05/15/2022

Active learning for computational simulations: Application to TRISO fuel failure analysis

Som L. Dhulipala
Computational Scientist in Uncertainty Quantification
Computational Mechanics and Materials, INL

Workshop on scientific machine learning for nuclear engineering applications

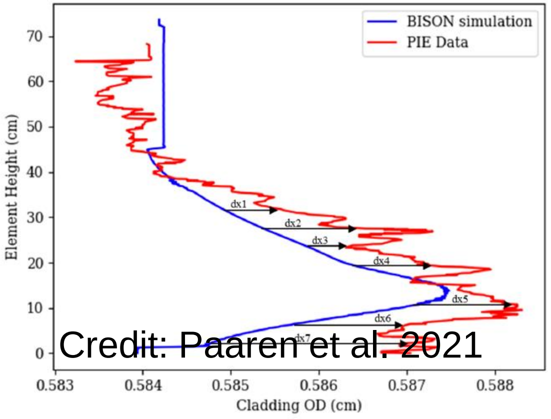
Collaborators: Ben Spencer, Wen Jiang, Zach Prince, Vincent Laboure, Jason Hales, Chandu Bolisetti, Yifeng Che, Peter German, Mike Shields (Johns Hopkins), Promit Chakroborty (Johns Hopkins)

Johns Hopkins University

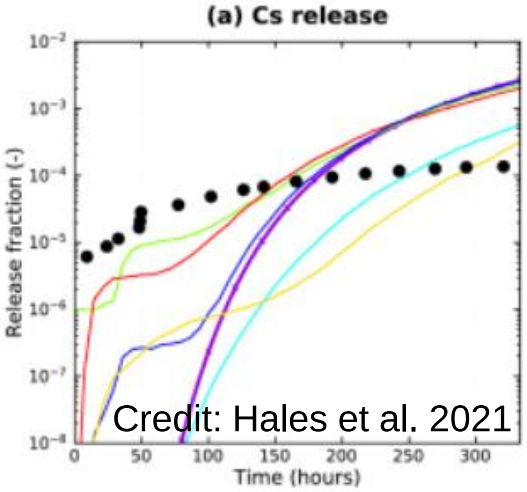
INL Idaho National Laboratory

Motivation: Computational Modeling and Simulations

Forward modeling

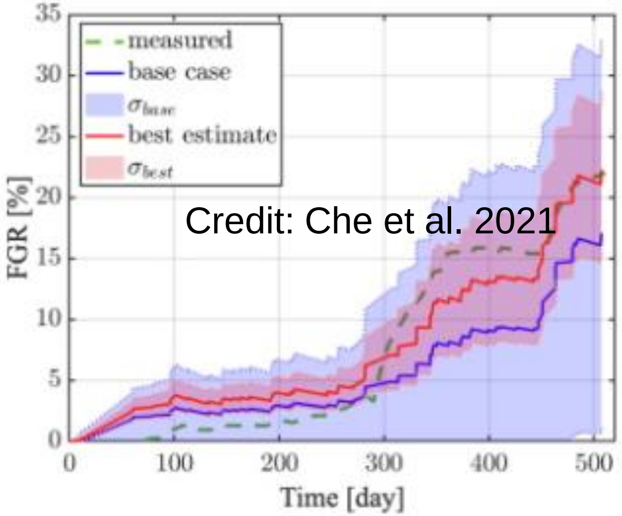


Credit: Paaren et al. 2021



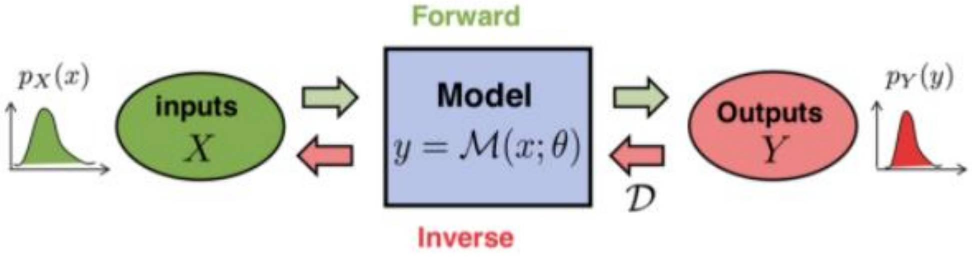
Credit: Hales et al. 2021

Inverse modeling



Credit: Che et al. 2021

Coupled modeling



Credit: Zhang 2020

Probabilistic ML: UQ + ML

- Data
  - High quality, small amount
  - Low quality, large (or small) amount

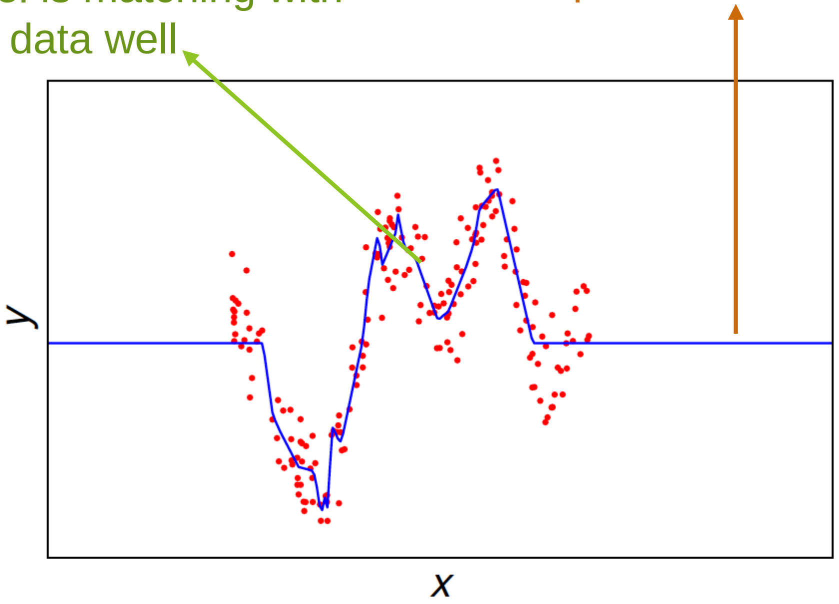
- Computational simulations are inherently inaccurate (Hendrickson 2020 DOE ASCR@40)
  - Modeling uncertainties (known knowns)
  - Epistemic uncertainties (known unknowns)
  - Aleatoric uncertainties (unknown unknowns)

- Increased complexity with the use of ML to accelerate modeling and simulations

- **UQ critical to assess the reliability of model predictions**

UQ ML

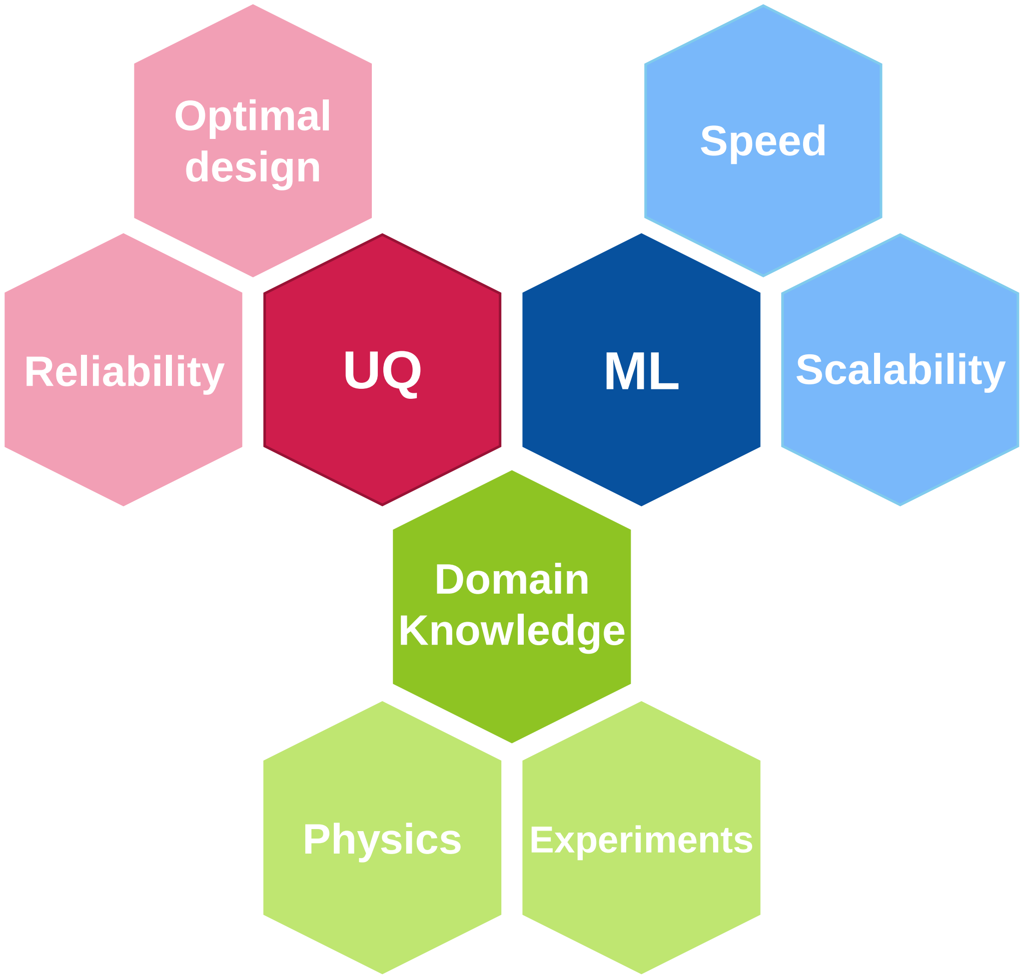
- All models are wrong, but models that know when they're wrong are useful (Lakshminarayanan et al. 2021)



(Credit: Ober et al. 2021)

Outline

- **Prediction**
  - Deterministic and Bayesian predictions
  - A simple Bayesian surrogate: Gaussian Process
  - Beyond Gaussian Processes
- **Inference**
  - Sampling
  - Markov Chain Monte Carlo
  - Beyond MCMC: Hybrid MCMC and sampling as optimization
- **Active learning and Multifidelity modeling**
- **TRISO nuclear fuel failure analysis**
- **Ongoing work:**
  - **MOOSE stochastic tools module**
  - **Monte Carlo with Hamiltonian Neural Nets**



Combo of the above three benefits computational tasks

Prediction

- Prediction problem:

$$\boldsymbol{y} = f(\boldsymbol{x};\, \boldsymbol{\theta})$$

$$\boldsymbol{\theta}^* = Argmin_{\boldsymbol{\theta}}\, \boldsymbol{L} \qquad [= Argmax_{\boldsymbol{\theta}}\, p(\boldsymbol{y}|\boldsymbol{x};\, \boldsymbol{\theta})]$$

- **Blackbox prediction**

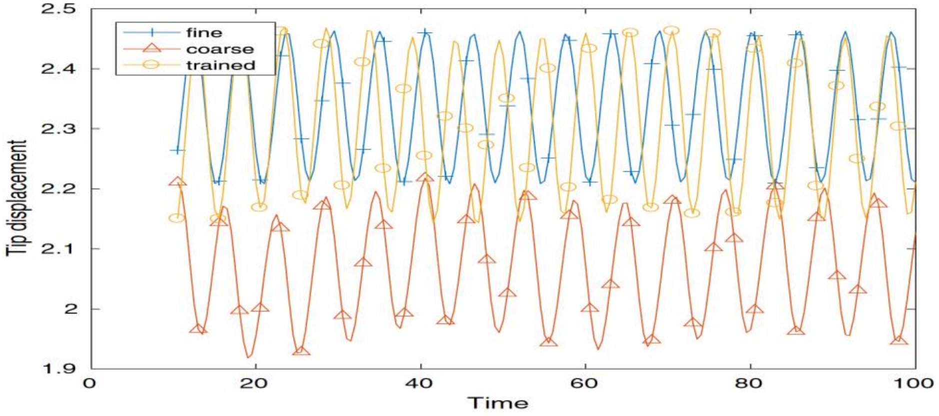
- **Auxiliary mesh refinement:** Baiges et al. 2019 Neural network correction term in linear algebraic equations

- **Constitutive relations:** Wang et al. 2019 Reinforcement Learning to combine phenomenological and data-driven relations

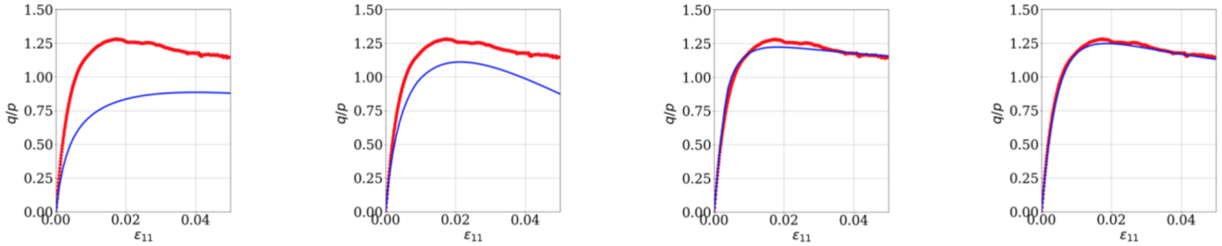
- **Physics in loss function:** Raissi et al. 2019 With training set loss, incorporate differential eq loss and IC/BC loss (Perdikaris 2020, LLNL seminar)

- **Eighty Years of FEM by Liu, Li, and Park 2021**

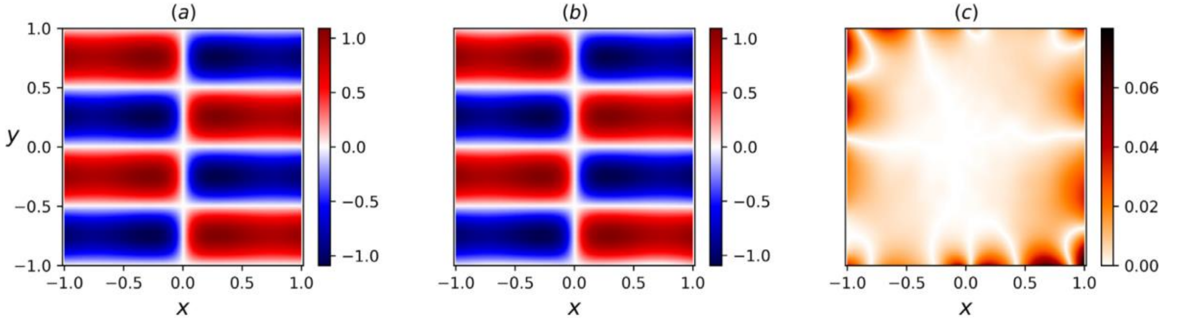
Fluid-structure interaction Credit: Baiges et al. 2019



Soil constitutive behavior prediction Credit: Wang et al. 2019

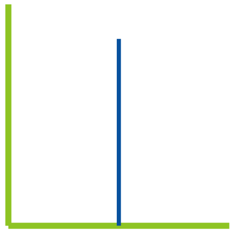


Heat eqn. Credit: Haghighat and Juanes 2019



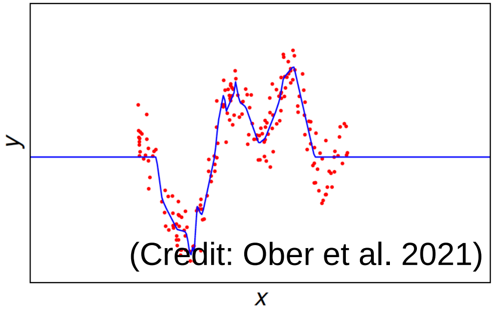
Deterministic and Bayesian predictions

- Define $\boldsymbol{y} = f(\boldsymbol{x};\ \boldsymbol{\theta})$
- Solve $\boldsymbol{\theta}^* = Argmax_{\boldsymbol{\theta}}\, p(\boldsymbol{y}|\boldsymbol{x};\ \boldsymbol{\theta})$



