Foundational Methods for Foundation Models for Scientific Machine Learning

Michael W. Mahoney

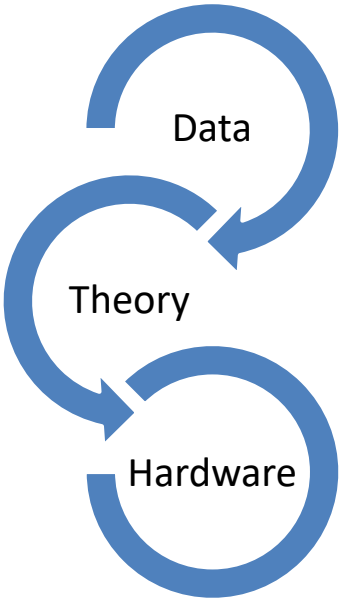
ICSI, LBNL, & Dept of Statistics, UC Berkeley

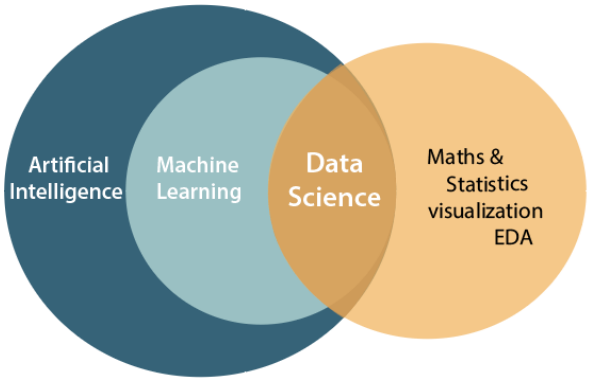
April 2024

Success Drivers of NNs in Modern ML

The key drivers for the success of NNs for Data Science applications are:

- **Data**: Abundance of training data (e.g., ImageNet)
- **Theory**: Better theory, especially for stochastic optimization
- **Hardware**: Ability to train on large data with GPU hardware





AI: Techniques that enable machines to have human-level of intelligence

ML: Methodologies to learn patterns in data and perform predictions

Data Science: Methods to draw insights from data (through math, stats, visualization, etc.)

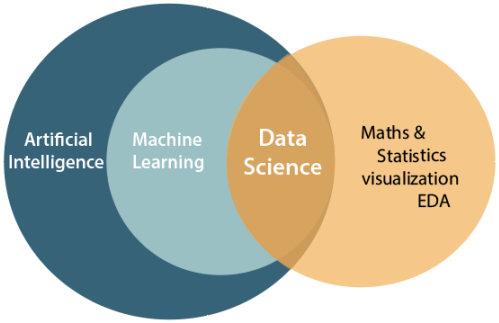
Illustration from H. Sri Kovela

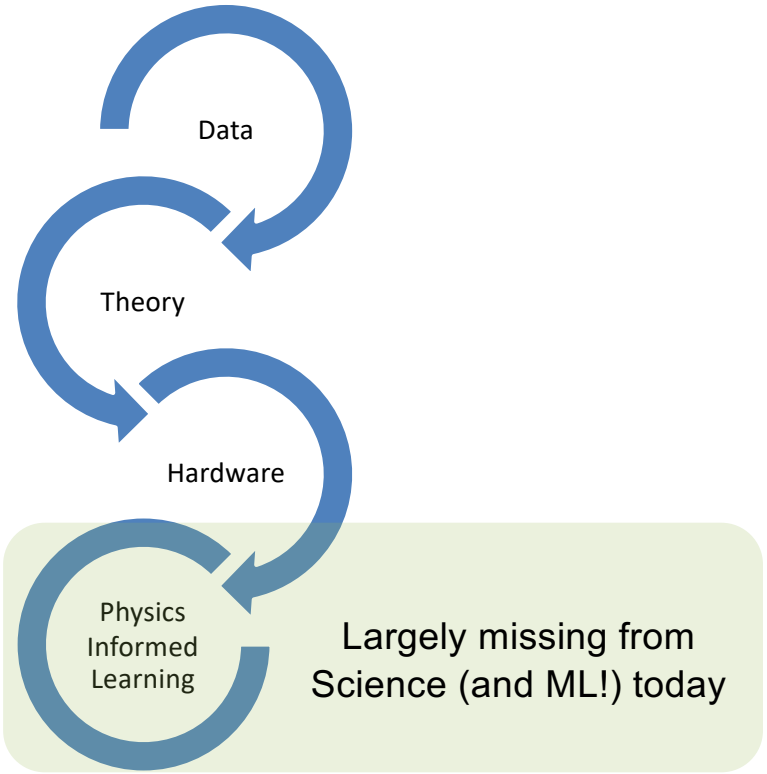
Physics Informed Learning?

Computational Science is an important tool that we can use to incorporate physical invariances into learning, but until recently it was missing from mainstream ML.

"**Computational Science** can **analyze past events** and **look into the future**. It can explore the effects of thousands of scenarios for or in lieu of actual experiment and be used to study events beyond the reach of expanding the boundaries of experimental science"
–Tinsley Oden, 2013

To make further progress in ML it is crucial that we incorporate computational science into learning.





J. Tinsley Oden's Commemorative Speech: "THE THIRD PILLAR: The Computational Revolution of Science and Engineering", Honda Prize, 2013.

Questions?

- Q0: What is a “Foundation Model”?
- Q1: Can we hope to train a “Foundation Model” for SciML?
- Q2: Would incorporating physical knowledge help? If so, how to do it?
- Q3: Foundations?
- Q4: Implementations?
- Q6: Applications?
- Q6: Looking forward?

Questions?

- **Q0: What is a "Foundation Model"?**
- Q1: Can we hope to train a "Foundation Model" for SciML?
- Q2: Would incorporating physical knowledge help? If so, how to do it?
- Q3: Foundations?
- Q4: Implementations?
- Q6: Applications?
- Q6: Looking forward?

What is a foundation model?

Informally:

- “a machine learning model that is trained on broad data such that it can be applied across a wide range of use cases”

HAI-CRFM Foundation Model Report:

- “any model that is trained on broad data (generally using self-supervision at scale) that can be adapted (e.g., fine-tuned) to a wide range of downstream tasks”

US President:

- “an AI model that is trained on broad data; generally uses self-supervision; contains at least tens of billions of parameters; is applicable across a wide range of contexts”

US House of Representatives:

- “... at least one billion parameters, ... exhibits, or could be easily modified to exhibit, high levels of performance at tasks that could pose a serious risk to security ..."

What is a foundation model?

General-purpose technologies that can support a diverse range of use cases.

Built using well-established techniques from ML:

- NNs, self-supervised learning, transfer learning, etc.

New paradigm in ML:

- general-purpose models are “reusable infrastructure,” instead of bespoke/one-off solutions
- building foundation models is highly resource-intensive (100M - 1B USD, people, data, compute)
- adapting a foundation model for a specific use case or using it directly is much less expensive.

Term was created/popularized by Stanford Institute for Human-Centered Artificial Intelligence (HAI) Center for Research on Foundation Models (CRFM):

- Bommasani et al. ``On the Opportunities and Risks of Foundation Models'' arXiv:2108.07258.

What is a foundation model?

Other possible names:

- large language model - too narrow, given the focus is not only language
- self-supervised model - too specific, to the training objective
- pretrained model - suggests the important action happened after pretraining
- foundational model - suggests the model provides fundamental principles

Foundation model:

- emphasize the intended function (i.e., amenability to subsequent further development) rather than modality, architecture, or implementation.

Early examples were language models (LMs) like Google's BERT and OpenAI's GPT-n series.

More recently, developed across a range of modalities:

- images; music; time series; robotic control; etc.
- Lots of areas of science: astronomy, radiology, climate, genomics, coding, mathematics, etc.

Questions?

- Q0: What is a “Foundation Model”?
- **Q1: Can we hope to train a “Foundation Model” for SciML?**
- Q2: Would incorporating physical knowledge help? If so, how to do it?
- Q3: Foundations?
- Q4: Implementations?
- Q6: Applications?
- Q6: Looking forward?

Just call ChatGPT? Or apply the M.O. of ML to Science?

Option 1:

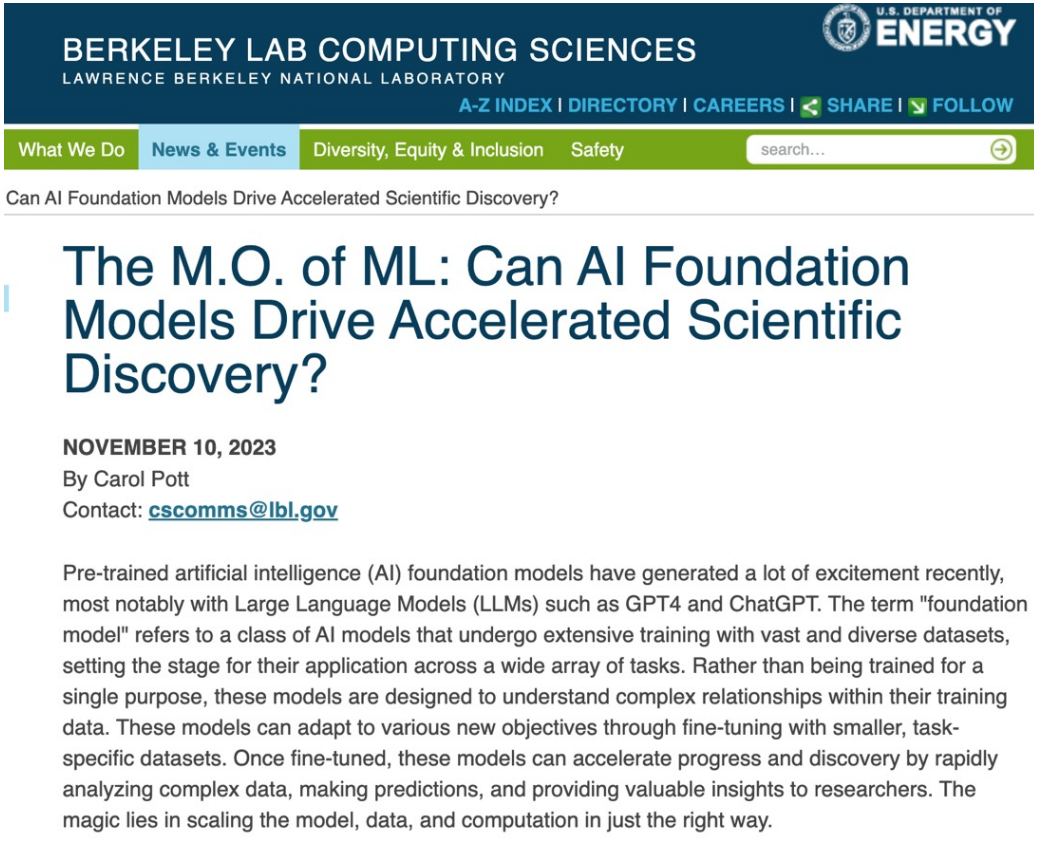
- Ask ChatGPT (or whatever LLM), post fine-tuning, to hypothesize new drugs, or what comes after the Top quark, or …

Option 2:

- Use ChatGPT embeddings in a model for some other scientific objective.

Option 3:

- Understand the *methodology of ML**
- Apply that methodology to Scientific data
- Multi-modal Scientific data could be text
- It could be simulation, experiment, etc.
- Incorporate spatio-temporal inductive biases into architecture and compute
- Develop foundations for SciML



BERKELEY LAB COMPUTING SCIENCES
LAWRENCE BERKELEY NATIONAL LABORATORY
U.S. DEPARTMENT OF ENERGY
A-Z INDEX | DIRECTORY | CAREERS | SHARE | FOLLOW
What We Do | News & Events | Diversity, Equity & Inclusion | Safety | search…
Can AI Foundation Models Drive Accelerated Scientific Discovery?

The M.O. of ML: Can AI Foundation Models Drive Accelerated Scientific Discovery?

NOVEMBER 10, 2023
By Carol Pott
Contact: cscomms@lbl.gov

Pre-trained artificial intelligence (AI) foundation models have generated a lot of excitement recently, most notably with Large Language Models (LLMs) such as GPT4 and ChatGPT. The term "foundation model" refers to a class of AI models that undergo extensive training with vast and diverse datasets, setting the stage for their application across a wide array of tasks. Rather than being trained for a single purpose, these models are designed to understand complex relationships within their training data. These models can adapt to various new objectives through fine-tuning with smaller, task-specific datasets. Once fine-tuned, these models can accelerate progress and discovery by rapidly analyzing complex data, making predictions, and providing valuable insights to researchers. The magic lies in scaling the model, data, and computation in just the right way.

**Scale data size, model size, and compute so none of them saturate, then transfer learn.*

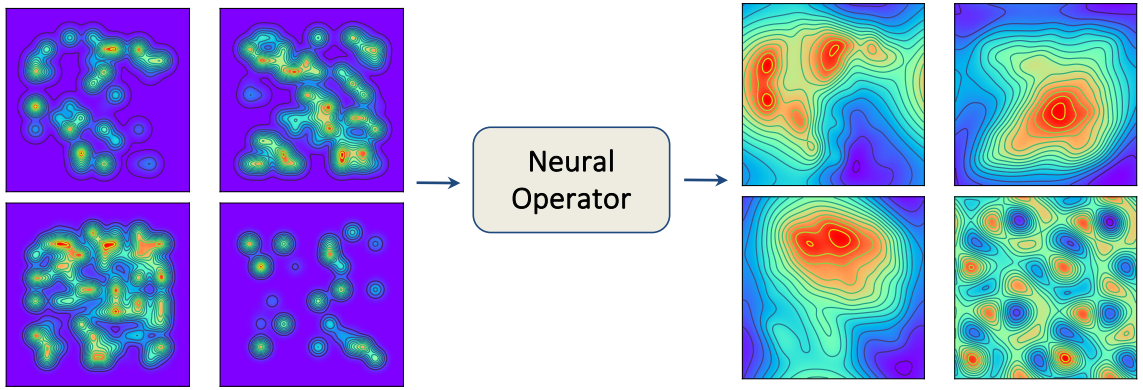
Foundation models for SciML?

Create and pre-train on diverse PDE systems

Vary/Sample all inputs (PDE coefficients, source functions, …)

Include multiple differential operators, predict PDE solution

$$\nabla \cdot \mathrm{K} \nabla u + v \cdot \nabla u + \ \ldots = f$$



Inputs

f, K, v, …

Outputs

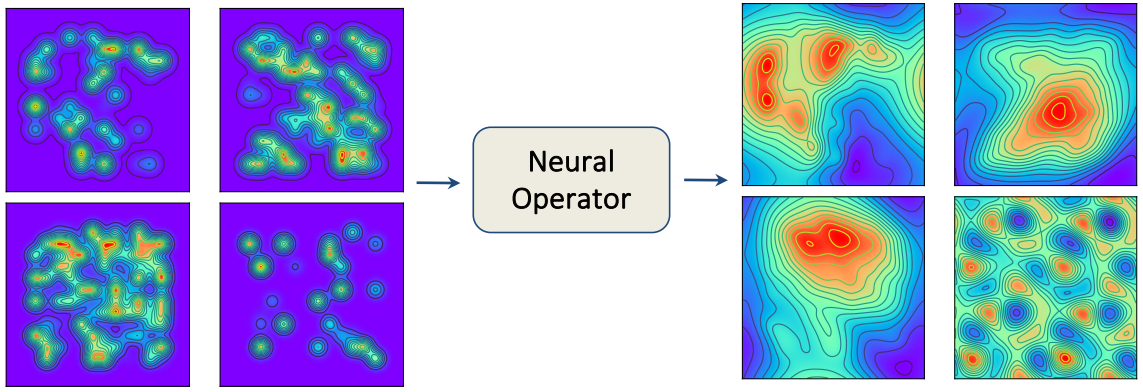
u

Foundation models for SciML?

