机器学习与物理模型

Machine Learning and Physics-based Modeling:

How can we construct interpretable and truly reliable physical models using *concurrent* machine learning?

鄂维南

Princeton University

Joint work with:

Jiequn Han, Han Wang, Linfeng Zhang

Roberto Car, Chao Ma, Zheng Ma, Huan Lei, ...

Outline

1. PDEs and fundamental laws of physics
2. Machine learning
3. Concurrent learning
4. Molecular modeling
5. Kinetic model for gas dynamics
6. Concluding remarks

Outline

1. PDEs and fundamental laws of physics
2. Machine learning
3. Concurrent learning
4. Molecular modeling
5. Kinetic model for gas dynamics
6. Concluding remarks

Two main themes of scientific research

- 寻求基本原理
  Physics: Newton's laws, Maxwell equations, Quantum mechanics
- 解决实际问题
  "Engineering" (industrial) problems: 制造行业，材料，医疗行业。。。

基本原理：Paul 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. "

- 除少数物理领域（高能物理，核物理，天体物理）以外，对我们日常生活中碰到的问题，如 化学, 材料, 生物, 工程 领域中的问题，量子力学就足够了。
- 困难的是，量子力学数学上太困难了。

$$i\partial_t \Psi = \mathbb{H}\Psi, \quad \Psi = \Psi(\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N)$$

$\mathbb{H}$= 系统的哈密顿算符

怎样从基本原理出发解决实际问题?

- 数值方法
- 简化模型
- 多尺度模型／算法

简化模型

对简化模型的要求:

- express fundamental physical principles (e.g. conservation laws)
- obey physical constraints (e.g. symmetries, frame-indifference, Galilean invariance)
- (universally) accurate (transferrable): physical parameters can be measured using simple experiments
- physically meaningful (interpretable)

Very successful example: Euler's equation for gas dynamics (for dense gas)

A not very successful example: extended Euler equation for rarified gas (e.g. Grad's 13-moment equation)

$$\mathrm{Kn} = \frac{\text{分子平均自由程}}{\text{系统尺度}}$$

怎样简化模型?

- 物理方法
  - generalized hydrodynamics: Onsager
  - Landau: gradient expansion, weakly nonlinear theory
  - successful example: Ericksen-Leslie equation for liquid crystals
  - unsuccessful example: Non-Newtonian fluids
- 数学方法
  - 渐近分析, e.g. PLK (Poincaré-Lighthill-Kuo) method
  - projection onto principal components
  - truncation (e.g. Lorenz system)

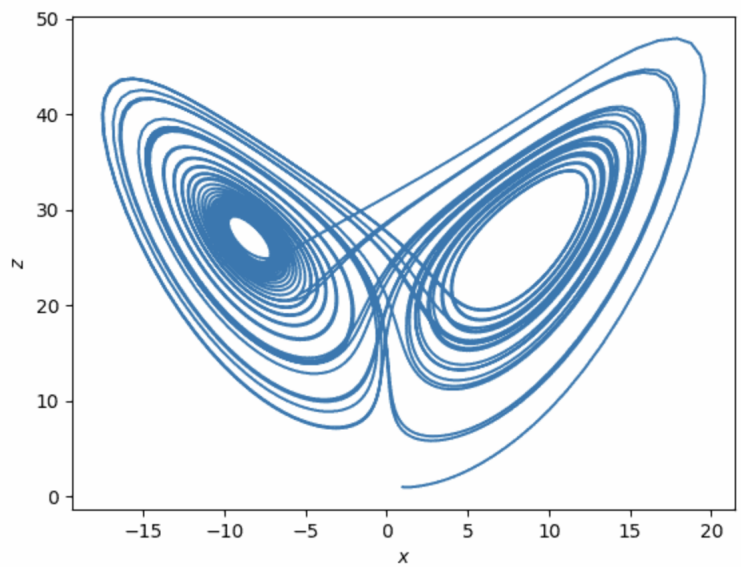
Lorenz system

Drastically simplified model for 2D incompressible flow with buoyancy and gravity.

- $x = \psi_{11} \sim$ convective intensity
- $y = T_{11} \sim$ temperature difference between descending and ascending currents
- $z = T_{02} \sim$ difference in vertical temperature

$$\dot{x} = \sigma(y - x), \quad \dot{y} = -xz + rx - y, \quad \dot{z} = xy - bz$$

where $\sigma = \nu/\kappa, r = Ra/Ra_c$.



Question: Is this a good model for the original problem?

数值方法

- 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
- .....

低维问题基本解决，高维（或多个自由度）问题极度困难

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

多尺度方法/物理力学

借助微观模型解决宏观问题, by performing (many) small scale microscale simulations (see Heterogeneous Multi-scale Method by E and Engquist).

Difficulty:

- Microscale models are not very reliable either
- The scale of microscale simulation is still out of reach
- Lack of clear separation of scales

Many problems remain

Within mechanics:

- equations for solids? nonlinear elasticity, plasticity, visco-plasticity, crack propagation
- non-Newtonian fluids (generalized Newtonian, Maxwell models, co-rotational, co-deformational)
- turbulence: RANS ($k - \varepsilon$ models), LES
- rarefied gas dynamics

Outside of mechanics

- Density functional theory: Exchange-correlation functional
- Molecular dynamics: Potential energy surface (PES)
- Coarse-graind molecular dynamics: Free energy function

In the absence of systematic methods, one has to resort to ad hoc approaches.

Machine learning comes to the rescue

Modern machine learning provides a way to solve the CoD problem.

Two objectives:

- interpretable and truly reliable physical models with machine learning
- multi-scale modeling in situations without scale separation

Remark: 机器学习不是解决困难问题的万能工具

- 怎样用好机器学习是一个nontrivial problem

Outline

1. PDEs and fundamental laws of physics
2. Machine learning
3. Concurrent learning
4. Molecular modeling
5. Kinetic model for gas dynamics
6. Concluding remarks

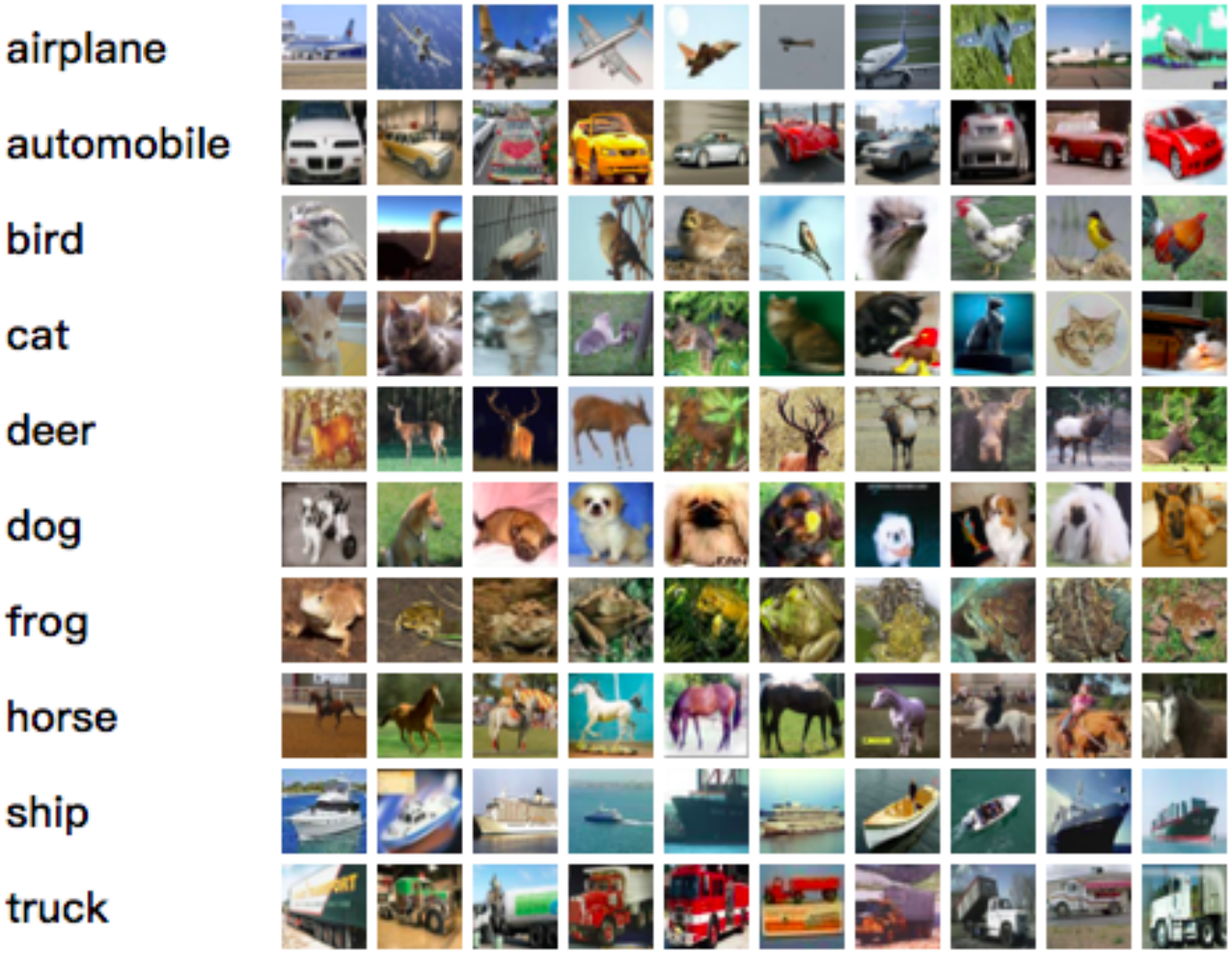
Basic example: Supervised learning

Given $S = \{(\boldsymbol{x}_j, y_j = f^*(\boldsymbol{x}_j)), j \in [n]\}$, learn f^*.

1. This is a problem about function approximation.
2. Based on finite pieces of "labeled" data
   - regression (f^* is continuous) vs classification (f^* is discrete)
   - will neglect measurement noise (not crucial for the talk)
   - assume $\boldsymbol{x}_j \in X = [0,1]^d$. d is typically quite large
   - notation: $\mu =$ the distribution of $\{\boldsymbol{x}_j\}$

In practice, one divides S into two subsets, a training set and a testing set.

Classification example: Cifar 10



- Input: $\boldsymbol{x} \in [0,1]^d$ with $d = 32 \times 32 \times 3 = 3072$.
- Output: $f^* \in \{0, 1, \ldots, 9\}$.

ML and approximation theory

Classical approximation theory:

- polynomials, piece-wise polynomials, wavelets, splines
- curse of dimensionality:

$$\|f^* - f_m\| \sim m^{-\alpha/d}\Gamma(f^*)$$

ML: high dimensionality

Dealing with high dimensionality: Monte Carlo integration

$$I(g) = \int_{[0,1]^d} g(\boldsymbol{x}) d\mu, \quad I_n(g) = \frac{1}{n} \sum_j g(\boldsymbol{x}_j)$$

Trapezoidal rule: $I(g) - I_n(g) \sim m^{-\alpha/d} \Gamma(g)$

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

$$\mathbb{E}(I(g) - I_n(g))^2 = \frac{\mathsf{var}(g)}{n}, \quad \mathsf{var}(g) = \int_X g^2(\boldsymbol{x}) d\boldsymbol{x} - \left(\int_X g(\boldsymbol{x}) d\boldsymbol{x} \right)^2$$

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

However, $\mathsf{var}(g)$ can be very large in high dimension. Variance reduction!

