Model Error Estimation and Uncertainty Quantification of Machine Learning Interatomic Potentials

Khachik Sargsyan,
Logan Williams, Habib N. Najm

Sandia National Laboratories, Livermore, CA

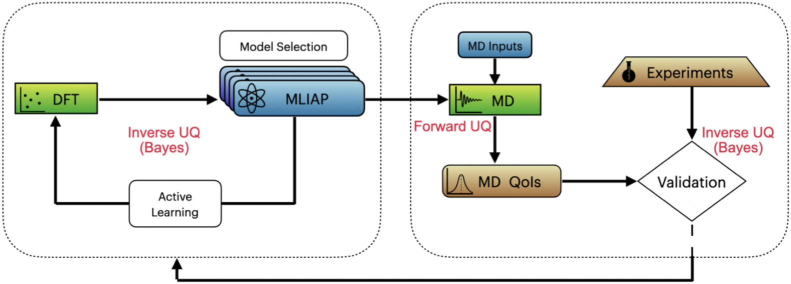
Error control in first-principles modelling
CECAM-EPFL, Lausanne, Switzerland
June 20-24, 2022

Acknowledgements

- Aidan Thompson, Mary Alice Cusentino, Mitchell Wood, Ember Sikorski (SNL), Katherine Johnston (SNL, U Washington)
- DOE, Office of Science,
  - Fusion Energy Sciences (FES)
  - Advanced Scientific Computing Research (ASCR)

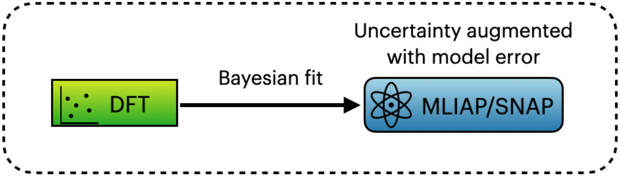
Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.

Outline



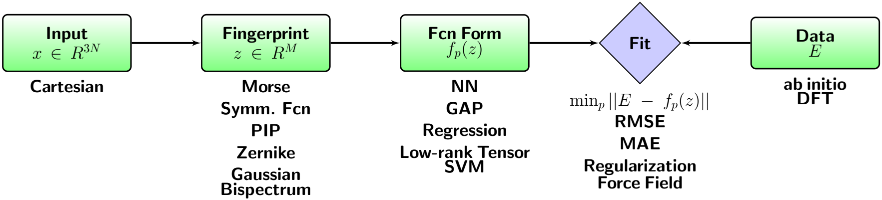
Model Selection
DFT
MLIAP
Inverse UQ
(Bayes)
Active
Learning
MD Inputs
MD
Forward UQ
MD QoIs
Experiments
Inverse UQ
(Bayes)
Validation

Outline

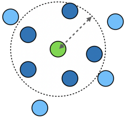


- UQ for machine learning interatomic potentials (MLIAP)
    - ... for uncertainty propagation
    - ... for active learning
    - ... for model selection
- Bayesian approach
    - More focus on linear regression models:
      Spectral Neighbor Analysis Potential (SNAP)
    - Importance of noise model, embedded model error construction
    - Relation to variational inference

ML Interatomic Potentials (MLIAP): supervised ML



- Partition the interatomic interaction energy into individual contributions of the atoms $E_{\text{total}} = \sum_{i=1}^{N} E_i$
- Assume flexible functional forms of each such contribution
  - Function of positions of the neighboring atoms
  - $O(100)$ parameters



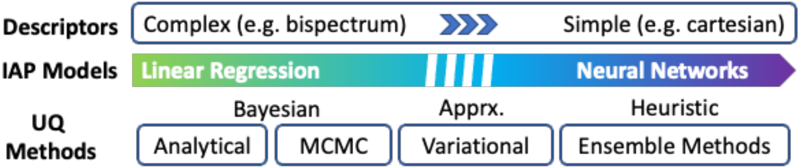
- Require the energy, forces and/or stresses predicted by a MLIAP to be close to those obtained by a quantum mechanical model on some atomic configurations (a.k.a. training set)

