Deep Learning in Scientific Computing

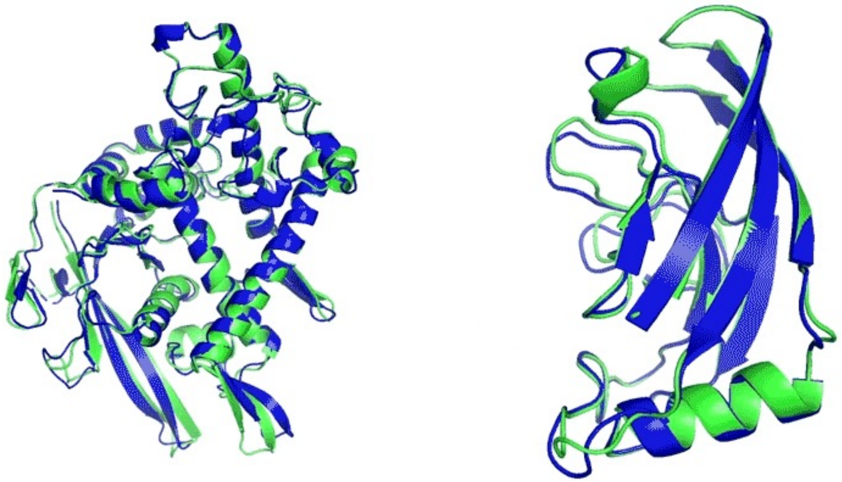
Course Introduction

Spring Semester 2023

Siddhartha Mishra
Ben Moseley

ETH zürich

A scientific revolution?



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

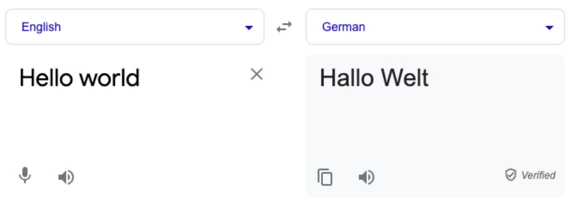
John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain,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

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

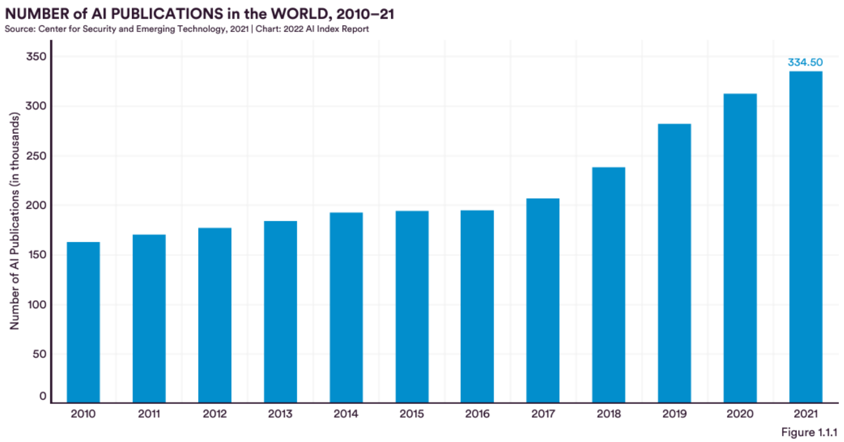
Abstract

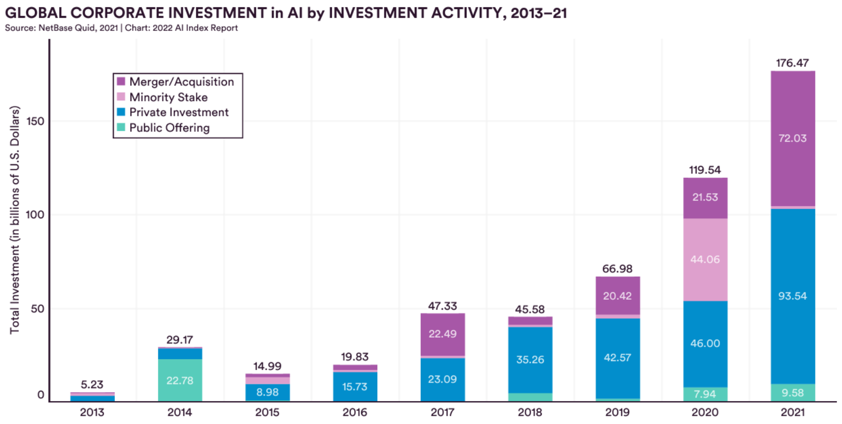
Proteins are essential to life, and understanding their structure can facilitate a mechanistic understanding of their function. Through an enormous experimental effort[1,2,3,4], the structures of around 100,000 unique proteins have been determined[5], but this represents a small fraction of the billions 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

The rise of deep learning



Source: Google Translate

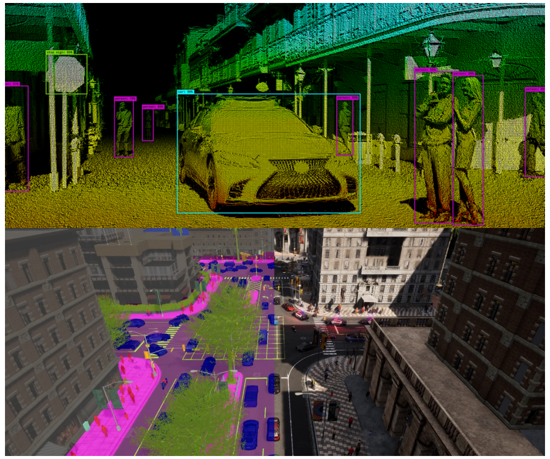




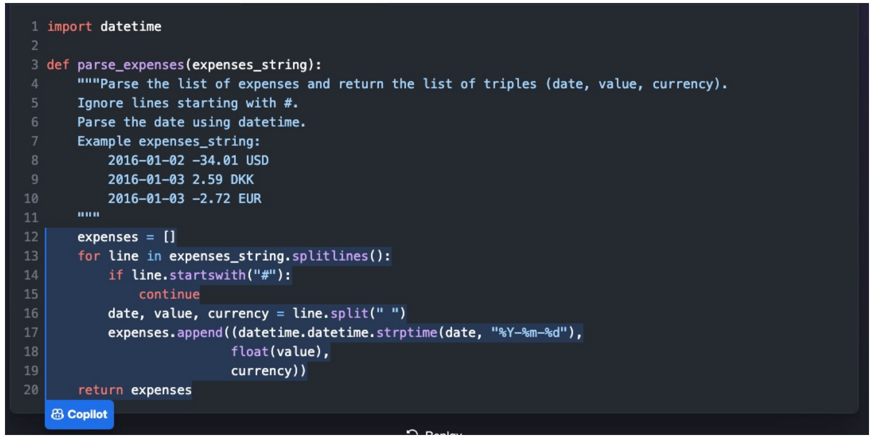
Source: AI Index Report, Stanford University

Prompt:
“a photograph of an astronaut riding a horse”

Source: Stable Diffusion
Rombach et al, High-Resolution Image Synthesis with Latent Diffusion Models, CVPR (2022)

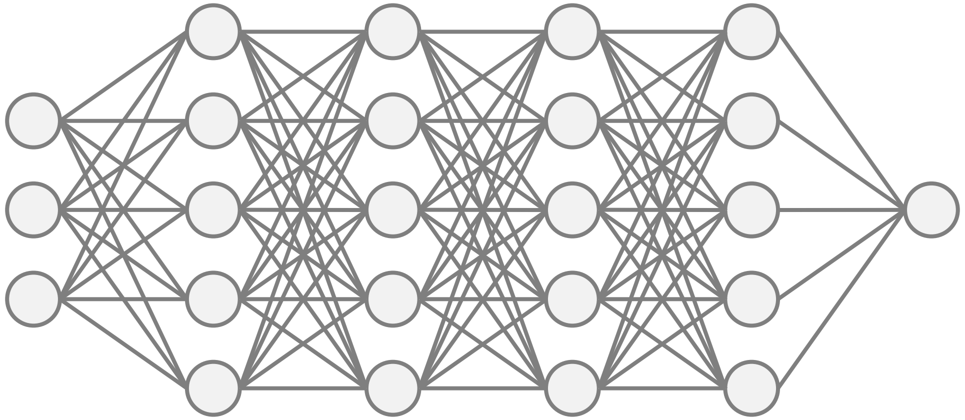


Source: Machine Learning for Autonomous Driving Workshop, NeurIPS (2022)



Source: GitHub Copilot

What is deep learning?



Probability(Dog) = 1

Input

$$x$$

Model

$$NN(x, \theta)$$

Output

$$y = NN(x, \theta)$$

For example:

$$y = W_2 \sigma(W_1 x + b_1) + b_2$$

Deng et al, ImageNet: A large-scale hierarchical image database, CVPR (2009)

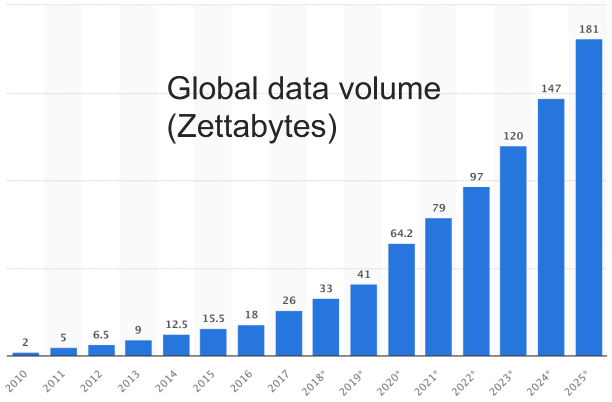
Trained (θ learned) using:
- Many (thousands of) training examples
- An appropriate loss function
- An optimisation algorithm, e.g. (stochastic) gradient descent

Neural networks are simply **flexible functions** fit to data

Why now?

Neural networks date back to the 1950's – so why is deep learning so popular today?

Rapidly increasing amounts of data

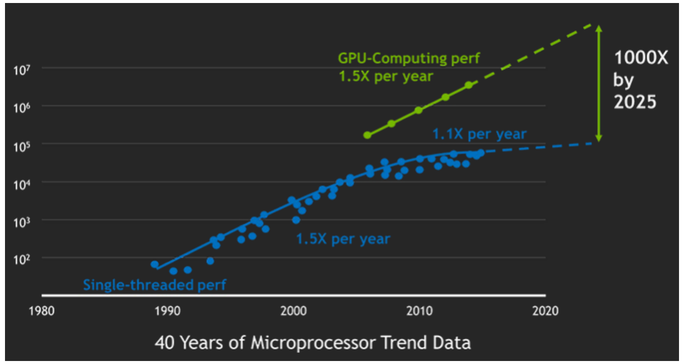


Source: Statista

IMAGENET

WIKIPEDIA The Free Encyclopedia

Hardware improvements



Source: NVIDIA

- Graphical processing units (GPUs)
- Highly optimised for deep learning (massively parallel)