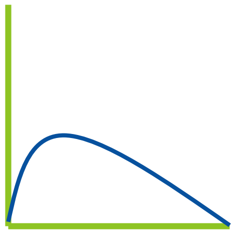
θ^*

- Predict $\boldsymbol{y} = f(\boldsymbol{x}; \boldsymbol{\theta}^*)$



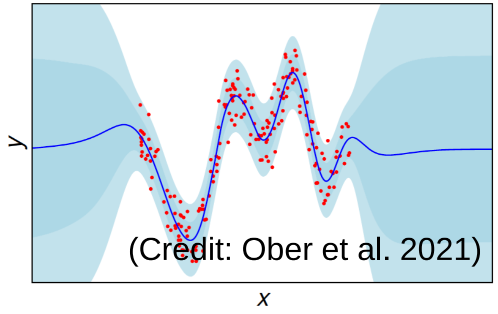
(Credit: Ober et al. 2021)

- Define $\boldsymbol{y} = f(\boldsymbol{x};\ \boldsymbol{\theta})$
- Solve $p(\boldsymbol{\theta}|\boldsymbol{x}, \boldsymbol{y}) \propto p(\boldsymbol{y}|\boldsymbol{x};\ \boldsymbol{\theta})\, p(\boldsymbol{\theta})$ [Hoff 2009]



θ

- Predict $p(\boldsymbol{y}|\boldsymbol{x}) = \int p(\boldsymbol{y}|\boldsymbol{x};\ \boldsymbol{\theta})\, p(\boldsymbol{\theta}|\boldsymbol{x}, \boldsymbol{y})\ d\theta$ [Do 2008]

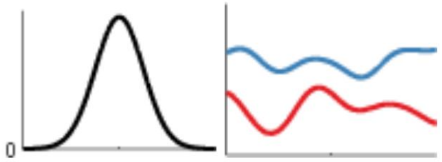


(Credit: Ober et al. 2021)

A simple Bayesian surrogate: Gaussian Process

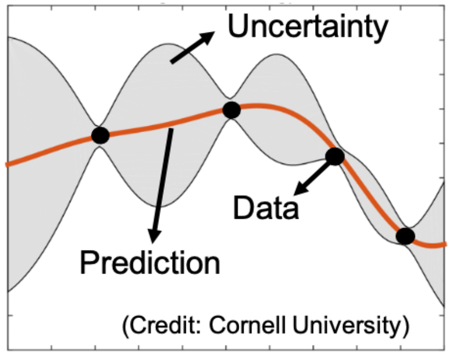
- Probabilities over functions: $f(\boldsymbol{X}) \sim \mathcal{N}(m(\boldsymbol{X}), k(\boldsymbol{X}, \boldsymbol{X}'))$
- Predictive distribution:

$$p(\boldsymbol{y}_* \mid \boldsymbol{X}, \boldsymbol{X}_*, \boldsymbol{y}) \sim \mathcal{N}\Big(k(\boldsymbol{X}_*, \boldsymbol{X})\, k(\boldsymbol{X}, \boldsymbol{X})^{-1}\, \boldsymbol{y},\; k(\boldsymbol{X}_*, \boldsymbol{X}_*) - k(\boldsymbol{X}_*, \boldsymbol{X})\, k(\boldsymbol{X}, \boldsymbol{X})^{-1}\, k(\boldsymbol{X}, \boldsymbol{X}_*) \Big)$$



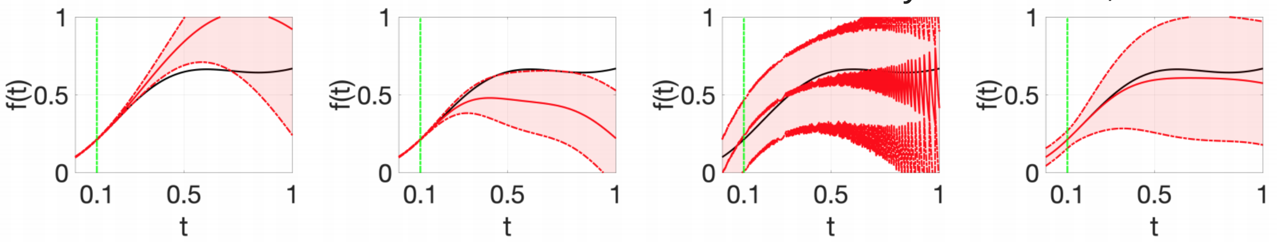
$$k_{\mathrm{SE}}(x, x') = \sigma^2 \exp\left(-\frac{(x-x')^2}{2\ell^2}\right)$$

(Kernel Cookbook by Duvenaud)



- Closed form solution; kernel params optimized using SGD
- SE kernel: Universal approximator [Micchelli et al. 2006]
- Robust UQ

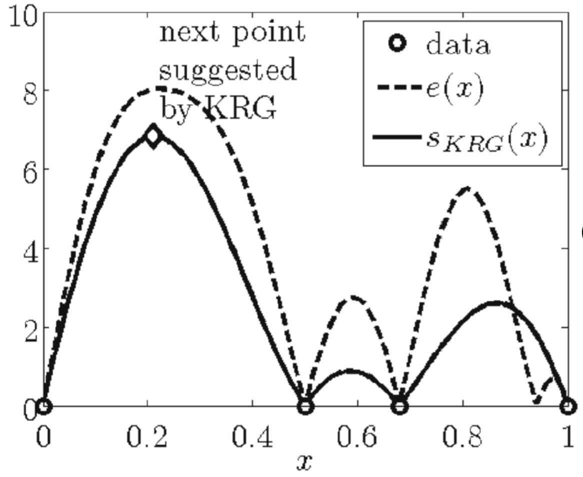
1st order ODE Credit: Anonymous 2021, ICLR



(a) SKL (b) DKL (c) LFM (d) PI-DKL

- **Physics in Gaussian Process:** Anonymous 2021, ICLR Physics informed neural network embedded in GP kernel (technically called deep kernel learning)

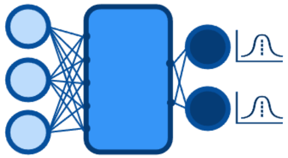
- **Optimal design:** Viana et al. 2021 Gaussian process UQ estimate tells the next best training point



Credit: Viana et al. 2021

Beyond Gaussian Processes

- GP limitations: Excessive smoothing, high-dimensional data, Complexity O(n^3)
- BNN: Bayesian Neural Network

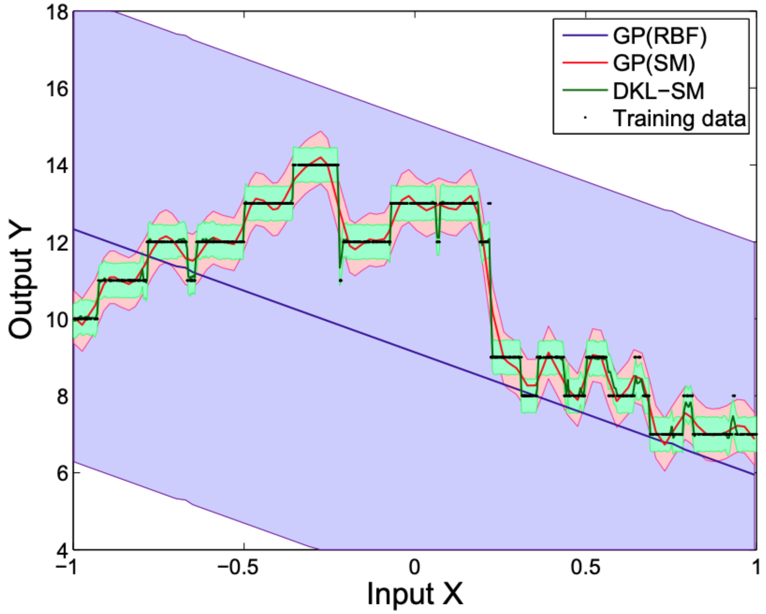


- Bayesian surrogates:

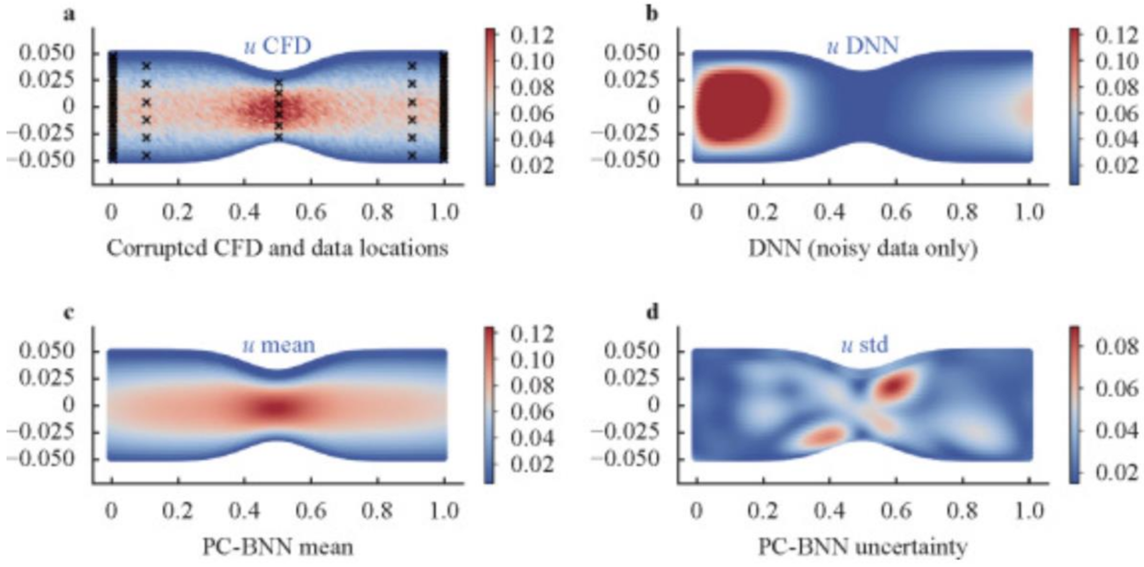
A spectrum of Bayesian predictive models* [Dhulipala et al. 2021]

- **Navier-Stokes embedded BNN:** Sun and Wang 2020 Flow reconstruction from sparse and noisy data

Step function Credit: Wilson et al. 2016



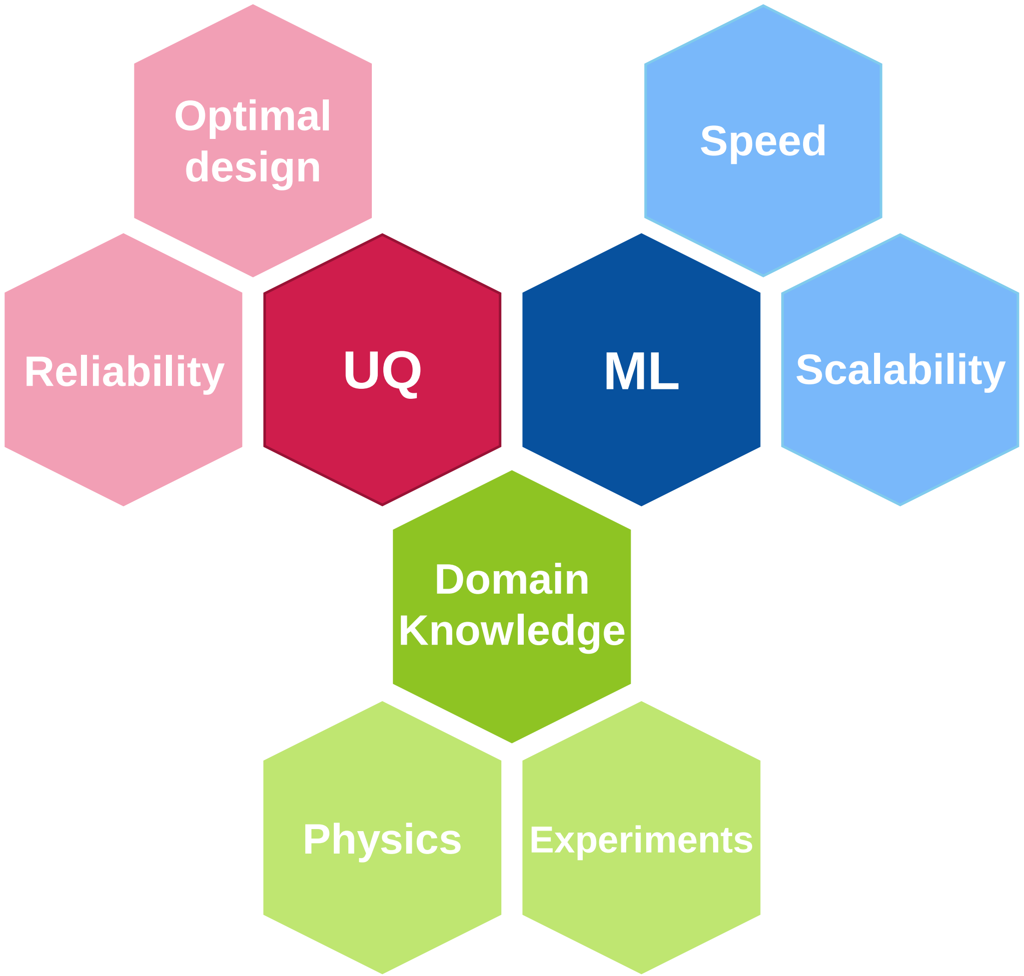
BNN with Navier-Stokes Credit: Sun and Wang 2020



*This is incomplete and for illustration only

Outline

- **Prediction**
  - Deterministic and Bayesian predictions
  - A simple Bayesian surrogate: Gaussian Process
  - Beyond Gaussian Processes
- **Inference**
  - Sampling
  - Markov Chain Monte Carlo
  - Beyond MCMC: Hybrid MCMC and sampling as optimization
- **Active learning and Multifidelity modeling**
- **TRISO nuclear fuel failure analysis**
- **Ongoing work:**
  - **MOOSE stochastic tools module**
  - **Monte Carlo with Hamiltonian Neural Nets**