Create and pre-train on diverse PDE systems

Vary/Sample all inputs (PDE coefficients, source functions, …)
Include multiple differential operators, predict PDE solution

$$\nabla \cdot \mathrm{K} \nabla u + v \cdot \nabla u + \ \ldots = f$$

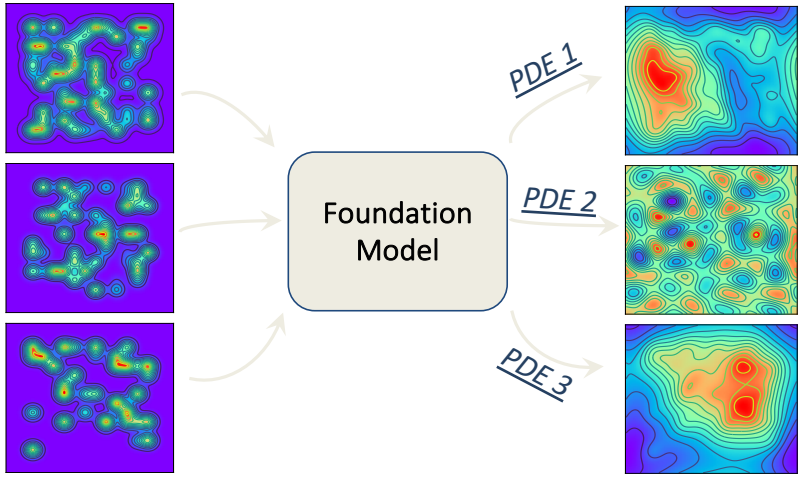


Inputs
f, K, v, …

Outputs
u

Foundation Models for SciML

Solve multiple systems using the same pre-trained model, outperforming training from scratch



Inputs
f, K, v, …

Outputs
u

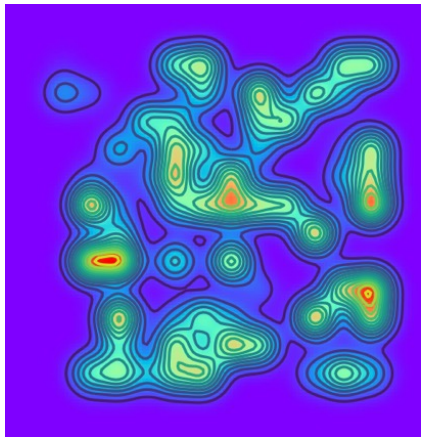
Characterize Scaling and Transfer Behavior

- Three common PDE systems: Poisson, Advection-Diffusion, Helmholtz
- Choose FNO as the baseline model for all experiments
- General strategy for any system:
  - Pre-train an FNO on a "large" pre-training dataset
  - Fine-tune on a downstream dataset and compare with training from scratch (from randomly initialized weights) on this dataset
- Analysis by controlling:
  - Downstream dataset availability
  - Model size (parameter count)
  - Extent of out-of-distribution (OOD) of downstream dataset using underlying physics

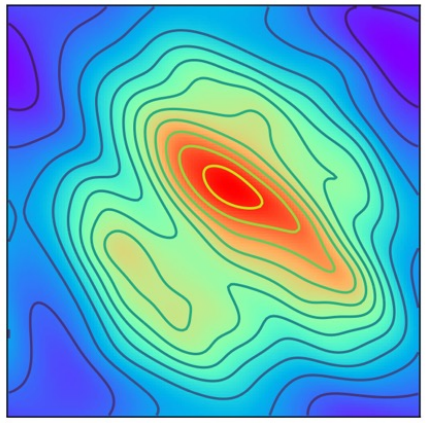
PDE system 1: Poisson's equation

$$-\text{div}\, \boldsymbol{K} \nabla u = f$$

- Inputs: diffusion tensor (K), forcing/source function (f)
- Outputs: solution to the PDE (u)
- Training dataset: sample diffusion tensors and source functions from a training distribution
  - Source: $f(\boldsymbol{x}) = \sum_{i=1}^{n_g} \phi_i(\boldsymbol{x}) p_i$
    - Linear combination of radial basis functions
  - Diffusion: Sample eigenvalues of diffusion tensor and diffusion direction
    - Control physics through eigenvalue



f

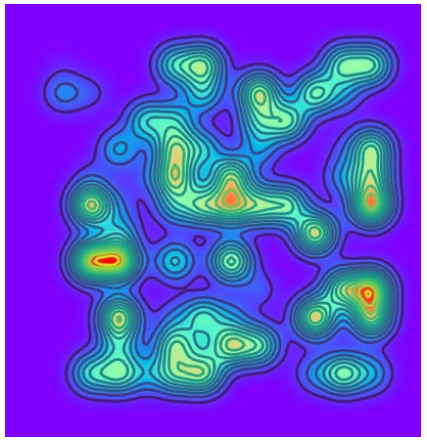


u

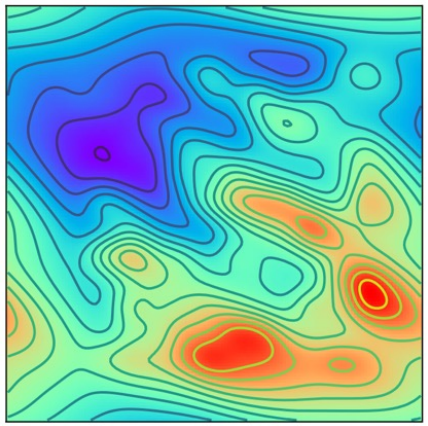
PDE System 2: Advection-Diffusion

$$-\text{div}\, \boldsymbol{K}\nabla u + \boldsymbol{v} \cdot \nabla u = f$$

- Inputs: diffusion tensor (K), advection vector (v) , forcing/source function (f)
- Outputs: solution to the PDE (u)
- Training dataset: sample diffusion tensors, advection vectors, source functions from a training distribution
  - Source and diffusion as in System 1
  - Sample advection directions and scales
    - Control physics through ratio of advection to diffusion



f

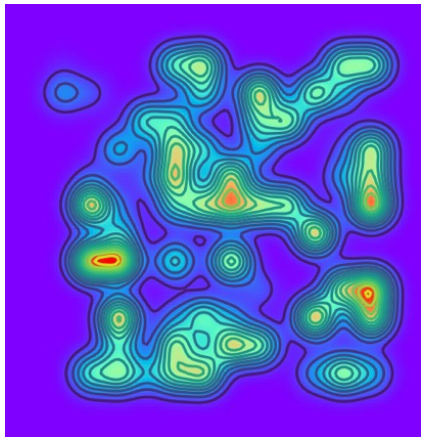


u

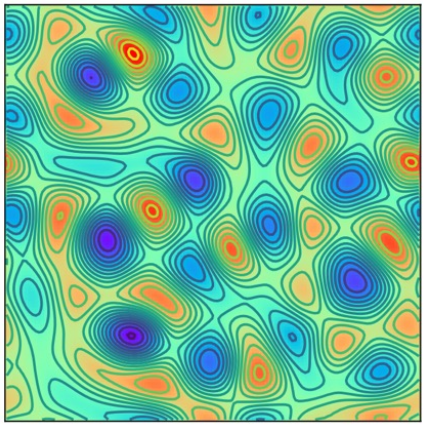
PDE System 3: Helmholtz

$$-\Delta u + \omega u = f$$

- Inputs: wavenumber (w), forcing/source function (f)
- Outputs: solution to the PDE (u)
- Training dataset: sample wavenumbers, source functions from a training distribution
  - Sources as in System 1/2
  - Sample wavenumbers
    - Control physics through wavenumber

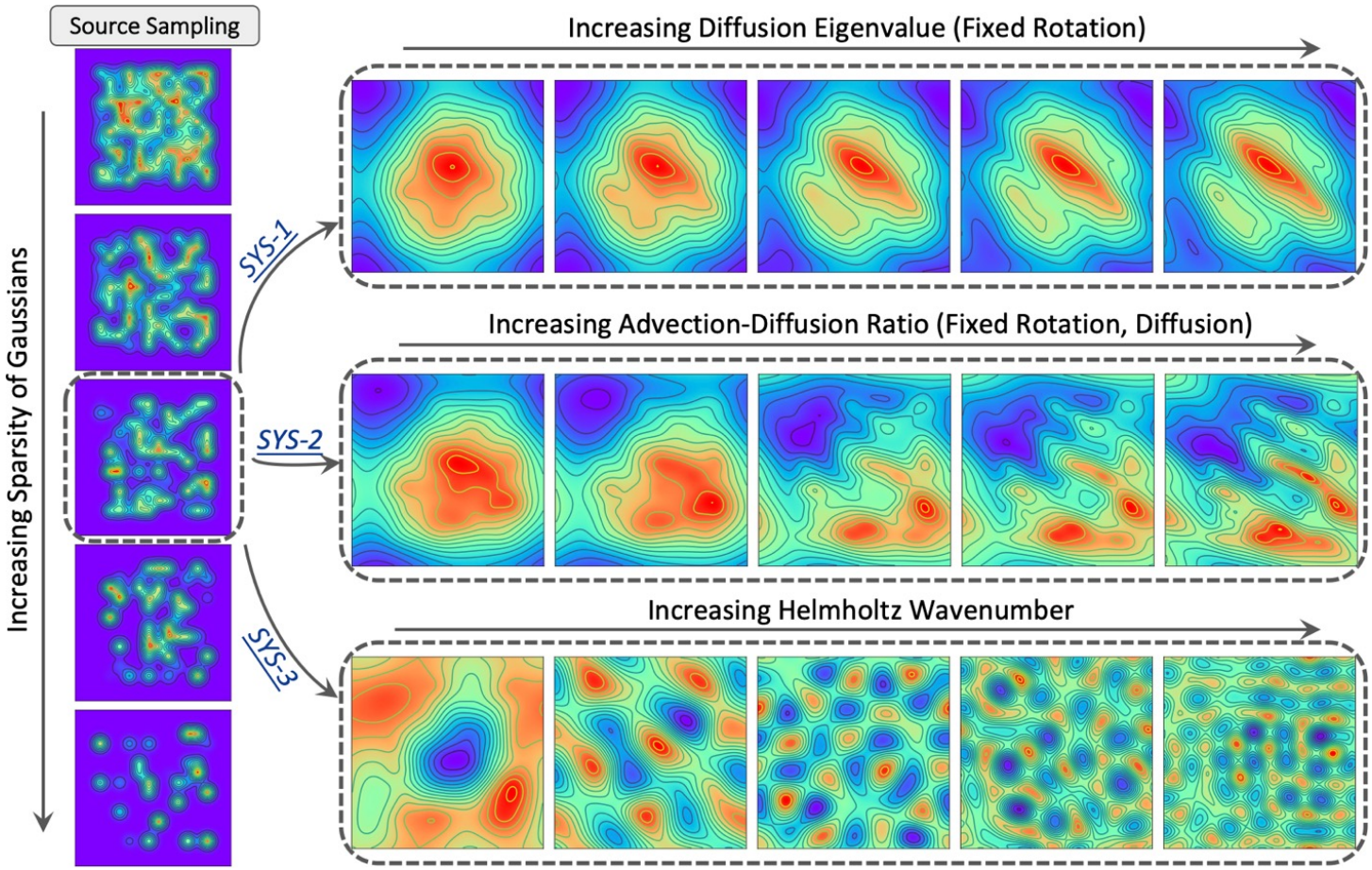


f



u

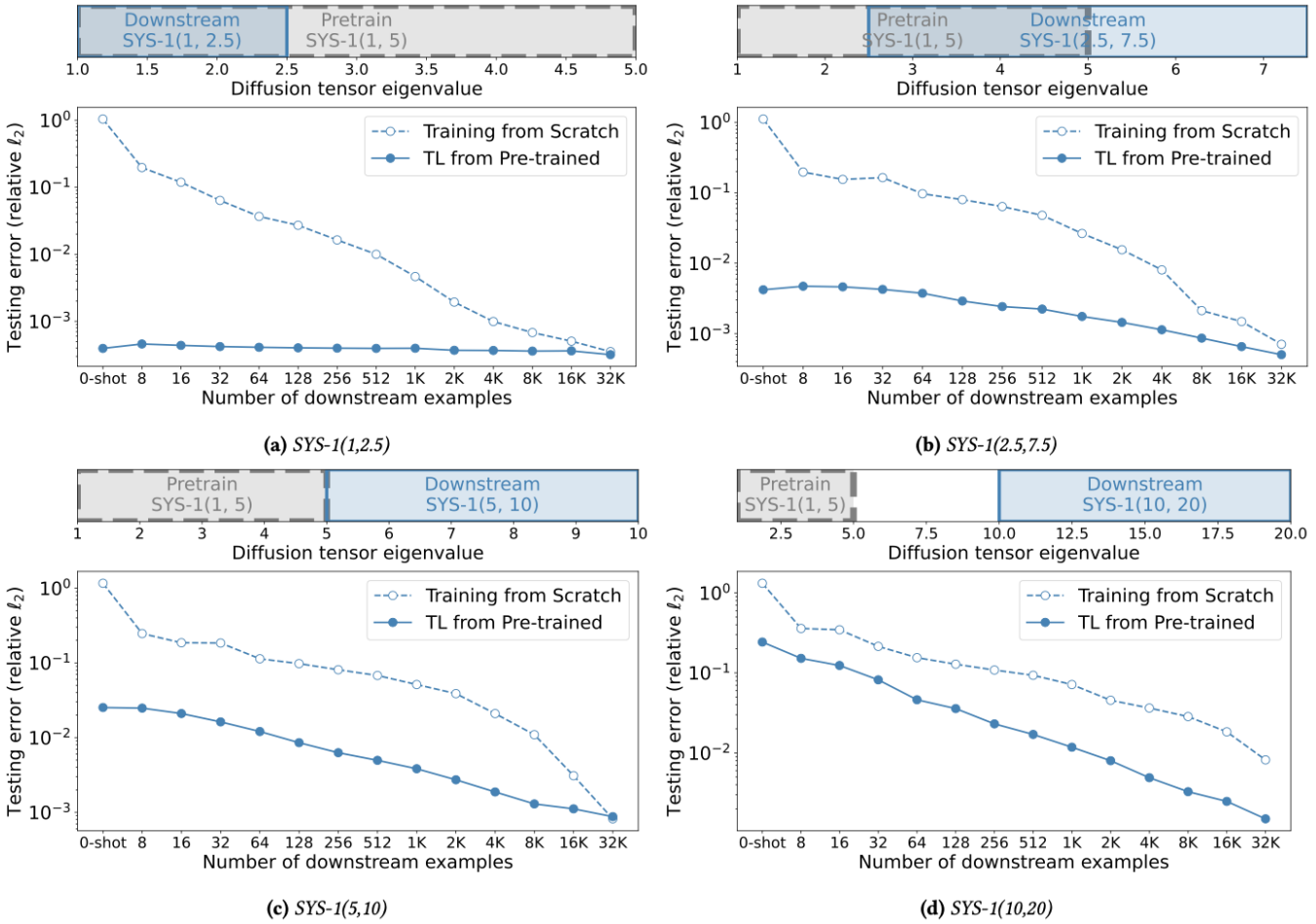
Physics control knobs for changing solutions



Pre-training and downstream task adaptation

- Pre-train: on 32K examples
  - MSE loss
- Downstream datasets: Adapt/TL to new dataset
  - with the same physics distribution
  - with systematically larger deviations from the pre-training distributions
- Adaptation:
  - Zero-shot: No fine-tuning, directly apply pre-trained model
  - Few-shot: Fine-tuning the full model with small number of examples
- Analysis:
  - Control #downstream examples, #model parameters
- Model Architecture is FNO:
  - Instance normalization: allows FNO to process inputs with different scales
  - Model size: scale up by controlling embedding dimension of the lifting layer and the hard threshold mode cutoff