Software improvements

PyTorch

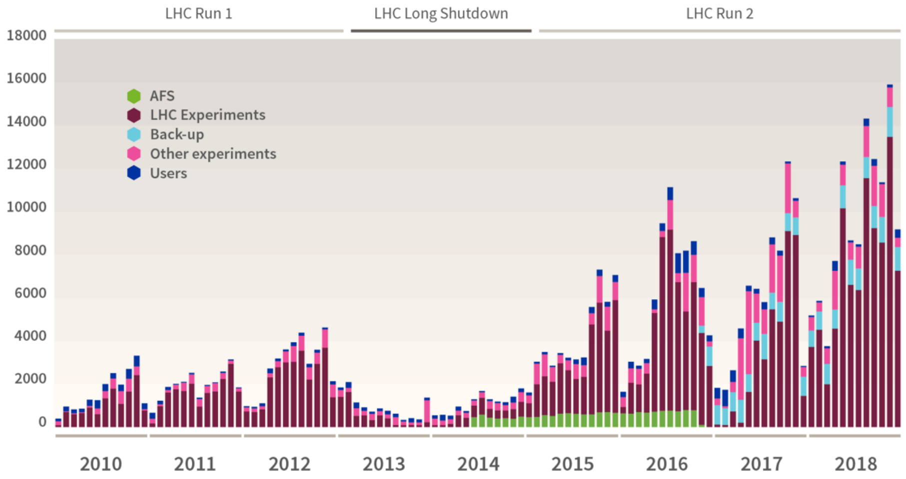
TensorFlow

Keras

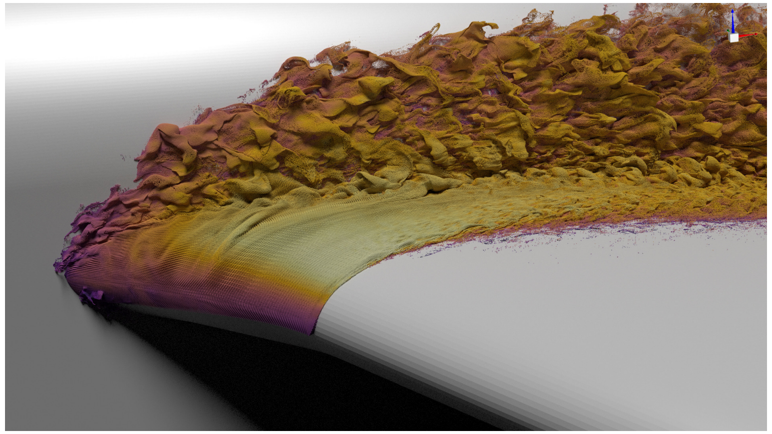
- Mature deep learning frameworks
- Better training algorithms

Grand challenges in science

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



Air flow over a wing: wall-modeled large eddy simulation (~20 million core-hours)
Source: NASA Ames



~5,000 exoplanets discovered to date

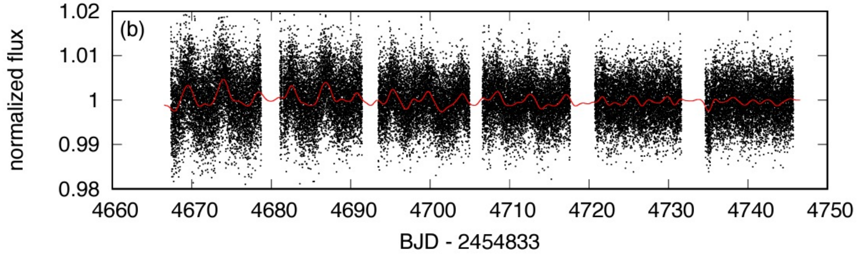
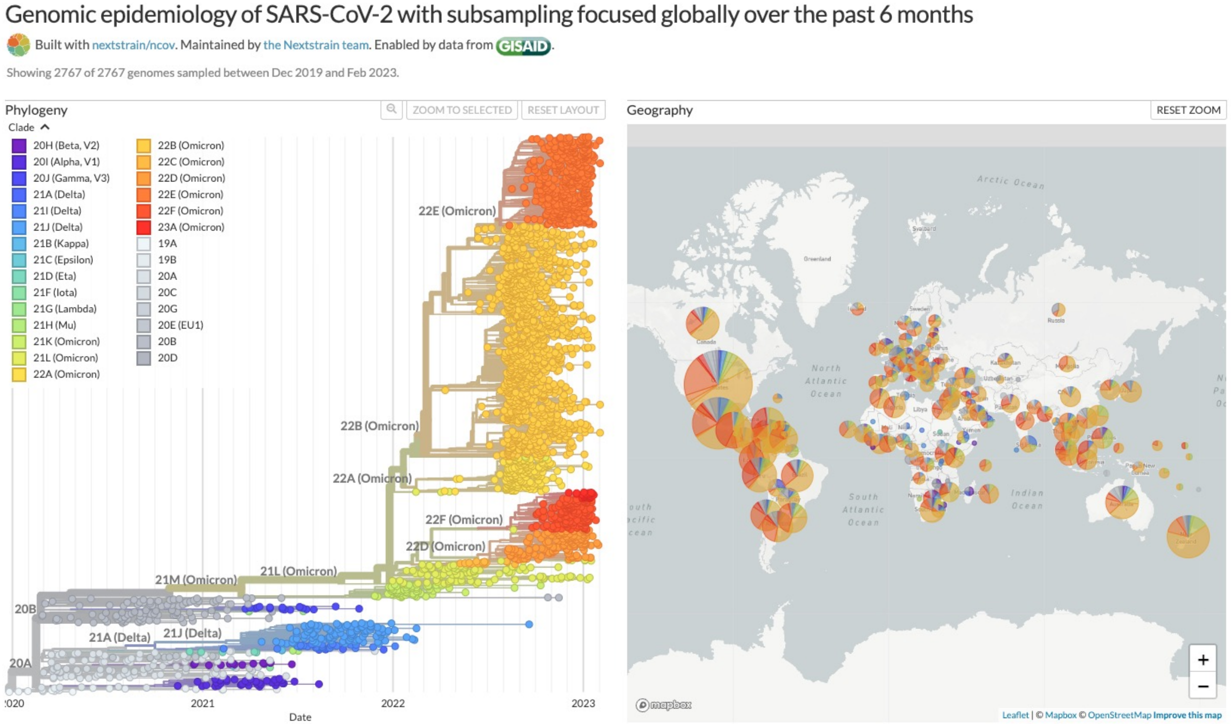


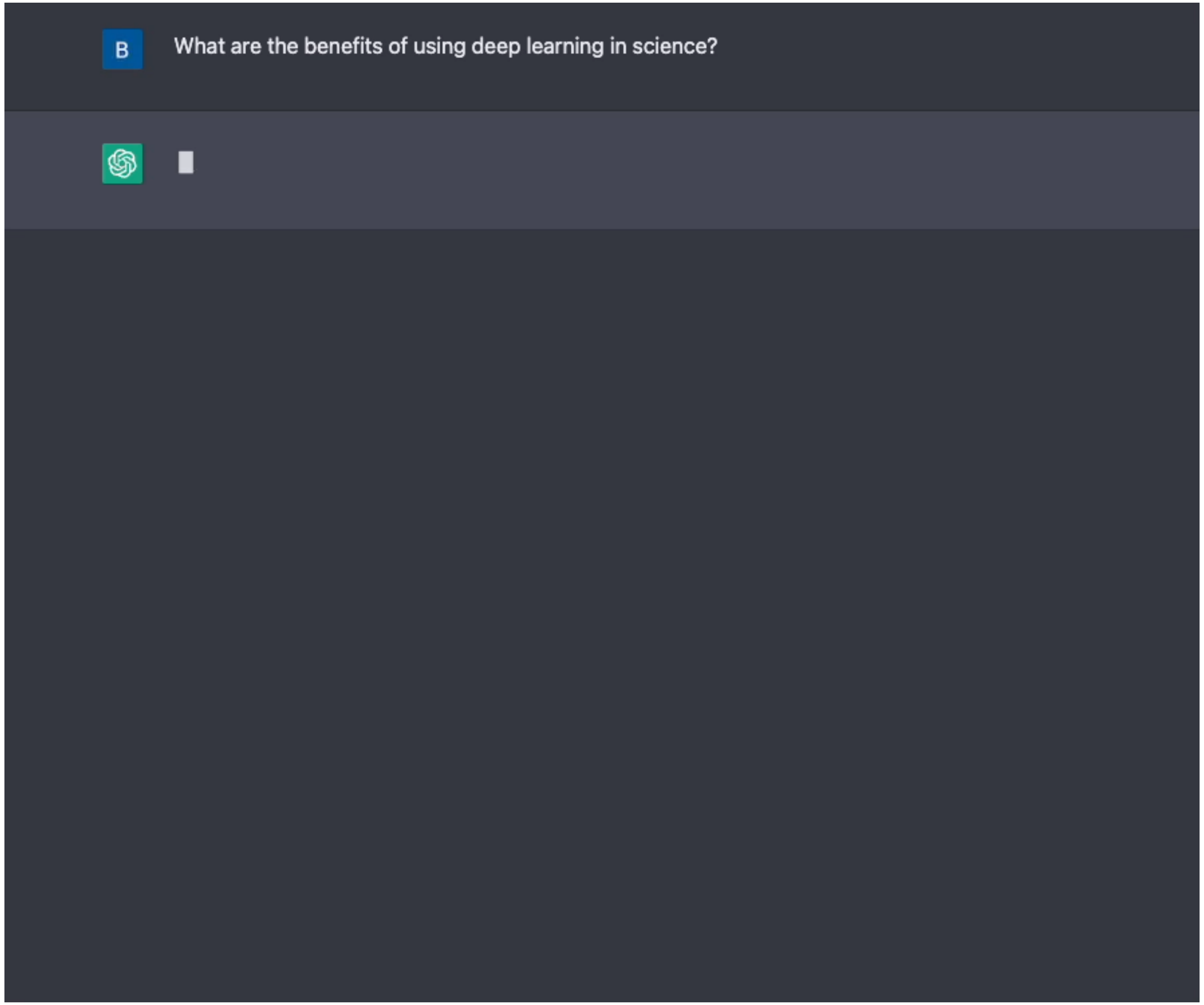
Figure 1. Light curves of K2-415 obtained by K2 (top; K2SFF) and TESS (bottom; PDC-SAP). Those data were taken at long (≈ 29 minutes) and short (2 minutes) cadences for K2 and TESS light curves, respectively. The red solid line in each panel represents the GP regression to the observed light curve (see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

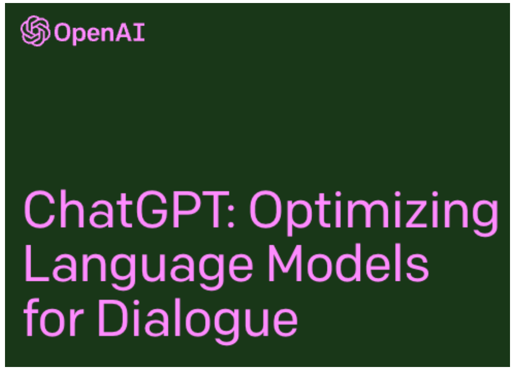
Hirano et al, An Earth-sized Planet around an M5 Dwarf Star at 22 pc, ArXiv (2023)



Source: Nextstrain

Deep learning for science

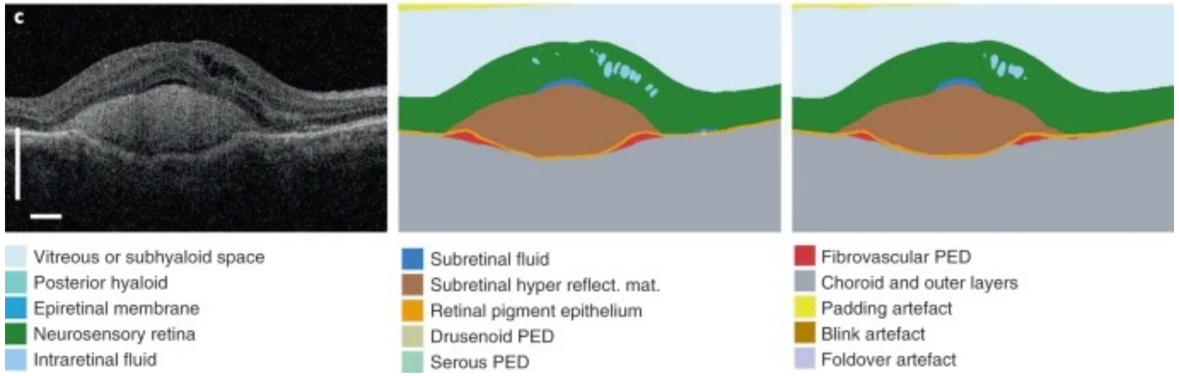




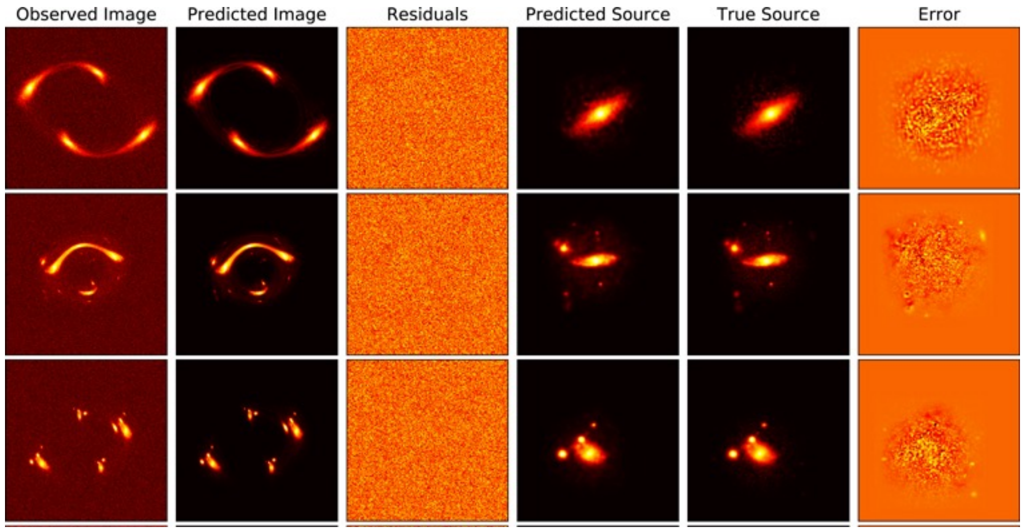
Source: OpenAI

A scientific revolution?