Representing functions: An illustrative example

Traditional approach:

$$f(\boldsymbol{x}) = \int_{\mathbb{R}^d} a(\boldsymbol{\omega}) e^{i(\boldsymbol{\omega},\boldsymbol{x})} d\boldsymbol{\omega}, \quad f_m(\boldsymbol{x}) = \frac{1}{m} \sum_j a(\boldsymbol{\omega}_j) e^{i(\boldsymbol{\omega}_j,\boldsymbol{x})}$$

$\{\boldsymbol{\omega}_j\}$ is a fixed grid, e.g. uniform.

$$\|f - f_m\|_{L^2(X)} \leq C_0 m^{-\alpha/d} \|f\|_{H^\alpha(X)}$$

"New" approach:

$$f(\boldsymbol{x}) = \int_{\mathbb{R}^d} a(\boldsymbol{\omega}) e^{i(\boldsymbol{\omega},\boldsymbol{x})} \pi(d\boldsymbol{\omega}) = \mathbb{E}_{\boldsymbol{\omega}\sim\pi} a(\boldsymbol{\omega}) e^{i(\boldsymbol{\omega},\boldsymbol{x})}$$

where π is a probability measure on $\mathbb{R}^d$. Let $\{\boldsymbol{\omega}_j\}$ be an i.i.d. sample of π.

$$\mathbb{E}|f(\boldsymbol{x}) - \frac{1}{m} \sum_{j=1}^{m} a(\boldsymbol{\omega}_j) e^{i(\boldsymbol{\omega}_j,\boldsymbol{x})}|^2 = \frac{\mathsf{var}(f)}{m}$$

where

$$\mathsf{var}(f) = \mathbb{E}_{\boldsymbol{\omega}\sim\pi} |a(\boldsymbol{\omega})|^2 - f(\boldsymbol{x})^2$$

Two types of machine learning models

(1). Models that suffer from CoD:

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

- piecewise polynomial approximation
- wavelets with fixed wavelet basis

(2). Models that don't suffer from CoD: For example

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

These are "Monte-Carlo-like" bounds, γ_1 and γ_2 play the role of variance in Monte Carlo.

- random feature models
- neural network models

Outline

1. PDEs and fundamental laws of physics
2. Machine learning
3. Concurrent learning
4. Molecular modeling
5. Kinetic model for gas dynamics
6. Concluding remarks

Sequential vs concurrent learning

Where are the data sets? (It is very expensive to get the data)

- **sequential learning**: first collect labeled data $\{\boldsymbol{x}_j, y_j\}$, then perform learning
- **concurrent learning**: generate the data set on the fly as learning proceeds

compare with "**active learning**": having unlabeled data $\{\boldsymbol{x}_j\}$, and decide which ones to label and use them to perform learning

concurrent learning: generate "**optimal data set**" (both unlabeled and labeled, representative enough yet as small as possible)

the latter is a more interactive process

The exploration-labeling-training (ELT) procedure for concurrent learning

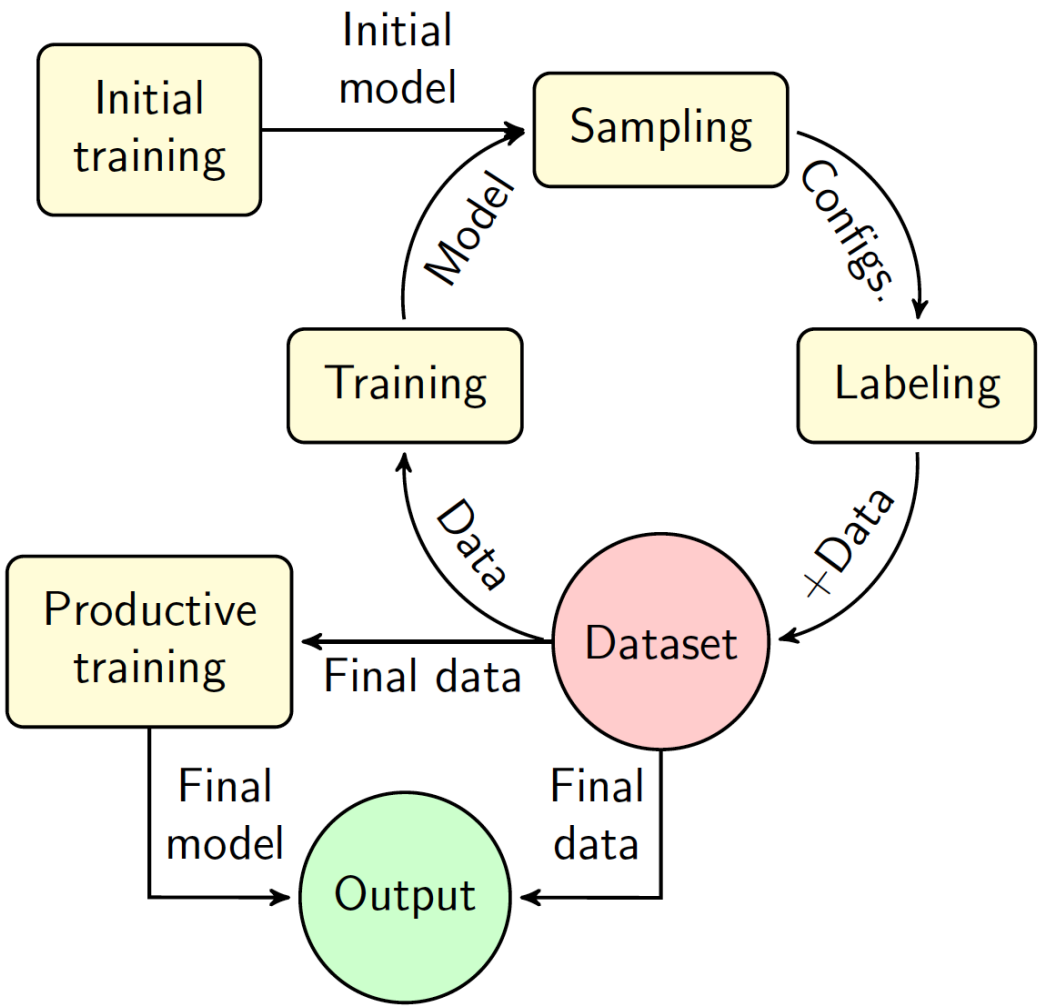
Zhang, Wang and E (2018), J. Chem. Phys.

Start out with no (macro-scale) model, no data; but with a micro-scale model.
Repeat the following steps:

1. **exploration**: explore the configuration space, and decide which configurations need to be labeled.
2. **labeling**: compute the micro-scale solutions for the configurations that need to be labeled. This is our data set.
3. **training**: train the macro-scale model, and use it to help the exploration

Similar to "active learning" but more interactive.....

The ELT algorithm



Indicator: $\epsilon = \max_i \sqrt{\langle \| \boldsymbol{f}_i - \bar{\boldsymbol{f}}_i \|^2 \rangle}, \ \bar{\boldsymbol{f}}_i = \langle \boldsymbol{f}_i \rangle$

Outline

1. PDEs and fundamental laws of physics
2. Machine learning
3. Concurrent learning
4. Molecular modeling
5. Kinetic model for gas dynamics
6. Concluding remarks

Molecular dynamics

Traditional dilemma: accuracy vs cost.

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

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

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 ($r_{ij} = |\boldsymbol{R}_i - \boldsymbol{R}_j|$):

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

Integrating ML with molecular modeling

New paradigm:

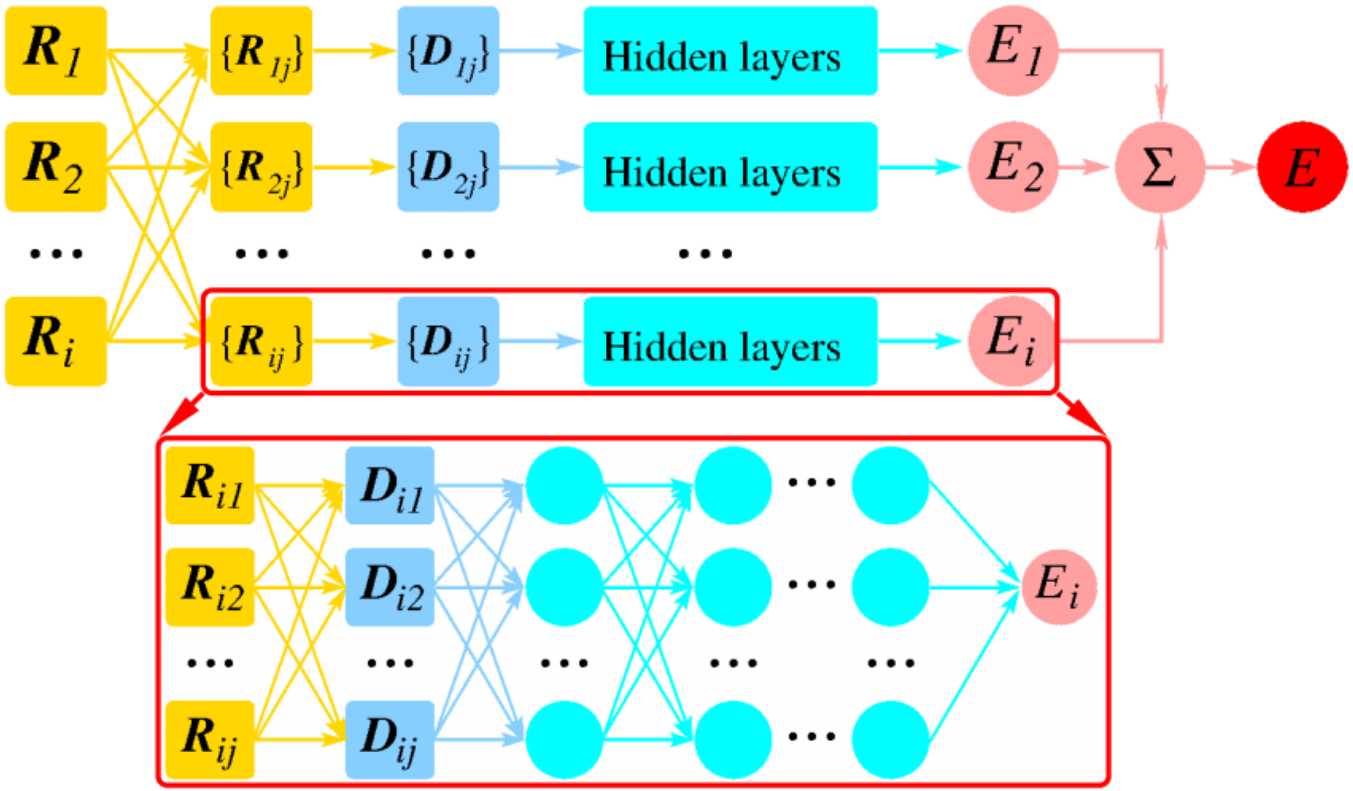
- quantum mechanics model – data generator
- machine learning – parametrize (represent) the model
- molecular dynamics – simulator

Important issues:

- How do we make sure that the data is good enough?
- How do we enforce physical constraints (symmetries)?

