UNCERTAINTY QUANTIFICATION IN SCIENTIFIC MACHINE LEARNING

Uncertainty in Scientific Machine Learning from PDEs to Molecules

Advanced School on Foundation Models for Scientific Discovery

The Abdus Salam International Centre for Theoretical Physics, Italy

DARIO COSCIA

SISSA, University of Amsterdam

LECTURE OUTLINE 2/2

- **Metrics for Uncertainty Quantification**
  - Scoring Rules
  - Calibration and Sharpness
  - Correlation

- **Uncertainty Quantification for Sequence Models**
  - Introduction to Autoregressive Neural Operators
  - Functional Generative Networks, PDE-Refiner, BARNN

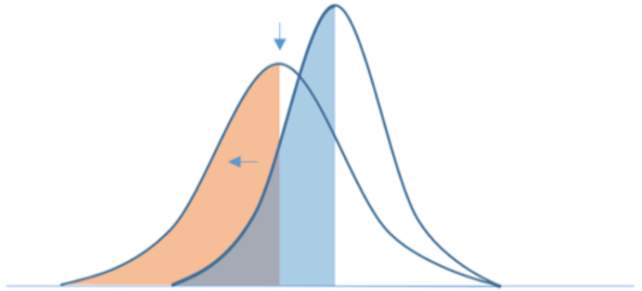
- **Uncertainty Quantification for Message Passing Network**
  - Introduction to Graph Networks and Machine Learning Interatomic Potentials
  - UQ in Machine Learning Interatomic Potentials

SHORT RECAP...

Take Aways: 1st There are two types of uncertainty: irreducible uncertainty (aleatoric) and model uncertainty (epistemic).

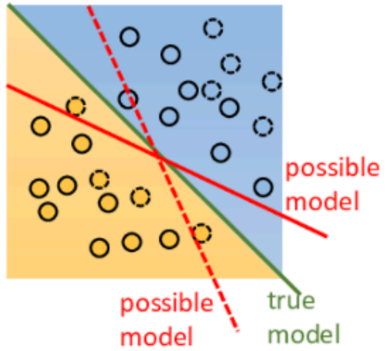
***Aleatoric** Uncertainty*

$$y_{\text{data}} = \mathcal{M}_{\text{true}}(x) + z \quad , z \sim p(z)$$



***Epistemic** Uncertainty*

$$\mathcal{M}_{\text{model}}(x \mid \boldsymbol{\omega}) \quad , \boldsymbol{\omega} \sim p(\boldsymbol{\omega})$$



SHORT RECAP...

***Take Aways**: 2nd* The Bayesian approach enables modelling both epistemic and aleatoric uncertainties

$$p(y \mid x) = \int p(y \mid x, \omega) p(\omega \mid \mathcal{D}) \mathrm{d}\omega \qquad \text{with,} \qquad p(\omega \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \omega) p(w)}{p(\mathcal{D})}$$

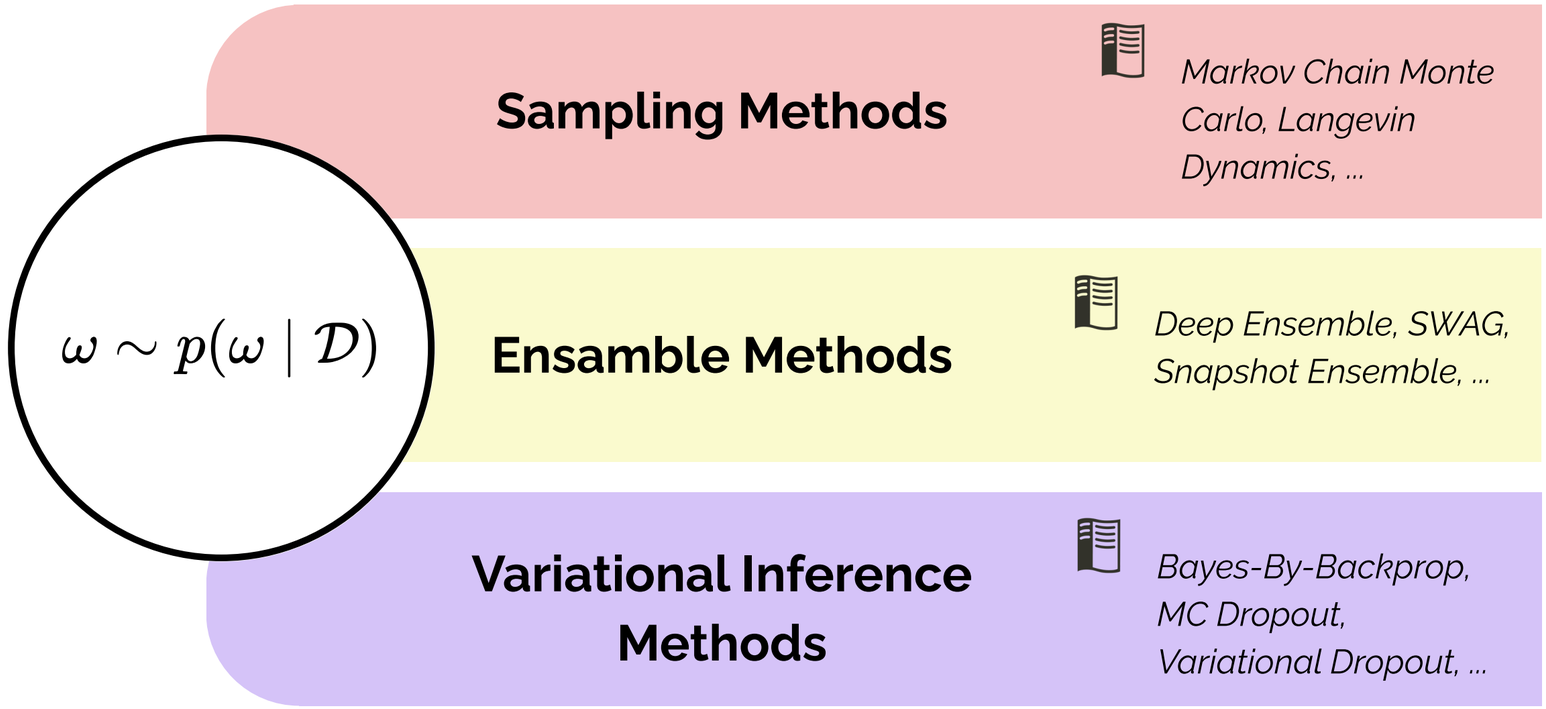
Using the **law of total variance** we can decompose the uncertainty

$$\mathbb{V}\mathrm{ar}[y \mid x] = \mathbb{E}_{\omega \sim p(\omega \mid \mathcal{D})} \left[\mathbb{V}\mathrm{ar}[y \mid x, \omega] \right] + \mathbb{V}\mathrm{ar}_{\omega \sim p(\omega \mid \mathcal{D})} \left[\mathbb{E}[y \mid x, \omega] \right]$$

aleatoric ***epistemic***

SHORT RECAP...

Take Aways: *3rd* Many ways to compute the posterior distribution



A SMALL NOTE ON LIKELIHOOD FUNCTION

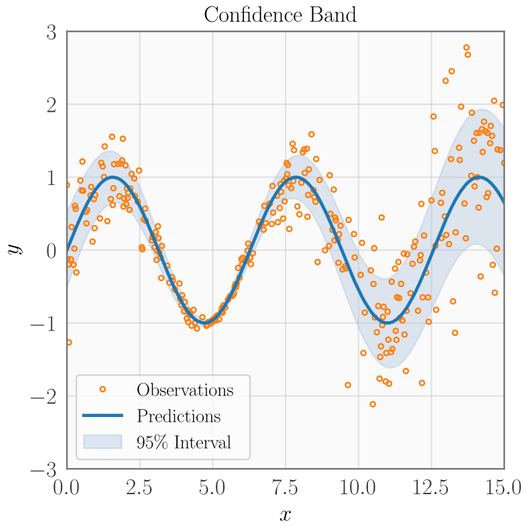
Setting: We are given a dataset $\mathcal{D} = \{x_i, y_i\}_{i=1}^N$ of *N* observations, and we want a model such that $y_i = \phi_\omega(x_i)$ (regression task), **which likelihood do I choose?**

Gaussian model: $$p(\mathcal{D} \mid \omega) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_i - \phi_\omega(x_i))^2}{2\sigma^2}} \implies -\nabla_\omega \log p(\mathcal{D} \mid \omega) \propto \nabla_\omega \mathrm{MSE}(y, \phi_\omega(x))$$

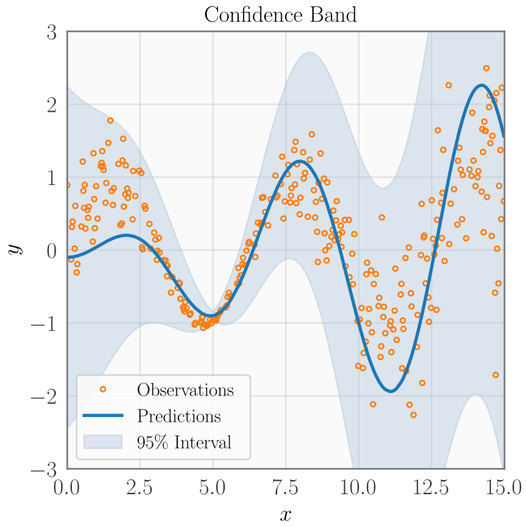
Laplace model: $$p(\mathcal{D} \mid \omega) = \prod_{i=1}^{N} \frac{1}{2b} e^{-\frac{|y_i - \phi_\omega(x_i)|}{b}} \implies -\nabla_\omega \log p(\mathcal{D} \mid \omega) \propto \nabla_\omega \mathrm{MAE}(y, \phi_\omega(x))$$

HOW TO MEASURE UNCERTAINTY?