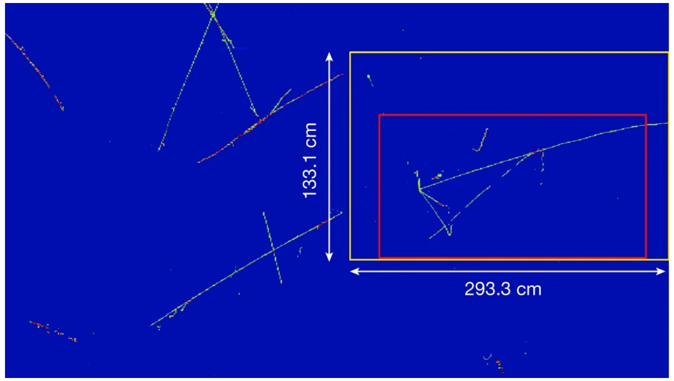
De Fauw, et al. Clinically applicable deep learning for diagnosis and referral in retinal disease, Nature Med (2018)



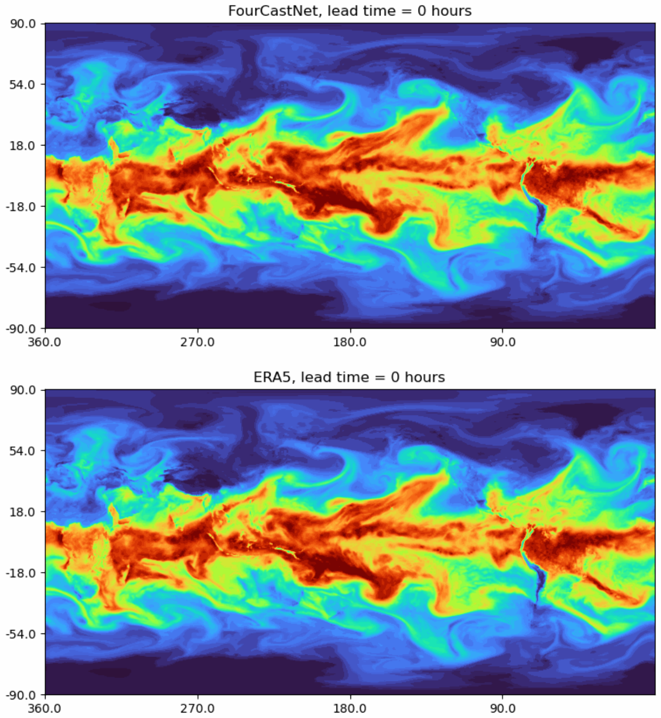
Morningstar et al, Data-driven Reconstruction of Gravitationally Lensed Galaxies Using Recurrent Inference Machines, ApJ (2019)



Acciarri et al, Convolutional neural networks applied to neutrino events in a liquid argon time projection chamber, JINST (2017)



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



Flaws of deep learning

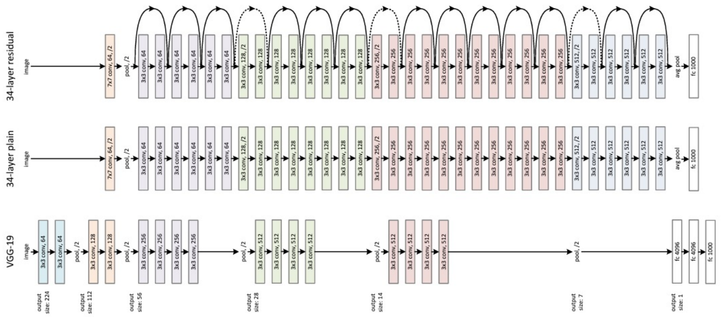
2.1 Model and Architectures

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

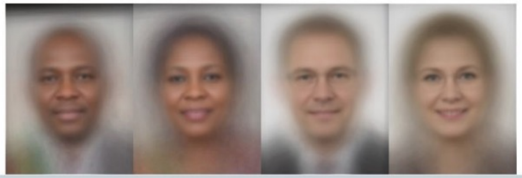
Table 2.2 shows the final mixture of datasets that we used in training. The CommonCrawl data was downloaded from 41 shards of monthly CommonCrawl covering 2016 to 2019, constituting 45TB of compressed plaintext before filtering and 570GB after filtering, roughly equivalent to 400 billion byte-pair-encoded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

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



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

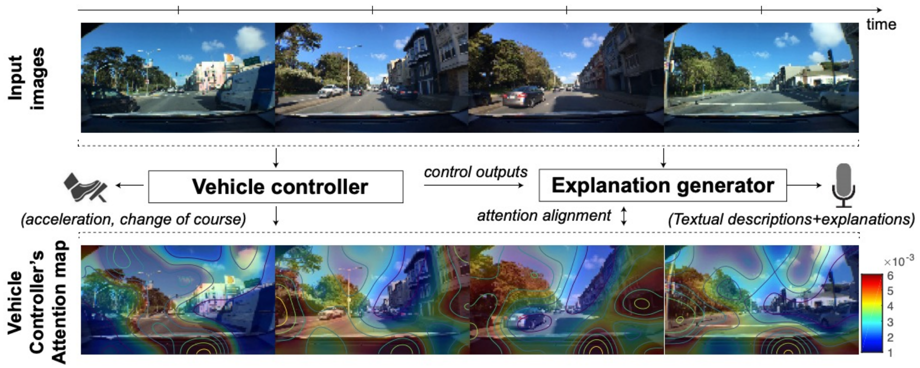
Buolamwini et al, Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification, PMLR (2018)



Gender Classifier	Darker Male	Darker Female	Lighter Male	Lighter Female	Largest Gap
Microsoft	94.0%	79.2%	100%	98.3%	20.8%
FACE++	99.3%	65.5%	99.2%	94.0%	33.8%
IBM	88.0%	65.3%	99.7%	92.9%	34.4%

Figure 1. Intersectional Skin Type and Gender Classification Accuracy Disparities.

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



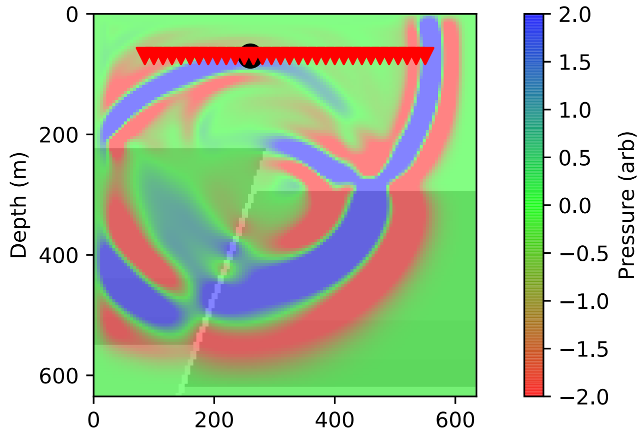
Example of textual descriptions + explanations:
Ours: *"The car is driving forward + because there are no other cars in its lane"*
Human annotator: *"The car heads down the street + because the street is clear."*

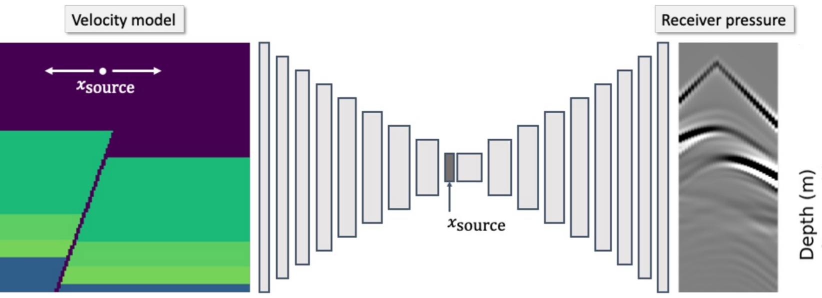
The challenge of generalisation

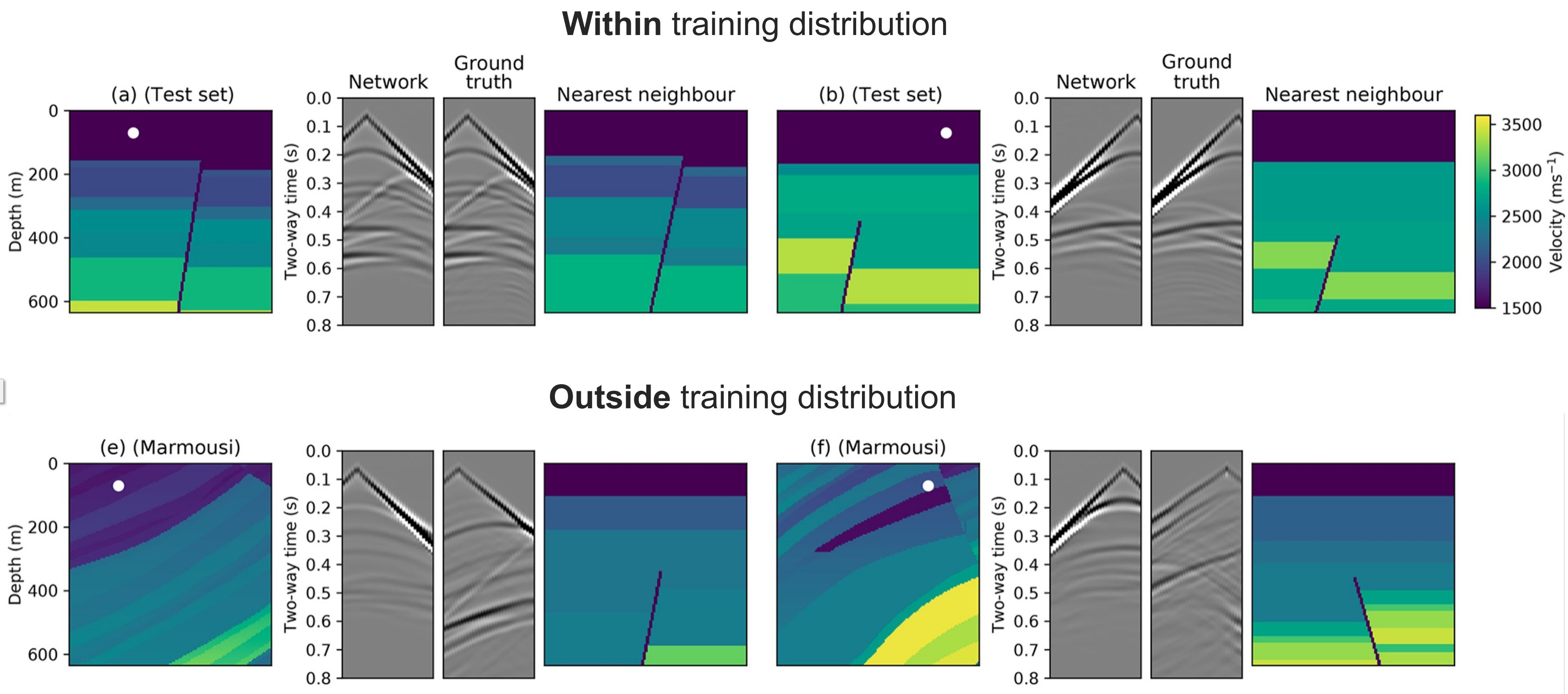
Labrador retriever 52%
Chesapeake Bay retriever 7%
golden retriever 5%
Canis dingo 4%
bloodhound, sleuthhound 3%

