Model Error Estimation for Active Learning of Interatomic Potential Models

Khachik Sargsyan[1],
Logan Williams[2], Habib N. Najm[1]

[1]Sandia National Laboratories, Livermore, CA
[2]now at Lawrence Livermore National Laboratory, Livermore, CA

SIAM CSE, Amsterdam, Netherlands
Feb 26 - Mar 4, 2023

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

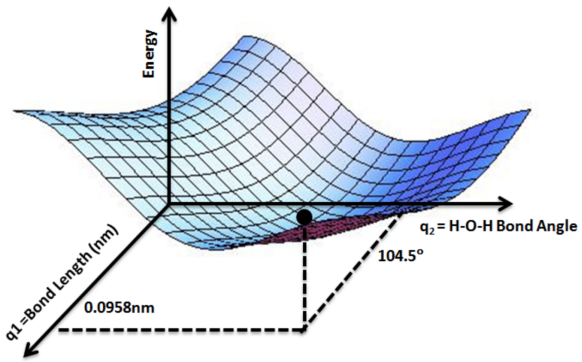
- Motivation: potential energy surface approximation
  - Machine learning for interatomic potentials (MLIAP)
  - Focus on SNAP potentials: linear regression
- Bayesian estimation of MLIAPs
  - Importance of noise model assumptions
  - **Embedded model error** approach and likelihood construction
- Active learning (AL) strategies for ML
  - Uncertainty-informed AL
  - Preliminary results

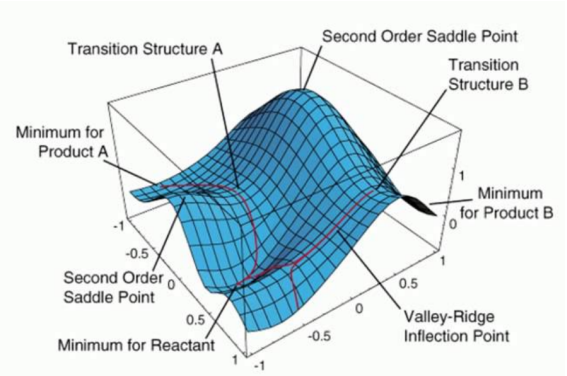
Target: Potential energy surface (PES) approximation

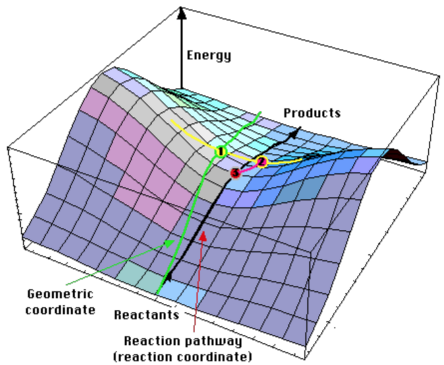
$$E = f(x)$$

x represents coordinates/descriptors
E is energy

- Accurate and fast surrogates for PES to replace quantum mechanical computations for studies requiring many PES inquiries
  - saddle point search, transition paths, barrier heights
  - rapid assessment of reaction characteristics
  - automate the discovery of reactive pathways



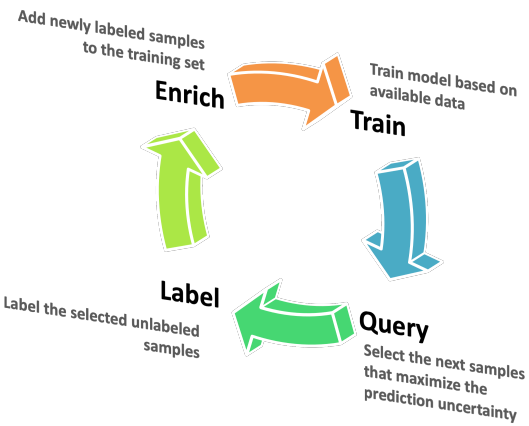




Active Learning and UQ are needed

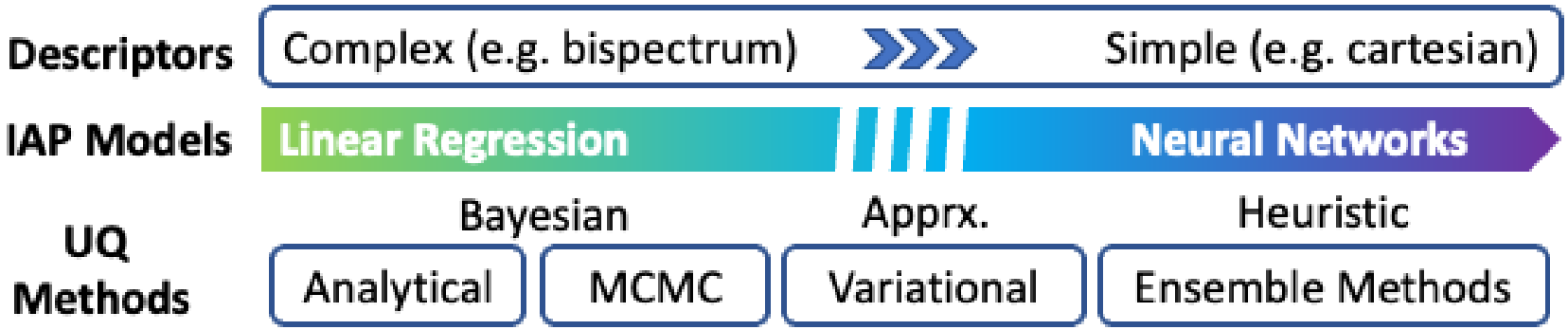
Greater test accuracy with fewer training samples

- Two challenging regimes:
  - Interpolation: developing a reliable problem-specific IAP that would accurately interpolate within the training domain is nontrivial
  - Extrapolation: prediction outside the training domain is even harder



- Key: *query strategy*, whether to query high-fidelity quantum mechanical (QM) simulation or not.
  - If such decision can be made reliably, then one does not need to start with a very good training set
  - Rely on a well-calibrated uncertainty estimate

Focus on SNAP (Left end of the figure)



- [Thompson et al., 2015] SNAP: "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: $c \sim \mathcal{N}\left((B^TB)^{-1}B^Ty, \sigma^2(B^TB)^{-1}\right)$

Posterior pushed-forward uncertainty does not capture true discrepancy

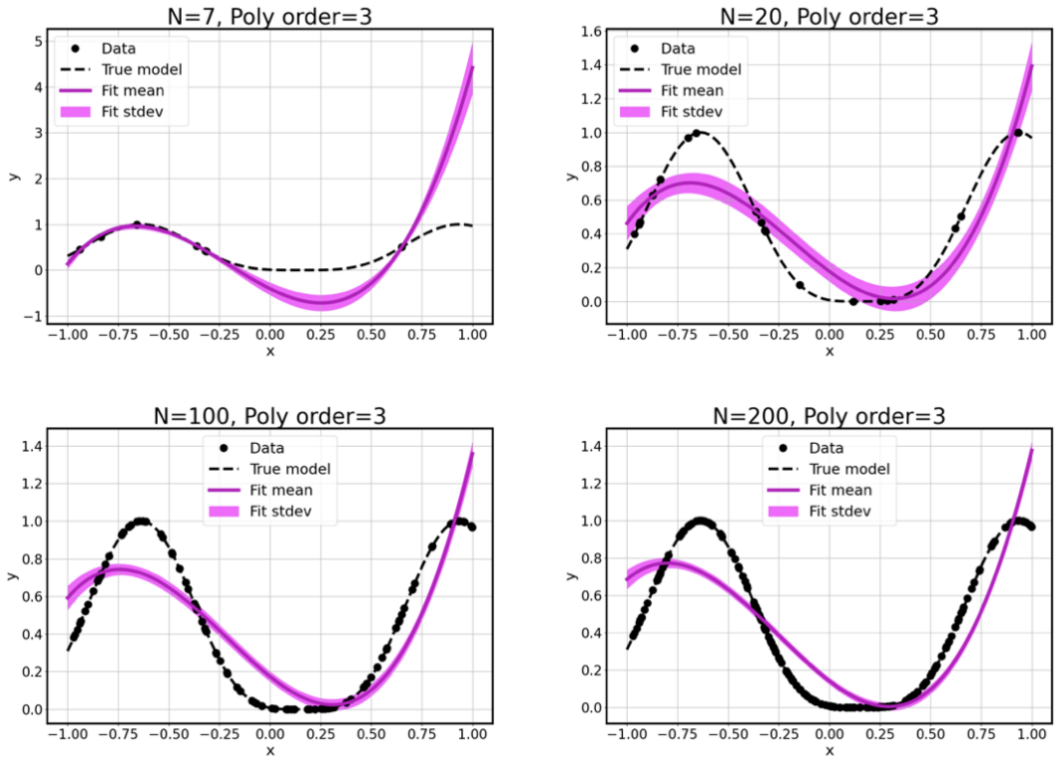
Synthetic data

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

Cubic fit

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

More data leads to overconfident prediction



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

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

Model ≠ 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. hi-fi (QM) is data, and low-fi (SNAP) is model

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:
For linear models, there is no formal distinction between internal and external corrections: but the error is now model-informed.

