

Efficient Uncertainty Quantification Using High Performance Computing and Scientific Machine Learning

Jim Warner[1] & Patrick Leser[1]

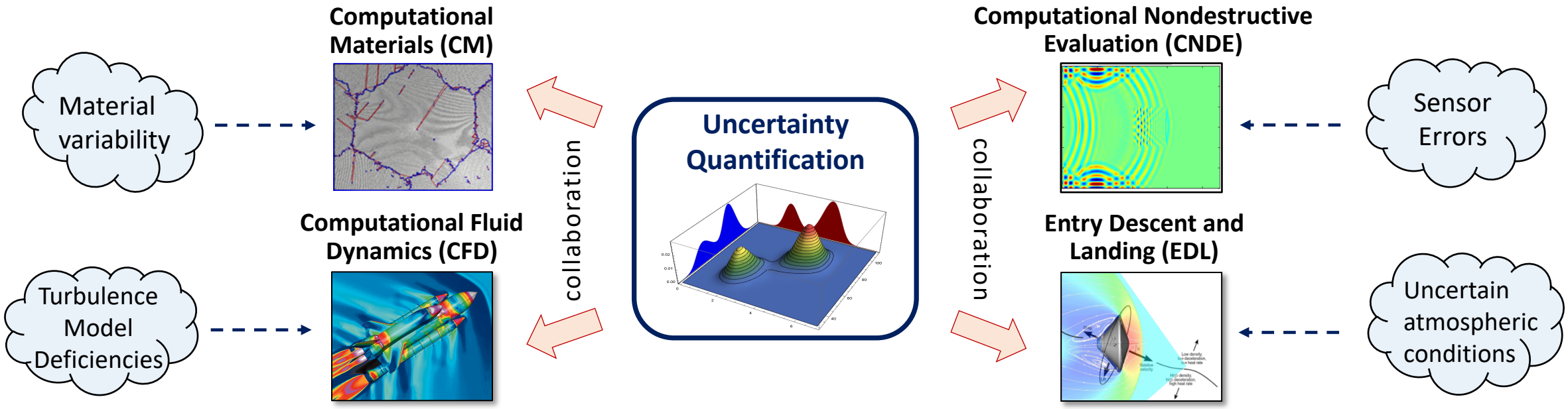
Team: **Theodore Lewitt[2], Julian Cuevas[3]**, Geoff Bomarito[1], Paul Leser[1], Luke Morrill[4], Samantha Niemoeller[5]

1. NASA LaRC D309
2. University of Southern California (**NIFS**)
3. University of Puerto Rico (**NIFS**)
4. Georgia Tech (**NIFS**), NASA LaRC E302
5. UCLA (**NIFS**)

Outline

- Overview: Uncertainty quantification (UQ) under HPCI
- Scientific Machine Learning for UQ
- Physics-informed generative adversarial networks (PI-GANs) for virtual mechanical testing
- Conclusion

UQ as an HPCI Discipline



- Goal of UQ: *Produce **reliable** model predictions with associated **confidence** in the presence of uncertainty*
  - Risk reduction; decreased margins/costs; enable reliability-based design; confidence for decision makers
  - Effective UQ is a prerequisite for NASA paradigms like Digital Twin and certification by analysis
- Traditional UQ methods are **intractable** for expensive, physics-based simulations; Faster but **less accurate** data-driven surrogate models are used

HPCI UQ Overview

Goals:

- Develop state-of-the-art UQ algorithms with HPC implementations released as open source software
- Demonstrate effectiveness & generality of software on test cases across HPCI disciplines

Software

SROMPy
Stochastic Reduced Order Models with Python
github.com/nasa/SROMPy

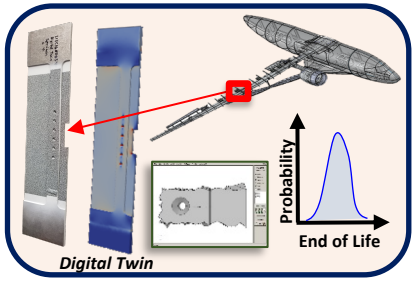
MXMCPy
Multi-Model Monte Carlo with Python
github.com/nasa/MXMCPy

SMCPy
Sequential Monte Carlo with Python
github.com/nasa/SMCPy

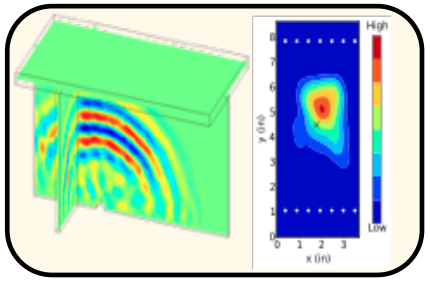
bingo
Symbolic Regression with Python
github.com/nasa/bingo

Applications

Structural Health Management (CM)



Damage Diagnosis (CNDE)

