Workshop on Scalable Physics-Informed Neural Networks

Session 1 - Introduction to SciML and PINNs

Dr. Ben Moseley

Scalable
Scientific Machine Learning
Lab

IMPERIAL

Poll

Journal of Computational Physics
Volume 378, 1 February 2019, Pages 686-707

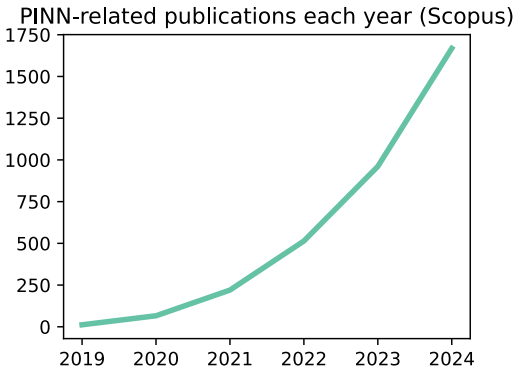
Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations

M. Raissi [a], P. Perdikaris [b], G.E. Karniadakis [a]

Show more

Add to Mendeley Share Cite

https://doi.org/10.1016/j.jcp.2018.10.045



TITLE-ABS-KEY ("physics-informed neural network" OR "physics informed neural network")

Highlights

- We put forth a deep learning framework that enables the synergistic combination of mathematical models and data.
- We introduce an effective mechanism for regularizing the training of
- The propos
discovery

ScienceDirect
https://www.sciencedirect.com › science › article › pii

Physics-informed neural networks: A deep learning ...

by M Raissi · 2019 · Cited by 17919 — We introduce physics-informed neura
networks that are trained to solve supervised learning tasks while respec

Abstract

Journal of Computational Physics
Volume 378, 1 February 2019, Pages 686-707

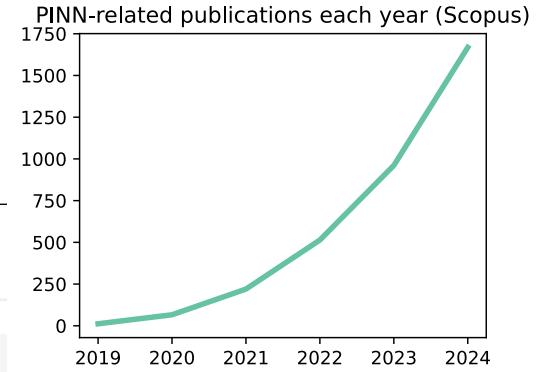
Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations

M. Raissi [a], P. Perdikaris [b], G.E. Karniadakis [a]

Show more

+ Add to Mendeley Share Cite

https://doi.org/10.1016/j.jcp.2018.10.045



TITLE-ABS-KEY ("physics-informed neural network" OR "physics informed neural network")

Highlights

- We put forth a deep learning framework that enables the synergistic combination of mathematical models and data.
- We introduce an effective mechanism for regularizing the training of
- The propos
discovery

ScienceDirect
https://www.sciencedirect.com › science › article › pii

Physics-informed neural networks: A deep learning ...
by M Raissi · 2019 · Cited by 17919 — We introduce physics-informed neura
networks that are trained to solve supervised learning tasks while respec

Abstract

Can physics-informed neural networks beat the finite element method?

Tamara G Grossmann, Urszula Julia Komorowska, Jonas Latz, Carola-Bibiane Schönlieb

IMA Journal of Applied Mathematics, Volume 89, Issue 1, January 2024, Pages 143–174,
https://doi.org/10.1093/imamat/hxae011

7. Discussion and conclusions

After having investigated each of the PDEs on its own, let us now discuss and draw conclusions from the results as a whole. Considering the solution time and accuracy, PINNs are not able to beat FEM in our study. In all the examples that we have studied, the FEM solutions were faster at the same or at a higher accuracy.

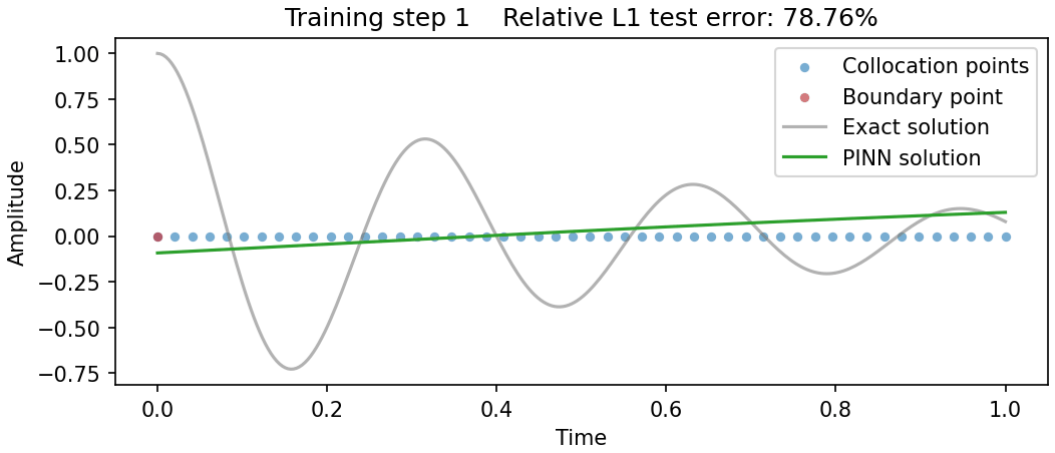
Solving inverse problems in physics by optimizing a discrete loss: Fast and accurate learning without neural networks

Petr Karnakov, Sergey Litvinov, Petros Koumoutsakos Author Notes

PNAS Nexus, Volume 3, Issue 1, January 2024, pgae005,
https://doi.org/10.1093/pnasnexus/pgae005

Conclusion

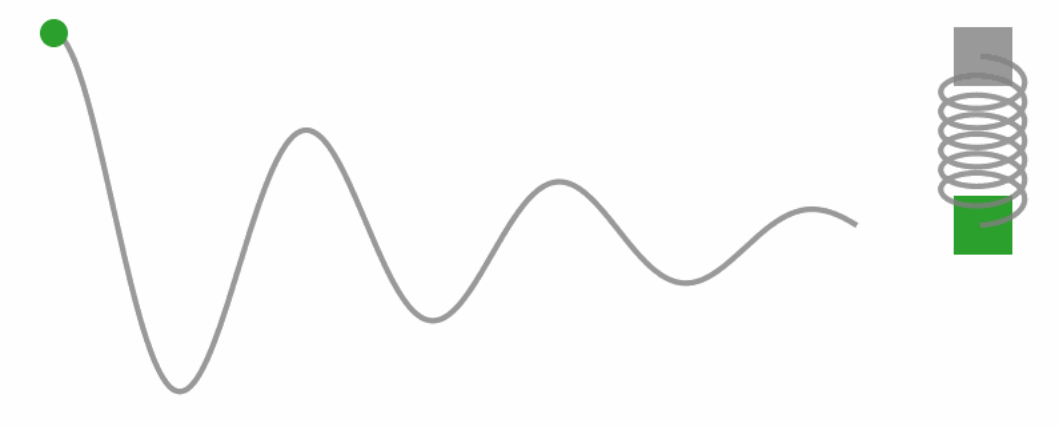
We introduce the ODIL framework for solving inverse problems for PDEs by casting their discretization as an optimization problem and applying optimization techniques that are widely available in machine-learning software. The concept of casting the PDE as is closely related to the neural network formulations proposed by (15–17) and recently revived as PINNs. However, the fact that we use the discrete approximation of the equations allows for ODIL to be orders of magnitude more efficient in terms of computational cost and accuracy compared to the PINN for which complex flow problems “remain elusive” (71).

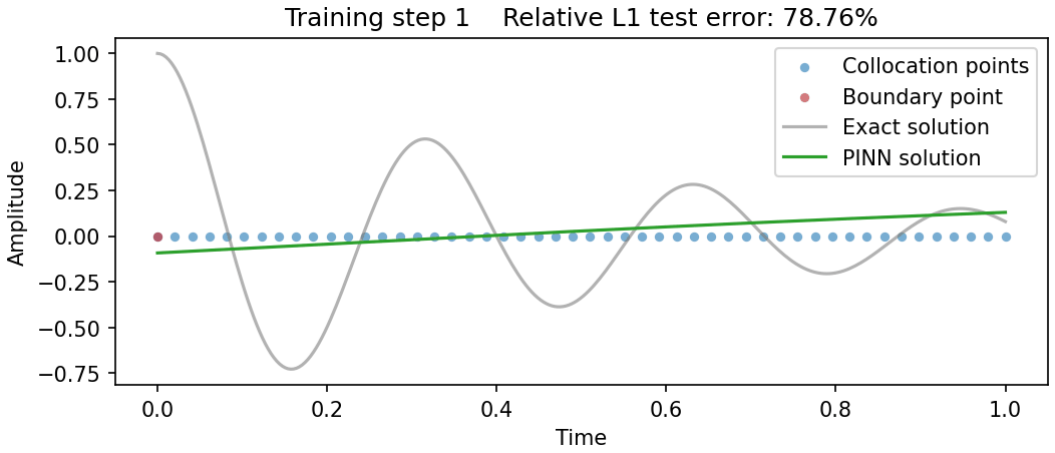


PINN (PyTorch) baseline

82 seconds

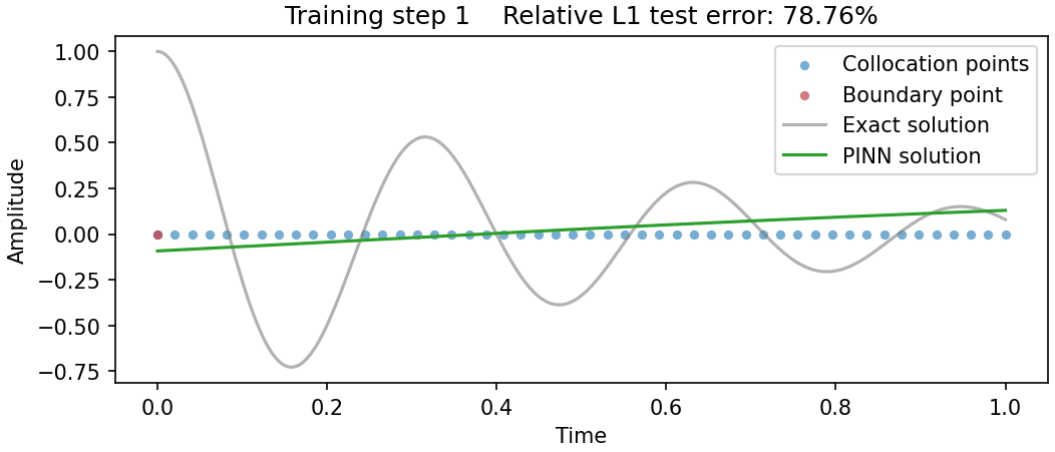
Problem: damped harmonic oscillator





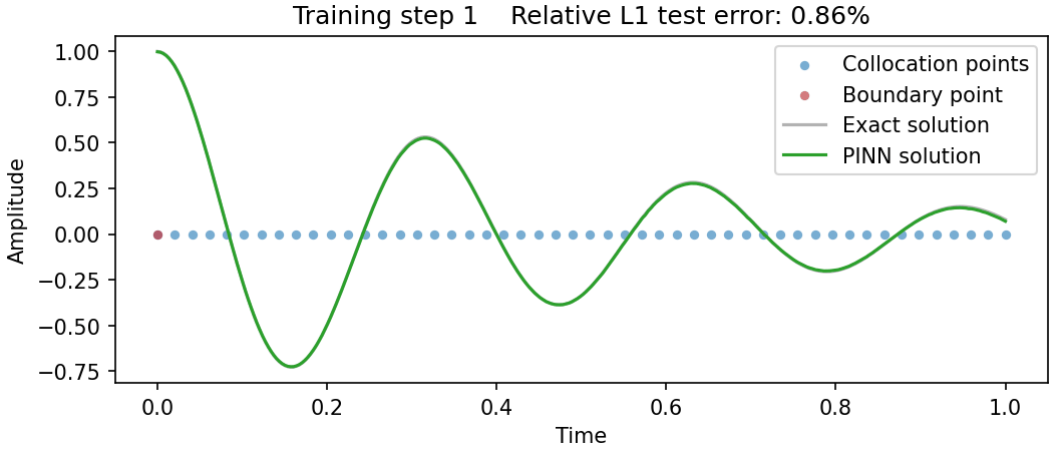
PINN (PyTorch) baseline

82 seconds



PINN **(JAX)**

11 seconds



PINN **(JAX) + Domain Decomposition + Numerical Linear Algebra**

0.002 seconds

Workshop overview

Session 1: **Intro to PINNs**

- *Lecture* (1 hr): Introduction to SciML and PINNs
- *Code-along* (30 min): Training a PINN in PyTorch

Session 2: **Accelerating PINNs with JAX**

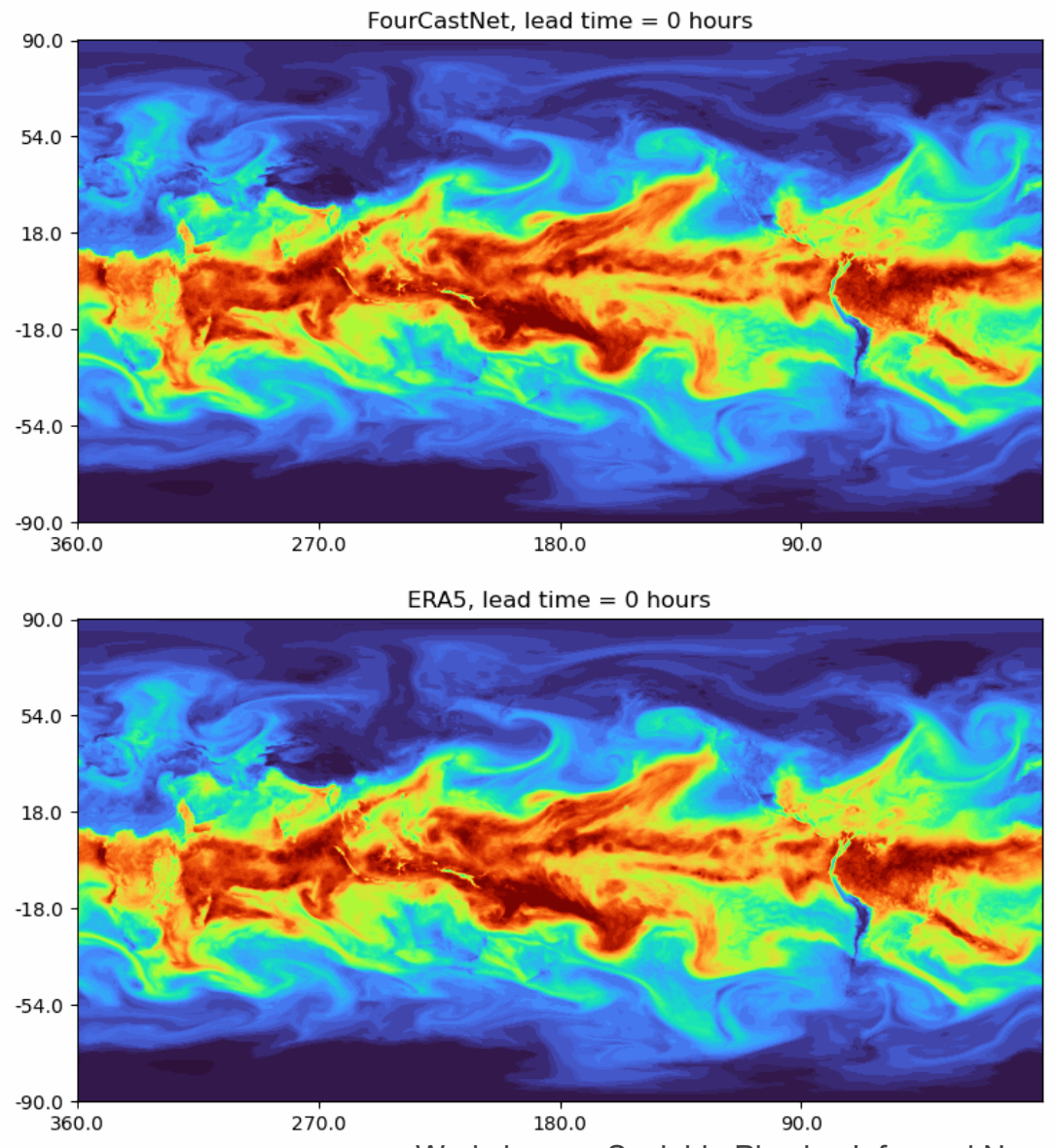
- *Lecture* (30 min): Introduction to JAX
- *Practical* (1 hr): Introduction to JAX and coding a PINN from scratch in JAX

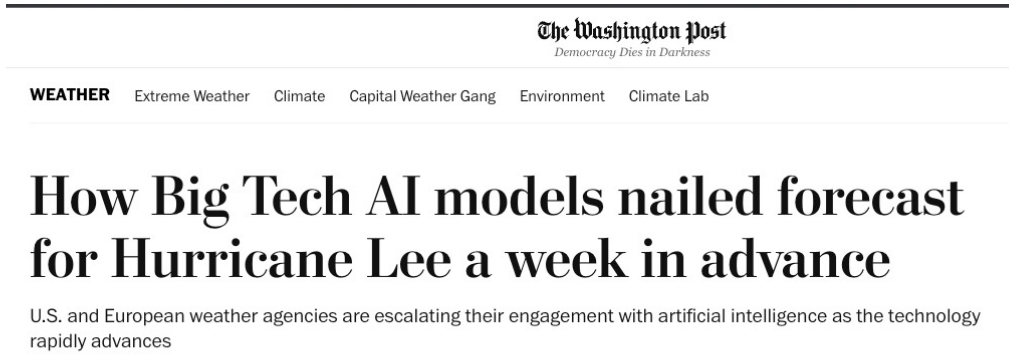
Session 3: **Accelerating PINNs with domain decomposition and NLA**

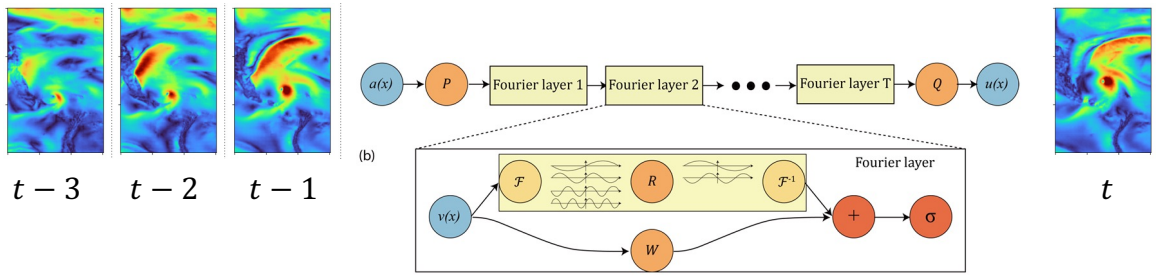
- *Lecture* (30 min): Challenges with PINNs and improving their performance with domain decomposition and numerical linear algebra
- *Practical* (1 hr): Coding finite basis PINNs and extreme learning machine FBPINNs in JAX

AI for Science

AI for climate science

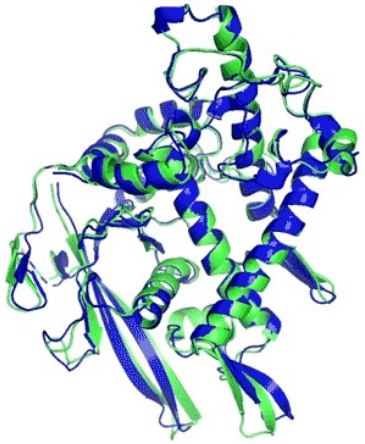




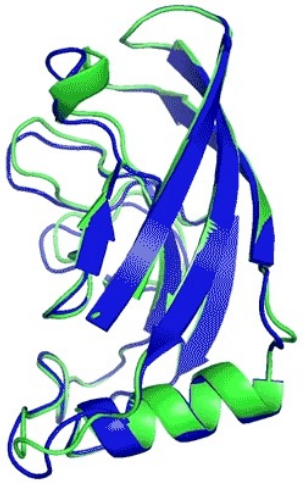


Pathak et al, FourCastNet: A Global Data-driven High-resolution Weather Model using Adaptive Fourier Neural Operators, ArXiv (2022)