Towards SciML foundations: OOD transfer behavior



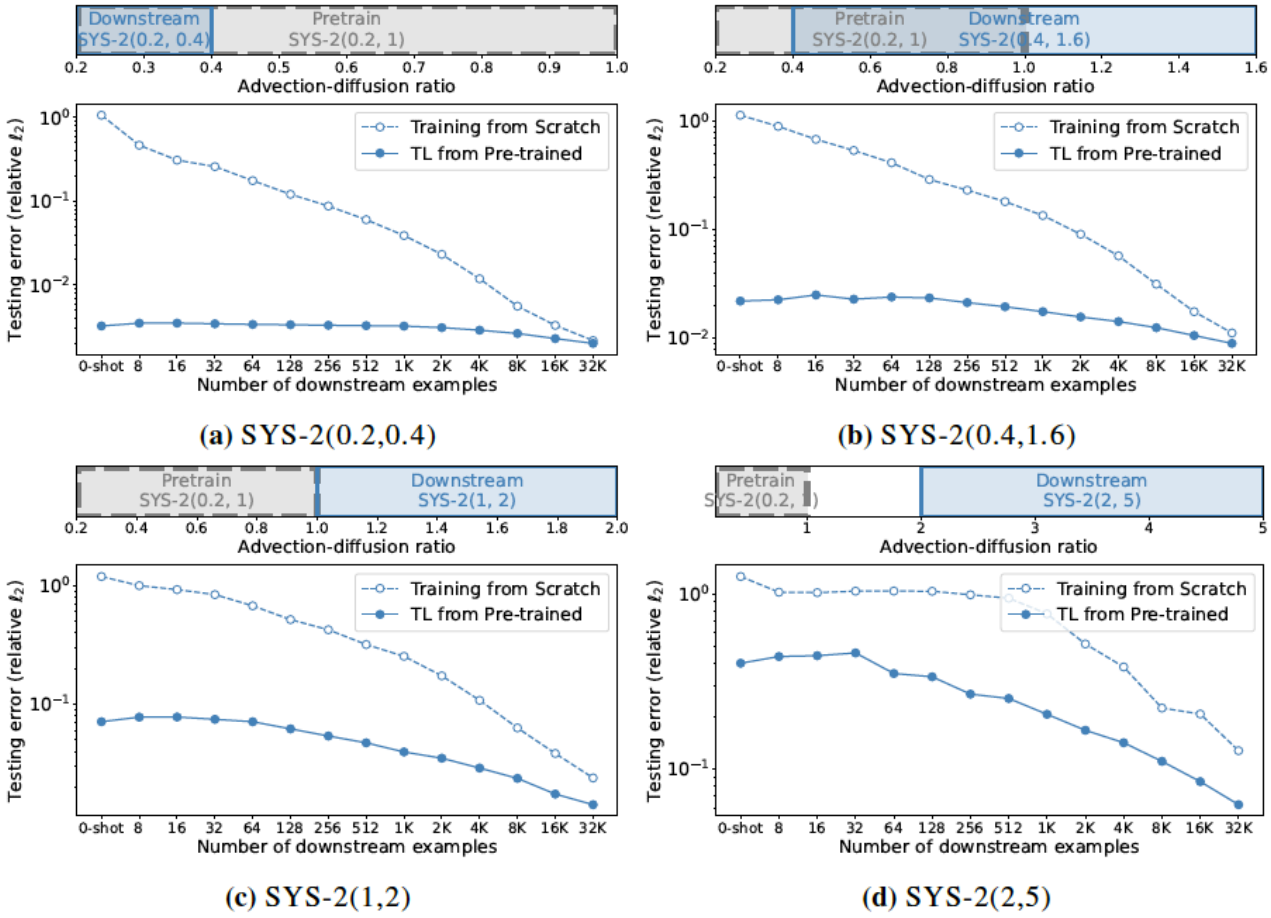
(a) *SYS-1(1,2.5)*

(b) *SYS-1(2.5,7.5)*

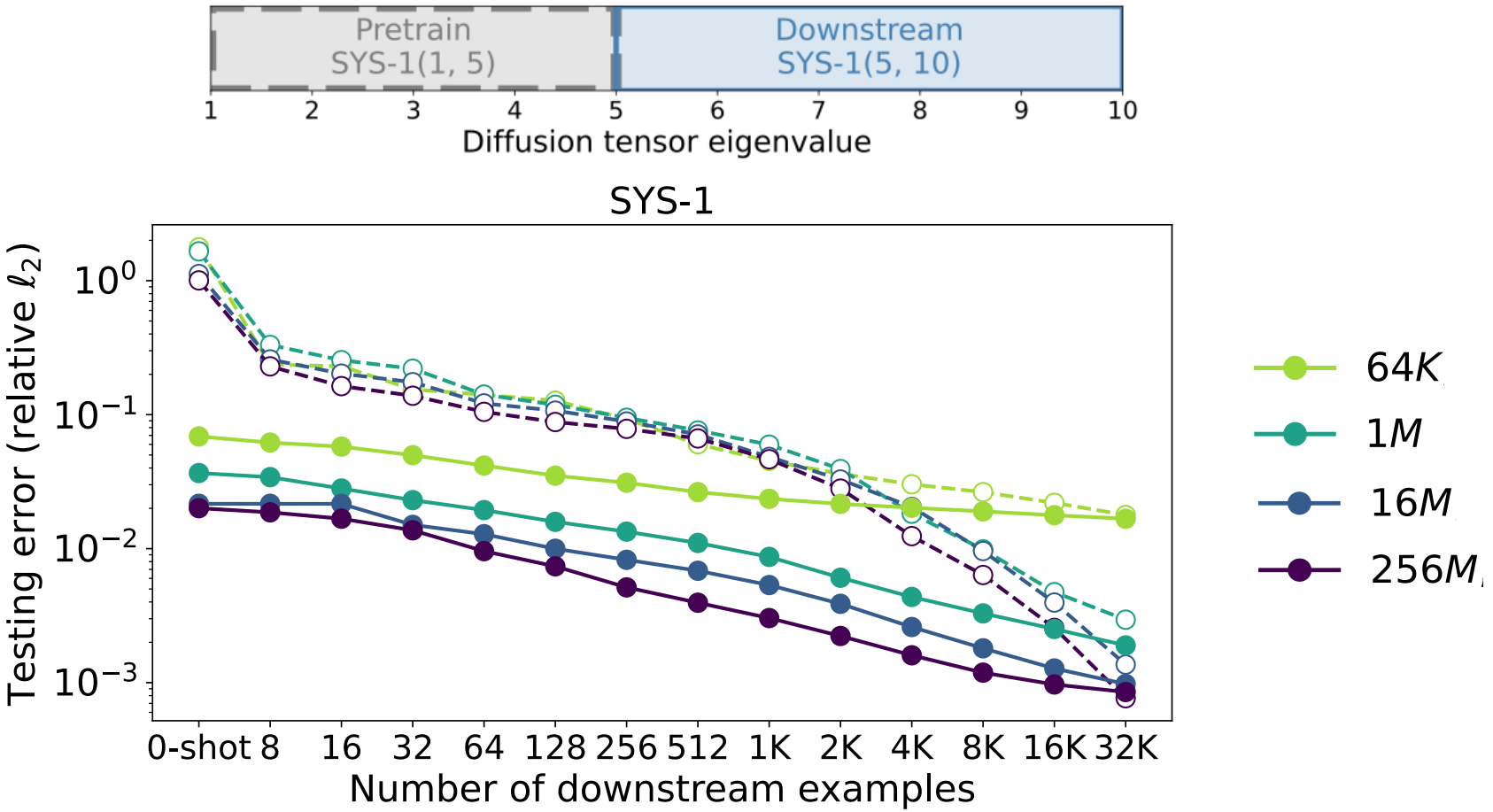
(c) *SYS-1(5,10)*

(d) *SYS-1(10,20)*

Other systems show similar OOD transfer behavior



Transfer behavior with model size



"Towards Foundation Models for Scientific Machine Learning: Characterizing Scaling and Transfer Behavior," Subramanian, et al., arXiv:2306.00258, NeurIPS23

[Submitted on 1 Jun 2023]

Towards Foundation Models for Scientific Machine Learning: Characterizing Scaling and Transfer Behavior

Shashank Subramanian, Peter Harrington, Kurt Keutzer, Wahid Bhimji, Dmitriy Morozov, Michael Mahoney, Amir Gholami

Pre-trained machine learning (ML) models have shown great performance for a wide range of applications, in particular in natural language processing (NLP) and computer vision (CV). Here, we study how pre-training could be used for scientific machine learning (SciML) applications, specifically in the context of transfer learning. We study the transfer behavior of these models as (i) the pre-trained model size is scaled, (ii) the downstream training dataset size is scaled, (iii) the physics parameters are systematically pushed out of distribution, and (iv) how a single model pre-trained on a mixture of different physics problems can be adapted to various downstream applications. We find that-when fine-tuned appropriately-transfer learning can help reach desired accuracy levels with orders of magnitude fewer downstream examples (across different tasks that can even be out-of-distribution) than training from scratch, with consistent behavior across a wide range of downstream examples. We also find that fine-tuning these models yields more performance gains as model size increases, compared to training from scratch on new downstream tasks. These results hold for a broad range of PDE learning tasks. All in all, our results demonstrate the potential of the "pre-train and fine-tune" paradigm for SciML problems, demonstrating a path towards building SciML foundation models. We open-source our code for reproducibility.

Comments: 16 pages, 11 figures

Subjects: **Machine Learning (cs.LG)**; Numerical Analysis (math.NA)

Cite as: arXiv:2306.00258 **[cs.LG]**

(or arXiv:2306.00258v1 **[cs.LG]** for this version)

https://doi.org/10.48550/arXiv.2306.00258

Many open problems/limitations

- **Going beyond simulation data to with "real" experimental/observational data**
  - Our data comes from numerical simulations of dynamical systems with known coefficients.
  - Need to test data from observations or (simulations + observations)

- **NN architecture**
  - We did not change the FNO model architecture.
  - We know it is not the right model for all kinds of SciML problems.

- **More complex PDEs**
  - 3D, time, space-time, high-resolution, multi-scale

- **Self-supervision in pre-training**
  - Physics losses, spatiotemporal masking (from CV)
  - Inductive biases to be continuous, well-posed w.r.t. constraints/discontinuities

Unsupervised pretraining and in-context learning?

Computer Science > Machine Learning

[Submitted on 24 Feb 2024]

Data-Efficient Operator Learning via Unsupervised Pretraining and In-Context Learning

Wuyang Chen, Jialin Song, Pu Ren, Shashank Subramanian, Dmitriy Morozov, Michael W. Mahoney

Recent years have witnessed the promise of coupling machine learning methods and physical domain-specific insight for solving scientific problems based on partial differential equations (PDEs). However, being data-intensive, these methods still require a large amount of PDE data. This reintroduces the need for expensive numerical PDE solutions, partially undermining the original goal of avoiding these expensive simulations. In this work, seeking data efficiency, we design unsupervised pretraining and in-context learning methods for PDE operator learning. To reduce the need for training data with simulated solutions, we pretrain neural operators on unlabeled PDE data using reconstruction-based proxy tasks. To improve out-of-distribution performance, we further assist neural operators in flexibly leveraging in-context learning methods, without incurring extra training costs or designs. Extensive empirical evaluations on a diverse set of PDEs demonstrate that our method is highly data-efficient, more generalizable, and even outperforms conventional vision-pretrained models.

Subjects: **Machine Learning (cs.LG)**; Machine Learning (stat.ML)
Cite as: arXiv:2402.15734 **[cs.LG]**
(or arXiv:2402.15734v1 **[cs.LG]** for this version)
https://doi.org/10.48550/arXiv.2402.15734

"Data-Efficient Operator Learning via Unsupervised Pretraining and In-Context Learning," Chen, Song, Ren, Subramanian, Morozov, and Mahoney, arXiv:2402.15734

Questions?

- Q0: What is a “Foundation Model”?
- Q1: Can we hope to train a “Foundation Model” for SciML?
- **Q2: Would incorporating physical knowledge help? If so, how to do it?**
- Q3: Foundations?
- Q4: Implementations?
- Q6: Applications?
- Q6: Looking forward?

Combining domain-driven and data-driven models?

Characterizing possible failure modes in physics-informed neural networks

Aditi S. Krishnapriyan[*,1,2], **Amir Gholami**[*,2],
Shandian Zhe[3], **Robert M. Kirby**[3], **Michael W. Mahoney**[2,4]
[1]Lawrence Berkeley National Laboratory, [2]University of California, Berkeley,
[3]University of Utah, [4]International Computer Science Institute
{aditik1, amirgh, mahoneymw}@berkeley.edu, {zhe, kirby}@cs.utah.edu

Abstract

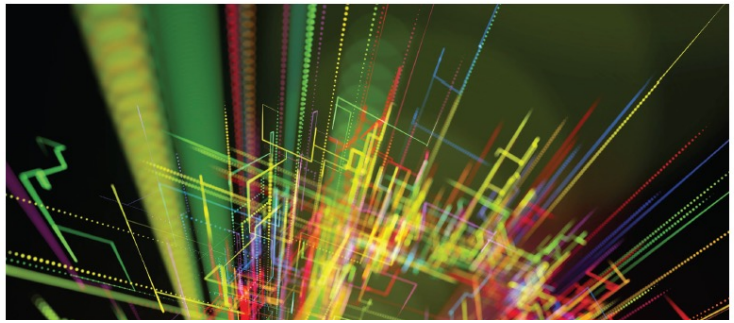
Recent work in scientific machine learning has developed so-called physics-

Science | DOI:10.1145/3524015 Chris Edwards

Neural Networks Learn to Speed Up Simulations

Physics-informed machine learning is gaining attention, but suffers from training issues.

PHYSICAL SCIENTISTS AND engineering research and development (R&D) teams are embracing neural networks in attempts to accelerate their simulations. From quantum mechanics to the prediction of blood flow in the body, numerous teams have reported on speedups in simulation by swapping conventional finite-element solvers for models trained on various combinations of experimental and synthetic data.



Physical Laws as Additional Sources of Data?