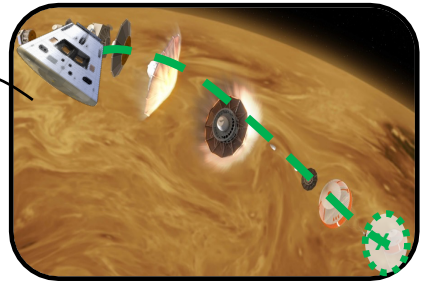


Frangible Joint Reliability (CM)

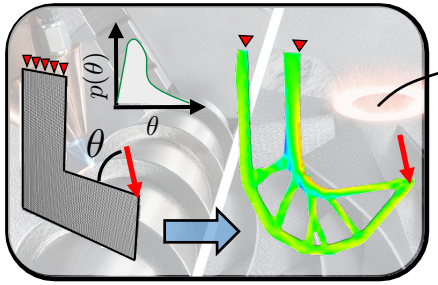


Trajectory Simulation (EDL)

~50X speedup predicting landing location with MXMCPy



Design Under Uncertainty (CM)



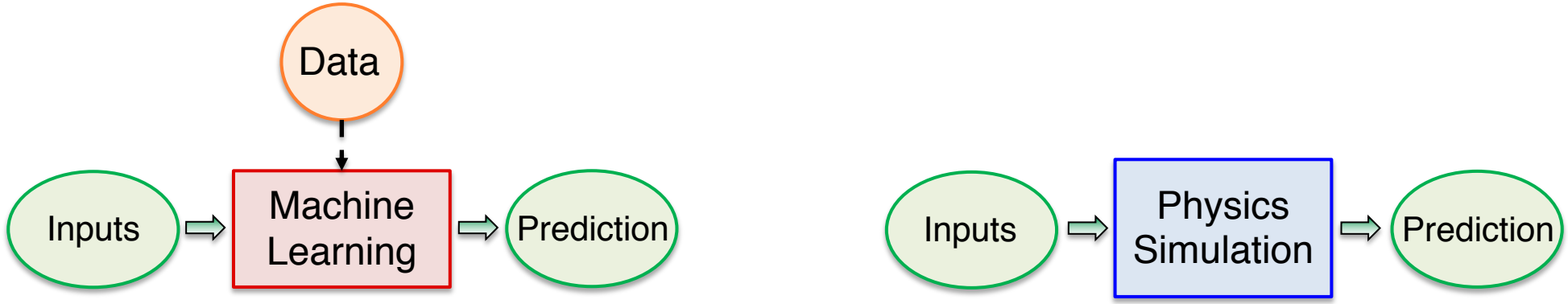
~20X speedup generating optimal robust designs with SROMPy

Outline

- Overview: Uncertainty quantification (UQ) under HPCI
- **Scientific Machine Learning for UQ**
- Physics-informed generative adversarial networks (PI-GANs) for virtual mechanical testing
- Conclusion

Scientific Machine Learning

***Big decisions** need more than big data, they **need big models** too**



- Efficient predictions
- Fast deployment
- Widely accessible from open-source libraries

- Interpretable predictions
- Theoretically rigorous
- Historically-proven legacy codes

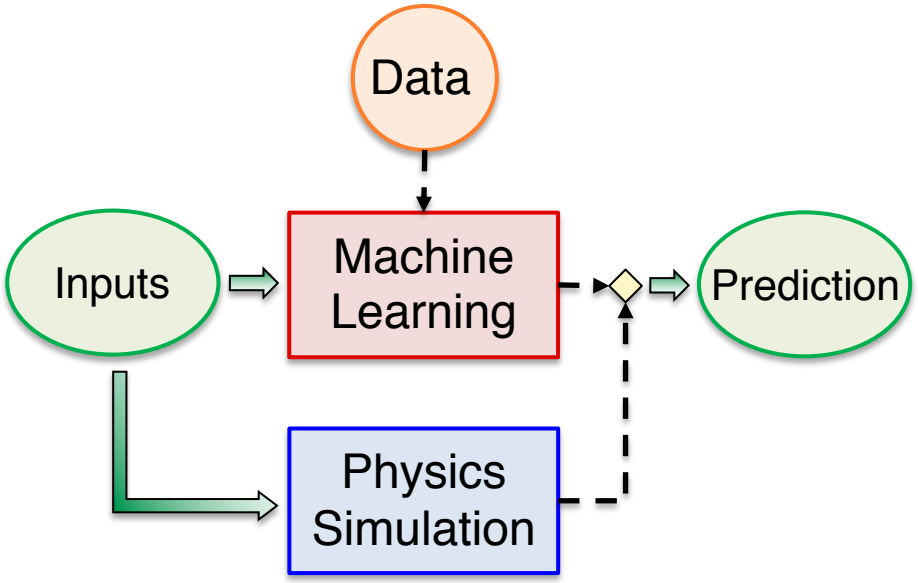
Scientific Machine Learning (ML)