AI for chemistry



T1037 / 6vr4
90.7 GDT
(RNA polymerase domain)



T1049 / 6y4f
93.3 GDT
(adhesin tip)

● Experimental result
● Computational prediction

nature

Explore content | About the journal | Publish with us

nature > articles > article

Article | Open Access | Published: 15 July 2021

Highly accurate protein structure prediction with AlphaFold

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, ... Demis Hassabis + Show authors

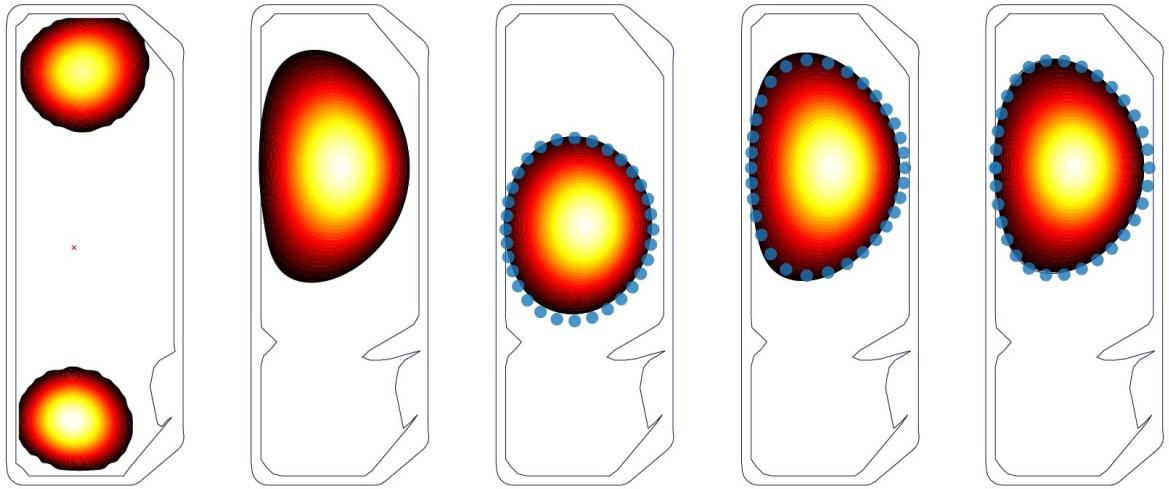
Nature **596**, 583–589 (2021) | Cite this article

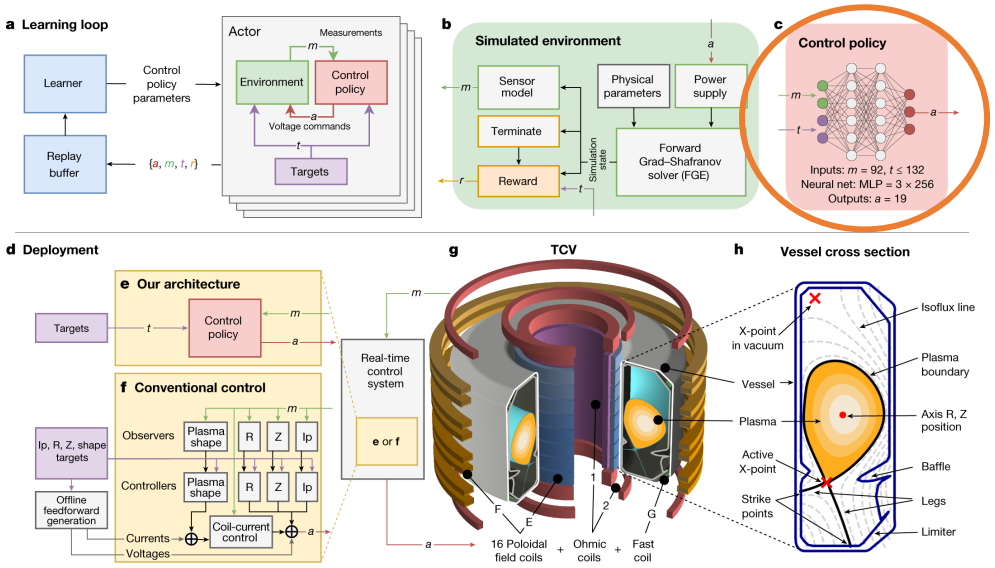
958k Accesses | **5511** Citations | **3408** Altmetric | Metrics

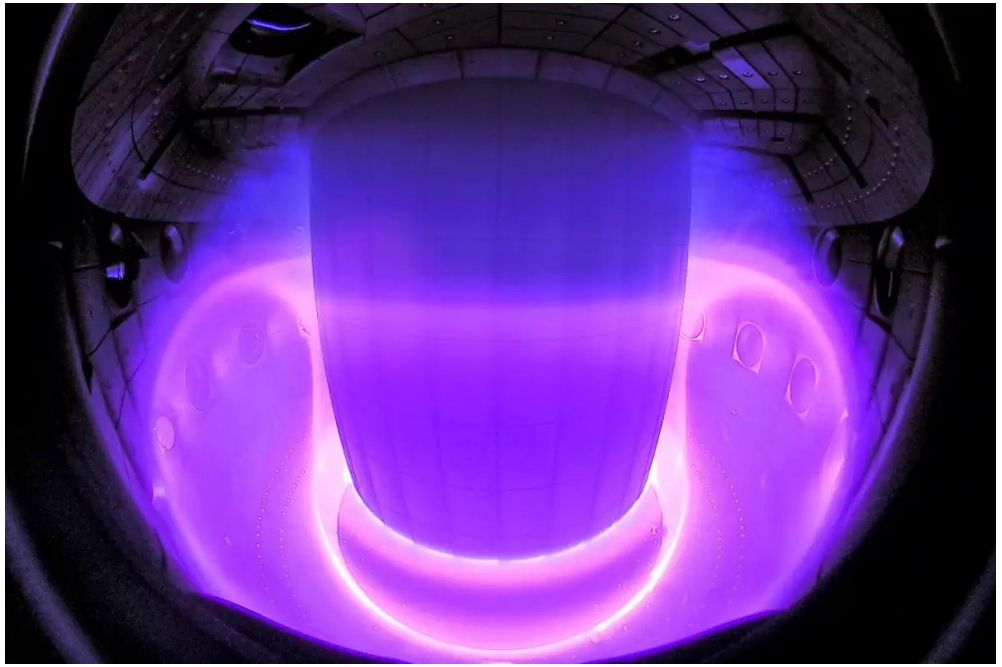
Abstract

Proteins are essential to life, and understanding their structure can facilitate a mechanistic understanding of their function. Through an enormous experimental effort[1,2,3,4], the structures of around 100,000 unique proteins have been determined[5], but this represents a small fraction of the billions of known protein sequences[6,7]. Structural coverage is bottlenecked by the months to years of painstaking effort required to determine a single protein structure. Accurate computational approaches are needed to address this gap and to enable large-scale structural bioinformatics. Predicting the three-dimensional structure that a protein will adopt based solely on its amino acid sequence—the structure prediction component of the 'protein folding problem'[8]—has been an important open research problem for more than 50 years[9]. Despite recent progress[10,11,12,13,14], existing methods fall far short of atomic accuracy, especially when no homologous structure is available. Here we provide the first computational method that can regularly predict protein structures with atomic accuracy even in cases in which no similar structure is known. We validated an entirely redesigned version of our neural network-based model, AlphaFold, in the challenging 14th

AI for fusion







Variable Configuration Tokamak (TCV) in Lausanne, Switzerland
Source: DeepMind & SPC/EPFL

nature

Explore content · About the journal · Publish with us

nature › articles › article

Article | Open access | Published: 16 February 2022

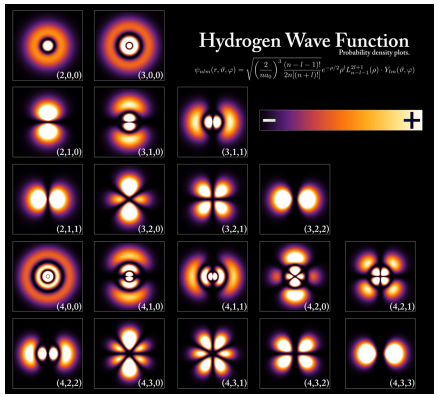
Magnetic control of tokamak plasmas through deep reinforcement learning

Jonas Degrave, Federico Felici, Jonas Buchli, Michael Neunert, Brendan Tracey, Francesco Carpanese, Timo Ewalds, Roland Hafner, Abbas Abdolmaleki, Diego de las Casas, Craig Donner, Leslie Fritz, Cristian Galperti, Andrea Huber, James Keeling, Maria Tsimpoukelli, Jackie Kay, Antoine Merle, Jean-Marc Moret, Seb Noury, Federico Pesamosca, David Pfau, Olivier Sauter, Cristian Sommariva, … Martin Riedmiller + Show authors

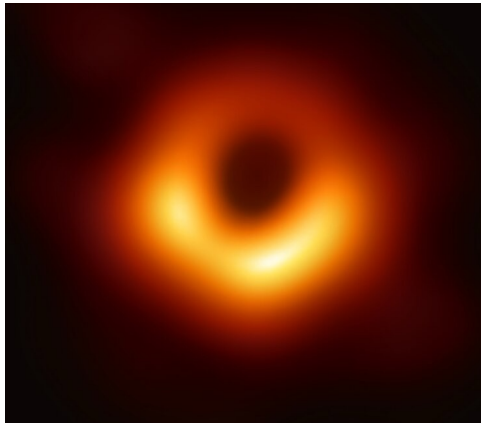
Nature **602**, 414–419 (2022) | Cite this article

206k Accesses | **223** Citations | **2430** Altmetric | Metrics

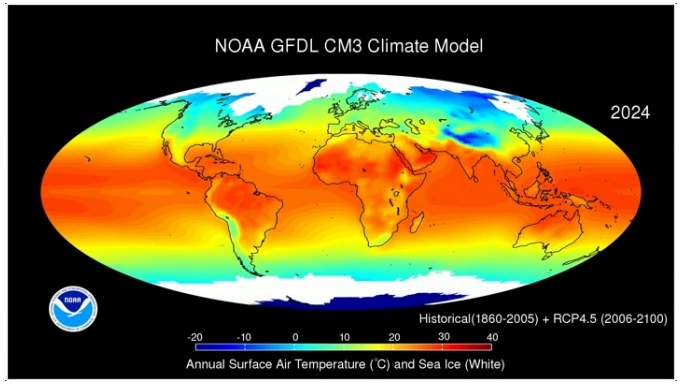
PDEs are the building blocks of science



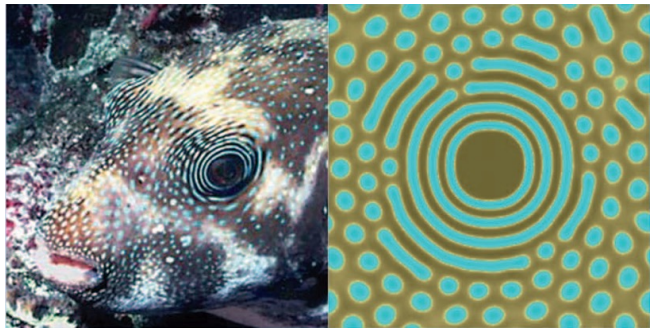
Source: Wikipedia



Source: The Event Horizon Telescope (2019)

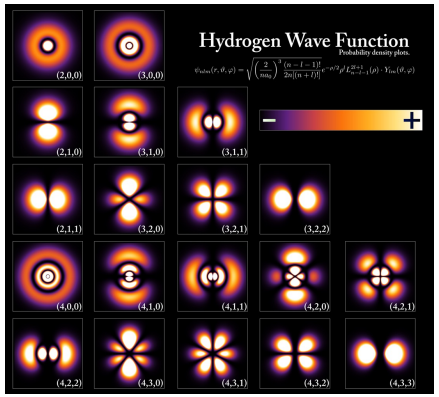


Source: NOAA



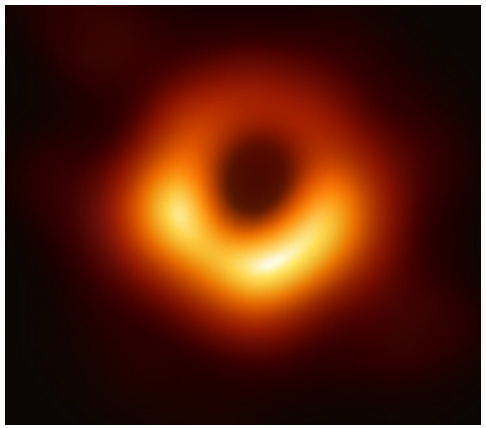
Source: Kondo and Miura, Science (2010)

PDEs are the building blocks of science



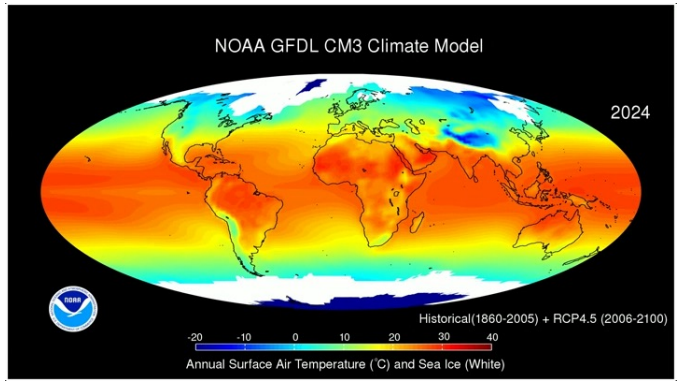
Source: Wikipedia

Schrödinger equation



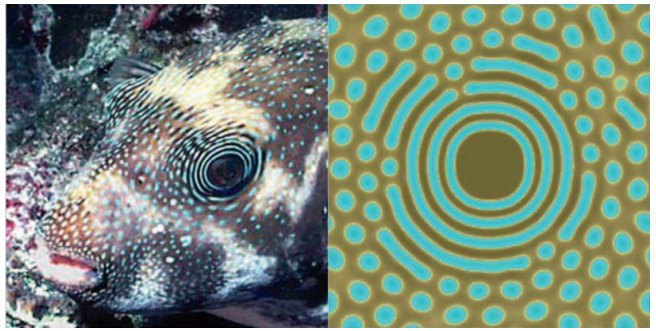
Source: The Event Horizon Telescope (2019)

Einstein field equations



Source: NOAA

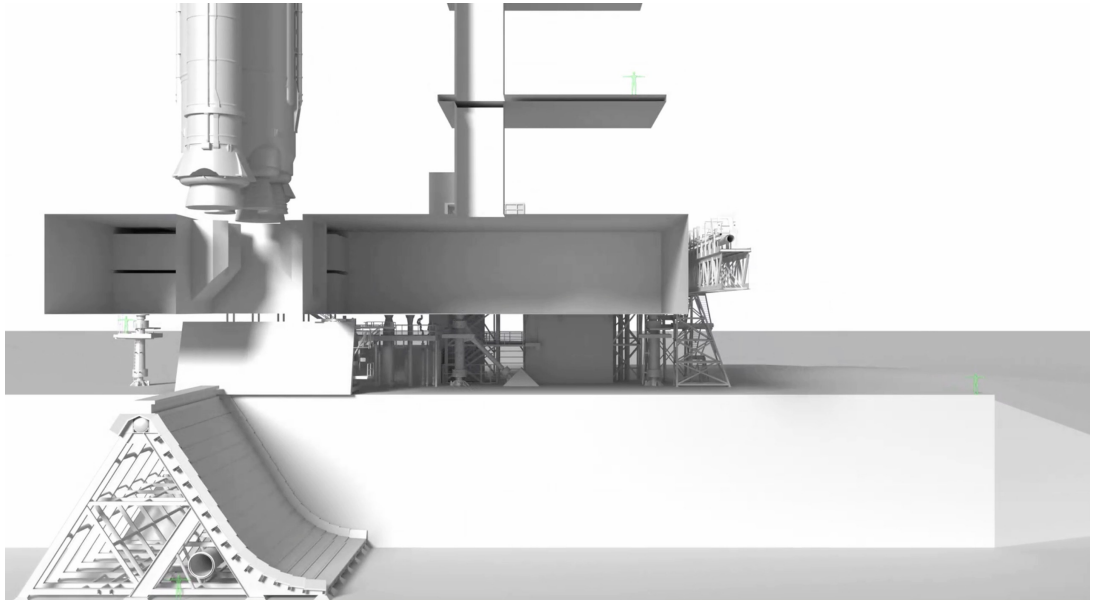
Navier-Stokes equations



Source: Kondo and Miura, Science (2010)

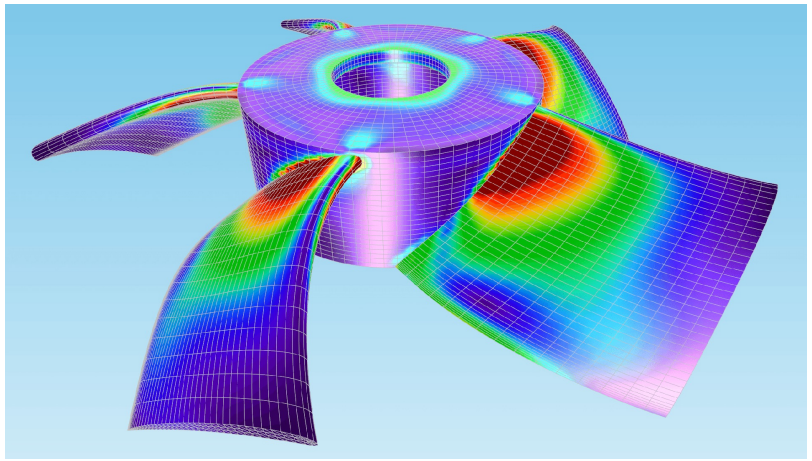
Reaction-diffusion equation

How can we solve PDEs?



Angel et al, Predicting SLS Launch Environment using a Novel Multiphase Formulation (NASA) SC22 (2022) Source: NASA

Required: 500 million grid cells, ran for several weeks on 8,000 cores, generating 400 TB (!)