- NNs **require a lot of data** to train
- In addition to data, we know laws/constraints that govern physical phenomena
  - Conservation of mass, momentum, energy
  - We often have approximate models that can predict the system behavior
- Uses of physical knowledge
  - Simulation data
  - Add to the loss – as a hard or soft constraint
  - Add inductive biases to the architecture

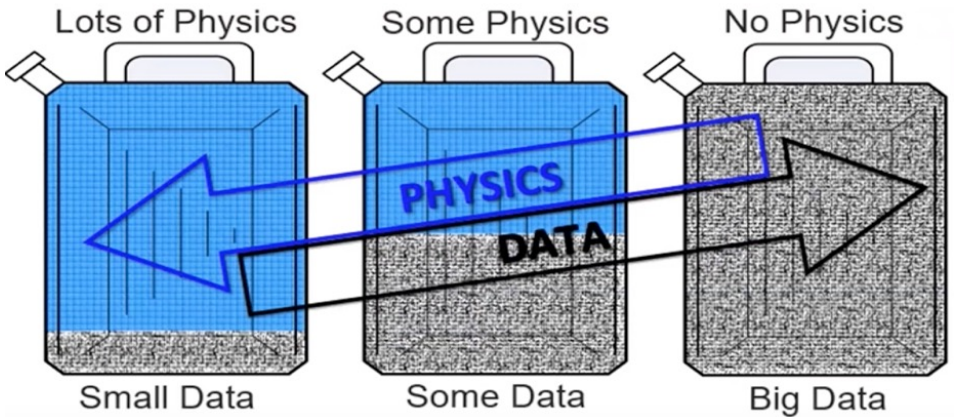


Illustration Credit: Prof. Karniadakis

Methods for Incorporating Physics into Learning

- **Method 1:** Enforce physical laws as hard constraints either in:
  - NN Architecture: still an open problem
  - Optimization: very difficult to train the NN with such constraints

- **Method 2:** Train on lots of data and let NN learn physics based operators
  - Neural Operator like methods

- **Method 3:** Use penalty methods and add the PDE residual to the loss.
  - PINN like methods: *very* easy to implement with any NN architecture

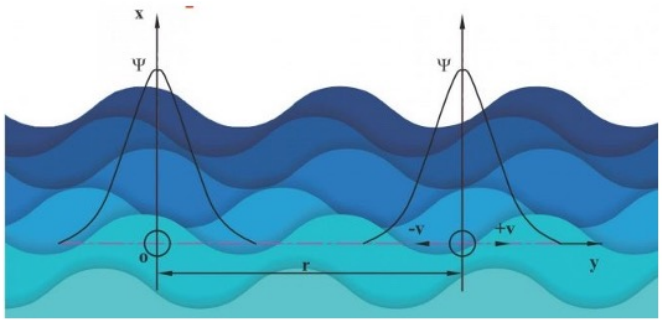
- **Method 4:** Use a combination of Neural Operator and PINNs
  - Uses a combination of observation data points as well as physical constraints added as soft penalty to the loss

Xu K, Darve E. Physics constrained learning for data-driven inverse modeling from sparse observations. arXiv preprint arXiv:2002.10521. 2020 Feb 24.
Raissi M, Perdikaris P, Karniadakis GE. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics. 2019 Feb 1;378:686-707.
*[Weinan et al. 2017; **Raissi et al. 2019**; Rackauckas et al. 2020; Hennigh et al. 2021; Lu et al. 2021]*
Li et al. 2021]
Etc.

Sounds good … but this is not the entire story

- There are a lot of subtleties in adding a soft-constraint:
  - Methods actually do not work so well, even for simple problems
- To study this, we chose three families of PDEs:
  - Advection (aka wave equation)
  - Reaction
  - Reaction-Diffusion
- Soft-constrained PINN-like models fail to learn relevant physics in all these cases
  - Since there are many moving parts to ML training
  - The relevant ML methodologies don't play well with scientific methodologies
  - Reasons for the failure modes are interesting and informative

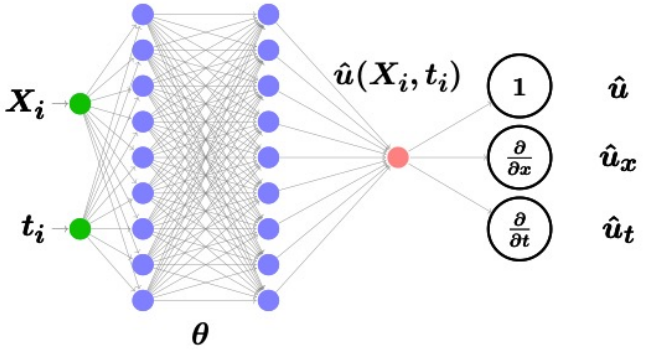
Advection Equation



$$\frac{\partial u}{\partial t} + \beta \frac{\partial u}{\partial x} = 0, \quad x \in \Omega,\ t \in [0, T],$$

Initial condition: $u(x, 0) = sin(x),$

Periodic boundary conditions: $u(0, t) = u(2\pi, t)$



$$\min_{\theta} \mathcal{L} = \lambda_{\mathcal{F}} \|\hat{u}_t + \beta \hat{u}_x\|_2^2 \quad \text{PDE Residual}$$

$$+ \|\hat{u}(x, 0) - sin(x)\|_2^2 \quad \text{Initial Condition}$$

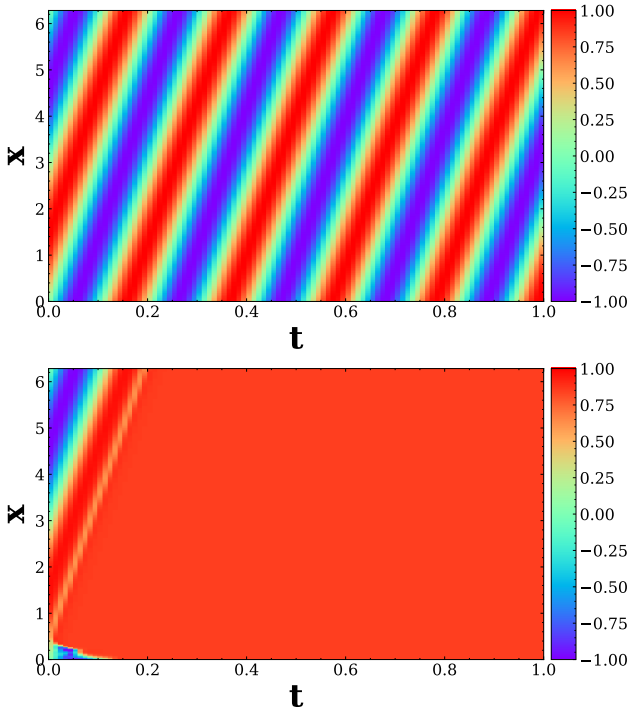
$$+ \|\hat{u}(x = 2\pi) - \hat{u}(x = 0)\|_2^2 \quad \text{Boundary Condition}$$

Krishnapriyan AS, **Gholami* A**, Zhe S, Kirby RM, Mahoney MW. Characterizing possible failure modes in physics-informed neural networks. NeurIPS, 2021.*

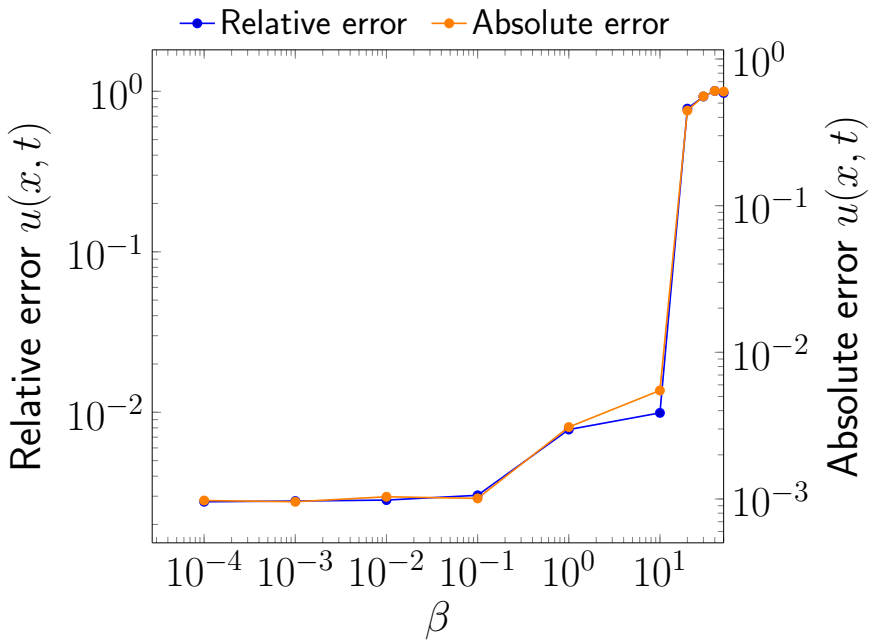
PINN can fail to learn Advection*

Exact Solution

PINN Prediction



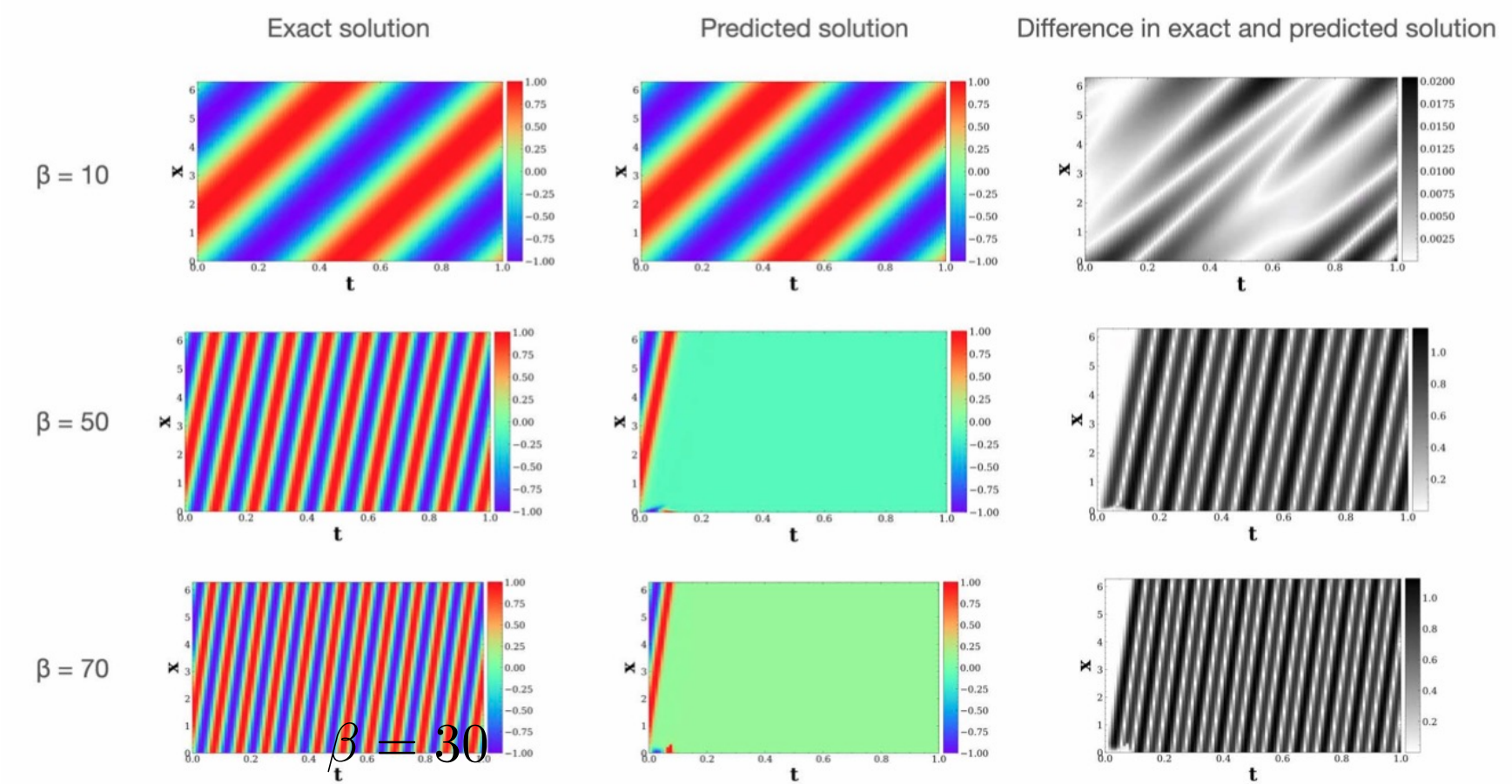
$\beta = 30$



*Nothing special about advection: same holds true for other PDEs..

*Krishnapriyan AS, **Gholami*A**, Zhe S, Kirby RM, Mahoney MW. Characterizing possible failure modes in physics-informed neural networks. NeurIPS, 2021.*

PINN can fail to learn Advection



*Nothing special about advection: same holds true for other PDEs..

*Krishnapriyan AS, **Gholami*A**, Zhe S, Kirby RM, Mahoney MW. Characterizing possible failure modes in physics-informed neural networks. NeurIPS, 2021.*

Training: Optimization Challenges with PINNs

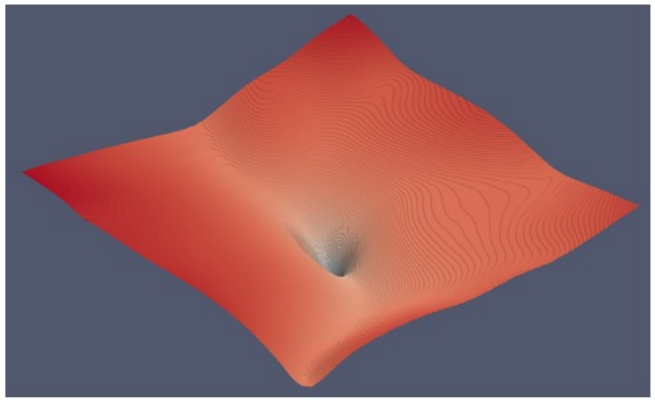
Data Loss Function:

$$\mathcal{L}_u = \|\hat{u} - u\|_2^2$$

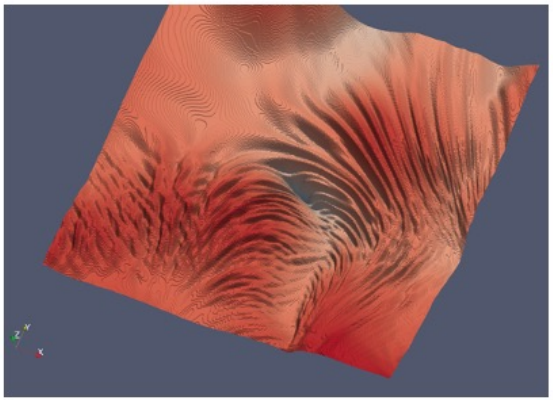
Physics Loss Function:

$$\mathcal{L}_{\mathcal{F}} = \|\hat{u}_t + \hat{u}\hat{u}_x - \hat{u}_{xx}\|_2^2$$

$$\min_{\theta} \mathcal{L} = \mathcal{L}_u + \lambda_{\mathcal{F}} \mathcal{L}_{\mathcal{F}}$$



Without Physics Loss



With Physics Loss

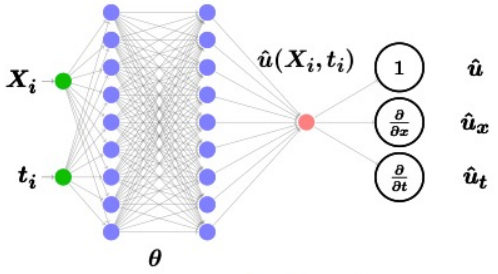
Illustration credit: Roman Amici, Mike Kirby

Krishnapriyan AS, **Gholami* A**, Zhe S, Kirby RM, Mahoney MW. Characterizing possible failure modes in physics-informed neural networks. NeurIPS, 2021.*

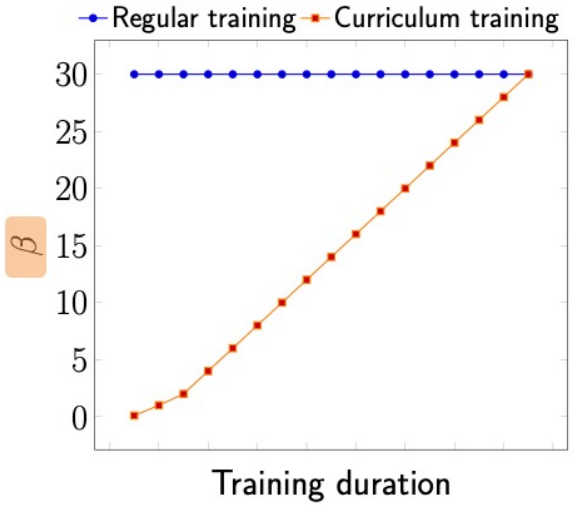
Rethinking PINN Training: Curriculum Learning

- The main idea is to start the training with **simple physical constraints** and introduce the complexities iteratively throughout learning
- First let the NN learn the simple problems, before penalizing it for learning the exact PDE

Example: For the advection equation, we start to train the NN with very small velocities, and slowly increase the velocity to the target one



$$\min_{\theta} \mathcal{L} = \lambda_{\mathcal{F}} \|\hat{u}_t + \beta \hat{u}_x\|_2^2$$
$$+ \|\hat{u}(x,0) - sin(x)\|_2^2$$
$$+ \|\hat{u}(x=2\pi) - \hat{u}(x=0)\|_2^2$$

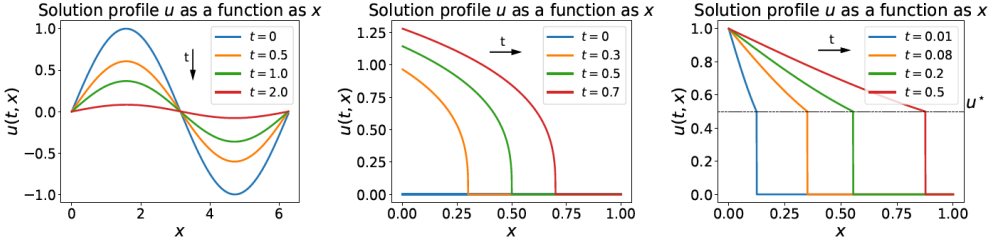


Questions?