MLIAP - desired features

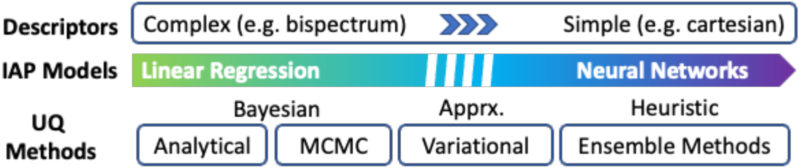
- Good input descriptors
- Accurate, fast-to-evaluate, analytic derivatives
- High-dimensional, flexible functional form
- Transferable/generalizable to unseen atomic configurations
- Account for physics:
  - invariant with respect to translation,rotation, and reflection of the space, and also permutation of chemically equivalent atoms
- Locality (depend on surrounding atoms only within a finite cut-off radius), but remain smooth with respect to atoms entering and leaving the local neighborhood
- **Equipped with uncertainty estimate**
  - for active learning, for MD propagation, ...

Enabling parametric fits with uncertainties

$$y \approx f_c(x)$$



Focus on SNAP (Left end of the figure)



- [Thompson et al., 2015] "Spectral neighbor analysis method for automated generation of quantum-accurate interatomic potentials", *J Comp Phys*, 2015.

$$f(q) = \sum_{k=0}^{K} c_k B_k(q)$$

- Linear expansion in parameters c.
- Bayesian inference: both MCMC and analytical posterior PDFs are feasible

(Bayesian) Parameter Inference

- Given a model $f(x,c)$ and data $y_i = y(x_i)$, calibrate parameters c.
  - Linear model $f(x,c) = Bc$ with coefficients c
  - NN model $f(x,c) = NN_c(x)$ with weights/biases c

- Bayesian least-squares fit:
  $p(c|y) \propto p(y|c)p(c) \propto \prod_{i=1}^N \exp\left(-\frac{(f(x_i,c)-y_i)^2}{2\sigma^2}\right)$

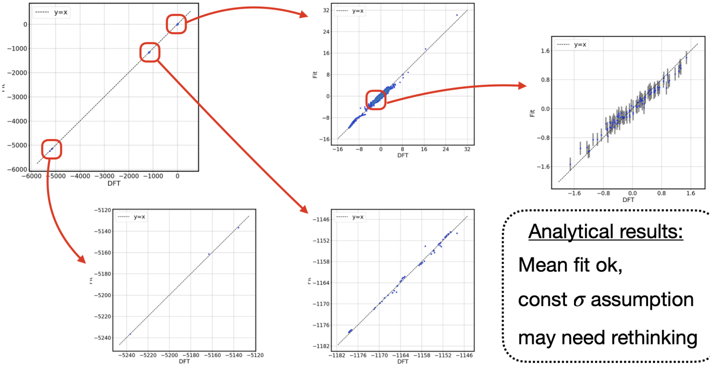
- ... corresponding data model $y_i = f(x_i,c) + \sigma \underbrace{\epsilon_i}_{\mathcal{N}(0,1)}$

- Exact answer for linear models: $c \sim \mathcal{N}\left((B^TB)^{-1}B^Ty, \sigma^2(B^TB)^{-1}\right)$

SNAP uncertainty with Tantalum data set

$$f(q) = \sum_{k=0}^{K} c_k B_k(q)$$

- Employed FitSNAP *https://github.com/FitSNAP/FitSNAP*



Analytical results:

Mean fit ok,

const σ assumption

may need rethinking

- Assumptions baked in likelihood form are crucial!
- i.i.d. gaussian noise with constant σ is not well founded.

Elephant in the room: model is assumed to be *the* correct model behind data

$$\underbrace{y_i = \overset{\text{Model}}{\underset{\text{Truth}}{f(x_i, c)}}}_{} + \overset{\text{Data err.}}{\sigma_i \epsilon_i}$$

Model $\neq$ Truth

- One gets biased estimates of parameters c (crucial if the model is physical, and/or c is propagated through other models)
- More data leads to overconfident predictions (we become more and more certain about the wrong values of the data)
- More evident when there is no (observational/experimental) data error: e.g. DFT is data, and MLIAP is model