Deep Potential: construction

Structure: composite neural networks (NNs). $E = \sum_i E^i$.

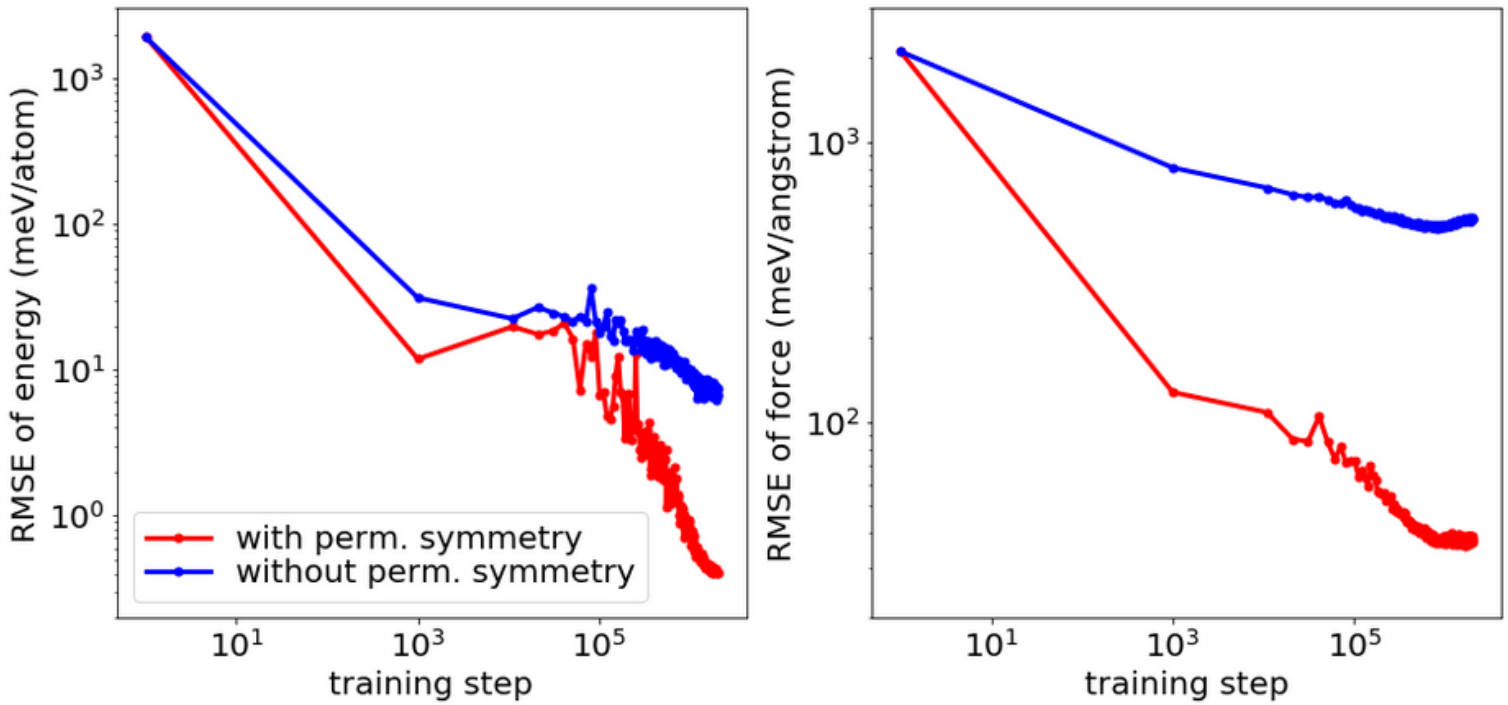


$$\mathcal{R}^i = \{\boldsymbol{R}_{1i}^T, \cdots, \boldsymbol{R}_{ji}^T, \cdots, \boldsymbol{R}_{N_i,i}^T\}^T, \ \boldsymbol{R}_{ji} = \boldsymbol{R}_j - \boldsymbol{R}_i.$$

Models of this type are naturally extensive.

The importance of preserving the symmetries

Symmetries: Translational, rotational and permutational



Preserving symmetry: Poor man's version

- remove translational and rotational symmetry by fixing a local frame of reference
- remove permutational symmetry by fixing an ordering of the atoms in the neighborhood

creates small discontinuity when atoms switch their orders.

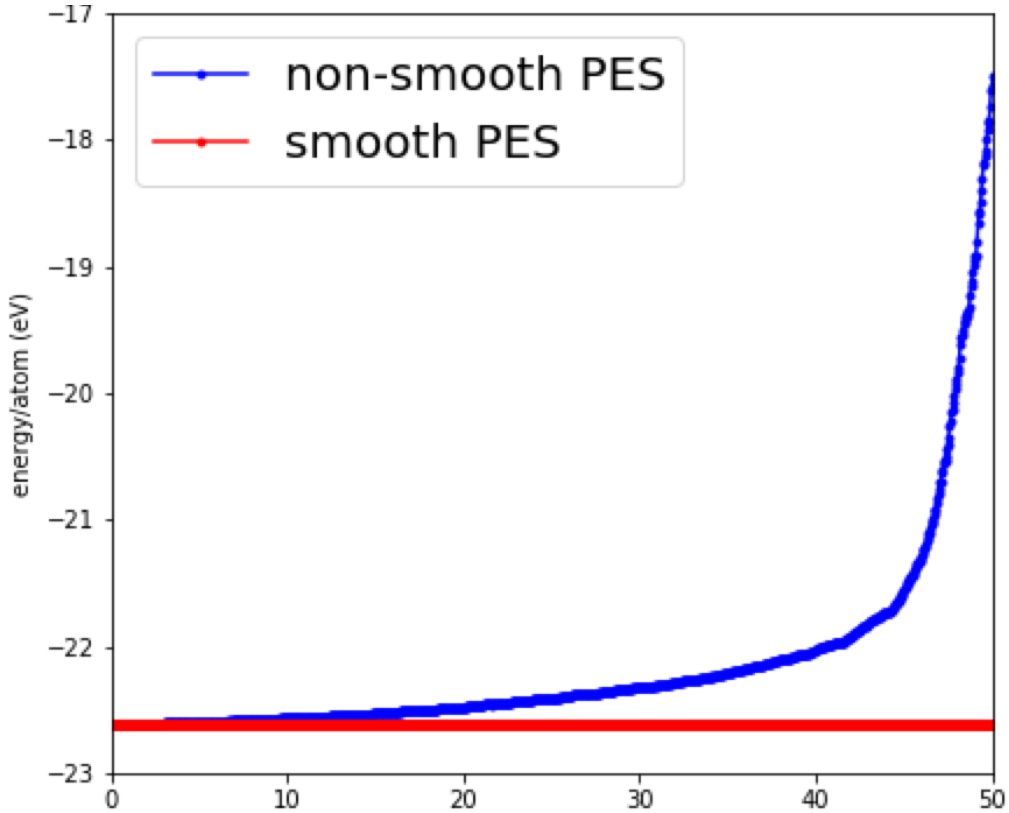


Figure: deep potential molecular dynamics (DPMD)

Preserving the symmetries

Translation, rotation, and permutation.

$$\hat{T}_{\boldsymbol{b}} f(\boldsymbol{r}) = f(\boldsymbol{r} + \boldsymbol{b}), \ \hat{R}_{\mathcal{U}} f(\boldsymbol{r}) = f(\boldsymbol{r}\mathcal{U}),$$

$$\hat{P}_{\sigma} f(\boldsymbol{r}) = f(\boldsymbol{r}_{\sigma(1)}, \boldsymbol{r}_{\sigma(2)}, ..., \boldsymbol{r}_{\sigma(N)})$$

- Translation and Rotation:

$$\Omega^{i}_{jk} = \boldsymbol{r}_{ji} \cdot \boldsymbol{r}_{ki}.$$

Lemma: Ω^{i}_{jk} is an overcomplete array of basic invariants with respect to rotation, reflection, and translation.

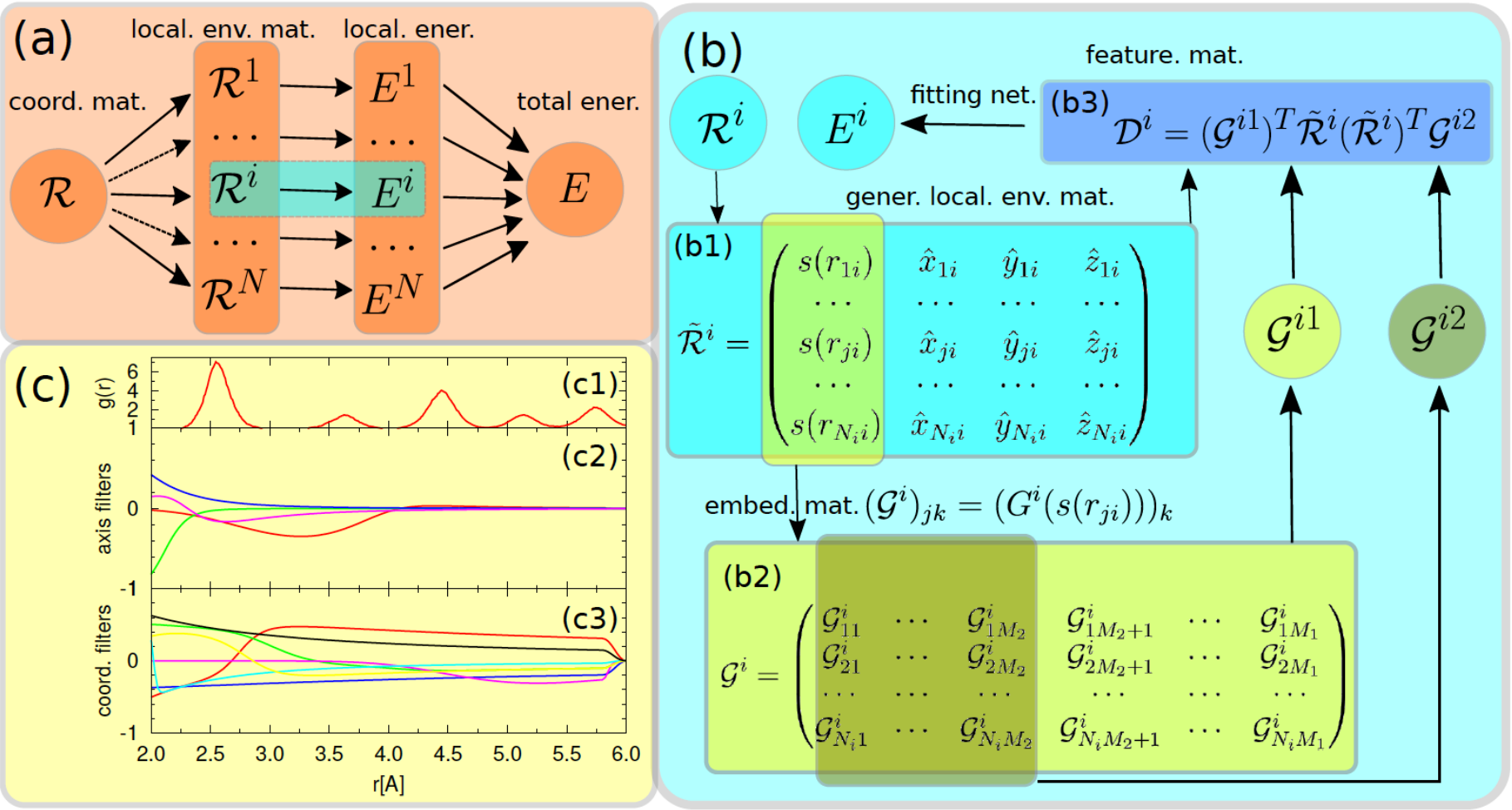
- Permutation:

$$\sum_{j \in \mathcal{N}(i)} g(\boldsymbol{r}_{ji}) \boldsymbol{r}_{ji}.$$

Lemma: A function $f(\boldsymbol{r}_{1i}, ..., \boldsymbol{r}_{ji}, ..., \boldsymbol{r}_{N_i i})$ is invariant to the permutation of instances in $\boldsymbol{r}_{ji}$, if and only if it can be decomposed in the form $\rho(\sum_{j \in \mathcal{N}(i)} g(\boldsymbol{r}_{ji}) \boldsymbol{r}_{ji})$, for suitable transformations g and ρ.