Selecting the best model is nuanced: one may be overconfident but accurate, while another may be conservative yet inaccurate



Model A



Model B

Uncertainty ToolBox: *https://github.com/uncertainty-toolbox/uncertainty-toolbox*

A (POSSIBLE) RECIPE TO MEASURE UNCERTAINTY

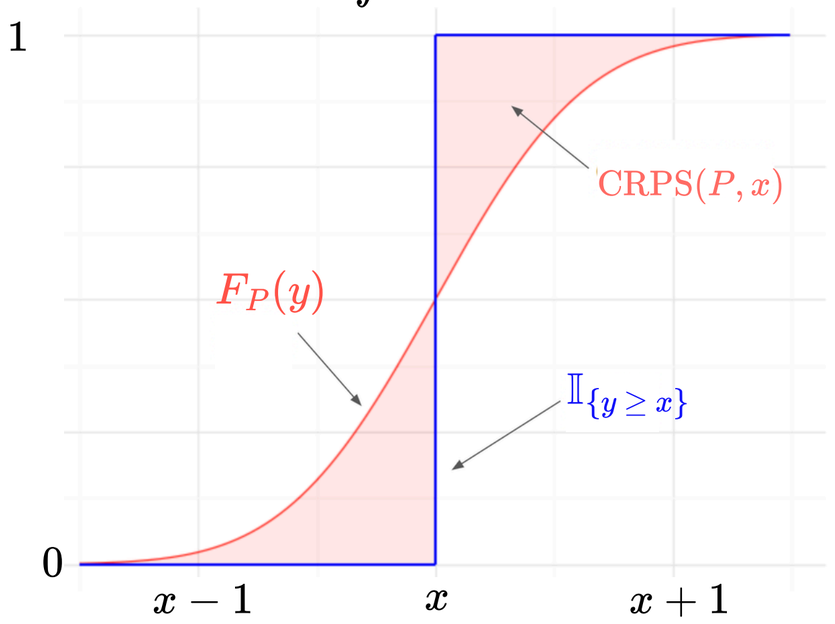
METRIC	WHEN TO USE	EXAMPLE
Probabilistic Scoring Rules	*Does the predicted distribution match observations?*	*NLL, CRPS*
Calibration	*Do confidence levels match confidence in reality?*	*ECE, Miscalibration Area*
Correlation	*Do your uncertainties reflect actual mistakes?*	*Pearson, Spearman*

PROBABILISTIC SCORING RULES

*A **scoring rule** is a way to measure how good a probabilistic forecast is.*

*A scoring rule is **proper** if the expected score is maximized when the forecasted probabilities match the forecaster's true beliefs.*

$$\mathrm{CRPS}(P, x) = \int (F_P(y) - \mathbb{I}_{\{y \geq x\}})^2 dy$$



Gaussian Example: $P = \mathcal{N}(\mu, \sigma)$

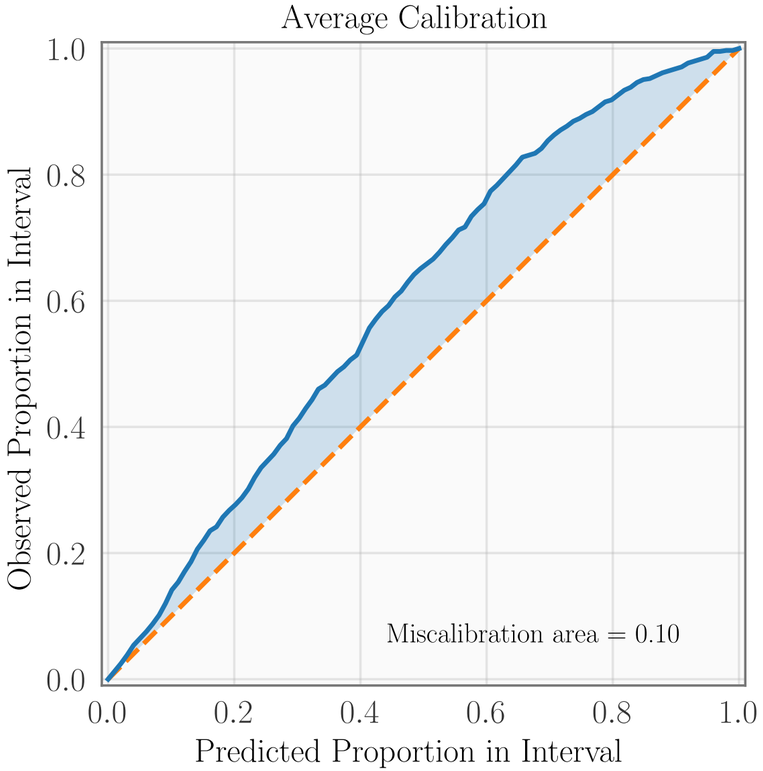
$$\mathrm{NLL}(P, x) = -\log P(x) = \frac{1}{2}\log(2\pi\sigma^2) + \frac{(x-\mu)^2}{2\sigma^2}$$

$$\mathrm{CRPS}(P, x) = \int (F_P(y) - \mathbb{I}_{\{y \geq x\}})^2 dy = \sigma\left[\frac{1}{\sqrt{\pi}} - 2\phi\left(\frac{x-\mu}{\sigma}\right) - \frac{x-\mu}{\sigma}\left(2\Phi\left(\frac{x-\mu}{\sigma}\right) - 1\right)\right]$$

CALIBRATION

*A **calibration** measure tells us how well predicted probabilities align with actual outcomes.*

*A model is **well-calibrated** if events predicted to occur with probability p actually happen approximately with proportion p.*



Gaussian Example: $P = \mathcal{N}(\mu, \sigma)$

$$\mathrm{ECE}(P) = \mathbb{E}_{p \sim U(0,1)}[|p - \mathbb{P}(x \leq F_P^{-1}(p))|] = \mathbb{E}_{p \sim U(0,1)}[|p - \mathbb{P}(x \leq \mu + \sigma\sqrt{2}\Phi^{-1}(2p-1)|]$$

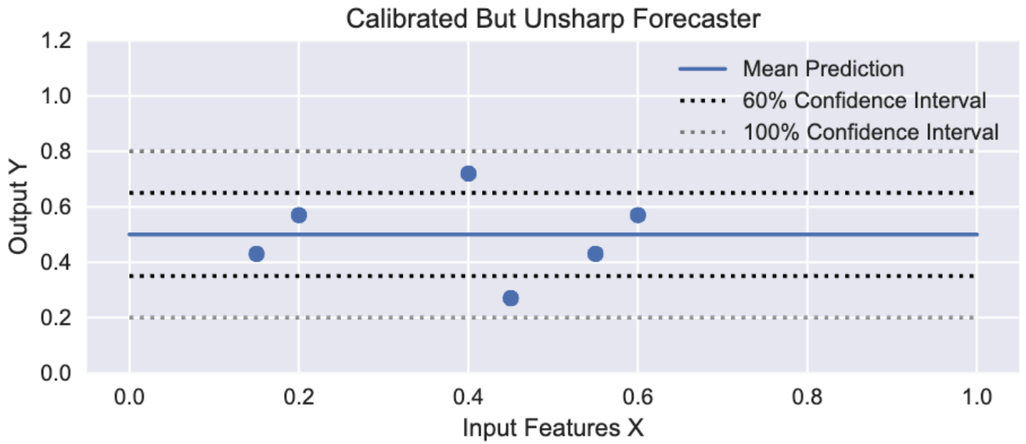
$$\mathrm{Area}(P) = \int_0^1 |p - \mathbb{P}(x \leq F_P^{-1}(p))|dp = \int_0^1 |p - \mathbb{P}(x \leq \mu + \sigma\sqrt{2}\Phi^{-1}(2p-1)|dp$$

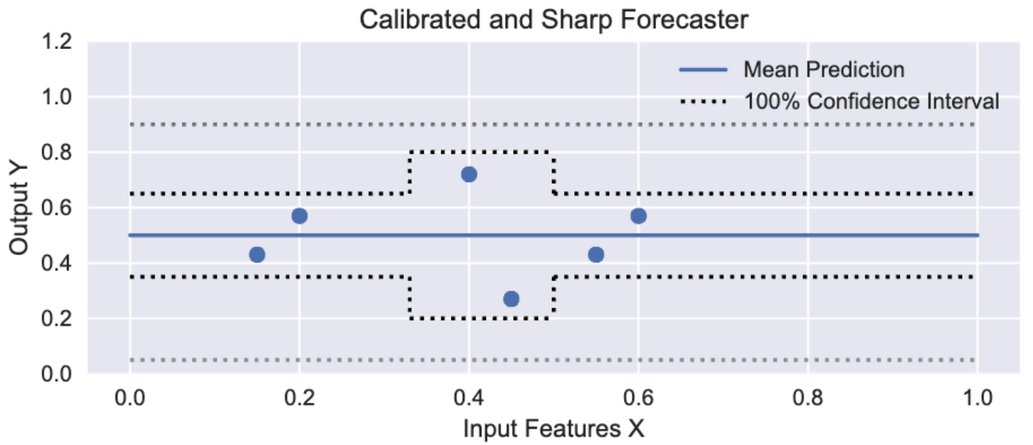
SHARPNESS

Calibration *by itself* ***is not sufficient*** *to produce a useful forecast!*

We also want the prediction to be sharp*, meaning it concentrates probability mass tightly around the output.*

$$\text{Sharp}(\text{P}, \text{x}) = \mathbb{V}ar_P(x)$$





CORRELATION

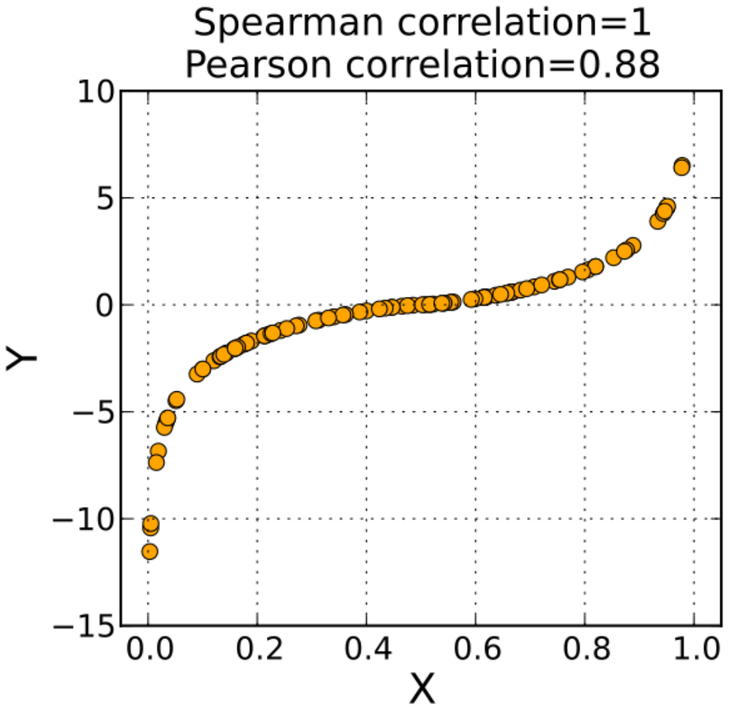
*A **correlation coefficient** measures how strongly two variables are related.*