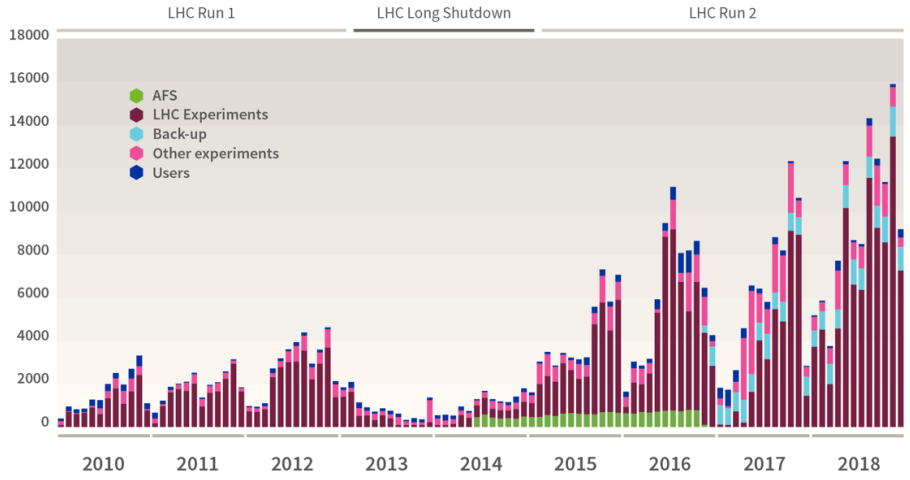
Mesh for finite element method
Source: COMSOL

Finite difference methods, finite element methods, finite volume methods, spectral methods, domain decomposition, mesh-free methods, …

Typically, **computational cost** is the main challenge

Grand challenges in science

Data (in terabytes) recorded on tapes at CERN month-by-month (2010–2018) (Source: CERN)



~5,000 exoplanets discovered to date

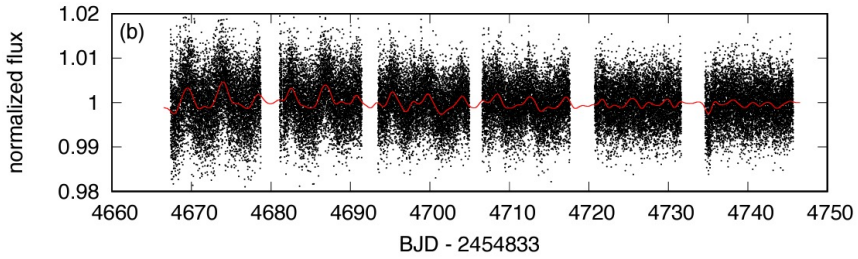
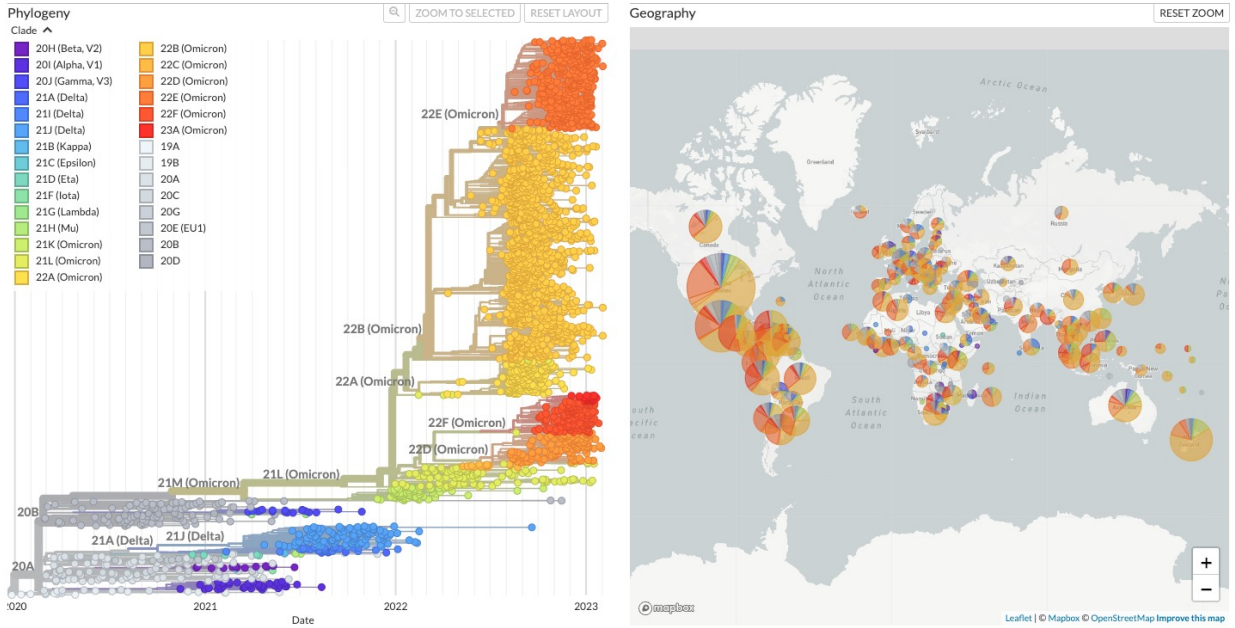


Figure 1. Light curves of K2-415 obtained by K2 (top; K2SFF) and TESS (bottom; PDC-SAP). Those data were taken at long (≈ 29 minutes) and short (2 minutes) cadences for K2 and TESS light curves, respectively. The red solid line in each panel represents the GP regression to the observed light curve (see Section 4.4).

Hirano et al, An Earth-sized Planet around an M5 Dwarf Star at 22 pc, The Astronomical Journal (2023)

Genomic epidemiology of SARS-CoV-2 with subsampling focused globally over the past 6 months

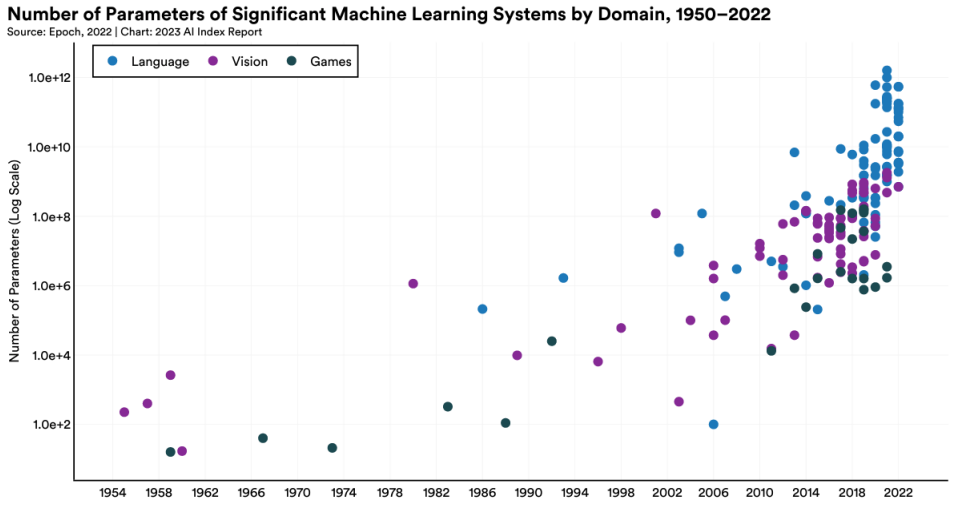


Source: Nextstrain

Challenges of deep learning

OpenAI

ChatGPT: Optimizing Language Models for Dialogue



2.1 Model and Architectures

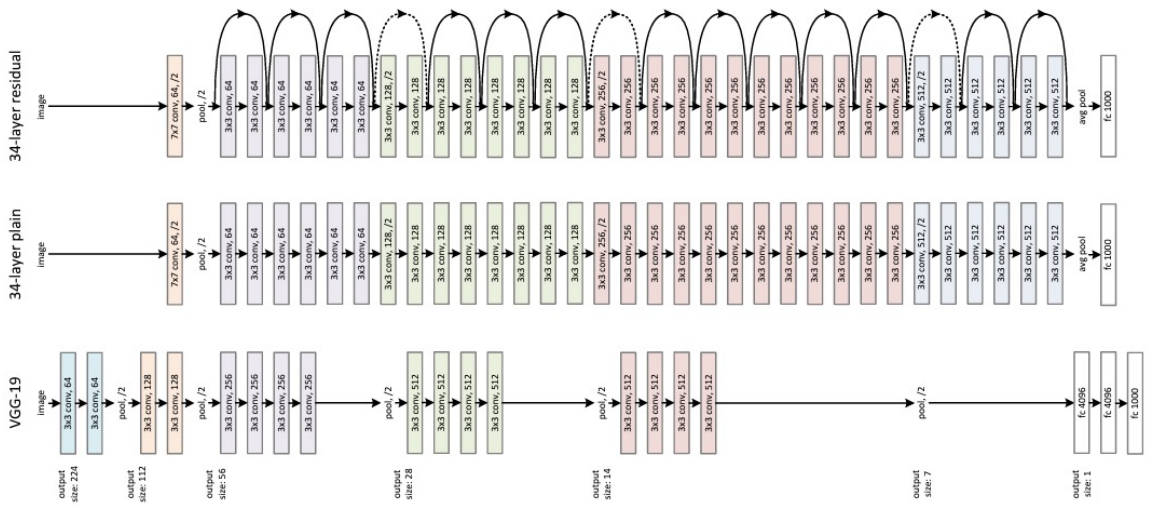
We use the same model and architecture as GPT-2 [RWC+19], including the modified initialization, pre-normalization, and reversible tokenization described therein, with the exception that we use alternating dense and locally banded sparse attention patterns in the layers of the transformer, similar to the Sparse Transformer [CGRS19]. To study the dependence of ML performance on model size, we train 8 different sizes of model, ranging over three orders of magnitude from 125 million parameters to 175 billion parameters, with the last being the model we call GPT-3. Previous work [KMH+20]

2.2 Training Dataset

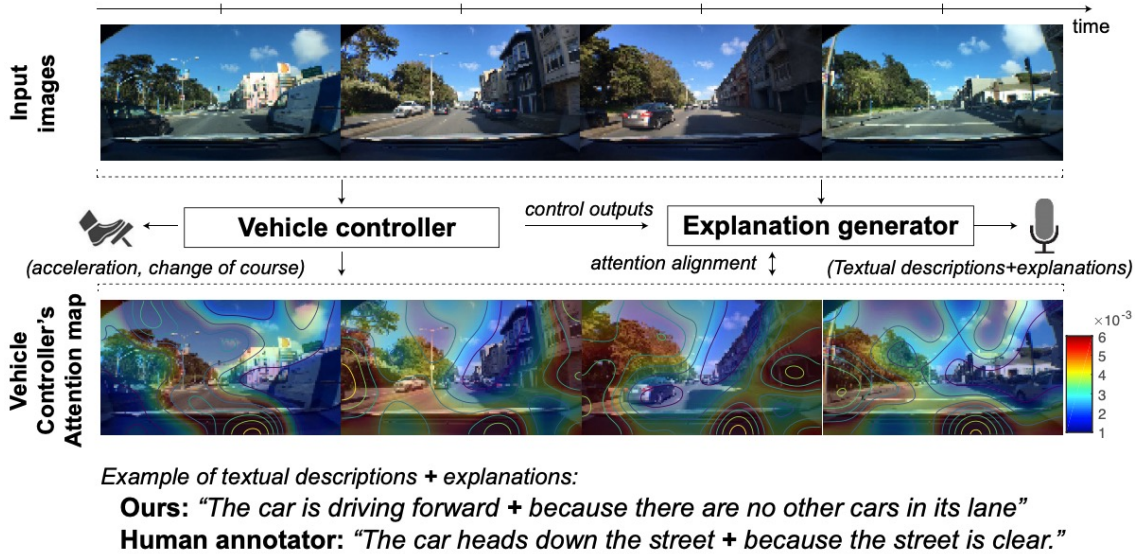
Table 2.2 shows the final mixture of datasets that we used in training. The CommonCrawl data was downloaded from 41 shards of monthly CommonCrawl covering 2016 to 2019, constituting 45TB of compressed plaintext before filtering and 570GB after filtering, roughly equivalent to 400 billion byte-pair-encoded tokens. Note that during training, datasets

Brown et al, Language Models are Few-Shot Learners, NeurIPS (2020)

Challenges of deep learning



He et al, Deep Residual Learning for Image Recognition, CVPR (2015)



Example of textual descriptions + explanations:

Ours: *"The car is driving forward + because there are no other cars in its lane"*

Human annotator: *"The car heads down the street + because the street is clear."*

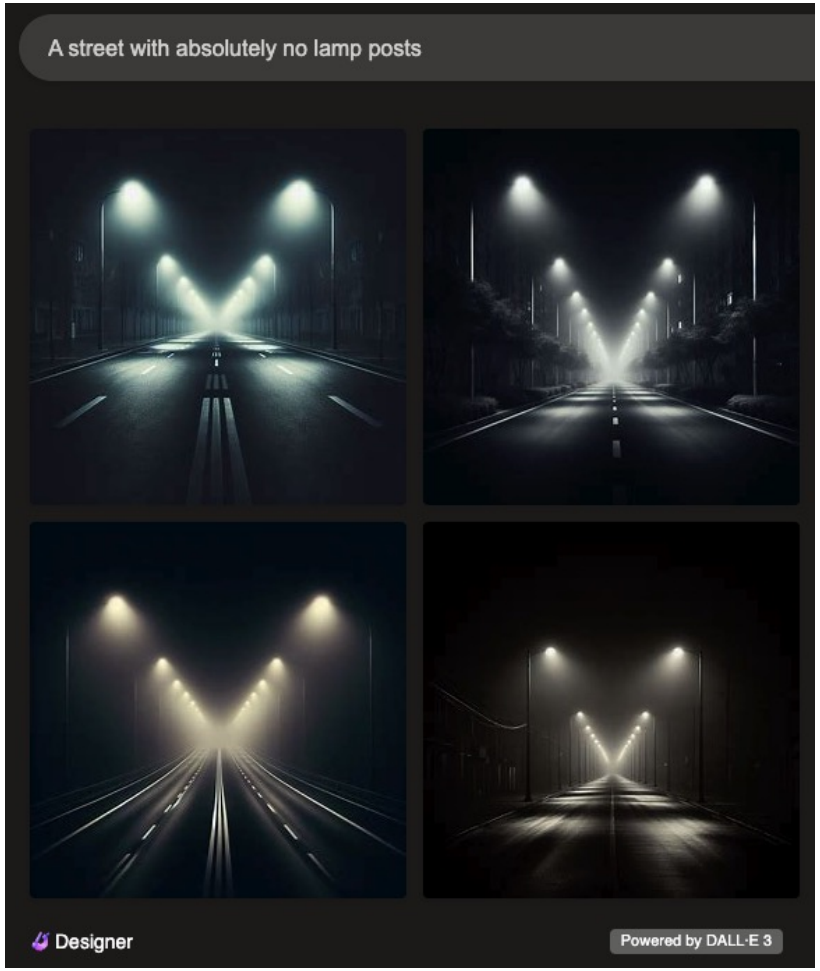
Kim et al, Textual Explanations for Self-Driving Vehicles, ECCV (2018)

The challenge of generalisation

Labrador retriever 52%
Chesapeake Bay retriever 7%
golden retriever 5%
Canis dingo 4%
bloodhound, sleuthhound 3%

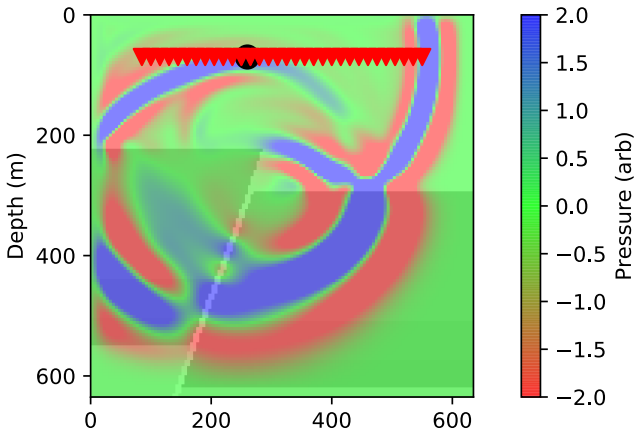
laboratory coat 40%
jeweler's loupe 8%
English foxhound 6%
soccer ball 4%
neck brace 3%