Deep Potential: smooth version

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 $(\tilde{\mathcal{R}}^i)^T\mathcal{G}^{i2}$.)

DP-GEN: generating optimal dataset using concurrent learning

The exploration-labeling-training procedure

- Exploration:
  - Sample the (T, p) space
  - For each value of (T, p), sample the canonical ensemble (using DPMD).
  - In addition, initialize the exploration with a variety of different initial configurations.
- Labeling: Using DFT (with periodic boundary condition)
- Training: Using "Deep Potential"

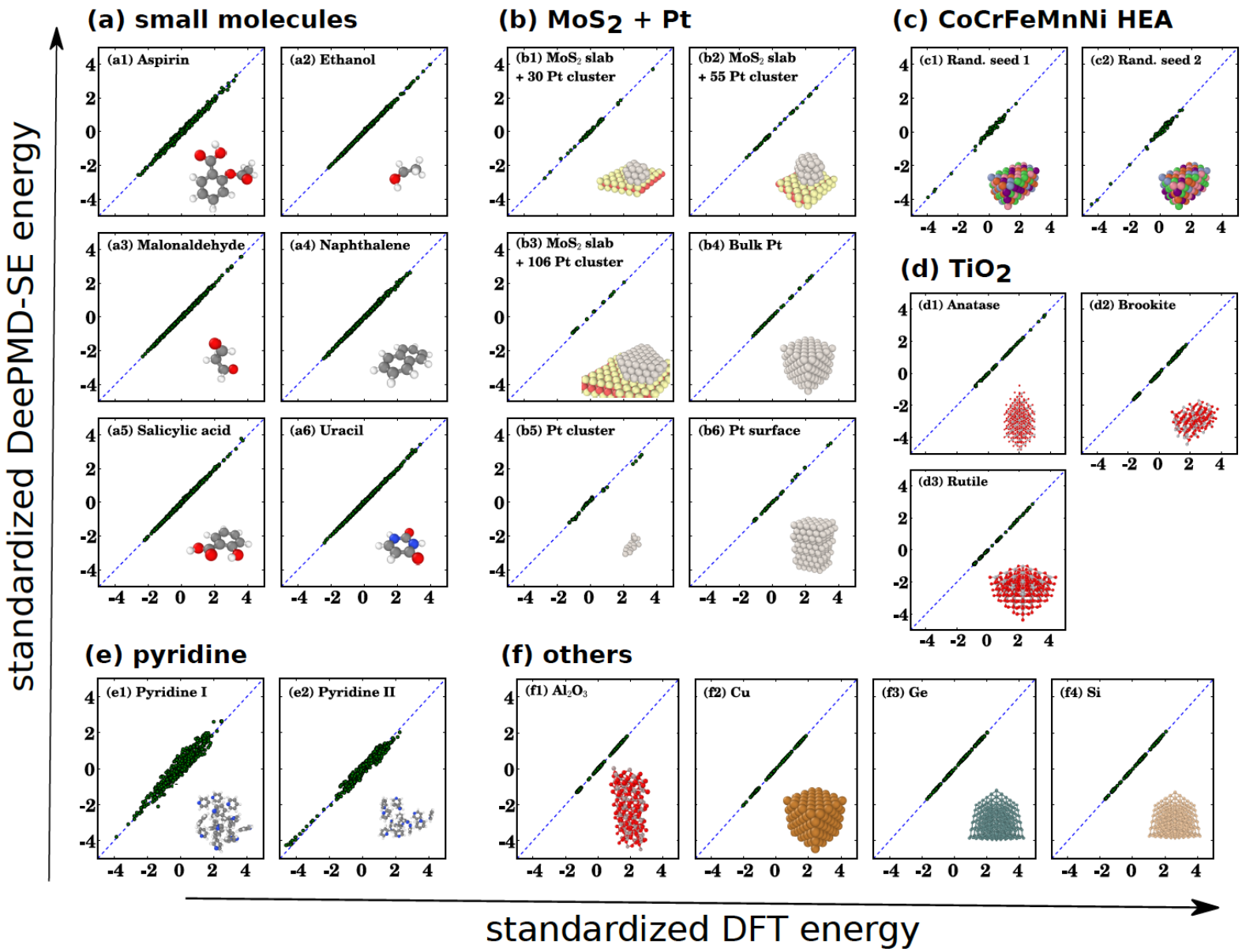
Example: Al, Mg, Al-Mg

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

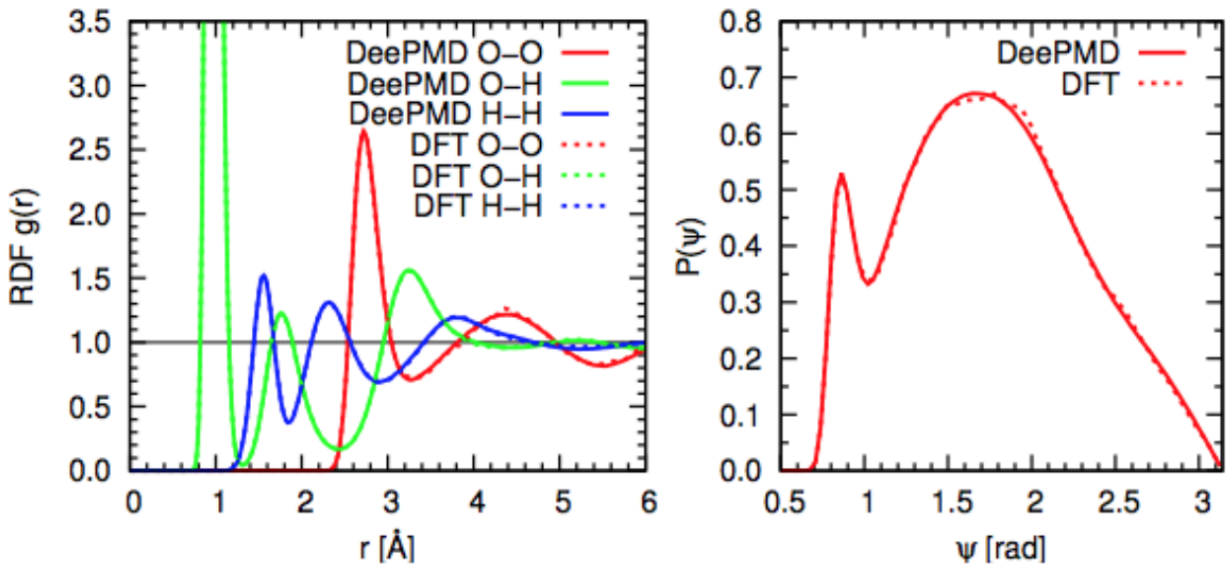
$\sim$0.005% configurations explored are selected for labeling.

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

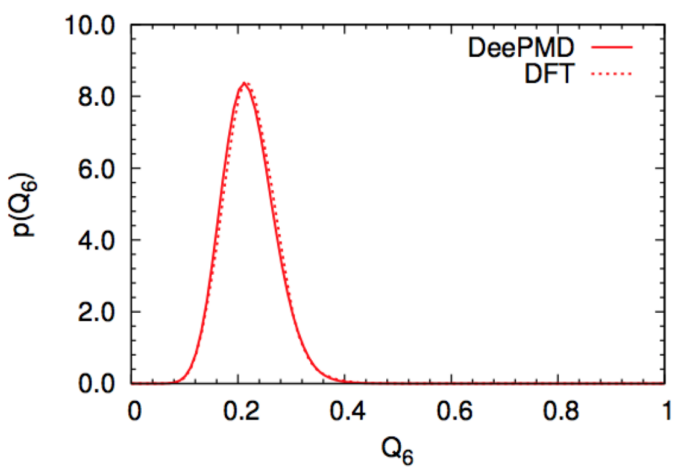


Case 2: structural information of DFT water

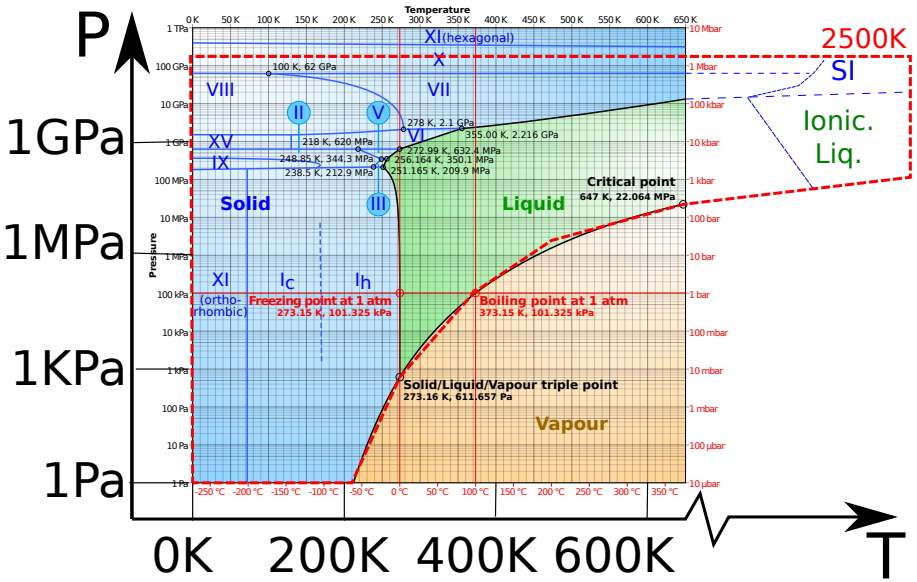
Radial and angular distribution function of liquid water (PI-AIMD):

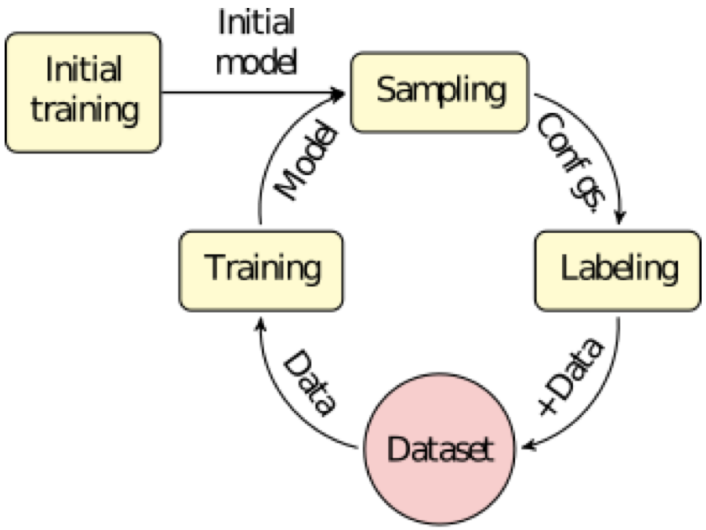


Distribution of the Steinhardt order parameter $\bar{Q}_6$:



