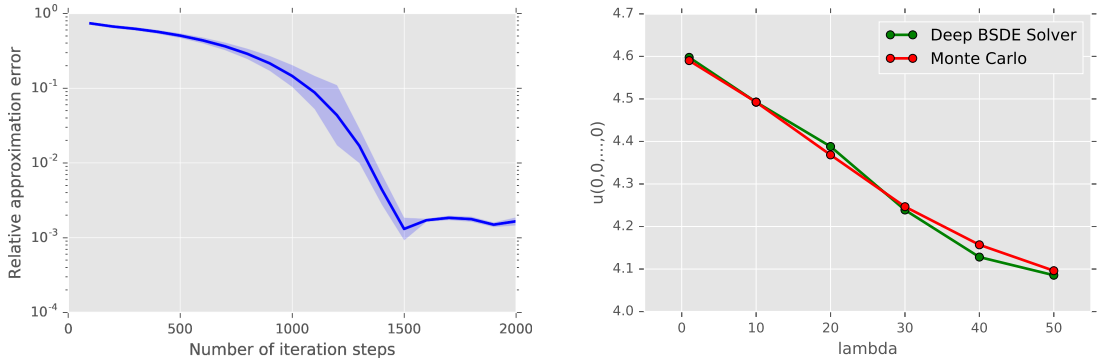


Figure: Left: Relative error of the deep BSDE method for $u(t{=}0, x{=}(0))$ when $\lambda = 1$, which achieves 0.17% in a runtime of 330 seconds. Right: $u(t{=}0, x{=}(0))$ for different λ.

Black-Scholes Equation with Default Risk for d=100

"exact" solution at $t = 0$ $x = (100, \ldots, 100)$ computed by the multilevel Picard method.

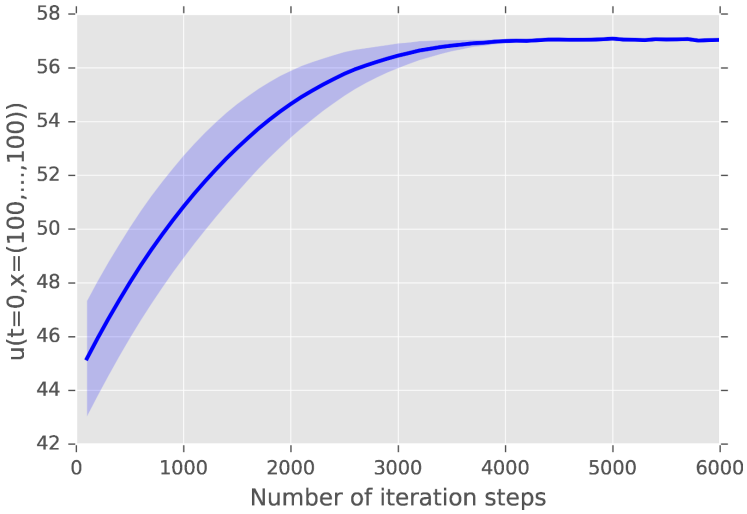


Figure: Approximation of $u(t{=}0, x{=}(100, \ldots, 100))$ against number of iteration steps. The deep BSDE method achieves a relative error of size 0.46% in a runtime of 617 seconds.

Has been applied to the pricing of basket options and path-dependent options.

Example 4: Mathematical principles of natural languages

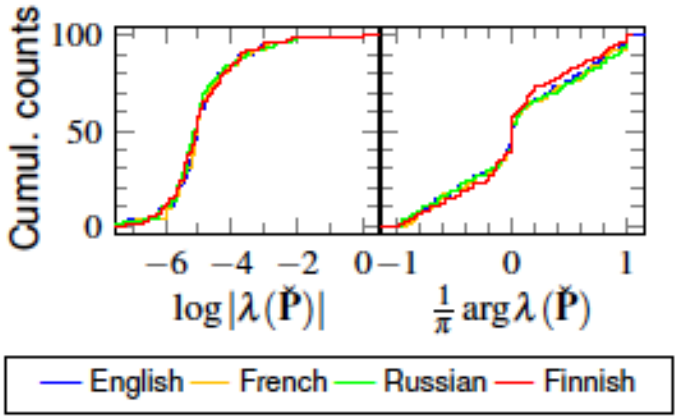
Diversity and universality of human languages at different scales:

- words
- sentences
- inter-sentences

What is semantics? Semantics are invariants under translation

Universal fingerprints for semantics

Markov eigenvalues and their invariance



Alice monolingual in language A

Bob monolingual in language B

Why are Markov spectra (nearly) invariant under translation?