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

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

Scientific machine learning (SciML)

Major problem

Despite big breakthroughs in science + AI

Naively using deep learning for scientific tasks usually leads to:

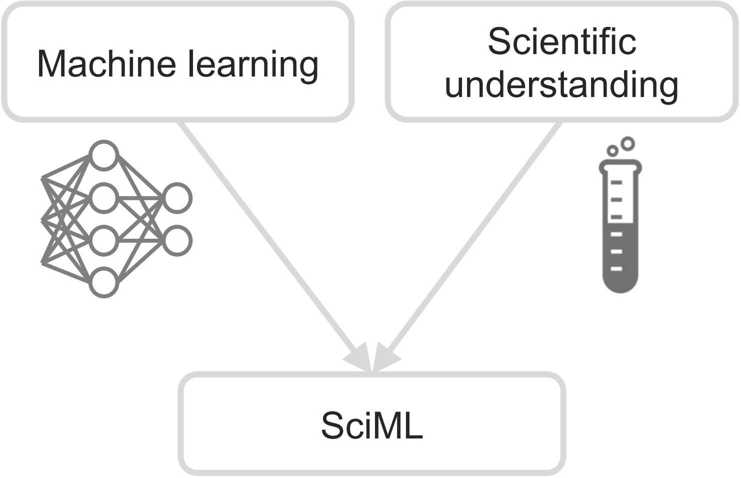
- Lack of interpretability
- Poor generalisation
- Lots of training data required

Do neural networks really "**understand**" the scientific tasks they are being applied to?

Traditional scientific method:

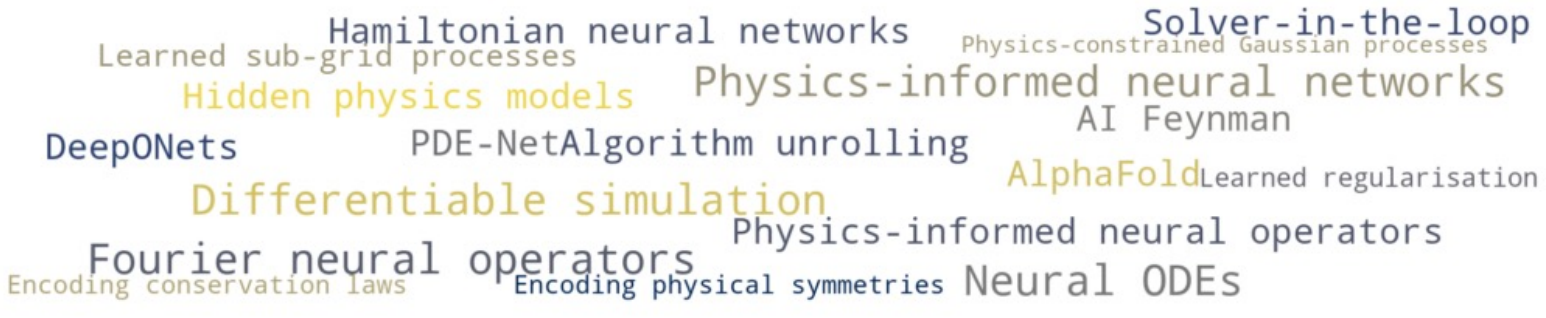
- Revolves around theory and experiment
- a good theory should be explainable and make **novel** predictions

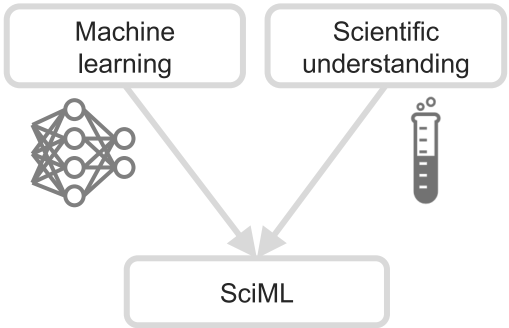
Solution



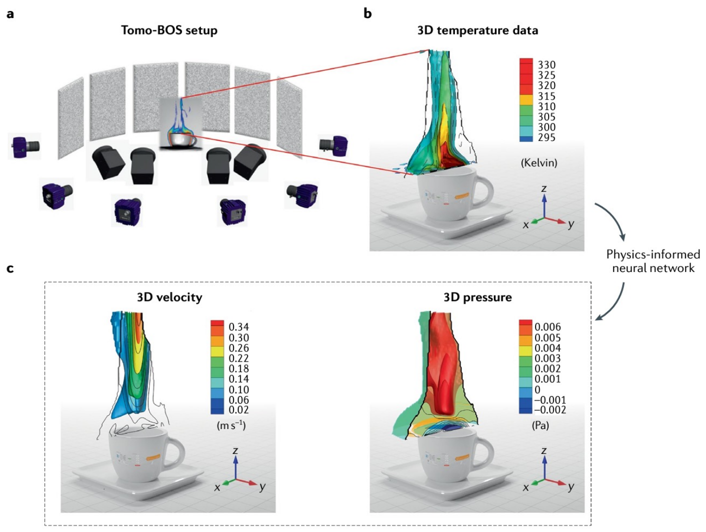
more **powerful, robust, interpretable** models

Scientific machine learning

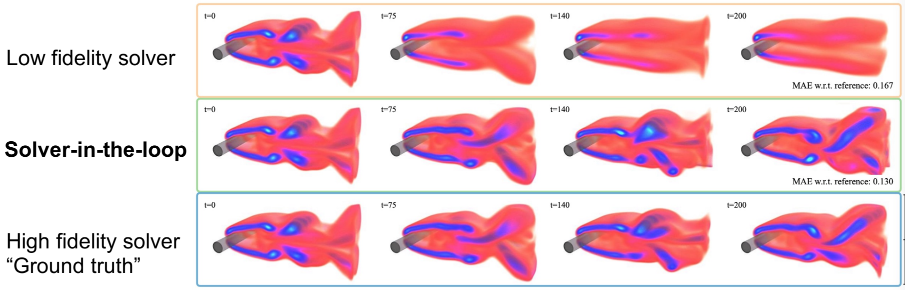




more **powerful, robust, interpretable** models



Cai et al, Flow over an espresso cup: inferring 3-D velocity and pressure fields from tomographic background oriented Schlieren via **physics-informed neural networks**, JFM (2021)



Um et al, **Solver-in-the-loop**: Learning from differentiable physics to interact with iterative PDE-solvers, NeurIPS (2020)

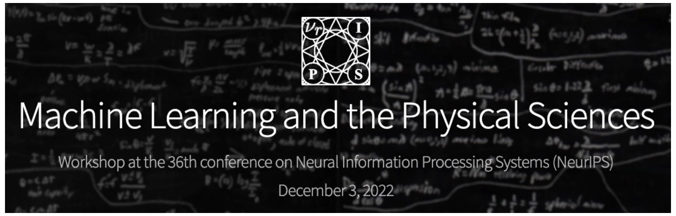
A rapidly growing field

ICLR 2023 Workshop on Physics for Machine Learning

Physics4ML

The Symbiosis of Deep Learning and Differential Equations (DLDE)

NeurIPS 2022 Workshop



ICERM

Scientific Machine Learning

Jan 28 - 30, 2019

AAAI-MLPS 2021

AAAI 2021 Spring Symposium on

Combining Artificial Intelligence and Machine Learning with Physics Sciences

Physics-informed neural networks # citations (reproduced from Cuomo et al, 2022)

National Science Foundation announces MIT-led Institute for Artificial Intelligence and Fundamental Interactions

IAIFI will advance physics knowledge — from the smallest building blocks of nature to the largest structures in the universe — and galvanize AI research innovation.

Laboratory for Nuclear Science
August 26, 2020

The U.S. National Science Foundation (NSF) announced today an investment of more than $100 million to establish five artificial intelligence (AI) institutes, each receiving roughly $20 million over five years. One of these, the NSF AI Institute for Artificial Intelligence and Fundamental Interactions (IAIFI), will be led by MIT's Laboratory for Nuclear Science (LNS) and become the intellectual home of more than 25 physics and AI senior researchers at MIT and Harvard, Northeastern, and Tufts universities.

Course learning objectives

- Aware of advanced **applications** of deep learning in scientific computing
- Familiar with the **design**, **implementation** and **theory** of these algorithms
- Understand the **pros/cons** of using deep learning
- Understand key scientific machine learning **concepts** and themes

Course timeline

Tutorials		Lectures	
Tue 3:15-14:00, HG E5		Fri 12:15-14:00, HG D1.1	
21.02.		24.02.	**Course introduction**
28.02.	Intro to PyTorch	03.03.	Introduction to deep learning I
07.03.	Simple DNNs in PyTorch	10.03.	Introduction to deep learning II
14.03.	Advanced DNNs in PyTorch	17.03.	Physics-informed neural networks – introduction and theory
21.03.	PINN exercises	24.03.	Physics-informed neural networks – applications
28.03.	Implementing PINNs I	31.03.	Physics-informed neural networks – extensions
04.04.	Implementing PINNs II	07.04.	
11.04.		14.04.	
18.04.	Introduction to projects	21.04.	Neural operators – introduction and theory
25.04.	Implementing neural operators I	28.04.	Neural operators – applications
02.05.	Implementing neural operators II	05.05.	Neural operators – extensions
09.05.	Operator learning exercises	12.05.	Graph and sequence models
16.05.	Project discussions	19.05.	Differentiable physics – introduction
23.05.	Implementing autodifferentiation	26.05.	Differentiable physics and neural differential equations
30.05.	Intro to JAX	02.06.	Future trends and overview of CAMLAB

Practical information

- Lectures will be in hybrid mode: Fri 12:15-14:00
- All materials will be available on course Moodle page:
    - Lecture slides + recordings
    - Lecture notes
    - Tutorial exercises
- Performance assessment:
    - Project work during the Semester
    - No exams!
- Tutorial organisers:
    - Roberto Molinaro, Emmanuel De Bezenac, Victor Armegioiu

5 min break

Lecture overview

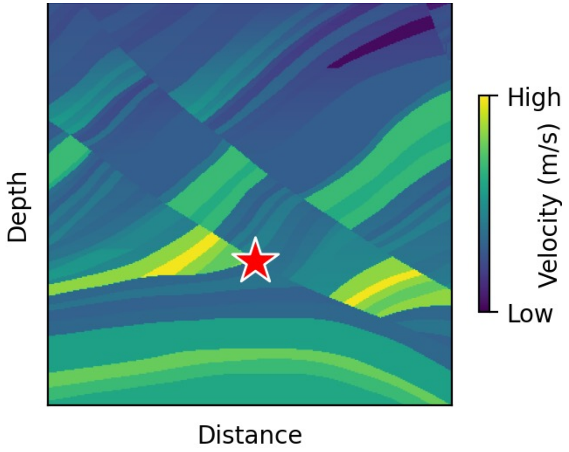
- Course motivation
- Key scientific tasks and how machine learning can help
  - Simulation
  - Inverse, control, and data assimilation problems
  - Equation discovery and anomaly detection
- Overview of scientific machine learning
  - Ways to incorporate scientific principles into deep learning

Key scientific tasks

Key scientific tasks: simulation

Simulation is:

- Crucial for practically all domains of science
- Essential for understanding the behaviour of complex phenomena
- Usually used as a starting point for other tasks (e.g. inverse / control / design problems)