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

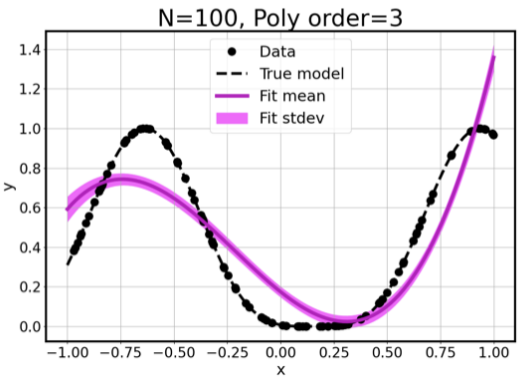
Synthetic data

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

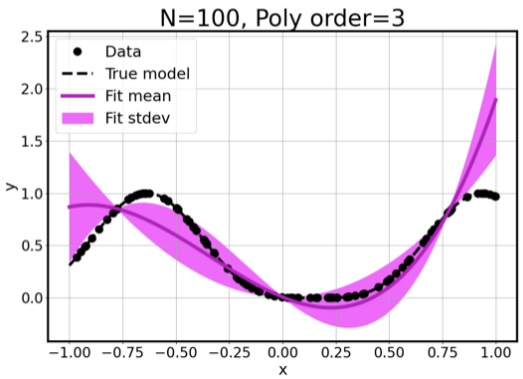
Cubic fit

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

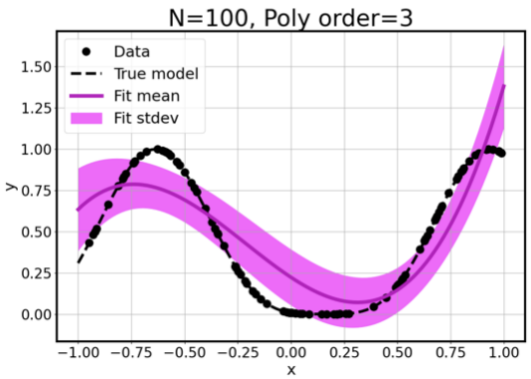
Classical case



Model error, IID likelihood

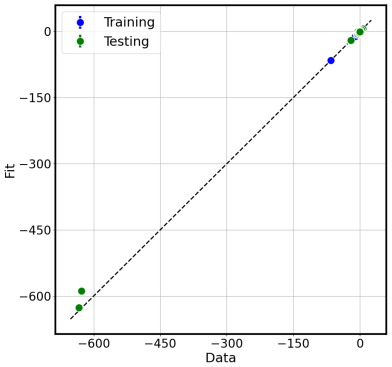


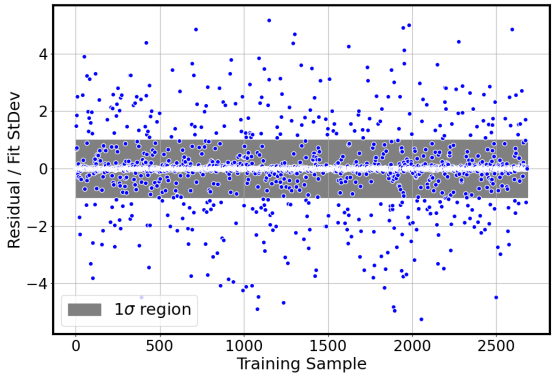
Model error, ABC likelihood

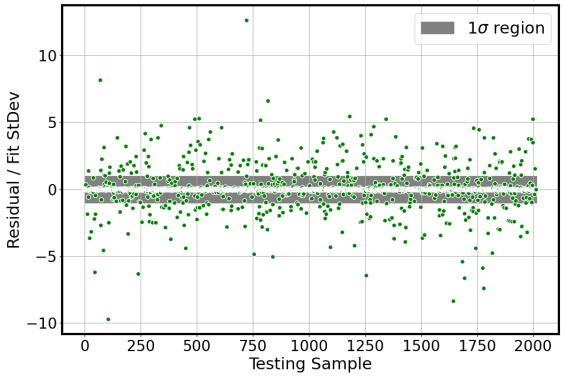


Uncertainty validation: W-ZrC Dataset

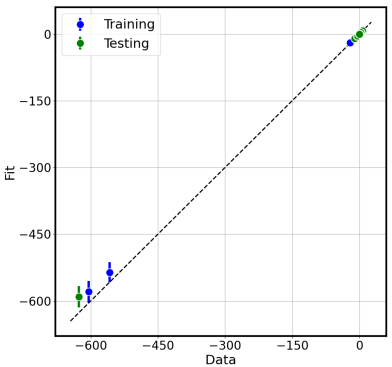
Uncertainty without model error

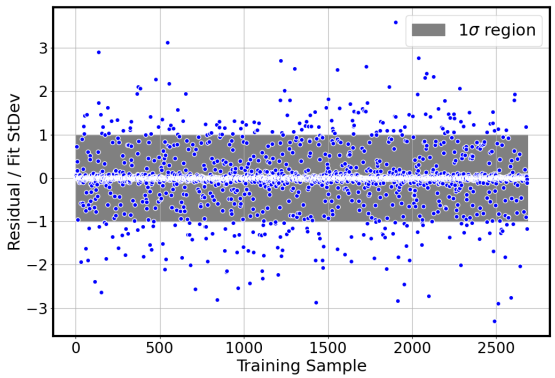


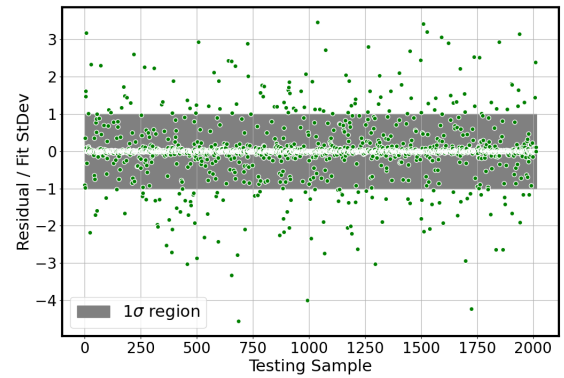




Uncertainty with model error



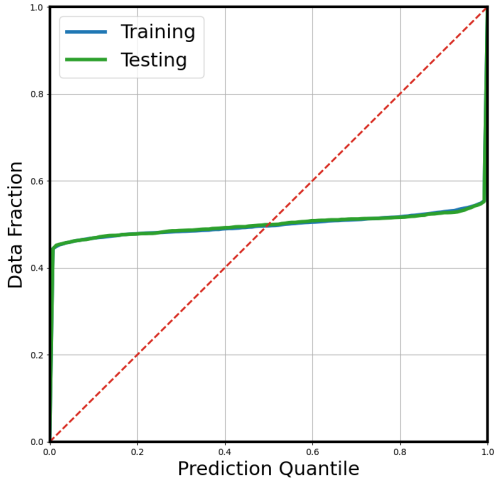


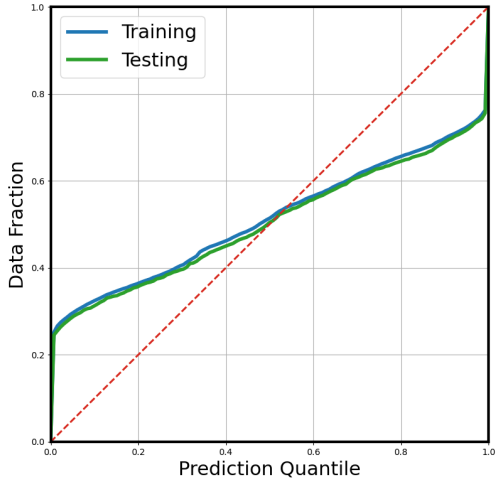


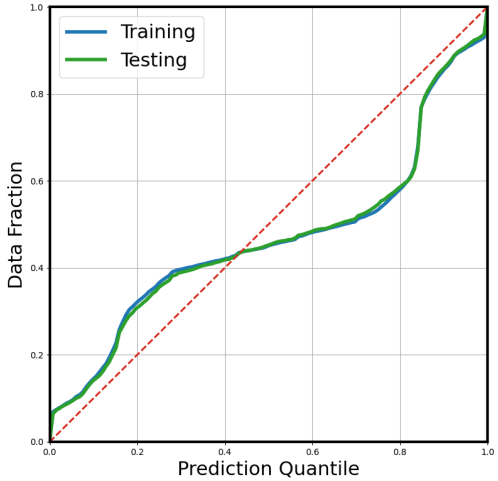
Uncertainty validation: two examples

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

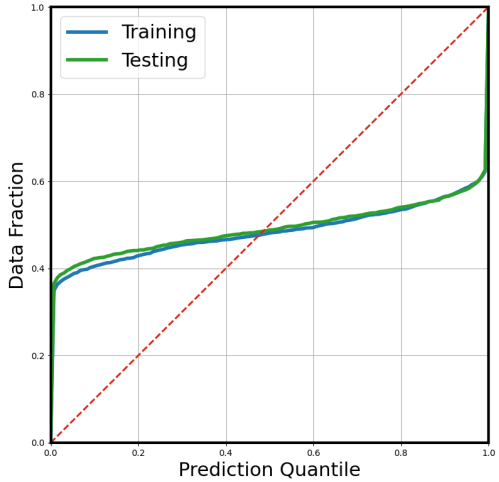
Ta

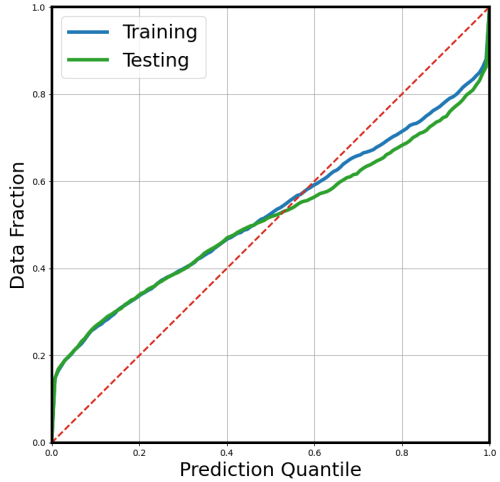


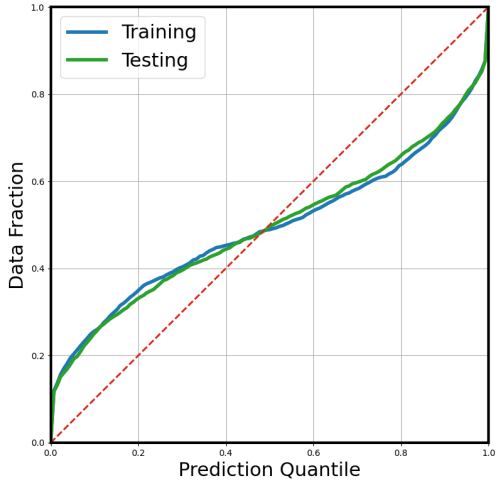




W-ZrC







Several challenges and choices

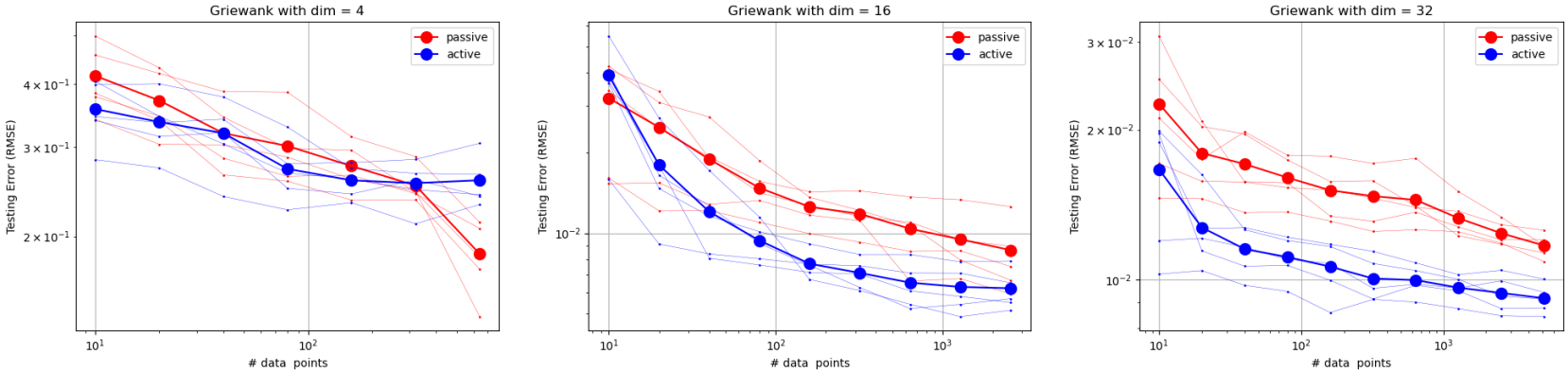
- Embedding type, e.g. additive/multiplicative

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

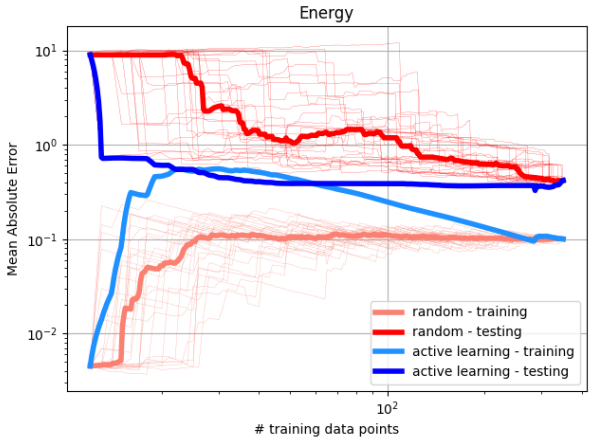
- Degenerate (Gaussian) likelihoods: resort to approximate Bayesian computation (ABC) or independent (IID) assumptions
- Difficult posterior PDFs for MCMC, choice of priors for embedding parameters
- Which coefficients to embed the model error in?
- Bonus: embedded model error approach is mechanically similar to variational inference (a.k.a. Bayesian NNs)

Notes and preliminary results on active learning

- Efficiency of active learning improves with higher dimension.