- Q0: What is a “Foundation Model”?
- Q1: Can we hope to train a “Foundation Model” for SciML?
- Q2: Would incorporating physical knowledge help? If so, how to do it?
- **Q3: Foundations?**
- Q4: Implementations?
- Q6: Applications?
- Q6: Looking forward?

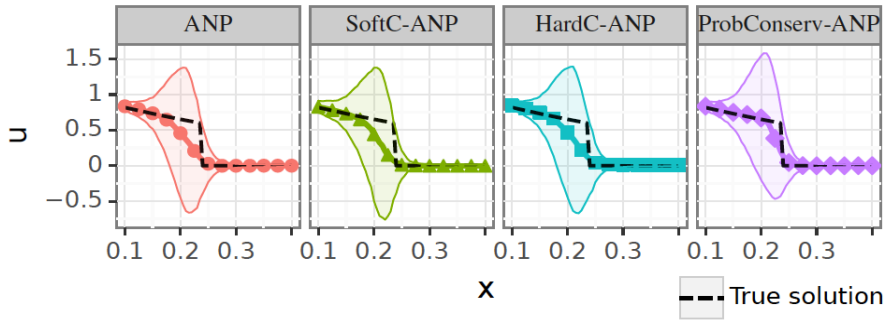
One way to address failure modes: ProbConserve

1. Compute mean and variance estimates
2. Update model (with oblique projection, depending on heteroscedasticity structure)
3. Good for sharp discontinuities

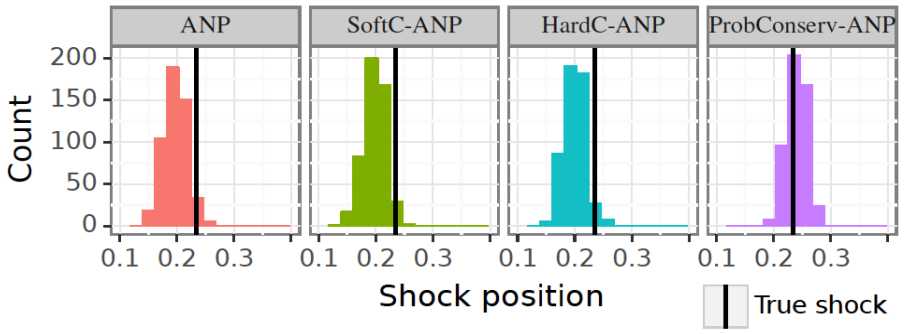


(a) easy: Diffusion equation ($k = 1$) (b) medium: PME ($k(u) = u^3$) (c) hard: Stefan (discont. $k(u)$)

Figure 1: Illustration of the "easy-to-hard" paradigm for PDEs, for the GPME family of conservation equations: (a) "easy" parabolic smooth (diffusion equation) solutions, with constant parameter $k(u) = k \equiv 1$; (b) "medium" degenerate parabolic PME solutions, with nonlinear monomial coefficient $k(u) = u^m$, with parameter $m = 3$ here; and (c) "hard" hyperbolic-like (degenerate parabolic) sharp solutions (Stefan equation) with nonlinear step-function coefficient $k(u) = \mathbf{1}_{u \geq u^\star}$, where $\mathbf{1}_{\mathcal{E}}$ is an indicator function for event $\mathcal{E}$.



(a) Solution profile.



(b) Posterior of the shock position.

"Learning Physical Models that Can Respect Conservation Laws," Hansen, Maddix, Alizadeh, Gupta, and Mahoney, arXiv:2302.11002, ICML23, Physica D (2024)
"Using Uncertainty Quantification to Characterize and Improve Out-of-Domain Learning for PDEs," Mouli, Maddix, Alizadeh, Gupta, Stuart, Mahoney, Wang, arXiv:2403.10642

Foundations more generally

Illustrative recent proof-of-principle directions:

- ContinuousNet: “numerical” convergence tests
- Traditional vs Modern ML UQ: Over- vs under-parameterized models
- Weight diagnostics: WeightWatcher Analysis and HTSR
- Time Series: LEM, ConvLEM, Chronos

Foundations more generally: ContinuousNet

1. Convergence test based on numerical analysis theory
2. Verifies whether a model has learned an underlying continuous dynamics
3. Good for super-resolution, iterative dynamics, etc.
4. Applies to NNs, SINDy, etc.

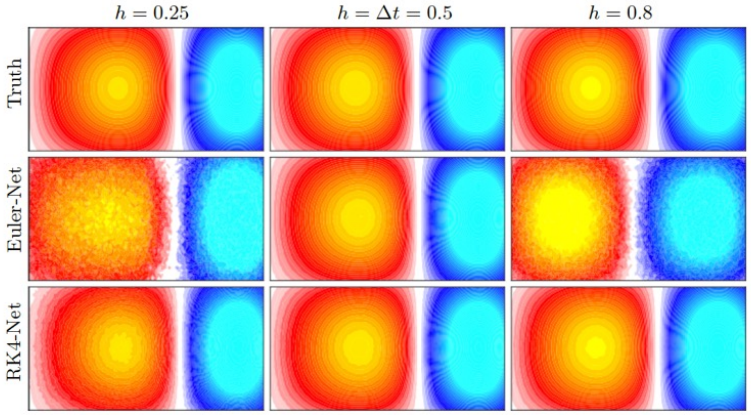
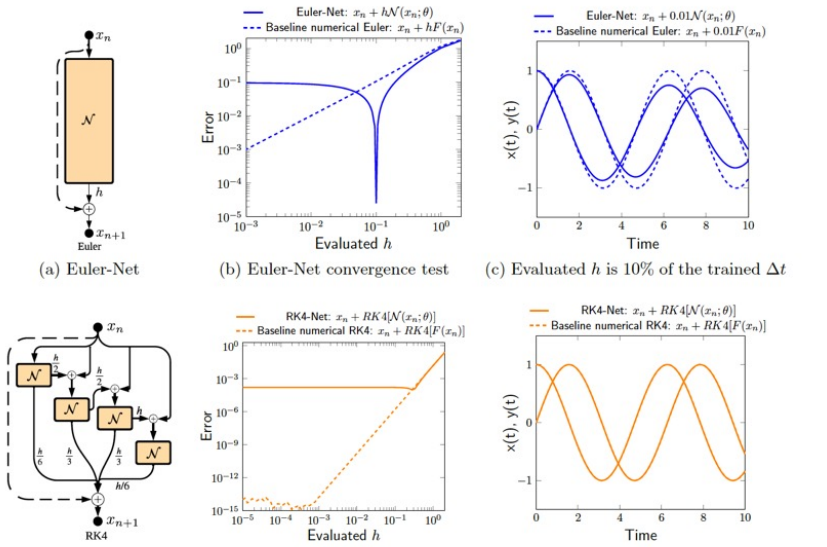


Figure 4: ***Double gyre fluid flow: Reconstructing fine-scale flow fields from coarse training***



(a) Euler-Net (b) Euler-Net convergence test (c) Evaluated h is 10% of the trained Δt

(d) RK4-Net (e) RK4-Net convergence test (f) Evaluated h is 10% of the trained Δt

Figure 2: ***Illustration of our convergence test with different ODE-Nets.*** *(a) Schematic of an ODE-Net*

"Learning continuous models for continuous physics," Krishnapriyan, Queiruga, Erichson, and Mahoney, arXiv:2202.08494, Comm Phys (2023)
"Continuous-in-Depth Neural Networks," Queiruga, Erichson, Taylor, and Mahoney, arXiv:2008.02389

Foundations more generally: traditional vs modern ML UQ

Traditional UQ versus Modern UQ in overparameterized vs underparameterized models

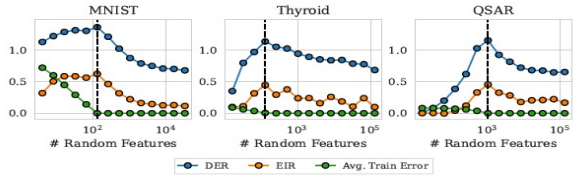


Figure 3: **Bagged random feature classifiers.** Blacked dashed line represents the interpolation threshold. Across all tasks, DER and EIR are maximized at this point, and then decrease thereafter.

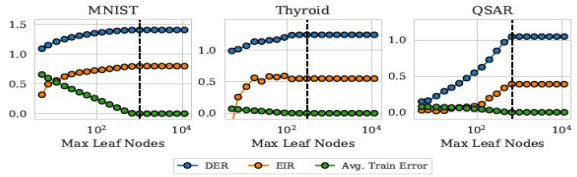
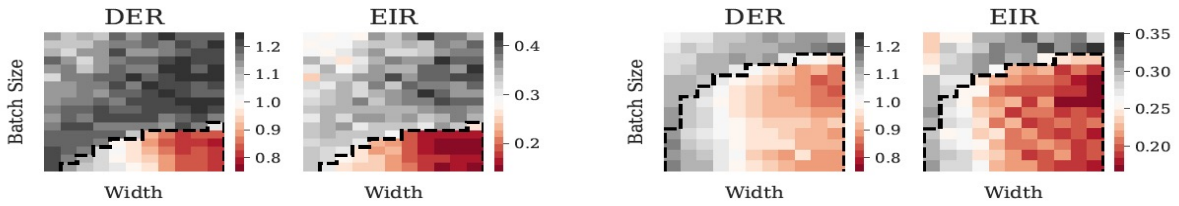


Figure 4: **Random forest classifiers.** Blacked dashed line represents the interpolation threshold. Across all tasks, DER and EIR are maximized at this point, and then remain constant thereafter.



(a) **Without LR decay.** (b) **With LR decay.**

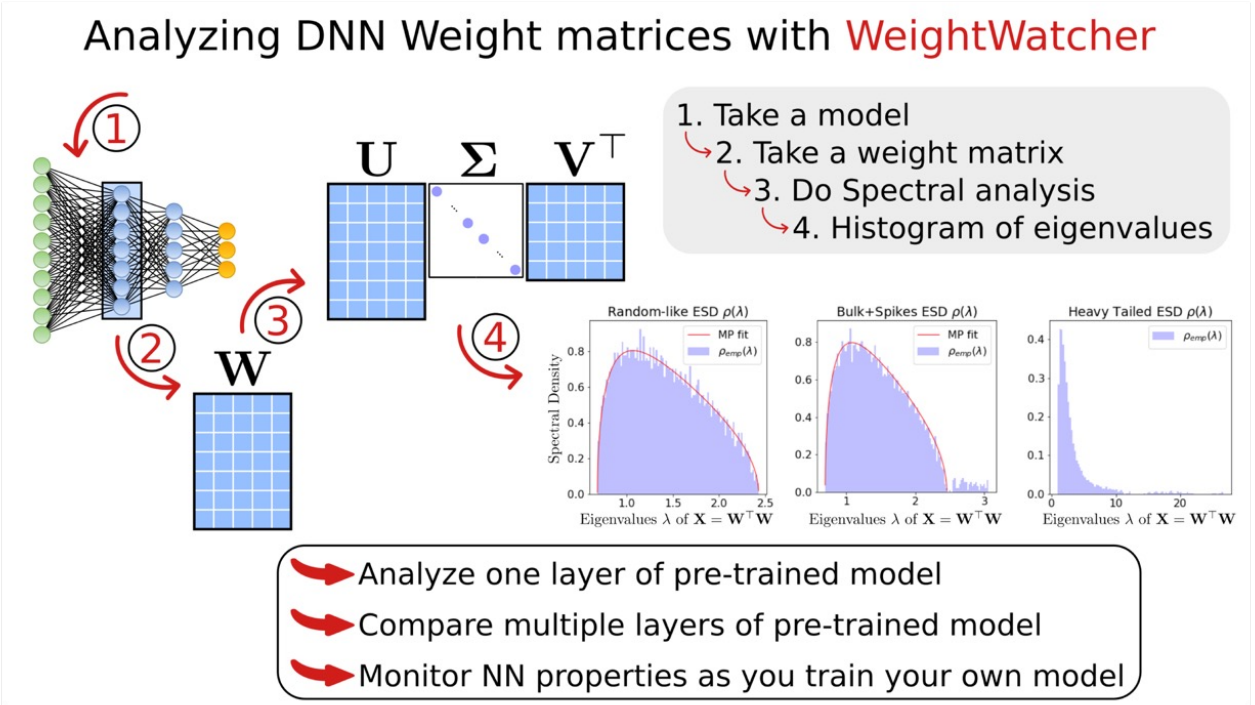
Figure 5: **Large scale studies of deep ensembles on ResNet18/CIFAR-10.** We plot the DER and EIR across a range of hyper-parameters, for two training settings: one with learning rate decay, and one without. The black dashed line indicates the *interpolation threshold*, i.e., the curve below which individual models achieve exactly zero training error. Observe that interpolating ensembles attain distinctly lower EIR than non-interpolating ensembles, and correspondingly have low DER (< 1), compared to non-interpolating ensembles with high DER (> 1).