Combo of the above three benefits computational tasks

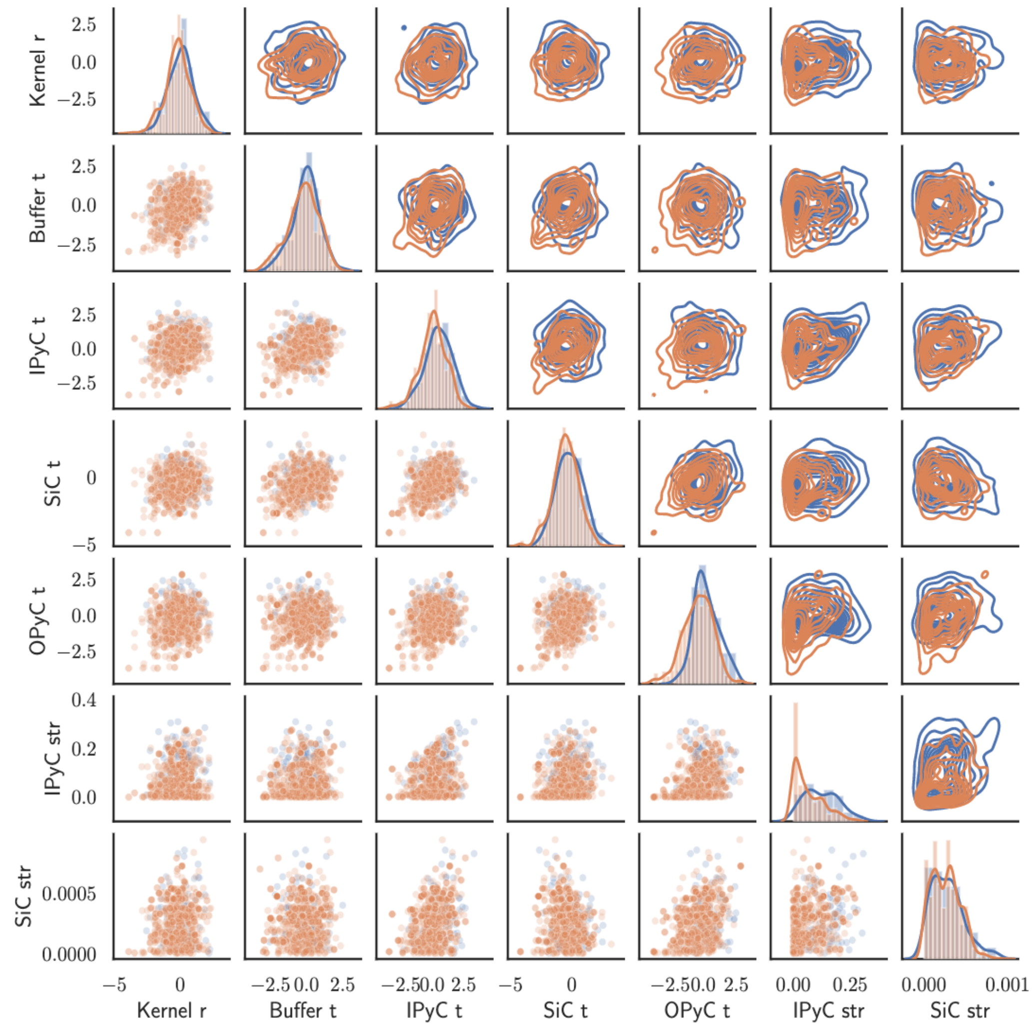
IDAHO NATIONAL LABORATORY

Inference: Sampling

- Probability distributions of input parameters that cause TRISO particle failure (Jiang et al. 2021)
- Update BISON fission gas release models given experimental data (Che et al. 2021)
- Parameter distributions of a Bayesian Neural Network

$$f(\boldsymbol{\theta}|\boldsymbol{Data})$$

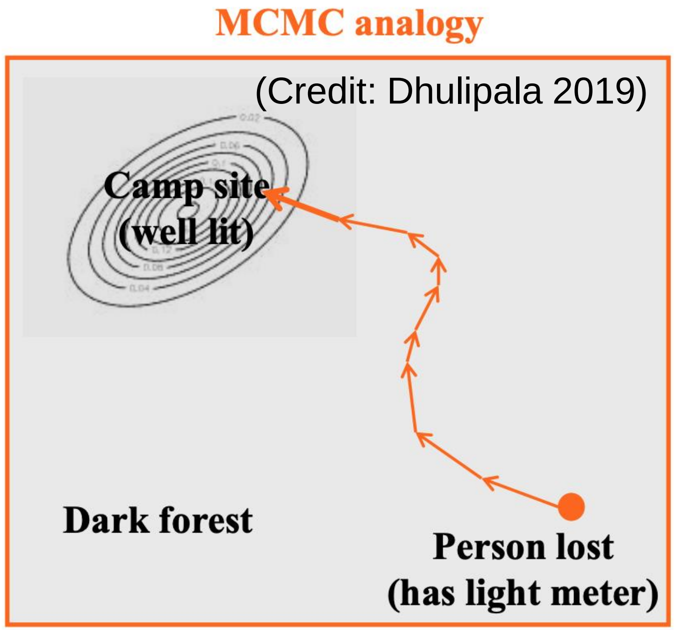
- Standard methods like Monte Carlo or Latin Hypercube very expensive or not applicable
- So, how do we sample efficiently from conditional distributions?

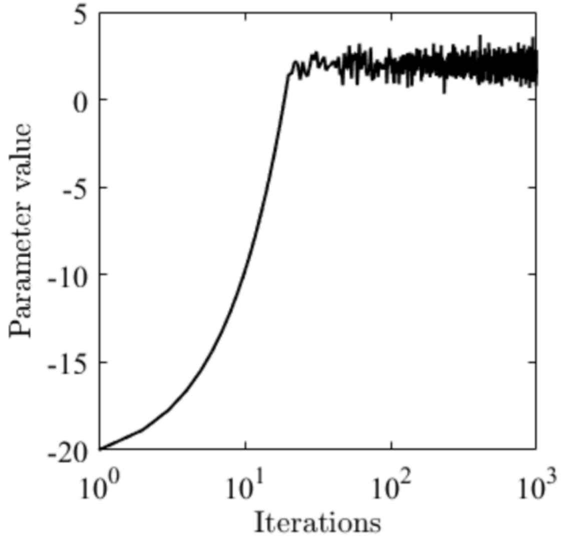


Distributions of input parameters causing TRISO particle failures

Markov Chain Monte Carlo

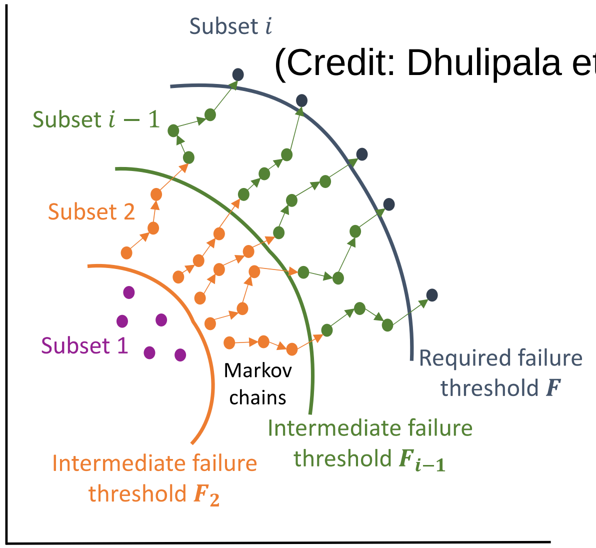
- Sample efficiently from conditional distributions $f(\boldsymbol{\theta}|\boldsymbol{Data})$
- **Dark forest:** Parameter space
- **Well lit camp site:** Required distribution to be sampled from
- **Light meter:** Acceptance ratio (or transition operator)
- Metropolis-Hastings: Popular MCMC algorithm (being implemented in MOOSE)
- **Does an MCMC algorithm always converge to the required distribution?**
  - Neal 1993 Detailed balance sufficient condition
  - Acceptance ratio satisfies detailed balance
- Variants of MCMC exist on how acceptance ratio designed



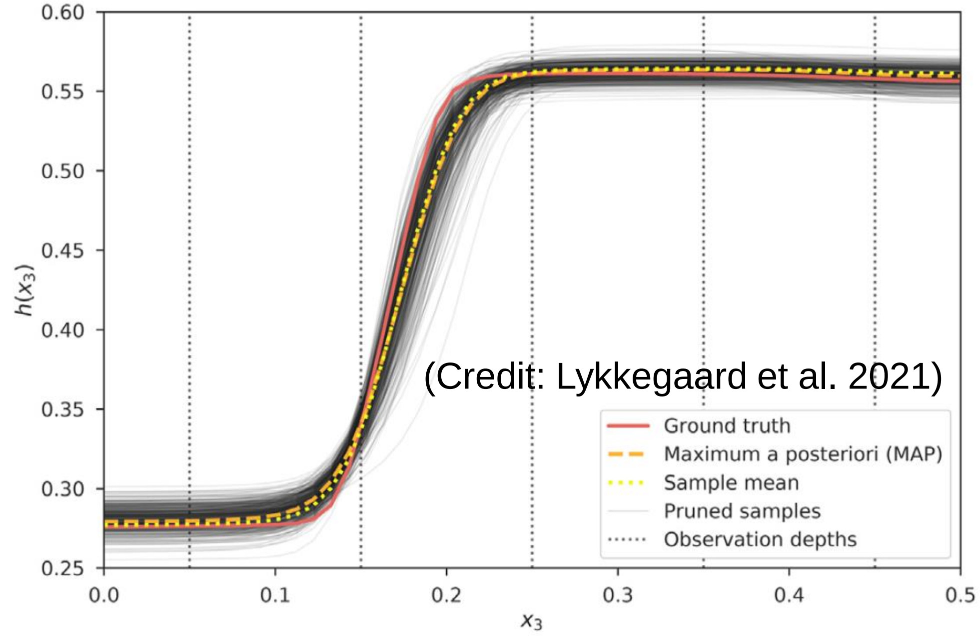


MCMC: Applications in the Computational Sciences

- **Rare events:** Dhulipala et al. 2021 Sample from parameter spaces that causes FE model to fail (Subset Simulation)
- **Inverse analysis:** Lykkegaard et al. 2021 Update porous flow model given ground water data
- **High-dimensional integration:** Mancang et al. 2011 Neutron transport equation using MCNP with MCMC techniques



(Credit: Dhulipala et al. 2021)



(Credit: Lykkegaard et al. 2021)

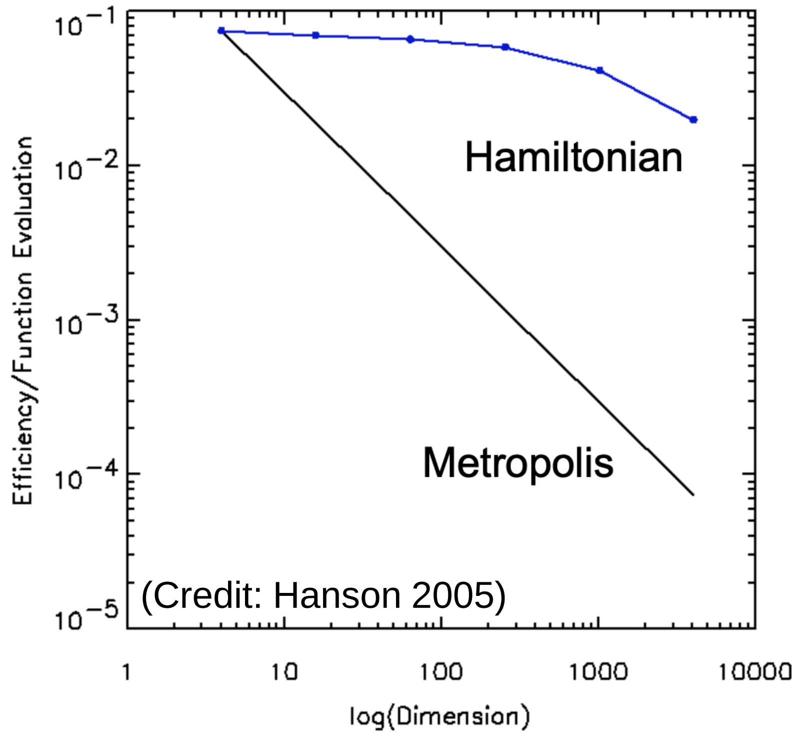
Beyond MCMC: Hybrid MCMC and sampling as optimization

- **MCMC limitations:** Poor high-dimensional scalability (convergence issues), many model evaluations required

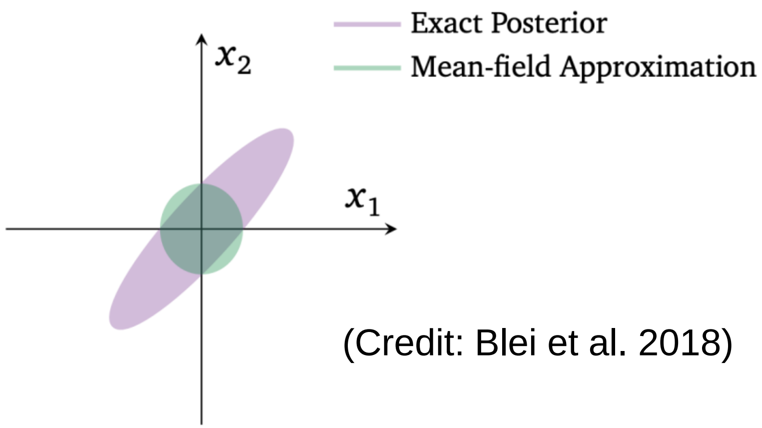
- **Hybrid MCMC (Hamiltonian Monte Carlo):** Neal 2011 Hamiltonian dynamics solved to propose the next sample. Very good scalability (Current “gold standard” for Bayesian Neural Networks)

$$z_1 = z_0 + \int_{t_0}^{t_1} \boldsymbol{I}\ \nabla H(\boldsymbol{z})\ dt$$

- **Optimization (approximate):** Blei et al. 2018 Variational inference transforms sampling to optimization. A variational family of distributions is assumed (e.g., exponential family). Distribution parameters are optimized



(Credit: Hanson 2005)



(Credit: Blei et al. 2018)

Outline

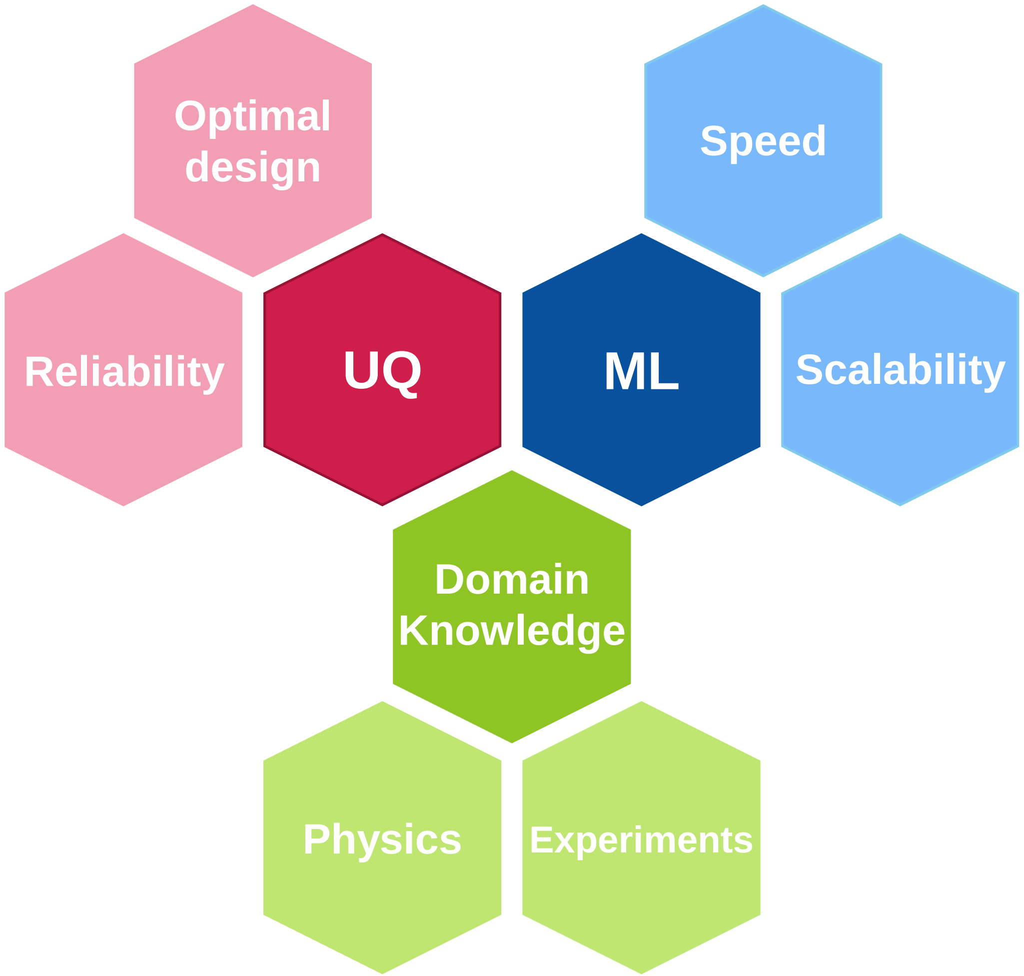
- **Prediction**
  - Deterministic and Bayesian predictions
  - A simple Bayesian surrogate: Gaussian Process
  - Beyond Gaussian Processes