- Clustering (or rather, de-clustering) is needed in pool-based active learning
- Preliminary results on a set of benchmark material science problems



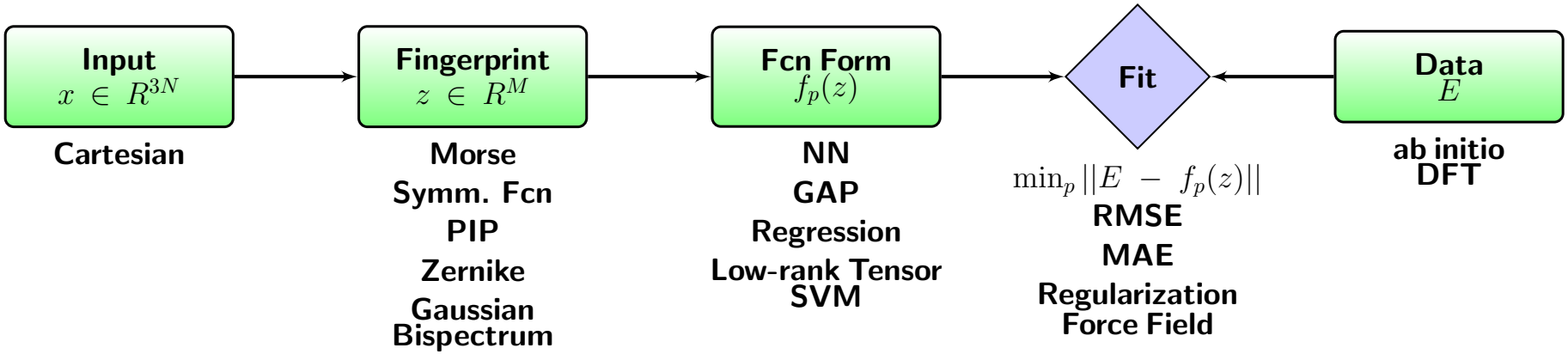
Summary

Thank you!

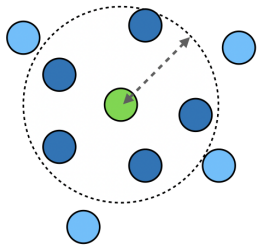
- Bayesian fit of parameterized interatomic potential models
  - Focused on linear models, but the framework applies to all
  - Noise assumptions are crucial
- Embedded model error
  - Statistical correction *inside* the model: joint inference of model parameters and the correction
  - Leads to model-driven noise model with baked-in uncertainty
  - 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...
- Active learning
  - Anchored in uncertainty estimation (and clustering)
  - Choices to make: query strategy, UQ method, metric of 'newness'...

Additional Material

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



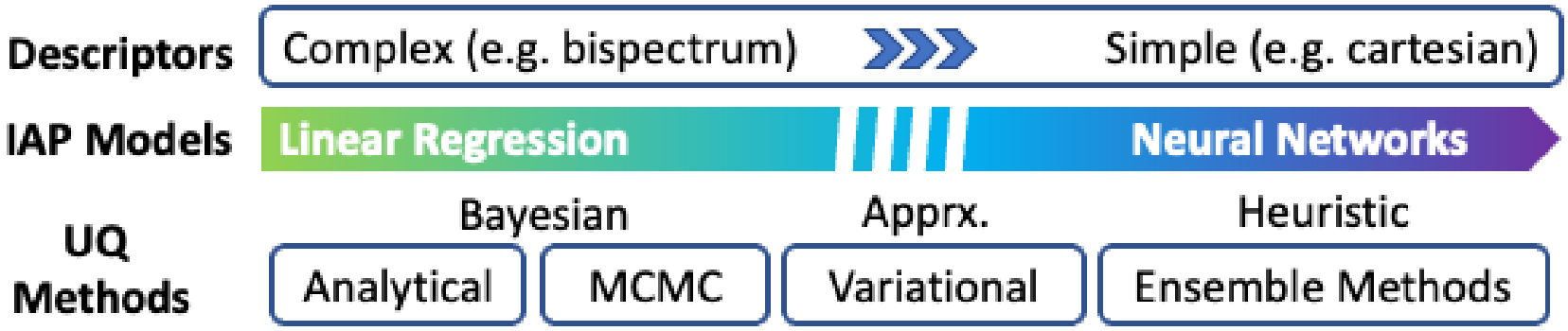
- Require the energy, forces and/or stresses predicted by a MLIAP to be close to those obtained by a quantum mechanical model on some atomic configurations (a.k.a. training set)

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

$$y \approx f_c(x)$$



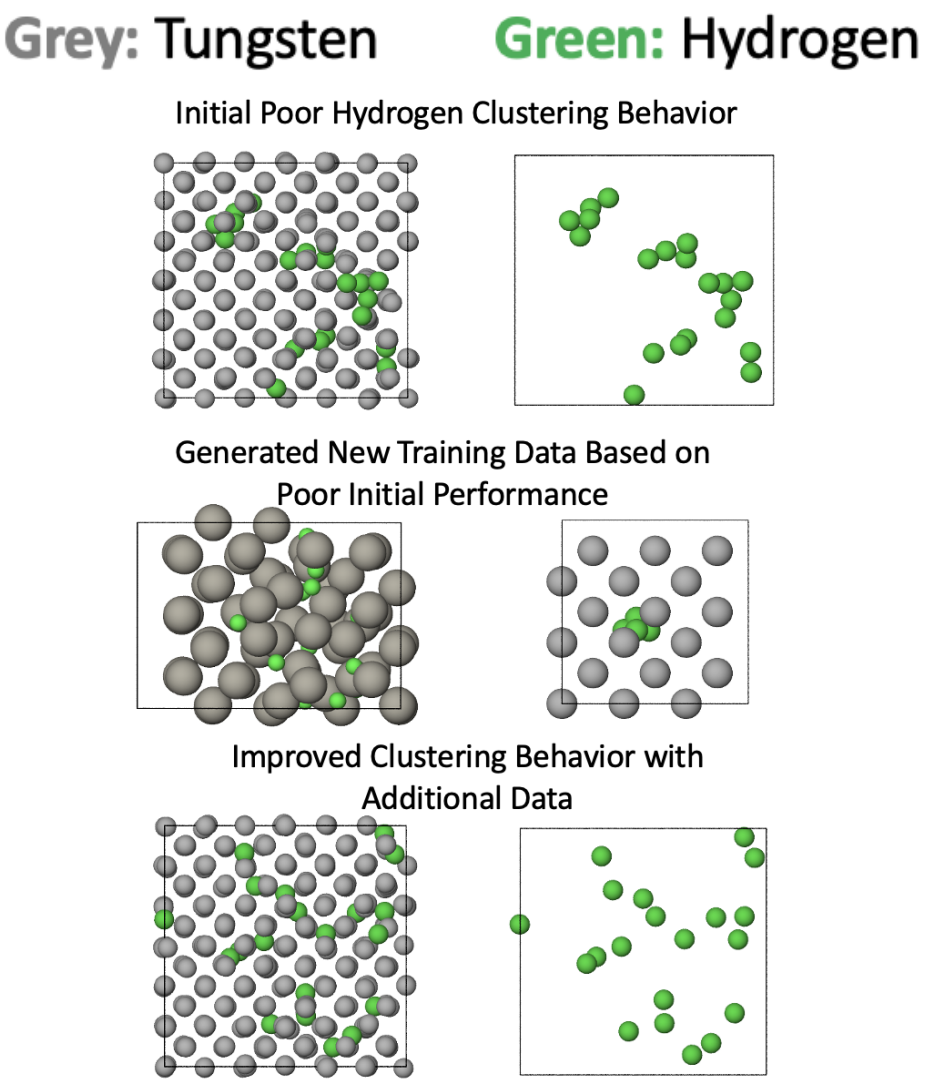
Uncertainty estimation options

$$y \approx f_c(x)$$

- Bayesian inference: $\overbrace{P(c|y)}^{\text{Posterior}} \propto \overbrace{P(y|c)}^{\text{Likelihood}} \overbrace{P(c)}^{\text{Prior}}$
  - Markov chain Monte Carlo sampling of posterior PDF
- Variational methods: $c \sim N(\mu, \Sigma)$ and optimize μ, Σ.
  - Largely, this is also called Bayesian Neural Networks
  - Stochastic gradient descent to minimize evidence lower bound
- Ensemble methods: many flavors.
  - Deep ensembles
  - Query-by-committee
  - Boosting/bagging

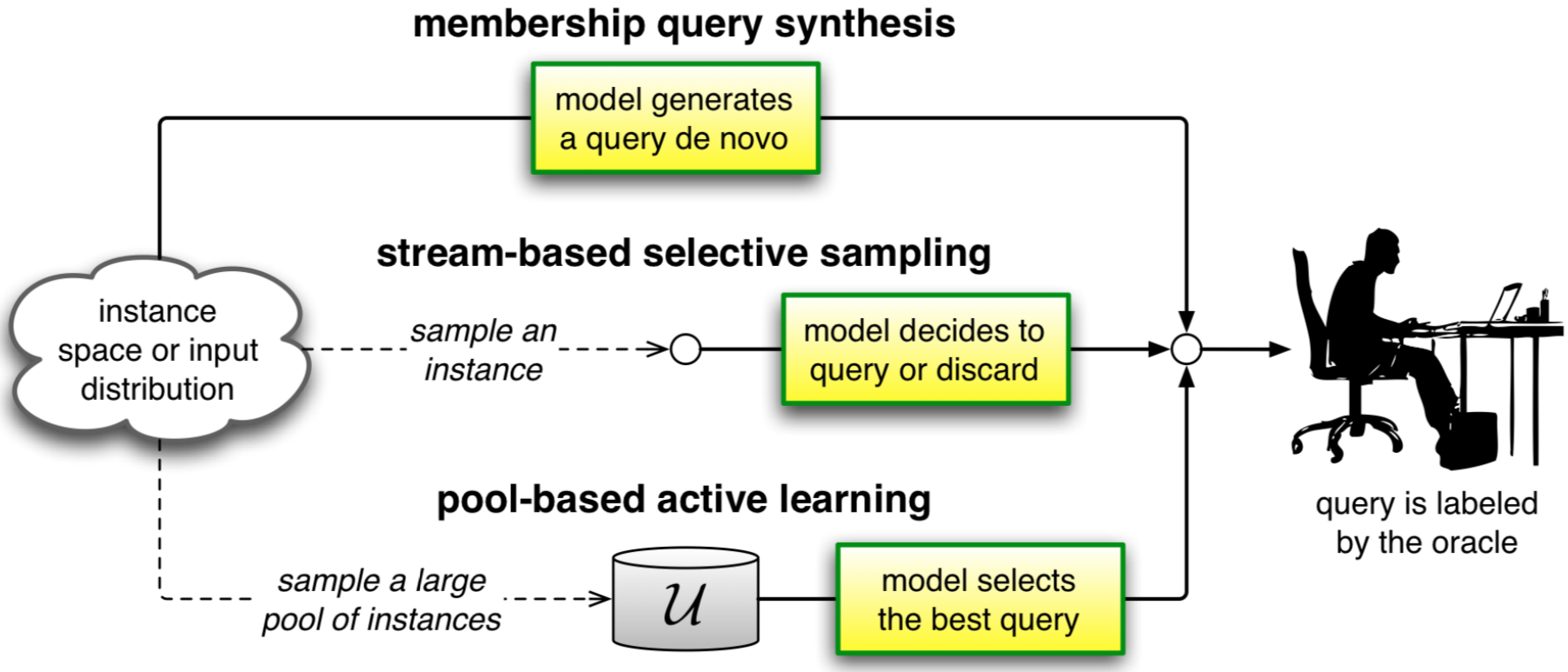
Training set selection is crucial

- Configurations chosen for training data influence results
- Example: W-H (tungsten/hydrogen) IAPs
- Initial IAPs resulted in hydrogen clusters in bulk tungsten, which should not occur
- Additional training data was generated and put into the training set
- Including these specific configurations prevented unphysical hydrogen clustering



Results from Mary Alice Cusentino (SNL), using LAMMPS software.