*Karen Willcox, Supercomputing 2019

Scientific ML: Two Approaches

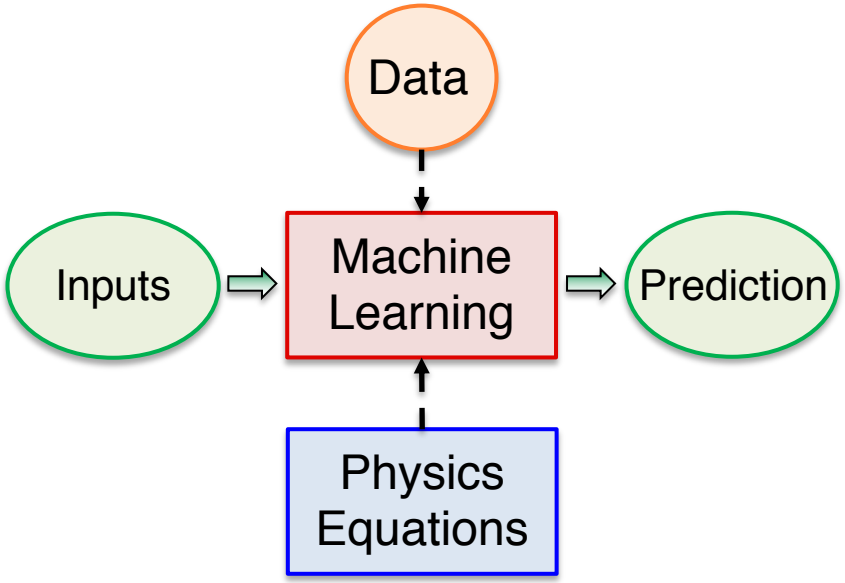
Focus of this talk

1. ***Fuse*** predictions from ML and physics-based models



- **Method:** Multi-model Monte Carlo simulation [1,2,3]

2. ***Integrate*** physical laws into the training of ML models



- **Method:** Physics-informed deep learning [4,5]

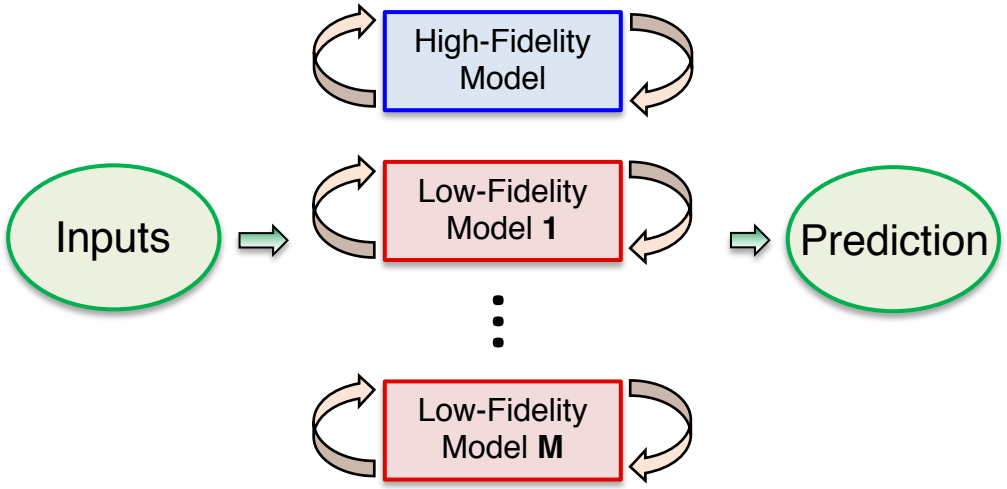
1. Multilevel Monte Carlo. Giles et al. 2015
2. Multifidelity Monte Carlo. Peherstorfer et al. 2016
3. Approximate Control Variates. Gorodetsky et al. 2019
4. Physics-informed neural networks. Raissi et al. 2019.
5. Physics-informed generative adversarial networks Yang et. al. 2018

Scientific ML: Two HPC Strategies

Focus of this talk

Multi-model Monte Carlo simulation

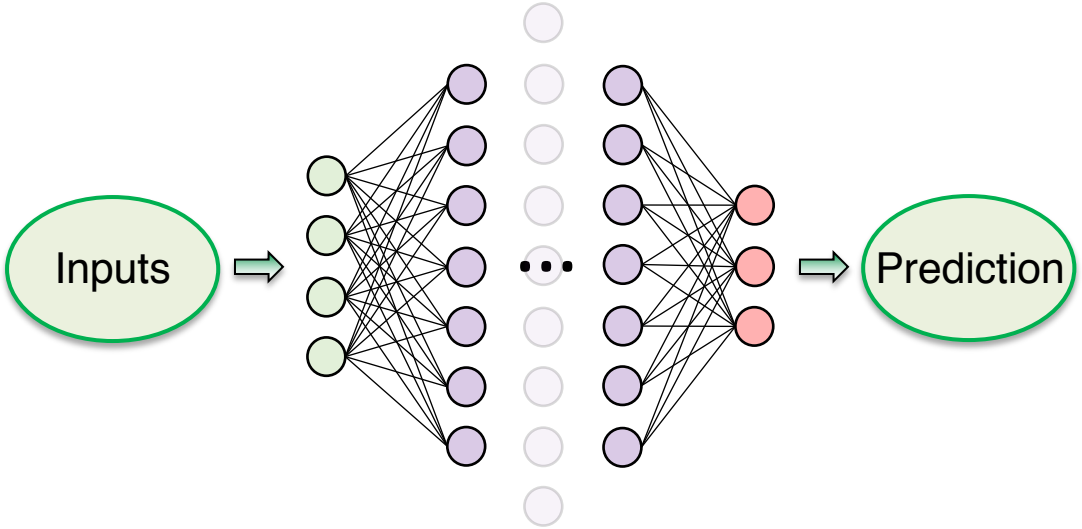
Repeatedly evaluates a suite of models with disparate run times