DP-GEN for water



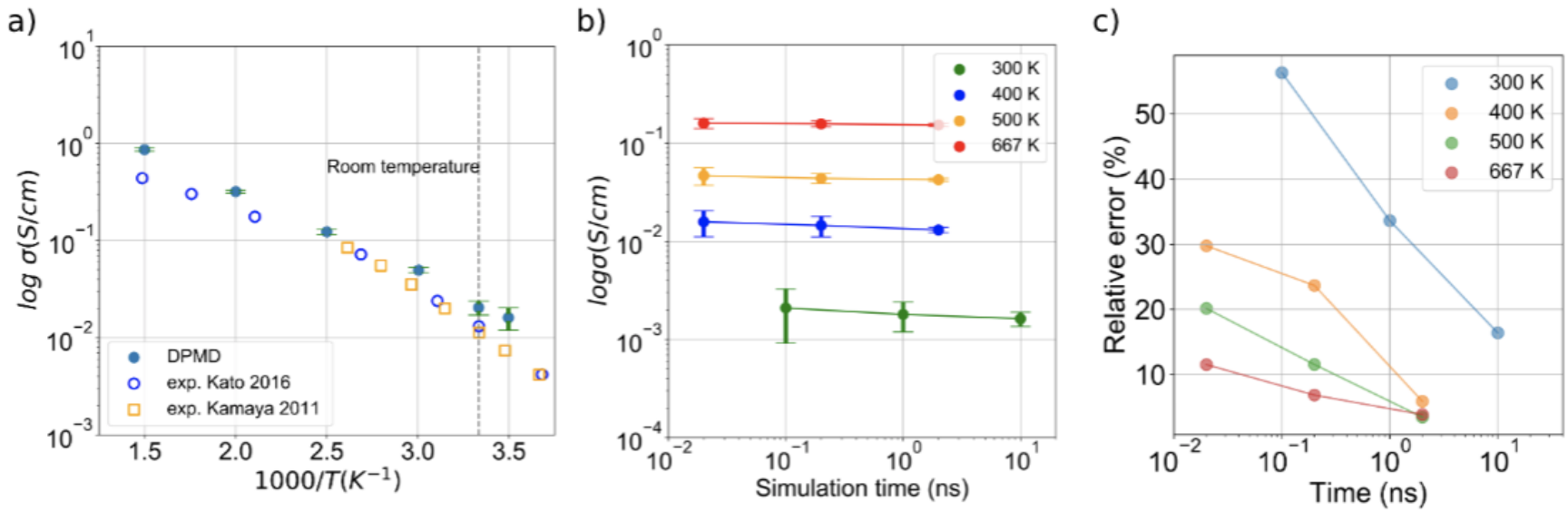


- **Reference model:** DFT at the classical SCAN level;
- **Starting configurations:** relaxed Ice I-XV at $T = 0$ K and equilibrated liquid at $T = 330$ K;
- **Range of thermodynamic conditions:** red dashed box;
- **number of MD snapshots:** DPMD exploration: 1.4 billion, DFT calculation: 32 thousand ($\sim$0.002% of the former).

  Typical AIMD trajectory: 100 thousand snapshots (50-100 *ps*).
- **number of DP-GEN iterations:** 100.

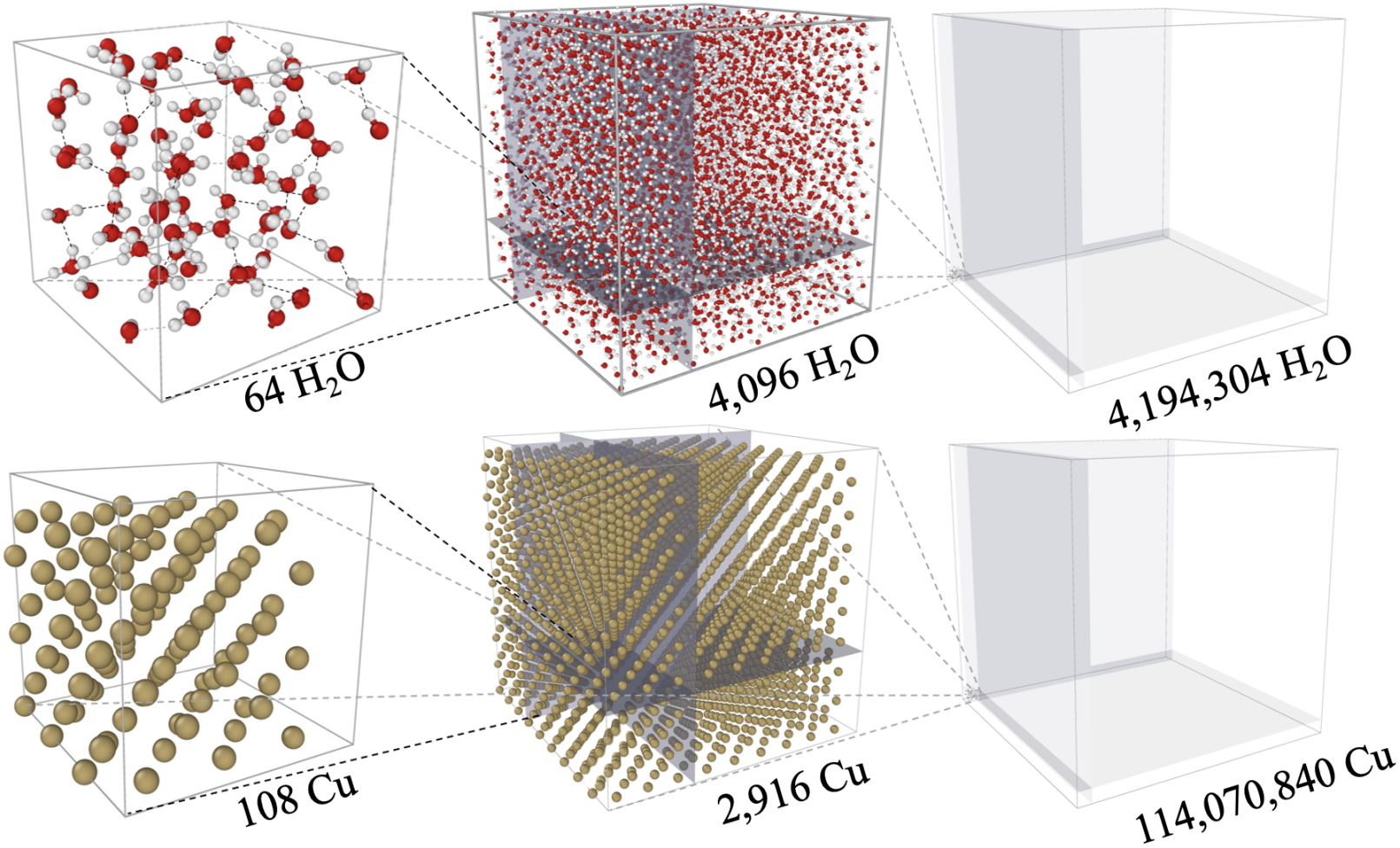
Lithium diffusion in solid-state electrolyte

Ability to handle multi-component systems, here the LiGePS-type systems.



Jun Cheng's group at Xiamen U

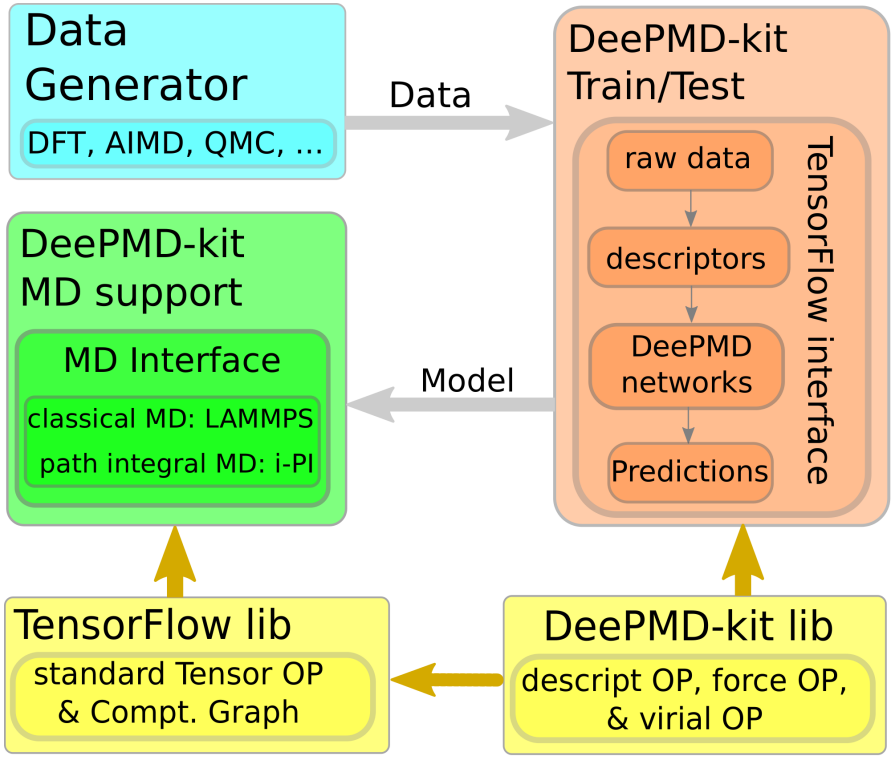
86 PFLOPS DeePMD simulation of 100M atoms

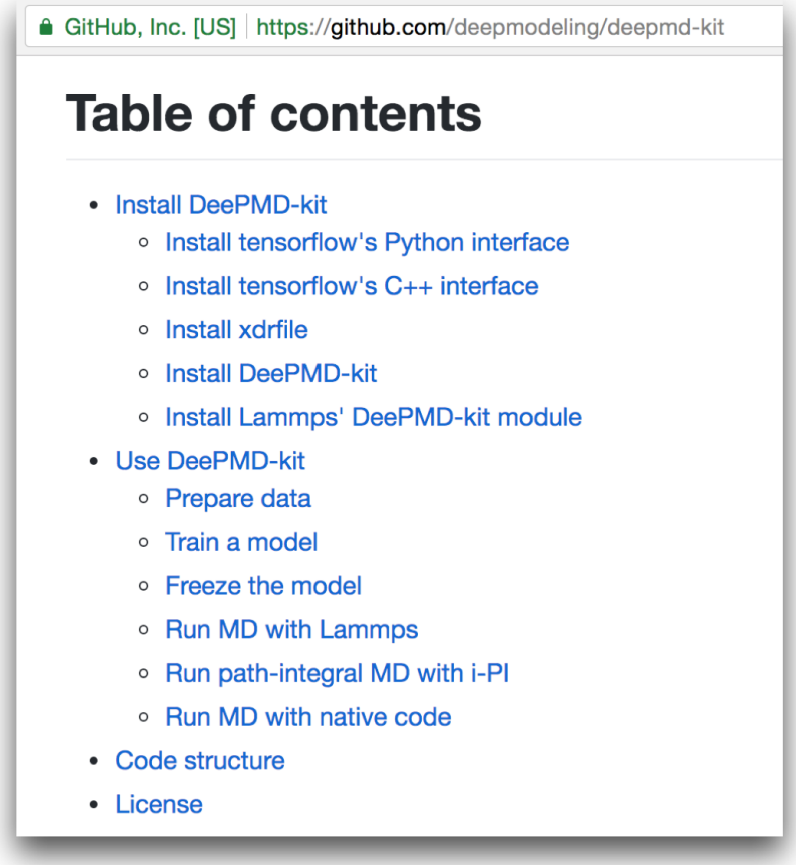


(a) AIMD + HPC; (b) DeePMD+1 GPU @ Home; (c) DeePMD+27360 GPUs @ Summit.