*The **Pearson correlation** measures how well a straight line describes the relationship between two numerical variables — it's about linear association.*

$$\rho_{\text{pearson}} = \frac{Cov(X,Y)}{\sigma_X \sigma_Y}$$

*The **Spearman correlation** measures how well the order of the values in one variable matches the order in another — it's about consistent ranking, even if the relationship is not linear.*

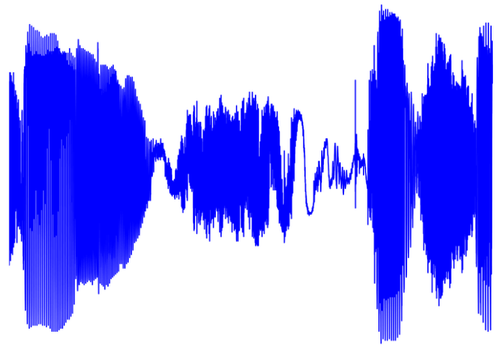
$$\rho_{\text{spearman}} = \frac{Cov(R[X],R[Y])}{\sigma_{R[X]} \sigma_{R[Y]}}$$



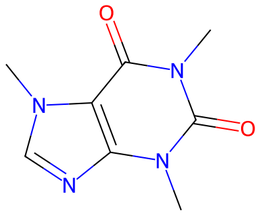
UNCERTAINTY QUANTIFICATION IN SEQUENCE MODELS

ChatGPT

Radford, Alec. "Improving language understanding by generative pre-training."

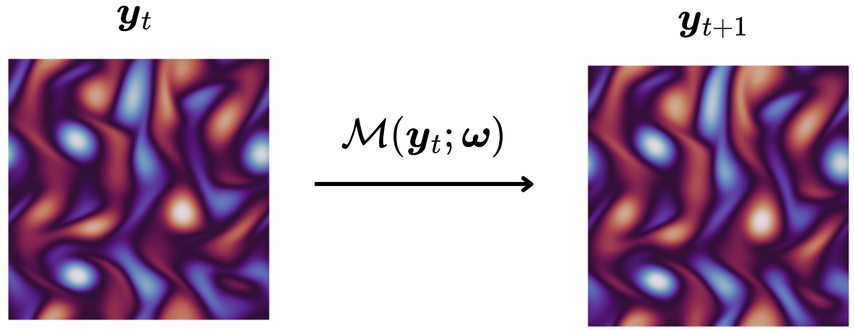


Van Den Oord, Aaron, et al. "Wavenet: A generative model for raw audio."



CN1C=NC2=C1C(=O)N(C(=O)N2C)C

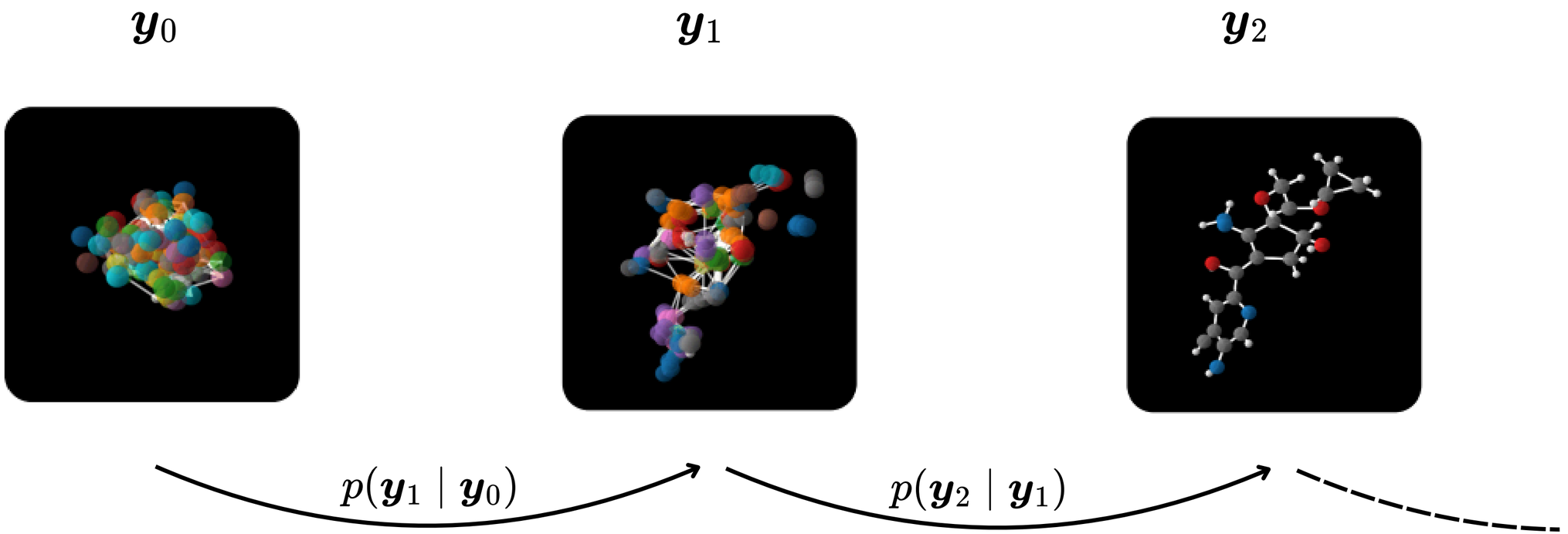
Segler, Marwin HS, et al. "Generating focused molecule libraries for drug discovery with recurrent neural networks."



Brandstetter, Johannes, Daniel Worrall, and Max Welling. "Message passing neural PDE solvers."

AUTOREGRESSIVE AND RECURRENT MODELS

Autoregressive and Recurrent models predict the **next state** in a sequence by processing the **previous states** in that sequence or by keeping a **memory of past states**



AUTOREGRESSIVE MODELS FOR LANGUAGE

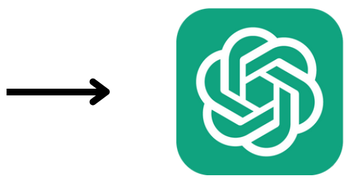
$$p(\boldsymbol{y}_0, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_T) = \prod_{t=1}^{T} p(\boldsymbol{y}_t \mid \boldsymbol{y}_{t-1}, \ldots, \boldsymbol{y}_0)$$

The **joint probability** distribution is a **product of factorized distributions** over states

The factorized distribution is usually a **parametric model** with optimizable deterministic weights

The cat is on the

$\boldsymbol{y}_0$ $\boldsymbol{y}_1$ $\boldsymbol{y}_2$ $\boldsymbol{y}_3$ $\boldsymbol{y}_4$



table

$\boldsymbol{y}_5$

AUTOREGRESSIVE MODELS FOR PDES

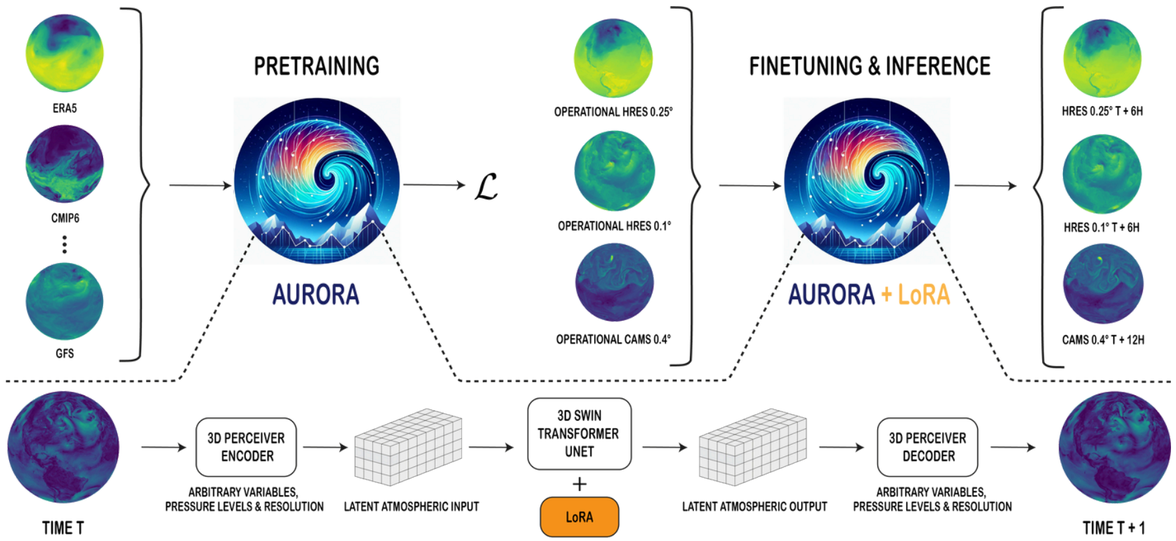
Autoregressive models can be used to build Neural Solvers to ***solve PDEs*** or perform ***weather forecast*** by iteratively applying the emulator on its output given initial states

Simple Autoregressive Model:

$$p(\boldsymbol{y}_0, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_T) = p(\boldsymbol{y}_0) \prod_{t=2}^{T} p(\boldsymbol{y}_t \mid \boldsymbol{y}_{t-1})$$

where the transition probability is a Neural Network (e.g Transformer based architetcures)

$$p(\boldsymbol{y}_t \mid \boldsymbol{y}_{t-1}, \boldsymbol{y}_{t-2}) = \mathrm{NN}(\boldsymbol{y}_{t-1}, \boldsymbol{y}_{t-2}; \boldsymbol{\omega})$$