- Thought experiment: measurement of $\mathbb{P}(\mathrm{W}_i^{\mathrm{A}} \to \mathrm{W}_j^{\mathrm{B}})$

Alice first thinks in A, then consults A→B.

Bob first consults A→B, then thinks in B.

$$\mathbf{P}_{\mathrm{A}}\mathbf{T}_{\mathrm{A}\to\mathrm{B}} = \mathbf{T}_{\mathrm{A}\to\mathrm{B}}\mathbf{P}_{\mathrm{B}}$$

- By *universal human nature*
- Ideal dictionary $\Longrightarrow \det(\mathbf{T}_{\mathrm{A}\to\mathrm{B}}) \neq 0 \Longrightarrow \sigma(\mathbf{P}_{\mathrm{A}}) = \sigma(\mathbf{P}_{\mathrm{B}})$

Summary

Common features:

- multi-scale
- classical multi-scale methods have trouble to deal with them
- **new machine learning models seem to be of big help in overcoming the curse of dimensionality.**

Outline

1. PDEs and fundamental laws of physics
2. Period 1: Solving differential equations numerically
3. Period 2: Multiscale, multi-physics modeling
4. Period 3: Integrating machine learning with physical modeling
   - Molecular modeling
   - Kinetic model for gas dynamics
   - Economics and Finance
   - Linguistics
5. Mathematical theory of machine learning
   - Example 1: Random feature model
   - Example 2: Two-layer neural networks
   - Example 3. Deep residual networks

Supervised learning: Approximating functions using samples

- Object of interest: (f^*, μ), where $f^* : \mathbb{R}^d \to \mathbb{R}^1$, μ is a prob measure on $\mathbb{R}^d$.
- Given a set of samples from μ, $\{\boldsymbol{x}_j\}_{j=1}^n$, and $\{y_j = f^*(\boldsymbol{x}_j)\}_{j=1}^n$
- Task: Approximate f^* using $S = \{(\boldsymbol{x}_j, y_j)\}_{j=1}^n$.

- Strategy: Construct some "hypothesis space" (space of functions) $\mathcal{H}_m$ ($m \sim$ the dimension of $\mathcal{H}_m$).
  - linear regression: $f(\boldsymbol{x}) = \beta \cdot \boldsymbol{x} + \beta_0$
  - generalized linear models: $f(\boldsymbol{x}) = \sum_{k=1}^m c_k \phi_k(\boldsymbol{x})$, where $\{\phi_k\}$ are linearly independent functions.
  - two-layer neural networks: $f(\boldsymbol{x}) = \sum_k a_k \sigma(\boldsymbol{b}_k \cdot \boldsymbol{x} + c_k)$, where σ is some nonlinear function, e.g. $\sigma(z) = \max(z, 0)$.
  - deep neural networks (DNN) : compositions of functions of the form above.
- Minimize the "empirical risk":

$$\hat{\mathcal{R}}_n(\theta) = \frac{1}{n} \sum_j (f(\boldsymbol{x}_j) - y_j)^2 = \frac{1}{n} \sum_j (f(\boldsymbol{x}_j) - f^*(\boldsymbol{x}_j))^2$$

Classical numerical analysis (approximation theory)

- Define a "well-posed" math model (the hypothesis space, the loss function, etc)
  - splines: hypothesis space $= C^1$ piecewise cubic polynomials the data

$$I_n(f) = \frac{1}{n}\sum_{j=1}^{n}(f(x_j) - y_j)^2 + \lambda \int |D^2 f(x)|^2 dx$$

  - finite elements: hypothesis space $= C^0$ piecewise polynomials
- Identify the right function spaces, e.g. Sobolev/Besov spaces
  - direct and inverse approximation theorem (Bernstein and Jackson type theorems): f can be approximated by trig polynomials in L^2 to order s iff $f \in H^s$, $\|f\|_{H^s}^2 = \sum_{k=0}^{s} \|\nabla^k f\|_{L^2}^2$.
  - functions of interest are in the right spaces (PDE theory, real analysis, etc).
- Optimal error estimates
  - A priori estimates (for piecewise linear finite elements, $\alpha = 1/d, s = 2$)

$$\|f_m - f^*\|_{H^1} \leq C m^{-\alpha} \|f^*\|_{H^s}$$

  - A posteriori estimates (say in finite elements):

$$\|f_m - f^*\|_{H^1} \leq C m^{-\alpha} \|f_m\|_h$$

Difference between ML and classical approximation theory

- high dimensionality
- finite amount of data

Empirical risk vs. population risk:

$$\hat{\mathcal{R}}_n(\theta) = \frac{1}{n}\sum_j (f(\boldsymbol{x}_j) - y_j)^2 = \frac{1}{n}\sum_j (f(\boldsymbol{x}_j) - f^*(\boldsymbol{x}_j))^2$$

$$\mathcal{R}(\theta) = \mathbb{E}(f(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2 = \int_{\mathbb{R}^d} (f(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2 d\mu$$