"The Interpolating Information Criterion for Overparameterized Models," Hodgkinson, van der Heide, Salomone, Roosta, and Mahoney, arXiv:2307.07785
"When are ensembles really effective?," Theisen, Kim, Yang, Hodgkinson, and Mahoney, arXiv:2305.12313, NeurIPS23
"Monotonicity and Double Descent in Uncertainty Estimation with Gaussian Processes," Hodgkinson, van der Heide, Roosta, and Mahoney, arXiv:2210.07612, ICML23

Foundations more generally: weight diagnostics

Use methods from disordered systems theory, random matrix theory and statistical physics to diagnose practical problems in state-of-the art neural networks

- "Predicting trends in the quality of state-of-the-art neural networks without access to training or testing data," Martin, Peng, and Mahoney, arXiv:2002.06716 (2020)
- "Statistical Mechanics Methods for Discovering Knowledge from Modern Production Quality Neural Networks, Martin and Mahoney," KDD (2019)
- "Traditional and Heavy-Tailed Self Regularization in Neural Network Models, Martin and Mahoney," ICML (2019)
- "Heavy-Tailed Universality Predicts Trends in Test Accuracies for Very Large Pre-Trained Deep Neural Networks," Martin and Mahoney, SDM (2019)
- "Implicit Self-Regularization in Deep Neural Networks: Evidence from Random Matrix Theory and Implications for Learning," Martin and Mahoney, arXiv:1810.01075 (2018)
- (https://github.com/CalculatedContent/ww-trends-2020)



Foundations more generally: Time Series

Published as a conference paper at ICLR 2022

LONG EXPRESSIVE MEMORY FOR SEQUENCE MODELING

T. Konstantin Rusch
ETH Zürich
trusch@ethz.ch

Siddhartha Mishra
ETH Zürich
smishra@ethz.ch

N. Benjamin Erichson
University of Pittsburgh
erichson@pitt.edu

Michael W. Mahoney
ICSI and UC Berkeley
mmahoney@stat.berkeley.edu

ABSTRACT

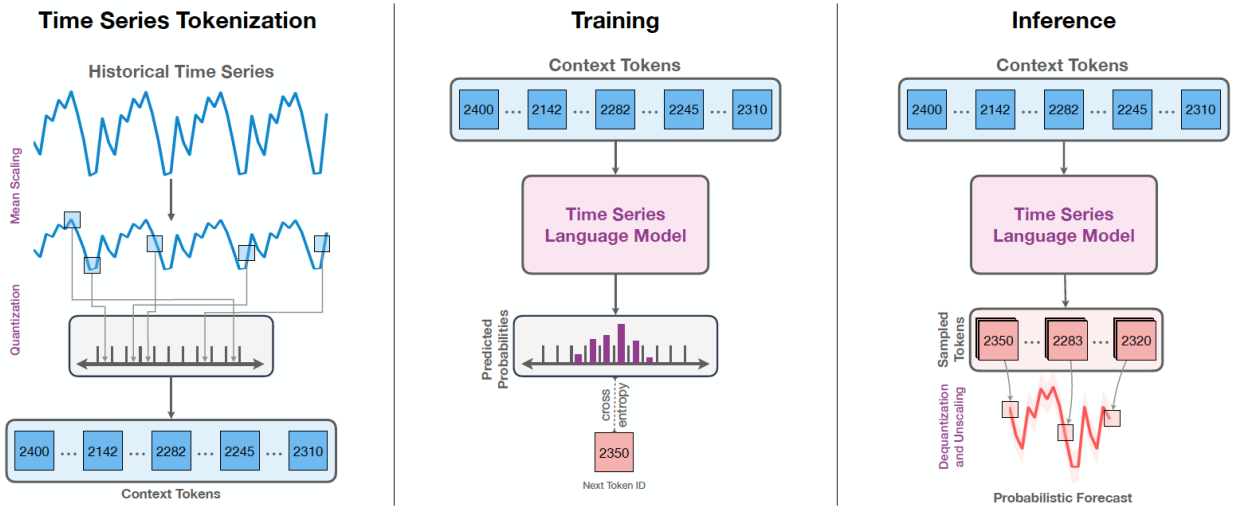


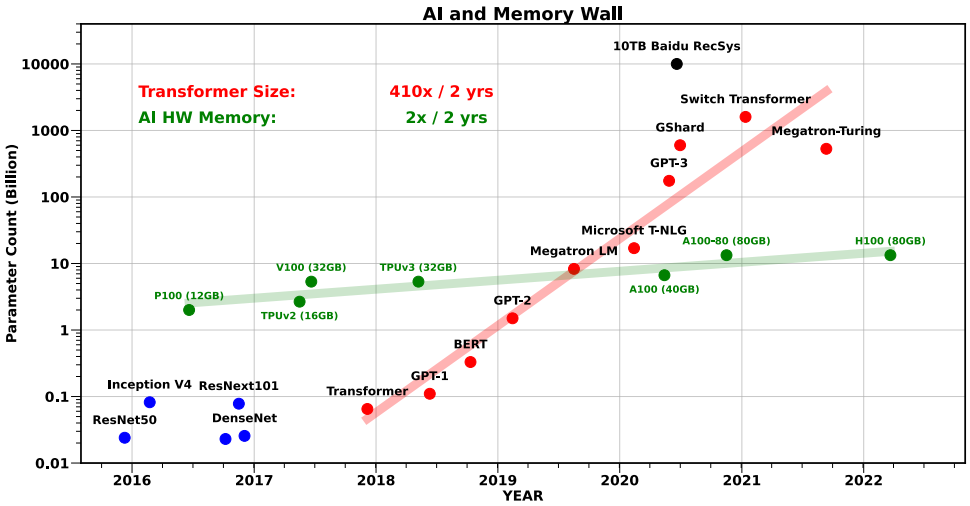
Figure 1: High-level depiction of CHRONOS. (**Left**) The input time series is scaled and quantized to obtain a sequence

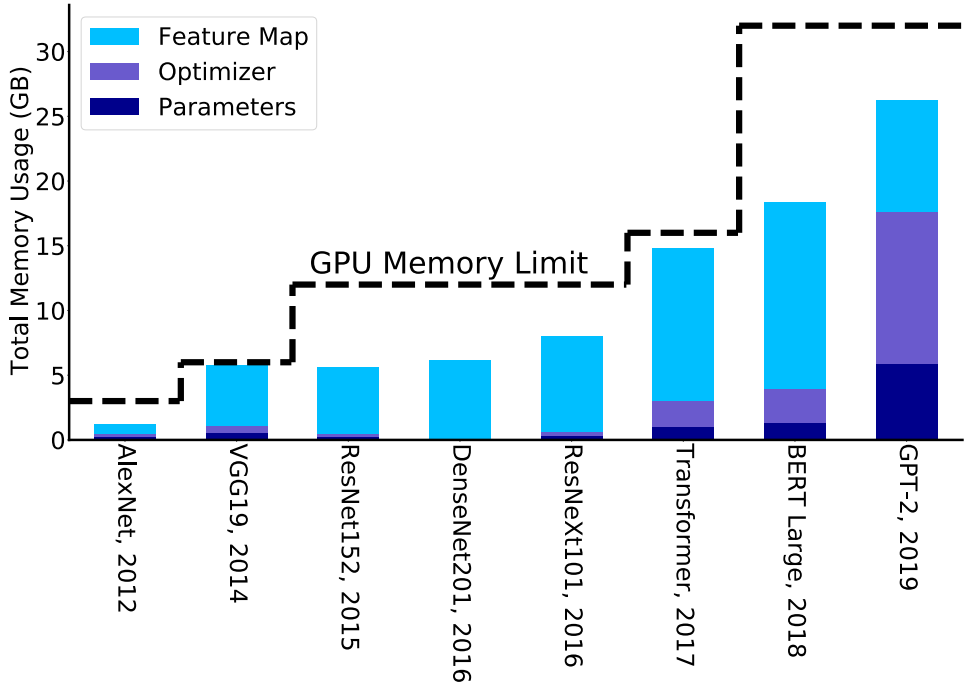
"Chronos: Learning the Language of Time Series," Ansari et al., arXiv:2403.07815

Questions?

- Q0: What is a “Foundation Model”?
- Q1: Can we hope to train a “Foundation Model” for SciML?
- Q2: Would incorporating physical knowledge help? If so, how to do it?
- Q3: Foundations?
- **Q4: Implementations?**
- Q6: Applications?
- Q6: Looking forward?

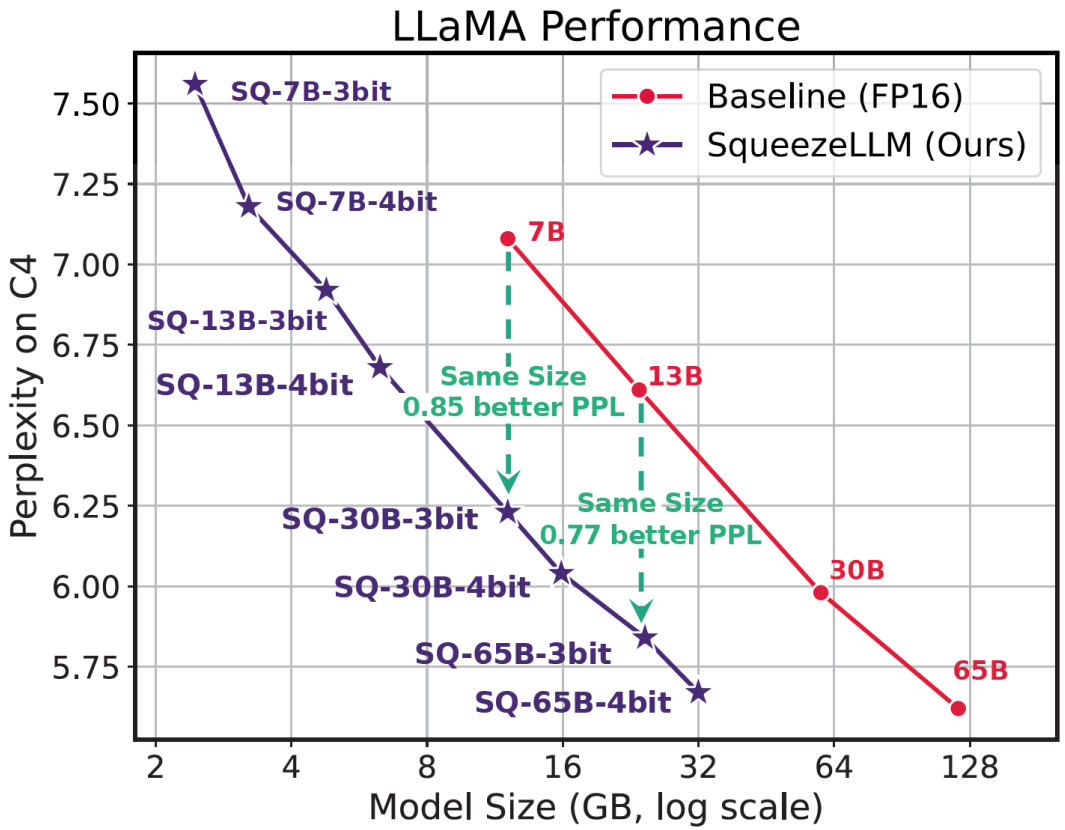
Model Size Increased Exponentially in 2018-22





Amir Gholami, Zhewei Yao, Sehoon Kim, Michael W. Mahoney, Kurt Keutzer, AI and Memory Wall, IEEE Micro, 2024.

SqueezeLLM Overview
Breaking Memory Wall with Dense-and-Sparse Quantization



Kim*, S., Hooper*, C., Gholami*, A., Dong, Z., Li, X., Shen, S., Mahoney, M.W. and Keutzer, K. SqueezeLLM: Dense-and-Sparse Quantization. *arXiv:2306.07629*.

Implementations: Quantization

Computer Science > Computer Vision and Pattern Recognition

[Submitted on 25 Mar 2021 (v1), last revised 21 Jun 2021 (this version, v3)]

A Survey of Quantization Methods for Efficient Neural Network Inference

Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W. Mahoney, Kurt Keutzer

As soon as abstract mathematical computations were adapted to computation on digital computers, the problem of efficient representation, manipulation, and communication of the numerical values in those computations arose. Strongly related to the problem of numerical representation is the problem of quantization: in what manner should a set of continuous real-valued numbers be distributed over a fixed discrete set of numbers to minimize the number of bits required and also to maximize the accuracy of the attendant computations? This perennial problem of quantization is particularly relevant whenever memory and/or computational resources are severely restricted, and it has come to the forefront in recent years due to the remarkable performance of Neural Network models in computer vision, natural language processing, and related areas. Moving from floating-point representations to low-precision fixed integer values represented in four bits or less holds the potential to reduce the memory footprint and latency by a factor of 16x; and, in fact, reductions of 4x to 8x are often realized in practice in these applications. Thus, it is not surprising that quantization has emerged recently as an important and very active sub-area of research in the efficient implementation of computations associated with Neural Networks. In this article, we survey approaches to the problem of quantizing the numerical values in deep Neural Network computations, covering the advantages/disadvantages of current methods. With this survey and its organization, we hope to have presented a useful snapshot of the current research in quantization for Neural Networks and to have given an intelligent organization to ease the evaluation of future research in this area.

Comments: Book Chapter: Low-Power Computer Vision: Improving the Efficiency of Artificial Intelligence
Subjects: **Computer Vision and Pattern Recognition (cs.CV)**
Cite as: arXiv:2103.13630 **[cs.CV]**
(or arXiv:2103.13630v3 **[cs.CV]** for this version)
https://doi.org/10.48550/arXiv.2103.13630

Implementations: HW-SW Co-design for Transformers

Full Stack Optimization of Transformer Inference: a Survey

Sehoon Kim*
sehoonkim@berkeley.edu
UC Berkeley

Coleman Hooper*
chooper@berkeley.edu
UC Berkeley

Thanakul Wattanawong
j.wat@berkeley.edu
UC Berkeley

Minwoo Kang
minwoo_kang@berkeley.edu
UC Berkeley

Ruohan Yan
yrh@berkeley.edu
UC Berkeley

Hasan Genc
hngenc@berkeley.edu
UC Berkeley

Grace Dinh
dinh@berkeley.edu
UC Berkeley

Qijing Huang
jennyhuang@nvidia.com
NVIDIA

Kurt Keutzer
keutzer@berkeley.edu
UC Berkeley

Michael W. Mahoney
mmahoney@stat.berkeley.edu
ICSI, LBNL, UC Berkeley

Yakun Sophia Shao
ysshao@berkeley.edu
UC Berkeley

Amir Gholami
amirgh@berkeley.edu
ICSI, UC Berkeley

ABSTRACT

Recent advances in state-of-the-art neural network architecture design have been moving toward Transformer models. These models achieve superior accuracy across a wide range of applications in computer vision, natural language processing, and speech recognition. This trend has been consistent over the past several years since Transformer models were originally introduced. However,

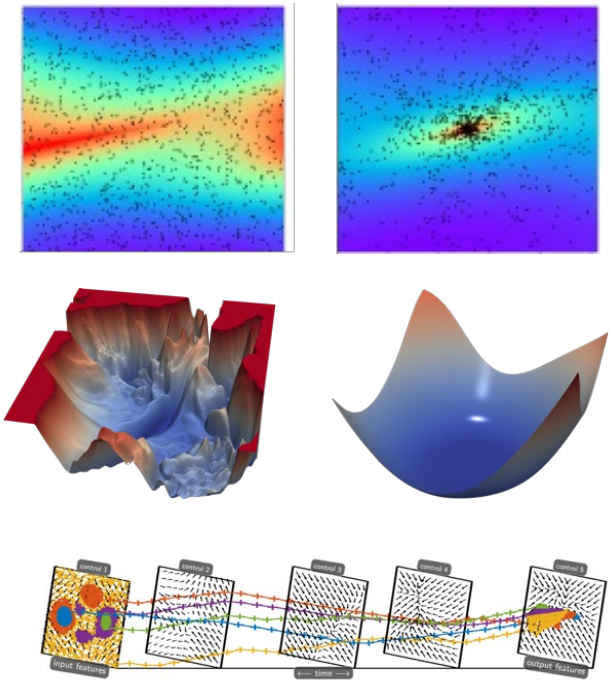
1 INTRODUCTION

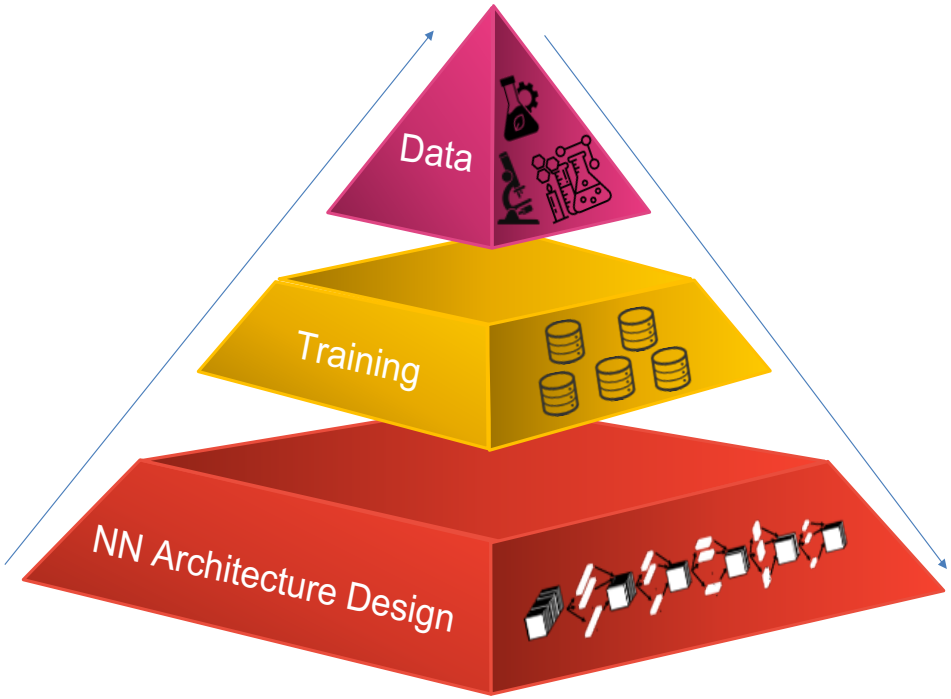
Deep learning models have scaled up to billions of parameters and billions of Multiply-Accumulate (MAC) operations during both training and inference. As a result, there has been a growing interest in computing these models efficiently and in deploying these compute and memory-intensive workloads on resource-constrained edge devices. These edge devices have tight energy and memory

Implementations: “Full stack” design

Rethink the *design*, *training, inference,* and *role of data* for successful application of NNs in SciML

- Different than computational design for ML/LLMs in industry
- Different than computational design in HPC and scientific simulation





Questions?