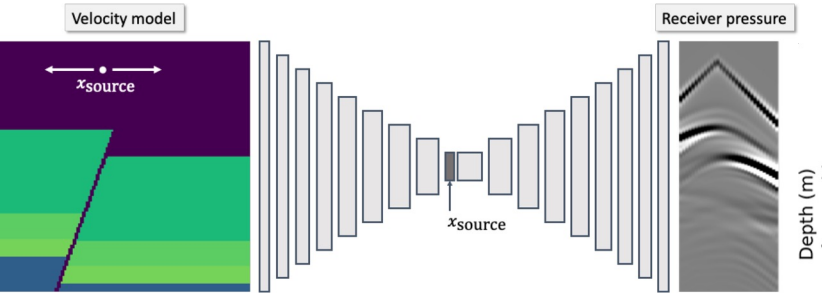
The challenge of reasoning

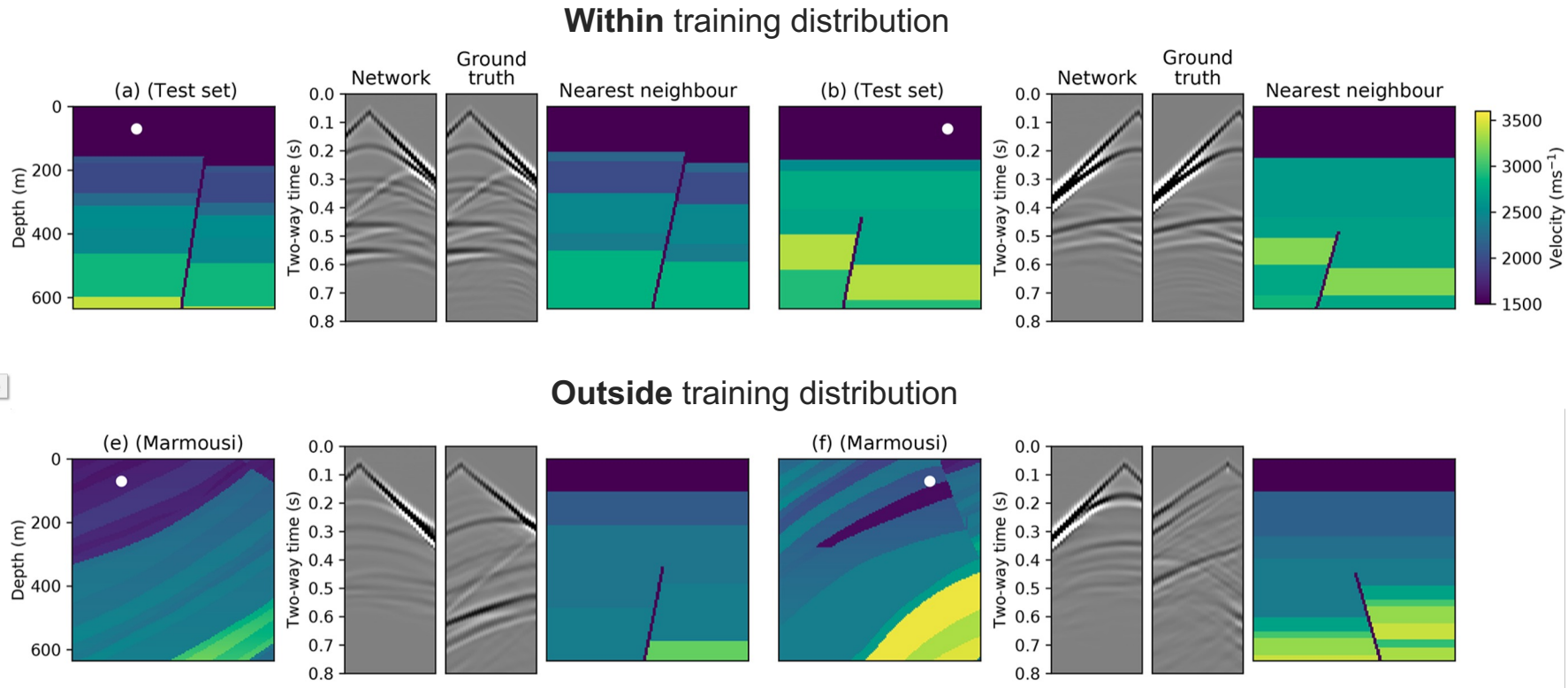


Source: DALL·E 3

Naïve application of deep learning



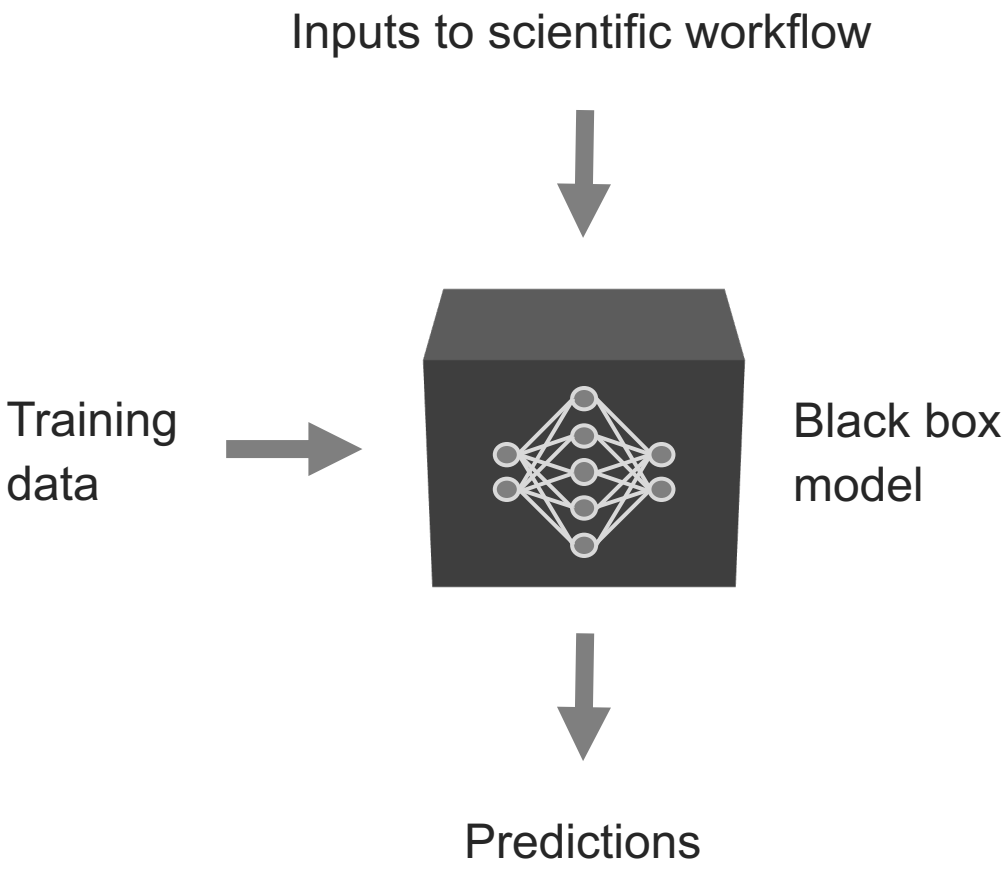




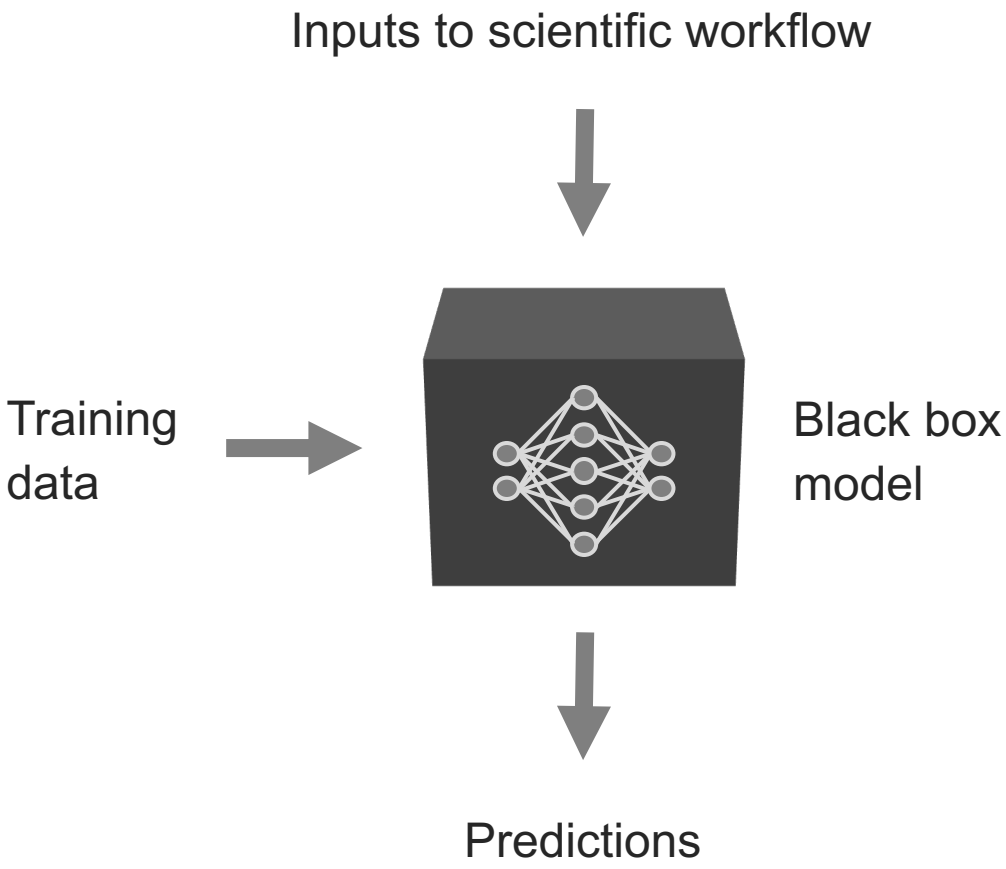
Moseley, B., Nissen-Meyer, T., & Markham, A. (2020). Deep learning for fast simulation of seismic waves in complex media. Solid Earth

What is Scientific Machine Learning (SciML)?

What is scientific machine learning?



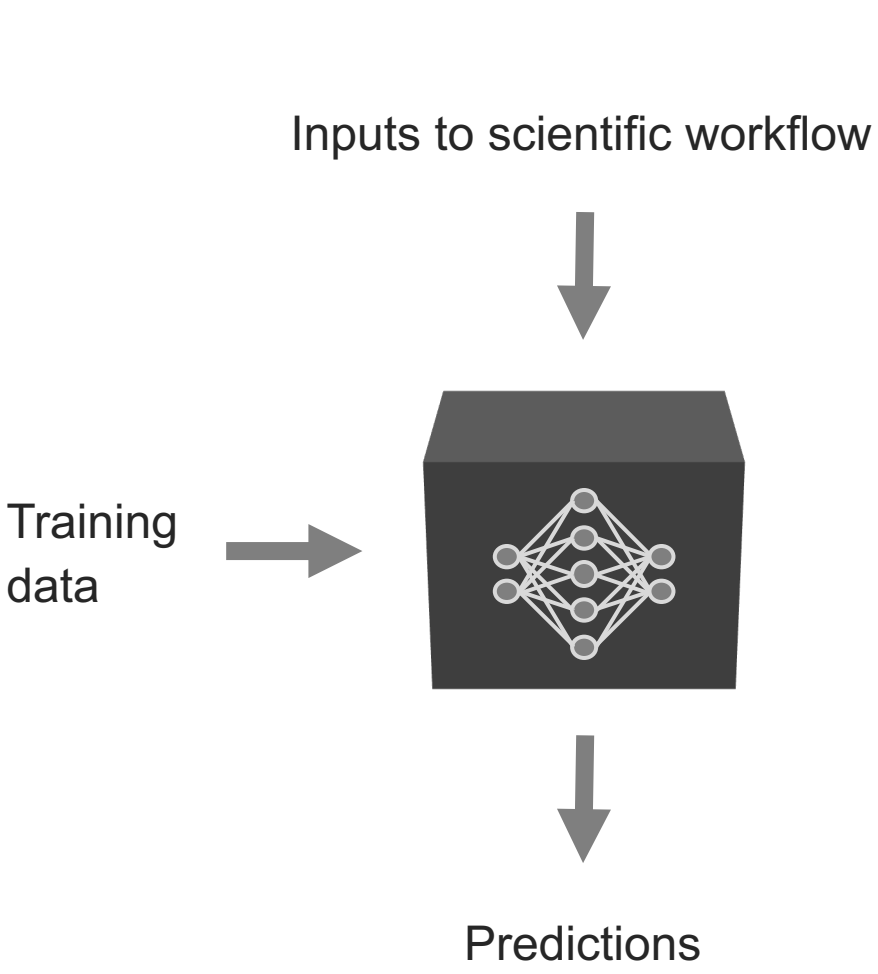
What is scientific machine learning?

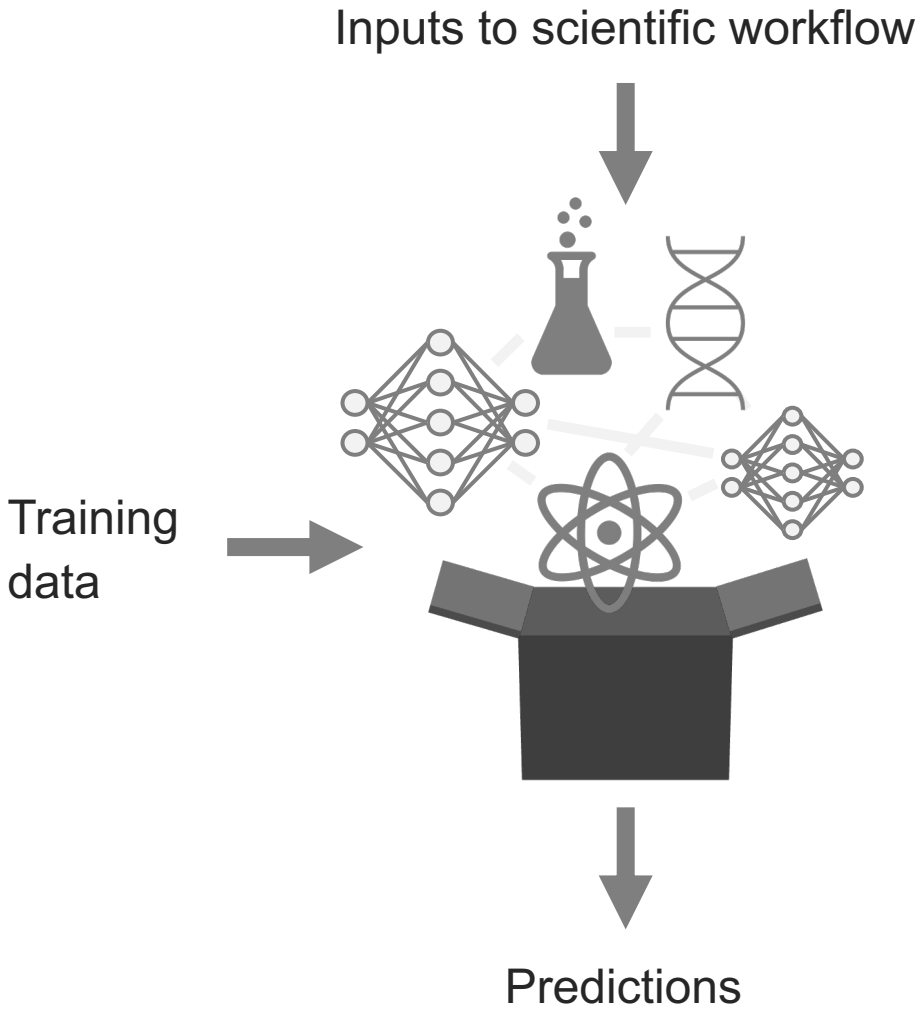


Suffers from:

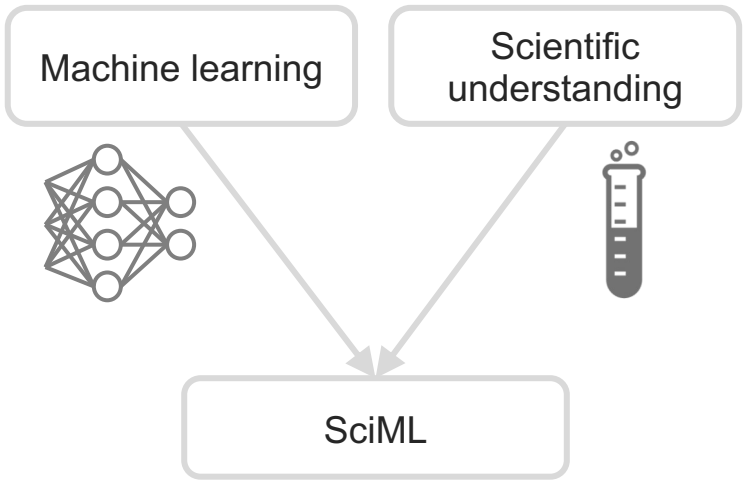
- Lack of **interpretability**
- Requires lots of training **data**
- Poor **generalisation**

What is scientific machine learning?

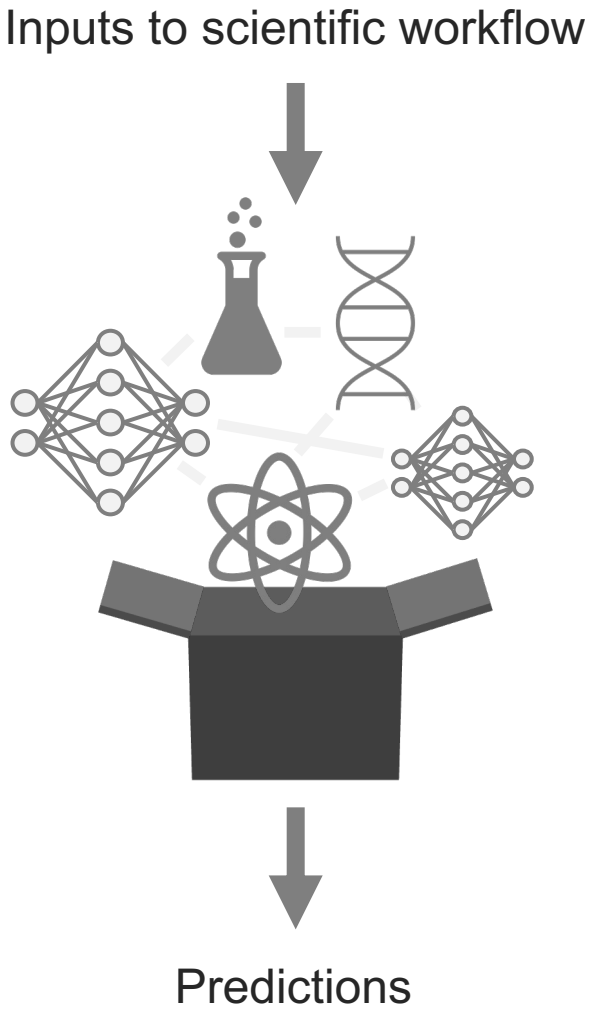




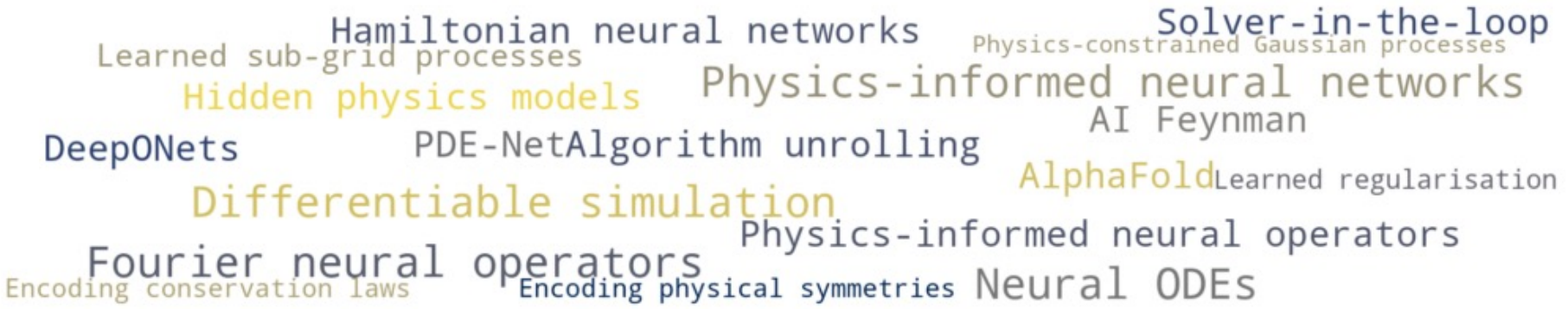
What is scientific machine learning?

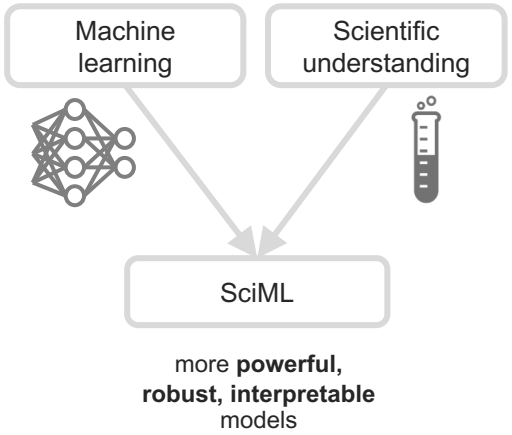


more **powerful, robust, interpretable** models

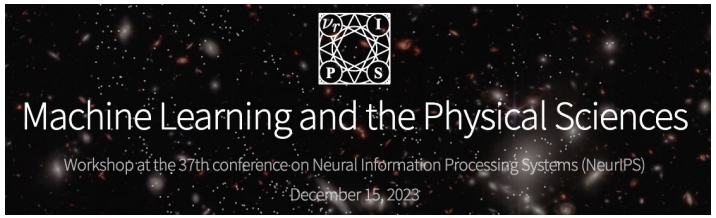


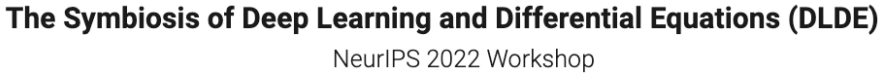
Scientific machine learning

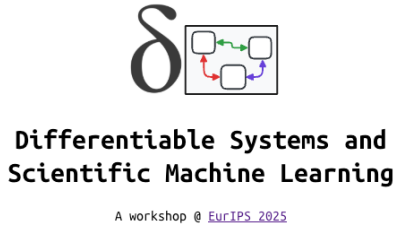


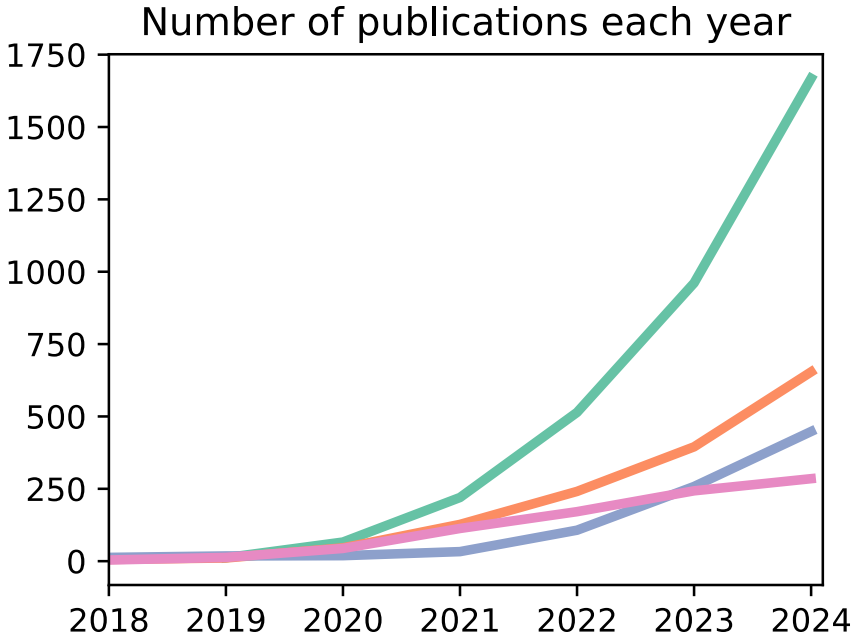


A rapidly growing field



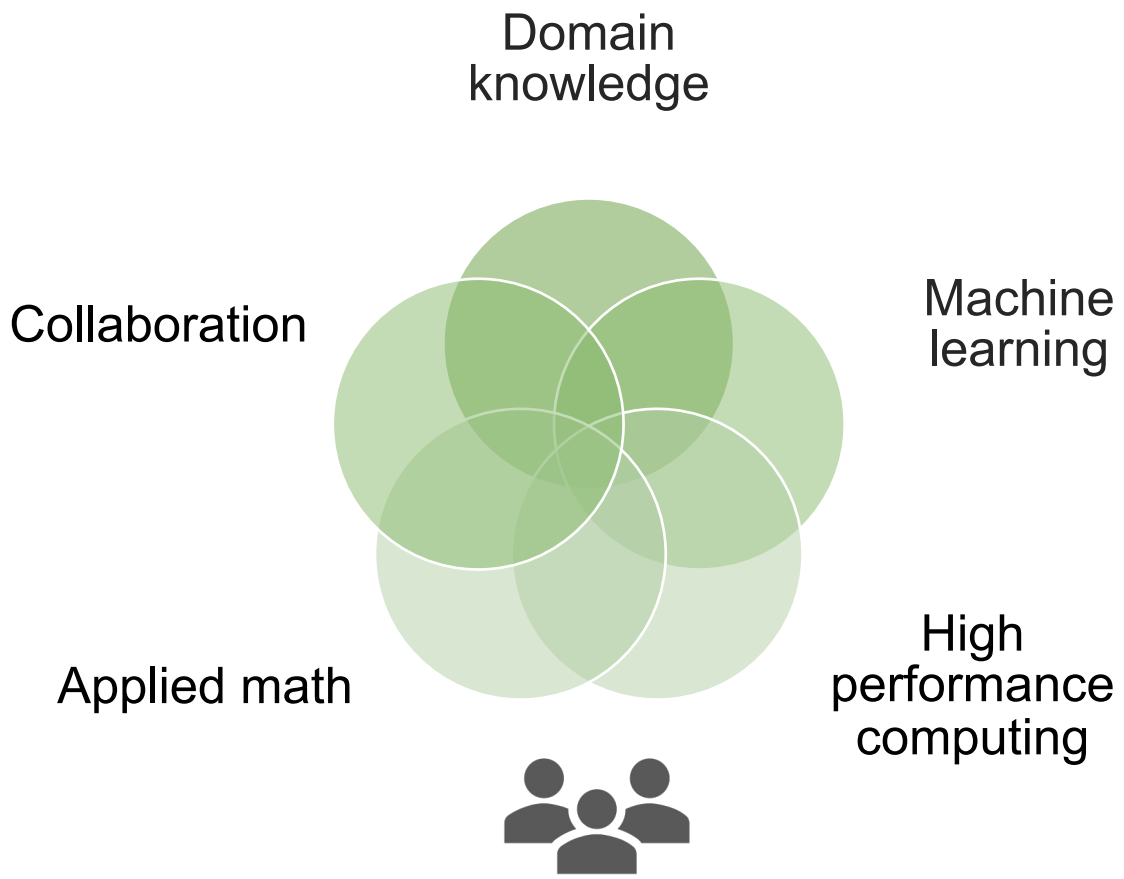






Source: Scopus keyword search (Oct 2025)