Another benchmark: High dimensional integration

Monte Carlo: $X = [0,1]^d, \{\boldsymbol{x}_j, j = 1, \cdots, n\}$ is uniformly distributed in X.

$$I(g) = \int_X g(\boldsymbol{x}) d\mu, \quad I_n(g) = \frac{1}{n} \sum_j g(\boldsymbol{x}_j)$$

$$\mathbb{E}(I(g) - I_n(g))^2 = \frac{1}{n} \mathrm{Var}(g)$$

$\mathrm{Var}(g) = \int_X g^2(\boldsymbol{x}) d\boldsymbol{x} - (\int_X g(\boldsymbol{x}) d\boldsymbol{x})^2$

The $O(1/\sqrt{n})$ rate is the best we can hope for.

However, $\mathrm{Var}(g)$ can be very large in high dimension. That's why variance reduction is important!

Estimating the generalization gap

"Generalization gap" $= \hat{\mathcal{R}}(\hat{\theta}) - \hat{\mathcal{R}}_n(\hat{\theta}) = I(g) - I_n(g), \quad g(\boldsymbol{x}) = (f(\boldsymbol{x}, \hat{\theta}) - f^*(\boldsymbol{x}))^2$

$$I(g) = \int_{X=[-1,1]^d} g(\boldsymbol{x}) d\mu, \quad I_n(g) = \frac{1}{n} \sum_j g(\boldsymbol{x}_j)$$

- For fixed $g = h$, we have
$$|I(h) - I_n(h)| \sim \frac{1}{\sqrt{n}}$$
- For Lipschitz functions (Wasserstein distance)
$$\sup_{\|h\|_{Lip} \leq 1} |I(h) - I_n(h)| \sim \frac{1}{n^{1/d}}$$
- For functions in Barron space, to be defined later
$$\sup_{\|h\|_{\mathcal{B}} \leq 1} |I(h) - I_n(h)| \sim \frac{1}{\sqrt{n}}$$

Rademacher complexity

Let $\mathcal{H}$ be a set of functions, and $S = (\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n)$ be a set of data points. Then, the Rademacher complexity of $\mathcal{H}$ with respect to S is defined as

$$\hat{R}_S(\mathcal{H}) = \frac{1}{n}\mathbb{E}_\xi \left[\sup_{h\in\mathcal{H}} \sum_{i=1}^{n} \xi_i h(\boldsymbol{x}_i) \right],$$

where $\{\xi_i\}_{i=1}^n$ are i.i.d. random variables taking values ± 1 with equal probability.

Theorem (Rademacher complexity and the generalization gap)

Given a function class $\mathcal{H}$, for any $\delta \in (0,1)$, with probability at least $1-\delta$ over the random samples $S = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n)$,

$$\sup_{h\in\mathcal{H}} \left| \mathbb{E}_{\boldsymbol{x}} [h(\boldsymbol{x})] - \frac{1}{n}\sum_{i=1}^{n} h(\boldsymbol{x}_i) \right| \le 2\hat{R}_S(\mathcal{H}) + \sup_{h\in\mathcal{H}} \|h\|_\infty \sqrt{\frac{\log(2/\delta)}{2n}}.$$

$$\sup_{h\in\mathcal{H}} \left| \mathbb{E}_{\boldsymbol{x}} [h(\boldsymbol{x})] - \frac{1}{n}\sum_{i=1}^{n} h(\boldsymbol{x}_i) \right| \ge \frac{1}{2}\hat{R}_S(\mathcal{H}) - \sup_{h\in\mathcal{H}} \|h\|_\infty \sqrt{\frac{\log(2/\delta)}{2n}}.$$