Bodnar, Cristian, et al. "A foundation model for the Earth system." Nature, 2025

PDE-REFINER

Refinement Process

The multistep refinement process allows more accurate modelling of all frequency components

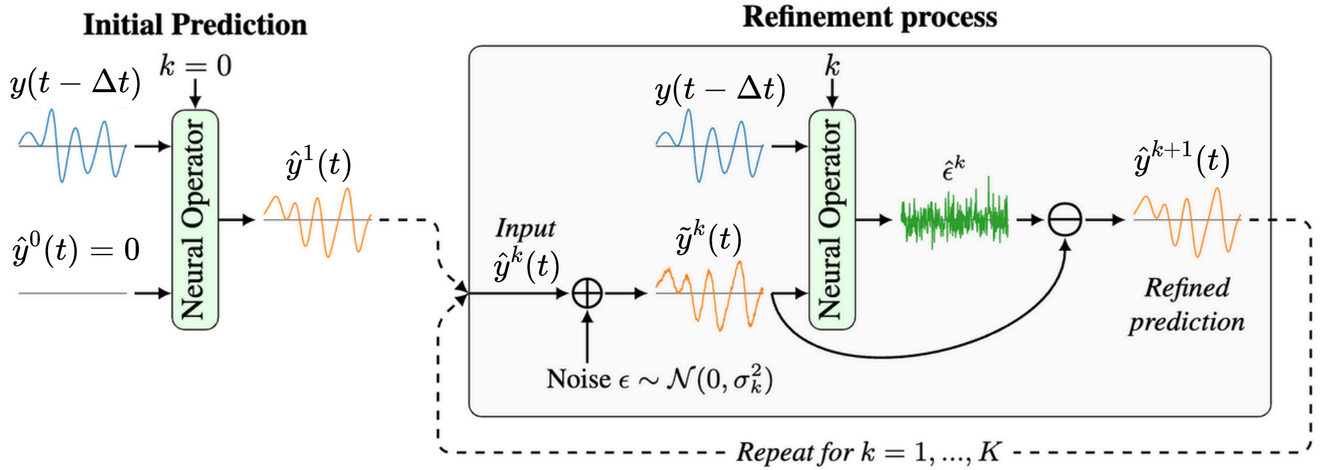
Denoising Loss Function

$$\mathcal{L}(\theta) = \mathbb{E}_{k\sim U(0,K)}[\mathcal{L}_k(\theta)]$$

with,

$$\mathcal{L}_0(\theta) = \|y_t - \mathrm{NN}(\hat{y}^0(t), y(t-\Delta t), 0)\|^2 \qquad \text{(one-step loss)}$$

$$\mathcal{L}_k(\theta) = \|\epsilon_k - \mathrm{NN}(y(t) + \sigma_k\epsilon, y(t-\Delta t), k)\|^2, k = 1, \ldots, K \qquad \text{(refinement loss)}$$



Lippe, Phillip, et al. "PDE-Refiner: Achieving Accurate Long Rollouts with Neural PDE Solvers", 2023.

FUNCTIONAL GENERATIVE NETWORKS

Ensemble Uncertainty

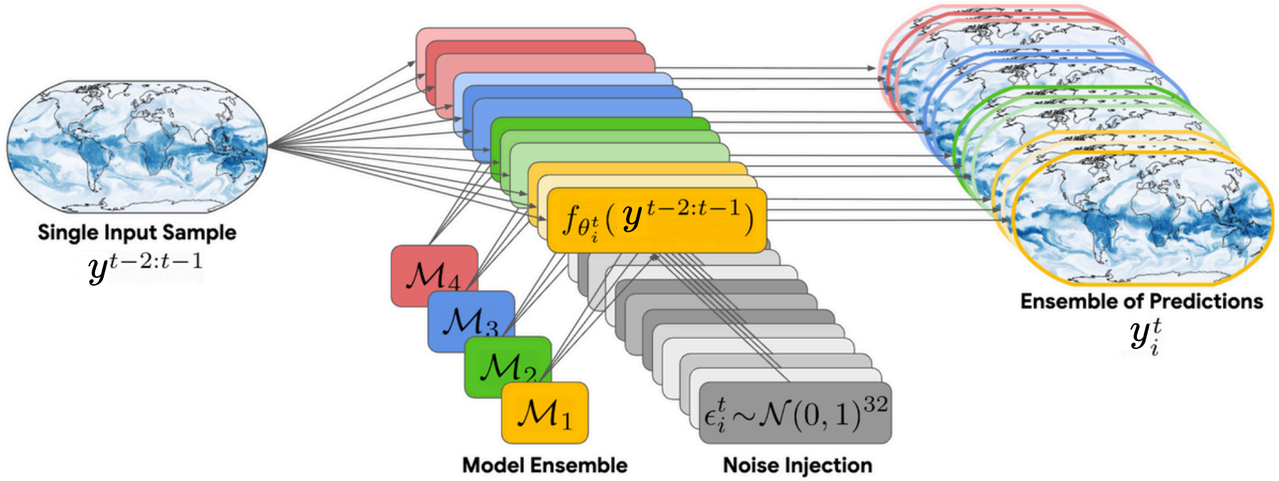
$$p(y^t \mid y^{t-2:t-1}) = \sum_i p(y^t \mid y^{t-2:t-1}, \omega)\delta(\omega - \theta_i)$$

Perturbation Uncertainty (only LayerNorm)

$$\theta_i^t = \theta_i + \Delta_i \cdot \epsilon_i^t, \quad \epsilon_i^t \sim \mathcal{N}(0,1)$$

Trained to minimize CRPS

$$\mathcal{L}(\theta_i, \Delta_i) = \frac{1}{N}\sum_n |y^t_{n;(\theta_i,\Delta_i)} - y^t| - \frac{1}{2N(N-1)}\sum_{n,n'} |y^t_{n;(\theta_i,\Delta_i)} - y^t_{n';(\theta_i,\Delta_i)}|$$



Alet, Ferran, et al. "Skillful joint probabilistic weather forecasting from marginals", 2025.

BARNN

Joint Weight-States update

Jointly sample in an alternating way weights and states

$$p(\boldsymbol{y}_0, \boldsymbol{\omega}_1, \boldsymbol{y}_1, \boldsymbol{\omega}_2, \ldots, \boldsymbol{y}_T) = \prod_{t=1}^{T} p(\boldsymbol{y}_t \mid \boldsymbol{y}_{<t}, \boldsymbol{\omega}_t) p(\boldsymbol{\omega}_t)$$

Variational Inference

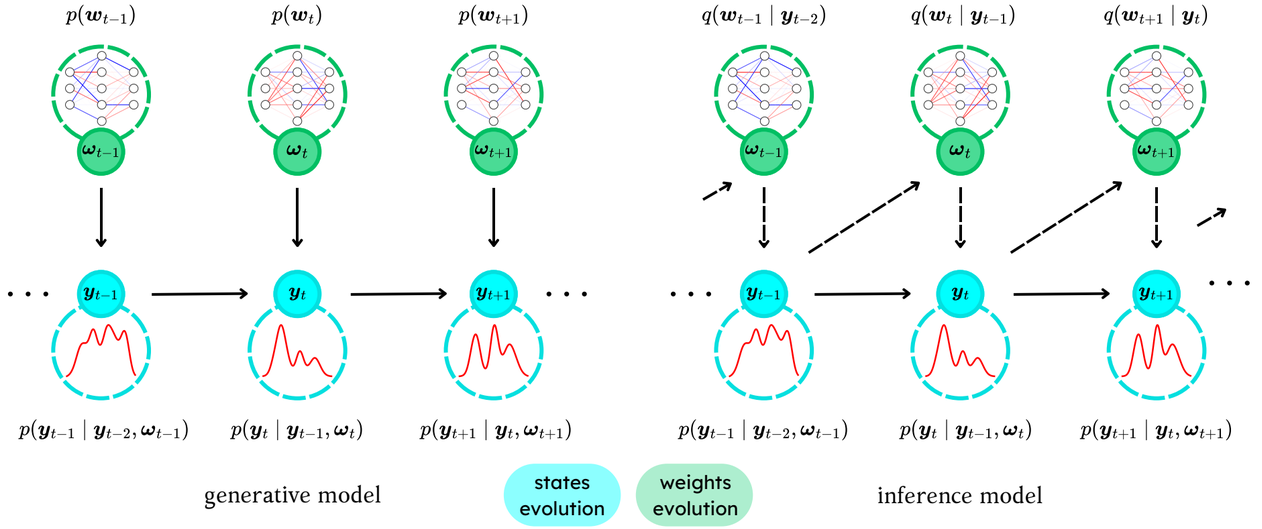
Use variational inference to optimise the neural network

$$q_{\boldsymbol{\phi}}(\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_T \mid \boldsymbol{y}_{<T}) = \prod_{t=1}^{T} q(\boldsymbol{\omega}_t \mid \boldsymbol{y}_{<t}) \quad \text{and,} \quad p(\boldsymbol{\omega}_t) = \int q(\boldsymbol{\omega}_t \mid \boldsymbol{y}_{<t}) p(\boldsymbol{y}_{<t}) d\boldsymbol{y}_{<t}$$

$$\mathcal{L}(\boldsymbol{\phi}) = \mathbb{E}_{t \sim \mathcal{U}[1,T]} \left[\mathbb{E}_{\boldsymbol{\omega}_t \sim q_{\boldsymbol{\phi}}} \left[\log p(\boldsymbol{y}_t \mid \boldsymbol{y}_{0<t}, \boldsymbol{\omega}_t) \right] - D_{KL} \left[q_{\boldsymbol{\phi}}(\boldsymbol{\omega}_t \mid \boldsymbol{y}_{<t}) \| p(\boldsymbol{\omega}_t) \right] \right]$$