- Q0: What is a “Foundation Model”?
- Q1: Can we hope to train a “Foundation Model” for SciML?
- Q2: Would incorporating physical knowledge help? If so, how to do it?
- Q3: Foundations?
- Q4: Implementations?
- **Q6: Applications?**
- Q6: Looking forward?

Example scientific challenges

Popular Past Challenges:

- Learn solutions to PDEs
- Learn operators new laws of physics
- Learn dynamical systems

Lesson 1: Don't solve a past problem that some well-established domain solves.*
Lesson 2: Don't solve domain problems that are only well-define to domain expert.**

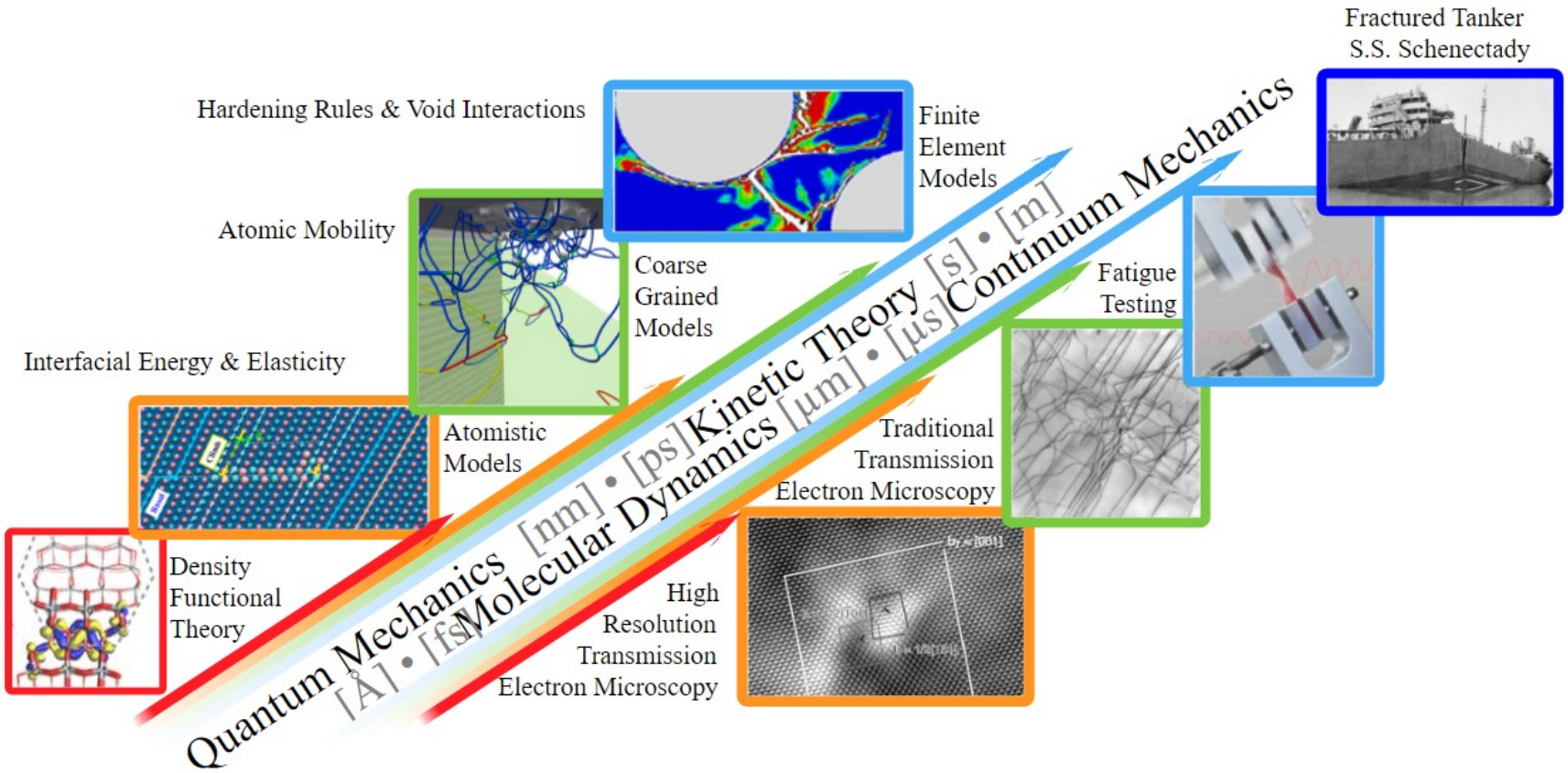
Important Future Challenges:

- **Extreme value forecasting/estimation**
- **Multi-scale modeling/analysis**
- **High-frequency inverse scattering**

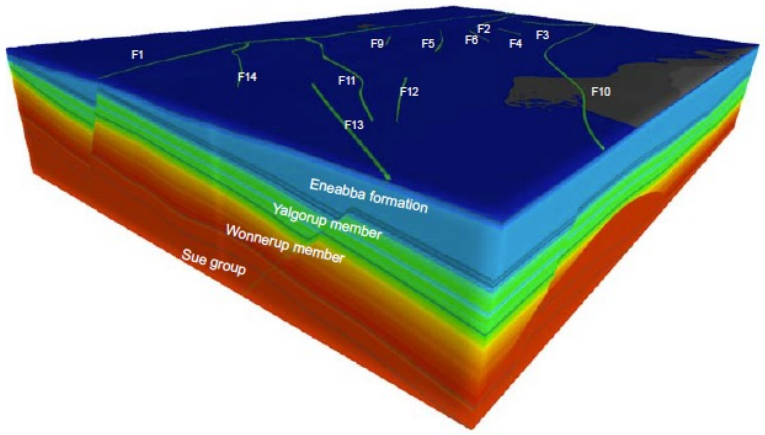
Goal: Focus on future challenges that are real scientific problems that cut across domains and that play well with ML methodologies.

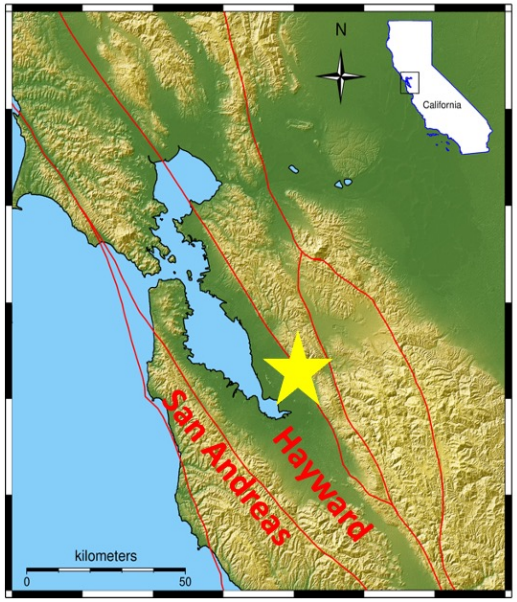
*They will beat you up, even if you do better than them.
**How ignorant can I be about your domain and still solve a problem you care about?

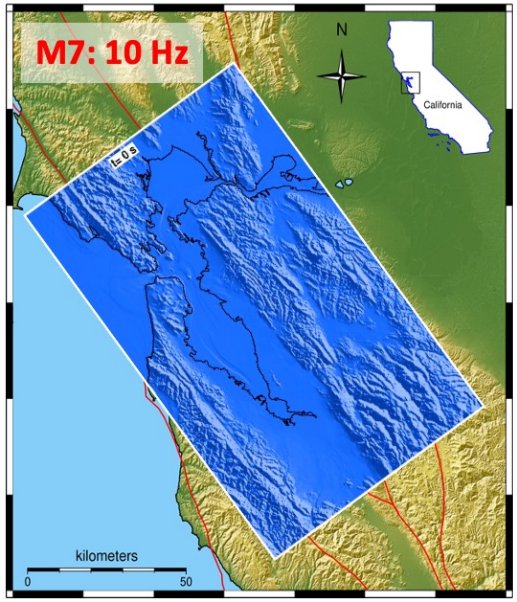
Example scientific challenges



Capturing wave propagation is a challenge



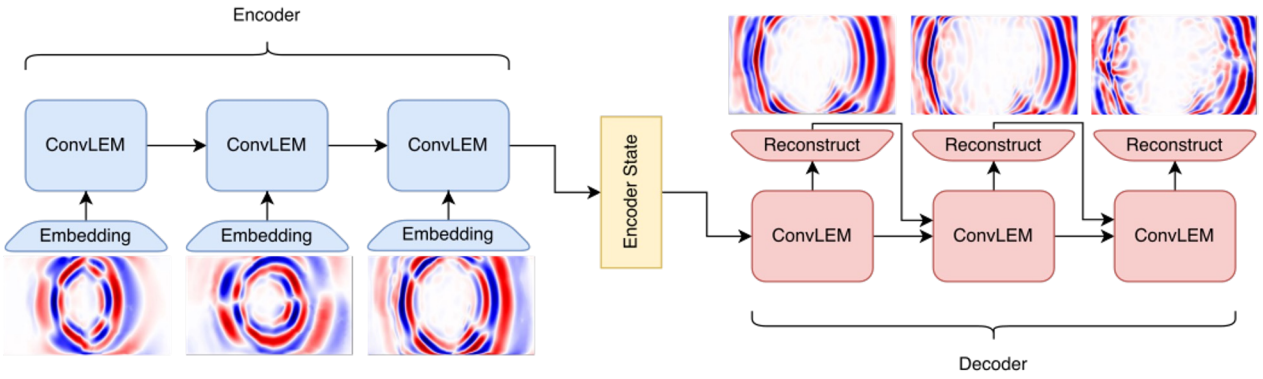




Many uses (in seismology and elsewhere):

- **Waveform forecasting for early warning**
- **Waveform simulation for HPC**
- **Waveform generation for imaging, ground motion prediction, source inversion**

Seq2Seq approach to forecast future waveforms

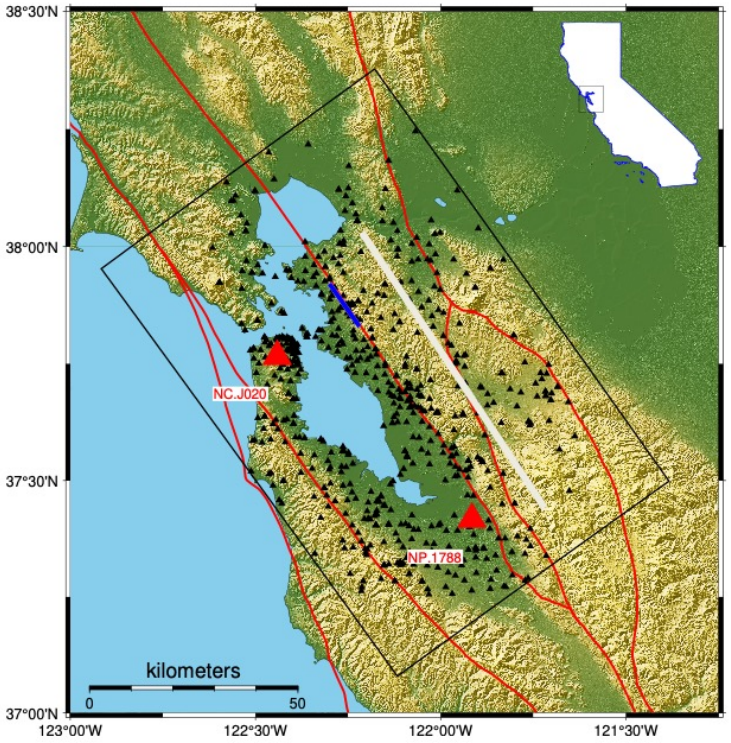


No need to know EQ magnitude, location, origin time, source parameters

- We use a sequence to sequence (***seq2seq***) approach to forecast seismic wavefields, consisting of encoder and decoder modules.
- The backbone of our ***seq2seq*** model is a novel convolutional long expressive memory model (***convLEM***), which allows us to jointly model multi-scale structure in space and time.
- The ConvLEM model is based on an input-driven systems of coupled ordinary differential equations, which can model multiple scales.

Work with Rie Nakata (LBNL), Ben Erichson (ICSI, LBNL), Dongwei Lyu (UCB), Arben Pitarka (LLNL)

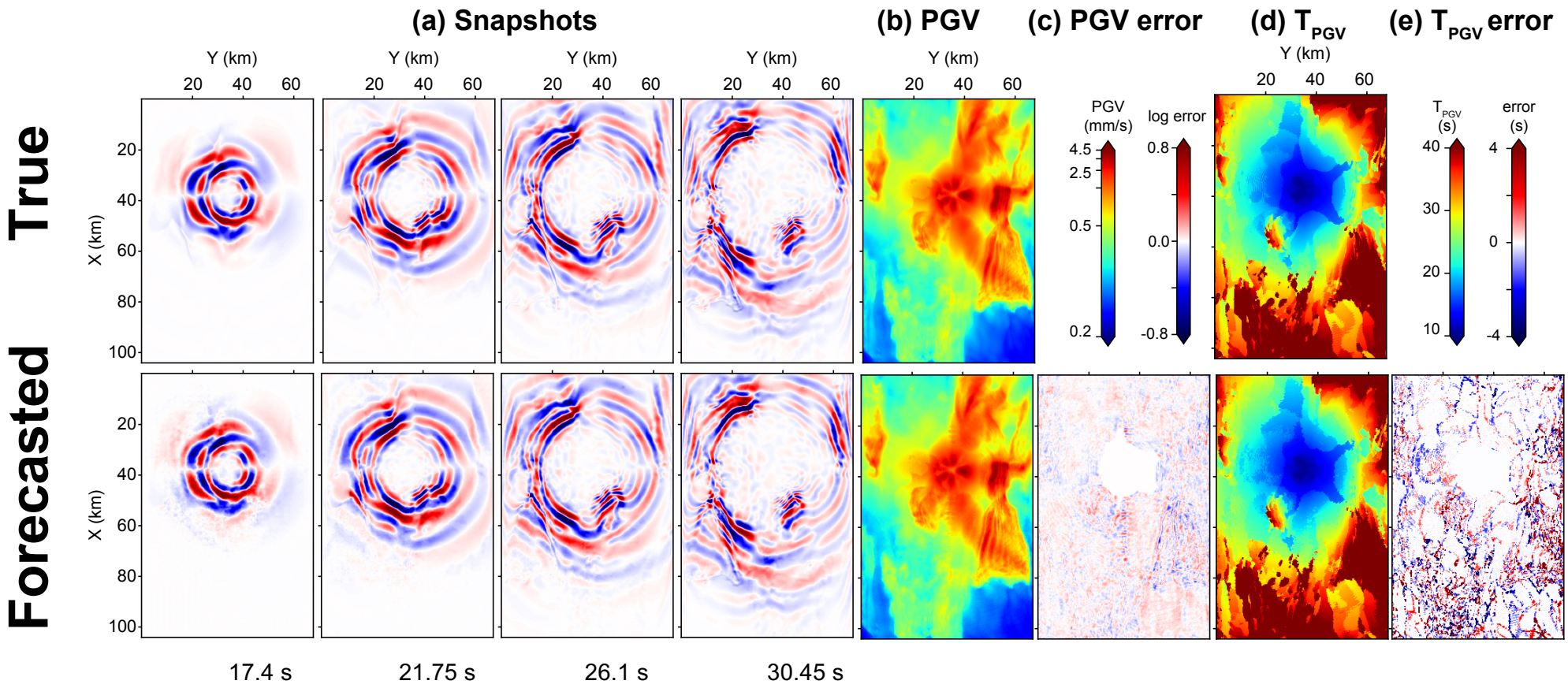
Waveform forecasting for synthetic point-source EQs