- If $\mathcal{H}$ = unit ball in Barron space: $\hat{R}_S(\mathcal{H}) \sim O(1/\sqrt{n})$
- If $\mathcal{H}$ = unit ball in Lipschitz space: $\hat{R}_S(\mathcal{H}) \sim O(1/n^{1/d})$
- If $\mathcal{H}$ = unit ball in C^0: $\hat{R}_S(\mathcal{H}) \sim O(1)$

Two types of machine learning models

(1). Models that suffer from the curse of dimensionality:

$$\text{generalization error} = O(m^{-\alpha/d} + n^{-\beta/d})$$

- piecewise polynomial approximation
- wavelets with fixed wavelet basis

(2). Models that don't suffer from the curse of dimensionality:

$$\text{generalization error} = O(\gamma_1(f^*)/m + \gamma_2(f^*)/\sqrt{n})$$

- random feature models: $\{\phi(\cdot, \omega), \omega \in \Omega\}$ is the set of "features". Given any realization $\{\omega_j\}_{j=1}^m$, i.i.d. with distribution π, $\mathcal{H}_m(\{\omega_j\}) = \{f_m(\boldsymbol{x}, \boldsymbol{a}) = \frac{1}{m}\sum_{j=1}^m a_j \phi(\boldsymbol{x}; \omega_j).\}$.
- two layer neural networks $\mathcal{H}_m = \{\frac{1}{m}\sum_{j=1}^m a_j \sigma(\boldsymbol{b}_j^T \boldsymbol{x} + c_j)\}$
- residual neural networks $\mathcal{H}_L = \{f(\cdot, \theta) = \alpha \cdot \boldsymbol{z}_{L,L}(\cdot)\}$

$$\boldsymbol{z}_{l+1,L}(\boldsymbol{x}) = z_{l,L}(\boldsymbol{x}) + \frac{1}{L}\boldsymbol{U}_l \sigma \circ (\boldsymbol{W}_l z_{l,L}(\boldsymbol{x})), \quad \boldsymbol{z}_{0,L}(\boldsymbol{x}) = \boldsymbol{V}\boldsymbol{x}$$

Example 1: Random feature model

$\{\phi(\cdot;\omega)\}$: collection of random features. π: prob distribution of the random variable ω.

Hypothesis space: Given any realization $\{\omega_j\}_{j=1}^m$, i.i.d. with distribution π

$$\mathcal{H}_m(\{\omega_j\}) = \{f_m(\boldsymbol{x}, \boldsymbol{a}) = \frac{1}{m}\sum_{j=1}^{m} a_j \phi(\boldsymbol{x};\omega_j).\}.$$

Looking for the right function space: Consider functions of the form

$$\mathcal{H}_k = \{f : f(\boldsymbol{x}) = \int a(\omega)\phi(\boldsymbol{x};\omega)d\pi(\omega)\}, \quad \|f\|_{\mathcal{H}_k}^2 = \mathbb{E}_{\omega\sim\pi}[|a(\omega)|^2]$$

This is related to the reproducing kernel Hilbert space (RKHS) with kernel:

$$k(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}_{\omega\sim\pi}[\phi(\boldsymbol{x};\omega)\phi(\boldsymbol{x}';\omega)]$$

A priori estimates of the regularized model

$$L_n(\theta) = \hat{\mathcal{R}}_n(\theta) + \lambda\sqrt{\frac{\log(2d)}{n}}\|\theta\|_{\mathcal{H}}, \qquad \hat{\theta}_n = \text{argmin } L_n(\theta)$$

where

$$\|\theta\|_{\mathcal{H}} = \left(\frac{1}{m}\sum_{j=1}^{m}|a_j|^2\right)^{1/2}$$

Theorem

Assume that the target function $f^* : [0,1]^d \mapsto [0,1] \in \mathcal{H}_k$. There exist constants C_0, C_1, C_2, such that for any $\delta > 0$, if $\lambda \geq C_0$, then with probability at least $1-\delta$ over the choice of training set, we have

$$\mathcal{R}(\hat{\theta}_n) \leq C_1\left(\frac{\|f^*\|_{\mathcal{H}_k}^2}{m} + \|f^*\|_{\mathcal{H}_k}\sqrt{\frac{\log(2d)}{n}}\right) + C_2\sqrt{\frac{\log(4C_2/\delta)+\log(n)}{n}}.$$

Example 2: Two-layer neural networks

$$\mathcal{H}_m = \{\frac{1}{m}\sum_{j=1}^{m} a_j \sigma(\boldsymbol{b}_j^T \boldsymbol{x} + c_j)\}$$