Posterior pushed-forward uncertainty does not capture true discrepancy

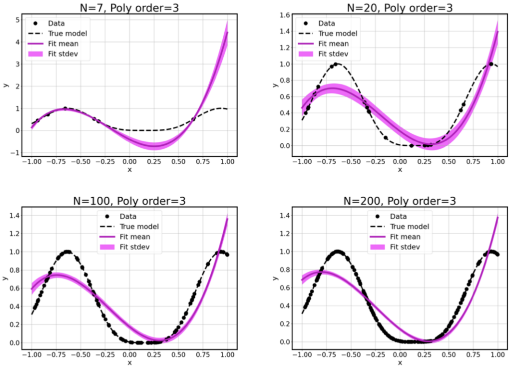
Synthetic data

$$y(x) = \sin^4(2x - 0.3)$$

Cubic fit

$$y_i \approx \sum_{k=0}^{3} c_k B_k(x)$$

More data leads to overconfident prediction



Capturing model error in data model (a.k.a. likelihood)

External correction (Kennedy-O'Hagan):

$$y_i = f(x_i, c) + \delta(x_i) + \sigma_i \epsilon_i$$

- Kennedy, O'Hagan, "Bayesian Calibration of Computer Models". *J Royal Stat Soc: Series B (Stat Meth)*, 63: 425-464, 2001.

Internal correction (embedded model error):

$$y_i = f(x_i, c + \delta(x_i)) + \sigma_i \epsilon_i$$

- Allows meaningful usage of calibrated model
- 'Leftover' noise term even with no data error
- Respects physics (not too relevant in our context)

- Sargsyan, Najm, Ghanem, "On the Statistical Calibration of Physical Models". *Int. J. Chem. Kinet.*, 47: 246-276, 2015.
- Sargsyan, Huan, Najm, "Embedded Model Error Representation for Bayesian Model Calibration". *Int. J. Uncert. Quantif.*, 9(4): 365-394, 2019.

- Typically requires uncertainty propagation in the likelihood computation
- For linear regression, we can take some shortcuts (see next)

Embedded Model Error for Linear Regression Models

Conventional (i.i.d. error term):

$$y_i \approx \sum_{k=0}^{P} c_k B_k(x_i) + \sigma_i \epsilon_i$$

Embed uncertainty in all or selected coefficients:

$$y_i \approx \sum_{k=0}^{P} (c_k + d_k \xi_k) B_k(x_i) = \overbrace{\sum_{k=0}^{P} c_k B_k(x_i)}^{\text{Model}} + \overbrace{\sum_{k=0}^{P} d_k B_k(x_i) \xi_k}^{\text{Model Error}}$$

Note:
No formal distinction between internal and external corrections:
but the error is now model-informed.

Joint MCMC inference of model parameters and model-error parameters

Conventional:

$$y_i \approx \sum_{k=0}^{P} c_k B_k(x_i) + \sigma_i \epsilon_i \qquad p(c|y) \propto \prod_{i=1}^{N} \exp\left(-\frac{(\sum_{k=0}^{P} c_k B_k(x_i) - y_i)^2}{2\sigma_i^2}\right)$$

Embedded:

$$y_i \approx \sum_{k=0}^{P}(c_k + d_k\xi_k)B_k(x_i) = \overbrace{\sum_{k=0}^{P} c_k B_k(x_i)}^{\text{Model}} + \overbrace{\sum_{k=0}^{P} d_k B_k(x_i)\xi_k}^{\text{Model Error}}$$

$$p(c,d|y) \propto \underbrace{p(y|c,d)}_{\text{Likelihood}}\ \underbrace{p(c,d)}_{\text{Prior}}$$

Note:

Both likelihood and prior selection are challenging.

Embedded Model Error: Two Approximate Likelihood Options

$$y_i \approx \sum_{k=0}^{P}(c_k + d_k\xi_k)\, B_k(x_i) \quad = \quad \sum_{k=0}^{P} c_k B_k(x_i) + \sum_{k=0}^{P} d_k B_k(x_i)\xi_k$$

Option 1: IID

$$p(c,d|y) \propto \prod_{i=1}^{N} \exp\left(-\frac{(\sum_{k=0}^{P} c_k B_k(x_i) - y_i)^2}{2\sum_{k=0}^{K} d_k^2 B_k(x_i)^2}\right)$$

Option 2: ABC

$$p(c,d|y) \propto \prod_{i=1}^{N} \exp\left(-\frac{(\sum_{k=0}^{P} c_k B_k(x_i) - y_i)^2 + (\sqrt{\sum_{k=0}^{P} d_k^2 B_k^2(x_i)} - \alpha|\sum_{k=0}^{P} c_k B_k(x_i) - y_i|)^2}{2\epsilon^2}\right)$$