Active Learning: selection of training configurations

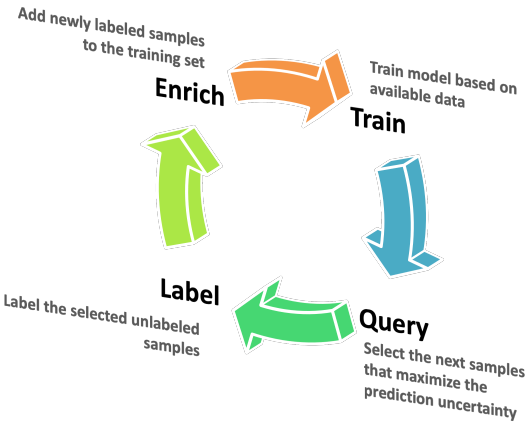


[B. Settles, “Active learning literature survey”, Computer Sciences Technical Report 1648, University of Wisconsin-Madison, 2009]

Active Learning: selection of training configurations

Greater test accuracy with fewer training samples

- Two flavors of the challenge:
  - Interpolation: developing a reliable problem-specific MLIAP that would accurately interpolates within the training domain is nontrivial
  - Extrapolation: prediction outside the training domain is even harder



- Key: *query strategy*, whether to query high-fidelity quantum mechanical (QM) simulation or not.
  - If such decision can be made reliably, then one does not need to start with a very good training set

Query Strategies:
almost all rely on some form of uncertainty estimate

- **Uncertainty sampling:** an active learner queries the instances about which it is least certain how to label.
- **Query-by-committee:** committee of competing models, and pick a query about which they most disagree. Need a measure of disagreement.
- **Expected model change:** which query would lead to greatest model change, e.g. largest gradient length.
- **Variance Reduction and Fisher Information Ratio:** minimizing the variance component of generalization error estimate (via Fisher Information)
- **Estimated error reduction:** Estimate the expected future error that would result if some new instance x is labeled and added to training set, and then select the instance that minimizes that expectation.

Optimality options

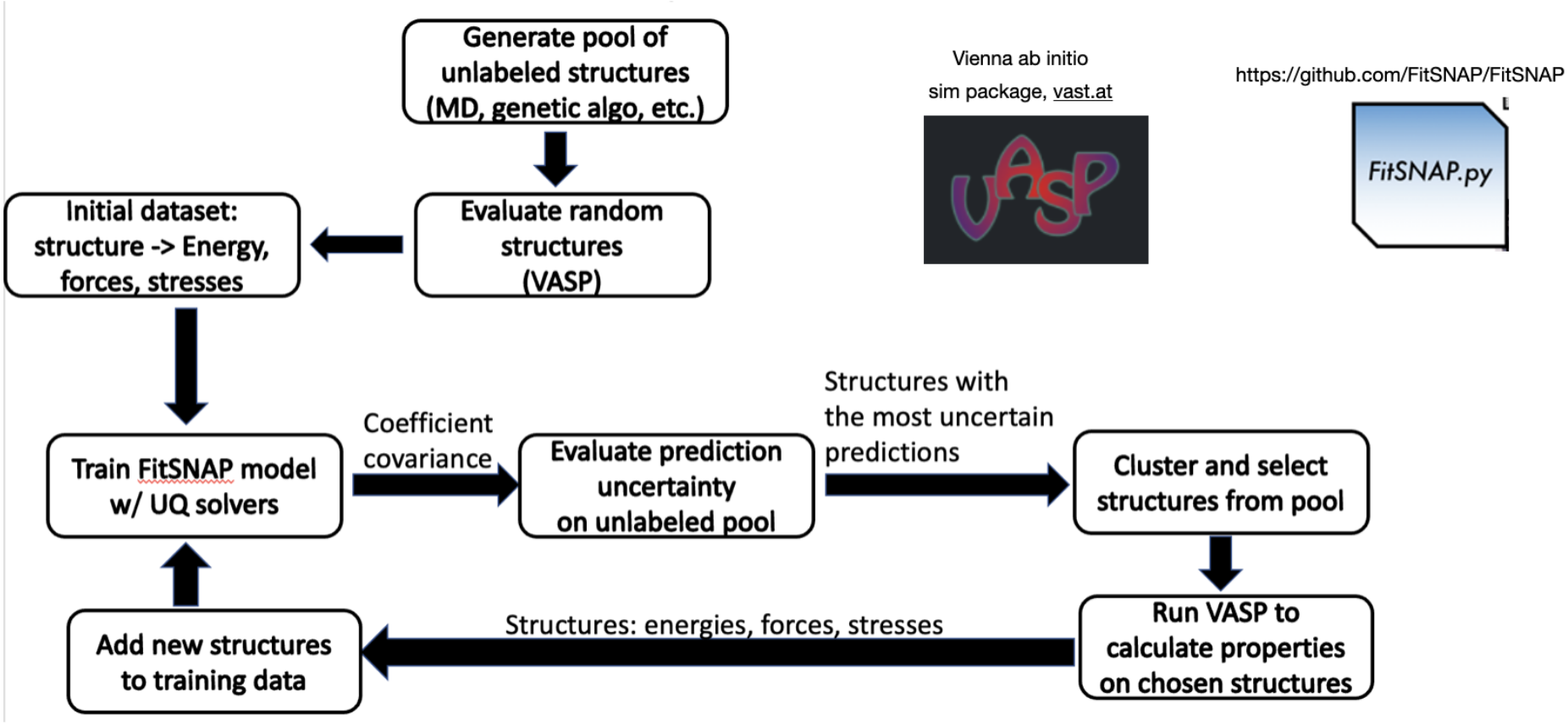
Straight out of wiki...

- **A**-optimality ("**average**" or **trace**)
  - One criterion is **A-optimality**, which seeks to minimize the trace of the inverse of the information matrix. This criterion results in minimizing the average variance of the estimates of the regression coefficients.
- **C**-optimality
  - This criterion minimizes the variance of a best linear unbiased estimator of a predetermined linear combination of model parameters.
- **D**-optimality (**determinant**)
  - A popular criterion is **D-optimality**, which seeks to minimize $|(X'X)^{-1}|$, or equivalently maximize the determinant of the information matrix X'X of the design. This criterion results in maximizing the differential Shannon information content of the parameter estimates.
- **E**-optimality (**eigenvalue**)
  - Another design is **E-optimality**, which maximizes the minimum eigenvalue of the information matrix.
- **T**-optimality
  - This criterion maximizes the trace of the information matrix.

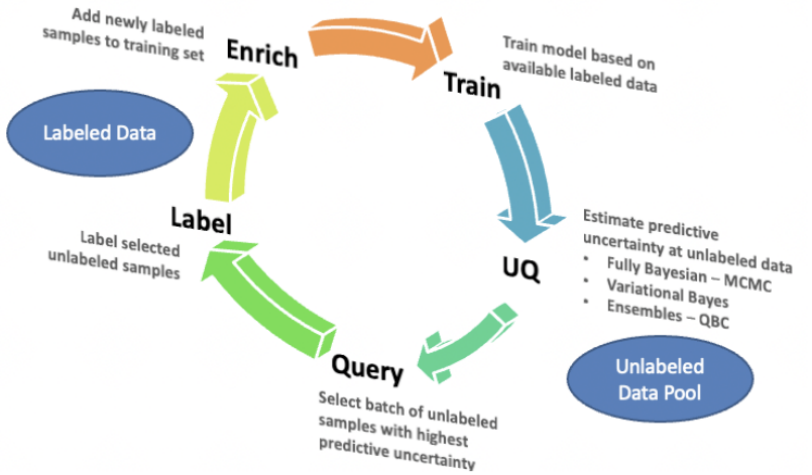
Other optimality-criteria are concerned with the variance of predictions:

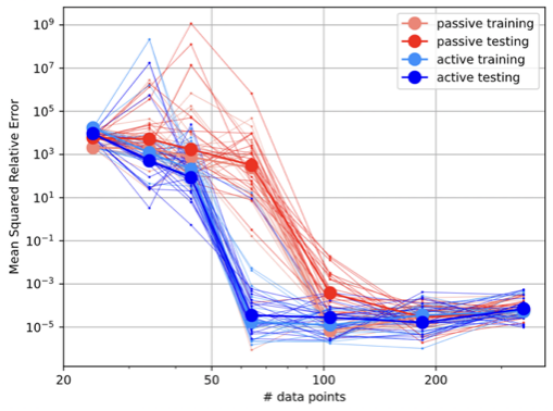
- **G**-optimality
  - A popular criterion is **G-optimality**, which seeks to minimize the maximum entry in the diagonal of the hat matrix $X(X'X)^{-1}X'$. This has the effect of minimizing the maximum variance of the predicted values.
- **I**-optimality (**integrated**)
  - A second criterion on prediction variance is **I-optimality**, which seeks to minimize the average prediction variance *over the design space*.
- **V**-optimality (**variance**)
  - A third criterion on prediction variance is **V-optimality**, which seeks to minimize the average prediction variance

Active Learning: current workflow



Active Learning: Query Options





Query-by-Committee (QBC)

- Launch K learners, each with fN training points (f=0.8)
- Evaluate the learners' performance at all points in the pool
- Select training points from the pool that correspond to the highest 'disagreement' and add them to the training set