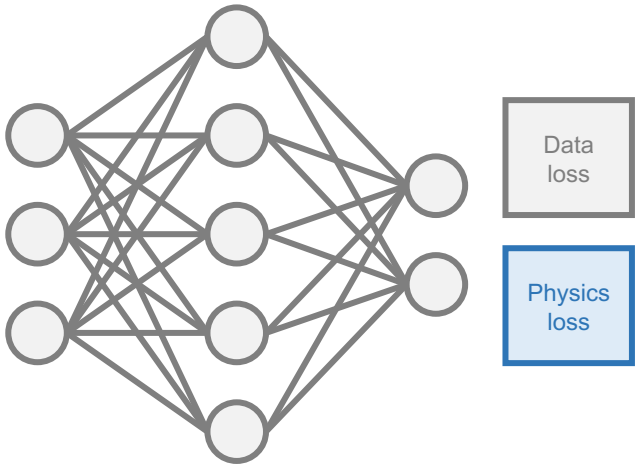
Interdisciplinarity drives SciML



How can we incorporate scientific principles into ML?

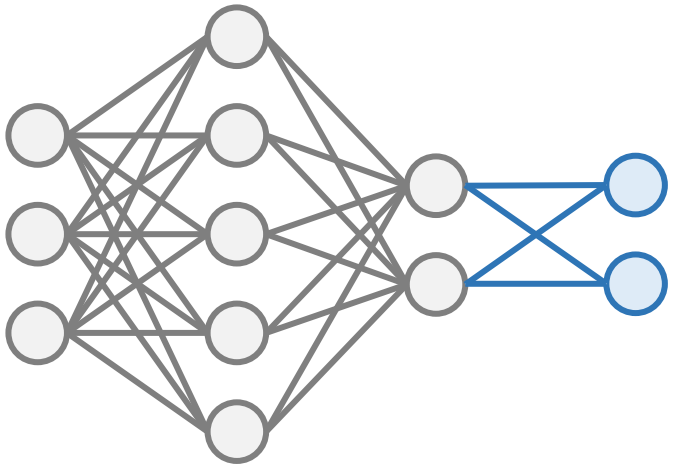
Ways to incorporate scientific principles into machine learning

Loss function



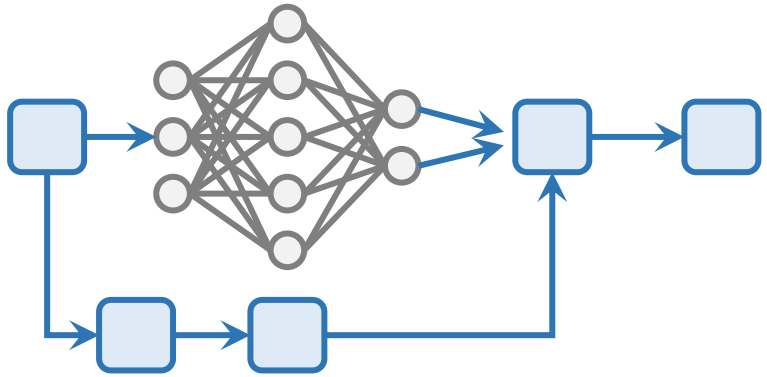
Example:
Physics-informed neural networks (add governing equations to loss function)

Architecture



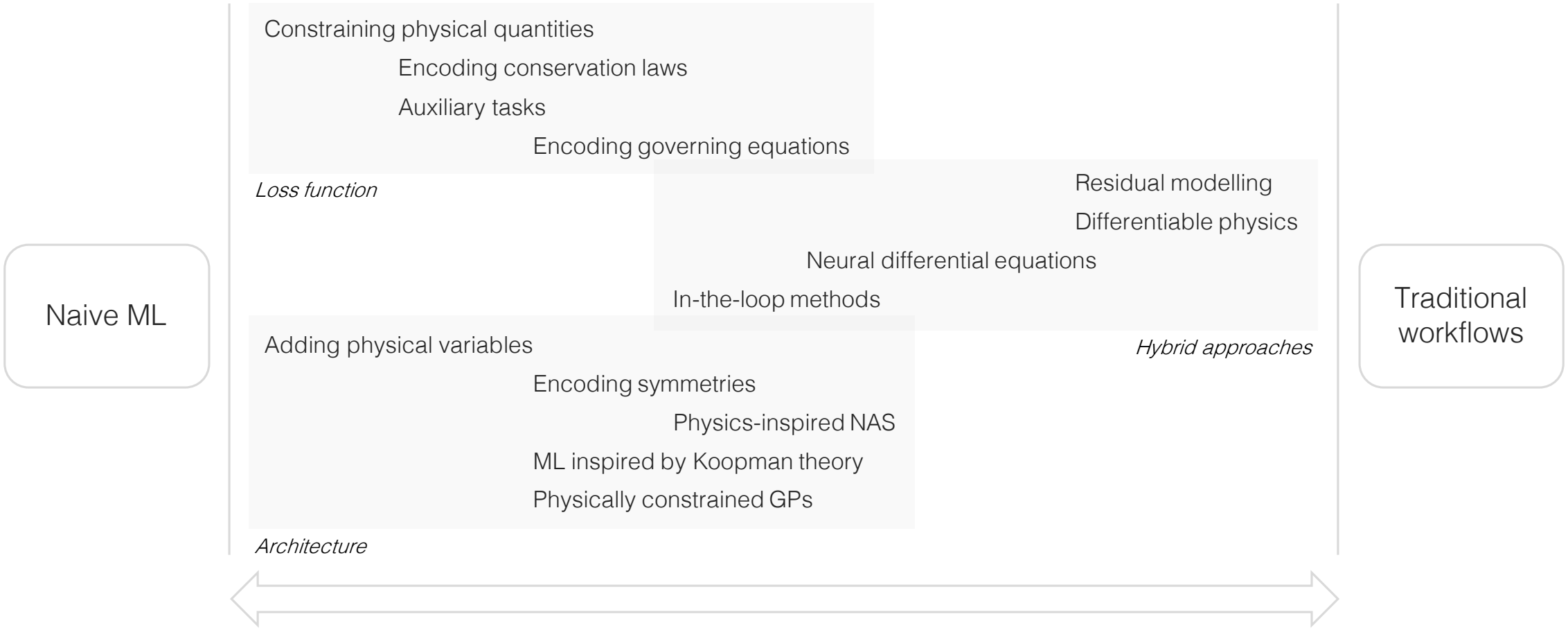
Example:
Encoding symmetries / conservation laws (e.g. energy conservation, rotational invariance)

Hybrid approaches



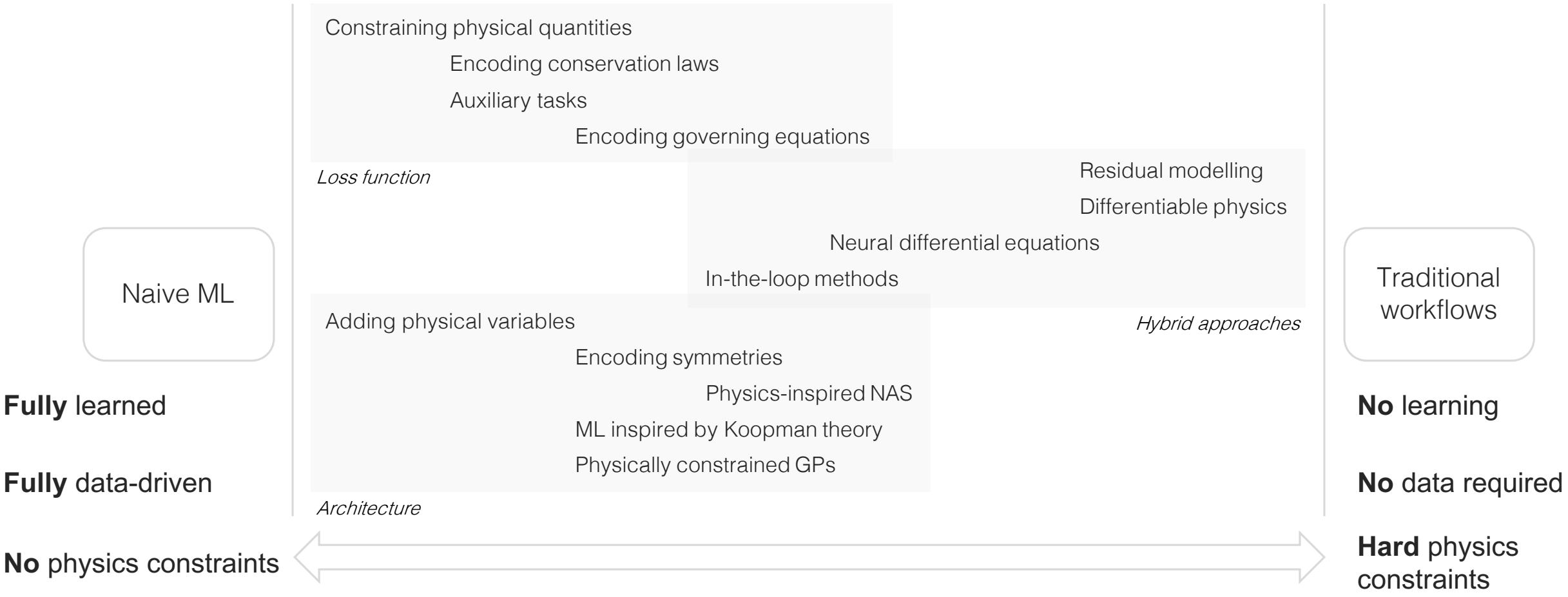
Example:
Neural differential equations (incorporating neural networks into PDE models)

A plethora of SciML techniques



Source: B Moseley, Physics-informed machine learning: from concepts to real-world applications, PhD thesis, 2022

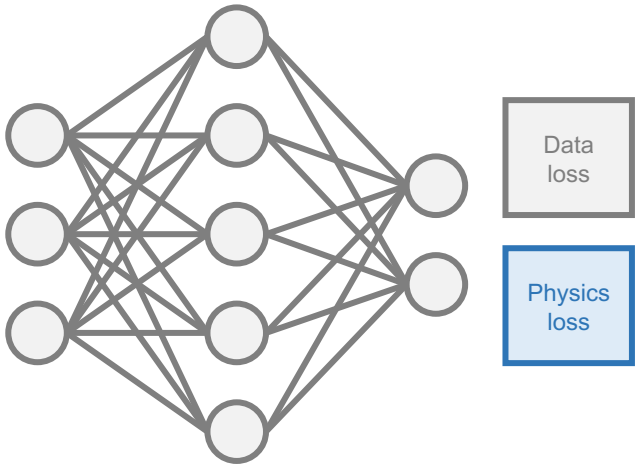
A plethora of SciML techniques



Source: B Moseley, Physics-informed machine learning: from concepts to real-world applications, PhD thesis, 2022

Ways to incorporate scientific principles into machine learning

Loss function



Physics-informed neural networks

Architecture

Example:
Encoding symmetries / conservation laws (e.g. energy conservation, rotational invariance)

Hybrid approaches

Example:
Neural differential equations (incorporating neural networks into PDE models)

AUGUST 28, 2021 BY ME

So, what is a physics-informed neural network?

Machine learning has become increasingly popular across science, but do these algorithms actually "understand" the scientific problems they are trying to solve? In this article we explain physics-informed neural networks, which are a powerful way of incorporating physical principles into machine learning.

A machine learning revolution in science

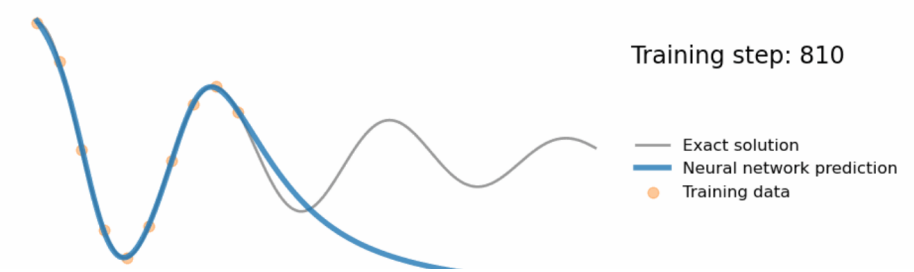
Machine learning has caused a fundamental shift in the scientific method. Traditionally, scientific research has revolved around theory and experiment: one hand-designs a well-defined theory and then continuously refines it using experimental data and analyses it to make new predictions.

But today, with rapid advances in the field of machine learning and dramatically increasing amounts of scientific data, data-driven approaches have become inc
theory is not required, and instead a machine learning algor
problem using data alone.

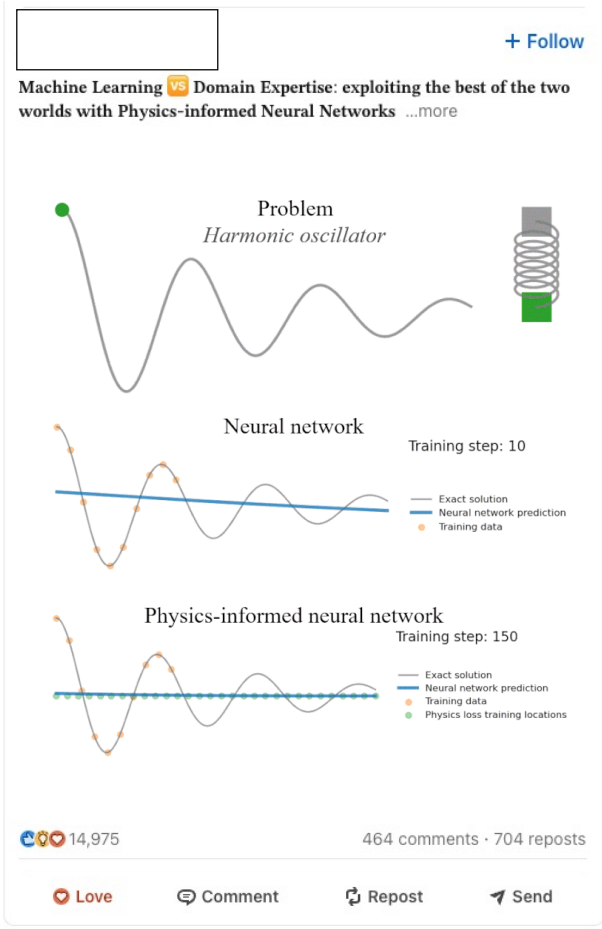
Learning to model experimental data

Let's look at one way machine learning can be used for scientific research. Imagine we are given some experimental data points that come from some unknown physical phenomenon, e.g. the orange points in the animation below.

A common scientific task is to find a *model* which is able to accurately predict new experimental measurements given this data.

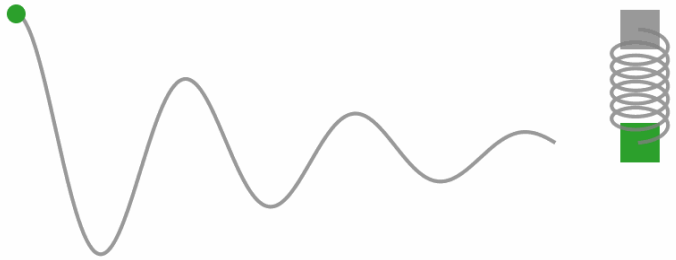


So, what is a physics-informed neural network?



What is a physics-informed neural network?

Problem: damped harmonic oscillator



$$m\frac{d^2u}{dt^2} + \mu\frac{du}{dt} + ku = 0$$

Initial conditions:

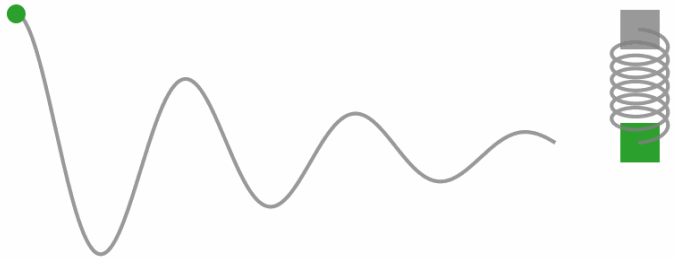
$$u(t=0) = 1$$
$$u_t(t=0) = 0$$

$u =$ displacement
$m =$ mass of oscillator
$\mu =$ coefficient of friction
$k =$ spring constant

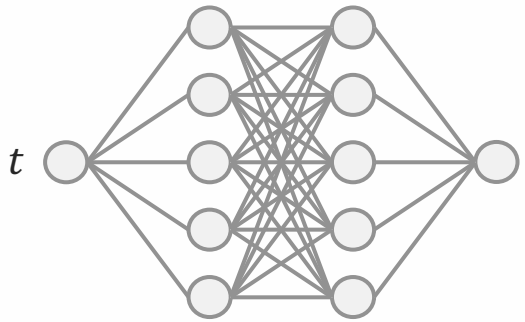
Raissi et al, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, JCP (2018)
Lagaris et al, Artificial neural networks for solving ordinary and partial differential equations, IEEE (1998)

What is a physics-informed neural network?

Problem: damped harmonic oscillator



$$m\frac{d^2u}{dt^2} + \mu\frac{du}{dt} + ku = 0$$



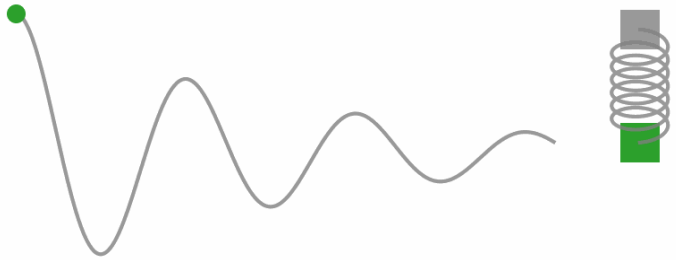
Key idea: use a neural network to directly approximate the solution

$$NN(t, \boldsymbol{\theta}) \approx u(t)$$

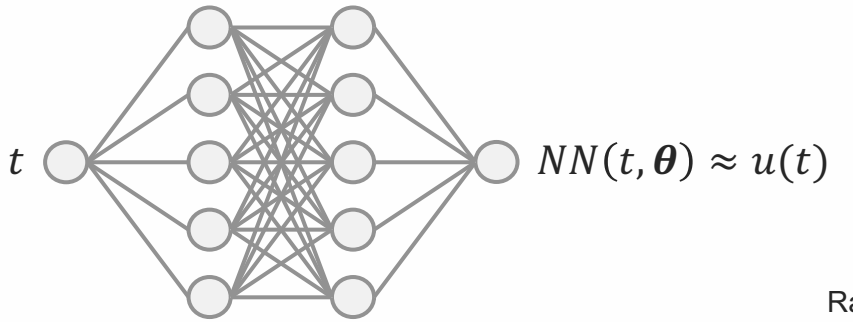
Raissi et al, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, JCP (2018)
Lagaris et al, Artificial neural networks for solving ordinary and partial differential equations, IEEE (1998)

What is a physics-informed neural network?