- **Approach:** Parallelize model evaluations with MPI
- **Challenge:** load balancing to ensure all processes have similar amount of work

Physics-informed deep learning

Trains a deep fully-connected neural network with complex loss function

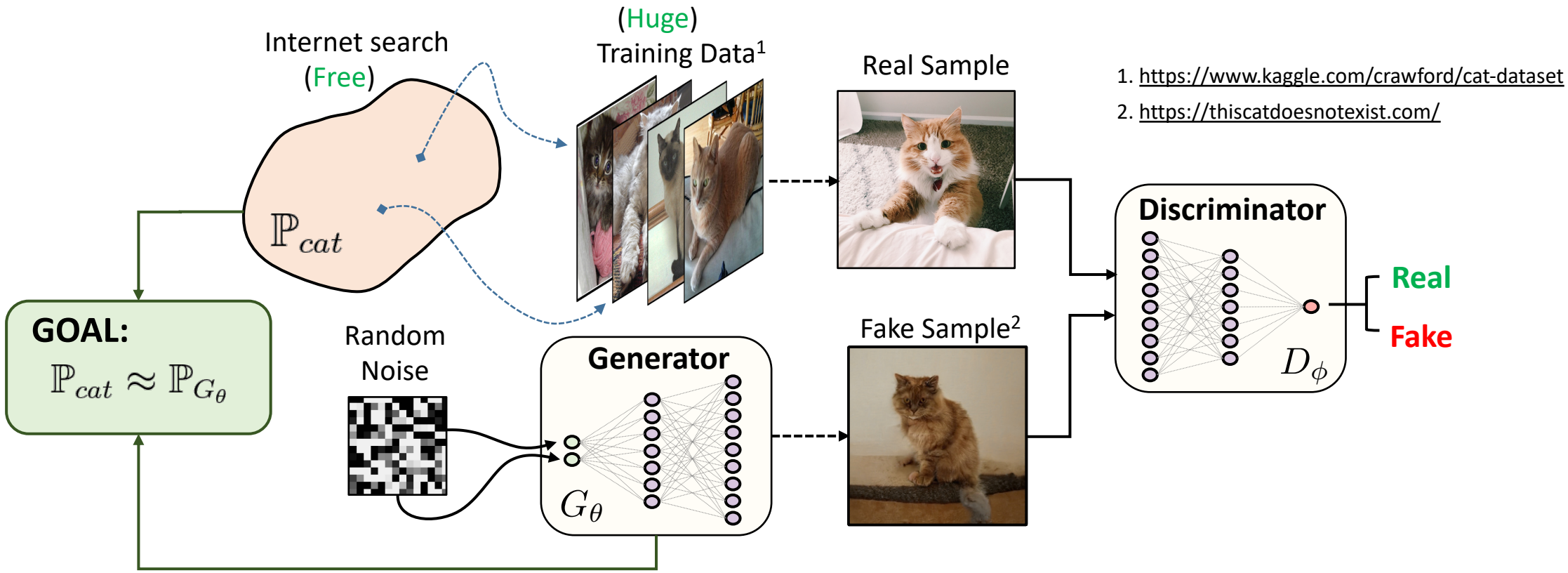


- **Approach:** GPU accelerated training using Tensorflow
- **Challenge:** selecting optimal network hyperparameters requires multiple training runs

Outline

- Overview: Uncertainty quantification (UQ) under HPCI
- Scientific Machine Learning for UQ
- **Physics-informed generative adversarial networks (PI-GANs) for virtual mechanical testing**
- Conclusion

Generative Adversarial Networks (GANs)



1. https://www.kaggle.com/crawford/cat-dataset
2. https://thiscatdoesnotexist.com/