Bayesian Uncertainty

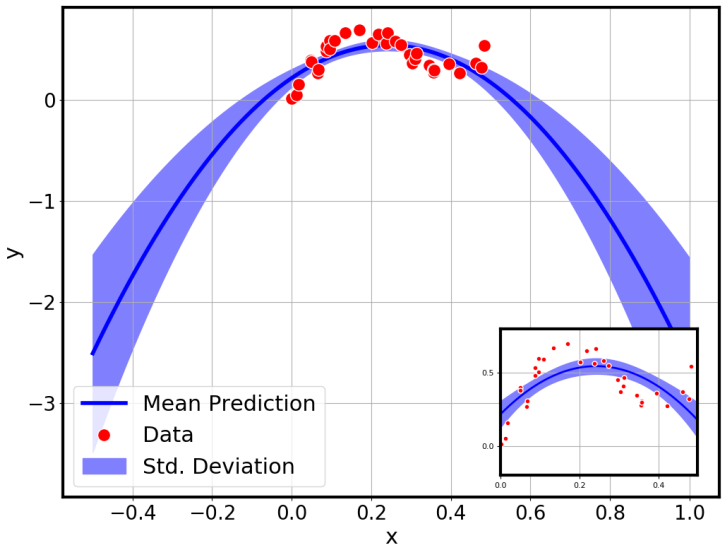
- Launch a single learner
- Evaluate its performance at all points in the pool
- Select training points from the pool that correspond to the highest posterior uncertainty and add them to the training set

Polynomial fit: Extrapolation scenario

Order=2

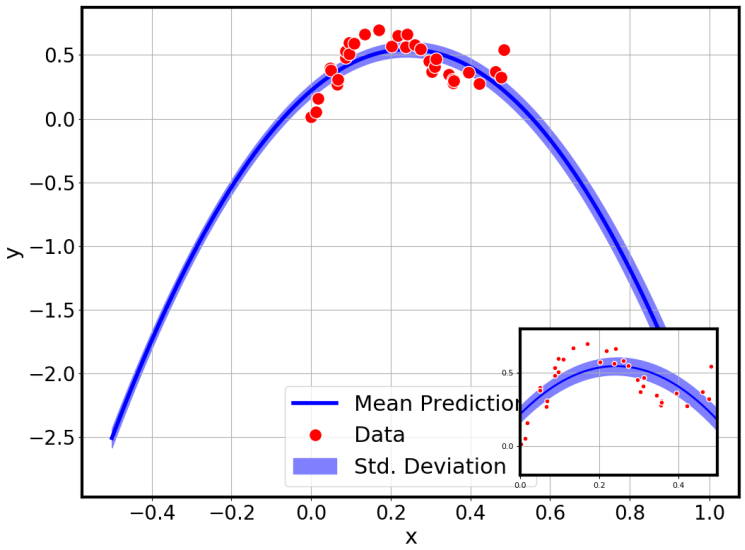
Full Bayesian Posterior

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



Variational Posterior

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



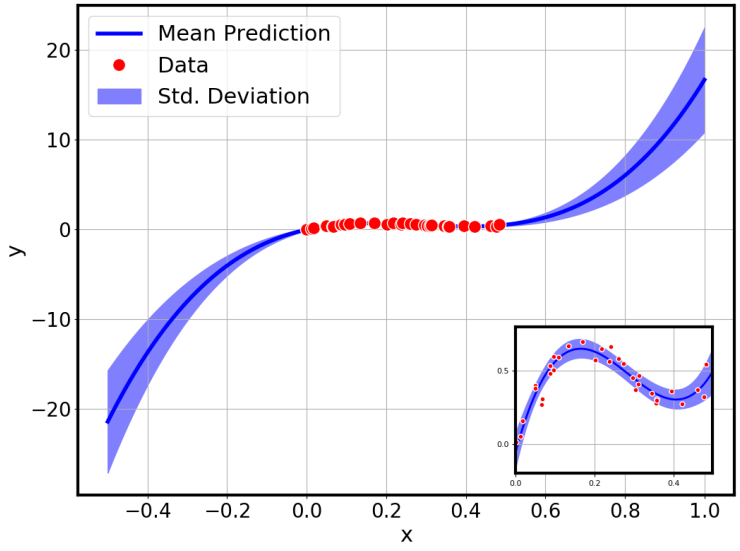
Variational posterior predictions heavily underestimate
both interpolative and extrapolative errors,
in the overparameterized regimes.

Polynomial fit: Extrapolation scenario

Order=3

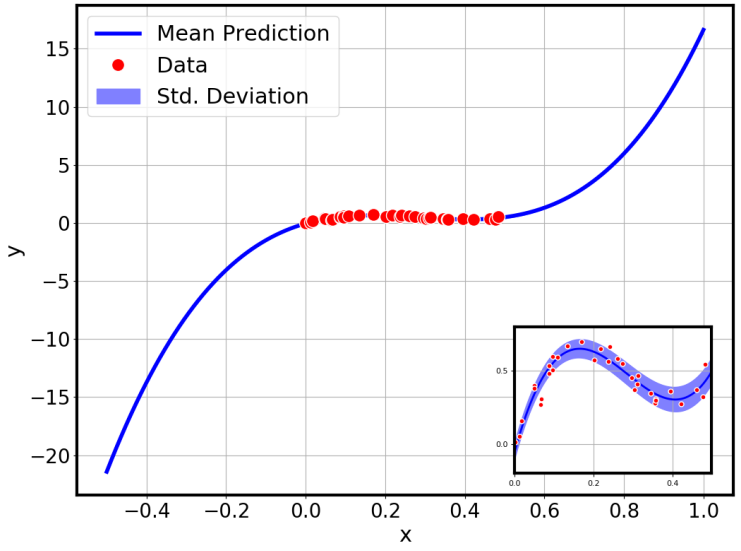
Full Bayesian Posterior

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



Variational Posterior

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



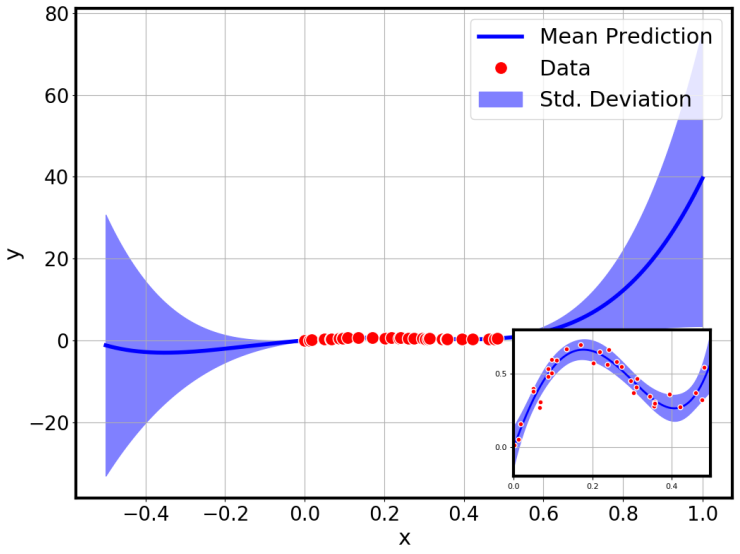
Variational posterior predictions heavily underestimate
both interpolative and extrapolative errors,
in the overparameterized regimes.

Polynomial fit: Extrapolation scenario

Order=4

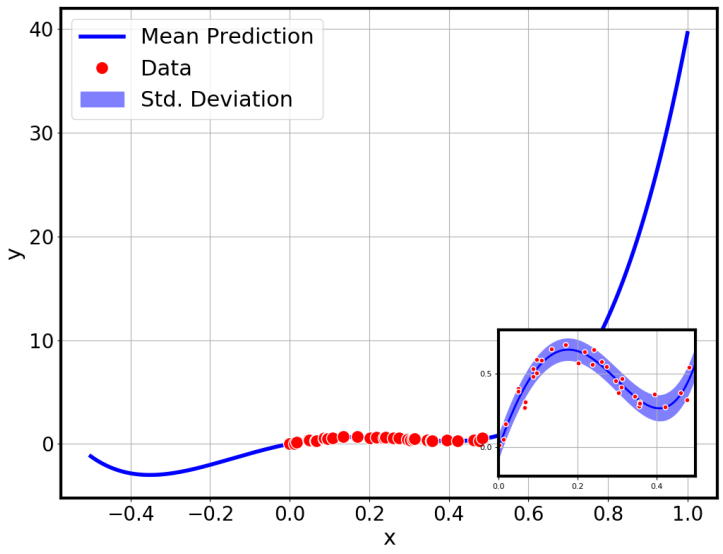
Full Bayesian Posterior

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



Variational Posterior

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



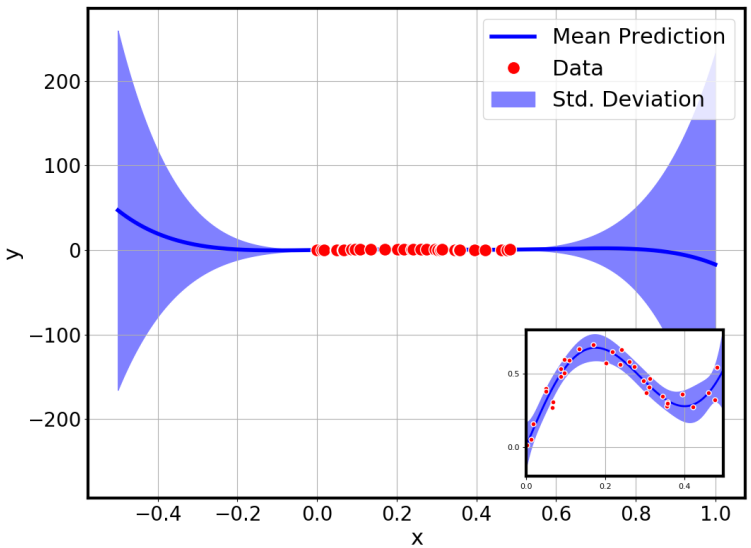
Variational posterior predictions heavily underestimate
both interpolative and extrapolative errors,
in the overparameterized regimes.

Polynomial fit: Extrapolation scenario

Order=5

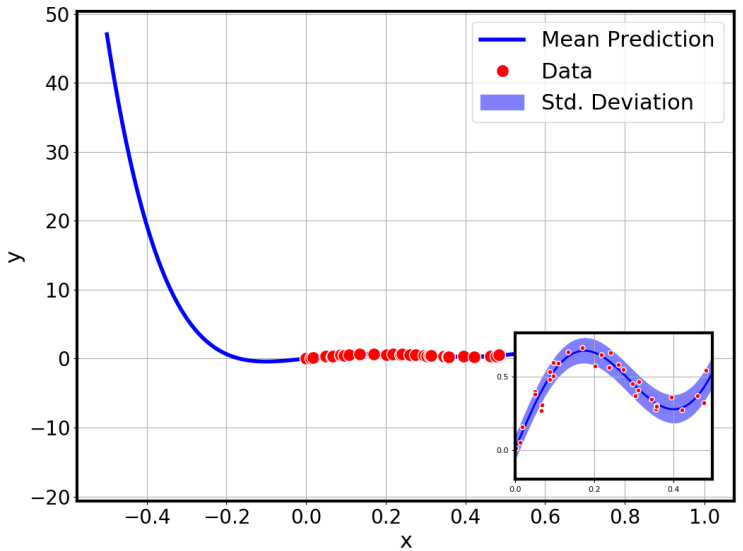
Full Bayesian Posterior

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



Variational Posterior

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



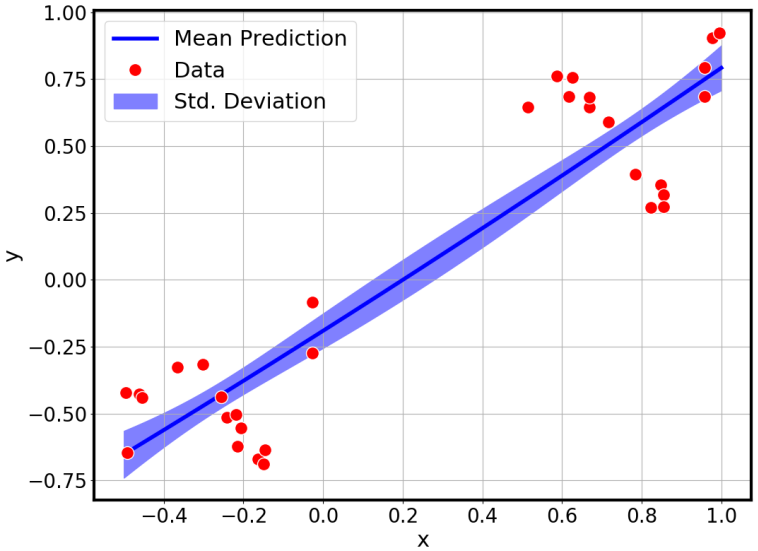
Variational posterior predictions heavily underestimate
both interpolative and extrapolative errors,
in the overparameterized regimes.