- **Inference**
  - Sampling
  - Markov Chain Monte Carlo
  - Beyond MCMC: Hybrid MCMC and sampling as optimization

- **Active learning and Multifidelity modeling**

- **TRISO nuclear fuel failure analysis**

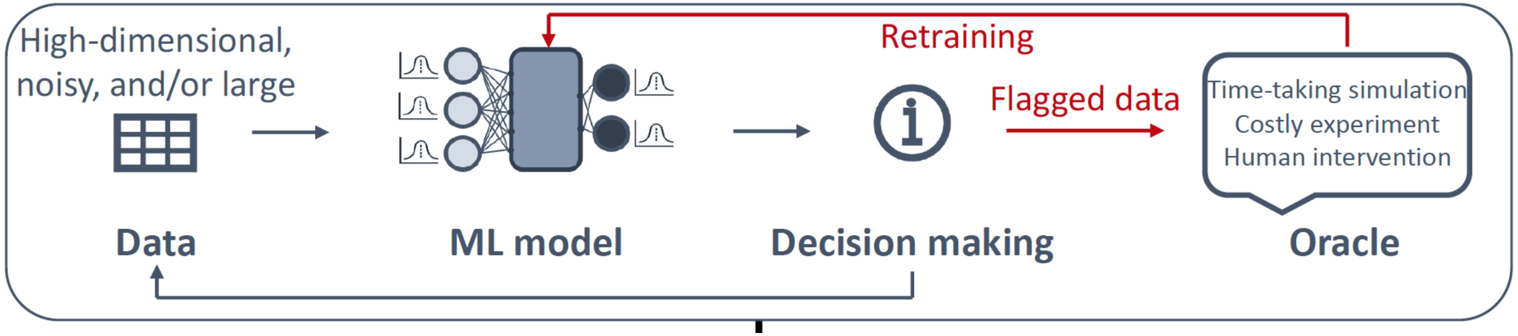
- **Ongoing work:**
  - **MOOSE stochastic tools module**
  - **Monte Carlo with Hamiltonian Neural Nets**



Combo of the above three benefits computational tasks

Active learning

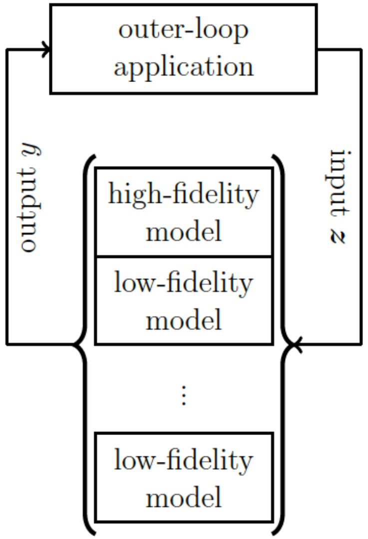
Principle of active learning (Bayesian ML model preferred)



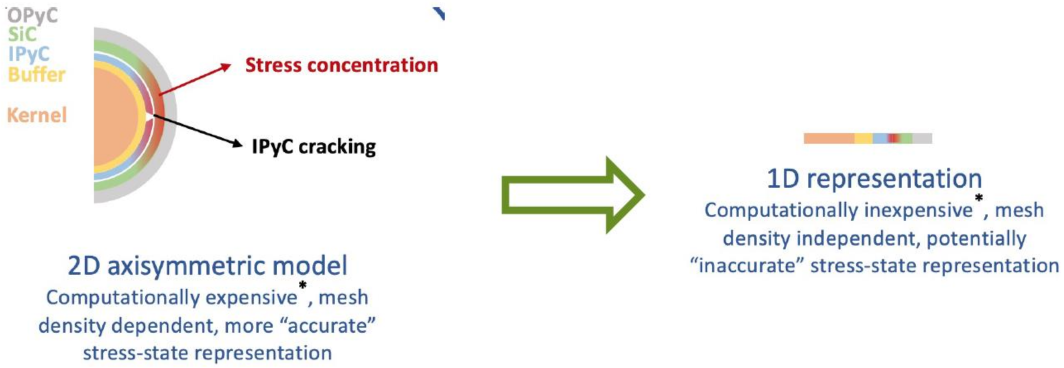
- ML model actively decides the next optimal training point
- Useful when dealing with expensive computational models or costly experiments as the ML model identifies the training point such that the information gain is optimized
- Probabilistic (Bayesian) ML preferred as it provides prediction uncertainty estimates---useful for designing learning functions

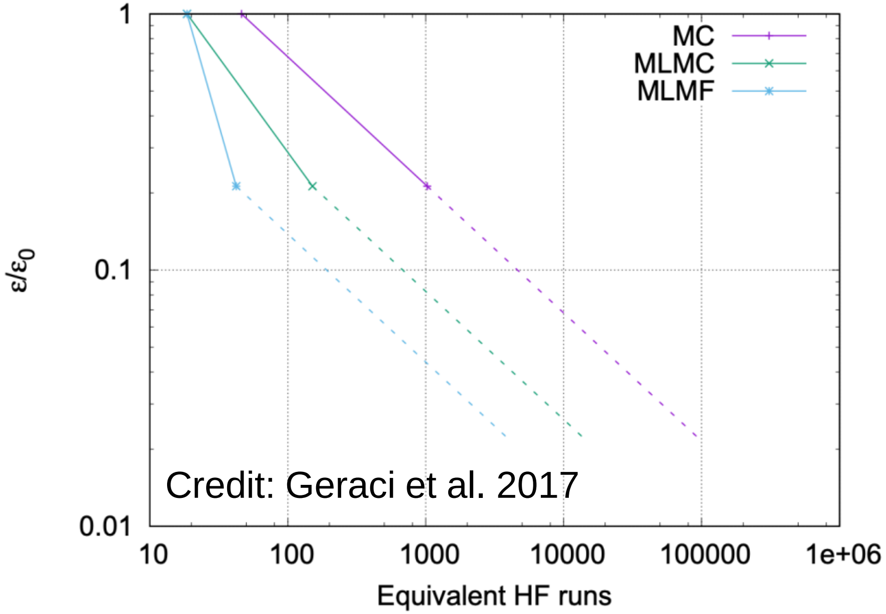
Multifidelity modeling

- TRISO model is a good example
- Pehertorfer et al. 2018 Multiple low-fidelity models can be considered
- Computational budget across multiple fidelity models constrained. Gorodetsky et al. 2020 approximate control variates framework
- Actively decide which modeling fidelity to call



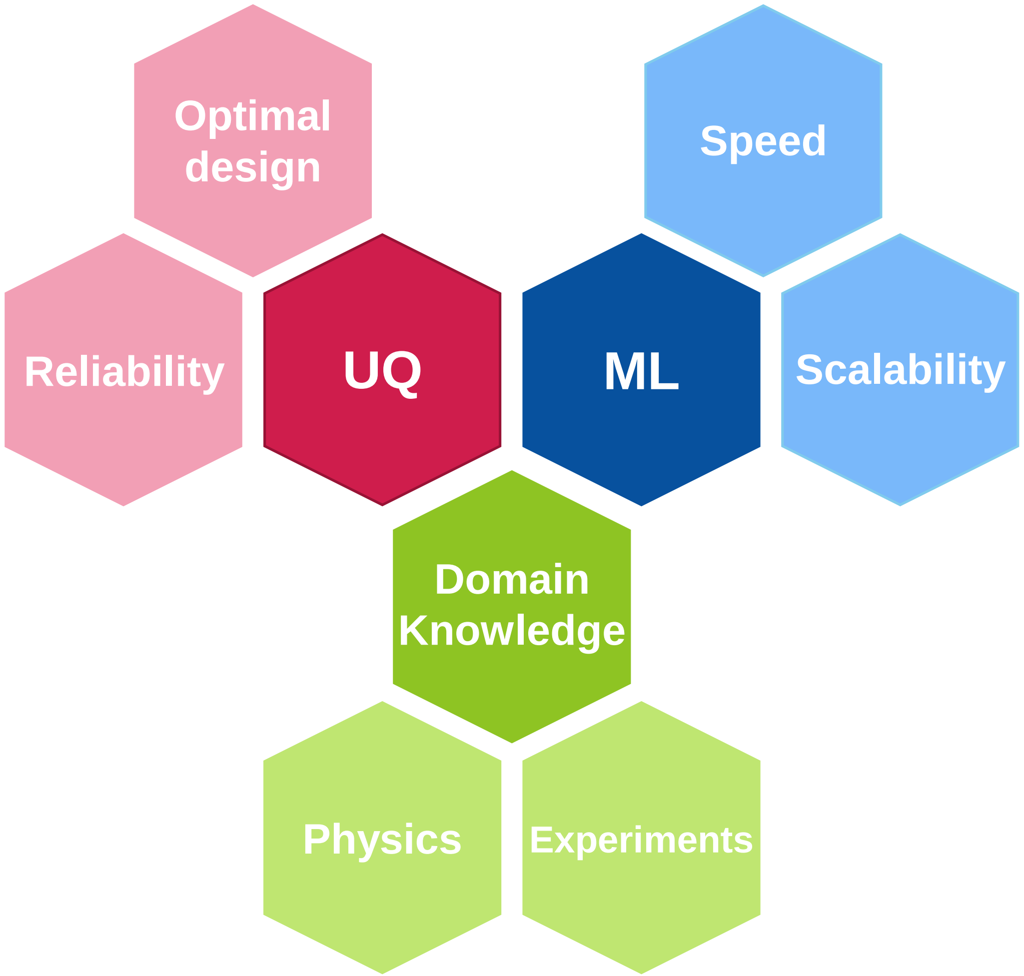
Credit: Pehertorfer et al. 2018





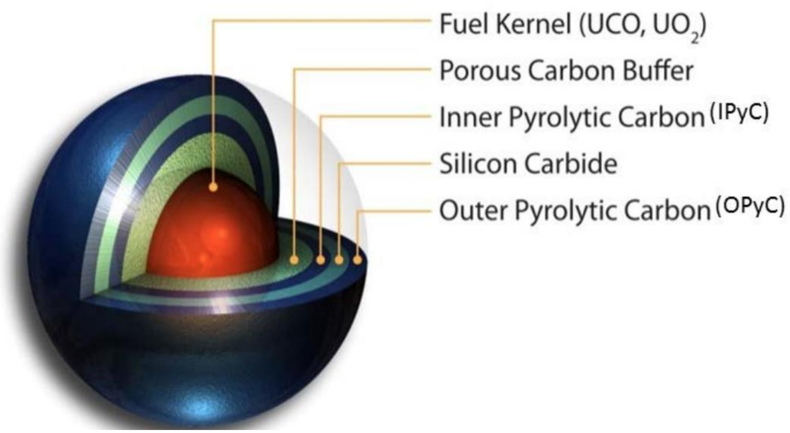
Outline

- **Prediction**
  - Deterministic and Bayesian predictions
  - A simple Bayesian surrogate: Gaussian Process
  - Beyond Gaussian Processes
- **Inference**
  - Sampling
  - Markov Chain Monte Carlo
  - Beyond MCMC: Hybrid MCMC and sampling as optimization
- **Active learning and Multifidelity modeling**
- **TRISO nuclear fuel failure analysis**
- **Ongoing work:**
  - **MOOSE stochastic tools module**
  - **Monte Carlo with Hamiltonian Neural Nets**

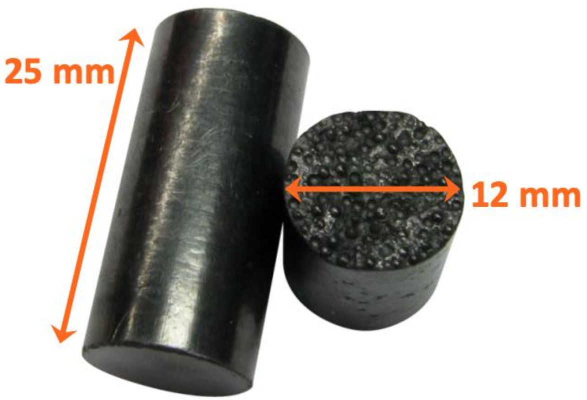


Combo of the above three benefits computational tasks

Motivation: TRISO, a robust nuclear fuel



Single TRISO particle of radius ~400 μm (Davenport 2016)



Fuel compact with numerous TRISO particles (Demkowicz 2016)

- TRISO stands for TRI-structural iSOtropic particle fuel
- Proposed for use in many advanced reactor concepts like micro-reactors owing to its robustness
- Interest from the DOE, DOD, and industries like Kairos Power, Xenergy
- Fuel kernel surrounded by several protective layers
- A fuel compact can have 1000s of tiny TRISO particles
- **Critical to analyze the failure rates of TRISO particle**: Impacts to reactor operation

Motivation: Expensive models, low failure rates

Heat

$$\rho\, C_p\, \frac{\partial T}{\partial t} + \nabla \cdot (-k \nabla T) - E_f\, \dot{F} = 0$$

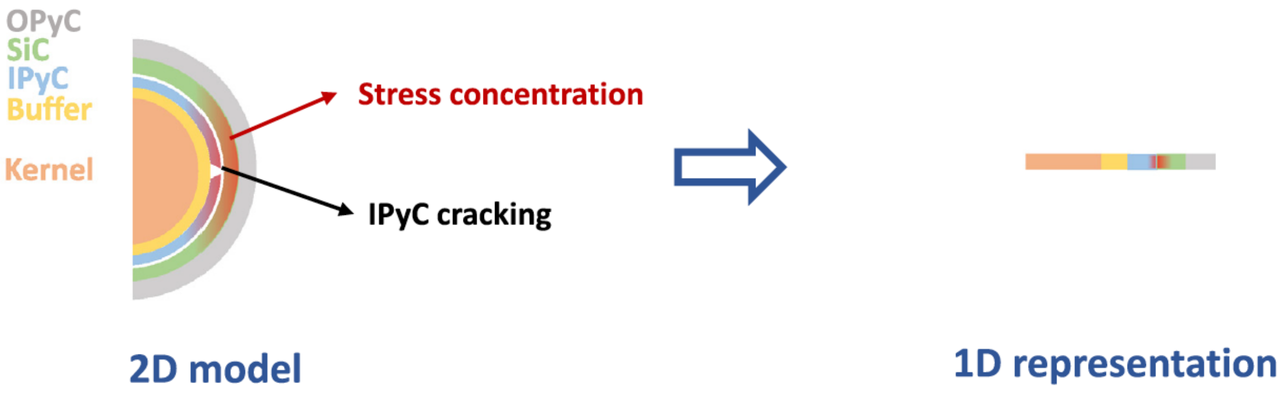
Momentum

$$\nabla \cdot \boldsymbol{\sigma} = 0$$

$$\boldsymbol{\sigma} = \mathcal{C} : (\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}_c - \boldsymbol{\varepsilon}_t - \boldsymbol{\varepsilon}_i)$$

MOOSE

Bison



(Jiang et al. 2021, Dhulipala et al. 2022)