Problem: damped harmonic oscillator



$$m\frac{d^2u}{dt^2} + \mu\frac{du}{dt} + ku = 0$$



Boundary loss

$$L(\theta) = \lambda_1(\hat{u}(t = 0, \boldsymbol{\theta}) - 1)^2 + \lambda_2\left(\frac{d\hat{u}}{dt}(t = 0, \boldsymbol{\theta}) - 0\right)^2$$

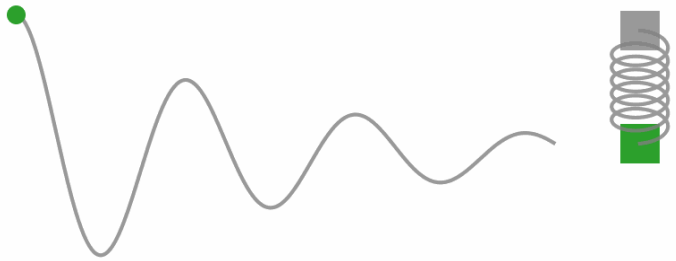
Physics loss (aka PDE residual)

$$+ \frac{1}{N_p}\sum_i^{N_p}\left(\left[m\frac{d^2}{dt^2} + \mu\frac{d}{dt} + k\right]\hat{u}(t_i, \boldsymbol{\theta})\right)^2$$

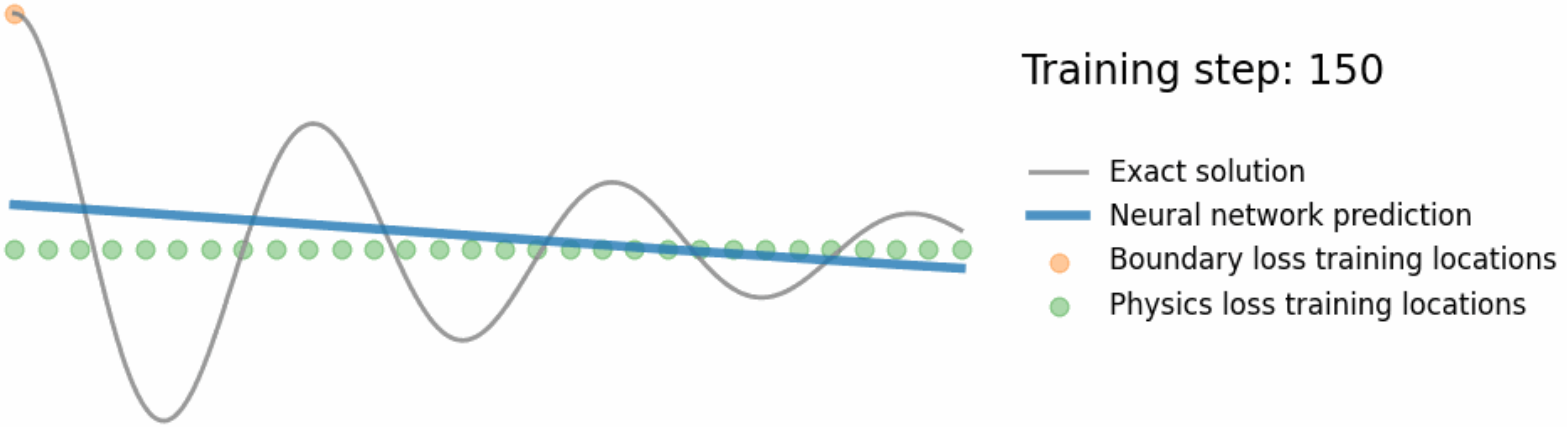
Raissi et al, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, JCP (2018)
Lagaris et al, Artificial neural networks for solving ordinary and partial differential equations, IEEE (1998)

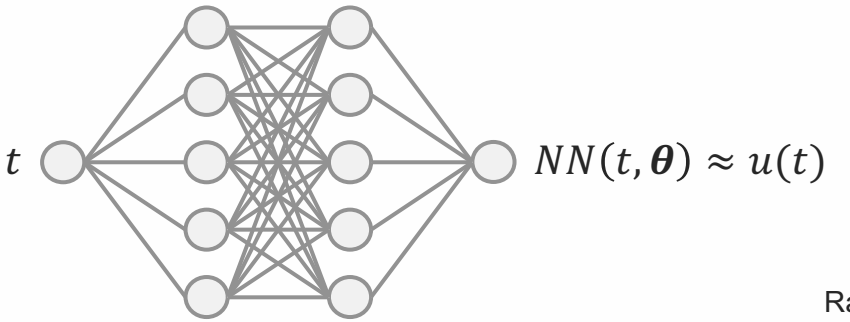
What is a physics-informed neural network?

Problem: damped harmonic oscillator



$$m\frac{d^2u}{dt^2} + \mu\frac{du}{dt} + ku = 0$$





Boundary loss

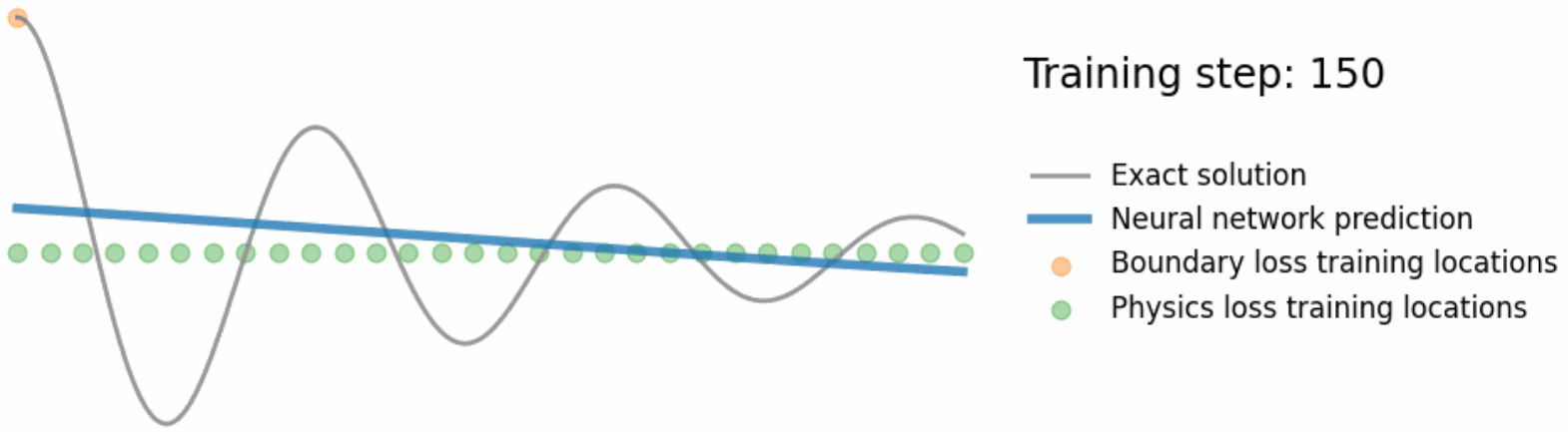
$$L(\theta) = \lambda_1(\hat{u}(t=0, \boldsymbol{\theta}) - 1)^2 + \lambda_2\left(\frac{d\hat{u}}{dt}(t=0, \boldsymbol{\theta}) - 0\right)^2$$

Physics loss (aka PDE residual)

$$+ \frac{1}{N_p}\sum_i^{N_p}\left(\left[m\frac{d^2}{dt^2} + \mu\frac{d}{dt} + k\right]\hat{u}(t_i, \boldsymbol{\theta})\right)^2$$

Raissi et al, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, JCP (2018)
Lagaris et al, Artificial neural networks for solving ordinary and partial differential equations, IEEE (1998)

What is a physics-informed neural network?



- $\{t_i\}_{i=1}^{N_p}$ are known as **collocation points**, which are sampled throughout the domain
- λ_1, λ_2 are scalar **hyperparameters** which **balance** the contribution of each loss term

Boundary loss

$$L(\theta) = \lambda_1 (\hat{u}(t=0, \boldsymbol{\theta}) - 1)^2 + \lambda_2 \left(\frac{d\hat{u}}{dt}(t=0, \boldsymbol{\theta}) - 0 \right)^2$$

Physics loss (aka PDE residual)

$$+ \frac{1}{N_p} \sum_i^{N_p} \left(\left[m \frac{d^2}{dt^2} + \mu \frac{d}{dt} + k \right] \hat{u}(t_i, \boldsymbol{\theta}) \right)^2$$

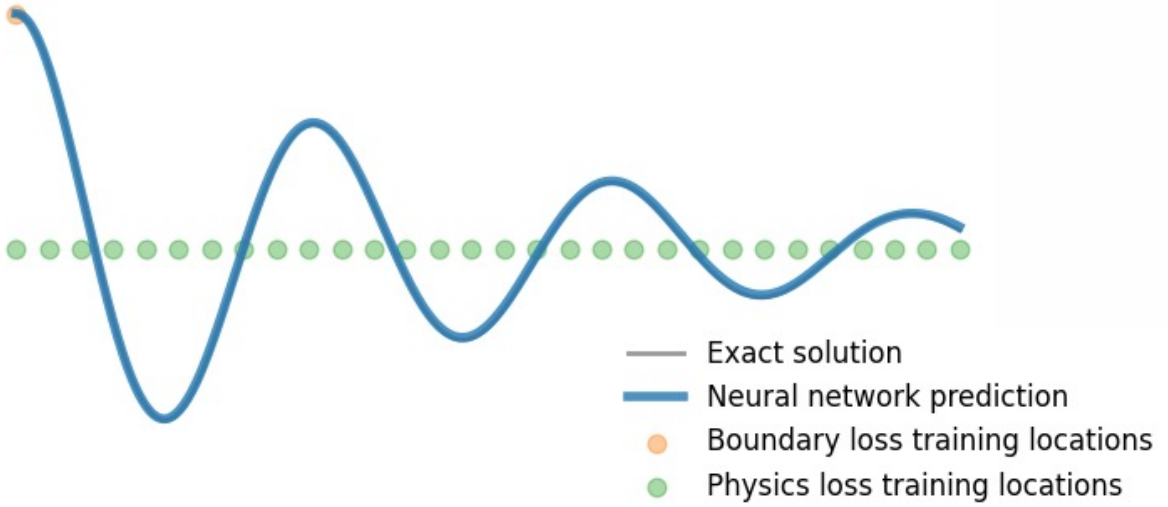
Raissi et al, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, JCP (2018)
Lagaris et al, Artificial neural networks for solving ordinary and partial differential equations, IEEE (1998)

What is a physics-informed neural network?

Training loop:

1. Sample boundary/ physics training points
2. Compute network outputs
3. Compute 1st and 2nd order gradient of network output **with respect to network input**
4. Compute loss
5. Compute gradient of loss function **with respect to network parameters**
6. Take gradient descent step

How can we compute the gradients (e.g. $\frac{d\hat{u}}{dt}$ and $\frac{dL}{d\theta}$) in steps 3 and 5?



Boundary loss

$$L(\theta) = \lambda_1(\hat{u}(t=0,\boldsymbol{\theta}) - 1)^2 + \lambda_2\left(\frac{d\hat{u}}{dt}(t=0,\boldsymbol{\theta}) - 0\right)^2$$

Physics loss (aka PDE residual)

$$+ \frac{1}{N_p}\sum_i^{N_p}\left(\left[m\frac{d^2}{dt^2} + \mu\frac{d}{dt} + k\right]\hat{u}(t_i,\boldsymbol{\theta})\right)^2$$

What is a physics-informed neural network?

Training loop:

1. Sample boundary/ physics training points
2. Compute network outputs
3. Compute 1st and 2nd order gradient of network output **with respect to network input**
4. Compute loss
5. Compute gradient of loss function **with respect to network parameters**
6. Take gradient descent step

How can we compute the gradients (e.g. $\frac{d\hat{u}}{dt}$ and $\frac{dL}{d\theta}$) in steps 3 and 5?

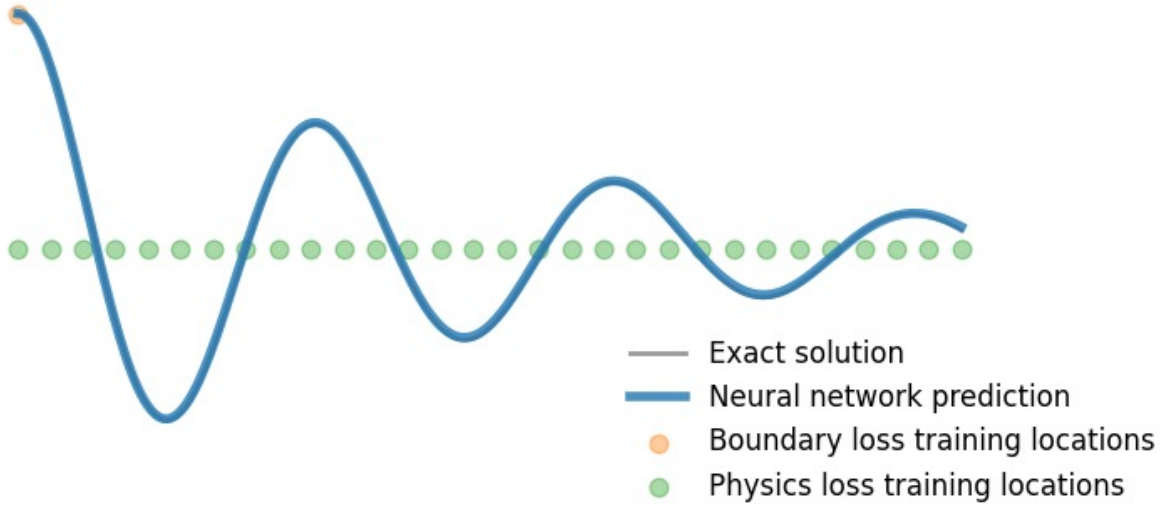
We can use **autograd!**

```
dudt = torch.autograd.grad(u, t)
```

torch.autograd.grad(*outputs*, *inputs*, *grad_outputs=None*, *retain_graph=None*, *create_graph=False*, *only_inputs=True*, *allow_unused=False*, *is_grads_batched=False*) [SOURCE]

Computes and returns the sum of gradients of outputs with respect to the inputs.

`grad_outputs` should be a sequence of length matching `output` containing the "vector" in vector-Jacobian product, usually the pre-computed gradients w.r.t. each of the outputs. If an output doesn't require_grad, then the gradient can be `None`).



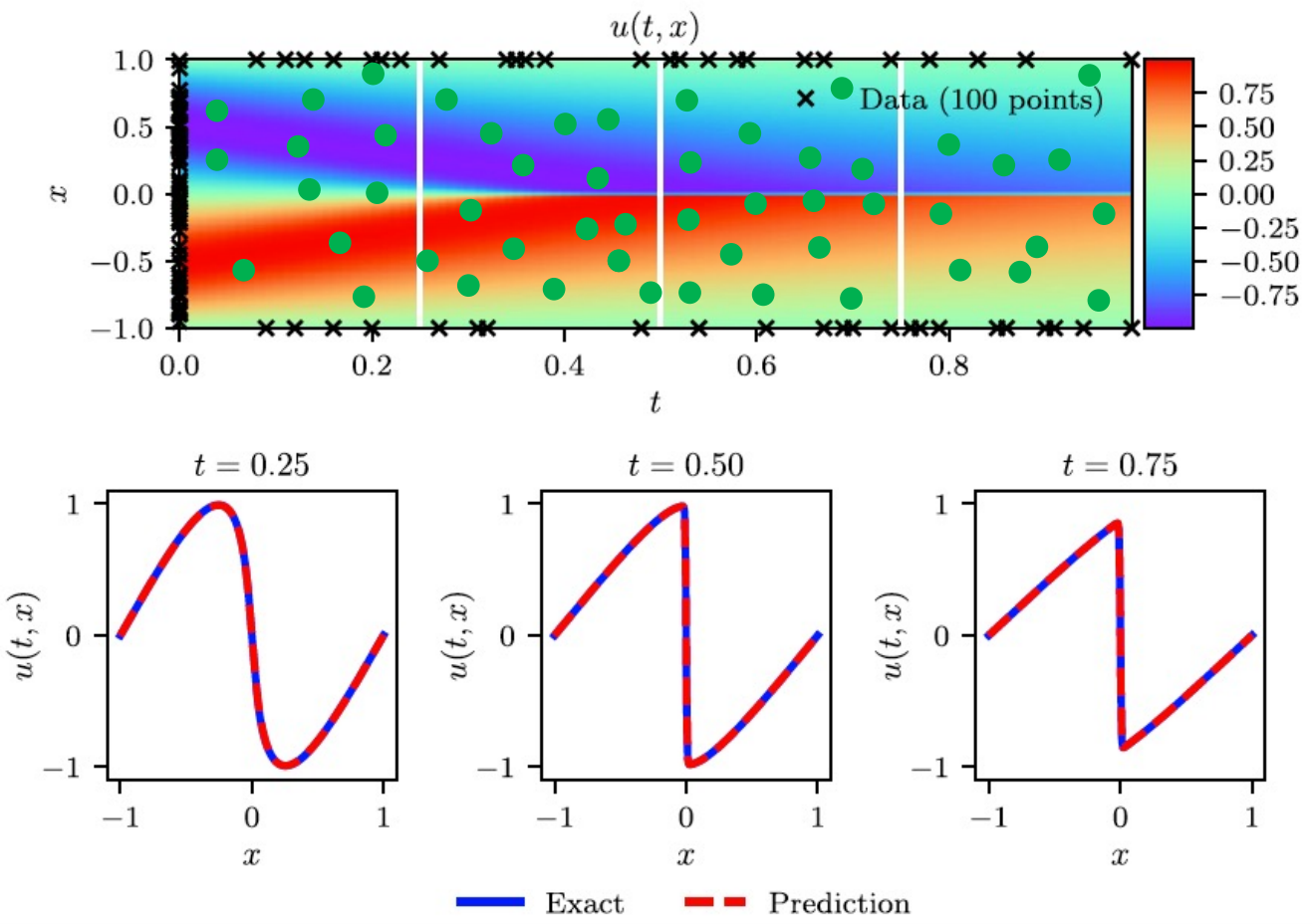
Boundary loss

$$L(\theta) = \lambda_1(\hat{u}(t=0,\boldsymbol{\theta}) - 1)^2 + \lambda_2\left(\frac{d\hat{u}}{dt}(t=0,\boldsymbol{\theta}) - 0\right)^2$$

Physics loss (aka PDE residual)

$$+ \frac{1}{N_p}\sum_{i}^{N_p}\left(\left[m\frac{d^2}{dt^2} + \mu\frac{d}{dt} + k\right]\hat{u}(t_i,\boldsymbol{\theta})\right)^2$$