D. Lu, et al, arXiv: 2004.11658; W. Jia, et al, arXiv: 2005.00223

Open-source softwares: DeePMD-kit



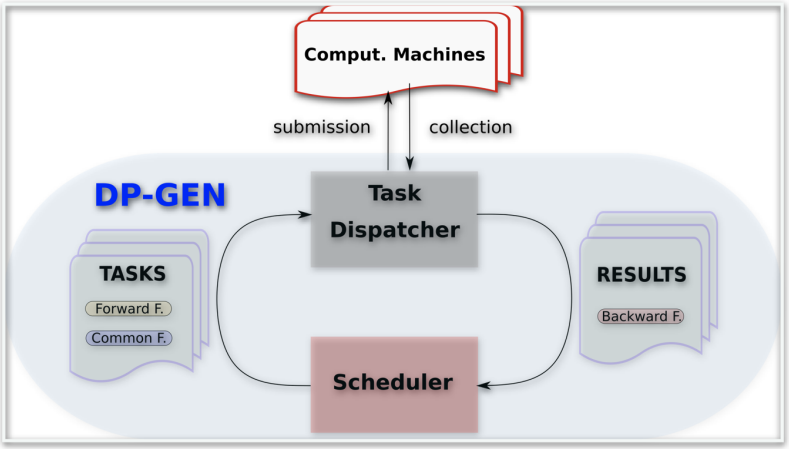


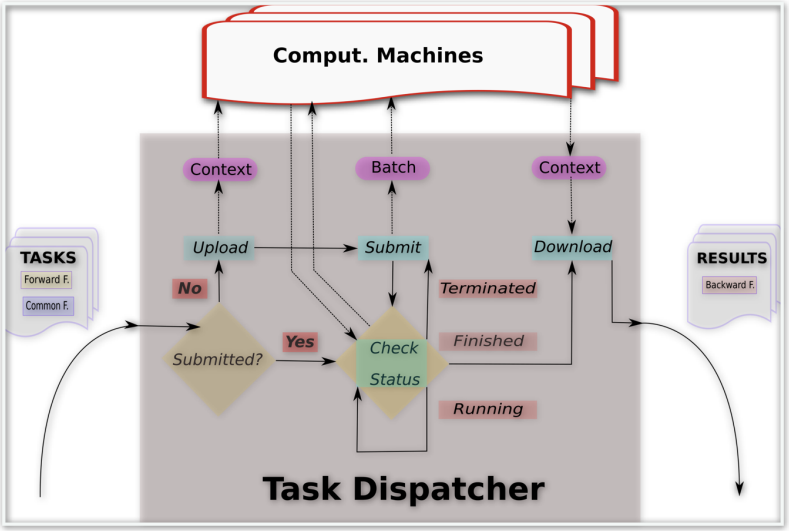
- TensorFlow: efficient network operators
- LAMMPS, i-PI; MPI/GPU support.

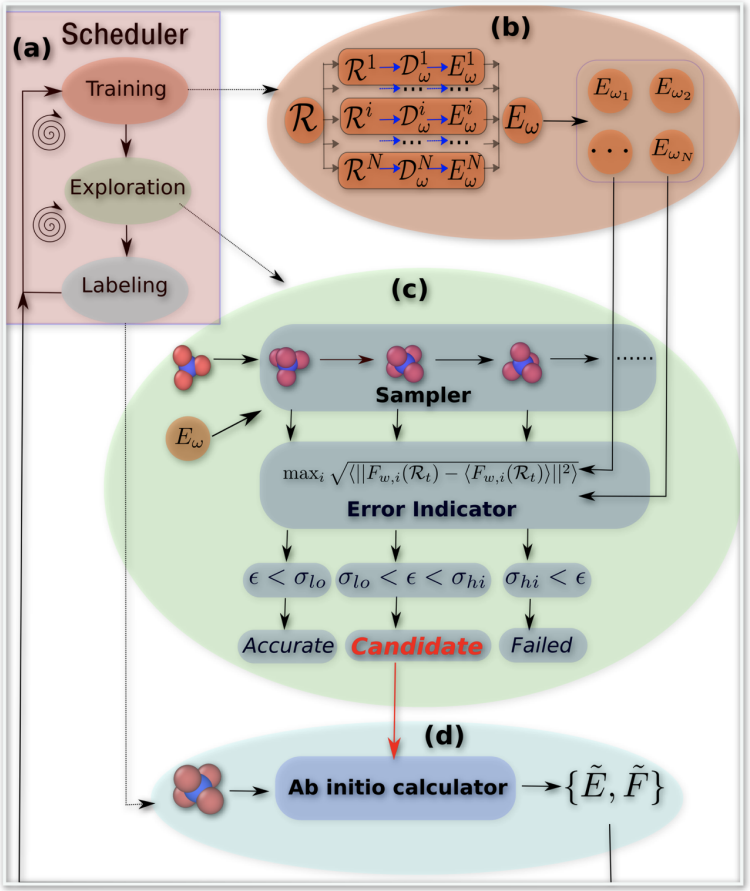
Free download from https://github.com/deepmodeling/deepmd-kit

H. Wang,. et al, .Comp.Phys.Comm., 0010-4655 (2018).

Open-source softwares: DP-GEN

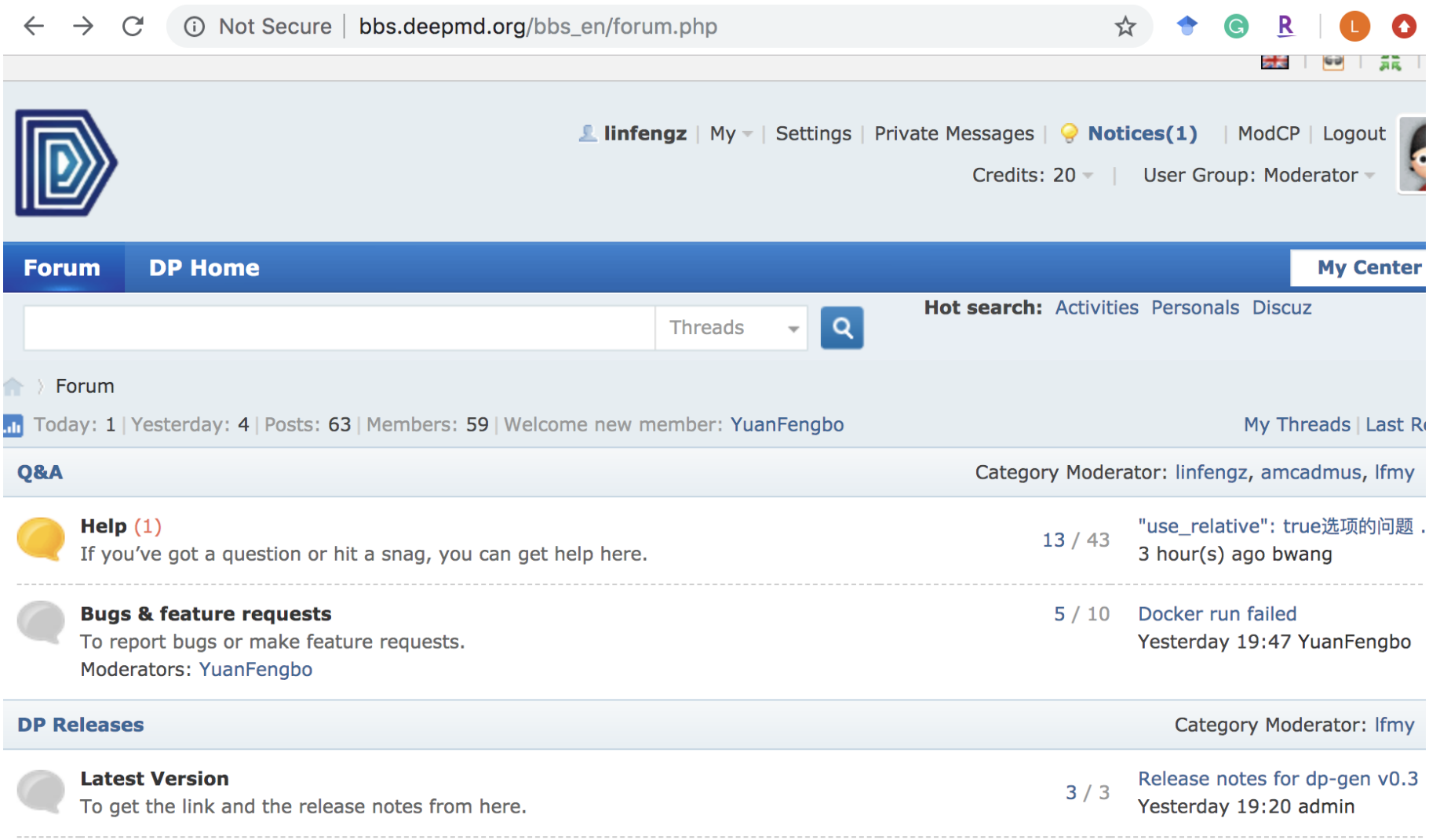






Free download from https://github.com/deepmodeling/dpgen

Discussion group



bbs.deepmd.org

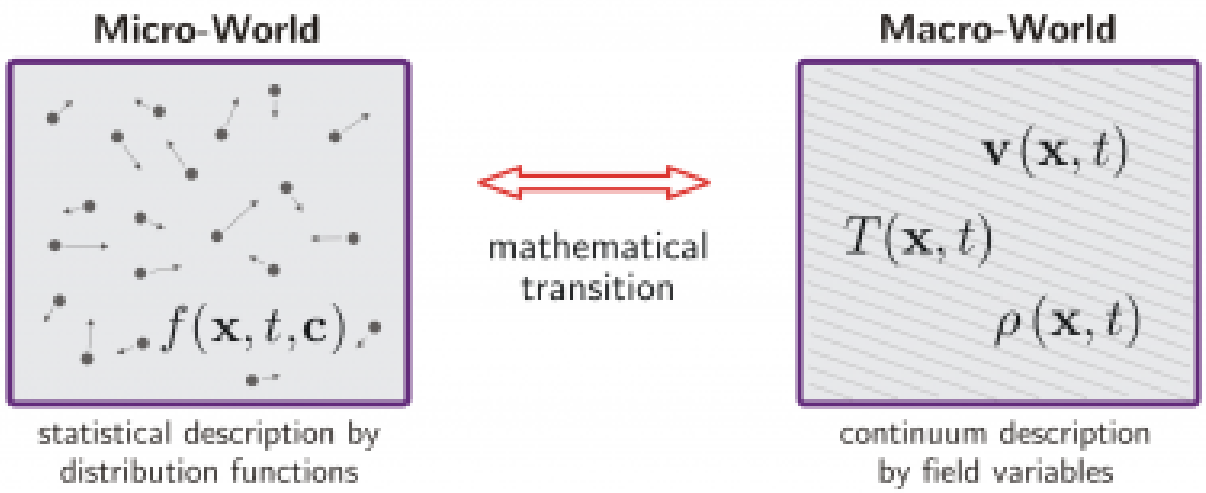
1. physical/chemical problems
   - understanding water (phase diagram of water, including reactive regions; phase transition: ice to water, ionic liquid to super-ionic ice; nuclear quantum effect: collective tunneling, isotope effect; reactive event: dissociation and recombination; water surface and 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.