Note:

Does not have to be MCMC: simply optimize the posterior for (c,d)

Pushed forward predictive uncertainty captures the true discrepancy from the data

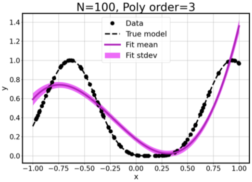
Synthetic data

$$y(x) = \sin^4(2x - 0.3)$$

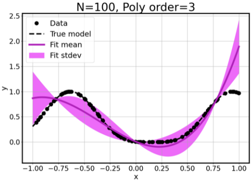
Cubic fit

$$y_i \approx \sum_{k=0}^{3} c_k B_k(x)$$

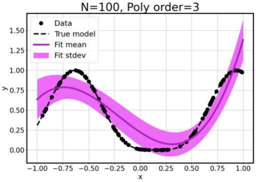
Classical case



Model error, IID likelihood

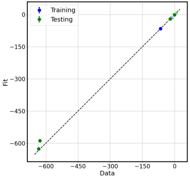


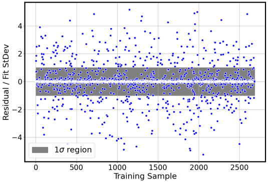
Model error, ABC likelihood

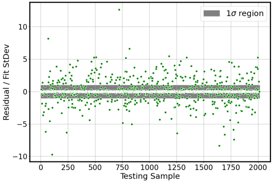


Uncertainty validation: W-ZrC Dataset

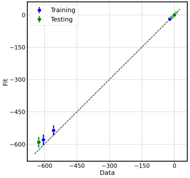
Uncertainty without model error

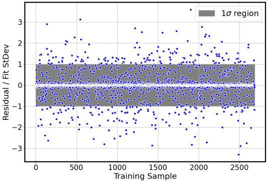


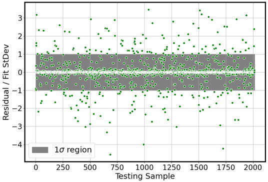




Uncertainty with model error





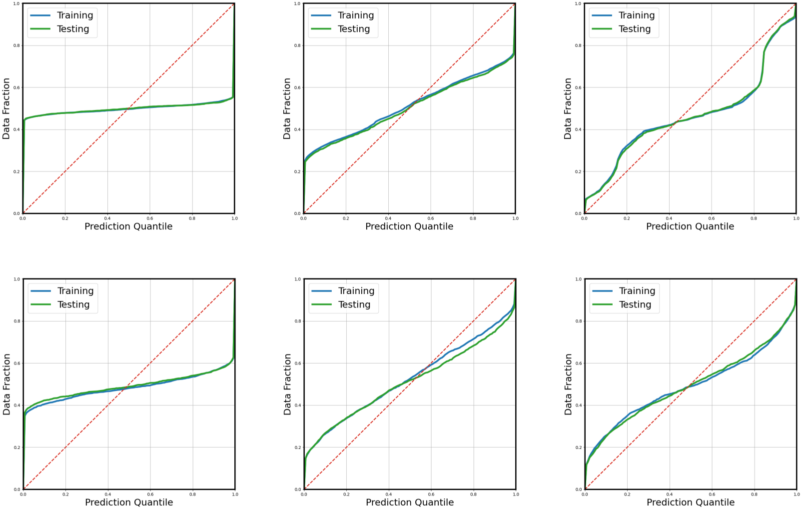


Uncertainty validation: two examples

Conventional | Embedded, IID Lik. | Embedded, ABC Lik.

Ta

W-ZrC



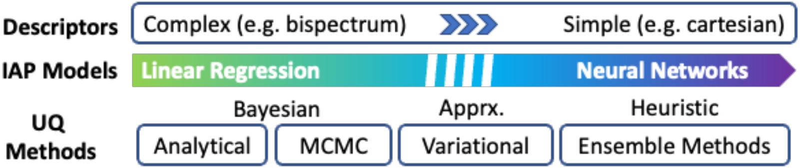
Model Error Wrapup: several challenges and choices