Polynomial fit: Interpolation scenario

Order=2

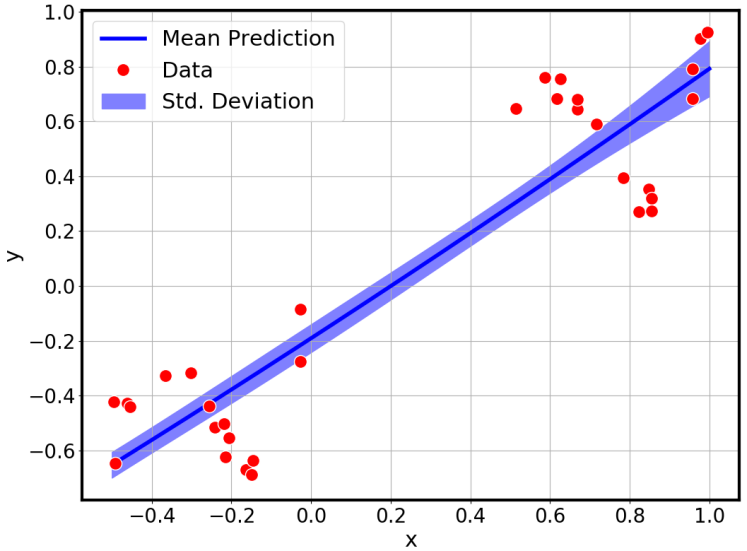
Full Bayesian Posterior

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



Variational Posterior

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



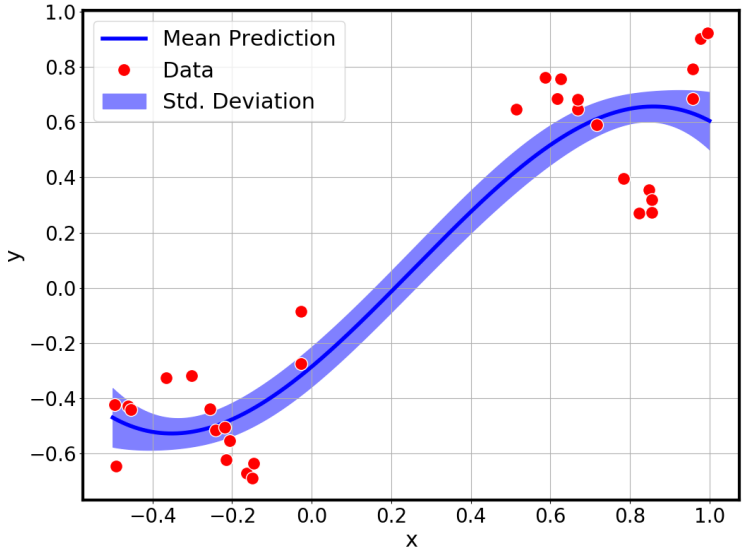
Variational posterior predictions heavily underestimate
both interpolative and extrapolative errors,
in the overparameterized regimes.

Polynomial fit: Interpolation scenario

Order=3

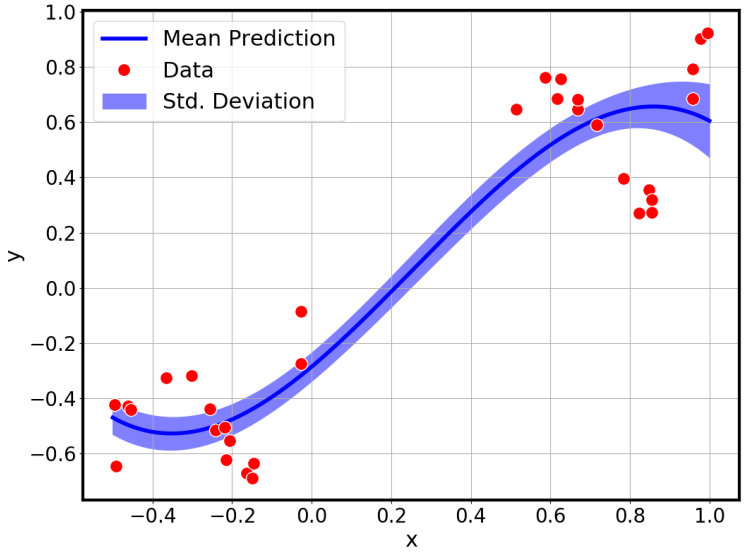
Full Bayesian Posterior

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



Variational Posterior

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



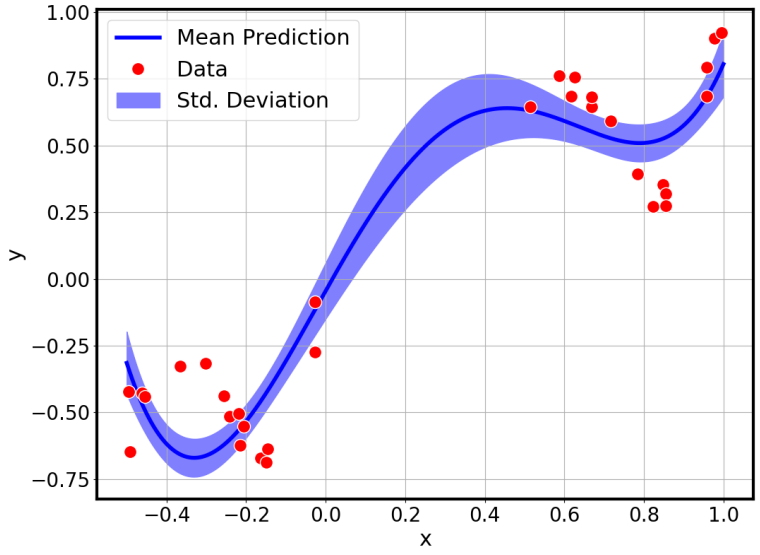
Variational posterior predictions heavily underestimate
both interpolative and extrapolative errors,
in the overparameterized regimes.

Polynomial fit: Interpolation scenario

Order=4

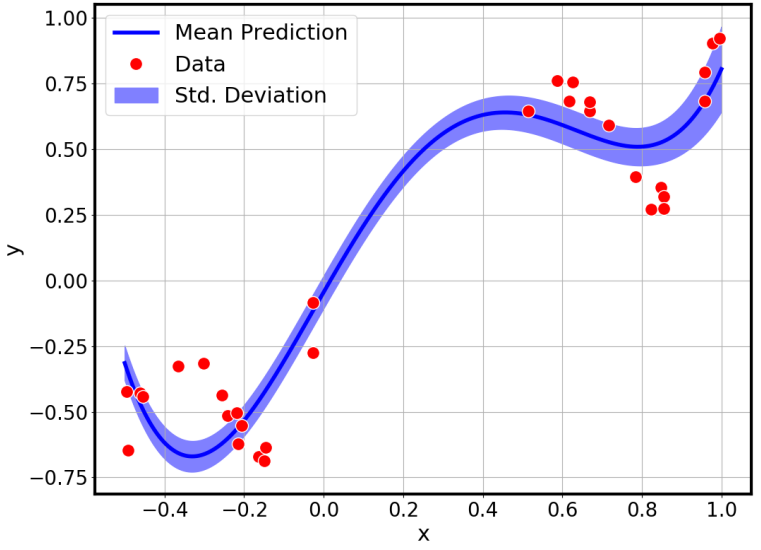
Full Bayesian Posterior

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



Variational Posterior

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



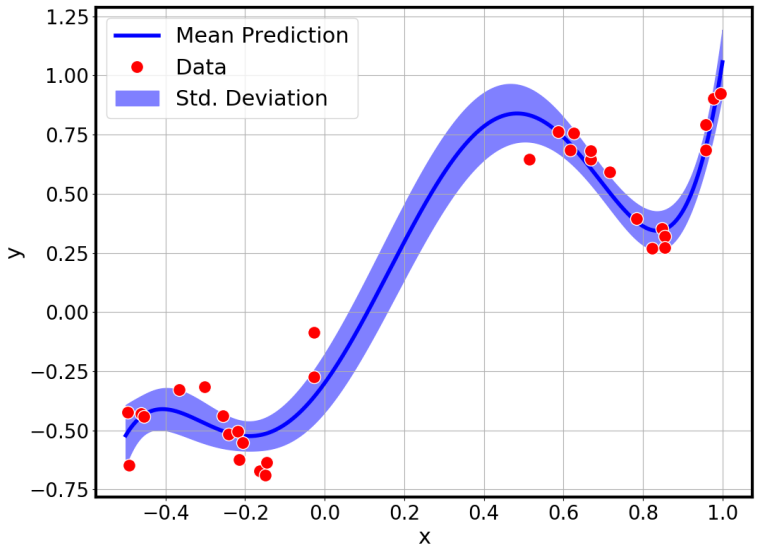
Variational posterior predictions heavily underestimate
both interpolative and extrapolative errors,
in the overparameterized regimes.