Scale to Large Networks

Local-reparametrization allos model's scale to large Neural Network $q(\boldsymbol{\omega}_t^l \mid \boldsymbol{y}_{<t}) = \mathcal{N}(\alpha_t^l \boldsymbol{\Omega}^{\boldsymbol{l}}, (\alpha_t^l \boldsymbol{\Omega}^{\boldsymbol{l}})^2) \quad \alpha_t^l = f(\boldsymbol{y}_{<t}; \boldsymbol{\delta})$

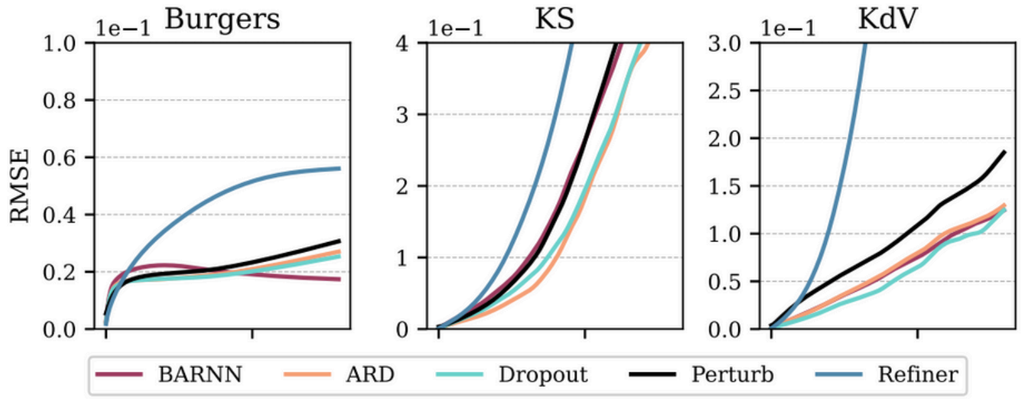


Coscia, Dario, et al. "BARNN: A Bayesian Autoregressive and Recurrent Neural Network", 2025.

UNCERTAINTY ON PDE MODELLING

Most models can **accurately solve PDEs** with **low point-wise RMSE** but **struggles in UQ scenarios** where lower NLL and ECE is expected to have reliable uncertainties

$$
\begin{aligned}
\textit{Burgers} \quad & \partial_t u + u\partial_x u - \nu\partial_{xx} u = 0 \\
\textit{Kuramoto Sivashinsky} \quad & \partial_t u + u\partial_x u + \partial_{xx} u + \partial_{xxxx} u = 0 \\
\textit{Korteweg de Vries} \quad & \partial_t u + u\partial_x u + \partial_{xxx} u = 0
\end{aligned}
$$



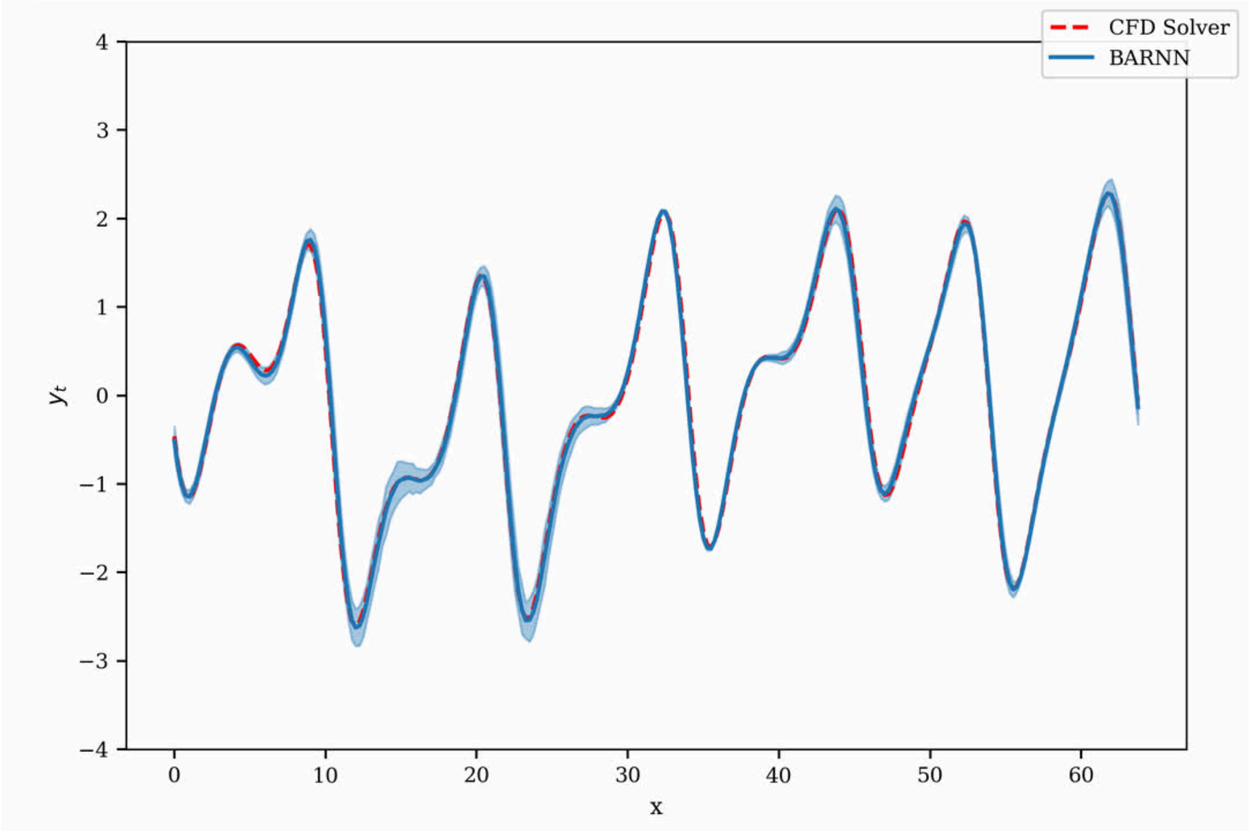
Model	NLL (↓)			ECE (↓)		
	Burgers	*KS*	*KdV*	*Burgers*	*KS*	*KdV*
BARNN	$\mathbf{-2.51}^{\pm 0.29}$	$\mathbf{-0.87}^{\pm 0.52}$	$\mathbf{-0.94}^{\pm 1.23}$	$\mathbf{0.05}^{\pm 0.03}$	$\mathbf{0.04}^{\pm 0.03}$	$\mathbf{0.10}^{\pm 0.01}$
ARD	$5.00^{\pm 7.11}$	$10.7^{\pm 11.6}$	break	$0.26^{\pm 0.10}$	$0.17^{\pm 0.01}$	$0.19^{\pm 0.05}$
Dropout	$0.06^{\pm 4.59}$	$-0.48^{\pm 0.65}$	$0.40^{\pm 4.57}$	$0.17^{\pm 0.15}$	$0.10^{\pm 0.01}$	$0.14^{\pm 0.07}$
Perturb	$-2.01^{\pm 0.08}$	$-0.65^{\pm 1.07}$	$-0.37^{\pm 1.32}$	$0.16^{\pm 0.01}$	$0.05^{\pm 0.02}$	$0.11^{\pm 0.06}$
Refiner	break	break	break	$0.30^{\pm 0.08}$	$0.23^{\pm 0.02}$	$0.34^{\pm 0.09}$

UNCERTAINTY ON PDE MODELLING

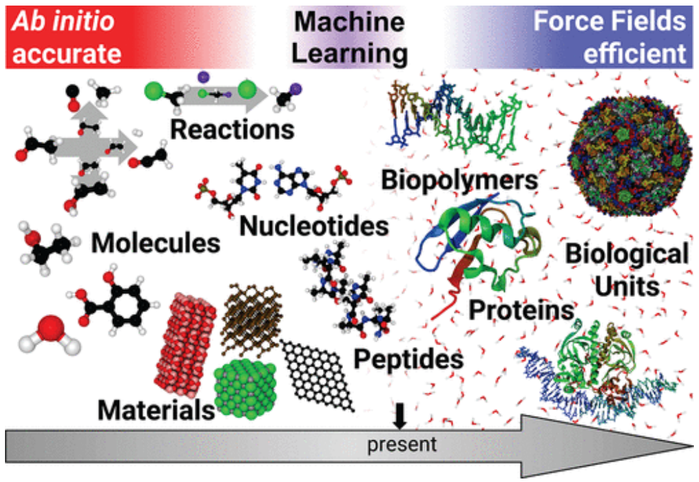
Confidence intervals are tighter in regions of small uncertainties and larger otherwise

(using BARNN model)

Kuramoto Sivashinsky $\partial_t u + u\partial_x u + \partial_{xx} u + \partial_{xxxx} u = 0$



UNCERTAINTY QUANTIFICATION IN MATERIAL SCIENCE



Unke, Oliver T., et al. "Machine learning force fields." Chemical Reviews (2021)

FAIR Chemistry

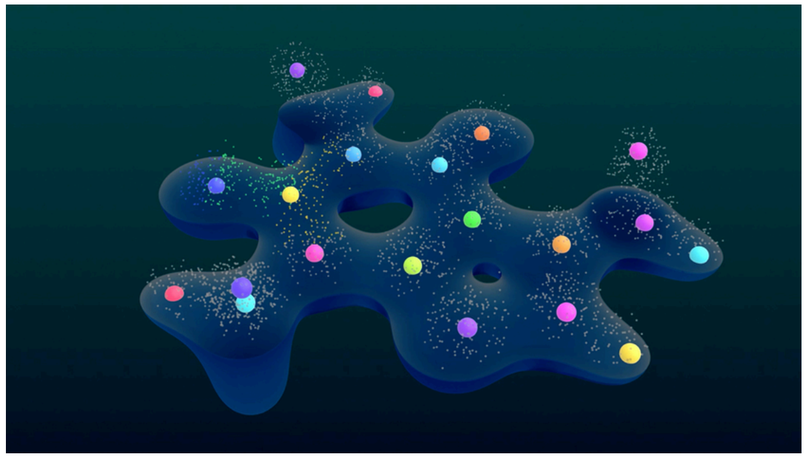
Wood, Brandon M., et al. "UMA: A Family of Universal Models for Atoms." arXiv preprint arXiv:2506.23971 (2025).

Orbital Materials

Rhodes, Benjamin, et al. "Orb-v3: atomistic simulation at scale." arXiv preprint arXiv:2504.06231 (2025).