- Embedding type, e.g. additive/multiplicative

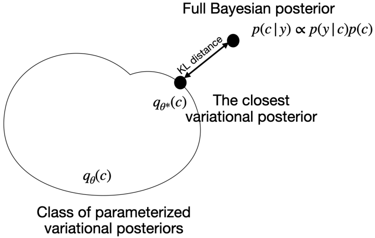
$$y_i \approx \sum_{k=0}^{P} (c_k + d_k \xi_k) B_k(x_i) \qquad \text{or} \qquad y_i \approx \sum_{k=0}^{P} (c_k + c_k d_k \xi_k) B_k(x_i)$$

- Degenerate (Gaussian) likelihoods: resort to approximate Bayesian computation (ABC) or independent (IID) assumptions
- Difficult posterior PDFs for MCMC, choice of priors for embedding parameters
- Which coefficients to embed the model error in?
- Connect predictive uncertainty and the residual error with an extrapolation metric
- Weighting between energies, forces and stresses

Variational inference is a compromise between Bayesian and Empirical approaches



Variational inference in a nutshell



$$KL(p_1||p_2) = \int \ln\left(\frac{p_1(x)}{p_2(x)}\right) p_1(x)dx$$

- e.g. Mean-Field Variational Inference (MFVI): ansatz $c \sim \mathcal{N}(\mu, \text{diag}(v))$ and find best (μ, v), i.e.
- minimize Kullback-Leibler distance to the full Bayesian posterior, $\text{argmin}_{(\mu,v)}\text{KL}(\mathcal{N}(\mu, \text{diag}(v))||\mathcal{N}(\mu_0, \Sigma))$,
- replaces sampling (MCMC) problem with an optimization problem.

Note the connection between variational inference and embedded model error

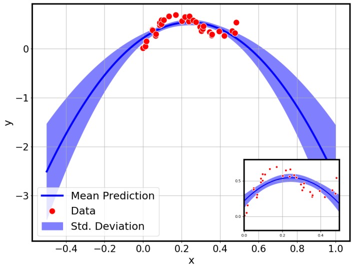
- Variational methods: $c \sim N(\mu, \Sigma)$ and optimize μ, Σ.
  - In NN context, this is largely called Bayesian Neural Networks
  - Minimize Kullback-Leibler distance via Stoch. Gradient Descent
- Embedded model error: $c \sim N(\mu, \Sigma)$ and optimize μ, Σ.
  - Minimize Gaussian approximation of output predictions (IID), or
  - Minimize statistics/moment matching criterion (ABC)

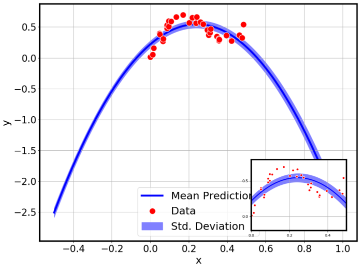
Next:
Overparameterized linear regression (mimicking NN) challenges mean-field variational inference outside training support.

Polynomial fit: Extrapolation scenario

Order=2

Full Bayesian Posterior	Variational Posterior
$\mathcal{N}(\mu, \Sigma)$	$\mathcal{N}(\mu, 1/\text{diag}(\Sigma^{-1}))$





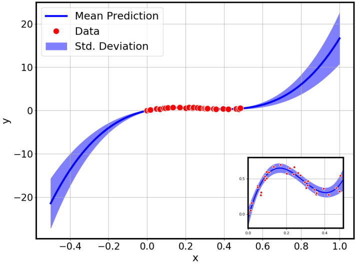
Variational posterior predictions heavily underestimate
both interpolative and extrapolative errors,
in the overparameterized regimes.

Polynomial fit: Extrapolation scenario

Order=3

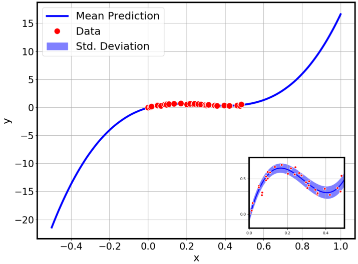
Full Bayesian Posterior

$\mathcal{N}(\mu, \Sigma)$



Variational Posterior

$\mathcal{N}(\mu, 1/\text{diag}(\Sigma^{-1}))$

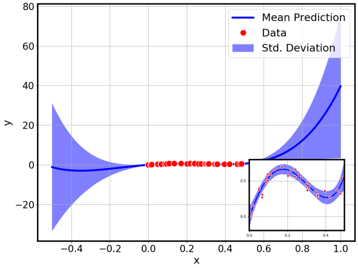


Variational posterior predictions heavily underestimate both interpolative and extrapolative errors, in the overparameterized regimes.