PINNs for solving viscous Burgers' equation

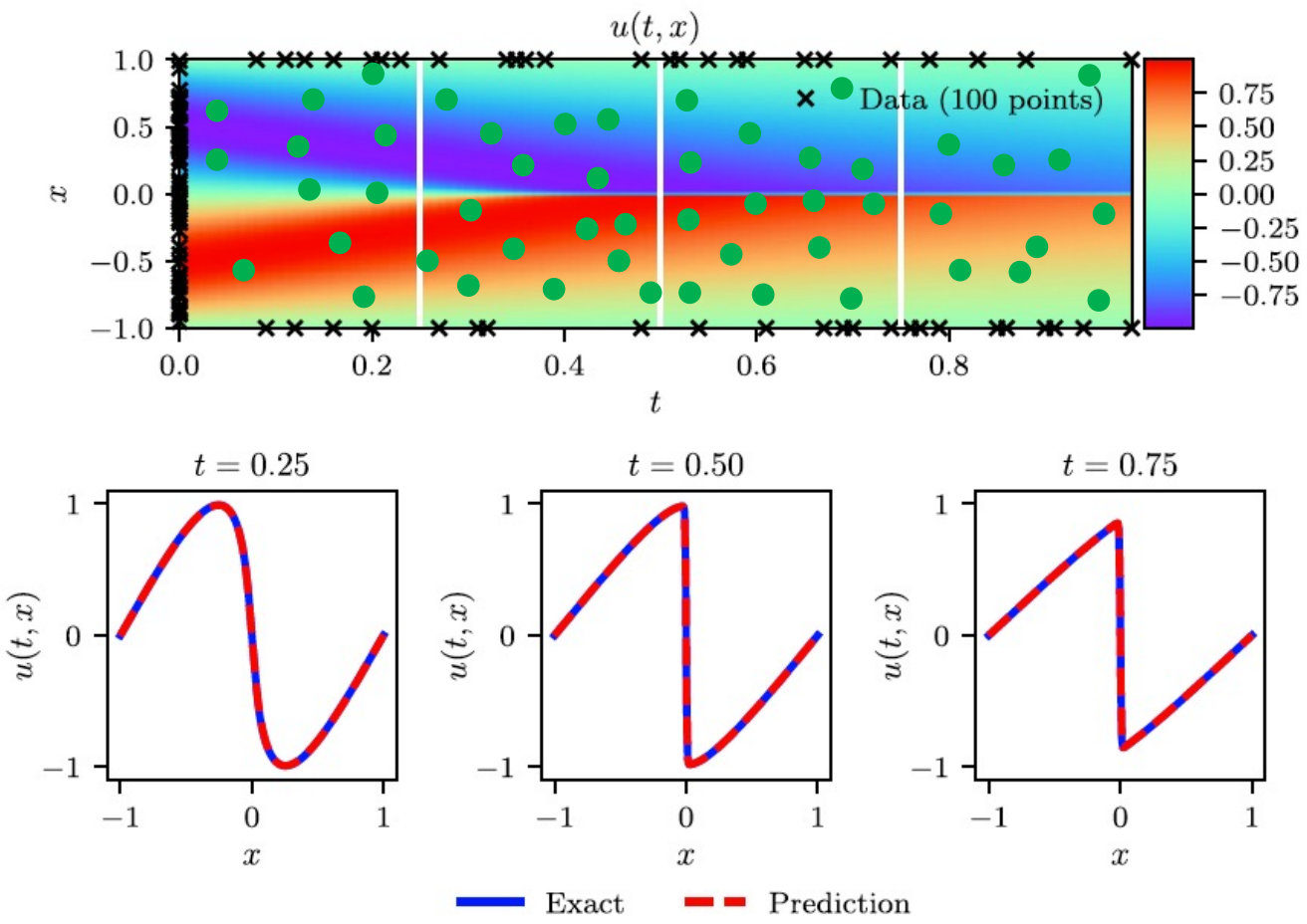


$$L_b(\boldsymbol{\theta}) = \frac{\lambda_1}{N_{b1}} \sum_j^{N_{b1}} \left(\hat{u}(x_j, 0, \boldsymbol{\theta}) + \sin(\pi x_j)\right)^2 + \frac{\lambda_2}{N_{b2}} \sum_k^{N_{b2}} (\hat{u}(-1, t_k, \boldsymbol{\theta}) - 0)^2 + \frac{\lambda_3}{N_{b3}} \sum_l^{N_{b3}} (\hat{u}(+1, t_l, \boldsymbol{\theta}) - 0)^2$$

$$L_p(\boldsymbol{\theta}) = \frac{1}{N_p} \sum_i^{N_p} \left(\left(\frac{\partial \hat{u}}{\partial t} + \hat{u} \frac{\partial \hat{u}}{\partial x} - \nu \frac{\partial^2 \hat{u}}{\partial x^2} \right) (x_i, t_i, \boldsymbol{\theta}) \right)^2$$

Raissi et al, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, JCP (2018)

PINNs for solving viscous Burgers' equation



Raissi et al, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, JCP (2018)

$$L_b(\boldsymbol{\theta}) = \frac{\lambda_1}{N_{b1}} \sum_j^{N_{b1}} \left(\hat{u}(x_j, 0, \boldsymbol{\theta}) + \sin(\pi x_j)\right)^2 + \frac{\lambda_2}{N_{b2}} \sum_k^{N_{b2}} (\hat{u}(-1, t_k, \boldsymbol{\theta}) - 0)^2 + \frac{\lambda_3}{N_{b3}} \sum_l^{N_{b3}} (\hat{u}(+1, t_l, \boldsymbol{\theta}) - 0)^2$$

$$L_p(\boldsymbol{\theta}) = \frac{1}{N_p} \sum_i^{N_p} \left(\left(\frac{\partial \hat{u}}{\partial t} + \hat{u} \frac{\partial \hat{u}}{\partial x} - \nu \frac{\partial^2 \hat{u}}{\partial x^2} \right) (x_i, t_i, \boldsymbol{\theta}) \right)^2$$

$\nu = 0.01/\pi$
$N_p = 10{,}000$ (Latin hypercube sampling)
$N_{b1} + N_{b2} + N_{b3} = 100$
Fully connected network with 9 layers, 20 hidden units (3021 free parameters)
Tanh activation function
L-BFGS optimiser

PINNs – an entire research field

SPRINGER LINK

Find a journal | Publish with us | Track your research | Search

Home > Journal of Scientific Computing > Article

Scientific Machine Learning Through Physics–Informed Neural Networks: Where we are and What's Next

Open access | Published: 26 July 2022

Volume 92, article number 88, (2022) Cite this article

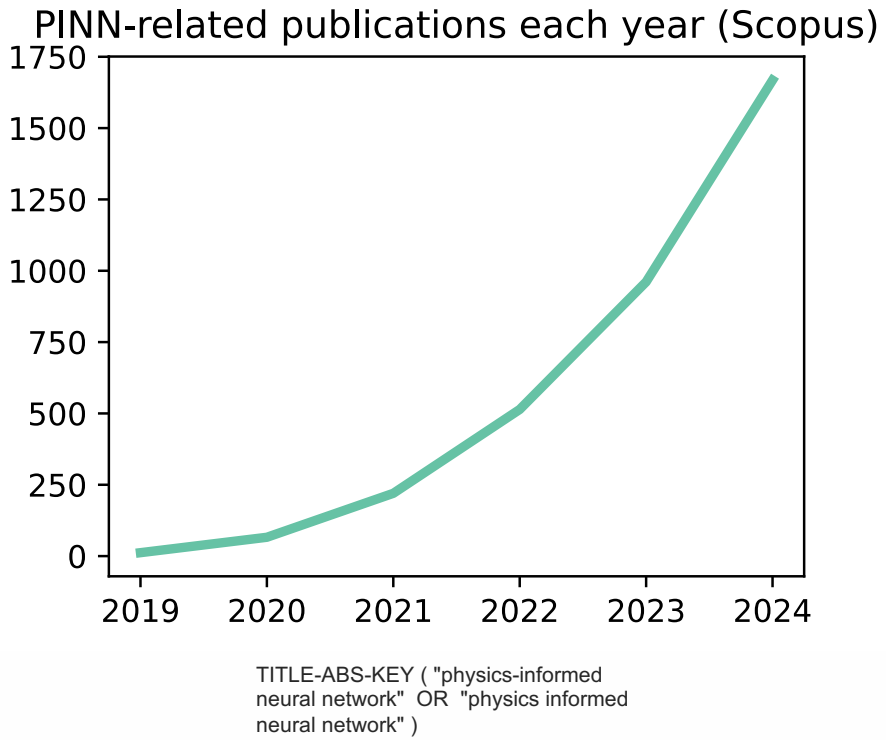
Download PDF | You have full access to this open access article

Salvatore Cuomo, Vincenzo Schiano Di Cola, Fabio Giampaolo, Gianluigi Rozza, Maziar Raissi & Francesco Piccialli

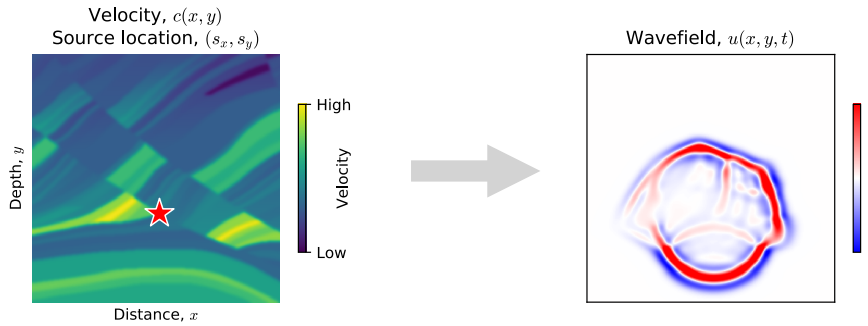
67k Accesses | 274 Citations | 7 Altmetric | Explore all metrics →

Abstract

Physics-Informed Neural Networks (PINN) are neural networks (NNs) that encode model equations, like Partial Differential Equations (PDE), as a component of the neural network itself. PINNs are nowadays used to solve PDEs, fractional equations, integral-differential equations, and stochastic PDEs. This novel methodology has arisen as a multi-task learning framework in which a NN must fit observed data while reducing a PDE residual. This article provides a comprehensive review of the literature on PINNs: while the primary goal of the study was to characterize these networks and their related advantages and



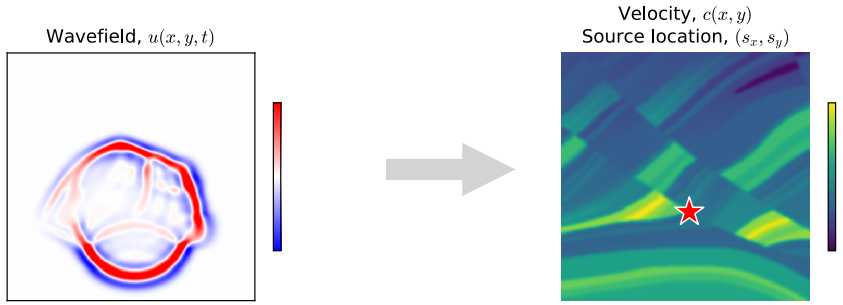
Key scientific tasks



Forward simulation

Estimate wavefield $u(x, y, t)$
Given velocity $c(x, y)$
and source location (s_x, s_y)

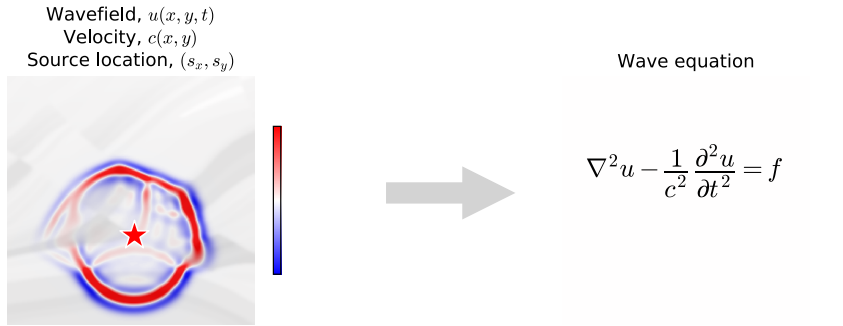
$$b = F(a)$$



Inversion

Estimate velocity $c(x, y)$
and source location (s_x, s_y)
Given wavefield $u(x, y, t)$

$$b = F(a)$$



Equation discovery

Estimate governing equation
Given wavefield $u(x, y, t)$,
velocity $c(x, y)$,
and source location (s_x, s_y)

$$b = F(a)$$

PINNs can be used to solve **forward**, **inverse** and **equation discovery** problems related to PDEs

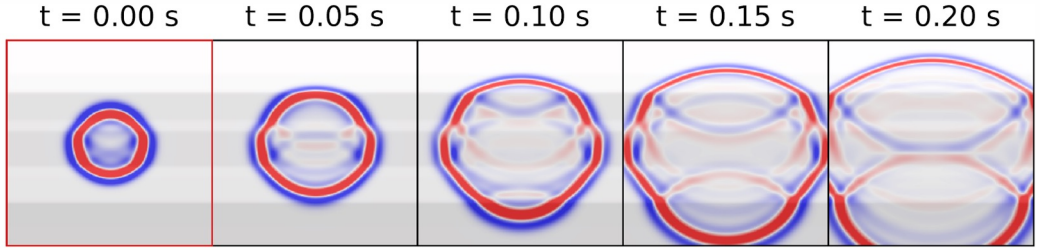
a = set of input conditions

F = physical model of the system (usually a PDE)

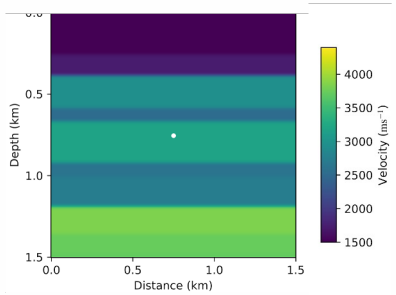
b = resulting properties given F and a

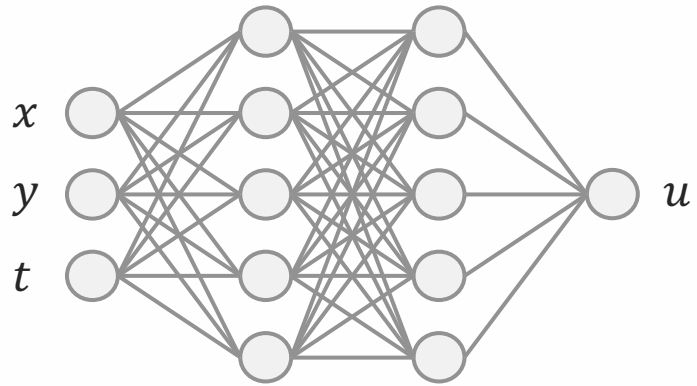
PINNs for solving wave equation

Ground truth FD simulation



Velocity model, $c(x, y)$





Moseley et al, Solving the wave equation with physics-informed deep learning, ArXiv (2020)

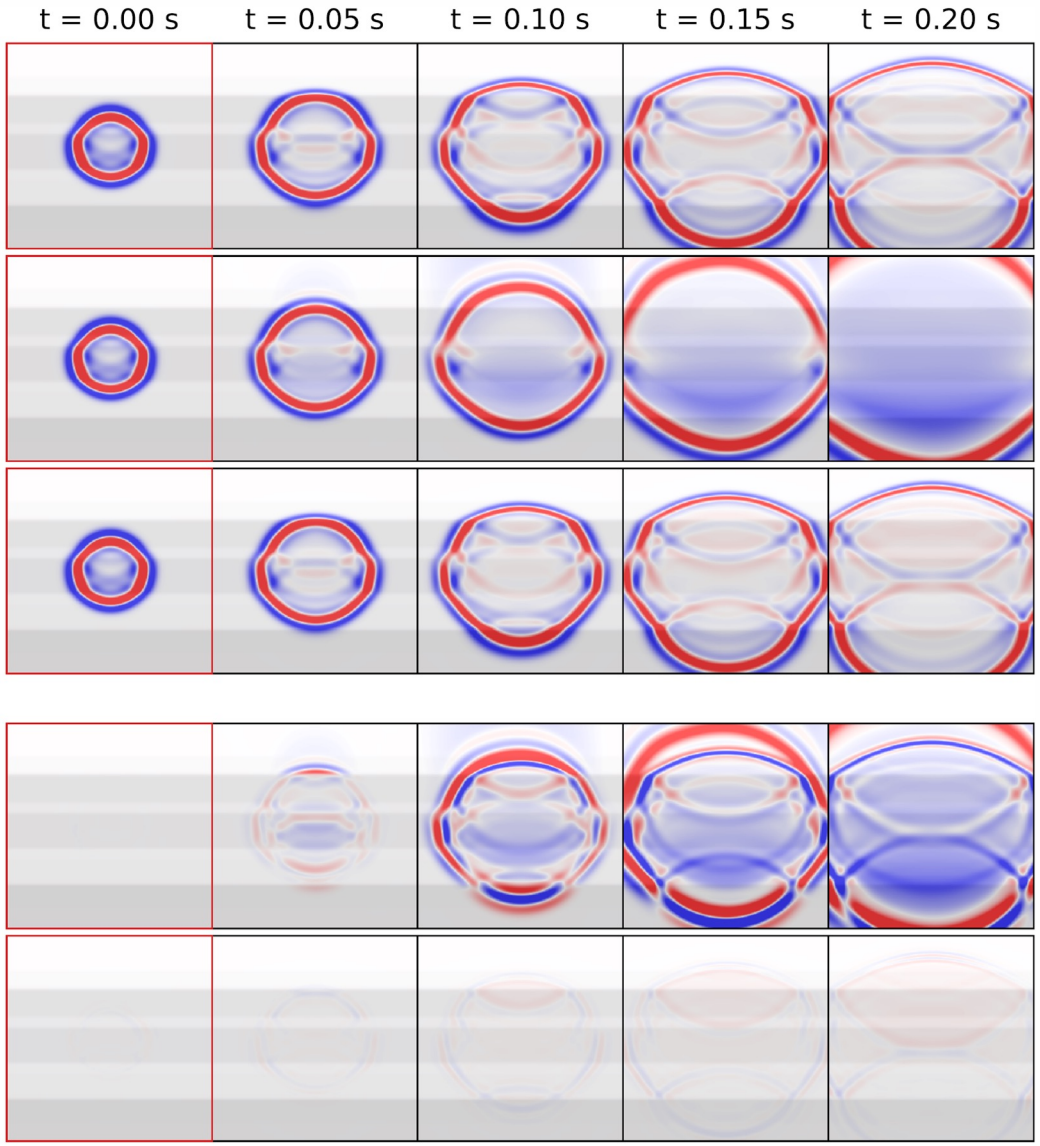
$$L_b(\boldsymbol{\theta}) = \frac{\lambda}{N_b}\sum_{j}^{N_b}\left(\hat{u}(x_j, y_j, t_j, \boldsymbol{\theta}) - u_{FD}(x_j, y_j, t_j)\right)^2$$

Boundary data from FD simulation (first 0.02 seconds)

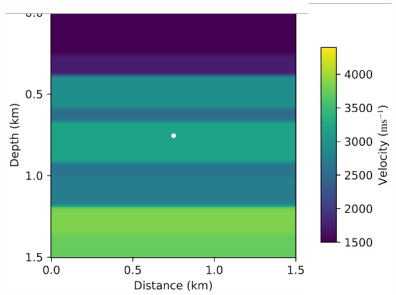
$$L_p(\boldsymbol{\theta}) = \frac{1}{N_p}\sum_{i}^{N_p}\left(\left[\nabla^2 - \frac{1}{c(x_i)^2}\frac{\partial^2}{\partial t^2}\right]\hat{u}(x_i, y_i, t_i, \boldsymbol{\theta})\right)^2$$

Physics loss training points randomly sampled over entire x-y-t domain (up to 0.2 seconds)