A SMALL INTRO TO MATERIAL SCIENCE

Atoms form the basic building blocks of molecules and materials, held together by their electrons. *This "electron glue" determines stability and key properties.* ***Accurately modeling it helps predict chemical reactions, drug effectiveness, material performance for carbon capture, and improvements in renewable energy storage***



Luise, Giulia, et al. "Accurate and scalable exchange-correlation with deep learning." arXiv preprint arXiv:2506.14665 (2025).

Kohn-Sham formalism is used to compute energy of the system

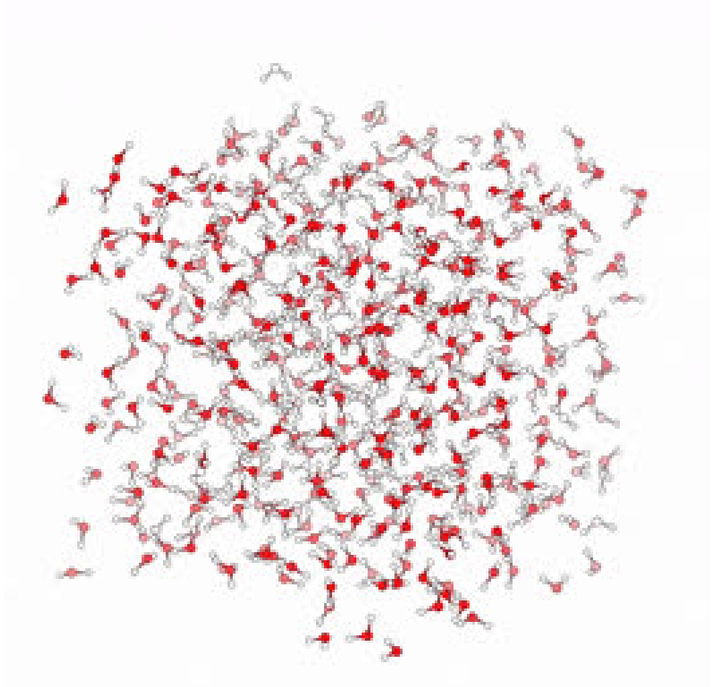
$$E = \min_{\rho} E_{\text{tot}}[\rho], \quad E_{\text{tot}}[\rho] = \int v(r)\rho(r)\,dr + \frac{1}{2}\int\int \frac{\rho(r)\rho(r')}{|r-r'|}drdr' + T_s[\rho] + E_{\text{xc}}[\rho]$$

where,

ρ	*electron density*	E_{xc}	*exchange correlation energy*
E_{tot}	*electron density*	T_s	*kinetic energy of the system*
$v(r)$	*nuclei external potential*		

A SMALL INTRO TO MATERIAL SCIENCE

Molecular dynamics (MD) *is a simulation method that applies* ***Newton's laws of motion*** *using* ***quantum computed Forces*** *to track how atoms and molecules move over time. It's like running a "virtual experiment," where you observe how a system of interacting particles evolves to explore their behavior and properties.*



A simple MD simulation of water (H2O)

Forces are computed by taking the negative gradient of the Energy (usually computed via DFT)

$$\mathbf{F} = -\nabla_{\mathbf{r}} E_{\text{tot}}$$

Then using Newton's law of motion we study the material over time

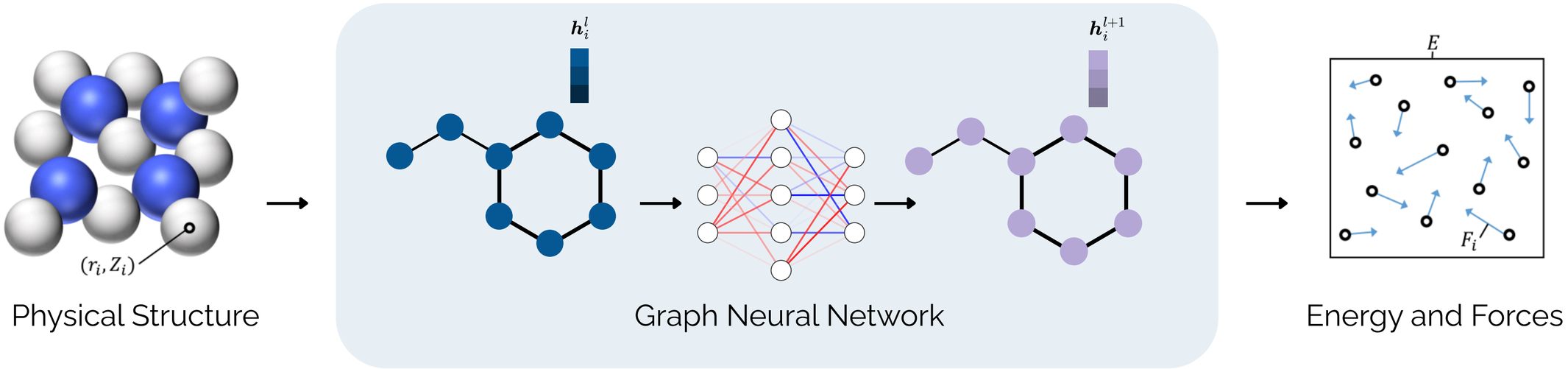
$$\frac{d\mathbf{r}}{dt} = \mathbf{v}, \quad \frac{d\mathbf{v}}{dt} = \frac{\mathbf{F}}{m}$$

MACHINE LEARNING FORCE FIELDS

Machine Learning Force Fields use ***data-driven models*** to predict ***structure energy and forces*** with high accuracy and efficiency. There are two types of Models:

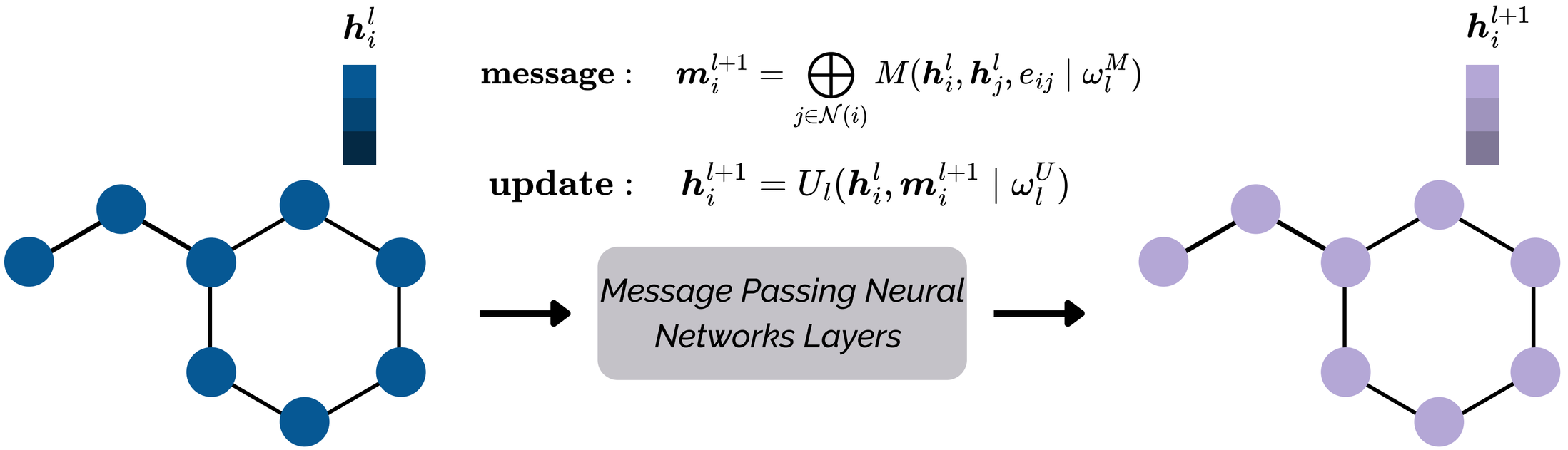
Conservative Models: Network predicts Energy. The Forces are obtain by taking the negative gradient with respect to positions using automatic differentiation. Example: ***MACE, UMA, ...***

Direct Models: Network predicts directly Energy and Forces. Example: ***ORB, ...***



TRAINING MESSAGE PASSING NETWORKS

Message Passing Neural Networks *(MPNNs) are trained to learn interactions by iteratively exchanging information (messages) between nodes in a graph. They capture complex local environments and many-body effects with high flexibility.*



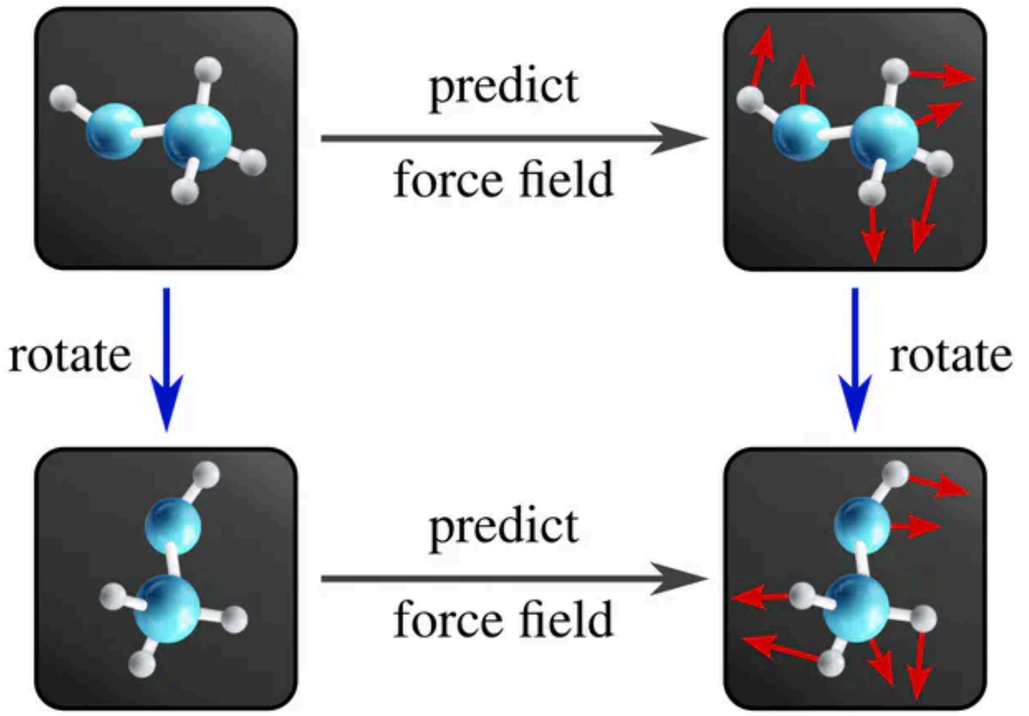
IS IT ALL?