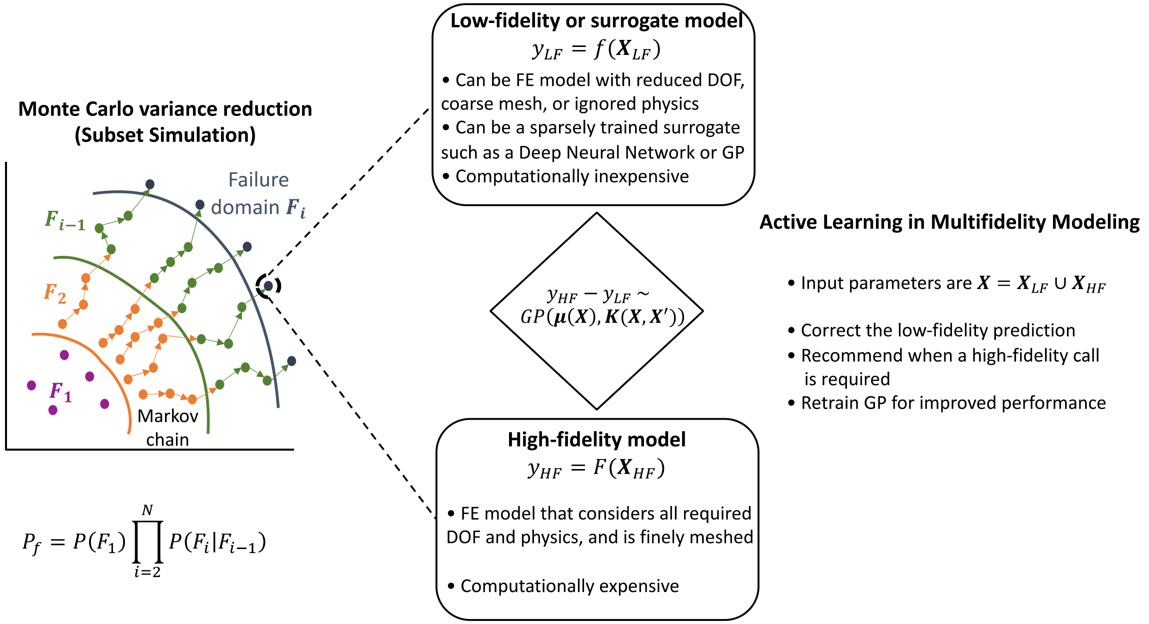
- Sophisticated material property relationships for the different protective layers in TRISO
- Numerically modeled using Bison fuel performance code based on MOOSE (Multiphysics Object Oriented Simulation Environment)
- **Failure mode:** SiC layer fracture most important. Caused by IPyC cracking induced stress conc.
- **1D model:** Fast (~ 11 seconds), approximates SiC stress conc. due to IPyC cracking
- **2D model:** Slow (~30 minutes), models SiC stress conc. using XFEM
- **Failure rates:** 1E-3 to 1E-7

Problem statement

$$P_f = \int_{\widetilde{F}(\boldsymbol{X})>\mathcal{F}} q(\boldsymbol{X})\, d\boldsymbol{X} \qquad P_f \approx \hat{P}_f = \frac{1}{N_m} \sum \mathbf{I}(\widetilde{F}(\boldsymbol{X}) > \mathcal{F})$$

(*F: Failure threshold; F(**X**): Model output; q(**X**): input distributions*)

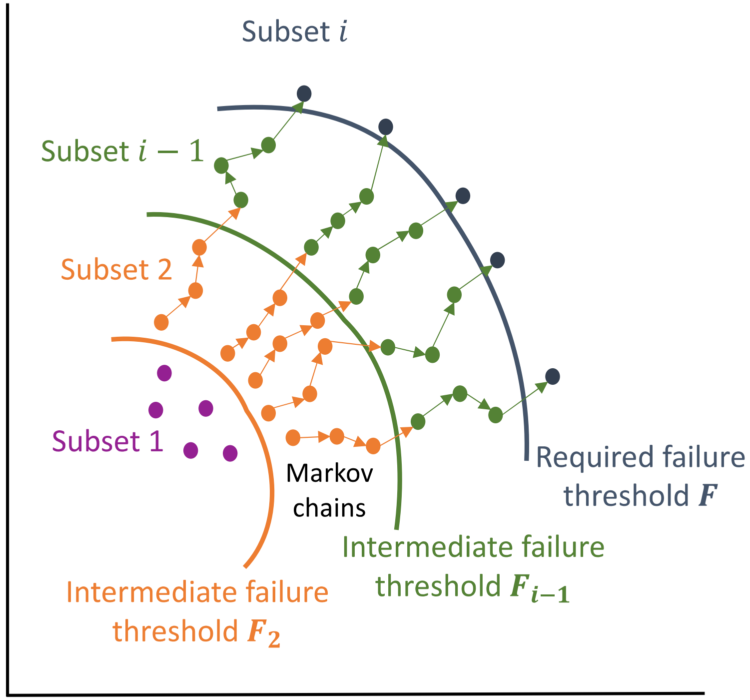
Proposed algorithm



Dhulipala et al. 2021

- Rare events estimation involves computing the multidimensional integral
- Monte Carlo and variance reduction methods require prohibitively large calls to the high-fidelity (HF) model
- **Multifidelity modeling:** Typically make assumptions about the modeling fidelities and/or require fixing the number of HF calls
- **Active learning:** Can breakdown for smaller failure probabilities (1E-4 or less) and/or require large number of Gaussian Process evaluations
- **Proposed: Active learning with multifidelity modeling**
  - Dynamically decides the HF calls
  - Flexibility over the LF model choice
  - Capable for Smaller failure probabilities
  - Doesn’t require large upfront GP evaluations

Background: Subset Simulation (variance reduction)

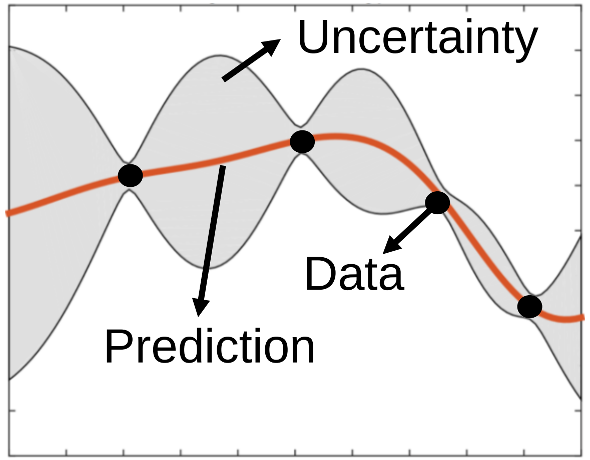


$$P_f = P(F_1)\prod_{i=2}^{N} P(F_i|F_{i-1})$$

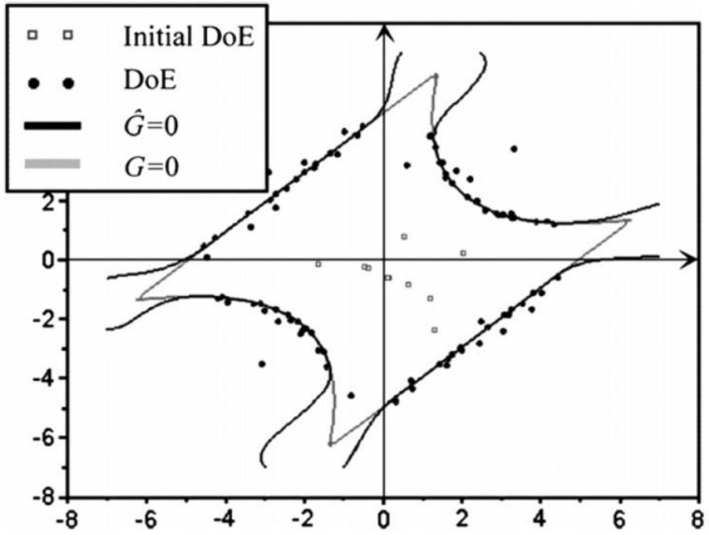
Proposed by Au and Beck (2001)

- Expresses small failure probabilities as a product of larger conditional probabilities (of the order 0.1)
- Creates intermediate failure thresholds before the required failure threshold
- An intermediate failure threshold is defined as the (1-x) percentile value of the samples in previous conditional level
- First conditional level: Direct Monte Carlo
- Subsequent conditional levels: Markov Chain Monte Carlo (Metropolis-Hastings or other variants)

Background: Active learning with Gaussian Process



(Credit: Cornell University)



(Echard et al. 2011)

- Posterior predictive distribution:

$$p(\boldsymbol{y}_* \mid \boldsymbol{X}, \boldsymbol{X}_*, \boldsymbol{y}) \sim \mathcal{N}\Big(k(\boldsymbol{X}_*, \boldsymbol{X})\, k(\boldsymbol{X}, \boldsymbol{X})^{-1}\, \boldsymbol{y},\; k(\boldsymbol{X}_*, \boldsymbol{X}_*) - k(\boldsymbol{X}_*, \boldsymbol{X})\, k(\boldsymbol{X}, \boldsymbol{X})^{-1}\, k(\boldsymbol{X}, \boldsymbol{X}_*) \Big)$$

- Both mean prediction and uncertainty quantification (quite robust under small training data)
- UQ enables the formulation of active learning functions: U-function (Echard et al. 2011), Expected Feasibility Function (Bichon et al. 2008)
- Active learning function decides when to call high-fidelity (HF) model in Monte Carlo schemes

Multifidelity active learning with Gaussian Process

Traditional U-function

GP prediction

Required failure threshold

$$U = \frac{|\mu_G(X) - F|}{\sigma_G(X)}$$

GP standard deviation

Multifidelity U-function

$y_{HF} = F(\boldsymbol{X}_{HF})$ High-fidelity model output

$y_{LF} = f(\boldsymbol{X}_{LF})$ Low-fidelity model output

$$\boldsymbol{X} = \boldsymbol{X}_{HF} \cup \boldsymbol{X}_{LF}$$

LF prediction

GP correction

$$U = \frac{|f(X_{LF}) + \bar{\epsilon}(X) - F|}{\sigma_\epsilon(X)}$$

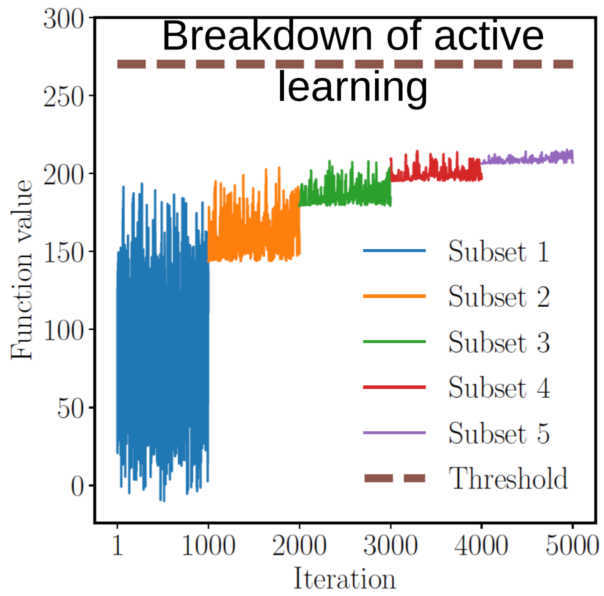
- Traditional active learning functions rely on Gaussian Process predictions entirely
- Performance of active learning schemes can be improved using multifidelity modeling
- U-function is extended to a multifidelity modeling setting owing to its simplicity
- A GP learns the differences between high-fidelity (HF) and low-fidelity (LF) predictions
- GP corrects the LF predictions for every test sample
- New multifidelity U-function to decide when to call the HF model

Coupled multifidelity active learning and Subset Simulation

Subset independent multifidelity U-function

Required failure threshold

$$U = \frac{|f(X_{LF}) + \bar{\epsilon}(X) - F|}{\sigma_\epsilon(X)}$$



Subset dependent multifidelity U-functions

Intermediate failure threshold

$$U^{MF}{}_s = \frac{|f(X_{LF}) + \bar{\epsilon}(X) - F_s|}{\sigma_\epsilon(X)}$$

(Intermediate conditional levels)

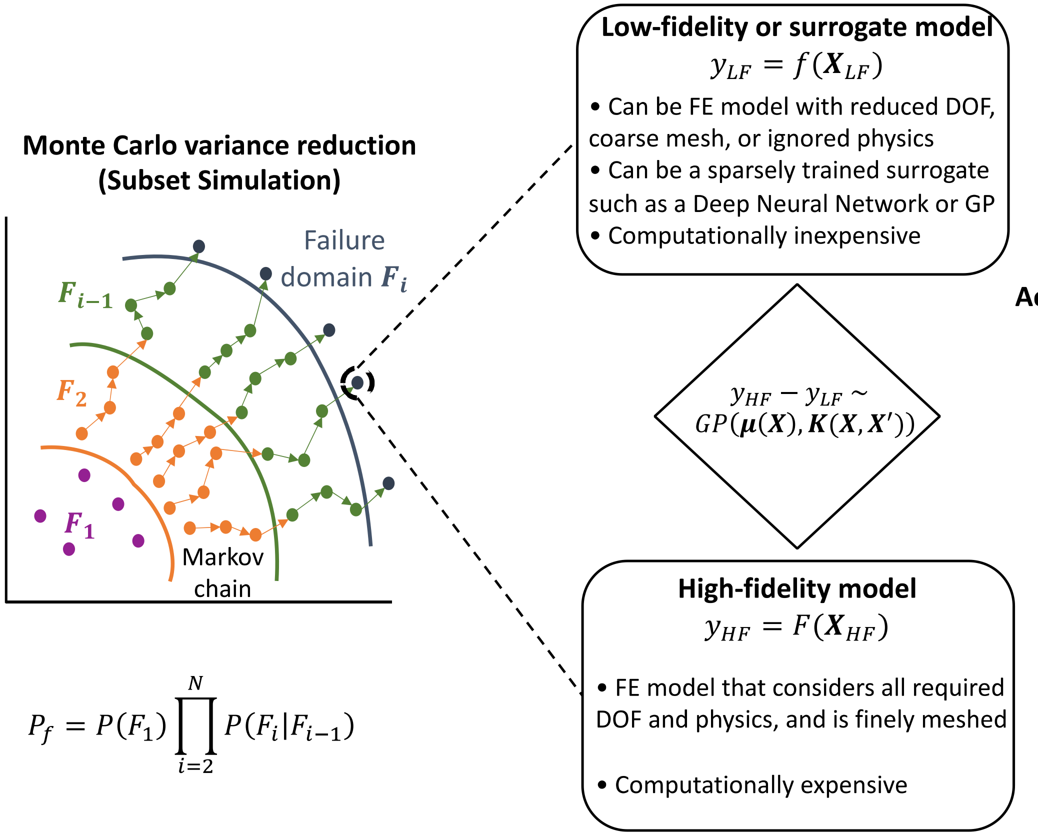
Required failure threshold

$$U^{MF}{}_s = \frac{|f(X_{LF}) + \bar{\epsilon}(X) - F|}{\sigma_\epsilon(X)}$$

(Final conditional level)

- Subset independent multifidelity U-function based on the required failure threshold
- Under smaller failure probabilities (~1E-5) differences between nominal model outputs and required failure threshold are large
- Active learning can breakdown as GP training is not triggered
- Subset dependent multifidelity U-functions are proposed
- Based on intermediate failure thresholds in Subset Simulation to trigger GP re-training
- Intermediate failure thresholds are estimated dynamically