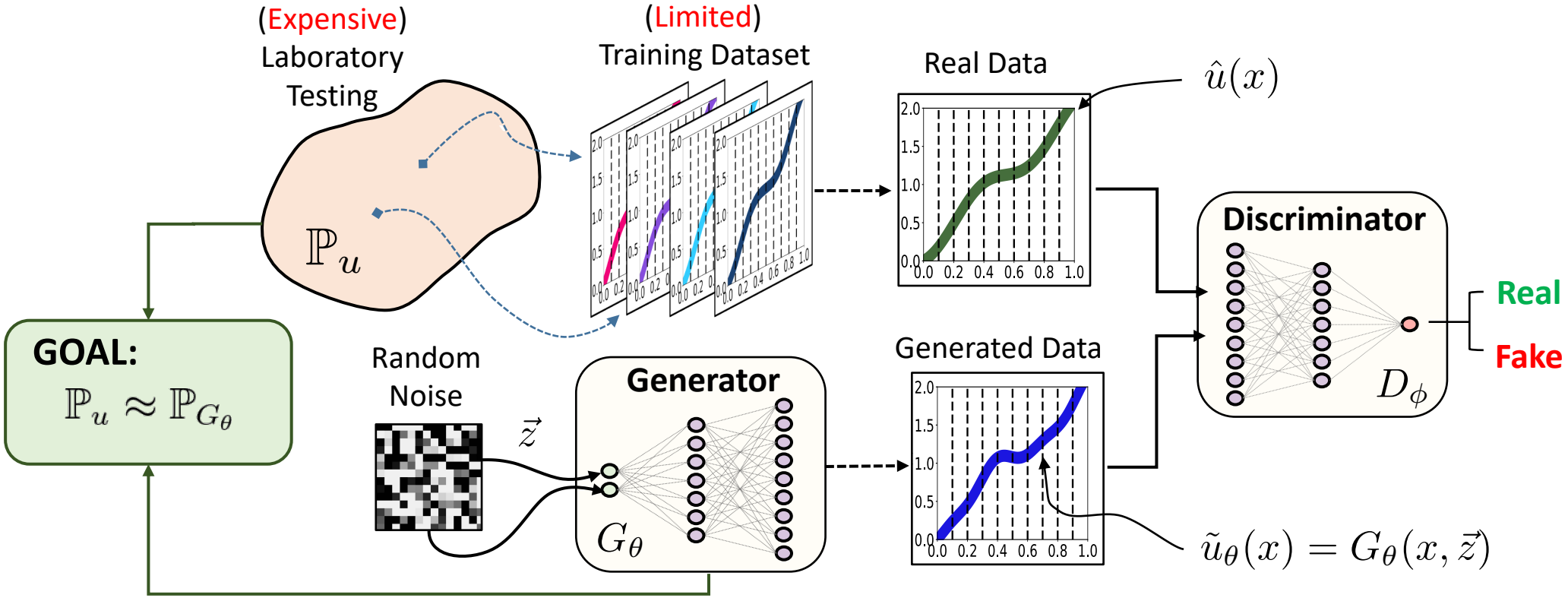
Measures how well the discriminator can distinguish between real and fake samples

Train GAN via:

$$\text{argmin}_{\theta,\phi} \underbrace{\mathcal{L}_D(D_\phi)}_{} + \underbrace{\mathcal{L}_G(G_\theta)}_{}$$

Measures how well the generator can "trick" the discriminator into classifying fake samples as real

Physics-Informed GANs (PI-GANs)



Physics

Governing PDE ($x \in \Omega$):

$$r(x) = \mathcal{N}_x[u(x)] - f(x) = 0$$

Boundary Condition ($x \in \Gamma$):

$$u(x) = v$$

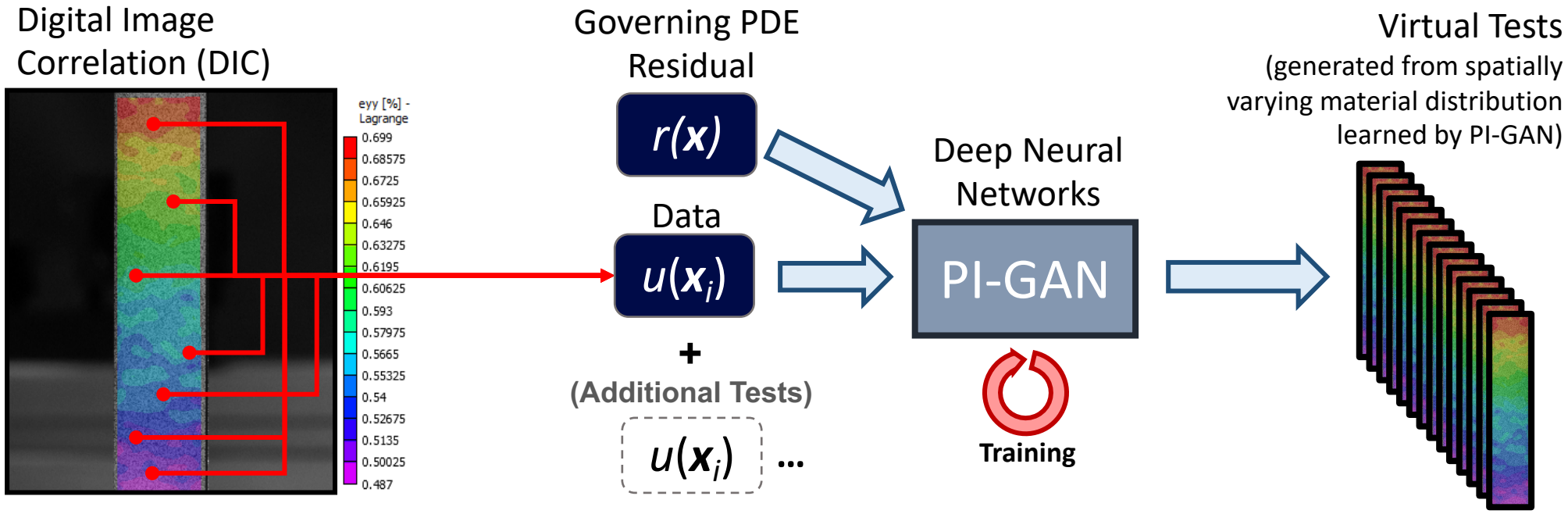
Train PI-GAN via:

$$\text{argmin}_{\theta,\phi} \; \mathcal{L}_D(D_\phi) + \mathcal{L}_G(G_\theta) + \mathcal{L}_{PDE}(G_\theta) + \mathcal{L}_{BC}(G_\theta)$$

$$\frac{1}{N_r}\sum_{i=1}^{N_r} \|\mathcal{N}_x[\tilde{u}_\theta(x^{(i)})] - f(x^{(i)})\|^2$$

$$\frac{1}{N_b}\sum_{i=1}^{N_b} \|\tilde{u}_\theta(x^{(i)}) - v\|^2$$

PI-GANs for Virtual Testing



- **Goal:** learn distribution of material stiffness, $E(x)$, using *only* DIC *deformation* data, $\{\hat{u}(x)\}$
- **Approach**: define two generators, $\tilde{u}_\theta(x) = G_\theta(x,z)$ and $\tilde{E}_\psi(x) = G_\psi(x,z)$, and relate through PDE:

Governing PDE:

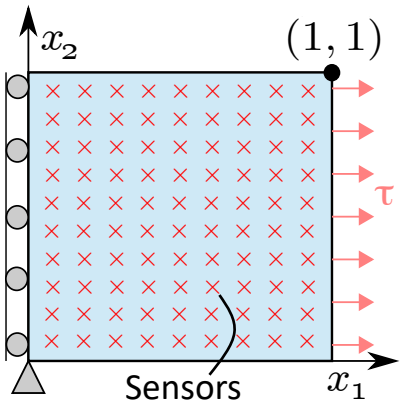
$$\nabla \cdot \boldsymbol{\sigma}(\mathbf{u}, E) = \mathbf{0}$$

$$\Rightarrow \quad \mathcal{L}_{PDE}(G_\theta, G_\psi) = \frac{1}{N_r}\sum_{i=1}^{N_r} \|\nabla \cdot \boldsymbol{\sigma}(\tilde{u}_\theta(x^{(i)}), \tilde{E}_\psi(x^{(i)}))\|^2$$

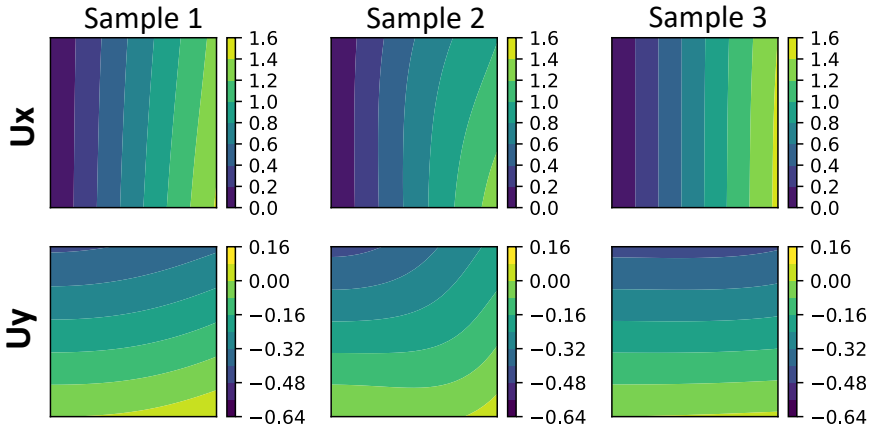
Numerical Proof-of-Concept

- Train a PI-GAN to generate realistic samples of a *lognormally-distributed*, *spatially-varying* material stiffness and corresponding deformations under load using *only* deformation data
- Synthetic measurement data generated via the finite element method