Machine Learning Force Fields are usually built to satisfied SO(3) equivariance/invariance

Example

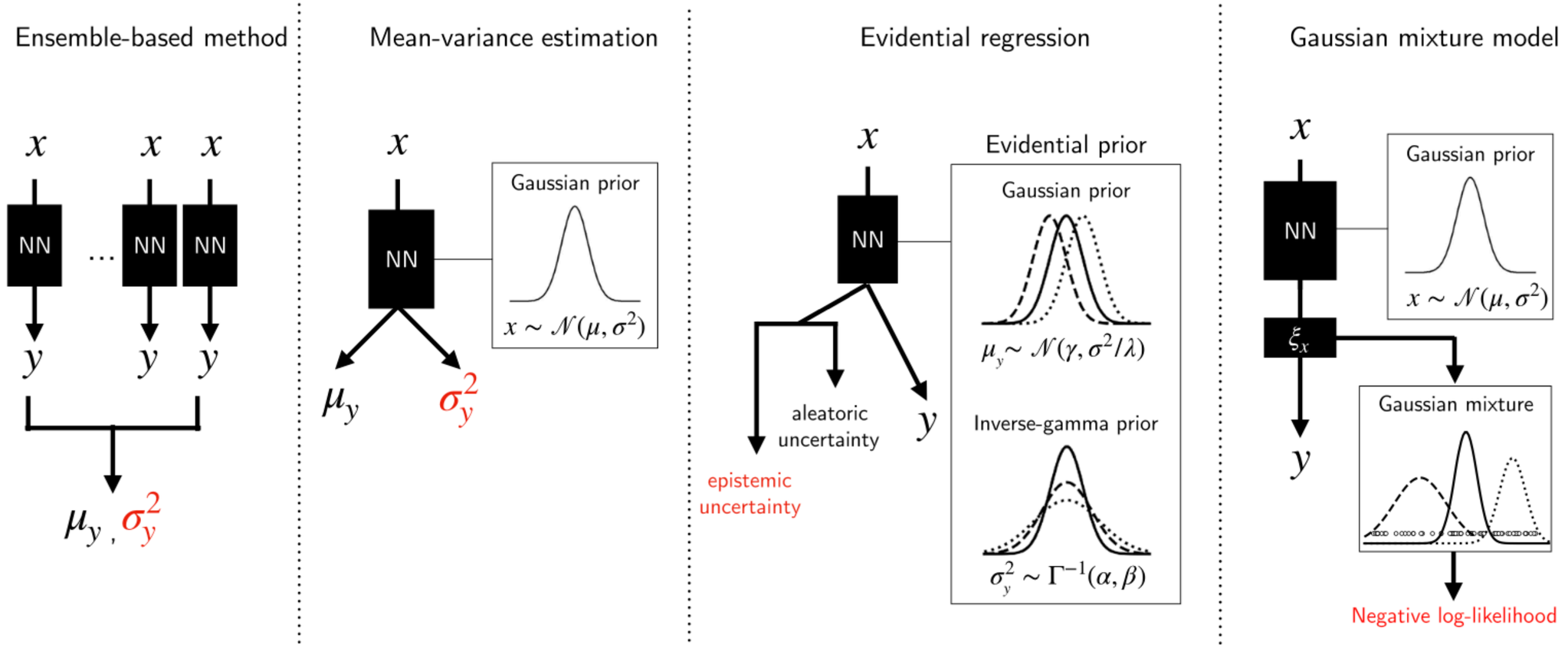
If a molecule is rotated, its energy stays the same (invariant), and its forces rotate accordingly (equivariant)

If we build stochastic networks (sampling weights) we must ensure equivariance/invariance in distribution!



Weiler, Maurice. "Equivariant and coordinate independent convolutional networks: A gauge field theory of neural networks", 2024.

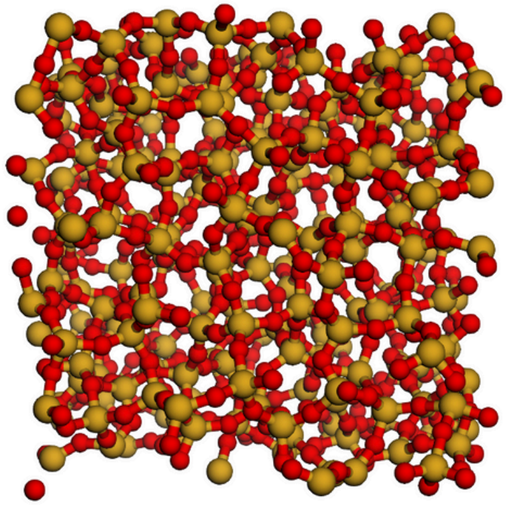
MODELLING UNCERTAINTY IN MLFFS

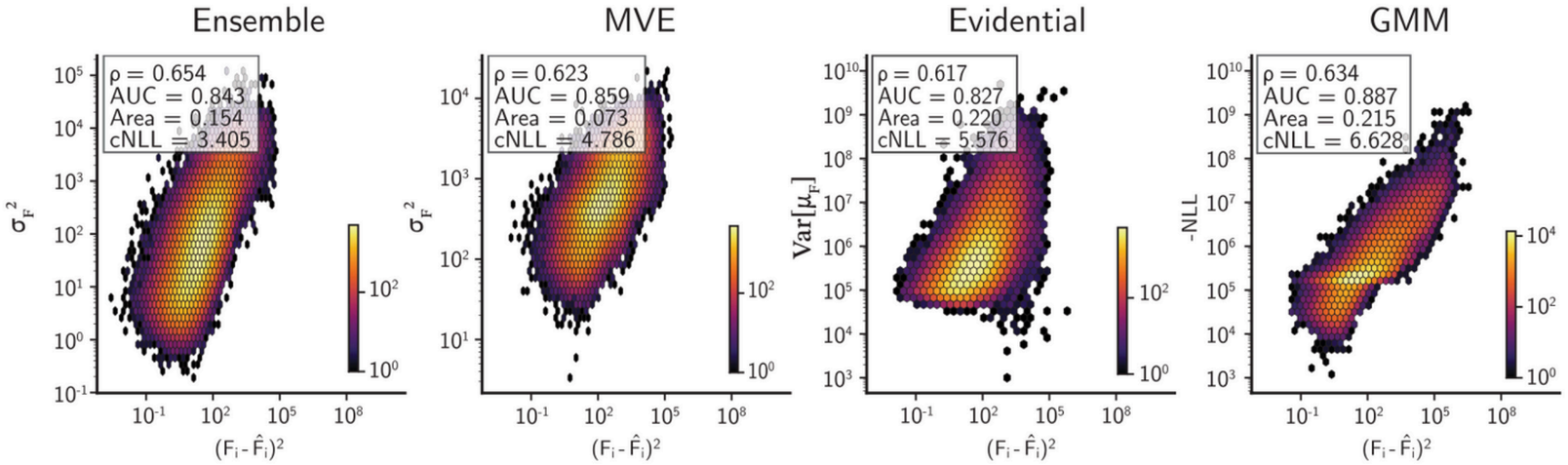


Tan, Aik Rui, et al. "Single-model uncertainty quantification in neural network potentials does not consistently outperform model ensembles." npj Computational Materials 9.1 (2023): 225.

MODELLING UNCERTAINTY IN MLFFS - SILICA

Silica glass is made up of large bulk structures (many atoms) and the computational cost of performing quantum chemistry calculations on these bulk systems is prohibitively high! ***MLFFs can help mitigating this problem, but reliable uncertainties are needed!***