Proposed active learning with multifidelity modeling



$$P_f = P(F_1) \prod_{i=2}^{N} P(F_i | F_{i-1})$$

Active Learning in Multifidelity Modeling

- Input parameters are $\boldsymbol{X} = \boldsymbol{X}_{LF} \cup \boldsymbol{X}_{HF}$
- Correct the low-fidelity prediction
- Recommend when a high-fidelity call is required
- Retrain GP for improved performance

- A GP is trained to learn the differences between HF and LF models (small number of samples)
- For each model evaluation in Subset Simulation, LF model is called
- LF model output is corrected using GP difference (HF-LF)
- Subset dependent U-function is computed to evaluate if HF call is required (threshold is 2)
- If HF call is made, the GP is retrained

Proposed algorithm: Statistical estimators

$$P_1 \approx \hat{P}_1 = \frac{1}{N}\sum_{i=1}^{N}\mathcal{P}_i \qquad \mathcal{P}_i = P(\mathbf{I}_i = 1) = \begin{cases} 1 \times \Phi_i + 0 \times (1-\Phi_i) = \Phi_i & \text{if } \mathbf{I}_{i,LF} = 1 \\ 0 \times \Phi_i + 1 \times (1-\Phi_i) = 1-\Phi_i & \text{if } \mathbf{I}_{i,LF} = 0 \end{cases}$$

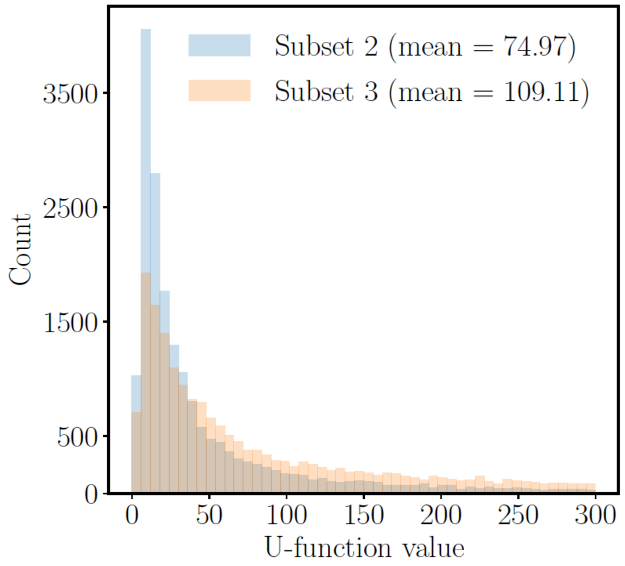
First conditional level

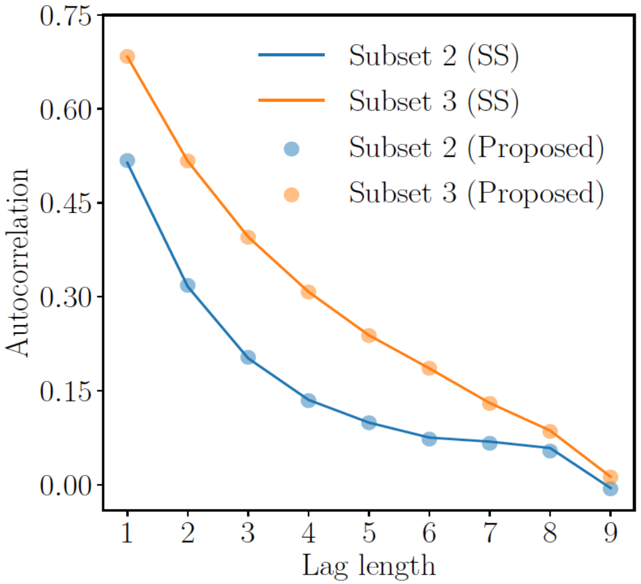
$$\gamma_1 \approx \hat{\gamma}_1 = \sqrt{\frac{1-\hat{P}_1}{\hat{P}_1\ N}}$$

$$P_{s|s-1} \approx \hat{P}_{s|s-1} = \frac{1}{N}\sum_{i=1}^{N_c}\sum_{k=1}^{N/N_c}\mathcal{P}^s_{ik} \quad \forall\, 1 < s \leq N_s,\ \mathcal{P}^s_{ik} = \begin{cases} 1 \times \Phi^s_{ik} + 0 \times (1-\Phi^s_{ik}) = \Phi^s_{ik} & \text{if } \mathbf{I}^s_{ik,LF} = 1 \\ 0 \times \Phi^s_{ik} + 1 \times (1-\Phi^s_{ik}) = 1-\Phi^s_{ik} & \text{if } \mathbf{I}^s_{ik,LF} = 0 \end{cases}$$

Subsequent conditional levels

$$\hat{\delta}_s = \sqrt{\frac{1-\hat{P}_{s|s-1}}{N\ \hat{P}_{s|s-1}}(1+\hat{\gamma}_s)} \qquad \hat{\gamma}_s = 2\sum_{k=1}^{N/N_c-1}\left(1-\frac{kN_c}{N}\ \hat{\rho}_s(k)\right)$$





- Traditional Subset Simulation estimators for conditional failure probabilities and coefficient of variations use indicator functions
- Due to reliance on a GP, these indicator functions change to probabilities because GP can have a slight error in mis-characterizing model failures (U-function threshold is 2)
- Updated statistical estimators are derived for the proposed algorithm
- For practical cases, U-function values are significantly greater than 2. Meaning, error in mis-characterizing model failures is negligible
- So, statistical estimators tend to Subset Simulation estimators

1D TRISO models failure analysis

Input parameters (7 and 11 uncertain)

Category	Parameter	Models 1 & 2	Models 3 & 4
Particle geometry	Kernel radius (μm)	$\mathcal{N}(213.35,\ 4.4)$	$\mathcal{N}(212.5,\ 5.0)$
	Buffer thickness (μm)	$\mathcal{N}(98.9,\ 8.4)$	$\mathcal{N}(100.0,\ 10.0)$
	IPyC thickness (μm)	$\mathcal{N}(40.4,\ 2.5)$	$\mathcal{N}(40.0,\ 3.0)$
	SiC thickness (μm)	$\mathcal{N}(35.2,\ 1.2)$	$\mathcal{N}(35.0,\ 2.0)$
	OPyC thickness (μm)	$\mathcal{N}(43.4,\ 2.9)$	$\mathcal{N}(40.0,\ 3.0)$
	Asphericity ratio	1.0	1.04
Fuel properties	Kernel density (gm/cm^3)	10.966	11.0
	Kernel theoretical density (gm/cm^3)	11.37	11.4
	Buffer density (gm/cm^3)	1.05	1.05
	Buffer theoretical density (gm/cm^3)	2.25	2.25
	IPyC density (gm/cm^3)	1.89	$\mathcal{N}(1.9,\ 0.02)$
	OPyC density (gm/cm^3)	1.907	$\mathcal{N}(1.9,\ 0.02)$
	IPyC anisotropy factor	1.0465	$\mathcal{N}(1.05,\ 0.005)$
	OPyC anisotropy factor	1.0429	$\mathcal{N}(1.05,\ 0.005)$
Layer strengths	IPyC strength	$\mathcal{W}(\sigma_{ms},\ 9.5)$	$\mathcal{W}(\sigma_{ms},\ 9.5)$
	SiC strength	$\mathcal{W}(\sigma_{ms},\ 6.0)$	$\mathcal{W}(\sigma_{ms},\ 6.0)$

uncertain

uncertain

uncertain

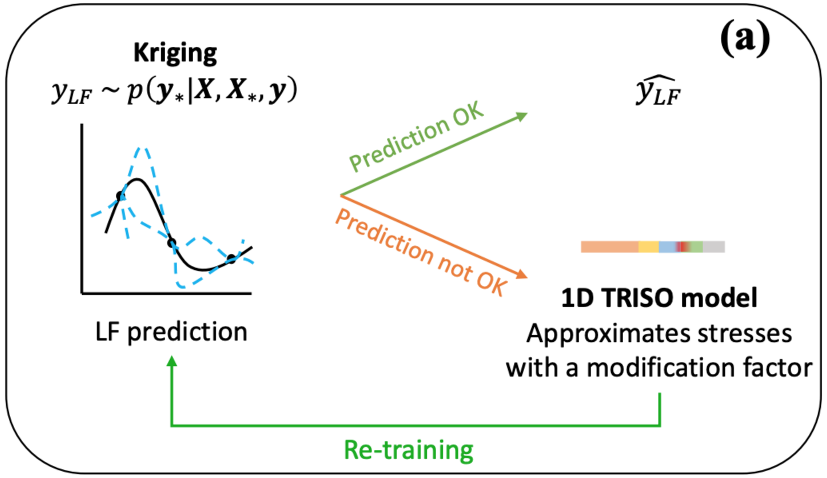
Output: SiC Stress – strength (> 0 is failure)

Irradiation temperatures

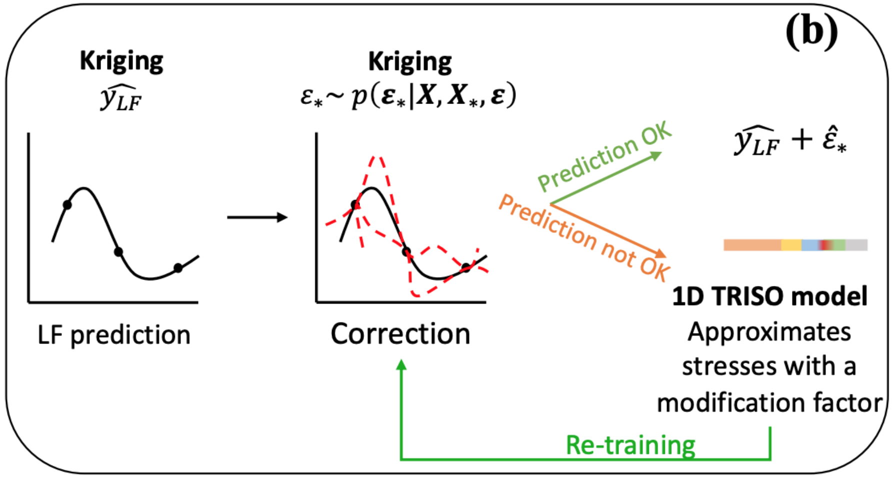
Model 1	Model 2	Model 3	Model 4
Type = Daily varying Max. = 1226.84°C Min. = 207.4°C	Type = Daily varying Max. = 1281.84°C Min. = 195.84°C	Type = Constant Value = 700.0°C	Type = Constant Value = 1000.0°C

1D TRISO models failure analysis

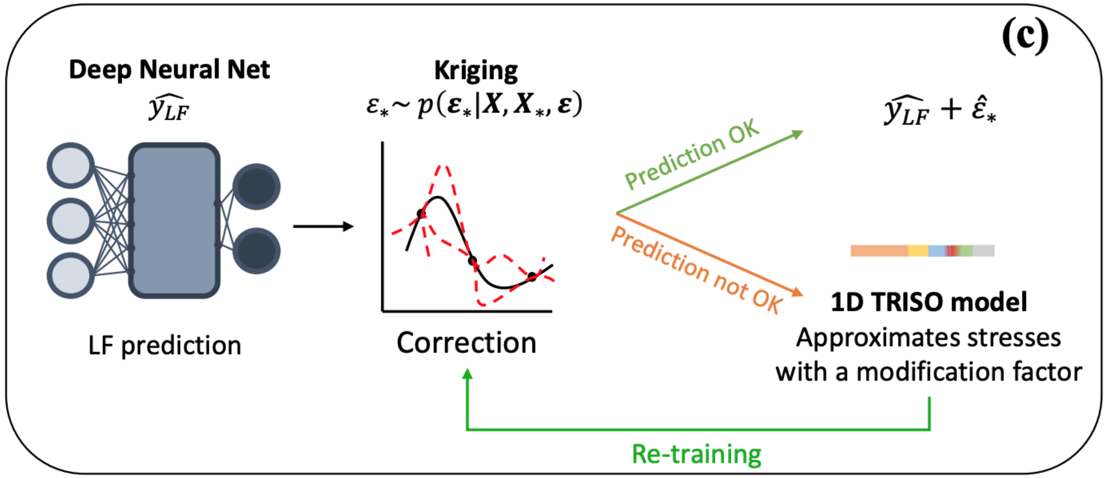
Only Kriging



Kriging (LF) + Kriging (correction)

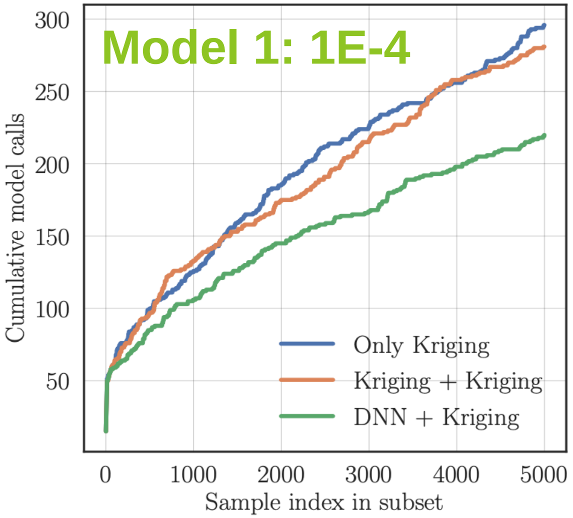


DNN (LF) + Kriging (correction)

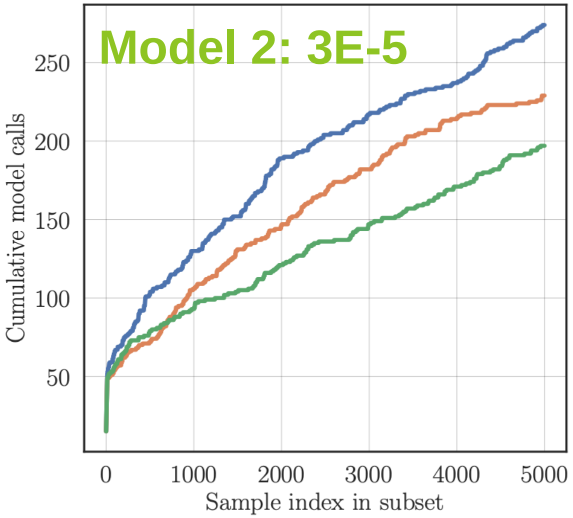


All surrogates trained on 12 evals of 1D TRISO output

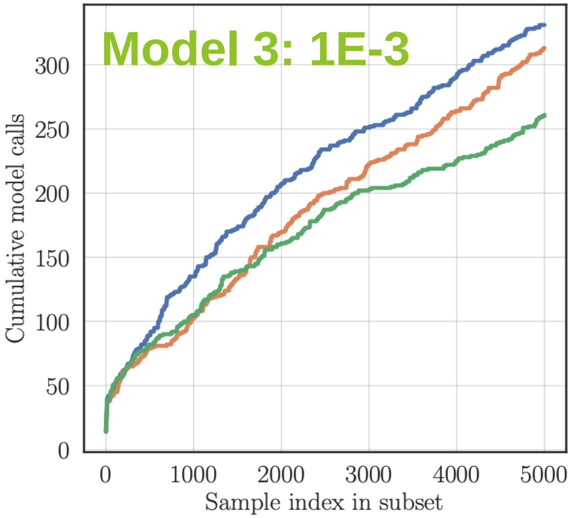
1D TRISO models failure analysis



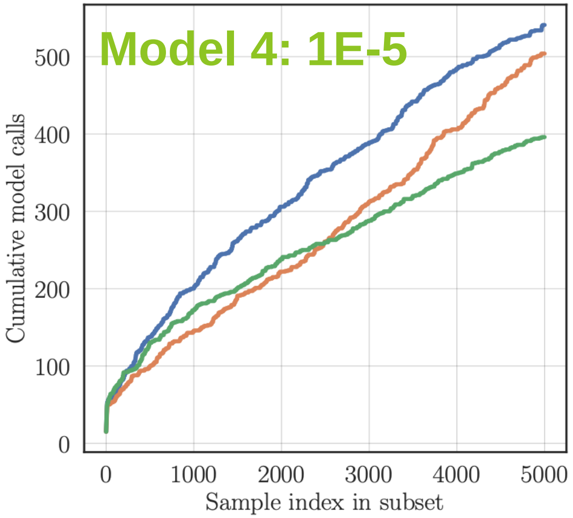
(a)