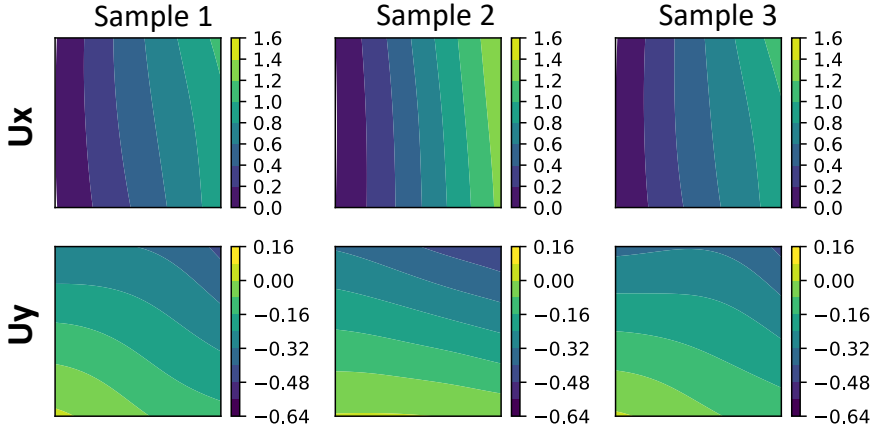
Domain



Deformation Data (Real) Samples



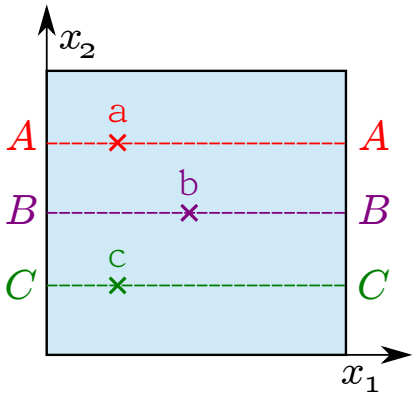
***PI-GAN Generated* (Fake) Samples**



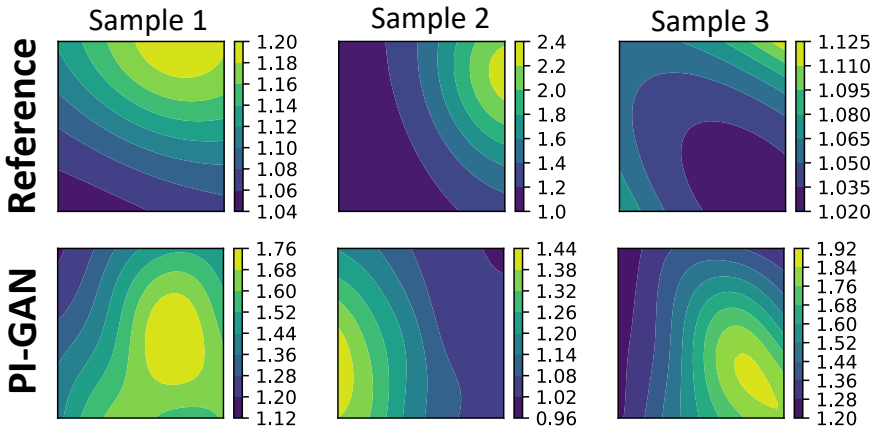
Numerical Proof-of-Concept

- Train a PI-GAN to generate realistic samples of a *lognormally-distributed*, *spatially-varying* material stiffness and corresponding deformations under load using *only* deformation data
- Synthetic measurement data generated via the finite element method