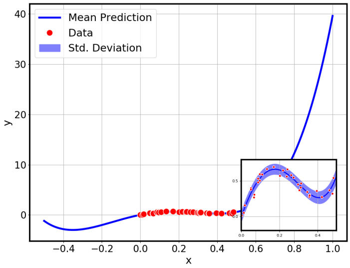
Polynomial fit: Extrapolation scenario

Order=4

Full Bayesian Posterior
$\mathcal{N}(\mu, \Sigma)$



Variational Posterior
$\mathcal{N}(\mu, 1/\text{diag}(\Sigma^{-1}))$



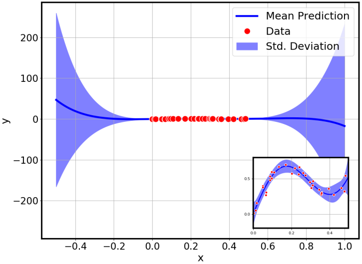
Variational posterior predictions heavily underestimate both interpolative and extrapolative errors, in the overparameterized regimes.

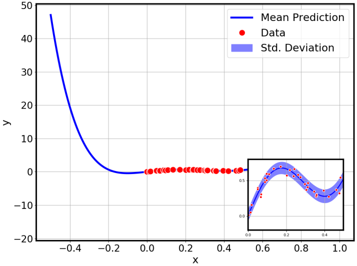
Polynomial fit: Extrapolation scenario

Order=5

Full Bayesian Posterior
$\mathcal{N}(\mu, \Sigma)$

Variational Posterior
$\mathcal{N}(\mu, 1/\text{diag}(\Sigma^{-1}))$



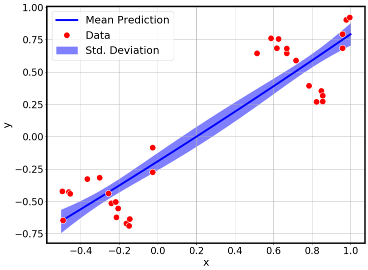


Variational posterior predictions heavily underestimate both interpolative and extrapolative errors, in the overparameterized regimes.

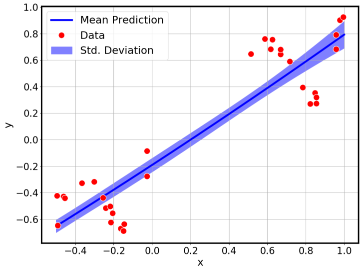
Polynomial fit: Interpolation scenario

Order=2

Full Bayesian Posterior
$\mathcal{N}(\mu, \Sigma)$



Variational Posterior
$\mathcal{N}(\mu, 1/\text{diag}(\Sigma^{-1}))$

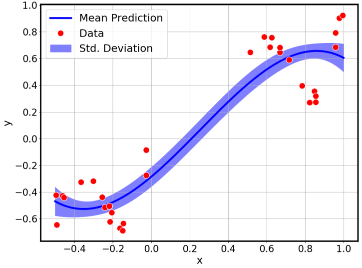


Variational posterior predictions heavily underestimate both interpolative and extrapolative errors, in the overparameterized regimes.

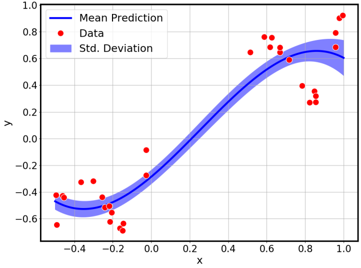
Polynomial fit: Interpolation scenario

Order=3

Full Bayesian Posterior
$\mathcal{N}(\mu, \Sigma)$



Variational Posterior
$\mathcal{N}(\mu, 1/\mathrm{diag}(\Sigma^{-1}))$

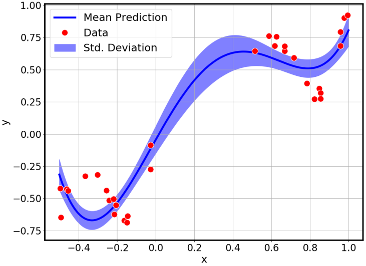


Variational posterior predictions heavily underestimate both interpolative and extrapolative errors, in the overparameterized regimes.