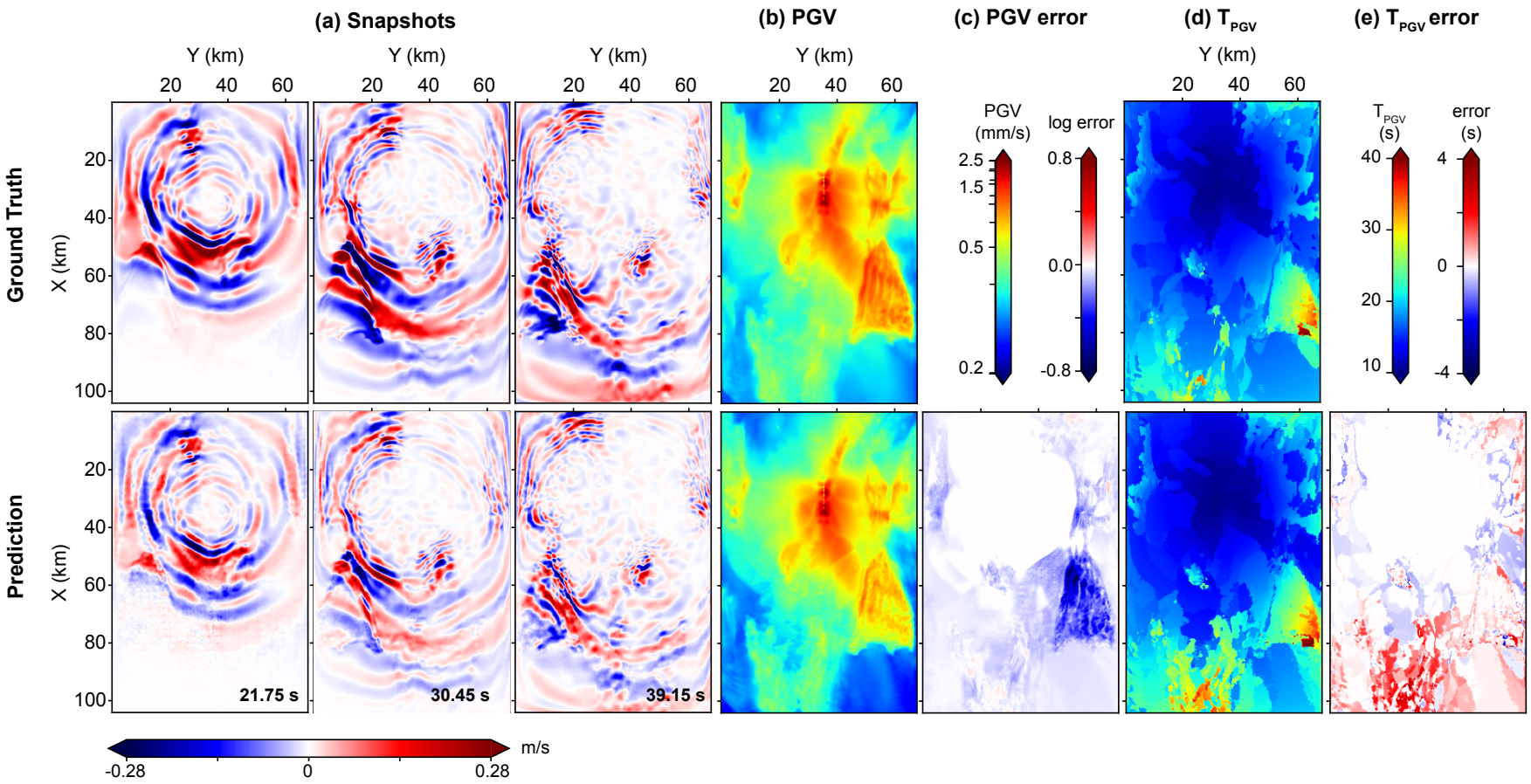
Use first 15.6 sec to predict 104.4 sec

- Strike-slip point-sources
- Source locations
  - Along fault: 1 km interval over 60 km
  - Depth: 2 – 15 km, 1 km interval
- A total of 960 simulations
- USGS Velocity models
- Finite-difference visco-elastic simulation (SW4)
- < 0.5 Hz
- Tested with dense regular grid and sparse irregular sensor locations

Forecasted waveforms, PGV, and arrival time



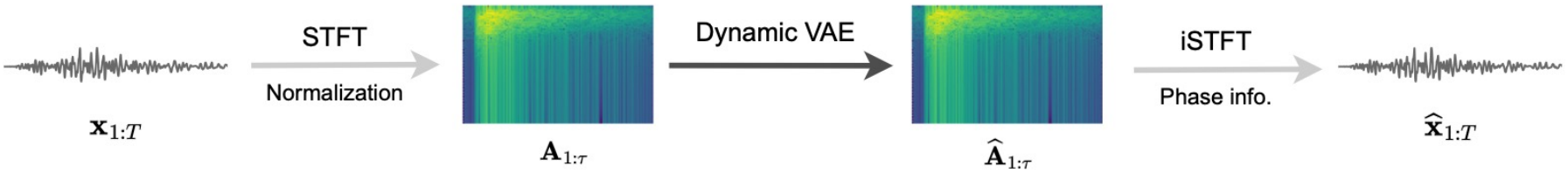
Generalization to unseen event: **M6**



Generative modeling for ground motion synthesis

Build a ***dynamic* Variational Autoencoder** to generate ground motions.

- Dynamic: incorporate temporal evolution by using RNN
- Use STFT: for capturing both time and frequency features.
- Conditional variables: coordinates of source and sensors, earthquake magnitudes.
- GAN models: difficult to optimize and suffer from mode collapse.

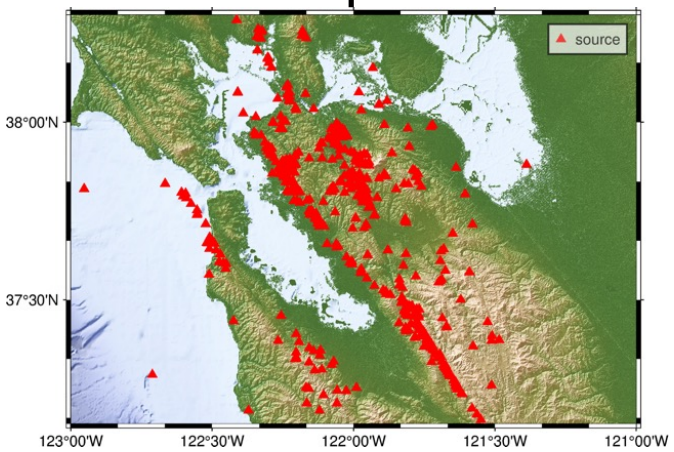


M, x_{src}, x_{rcv}

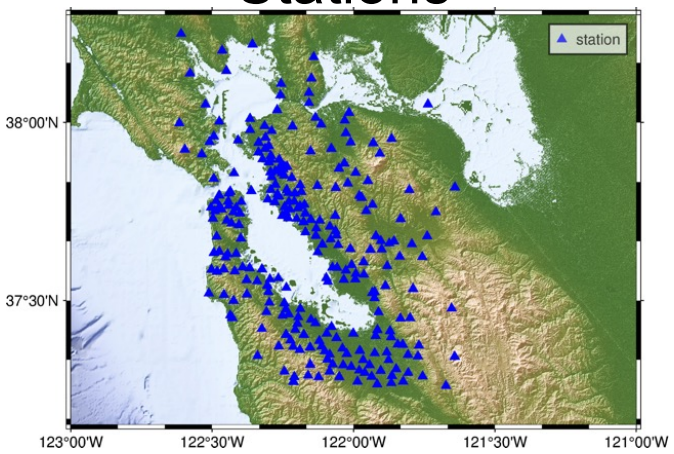
Work with P. Ren (LBNL), B. Erichson (LBNL, ICSI), R. Nakata (LBNL, MIT), N. Nakata (LBNL, MIT), M. Lacour, N. Abrahamson (UCB)

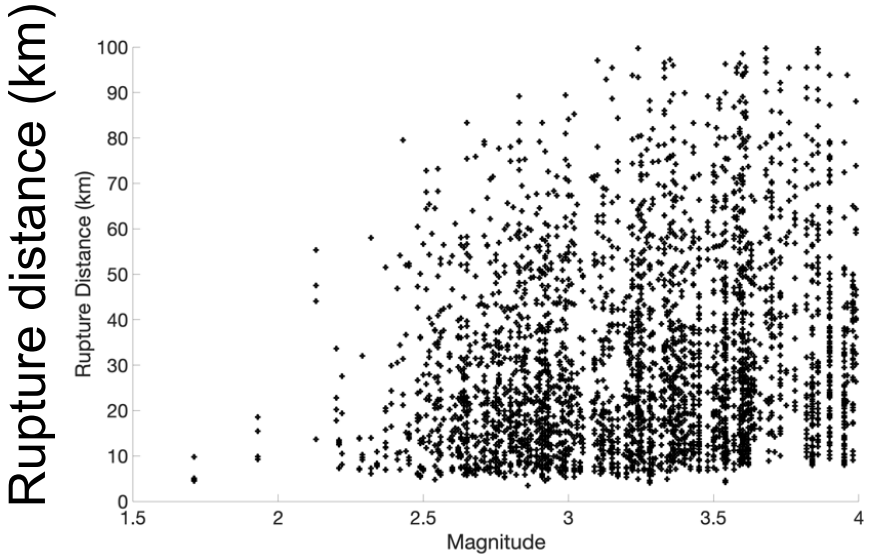
Training dataset for generative models

Earthquakes



Stations



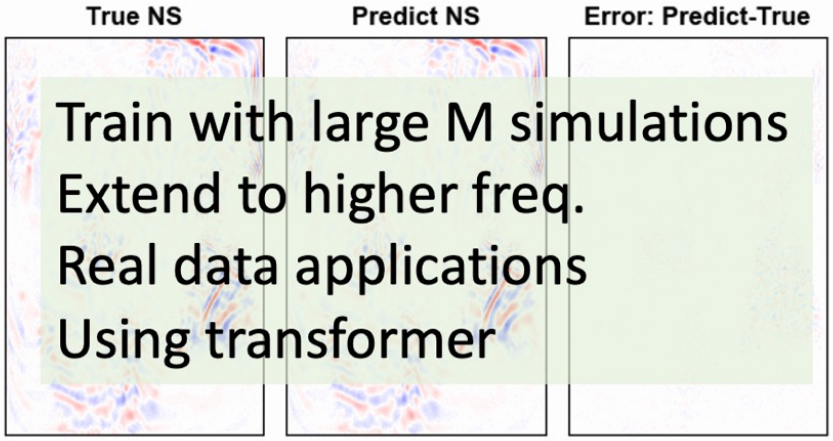


Magnitude

Use horizontal component recordings.
Downloaded data: 1,375,470
Training data: 5,194
Frequency range: 2-15 Hz

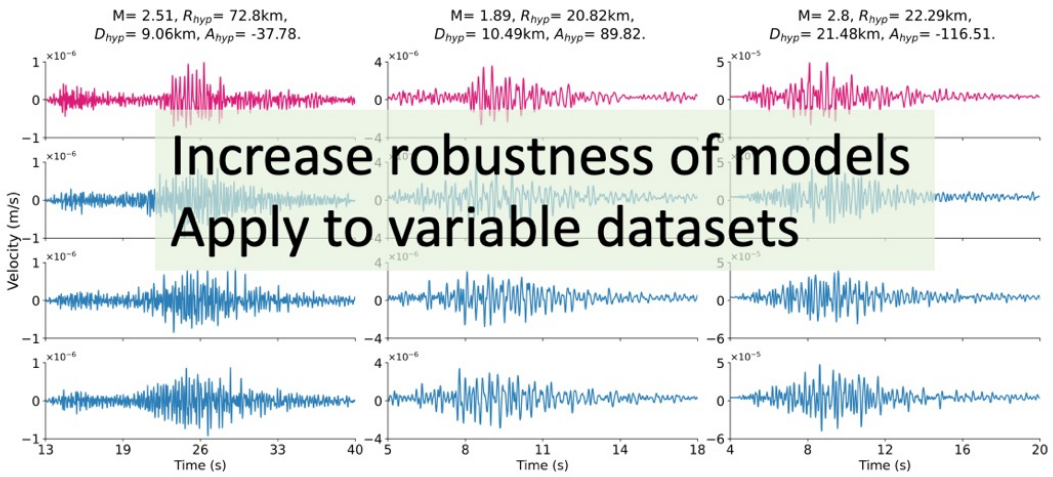
Summary

Waveform forecasting



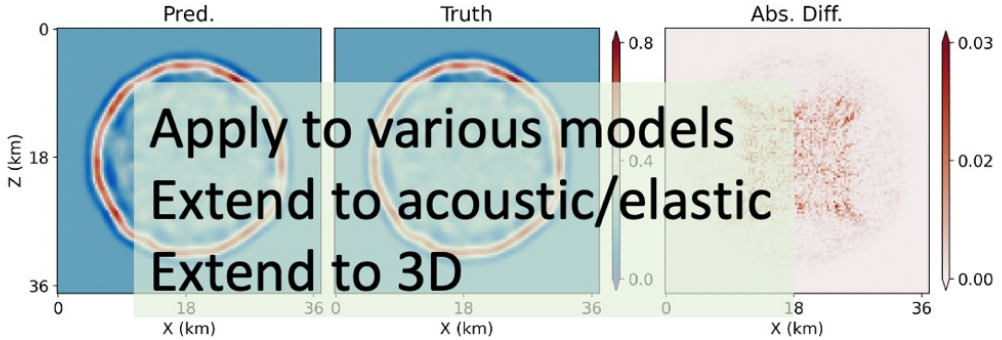
Train with large M simulations
Extend to higher freq.
Real data applications
Using transformer

Waveform generation



Increase robustness of models
Apply to variable datasets

Waveform simulation



Apply to various models
Extend to acoustic/elastic
Extend to 3D

Questions?

- Q0: What is a “Foundation Model”?
- Q1: Can we hope to train a “Foundation Model” for SciML?
- Q2: Would incorporating physical knowledge help? If so, how to do it?
- Q3: Foundations?
- Q4: Implementations?
- Q6: Applications?
- **Q6: Looking forward?**

Looking forward …

Foundation Models are infrastructure:

- A foundation upon which to do stuff
- Just like the computer, or iphone, or bridges, or electrical grid
- All these are impressive … until they are not

Look at history: computer science (industry) vs computational science (science)

- Very similar forcing functions
- Expect similar outcomes
- Do we compute on the metal or with multiple layers of abstraction?
- Do we fit SciML into the form factor provided by industrial LMs?

Question: How can we deliver on the promise of Scientific ML?

- Give it a strong, robust, principled foundations
- Rooted in both scientific principles and ML principles