Polynomial fit: Interpolation scenario

Order=5

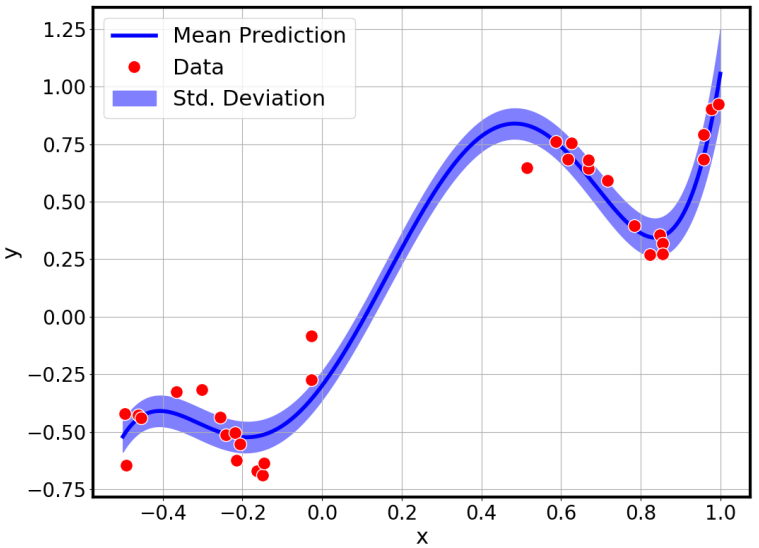
Full Bayesian Posterior

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



Variational Posterior

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



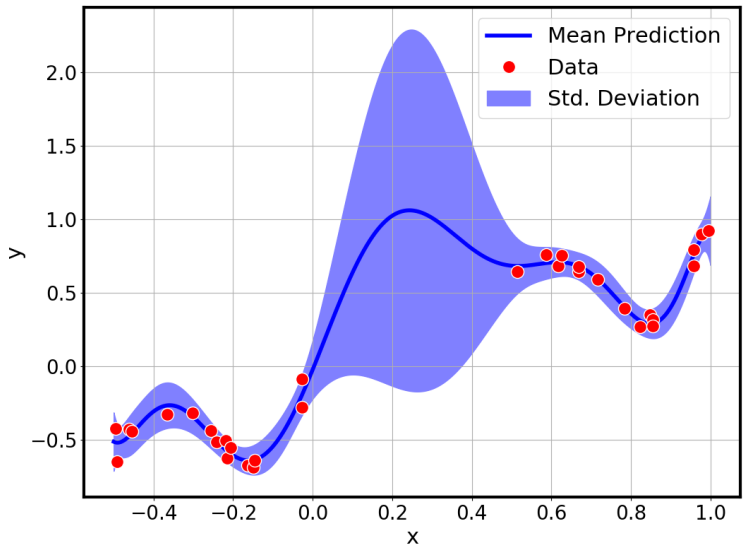
Variational posterior predictions heavily underestimate
both interpolative and extrapolative errors,
in the overparameterized regimes.

Polynomial fit: Interpolation scenario

Order=10

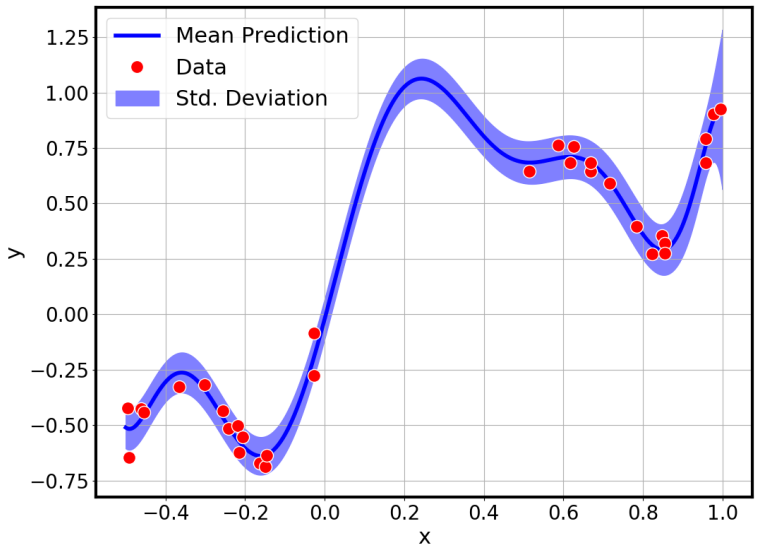
Full Bayesian Posterior

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



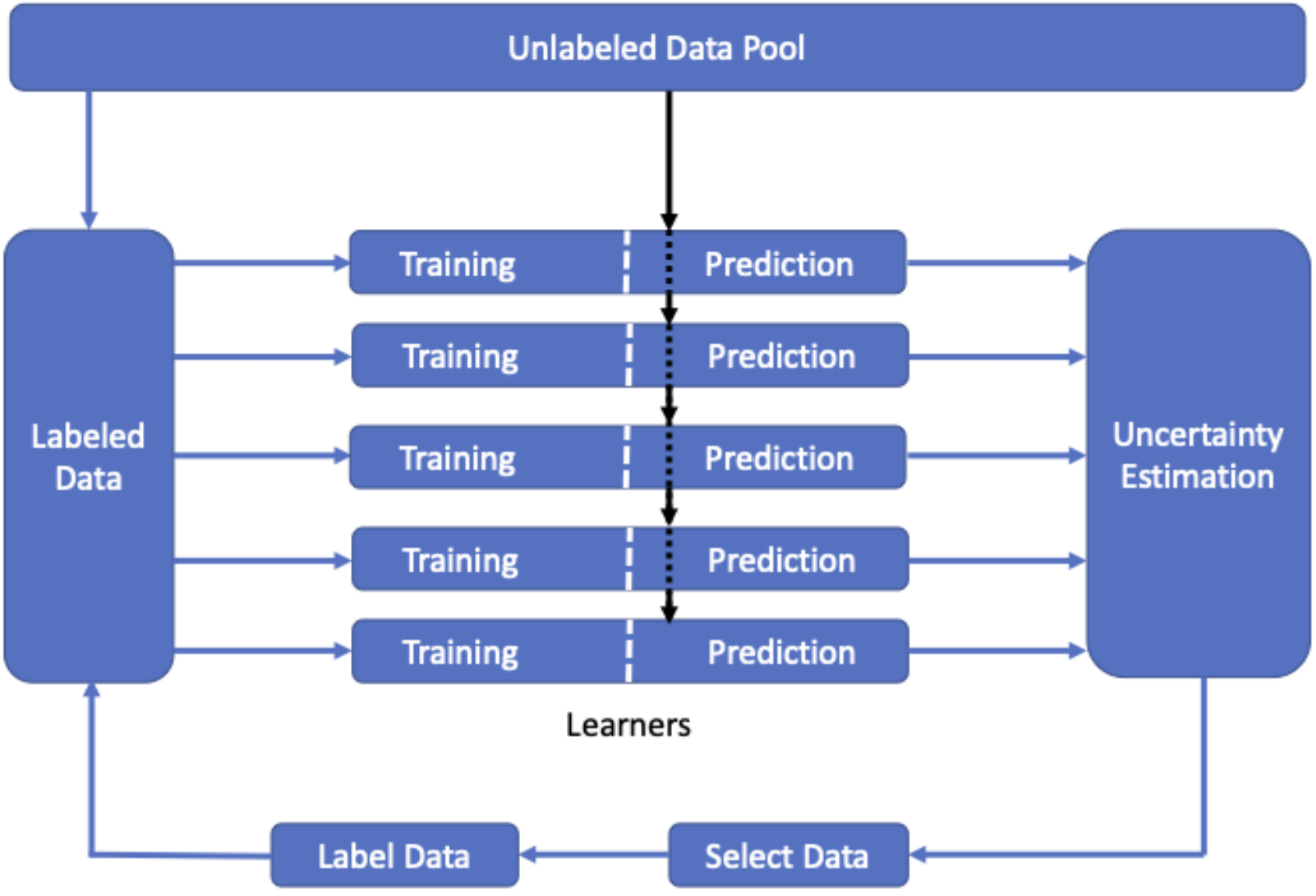
Variational Posterior

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



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

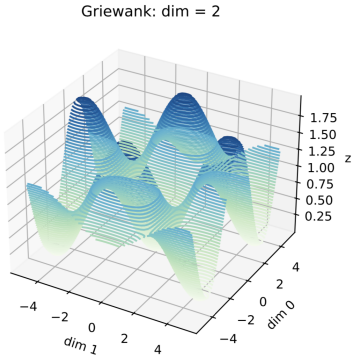
Query-by-Committee (QBC): algorithm sketch



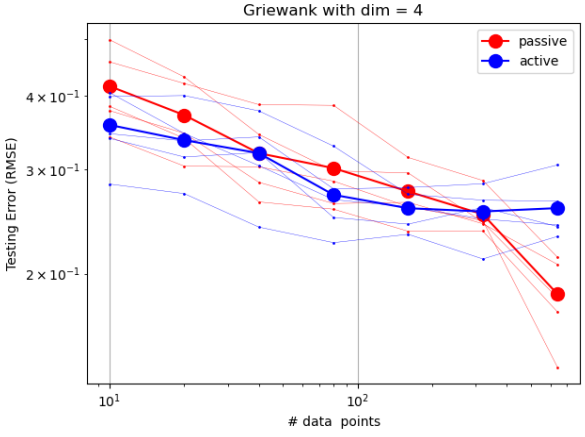
Query-by-Committee (QBC): algorithm outline

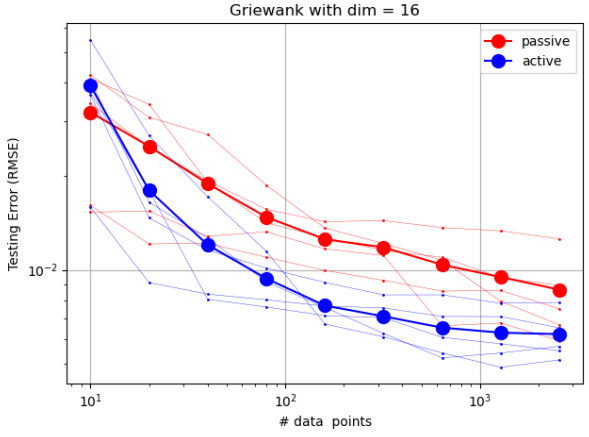
- Start with a large pool of P unlabeled points
- Select a training set of N points from the pool
- Launch K learners, each with fN randomly-chosen training points
  - Random sampling with replacement
  - Selection of fraction f determines data size per learner
    - diversity vs data size tradeoff
- Evaluate the learners' performance at all points in the pool
- Select M points from the pool, having highest 'disagreement', & add them to the training set
  - M choice, size of batch added per query, low error vs optimal choice
  - K-means clustering to discover geometry of selected data
  - Distribute data from clusters evenly among learners
    - Add fM points per learner with replacement
- Re-train, and repeat query to evaluate learners performance on prediction of unlabeled data in pool

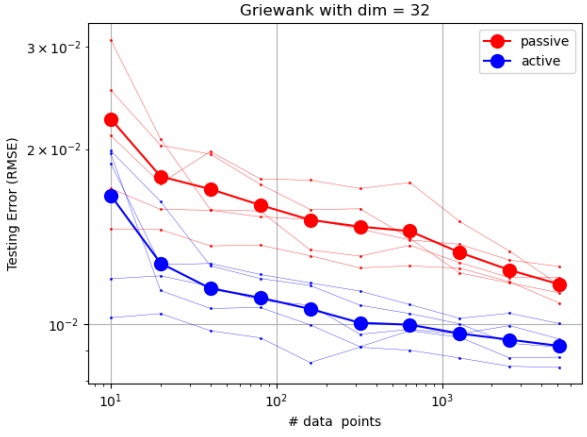
QBC: Griewank test function



$$f(\boldsymbol{x}) = 1 + \sum_{i=1}^{d} \frac{x_i^2}{4000} - \prod_{i=1}^{d} \cos\left(\frac{x_i}{\sqrt{i}}\right)$$