(b)



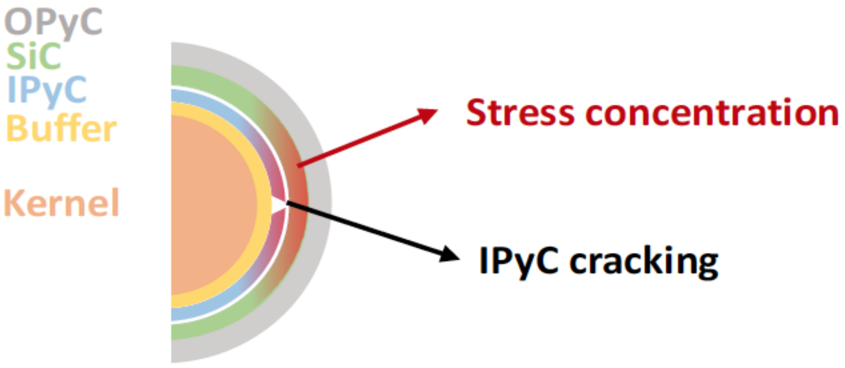
(c)

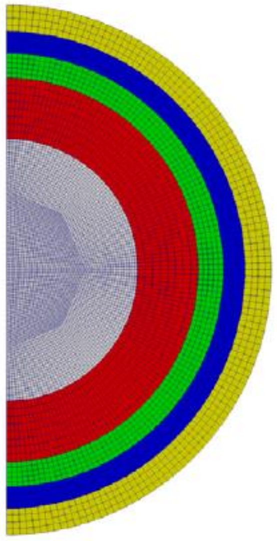


(d)

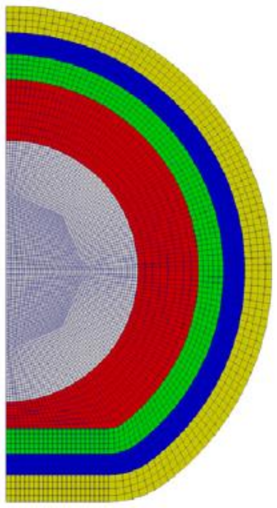
- All three strategies accurately predict the failure probabilities across the four models (COV ~0.08)
- Kriging + Kriging and DNN + Kriging require lesser calls to the 1D TRISO compared to Only Kriging
- DNN + Kriging 26% and 18% less calls than Kriging + Kriging and Only Kriging, respectively
- Possible reason for less calls: more information gain due to multifidelity models and better DNN regularization

2D TRISO models failure analysis





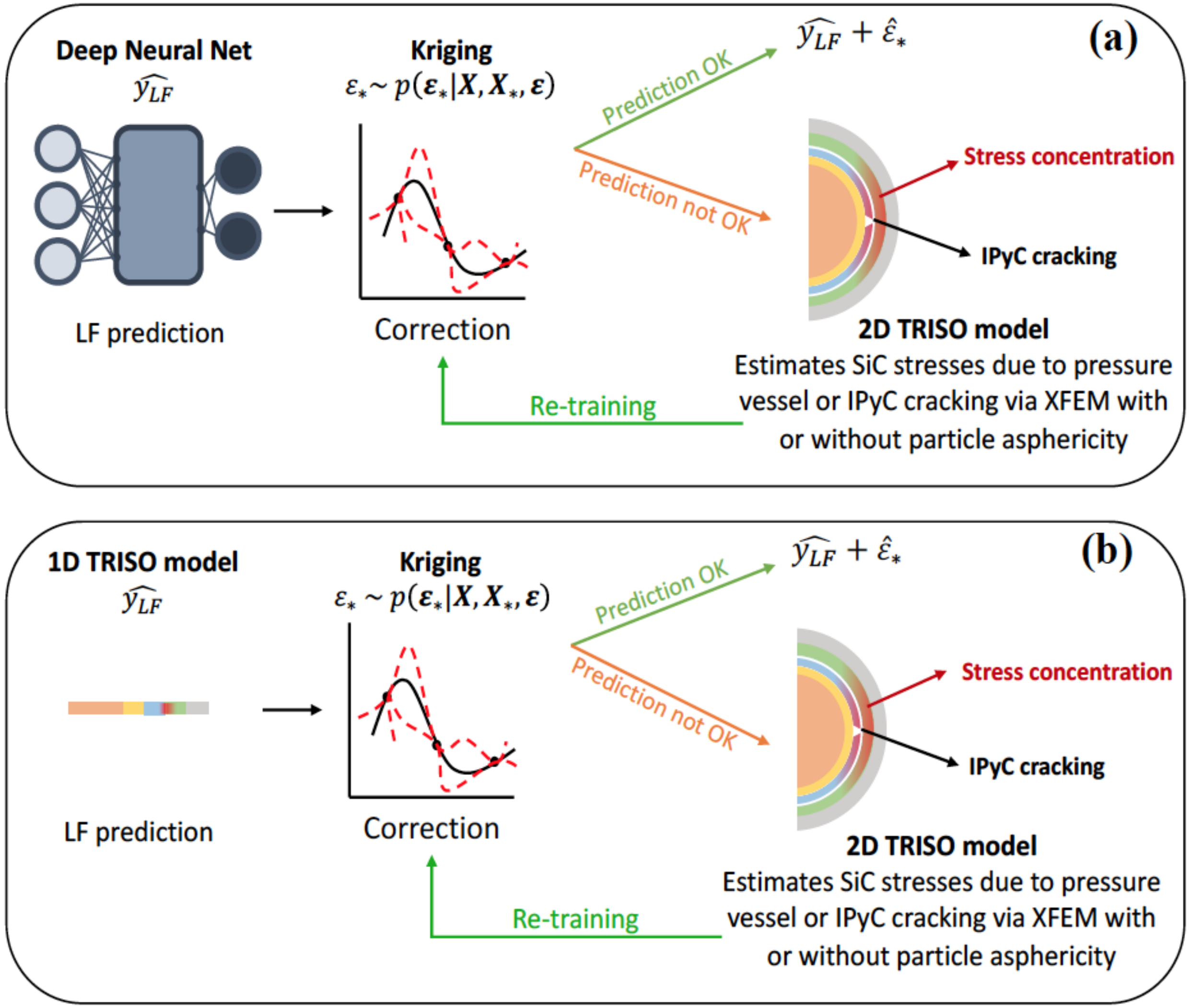
Spherical



Aspherical

- The 1-D models approximate stresses in the SiC layer based on modification factors
- These factors are calibrated by running evals of the 2-D model
- 2-D model explicitly models cracking in IPyC layer and stress conc. in SiC layer
- More accurate, but mesh density dependent. Therefore, computationally expensive (~30 mins)
- Same random input params: geometry, material props
- Same output: SiC stress – strength (> 0 failure)

2D TRISO models failure analysis

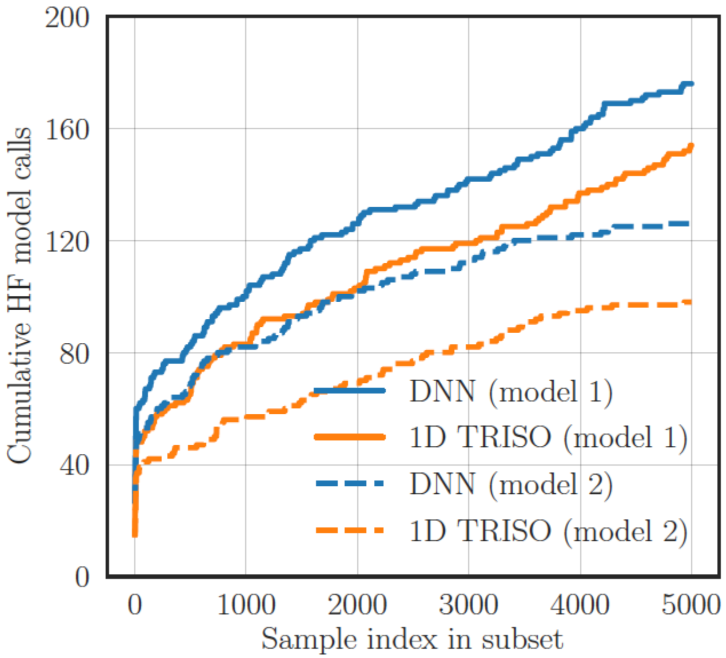


"Data-driven" strategy (DNN LF + Kriging correction)

"Physics-based" strategy (1D TRISO LF + Kriging correction)

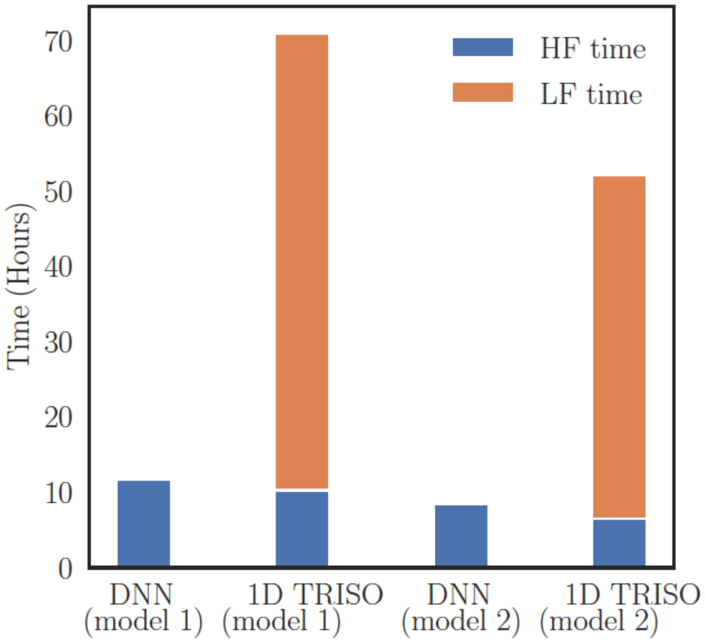
All surrogates trained on 12 evals of 2D TRISO output

2D TRISO models failure analysis



HF calls with sample index

Total simulation time

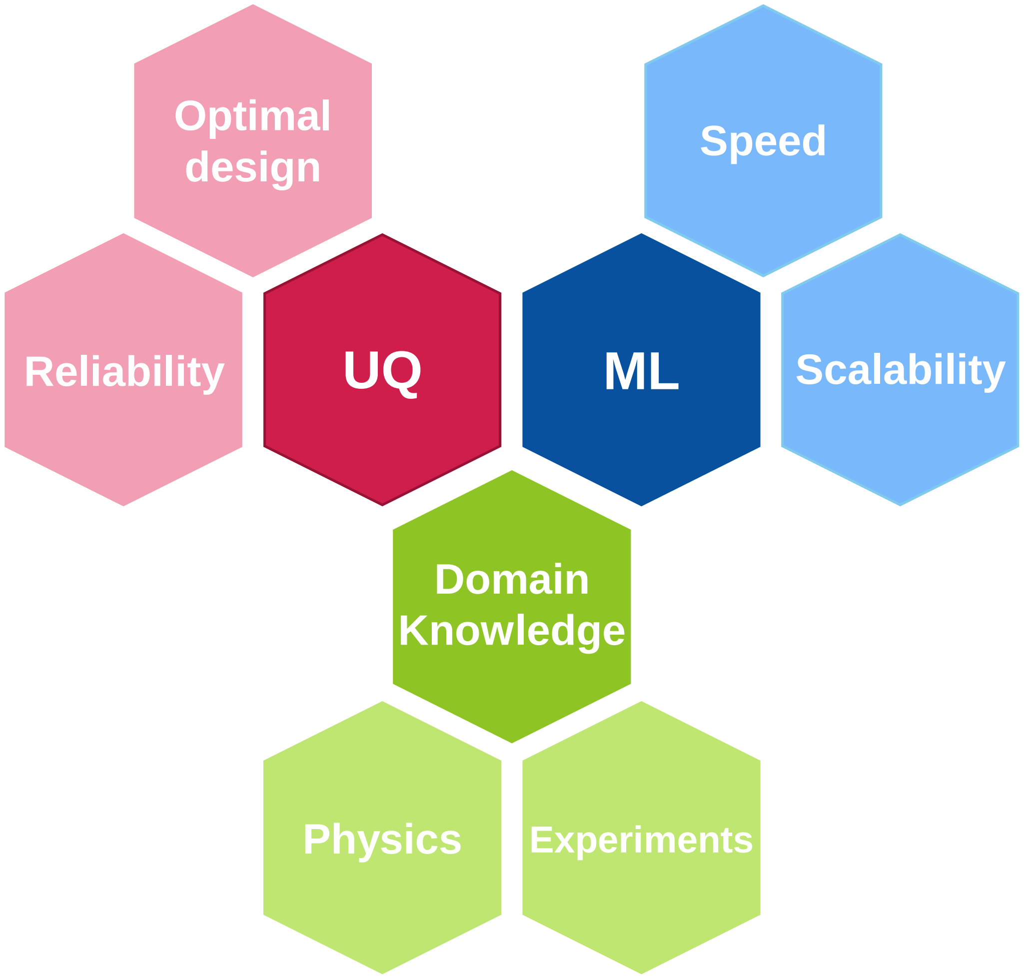


- Both “data-driven” and “physics-based” strategies accurately predict the failure probabilities for two models (COV ~ 0.08)
- “Physics-based” strategy which uses 1D TRISO LF requires 16% less calls to the 2D TRISO model
- “Data-driven” strategy has lesser overall simulation time because the DNN LF predictions are instantaneous
- 1D TRISO LF still requires 11 sec for each eval