Key scientific tasks: simulation

$$b = F(a)$$

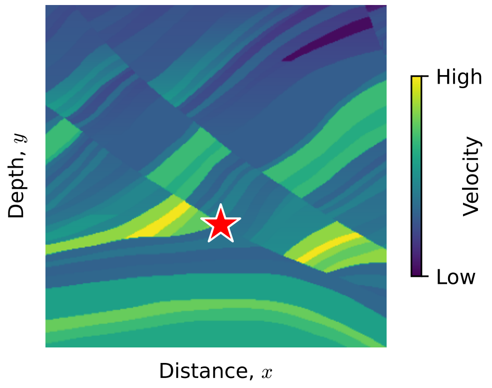
$a =$ set of input conditions

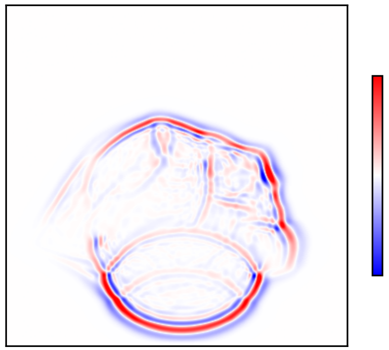
$F =$ physical model of the system

$b =$ resulting properties given F and a

Key scientific tasks: simulation

$$b = F(a)$$





$$\nabla^2 u - \frac{1}{c(x)^2}\frac{\partial^2 u}{\partial t^2} = \delta(x = s, t = 0)$$
$$u(x, t = 0) = 0$$
$$u'(x, t = 0) = 0$$

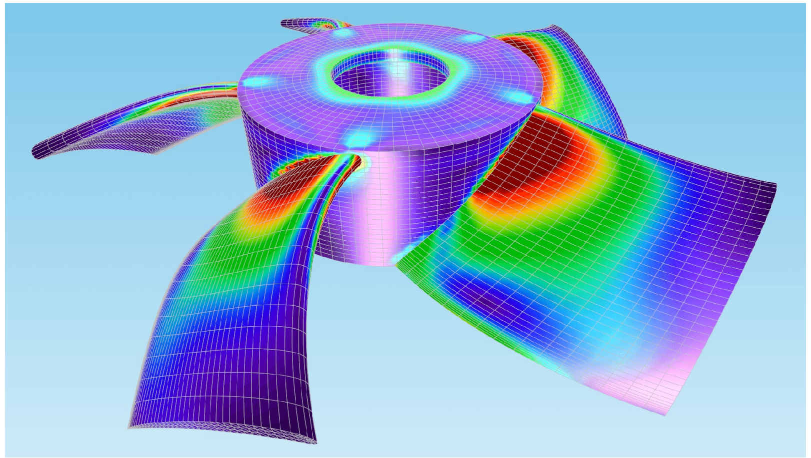
a = velocity model, $c(x, y)$, source location, s

b = wavefield, $u(x, t)$

F = A method for solving the wave equation (e.g. finite difference simulation)

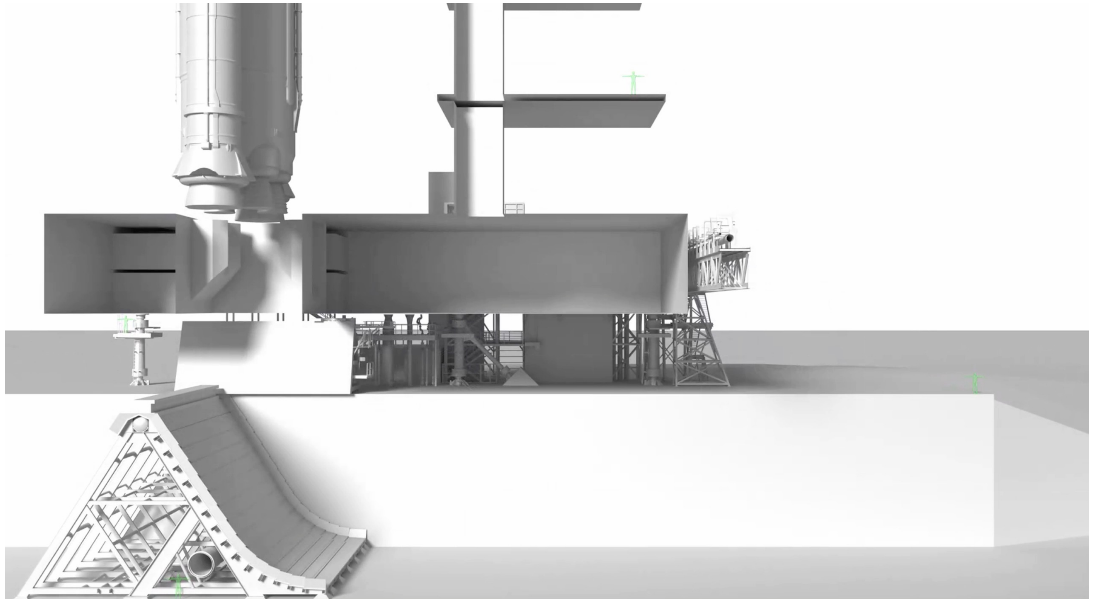
How can we solve simulation problems?

- Usually, analytical solutions do not exist, and we must resort to **numerical** modelling
- Many approaches exist, the most appropriate choice highly depends on the specific problem and scientific domain
- Popular methods (for systems modelled by PDEs) include:
  - finite difference methods, finite element methods, finite volume methods, spectral methods, domain decomposition, mesh-free methods, …



Mesh for finite element method
Source: COMSOL

Challenges of simulation



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

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

- Typically, **computational cost** is the main challenge
- Simulation can require **elaborate** parallel software implementations (especially for multi-scale, multi-physics systems) with 10,000s of lines of code
- Significant **human effort** is often required, e.g. in defining high-quality meshes for finite element simulations

How to Contribute to OpenFOAM

OpenFOAM is a large piece of software (of the order of 1 million lines of code) in a complex area of scientific application. Since its open source release in 2004, it has become the CFD software of choice for many thousands of people from industry, government laboratories, academic institutions, etc., who download OpenFOAM and use it for free. With such a large user base, sometimes working on mission-critical applications, we have a responsibility to maintain OpenFOAM as a robust, efficient, and scalable CFD software package for a wide range of applications.

Source: OpenFOAM

Key scientific tasks: inverse problems

$$b = F(a)$$

$a =$ set of input conditions

$F =$ physical model of the system

$b =$ resulting properties given F and a

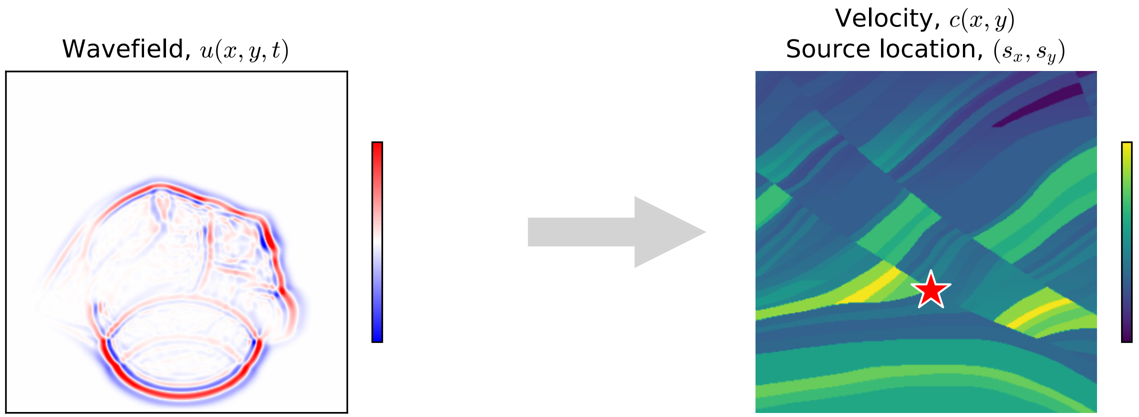
Key scientific tasks: inverse problems

$$b = F(a)$$

Inverse problems are **pervasive** across all domains of science and solving them is essential for many real-world tasks

Example inverse problems:

- Seismic imaging
- Magnetic resonance imaging
- Image denoising
- Estimating infection rates
- Design problems
- …



b = wavefield ($u(x, t)$)

a = velocity model, source location

F = A method for solving the PDE (e.g. finite difference simulation)

How can we solve inverse problems?

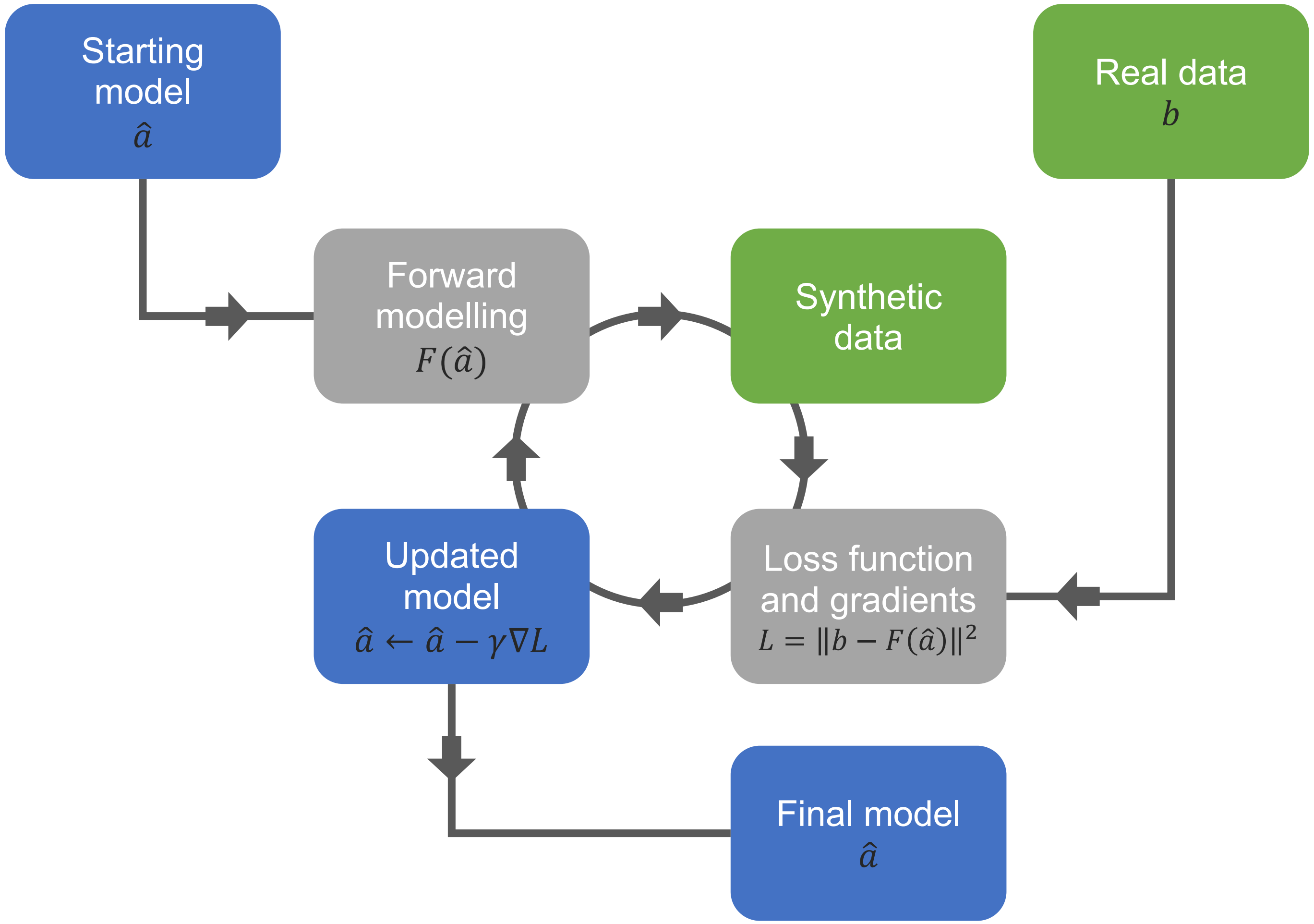
$$b = F(a)$$

- Fundamentally, inverse problems are **search** problem
- It is often useful to frame them as an optimisation problem, for example:

$$\min_{\hat{a}} \ ||b - F(\hat{a})||^2$$

- If F is differentiable, one option is to use gradient-based methods (e.g. **gradient descent**)
- Otherwise, we can use gradient-free methods (e.g. evolutionary algorithms, Bayesian optimisation, brute-force search, …)