Consider functions $f : X = [-1, 1]^d \mapsto \mathbb{R}$ of the following form

$$f(\boldsymbol{x}) = \int_\Omega a\sigma(\boldsymbol{b}^T \boldsymbol{x} + c)\rho(da, d\boldsymbol{b}, dc), \quad \boldsymbol{x} \in X$$

$\Omega = \mathbb{R}^1 \times \mathbb{R}^d \times \mathbb{R}^1$, ρ is a probability distribution on Ω.

$$\|f\|_{\mathcal{B}_p} = \inf_\rho \left(\mathbb{E}_\rho[|a|^p(\|\boldsymbol{b}\|_1 + |c|)^p]\right)^{1/p}$$

$$\mathcal{B}_p = \{f \in \mathcal{S}' : \|f\|_{\mathcal{B}_p} < \infty\}$$

What kind of functions admit such a representation?

Theorem (Barron and Klusowski (2016)): If $\int_{\mathbb{R}^d} \|\omega\|_1^2 |\hat{f}(\omega)| d\omega < \infty$, where $\hat{f}$ is the Fourier transform of f, then f can be represented as

$$\tilde{f}(\boldsymbol{x}) = f(\boldsymbol{x}) - (f(0) + \boldsymbol{x} \cdot \nabla f(0)) = \int_{\Omega} a\sigma(\boldsymbol{b}^T \boldsymbol{x} + c)\rho(da, d\boldsymbol{b}, dc)$$

where $\sigma(x) = \max(0, x)$. Moreover $f \in \mathcal{B}_\infty$. Furthermore, we have

$$\|\tilde{f}\|_{\mathcal{B}_\infty} \leq 2 \int_{\mathbb{R}^d} \|\omega\|_1^2 |\hat{f}(\omega)| d\omega$$

Direct and inverse approximation theorems

Theorem (Direct Approximation Theorem)

There exists an absolute constant C_0 such that

$$\|f - f_m\|_{L^2(X)} \leq \frac{C_0 \|f\|_{\mathcal{B}_2}}{\sqrt{m}}$$

Theorem (Inverse Approximation Theorem)

For $p > 1$, let

$$\mathcal{N}_{p,C} = \{ \frac{1}{m} \sum_{k=1}^{m} a_k \sigma(b_k^T \boldsymbol{x} + c_k) : \left(\frac{1}{m} \sum_{k=1}^{m} |a_k|^p (\|b_k\|_1 + c_k)^p \right)^{1/p} \leq C, m \in \mathbb{N}^+ \}.$$

Let f^* be a continuous function. Assume there exists a constant C and a sequence of functions $f_m \in \mathcal{N}_{p,C}$ such that

$$f_m(\boldsymbol{x}) \to f^*(\boldsymbol{x})$$

for all $\boldsymbol{x} \in X$. Then there exists a probability distribution ρ on Ω, such that

$$f^*(\boldsymbol{x}) = \int a\sigma(\boldsymbol{b}^T \boldsymbol{x} + c)\rho(da, d\boldsymbol{b}, dc),$$

for all $\boldsymbol{x} \in X$.

Complexity estimates

Theorem

Let $\mathcal{F}_Q = \{f \in \mathcal{B}_1, \|f\|_{\mathcal{B}_1} \leq Q\}$. Then we have

$$\hat{\mathcal{R}}_n(\mathcal{F}_Q) \leq 2Q\sqrt{\frac{2\ln(2d)}{n}}$$

A priori estimates for regularized model

where the path norm is defined by:

$$L_n(\theta) = \hat{\mathcal{R}}_n(\theta) + \lambda\sqrt{\frac{\log(2d)}{n}}\|\theta\|_{\mathcal{P}}, \qquad \hat{\theta}_n = \text{argmin } L_n(\theta)$$

$$\|\theta\|_{\mathcal{P}} = \frac{1}{m}\sum_{k=1}^{m}|a_k|(\|\boldsymbol{b}_k\|_1 + |c_k|) \quad (= \|f(\cdot;\theta)\|_{\mathcal{B}_1})$$

Theorem (Weinan E, Chao Ma, Lei Wu, submitted)