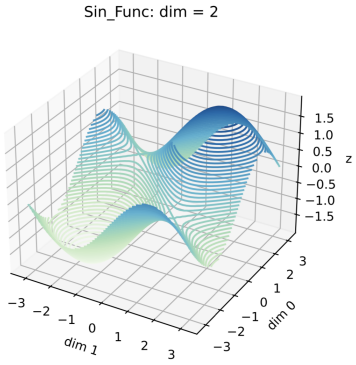




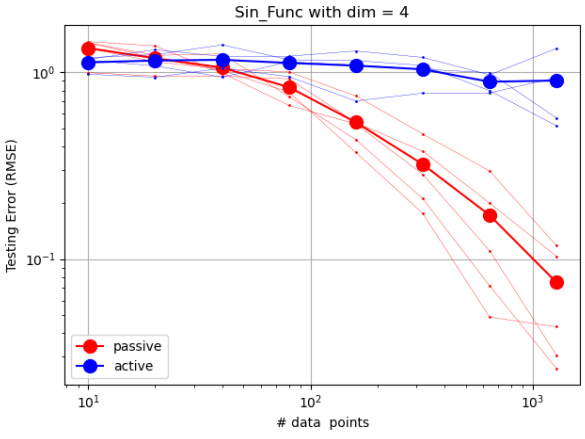


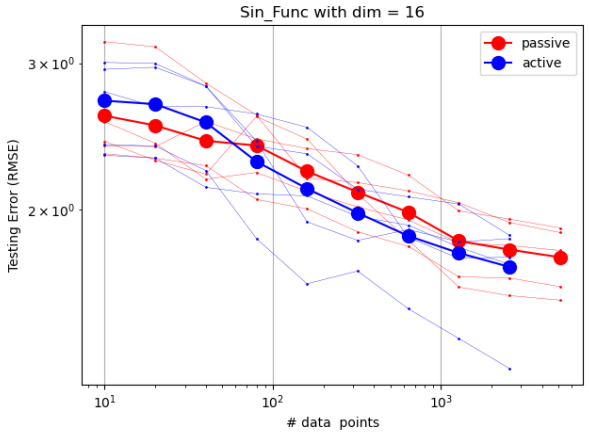
- Efficiency of active learning improves with higher dimension.

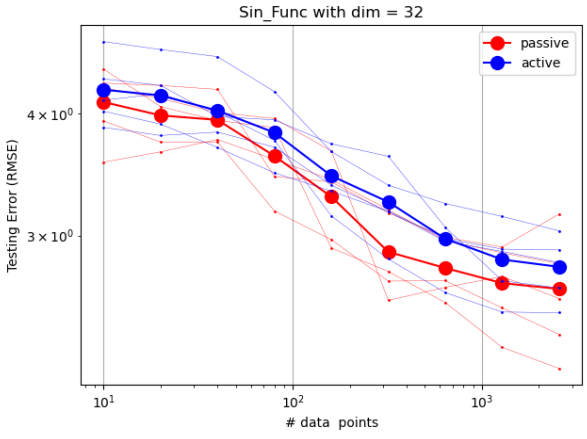
QBC: Sine test function



$$f(\boldsymbol{x}) = \sin\left(\sum_{i=1}^{d} x_i\right)$$

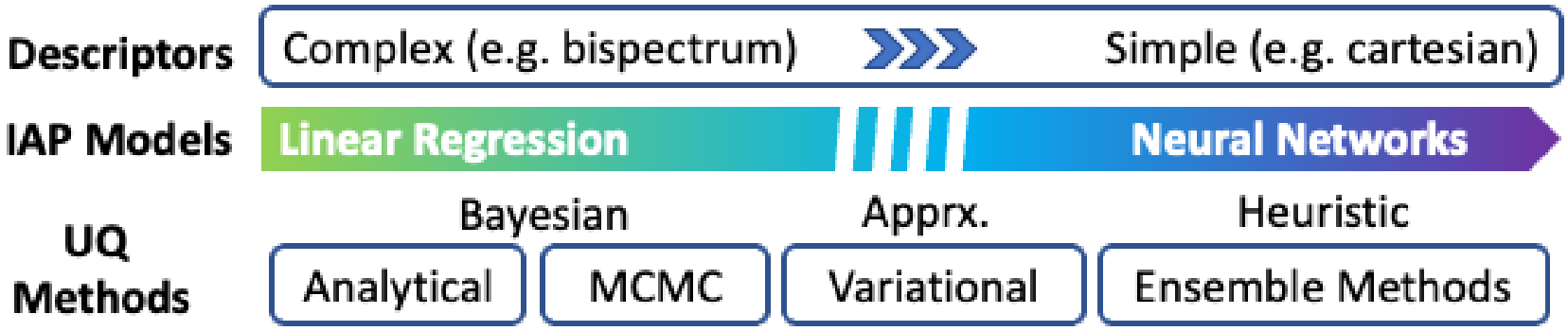




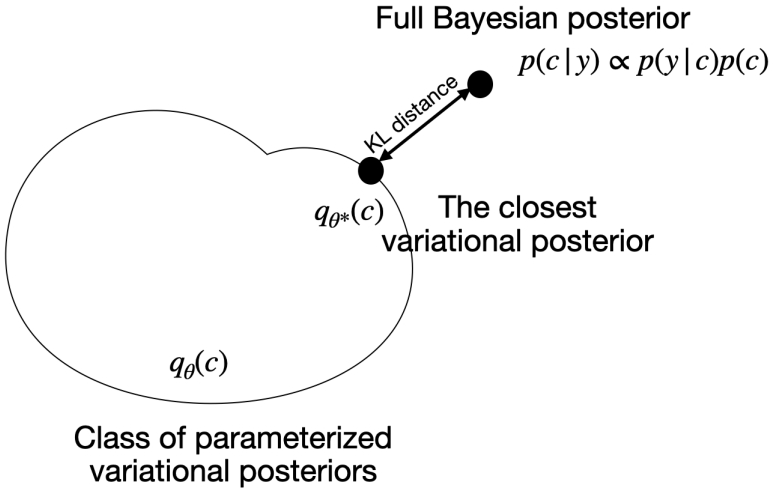


- In low-d, large pool size causes newly selected points to cluster.
- Potential solution: sample according to PDF $e^{-std(x)}$ to concentrate new points near high uncertainty region, but select elsewhere, too.

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.

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

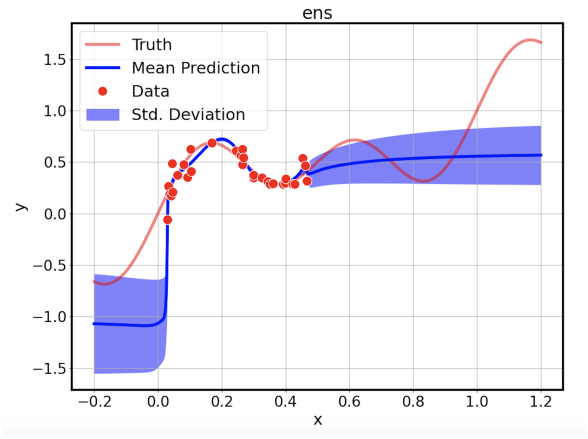
```
nn_mcmc = MCMCRegr(torch.nn.module)
```

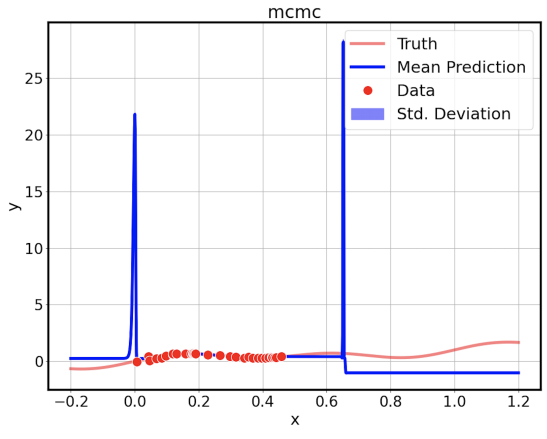
```
class MCMCRegr():
    def __init__(self, nnmodule, verbose=True):
        self.nnmodule = nnmodule
        self.verbose = verbose
```

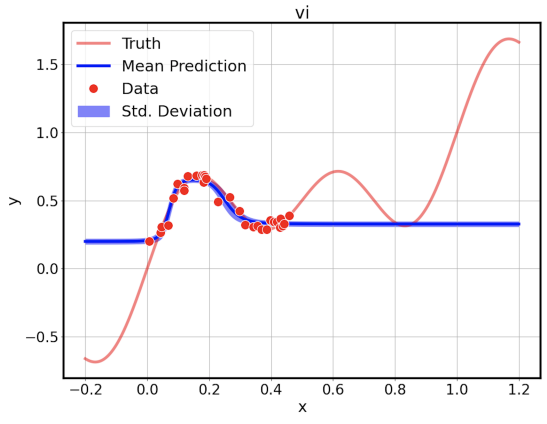
Option 3: NN learning with VI

```
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

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

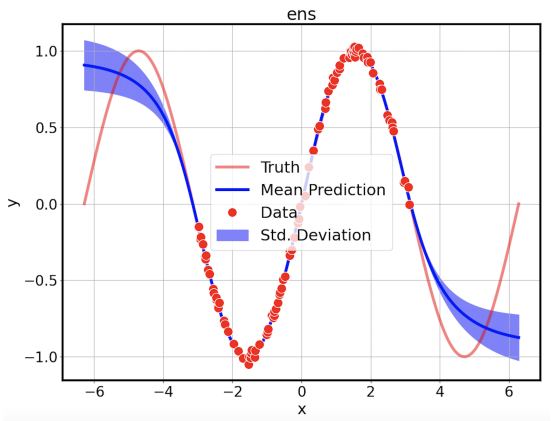
```
nn_mcmc = MCMCRegr(torch.nn.module)
```

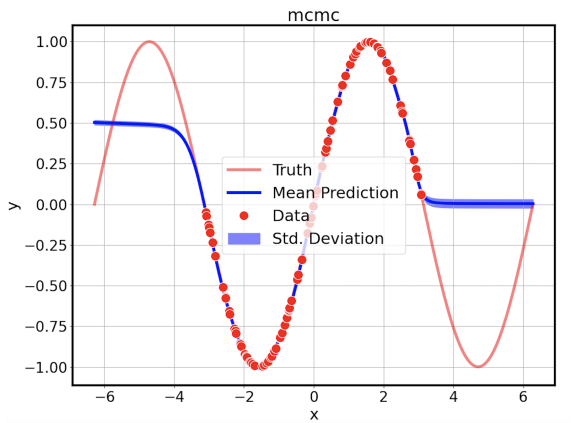
```
class MCMCRegr():
    def __init__(self, nnmodule, verbose=True):
        self.nnmodule = nnmodule
        self.verbose = verbose
```

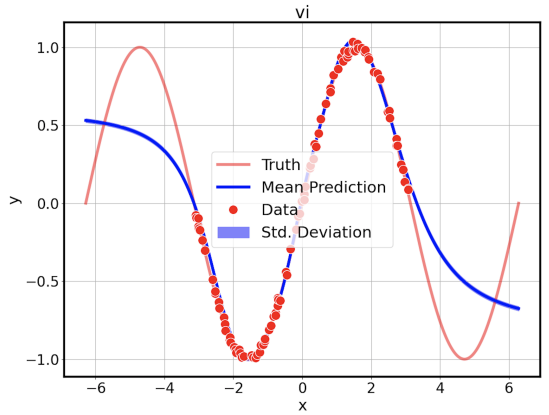
Option 3: NN learning with VI

```
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

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.