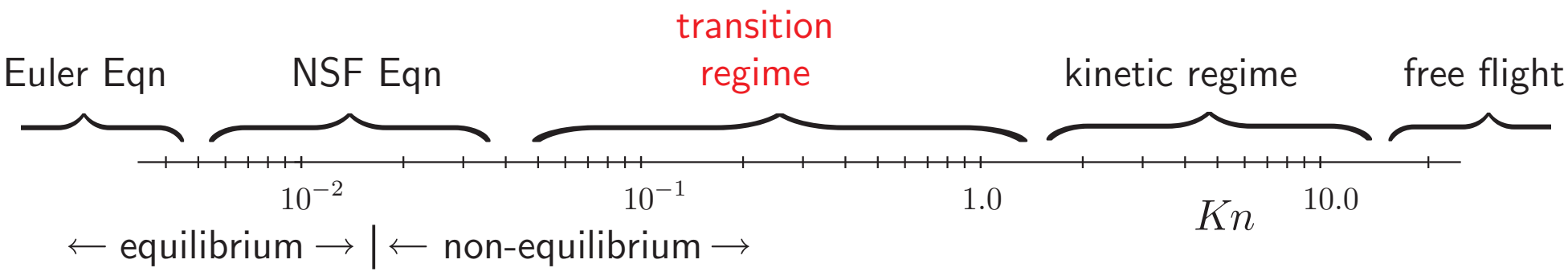
Outline

1. PDEs and fundamental laws of physics
2. Machine learning
3. Concurrent learning
4. Molecular modeling
5. Kinetic model for gas dynamics
6. Concluding remarks

Modeling gas dynamics

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





Boltzmann Equation

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

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

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

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

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

Conventional Moment Method

Proceed in 3 steps:

1. Start with the choice of a finite-dimensional linear subspace of functions of $\boldsymbol{v}$ (usually to be polynomials, e.g., Hermite polynomials).

2. Expand $f(\boldsymbol{x}, \boldsymbol{v}, t)$ using these functions as bases and take the coefficients as moments (including macroscopic variables ρ, $\boldsymbol{u}$, T, etc.).

3. Finally close the system with simplified assumptions, e.g., truncating moments of higher orders

$$\begin{cases} \partial_t \boldsymbol{U} + \nabla_{\boldsymbol{x}} \cdot \boldsymbol{F}(\boldsymbol{U}, \boldsymbol{W}) = 0, \\ \partial_t \boldsymbol{W} + \nabla_{\boldsymbol{x}} \cdot \boldsymbol{G}(\boldsymbol{U}, \boldsymbol{W}) = \dfrac{1}{\varepsilon} \boldsymbol{R}(\boldsymbol{U}, \boldsymbol{W}). \end{cases}$$

$\boldsymbol{G}(\boldsymbol{U}, bW) \sim \int \boldsymbol{v} w(\boldsymbol{v}) f d\boldsymbol{v}, \boldsymbol{R}(\boldsymbol{U}, \boldsymbol{W}) \sim \int Q(\boldsymbol{v}) w(\boldsymbol{v}) d\boldsymbol{v}$

For instance, in Grad 13-moment system, $(\boldsymbol{U}, \boldsymbol{W})$ is constructed based on the moments of the bases $\{1, \boldsymbol{v}, (\boldsymbol{v} - \boldsymbol{u}) \otimes (\boldsymbol{v} - \boldsymbol{u}), |\boldsymbol{v} - \boldsymbol{u}|^2 (\boldsymbol{v} - \boldsymbol{u})\}$.

Machine learning-based moment method

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

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.

2: Learn the Fluxes and Source Terms in the PDE

Recall the general conservative form of the moment system

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

Rewrite it into (variance reduction)

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

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

Our goal is to obtain ML models for $\tilde{\boldsymbol{F}}, \tilde{\boldsymbol{G}}, \boldsymbol{R}$ 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

The ELT algorithm

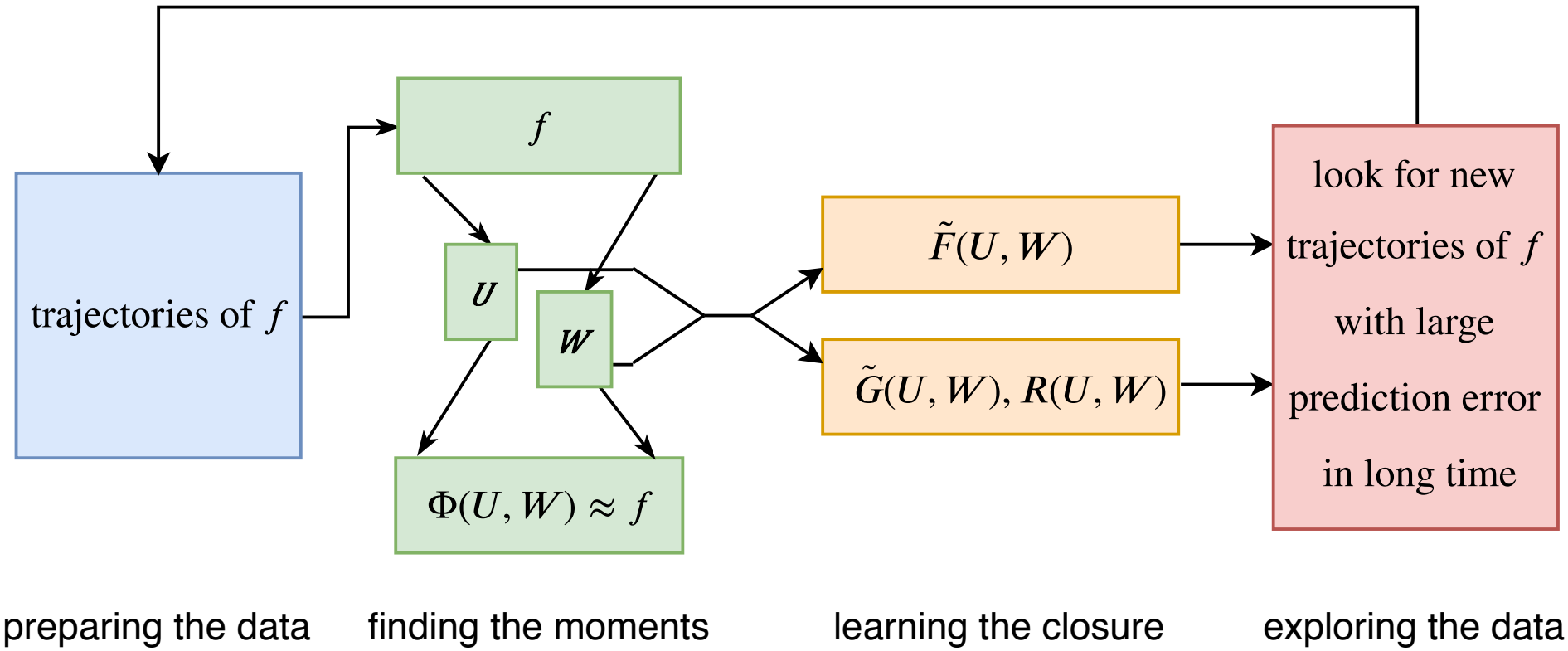


Figure: Schematic diagram of the machine learning-based moment method

- exploration: random initial conditions made up of waves and discontinuities
- labeling: solving kinetic equation (Boltzmann equation for Maxwell molecules)
- training: Galilean invariance

Galilean Invariant Moments

Galilean invariance of the Boltzmann equation:

$$f'(\boldsymbol{x}, \boldsymbol{u}, t) = f(\boldsymbol{x} - t\boldsymbol{u}', \boldsymbol{v} - \boldsymbol{u}', t).$$

Moments:

$$\boldsymbol{W}_{\mathsf{Gal}} = \Psi(f) = \int f(\boldsymbol{v}) \mathbf{w}\left(\frac{\boldsymbol{v} - \boldsymbol{u}}{\sqrt{T}}\right) \mathrm{d}\boldsymbol{v}.$$

Closure:

$$\partial_t \int_{\mathbb{R}^D} f(\boldsymbol{v}) \mathbf{w}\left(\frac{\boldsymbol{v} - \boldsymbol{u}_j}{\sqrt{T_j}}\right) \mathrm{d}\boldsymbol{v} + \nabla_{\boldsymbol{x}} \cdot \int_{\mathbb{R}^D} f(\boldsymbol{v}) \mathbf{w}\left(\frac{\boldsymbol{v} - \boldsymbol{u}_j}{\sqrt{T_j}}\right) \boldsymbol{v}^{\mathrm{T}} \mathrm{d}\boldsymbol{v} = \int_{\mathbb{R}^D} \frac{1}{\varepsilon} Q(f) \mathbf{w}\left(\frac{\boldsymbol{v} - \boldsymbol{u}_j}{\sqrt{T_j}}\right) \mathrm{d}\boldsymbol{v}.$$

$$\partial_t \boldsymbol{W}_{\mathsf{Gal}} + \nabla_{\boldsymbol{x}} \cdot \boldsymbol{G}_{\mathsf{Gal}}(\boldsymbol{U}, \boldsymbol{W}_{\mathsf{Gal}}; \boldsymbol{U}_j) = \frac{1}{\varepsilon} \boldsymbol{R}_{\mathsf{Gal}}(\boldsymbol{U}, \boldsymbol{W}_{\mathsf{Gal}}).$$

The data efficiency is better than the previous one since it learns the dynamical system more intrinsically.

$\varepsilon \sim$ Log10-Uniform$(-3, 1)$, constant across the domain; initial profiles consist of a combination of a few sin waves and shocks.

Size of dataset array: $200 \times 100 \times 48 \times 48 \times 100$. Specify $\boldsymbol{W} \in \mathbb{R}^9$.