Solving inverse problems with gradient descent



$$\min_{\hat{a}} \|b - F(\hat{a})\|^2$$

Loss function is:

$$L(\hat{a}) = \|b - F(\hat{a})\|^2$$

Gradient descent step:

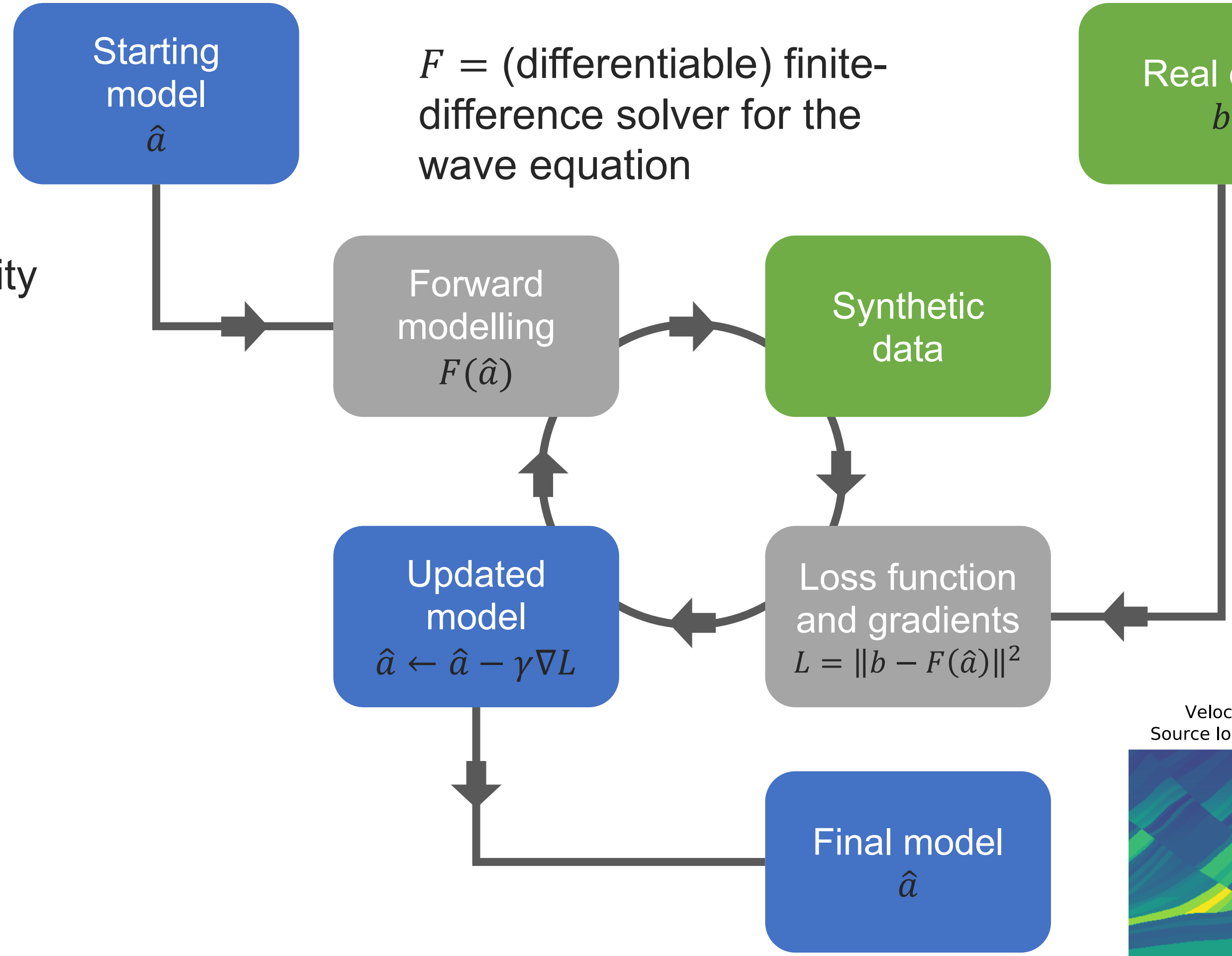
$$\hat{a} \leftarrow \hat{a} - \gamma \nabla L$$

- Requires F to be differentiable

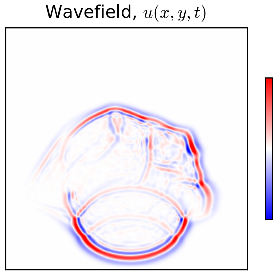
For later: note similarity to training deep neural networks

Solving inverse problems with gradient descent

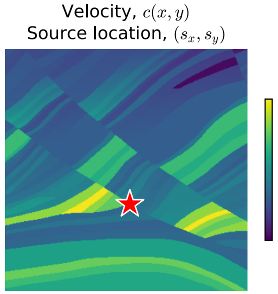
$\hat{a}$ = starting velocity model



F = (differentiable) finite-difference solver for the wave equation

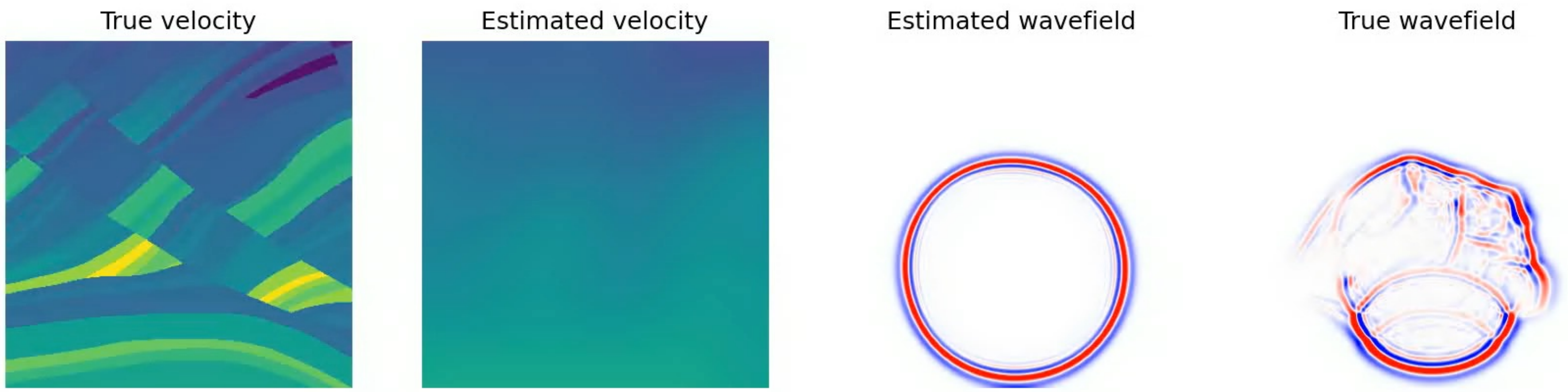


b = wavefield



a = true velocity model

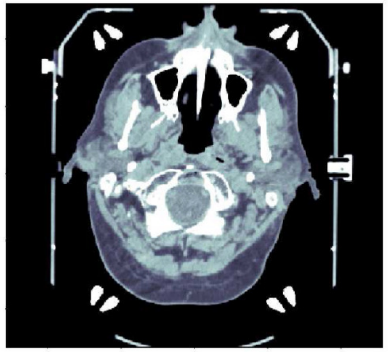
Solving inverse problems with gradient descent



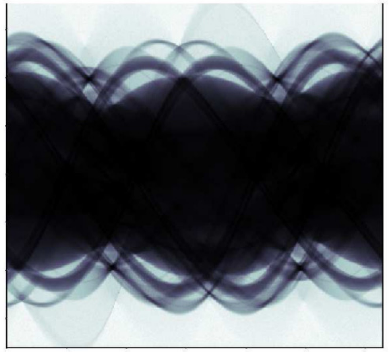
$$\min_{\hat{a}} \ \|b - F(\hat{a})\|^2$$

- In geophysics, this inverse problem is known as **full waveform inversion**

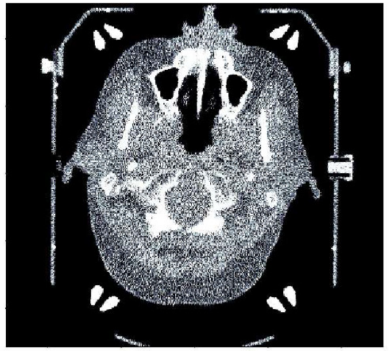
Challenges of inverse problems



Ground truth computed tomography image

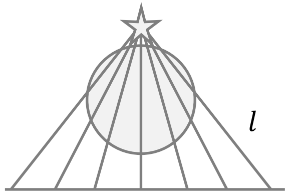


Resulting tomographic data (sinogram)



Result of inverse algorithm (filtered back-projection)

$a(x)$ $\quad$ $F(a)(l) = I_0 \exp(-\int_l a(x)\, dx))$ $\quad$ $\hat{a}(x)$



Adler et al, Solving ill-posed inverse problems using iterative deep neural networks, Inverse Problems (2017)

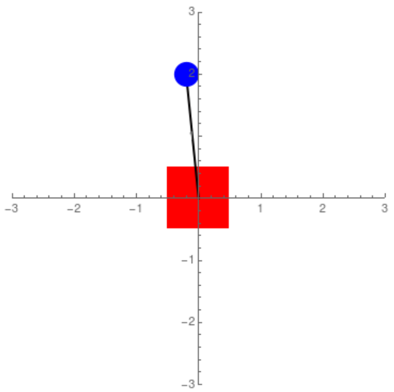
Typically, inverse problems are **incredibly challenging** to solve because:

- They are usually **ill-posed** (not enough information for a unique solution)
- Observed real-world data is usually **noisy** and **sparse**
- Often require forward modelling to be carried out thousands of times – making them extremely **computationally demanding**

Key scientific tasks: control and data assimilation

- Both are related to inverse problems

Control



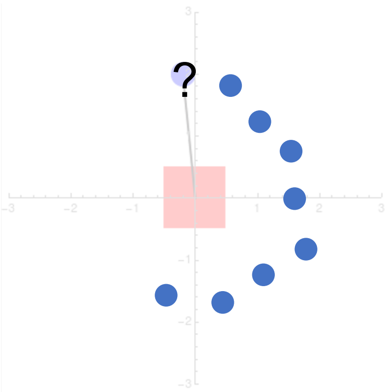
Source: Neil Gershenfeld / MIT

E.g.: Inverted pendulum

$$(M+m)\ddot{x} - ml\ddot{\theta}\cos\theta + ml\dot{\theta}^2\sin\theta = f$$
$$l\ddot{\theta} - g\sin\theta = \ddot{x}\cos\theta$$

θ = angle of pendulum
x = position of cart
f = external force on cart
l = length of rod
m, M = mass of pendulum, cart

Data assimilation



$$b = F(a)$$

$a = f(t)$, force applied to cart
b = pendulum stays balanced (i.e. $\theta(t) \to 0$)
F = method for solving equations of motion

$a = x(t=0), \theta(t=0)$,
b = Noisy measurements of $x(t_i), \theta(t_i)$
F = Method for solving equations of motion and noise model

Key scientific tasks: discovering unknown physics

$$b = F(a)$$

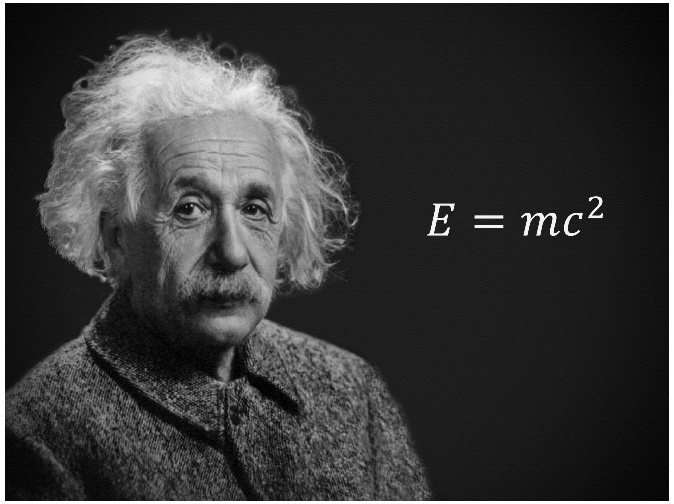
a = set of input conditions

F = physical model of the system

b = resulting properties given F and a

Challenges of discovering unknown physics

$$b = F(a)$$



- Historically, F (= laws of physics) has been found through remarkable human **intuition**
- Constant interplay between theory and experiment
- From a computational standpoint, discovering physics is like solving an inverse problem (trying to fit a model to observed data)
- But the model should be **explainable**, **generalisable** and make **novel** predictions

How can deep learning help?