PINNs for solving wave equation



Velocity model, $c(x, y)$

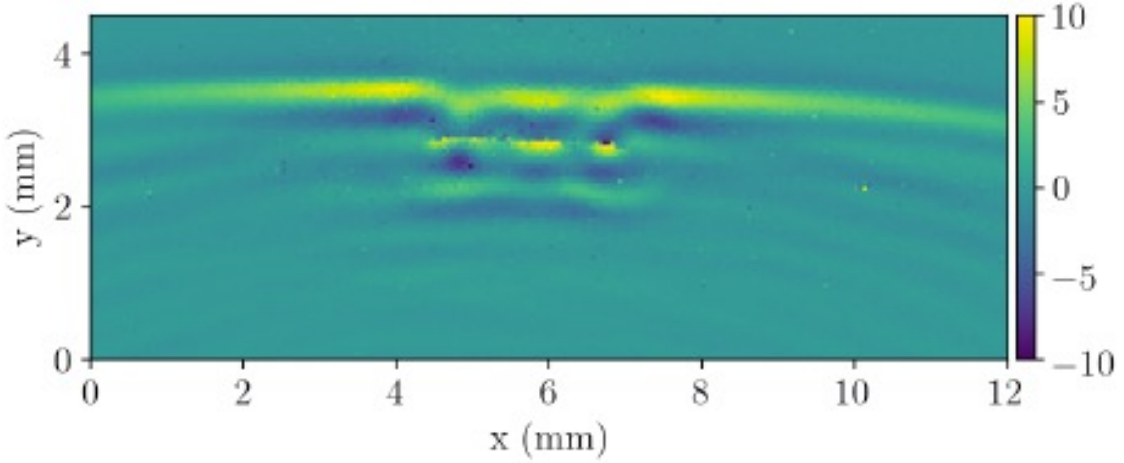


Moseley et al, Solving the wave equation with physics-informed deep learning, ArXiv (2020)

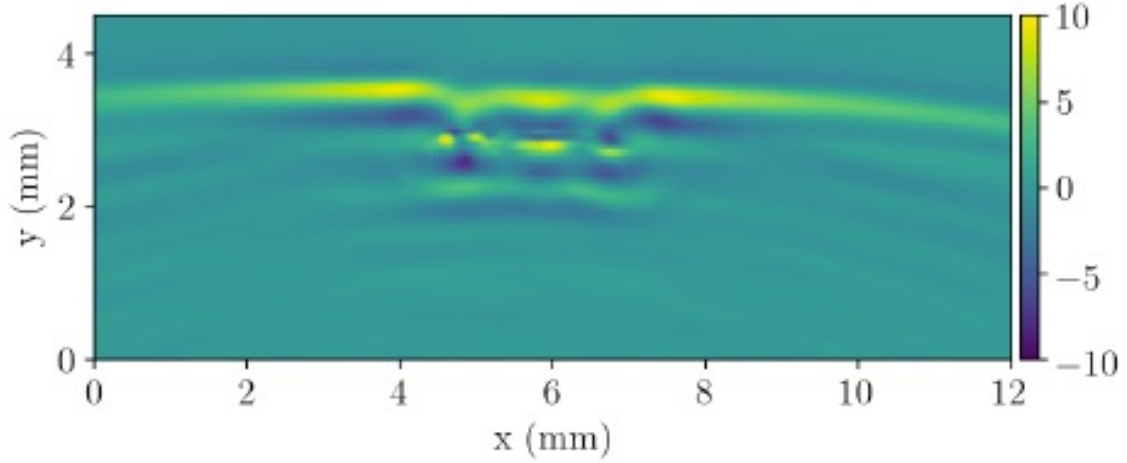
Mini-batch size $N_b = N_p = 500$ (random sampling)
Fully connected network with 10 layers, 1024 hidden units
Softplus activation
Adam optimiser

PINNs for inverse problems

Shukla et al, Physics-Informed Neural Network for Ultrasound Nondestructive Quantification of Surface Breaking Cracks, Journal of Nondestructive Evaluation (2020)



(a) Actual data at $t = 12.38\ \mu s$.

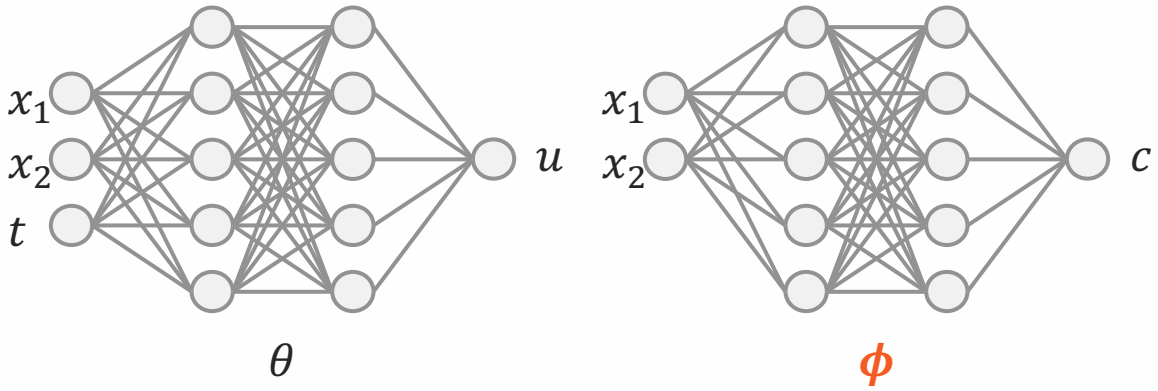


(b) Data recovered from PINN simulation at $t = 12.38\ \mu s$.

$$L_d(\boldsymbol{\theta}) = \frac{\lambda}{N_b}\sum_{j}^{N_b}\left(NN(x_j, y_j, t_j, \boldsymbol{\theta}) - u_{data}(x_j, y_j, t_j)\right)^2$$

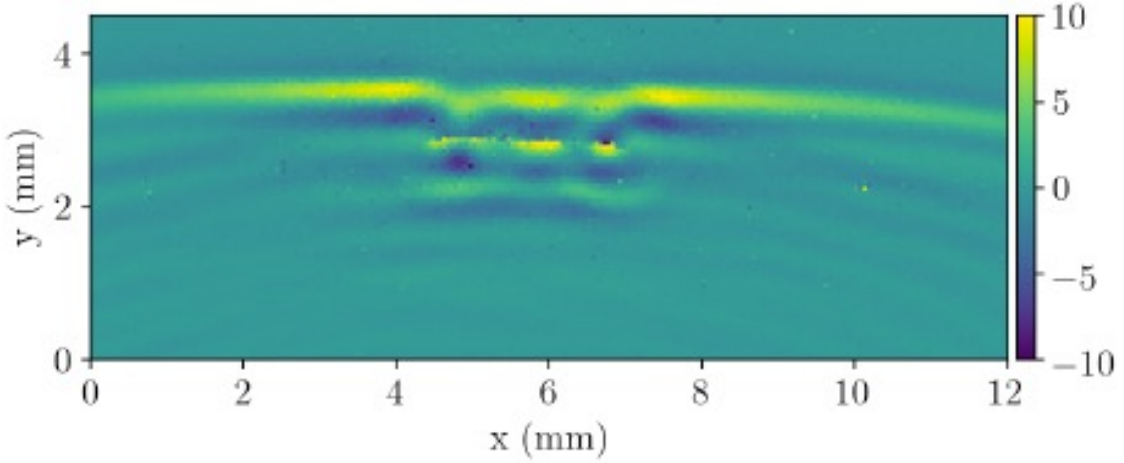
$$L_p(\boldsymbol{\theta}, \boldsymbol{\phi}) = \frac{1}{N_p}\sum_{i}^{N_p}\left(\left[\nabla^2 - \frac{1}{c(x_i, \boldsymbol{\phi})^2}\frac{\partial^2}{\partial t^2}\right] NN(x_i, y_i, t_i, \boldsymbol{\theta})\right)^2$$

Treat velocity model as **another** neural network, and simultaneously learn it

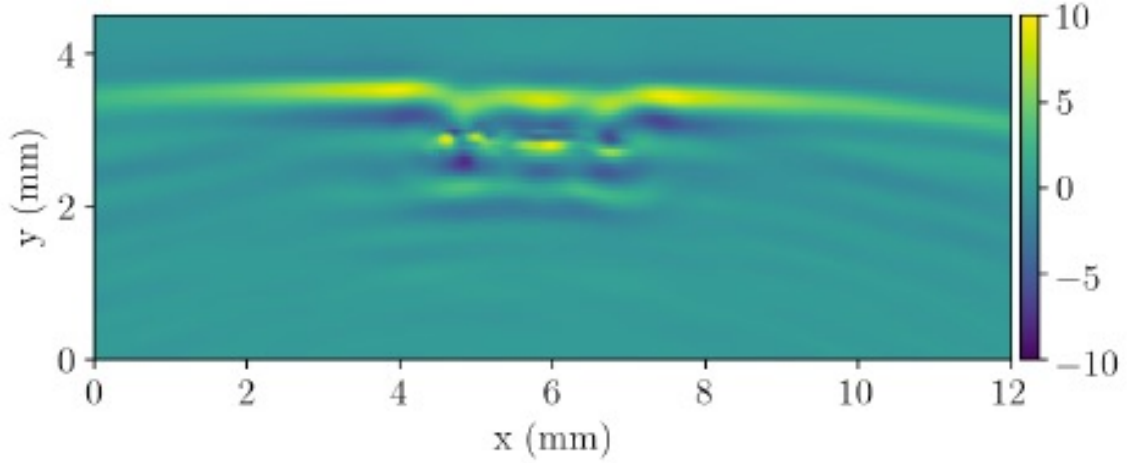


PINNs for inverse problems

Shukla et al, Physics-Informed Neural Network for Ultrasound Nondestructive Quantification of Surface Breaking Cracks, Journal of Nondestructive Evaluation (2020)



(a) Actual data at $t = 12.38\ \mu s$.

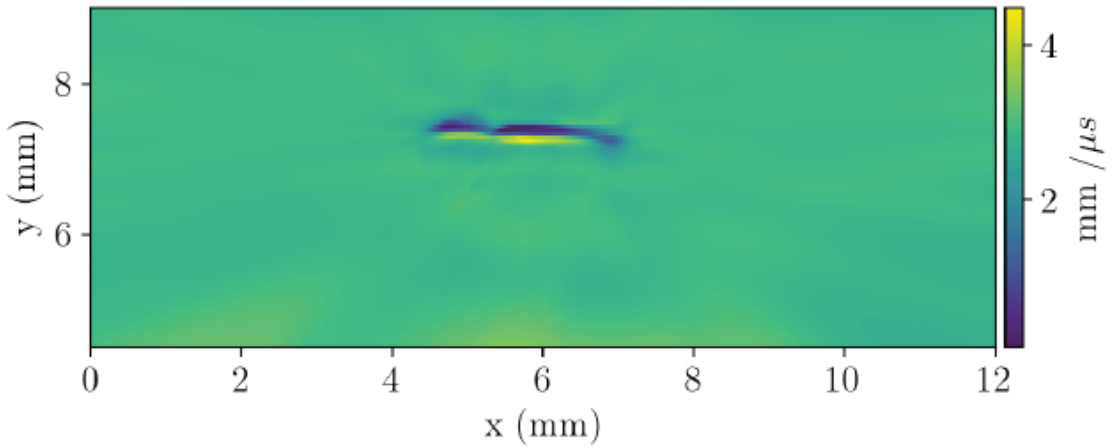


(b) Data recovered from PINN simulation at $t = 12.38\ \mu s$.

$$L_d(\boldsymbol{\theta}) = \frac{\lambda}{N_b}\sum_{j}^{N_b}\left(NN(x_j, y_j, t_j, \boldsymbol{\theta}) - u_{data}(x_j, y_j, t_j)\right)^2$$

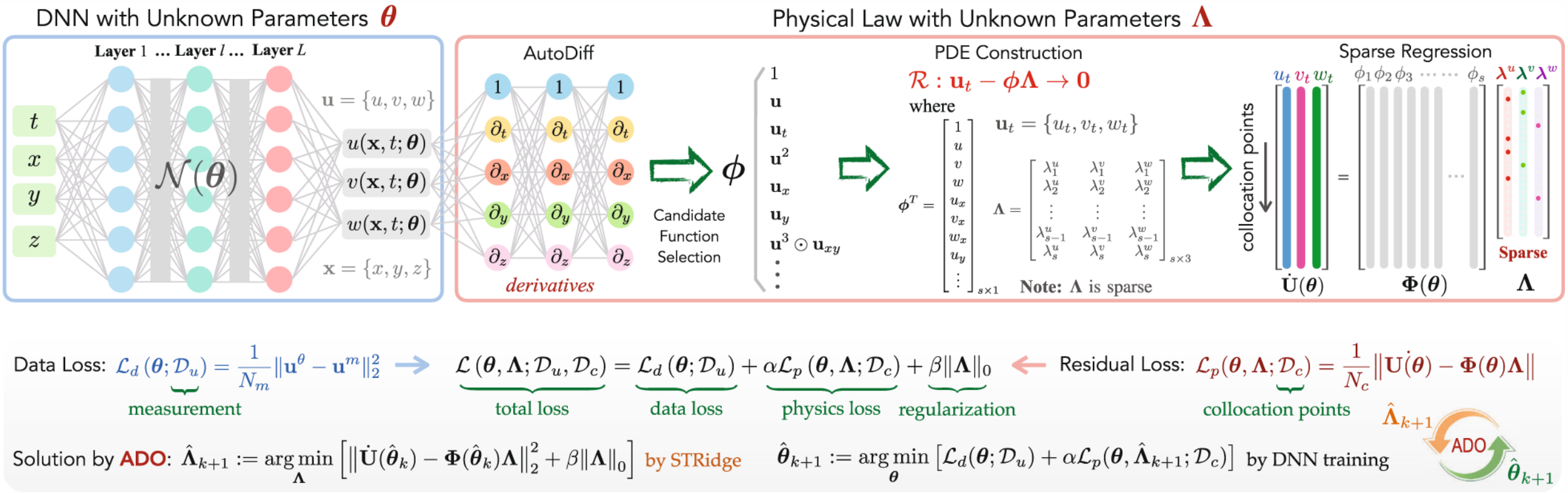
$$L_p(\boldsymbol{\theta}, \boldsymbol{\phi}) = \frac{1}{N_p}\sum_{i}^{N_p}\left(\left[\nabla^2 - \frac{1}{c(x_i, \boldsymbol{\phi})^2}\frac{\partial^2}{\partial t^2}\right] NN(x_i, y_i, t_i, \boldsymbol{\theta})\right)^2$$

Treat velocity model as **another** neural network, and simultaneously learn it



(d) Speed $v(x, y)$ recovered from PINN simulation.

PINNs for equation discovery



Data Loss: $\mathcal{L}_d(\boldsymbol{\theta}; \mathcal{D}_u) = \frac{1}{N_m}\|\mathbf{u}^\theta - \mathbf{u}^m\|_2^2$ (measurement)

$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\Lambda}; \mathcal{D}_u, \mathcal{D}_c) = \mathcal{L}_d(\boldsymbol{\theta}; \mathcal{D}_u) + \alpha\mathcal{L}_p(\boldsymbol{\theta}, \boldsymbol{\Lambda}; \mathcal{D}_c) + \beta\|\boldsymbol{\Lambda}\|_0$ (total loss; data loss; physics loss; regularization)

Residual Loss: $\mathcal{L}_p(\boldsymbol{\theta}, \boldsymbol{\Lambda}; \mathcal{D}_c) = \frac{1}{N_c}\|\dot{\mathbf{U}}(\boldsymbol{\theta}) - \boldsymbol{\Phi}(\boldsymbol{\theta})\boldsymbol{\Lambda}\|$ (collocation points)

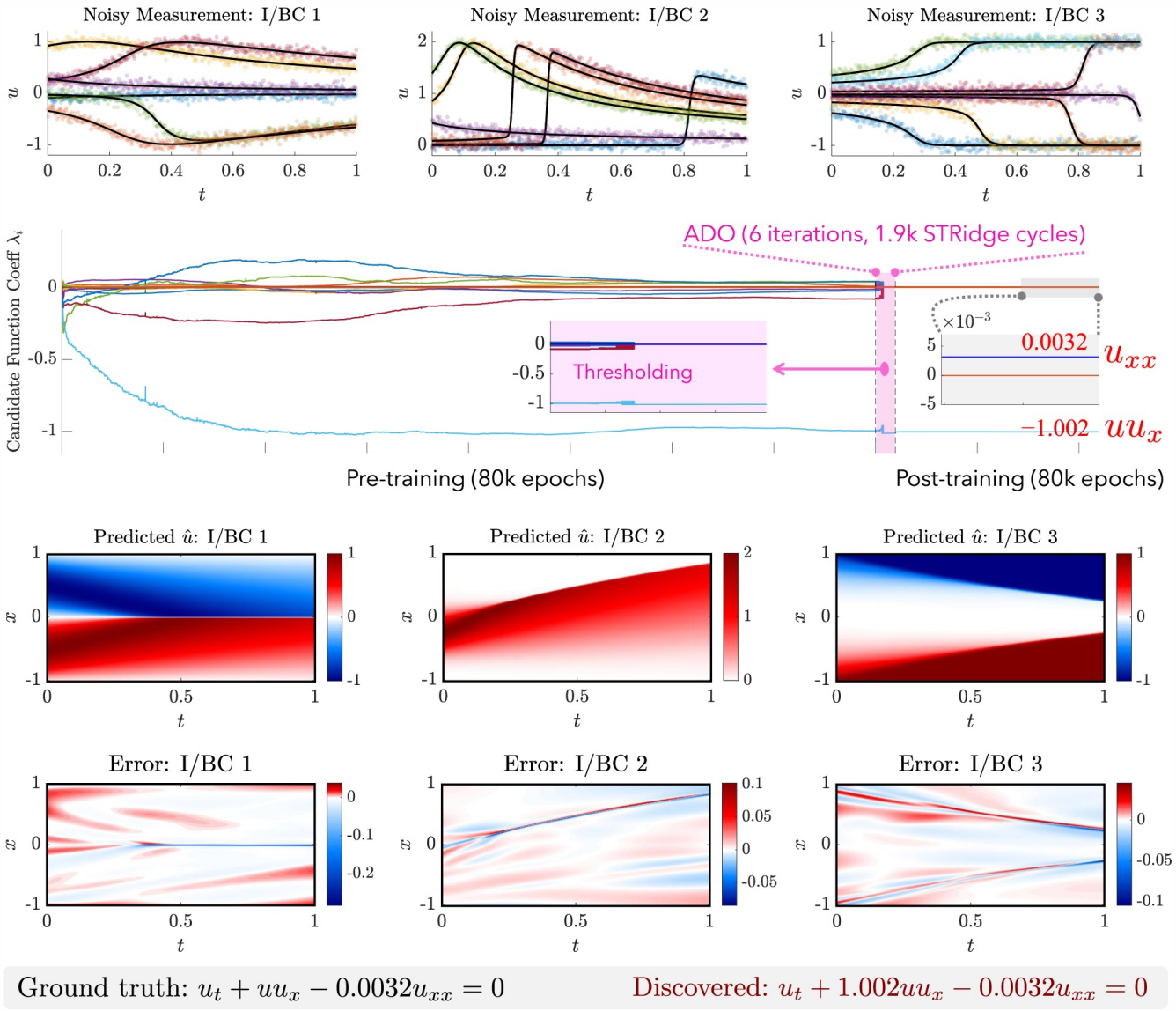
Solution by **ADO**: $\hat{\boldsymbol{\Lambda}}_{k+1} := \arg\min_{\boldsymbol{\Lambda}} \left[\|\dot{\mathbf{U}}(\hat{\boldsymbol{\theta}}_k) - \boldsymbol{\Phi}(\hat{\boldsymbol{\theta}}_k)\boldsymbol{\Lambda}\|_2^2 + \beta\|\boldsymbol{\Lambda}\|_0\right]$ by STRidge

$\hat{\boldsymbol{\theta}}_{k+1} := \arg\min_{\boldsymbol{\theta}} \left[\mathcal{L}_d(\boldsymbol{\theta}; \mathcal{D}_u) + \alpha\mathcal{L}_p(\boldsymbol{\theta}, \hat{\boldsymbol{\Lambda}}_{k+1}; \mathcal{D}_c)\right]$ by DNN training

- Trains by alternating between updating Λ and θ

Chen et al, Physics-informed learning of governing equations from scarce data, Nature communications (2021)

PINNs for equation discovery



- PINN “discovering” Burgers’ equation
- By combining datasets sampled under three different I/BCs with 10% noise

Chen et al, Physics-informed learning of governing equations from scarce data, Nature communications (2021)

Code along – Training a PINN in PyTorch

Follow along here:

github.com/**benmoseley/scalable-pinns-workshop**