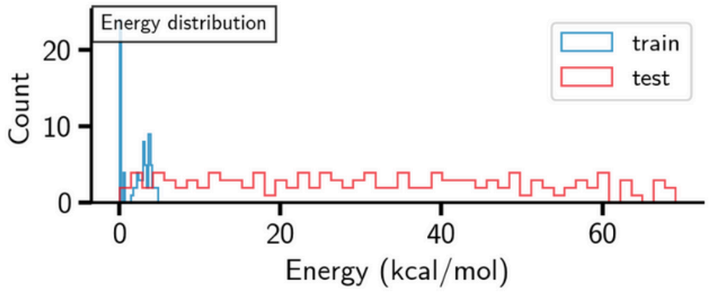


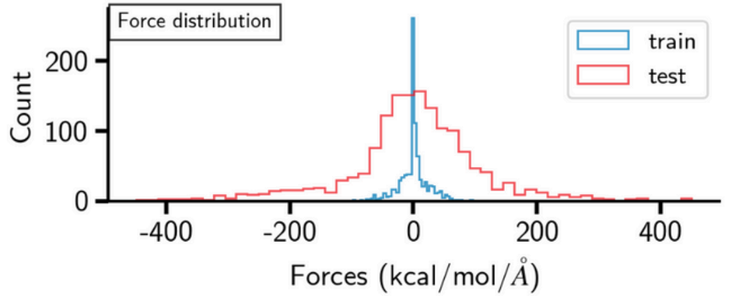


MODELLING UNCERTAINTY IN MLFFS - AMMONIA

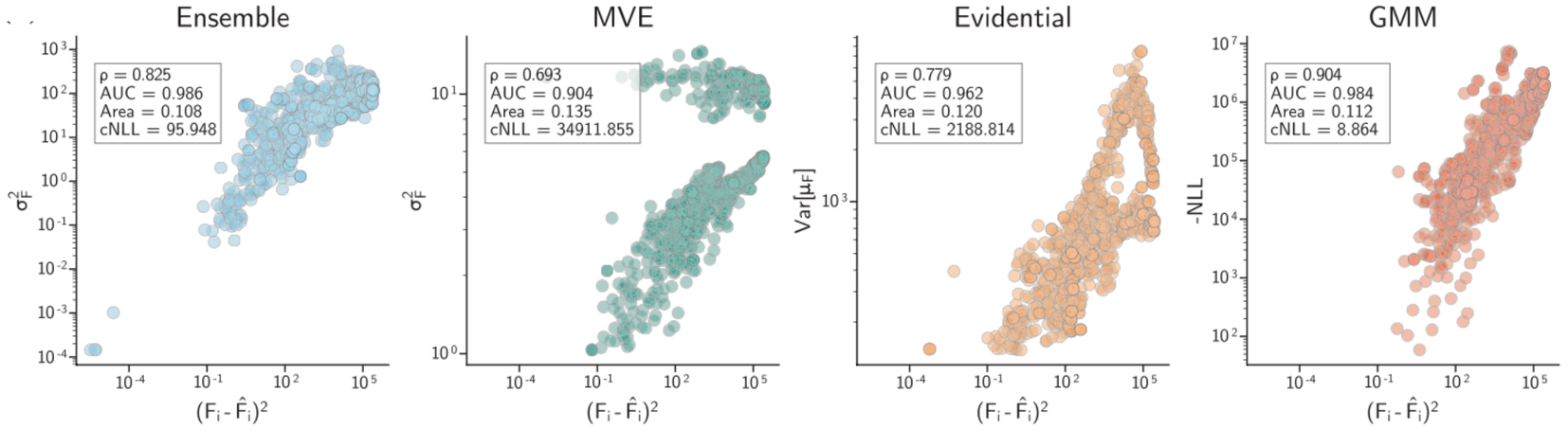
Ammonia NH3 is a simple molecule with a very interesting property: ***the nitrogen atom flips from one side of the plane formed by its substituents to the other***, like an umbrella turning inside out

Testing on heavy OOD data



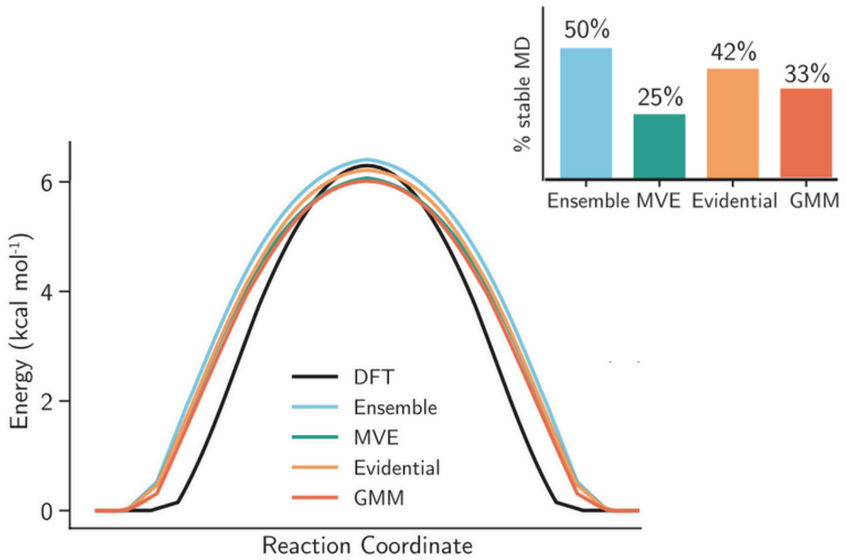


Both Deep Ensemble and GMM perform well!

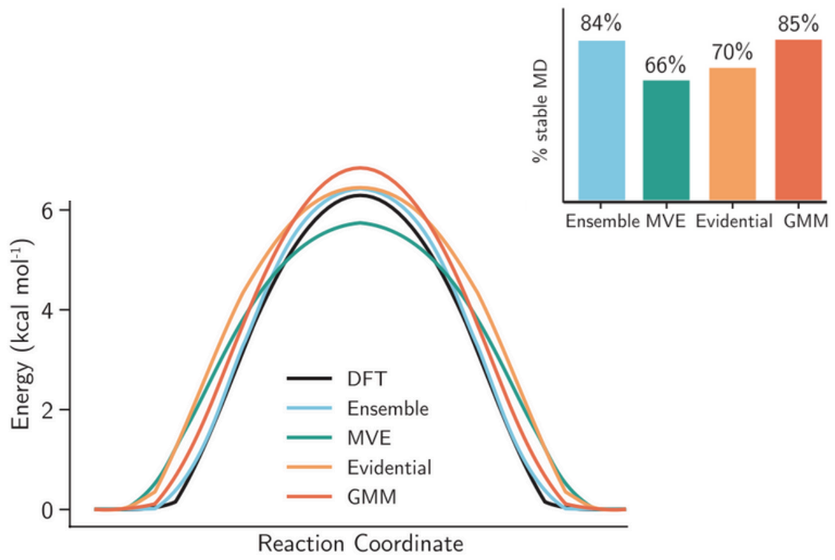


MODELLING UNCERTAINTY IN MLFFS - AMMONIA

Ammonia NH3 is a simple molecule with a very interesting property: ***the nitrogen atom flips from one side of the plane formed by its substituents to the other***, like an umbrella turning inside out



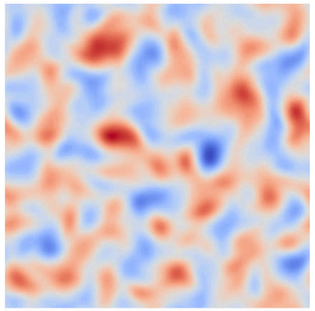
active learning

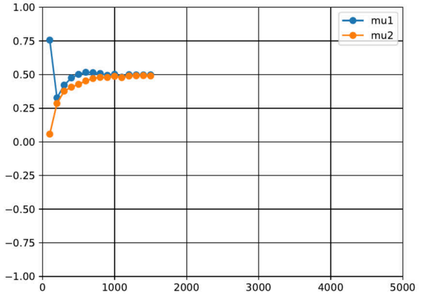


Ang, S. J., Wang, W., Schwalbe-Koda, D., Axelrod, S. & Gómez-Bombarelli, R. Active learning accelerates ab initio molecular dynamics on reactive energy surfaces. Chem 7, 1–32 (2021).

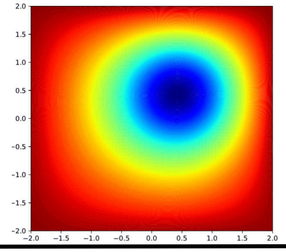
CODING SOFTWARE FOR SCIML

Fast Fluid Dynamics Simulations

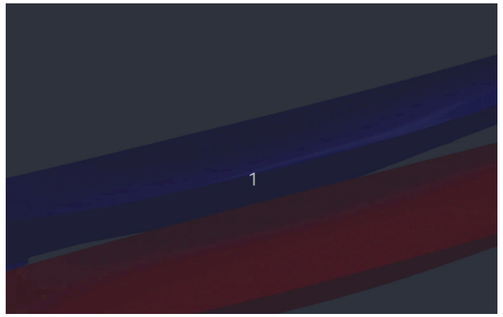


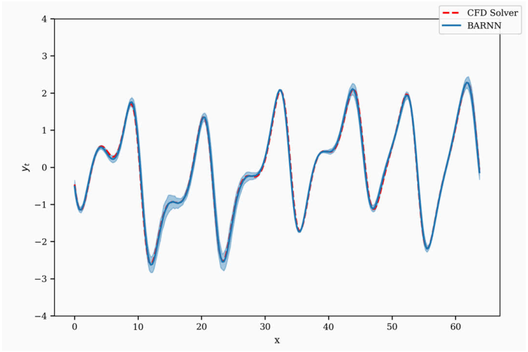


Inverse Problems

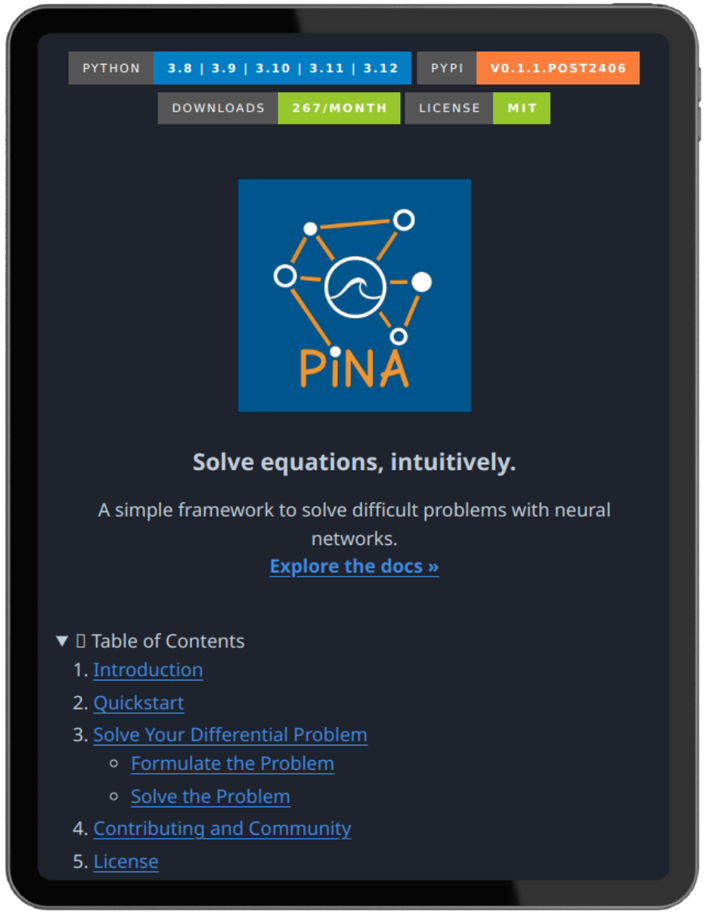


Physics Informed Deformation Learning





Uncertainty Quantification with BARNN



Coscia, Dario, et al. "Physics-informed neural networks for advanced modeling." Journal of Open Source Software 8.87 (2023): 5352.