How can deep learning help?

B What are the benefits of using deep learning in science?

Deep learning, a subset of machine learning, has revolutionized various scientific fields by offering numerous benefits. Some of the benefits of using deep learning in science include:

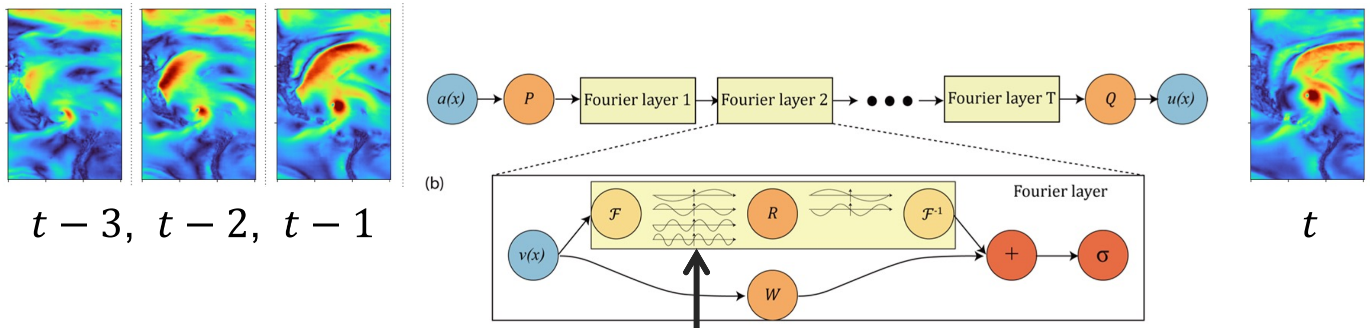
1. Improved accuracy: Deep learning algorithms can learn patterns in large amounts of data and make highly accurate predictions. This is particularly useful in fields like medical imaging where human-level accuracy is required to detect diseases and anomalies.
2. Automation: Deep learning algorithms can automate tasks that were previously performed manually, freeing up scientists' time to focus on other tasks. For example, in astronomy, deep learning algorithms can be used to automatically identify objects in images, saving researchers hours of manual analysis.
3. Enhanced understanding: By revealing patterns and relationships in data, deep learning algorithms can help scientists gain new insights into their fields. This is particularly useful in areas such as genomics where large amounts of data need to be analyzed to uncover important biological insights.
4. New applications: Deep learning can be applied to new areas that were previously difficult to tackle using traditional machine learning methods. For example, it has been used in chemistry to predict the properties of new chemical compounds and accelerate drug discovery.
5. Increased efficiency: By reducing the time required to analyze data and perform experiments, deep learning can increase the overall efficiency of scientific research. This can lead to new discoveries being made more quickly and cost-effectively.

Overall, deep learning has the potential to significantly enhance the speed and accuracy of scientific research, enabling scientists to make new discoveries and progress more quickly.

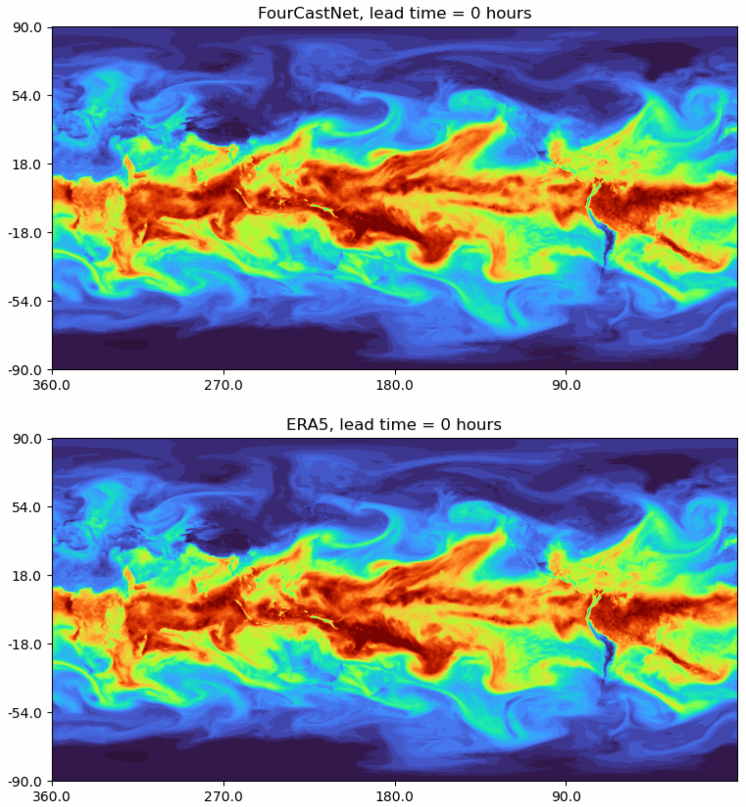
- Accuracy
- Efficiency
- Automation
- Discovery

Example - simulation

Fourier Neural Operator



Spectral convolution:
x = FFT(x)
x = x[:modes]
x = x * θ
x = IFFT(x)



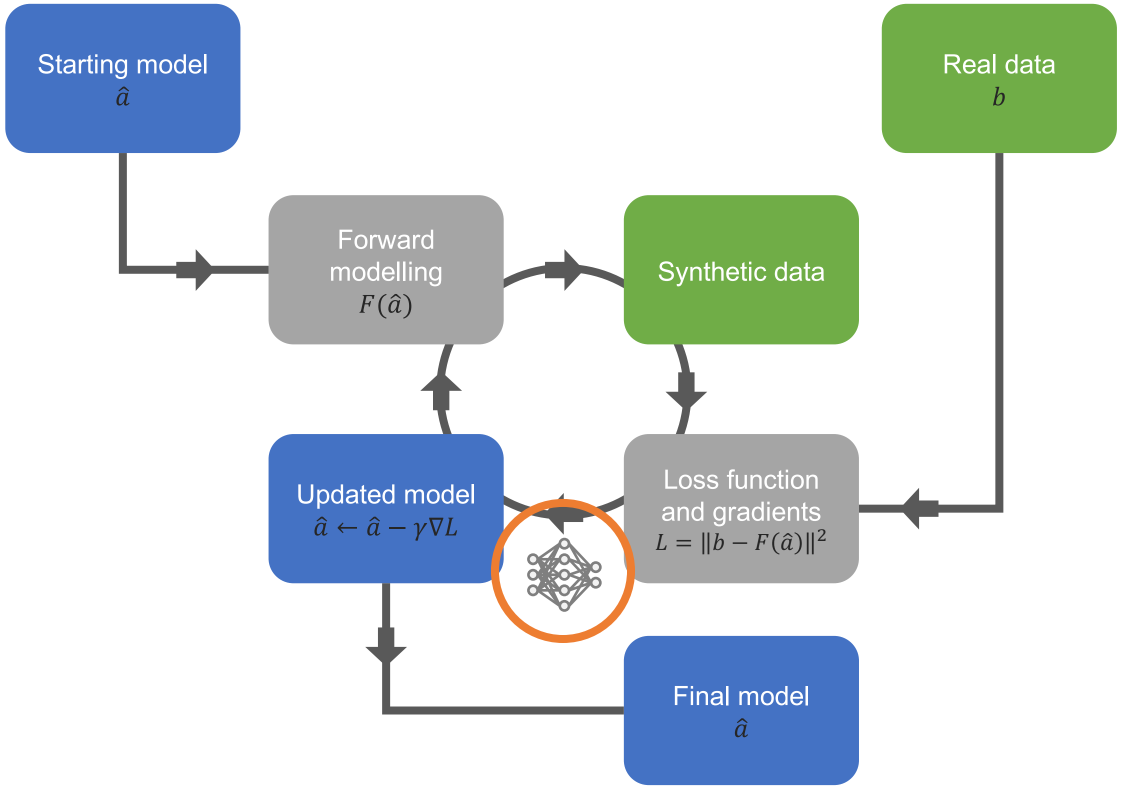
- Idea: use deep neural network to predict next timestep of multiple atmospheric variables, given previous timesteps
- Use a Fourier neural operator architecture
- **4-5 orders of magnitude** faster than numerical simulation

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

Li et al, Fourier Neural Operator for Parametric Partial Differential Equations, ICLR (2021)

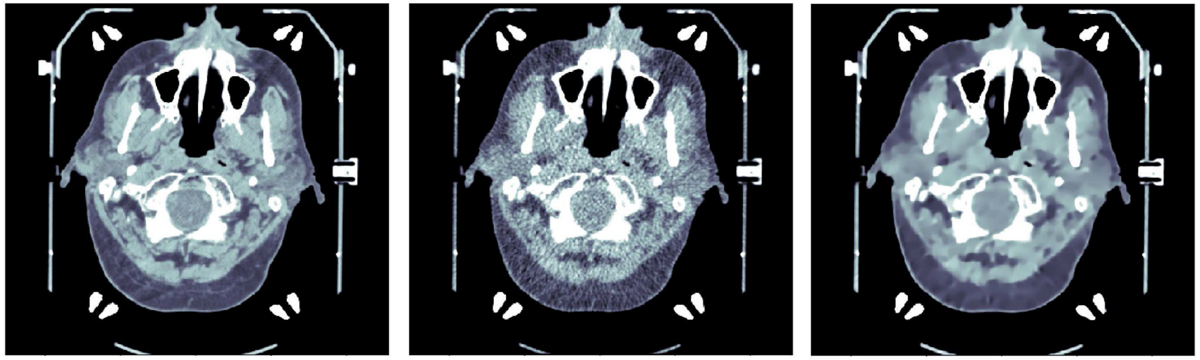
Example - inverse problems

Learned gradient descent



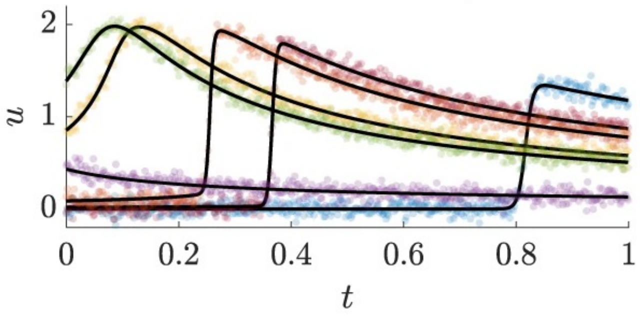
Adler et al, Solving ill-posed inverse problems using iterative deep neural networks, Inverse Problems (2017)

- Idea: use a neural network to **update** gradients before descent step
- May help escape local minima, converge quicker and **regularise** the problem
- Inputs to network: $\nabla L, \hat{a}, b$
- Outputs: updated gradient, $g = NN(\nabla L, \hat{a}, b; \theta)$
- New descent step: $\hat{a} \leftarrow \hat{a} - \gamma g$
- Network is trained by using many example inverse problems, and differentiating through gradient descent algorithm **end-to-end**