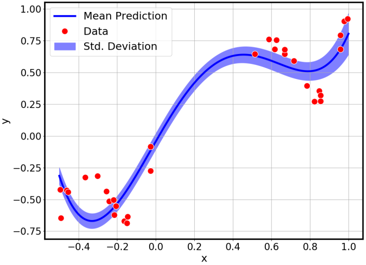
Polynomial fit: Interpolation scenario

Order=4

Full Bayesian Posterior
$\mathcal{N}(\mu, \Sigma)$



Variational Posterior
$\mathcal{N}(\mu, 1/\text{diag}(\Sigma^{-1}))$

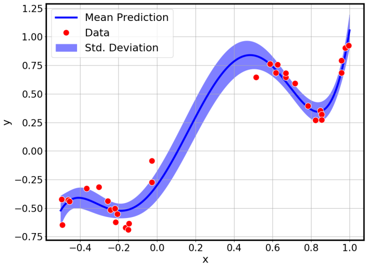


Variational posterior predictions heavily underestimate both interpolative and extrapolative errors, in the overparameterized regimes.

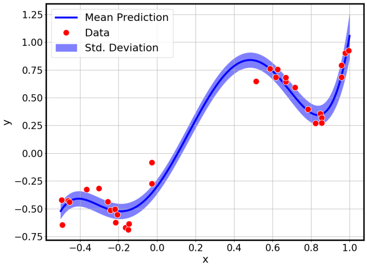
Polynomial fit: Interpolation scenario

Order=5

Full Bayesian Posterior
$\mathcal{N}(\mu, \Sigma)$



Variational Posterior
$\mathcal{N}(\mu, 1/\text{diag}(\Sigma^{-1}))$

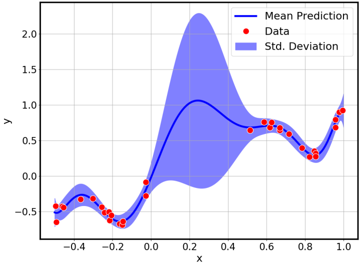


Variational posterior predictions heavily underestimate both interpolative and extrapolative errors, in the overparameterized regimes.

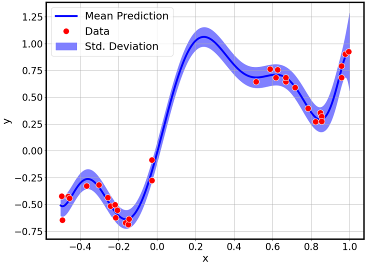
Polynomial fit: Interpolation scenario

Order=10

Full Bayesian Posterior
$\mathcal{N}(\mu, \Sigma)$



Variational Posterior
$\mathcal{N}(\mu, 1/\text{diag}(\Sigma^{-1}))$



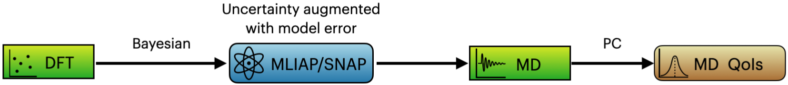
Variational posterior predictions heavily underestimate both interpolative and extrapolative errors, in the overparameterized regimes.

Summary Thank you!

- Bayesian fit of parameterized MLIAPs
  - Noise assumptions are crucial
- Embedded model error
  - Statistical correction *inside* the model: joint inference of model parameters and the correction
  - Leads to model-driven noise model
  - Meaningful model-error uncertainty capturing the true residual
  - A few shortcuts in linear regression models
  - Choices to make: priors, approximate likelihoods, MCMC sampler, where to embed...
- Variational inference
  - Approximate alternative to MCMC for nonlinear, complex models
  - Underestimates the uncertainty for overparameterized models: dangerous when extrapolating!
  - Mechanically similar to embedded model error, except the optimization objective/method (and, potentially, the interpretation!)