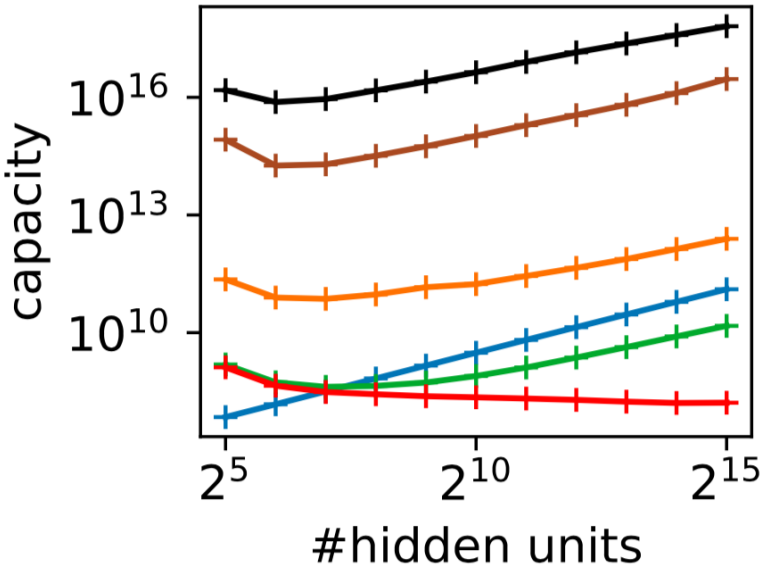
Assume that the target function $f^* : [0,1]^d \mapsto [0,1] \in \mathcal{B}_2$. There exist constants C_0, C_1, C_2, such that for any $\delta > 0$, if $\lambda \geq C_0$, then with probability at least $1-\delta$ over the choice of training set, we have

$$\mathcal{R}(\hat{\theta}_n) \leq C_1\left(\frac{\|f^*\|_{\mathcal{B}_2}^2}{m} + \|f^*\|_{\mathcal{B}_2}\sqrt{\frac{\log(2d)}{n}}\right) + C_2\sqrt{\frac{\log(4C_2/\delta)+\log(n)}{n}}.$$

Traditional results: A posteriori estimates

$$|\mathcal{R}(\theta) - \hat{\mathcal{R}}_n(\theta)| \leq C_1(|||\theta|||+1)\sqrt{\frac{\log(2d)}{n}} + C_2\sqrt{\frac{\log(4C_2(1+|||\theta|||))^2/\delta)}{n}}$$

where $|||\theta|||$ is some norm of θ (see e.g. Behnam Neyshabur, Zhiyuan Li, et al. *Towards Understanding the Role of Over-Parametrization in Generalization of Neural Networks* (2018)).



Deep residual networks

$$\begin{aligned} \boldsymbol{z}_{l+1,L}(\boldsymbol{x}) &= z_{l,L}(\boldsymbol{x}) + \frac{1}{L}\boldsymbol{U}_l \sigma \circ (\boldsymbol{W}_l z_{l,L}(\boldsymbol{x})), \\ \boldsymbol{z}_{0,L}(\boldsymbol{x}) &= \boldsymbol{V}\boldsymbol{x}, \quad f(\boldsymbol{x},\theta) = \alpha \cdot \boldsymbol{z}_{L,L}(\boldsymbol{x}) \end{aligned}$$

- compositional law of large numbers (express target function as a compositional expectation)
- compositional function spaces (Barron space is embedded in compositional function spaces)
- direct and inverse approximation theorem
- optimal scaling for the Rademacher complexity
- optimal a priori estimates for the regularized model

Conclusion

We are at the verge of a new scientific revolution that will impact mathematics and applied mathematics in fundamental ways.

- Integrating machine learning (Keplerian paradigm) with first principle based physical modeling (Newtonian paradigm) opens up a new (and powerful) paradigm for scientific research.
  Applied mathematics is the most natural platform for this integration.

- Theoretical foundation of machine learning = high dimensional numerical analysis

Acknowledgement

Collaborators:

- molecular modeling: Linfeng Zhang, Han Wang, Jiequn Han, Wissam Saidi
- kinetic equations: Jiequn Han, Chao Ma, Zheng Ma
- PDEs and finance: Jiequn Han and Arnulf Jentzen
- linguistics and natural languages: Yajun Zhou
- mathematical theory: Chao Ma, Lei Wu, Qingcan Wang

Supported by: ONR and iFlytek

MSML 2020

A new NIPS or ICML style annual conference, which also serves as a venue for publications: Mathematical and Scientific Machine Learning (MSML)

First meeting:

- Program Chairman: Jianfeng Lu (Duke) and Rachel Ward (Univ Texas/Austin)
- Time: July 15-17, 2020
- Location: Princeton
- Submission deadline: November 30, 2019
- website: http://msml-conf.org