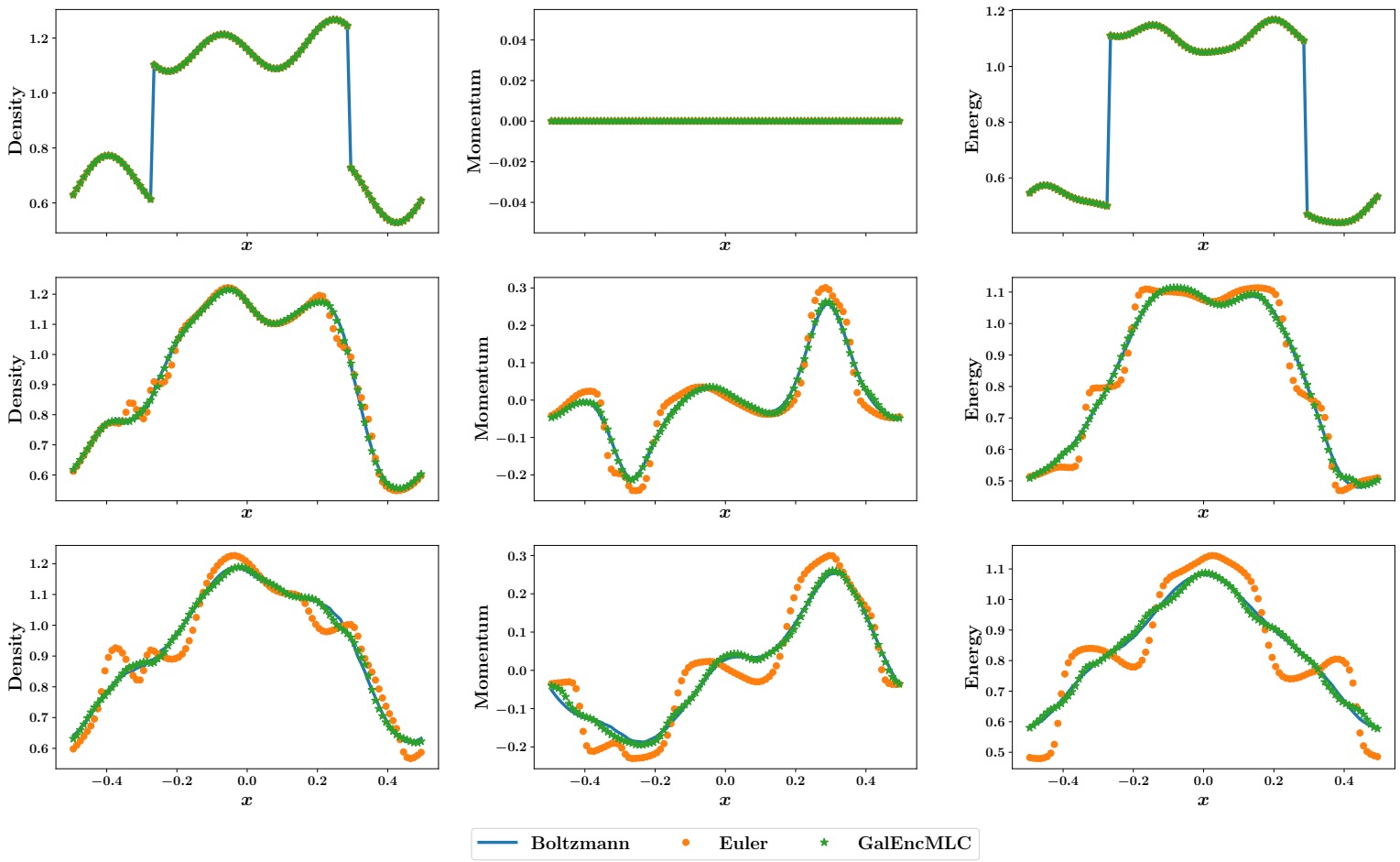


Figure: Sample profiles of $\rho, \rho\boldsymbol{u}, E$ (from left to right) at $t = 0, 0.05, 0.1$ (from top to bottom), $\varepsilon = 8.10$

ε varies from 10^{-3} to 10 in the domain; initial profiles are the same as before

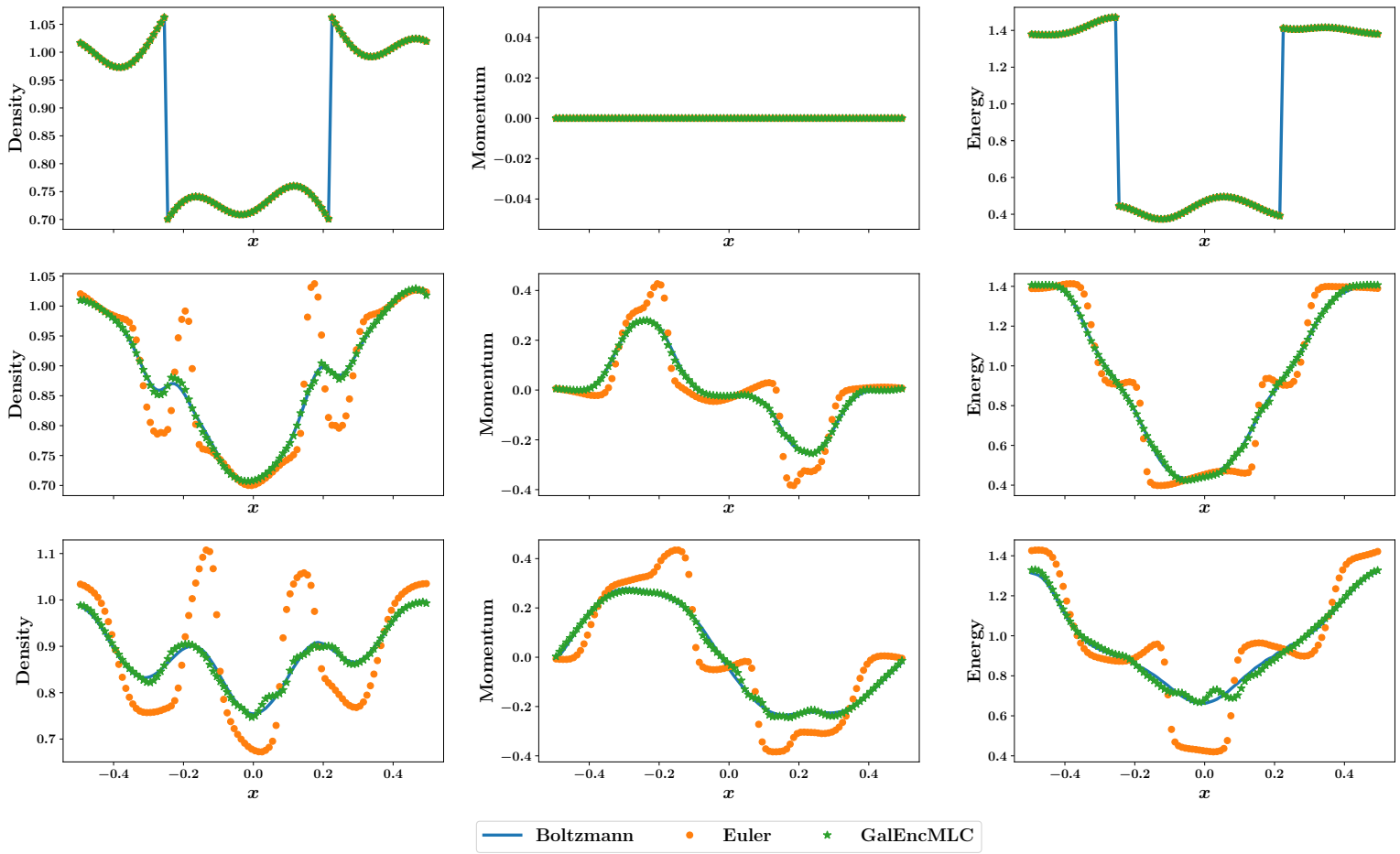
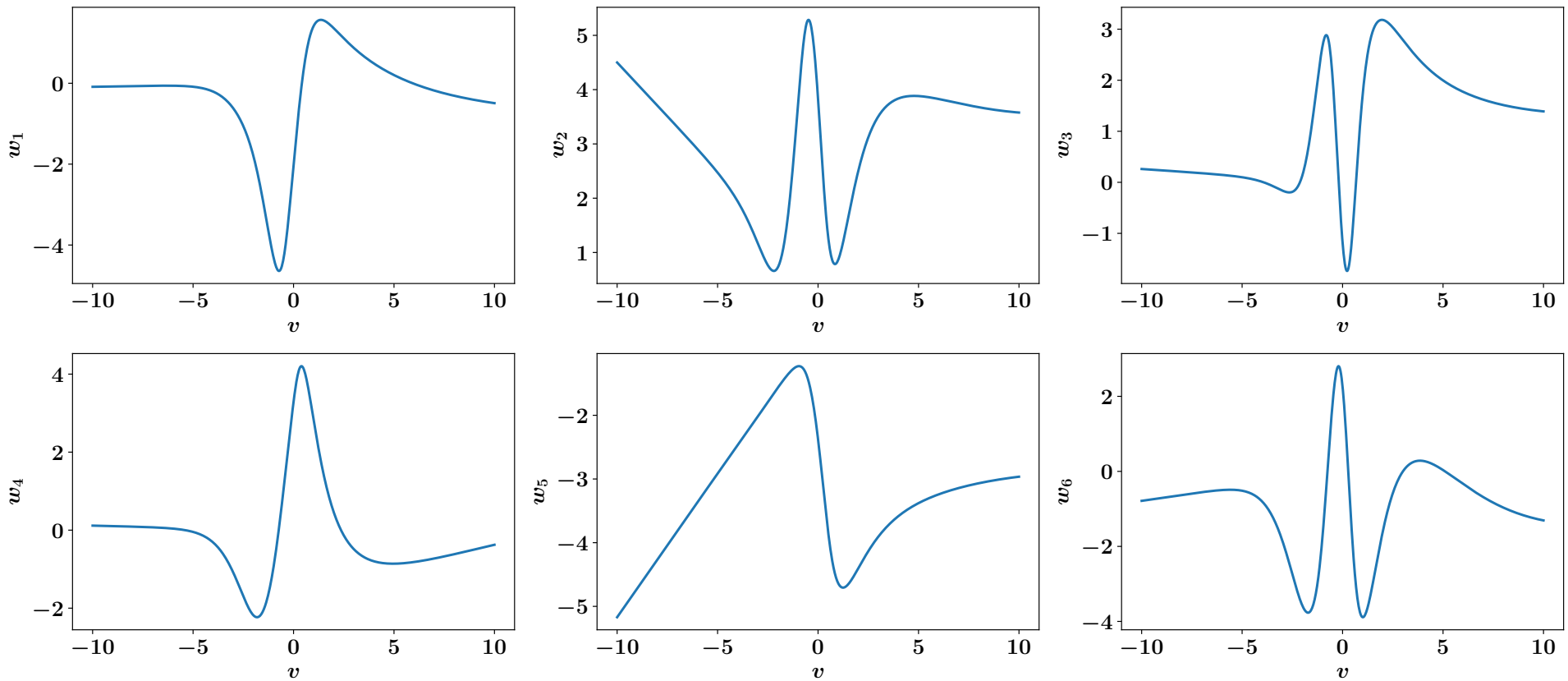


Figure: Profiles of $\rho, \rho\boldsymbol{u}, E$ (from left to right) at $t = 0, 0.05, 0.1$ (from top to bottom)

Numerical results

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



Outline

1. PDEs and fundamental laws of physics
2. Machine learning
3. Concurrent learning
4. Molecular modeling
5. Kinetic model for gas dynamics
6. Concluding remarks

Concluding remarks

- The integration of machine learning with physics-based modeling opens up a whole new chapter in the way we do scientific and engineering modeling
- Concurrent learning ensures generation of the optimal dataset
- It is important to take "physics" into account
- These are new physical models, not just algorithms