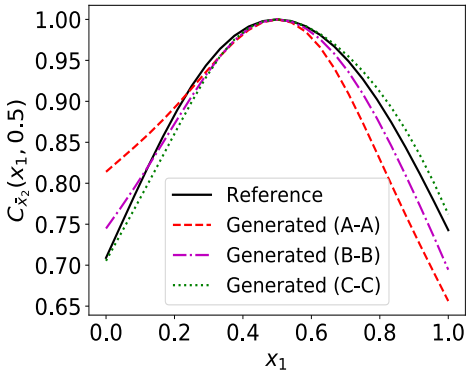
Solution Reference



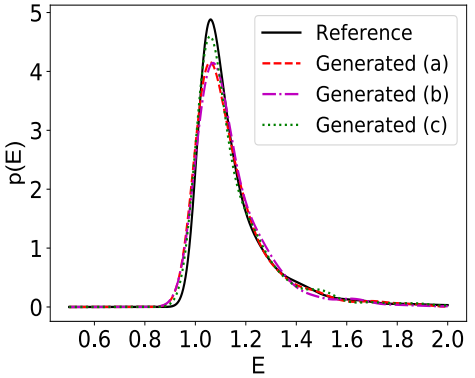
Stiffness Samples



Stiffness Spatial Correlation

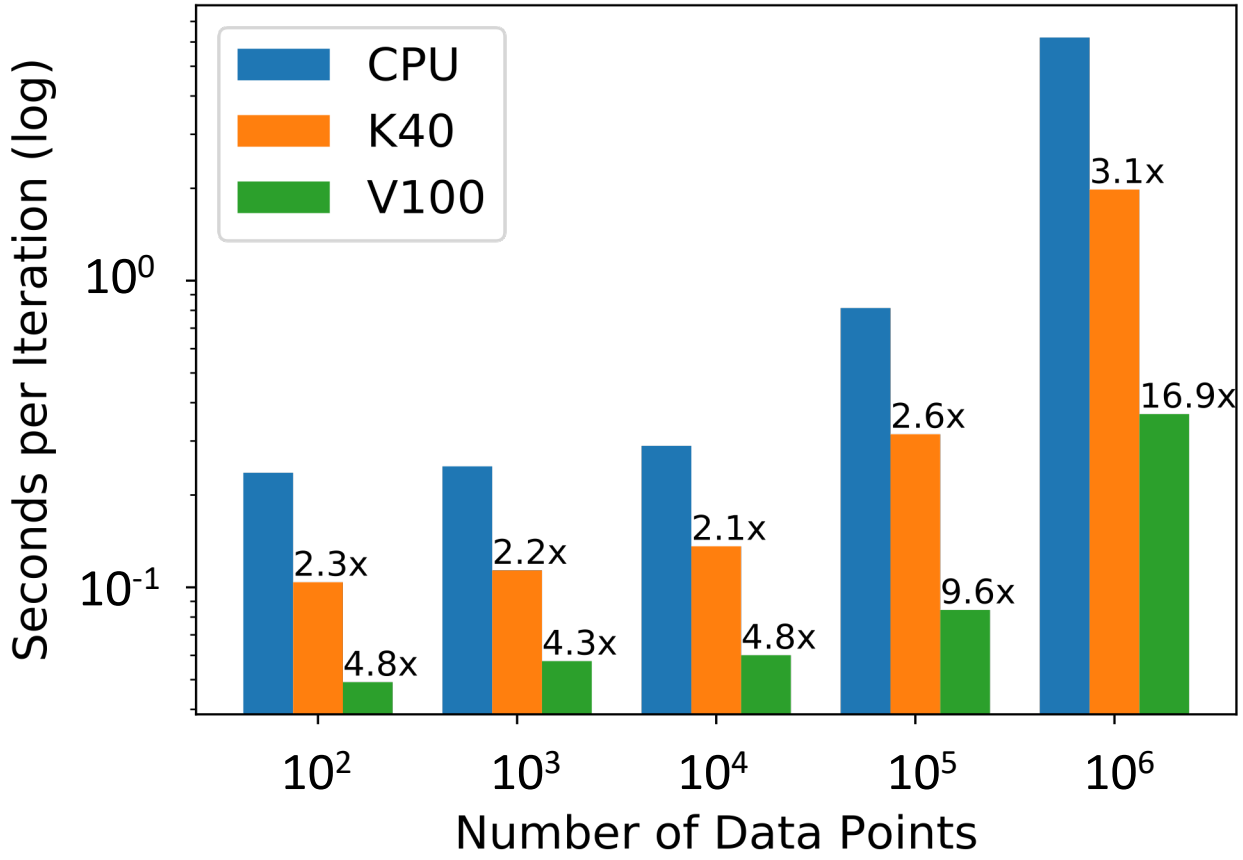


Stiffness Pointwise Probability Density



Numerical Proof-of-Concept

Performance of training PI-GANs on GPUs with the LaRC K Cluster



CPU: Dual socket 8 core 2.6 GHz Intel E5-2640v3 Haswell Node

GPUs:

- NVIDIA Tesla K40
- NVIDIA Tesla V100

➢ *GPU speedup is critical for scaling the approach to more realistic problems*

Conclusion

- *PIGANs for Virtual Mechanical Testing* made possible through HPCI support

Spring 2019	SIAM CSE Conference (**HPCI sponsored travel/training**)
Summer/Fall 2019	Summer/Fall 2019 – NIFS Intern Software Development (**HPCI funded internship**)
Fall 2019-Fall 2020	Fall 2019 - Fall 2020 – NASA IRAD Project awarded (***Project infusion***)
Spring 2020	Spring 2020 – NASA TM Published; Software Released (***Community impact***)

- More compute means more confidence with UQ for predictive science at NASA
- Scientific machine learning is an emerging paradigm that provides the advantages of physics-based & data-driven modeling for UQ
- PI-GAN IRAD under consideration for Phase II continuation; aiming for TRL↑ + infusion

Acknowledgments

Cara Leckey
HPCI Team/Community
LaRC K-Cluster
LaRC Training Office
NIFS Coordinators

Questions?

- Software:
  - Multi-Model Monte Carlo with Python (MXMCPy): github.com/nasa/MXMCPy
  - Sequential Monte Carlo with Python (SMCPy): github.com/nasa/SMCPy
  - Stochastic Reduced Order Models with Python (SROMPy): github.com/nasa/SROMPy
  - Symbolic Regression with Python (bingo): github.com/nasa/bingo
  - **PIGANs for material identification: github.com/nasa/pigans-material-id**
- Publications:
  - Stochastic reduced order models with Python (SROMPy). NASA/TM-2018-219824
  - Sequential Monte Carlo: Enabling real-time and high-fidelity prognostics. Proceedings of the Annual Conference of the PHM Society 2019.
  - Rapid uncertainty propagation for high-fidelity prognostics using SROMPy and Python. Proceedings of the Annual Conference of the PHM Society 2019.
  - Multi-model Monte Carlo with Python (MXMCPy). NASA/TM-2020-220585
  - **Inverse estimation of elastic modulus using physics-informed generative adversarial networks. NASA/TM-2020-5001300**