Additional Material

Uncertainty Propagation through MD



- PC intro setup; SNAP coefficients form a first order Gauss-Hermite Polynomial Chaos (PC)

$$E \approx \sum_{k=0}^{P} \underbrace{(c_k + d_k \xi_k)}_{\tilde{c}} B_k(x)$$

- Sample SNAP coefficients
- Evaluate MD QoIs
- Build PC for MD QoIs, possibly multilevel/multifidelity
- Evaluate PDF/statistics of QoIs
- Challenges: high-d input, noisy MD simulations

Uncertainty-enabling wrappers over PyTorch modules

Deterministic → **Probabilistic**

torch.nn.module → **wrapper(torch.nn.module)**

Option 1: ensemble NN

```
nn_ens = EnsRegr(torch.nn.module, nens=111)
```

```
class EnsRegr():
    def __init__(self, nnmodule, nens=1, verbose=False):
        self.nnmodel = nnmodule
        self.verbose = verbose
        self.nens = nens
```

Option 2: NN learning with MCMC

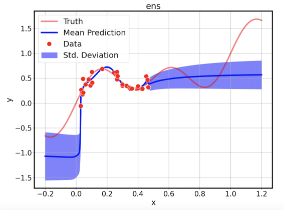
```
nn_mcmc = MCMCRegr(torch.nn.module)
```

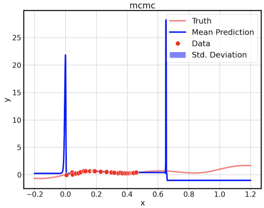
```
class MCMCRegr():
    def __init__(self, nnmodule, verbose=True):
        self.nnmodule = nnmodule
        self.verbose = verbose
```

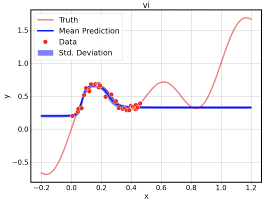
Option 3: NN learning with VI

```
nn_vi = VIRegr(torch.nn.module)
```

```
class VIRegr():
    def __init__(self, nnmodule, verbose=False):
        self.bmodel = BNet(nnmodule)
        self.verbose = verbose
```







- MCMC struggles with complex NNs; VI underestimates; Ensembles do well

Uncertainty-enabling wrappers over PyTorch modules

Option 1: ensemble NN

```
nn_ens = EnsRegr(torch.nn.module, nens=111)
```

```
class EnsRegr():
    def __init__(self, nnmodule, nens=1, verbose=False):
        self.nnmodel = nnmodule
        self.verbose = verbose
        self.nens = nens
```

Option 2: NN learning with MCMC

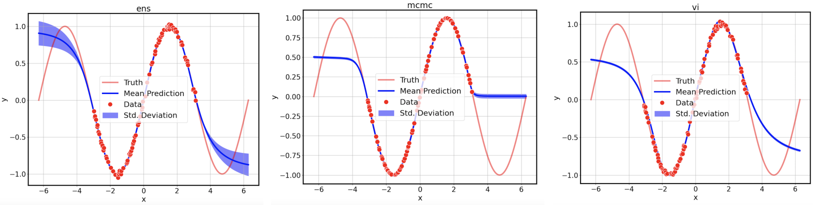
```
nn_mcmc = MCMCRegr(torch.nn.module)
```

```
class MCMCRegr():
    def __init__(self, nnmodule, verbose=True):
        self.nnmodule = nnmodule
        self.verbose = verbose
```

Option 3: NN learning with VI

```
nn_vi = VIRegr(torch.nn.module)
```

```
class VIRegr():
    def __init__(self, nnmodule, verbose=False):
        self.bmodel = BNet(nnmodule)
        self.verbose = verbose
```



- MCMC struggles with complex NNs; VI underestimates; Ensembles do well

Literature

Model error embedding

- [Sargsyan et al., 2019] “Embedded model error representation for Bayesian model calibration”, *Int. J. Uncertain. Quantif.*, 9(4), 2019.

MLIAPs

- [Thompson et al., 2015] “Spectral neighbor analysis method for automated generation of quantum-accurate interatomic potentials”, *J Comp Phys*, 2015.
- [J. Behler, 2014] “Representing potential energy surfaces by high-dimensional neural network potentials”, *J. Phys.: Condens. Matter*, 26, 2014.

Active learning

- [B. Settles, 2009] “Active learning literature survey”, *Comp Sci Tech Report 1648*, University of Wisconsin-Madison, 2009.

Active learning for MLIAPs

- [E. Podryabinkin, A. Shapeev, 2017] “Active learning of linearly parametrized interatomic potentials”, *Comp Mat Sci*, 140, 2017.
- [J. Vandermause et al., 2020] “On-the-fly active learning of interpretable Bayesian force fields for atomistic rare events”, *npj Computational Materials*, 6, 2020.