IDAHO NATIONAL LABORATORY

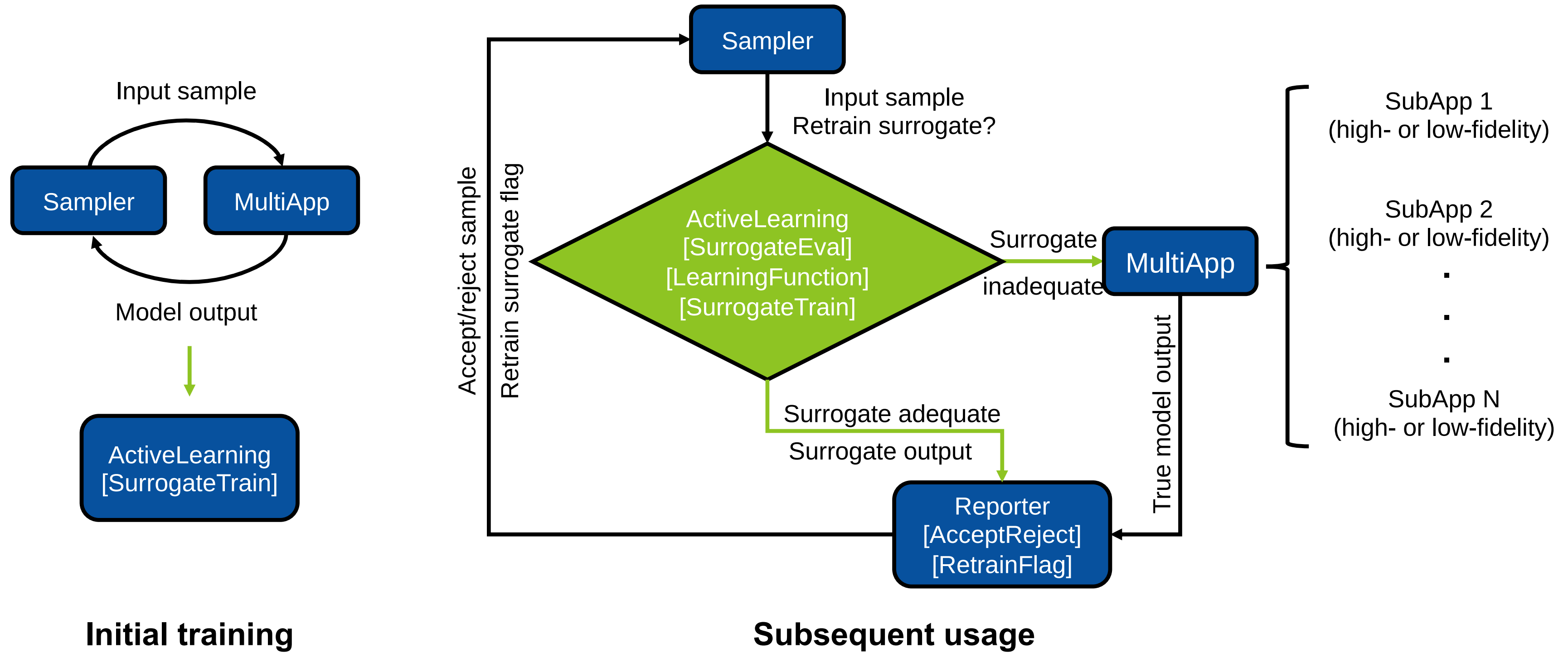
Outline

- **Prediction**
  - Deterministic and Bayesian predictions
  - A simple Bayesian surrogate: Gaussian Process
  - Beyond Gaussian Processes
- **Inference**
  - Sampling
  - Markov Chain Monte Carlo
  - Beyond MCMC: Hybrid MCMC and sampling as optimization
- **Active learning and Multifidelity modeling**
- **TRISO nuclear fuel failure analysis**
- **Ongoing work:**
  - **MOOSE stochastic tools module**
  - **Monte Carlo with Hamiltonian Neural Nets**



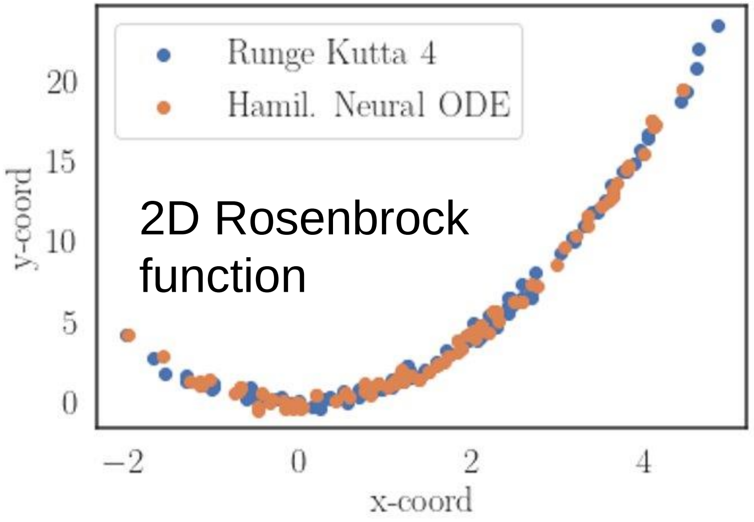
Combo of the above three benefits computational tasks

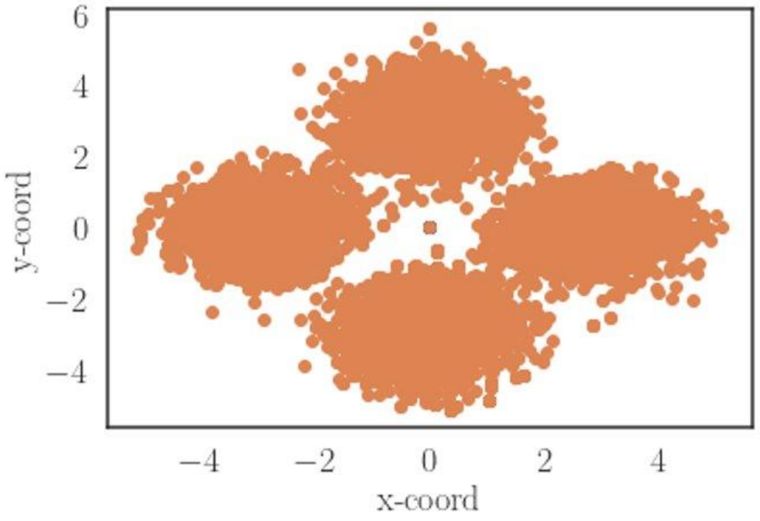
Adaptive sampling and active learning methods in MOOSE



- MOOSE: Multiphysics Object Oriented Simulation Environment (https://mooseframework.inl.gov/)
- Massively parallel, modular development, used for many applications
- Adaptive and active learning Monte Carlo algorithms in MOOSE Stochastic Tools Module

Monte Carlo with Hamiltonian Neural Networks





2D 4 Gaussian mixture

- Sampling from complex distributions can be performed more reliably with Hamiltonian Monte Carlo (HMC)

$$\boldsymbol{z}_1 = \boldsymbol{z}_0 + \int_{t_0}^{t_1} \boldsymbol{I}\ \nabla H(\boldsymbol{z})\ dt$$

- But gradient evaluations are computationally expensive!!

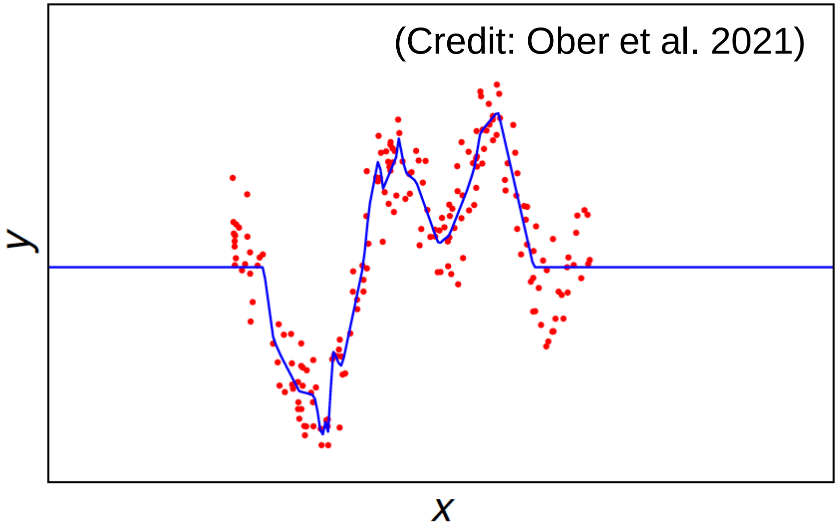
- Hamiltonian Neural ODEs learn the Hamiltonian dynamics and side-step gradient evaluations in HMC

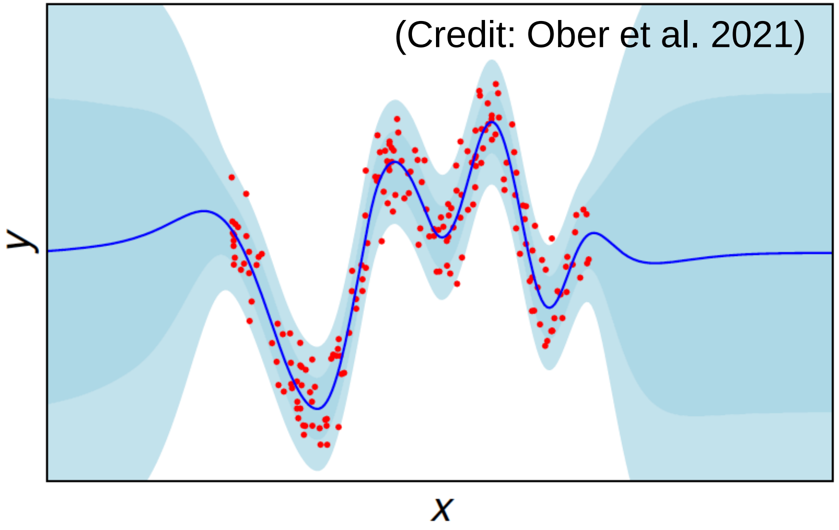
- In addition, they conserve the Hamiltonian

$$\mathcal{L}_{HNN} = \left\| \frac{\partial \mathcal{H}_\theta}{\partial \mathbf{p}} - \frac{\partial \mathbf{q}}{\partial t} \right\|_2 + \left\| - \frac{\partial \mathcal{H}_\theta}{\partial \mathbf{q}} - \frac{\partial \mathbf{p}}{\partial t} \right\|_2$$

- Useful for sampling from complex distributions efficiently

Thank you!
(Som.Dhulipala@inl.gov)





References

- Ober, S. W., Rasmussen, C. E., & van der Wilk, M. (2021). The promises and pitfalls of deep kernel learning. *arXiv preprint arXiv:2102.12108*.
- Hendrickson, B. (2020). *ASCR@ 40: Four Decades of Department of Energy Leadership in Advanced Scientific Computing Research. A Report from the Advanced Scientific Computing Advisory Committee (ASCAC)*. Krell Inst., Ames, IA (United States).
- Lakshminarayanan, B., Tran, D., & Snoek, J. (2021) Introduction to Uncertainty in Deep Learning. *NeurIPS Tutorial, virtual conference.*
- Wang, K., Sun, W., & Du, Q. (2019). A cooperative game for automated learning of elasto-plasticity knowledge graphs and models with AI-guided experimentation. *Computational Mechanics*, *64*(2), 467-499.
- Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, *378*, 686-707.
- Haghighat, E., & Juanes, R. (2021). Sciann: A keras/tensorflow wrapper for scientific computations and physics-informed deep learning using artificial neural networks. *Computer Methods in Applied Mechanics and Engineering*, *373*, 113552.
- Liu, W. K., Li, S., & Park, H. (2021). Eighty Years of the Finite Element Method: Birth, Evolution, and Future. *arXiv preprint arXiv:2107.04960*.
- Do, C. (2008). Gaussian Processes. *Stanford University lecture notes.* (http://cs229.stanford.edu/section/cs229-gaussian_processes.pdf)

References

- Hoff, P. D. (2009). *A first course in Bayesian statistical methods* (Vol. 580). New York: Springer.
- Dhulipala, S. L. N., Shields, M. D., Spencer, B. W., Bolisetti, C., Slaughter, A. E., Laboure, V. M., & Chakroborty, P. (2021). Adaptive and Efficient Rare Event Analysis using a Gaussian Process Modeling Fidelity Recommender System. *16th US National Congress on Computational Mechanics*. (https://www.researchgate.net/publication/352030774_Adaptive_and_Efficient_Rare_Event_Analysis_Using_a_Gaussian_Process_Modeling_Fidelity_Recommender_System)
- Duvenaud, D. *The Kernel Cookbook: Advice on Covariance Functions*. (https://www.cs.toronto.edu/~duvenaud/cookbook/)
- Micchelli, C. A., Xu, Y., & Zhang, H. (2006). Universal Kernels. *Journal of Machine Learning Research*, *7*(12).
- Baiges, J., Codina, R., Castanar, I., & Castillo, E. (2020). A finite element reduced-order model based on adaptive mesh refinement and artificial neural networks. *International Journal for Numerical Methods in Engineering*, *121*(4), 588-601.
- Wang, S., Xinling, W., and Perdikaris, P. (2020) *Why and when physics-informed neural networks fail to train.* LLNL seminar (https://www.youtube.com/watch?v=xvOsV106kuA)
- Wilson, A. G., Hu, Z., Salakhutdinov, R., & Xing, E. P. (2016, May). Deep kernel learning. In *Artificial intelligence and statistics* (pp. 370-378). PMLR.
- Sun, L., & Wang, J. X. (2020). Physics-constrained bayesian neural network for fluid flow reconstruction with sparse and noisy data. *Theoretical and Applied Mechanics Letters*, *10*(3), 161-169.

References

- Jiang, W., Hales, J. D., Spencer, B. W., Collin, B. P., Slaughter, A. E., Novascone, S. R., ... & Gardner, R. (2021). TRISO particle fuel performance and failure analysis with BISON. *Journal of Nuclear Materials*, *548*, 152795.
- Che, Y., Wu, X., Pastore, G., Li, W., & Shirvan, K. (2021). Application of Kriging and Variational Bayesian Monte Carlo method for improved prediction of doped UO2 fission gas release. *Annals of Nuclear Energy*, *153*, 108046.
- Dhulipala, S. L. N. (2019). *Bayesian methods for intensity measure and ground motion selection in performance-based earthquake engineering* (Doctoral dissertation, Virginia Tech).
- Neal, R. M. (1993). *Probabilistic inference using Markov chain Monte Carlo methods* (pp. 93-1). Toronto, ON, Canada: Department of Computer Science, University of Toronto.
- Dhulipala, S. L. N., Shields, M. D., Spencer, B. W., Bolisetti, C., Slaughter, A. E., Laboure, V. M., & Chakroborty, P. (2021). Active Learning with Multifidelity Modeling for Efficient Rare Event Simulation. *arXiv preprint arXiv:2106.13790*.
- Lykkegaard, M. B., Dodwell, T. J., & Moxey, D. (2021). Accelerating uncertainty quantification of groundwater flow modelling using a deep neural network proxy. *Computer Methods in Applied Mechanics and Engineering*, *383*, 113895.
- Mancang, L., Kan, W., & Dong, Y. (2011). Development of a Monte Carlo multi-group constants generation code.
- Neal, R. M. (2011). MCMC using Hamiltonian dynamics. *Handbook of markov chain monte carlo*, *2*(11), 2.
- Hanson, K. (2005). Bayesian analysis in Nuclear Physics. LANL. (https://kmh-lanl.hansonhub.com/talks/lansce05-t4vgr.pdf)

References

- Blei, D. M., Kucukelbir, A., & McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American statistical Association*, *112*(518), 859-877.
- Kontolati, K., Loukrezis, D., dos Santos, K. R., Giovanis, D. G., & Shields, M. D. (2021). Manifold learning-based polynomial chaos expansions for high-dimensional surrogate models. *arXiv preprint arXiv:2107.09814*.
- Gorodetsky, A. A., Geraci, G., Eldred, M. S., & Jakeman, J. D. (2020). A generalized approximate control variate framework for multifidelity uncertainty quantification. *Journal of Computational Physics*, *408*, 109257.
- Geraci, G., Eldred, M. S., & Iaccarino, G. (2017). A multifidelity multilevel Monte Carlo method for uncertainty propagation in aerospace applications. In *19th AIAA non-deterministic approaches conference* (p. 1951).
- Paaren, K. M., Lybeck, N., Mo, K., Medvedev, P., & Porter, D. (2021). Cladding Profilometry Analysis of Experimental Breeder Reactor-II Metallic Fuel Pins with HT9, D9, and SS316 Cladding. *Energies*, *14*(2), 515.
- Hales, J. D., Jiang, W., Toptan, A., & Gamble, K. A. (2021). Modeling fission product diffusion in TRISO fuel particles with BISON. *Journal of Nuclear Materials*, *548*, 152840.
- Zhang, J. (2020). Modern Monte Carlo methods for efficient uncertainty quantification and propagation: A survey. *Wiley Interdisciplinary Reviews: Computational Statistics*, e1539.