Ground truth — Traditional inversion — Learned gradient descent

Example - equation discovery



Ground truth: $u_t + uu_x - 0.0032u_{xx} = 0$

Discovered: $u_t + 1.002uu_x - 0.0032u_{xx} = 0$

$$u_t = \Lambda\phi$$

$$\Lambda = (\lambda_1 \;\; \lambda_2 \;\; \lambda_3 \;\; \lambda_4 \;\; \lambda_5 \;\; \dots) \qquad \phi = \begin{pmatrix} \hat{u}_x \\ \hat{u}_{xx} \\ \hat{u}_t \\ \hat{u}_{tt} \\ \hat{u}_{xt} \\ \dots \end{pmatrix}$$

- Idea: fit a neural network to observed data, then regress over a **library** of gradients to “discover” underlying equations
- Input: $\tilde{u}(x,t) =$ (noisy) observational data
- Fit a neural network to data, $\hat{u} = NN(x,t;\theta) \approx \tilde{u}$, using supervised learning
- Compute various gradients of network, for example $\hat{u}_x, \hat{u}_{xx}, \hat{u}_t, \hat{u}_{tt}, \hat{u}_{xt}$ ..., at many random (x,t) locations
- Carry out a (sparse) **linear regression** over combinations of these gradients to “discover” underlying equation

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

The state-of-the-art: scientific machine learning

Scientific machine learning (SciML)

Major problem

Despite big breakthroughs in science + AI

Naively using deep learning for scientific tasks usually leads to:

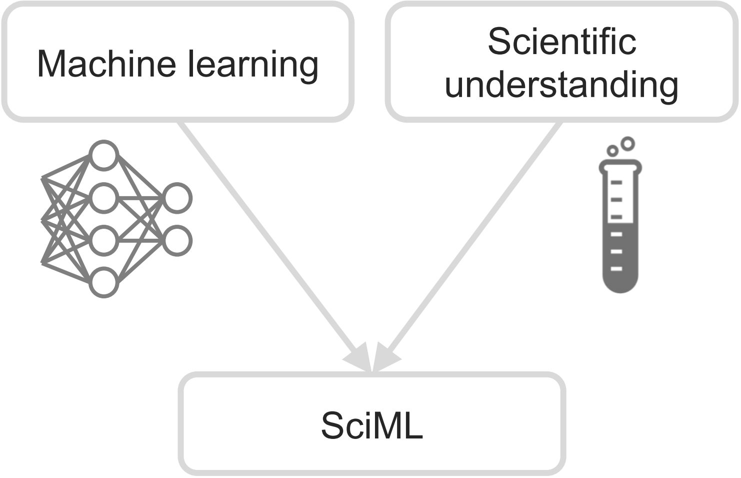
- Lack of interpretability
- Poor generalisation
- Lots of training data required

Do neural networks really "**understand**" the scientific tasks they are being applied to?

Traditional scientific method:

- Revolves around theory and experiment
- a good theory should be explainable and make **novel** predictions

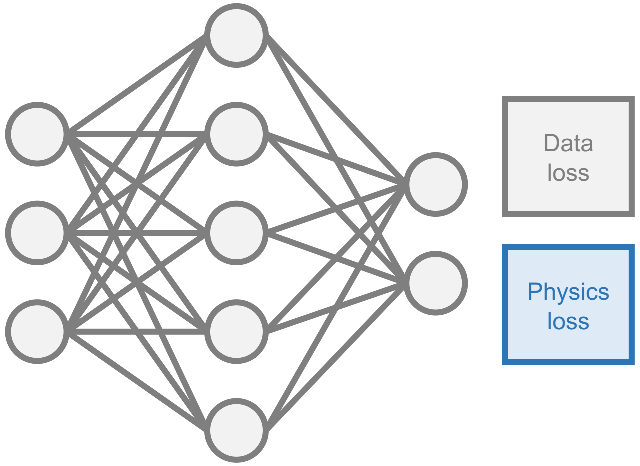
Solution



more **powerful, robust, interpretable** models

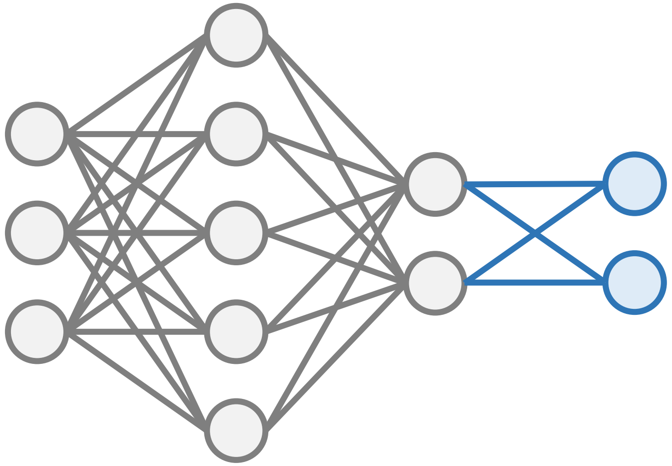
Ways to incorporate scientific principles into machine learning

Loss function



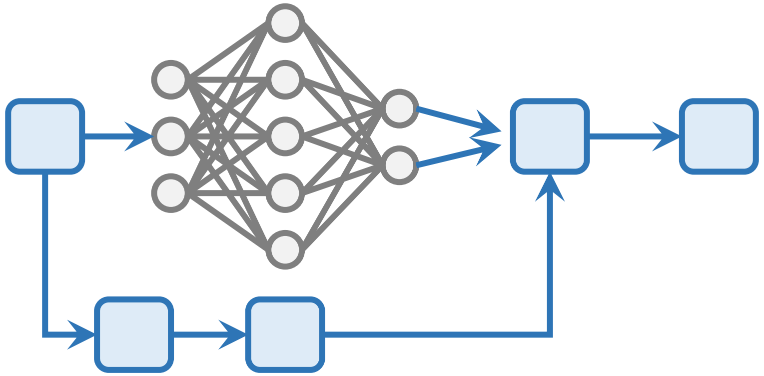
Example:
Physics-informed neural networks
(add governing equations to loss function)

Architecture



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

Hybrid approaches



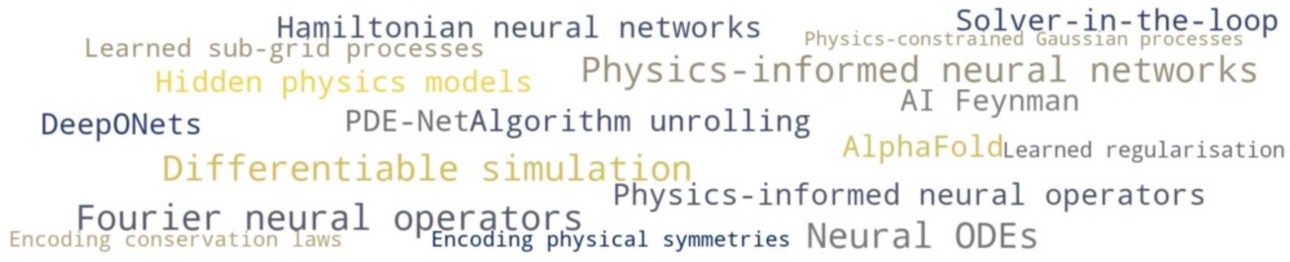
Example:
Neural differential equations
(incorporating neural networks into traditional PDE solvers)

A plethora of SciML techniques

Scientific task

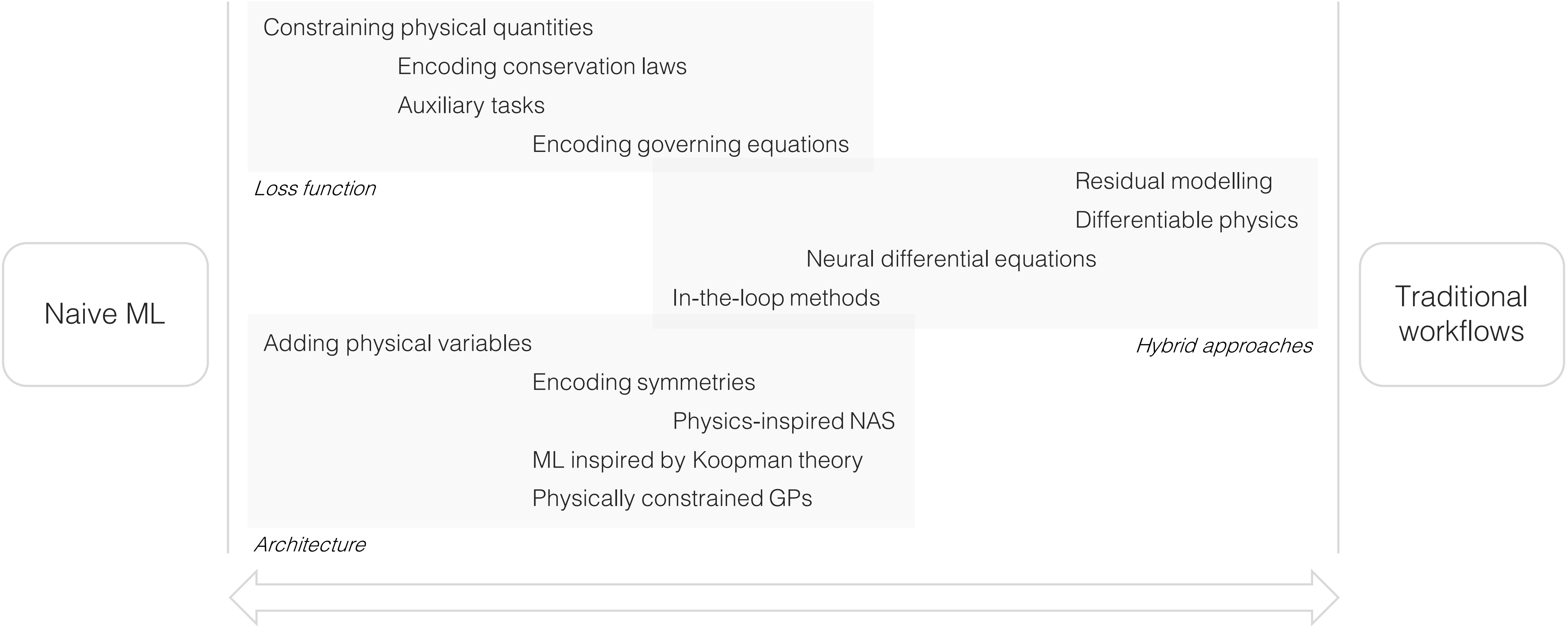
SciML technique

	Forward simulation	Inversion	Equation discovery
Architecture			
Adding physical variables	Daw et al.		
Encoding symmetries	Ling et al., Wang et al., Anderson et al., Schütt et al.		Udrescu et al.
Physics-inspired NAS	Ba et al., Panju and Ghodsi		
ML inspired by Koopman theory	Geneva and Zabaras, Lusch et al.		
Physically constrained GPs		Raissi et al., Raissi and Karniadakis	
Other approaches	Jumper et al., Mohan et al.		
Loss function			
Constraining physical quantities	Karpatne et al., Zhang et al., Benjamin Erichson et al., Xie et al., Brehmer et al.		
Encoding conservation laws	Beucler et al., Zeng et al.		Greydanus et al., Toth et al., Cranmer et al.
Auxiliary tasks	de Oliveira et al.		
Encoding governing equations	Raissi et al., Jin et al., Kharazmi et al., Yang et al., Wang et al., Wang et al., Li et al., Zhu et al., Geneva and Zabaras, Gao et al.	Jin et al., Chen et al., Yang et al.	Chen et al., Champion et al.
Hybrid approaches			
Residual modelling	Pawar et al.	Jiang et al.	
Differentiable physics		Ren et al., Minkov et al., Würfl et al., Zhang et al.	
Neural differential equations			Chen et al., Rackauckas et al., Long et al.
In-the-loop methods	Um et al., Rasp et al.	Adler and Öktem, Morningstar et al., Hammernik et al., Li et al., Lunz et al., Bora et al., Mosser et al.	



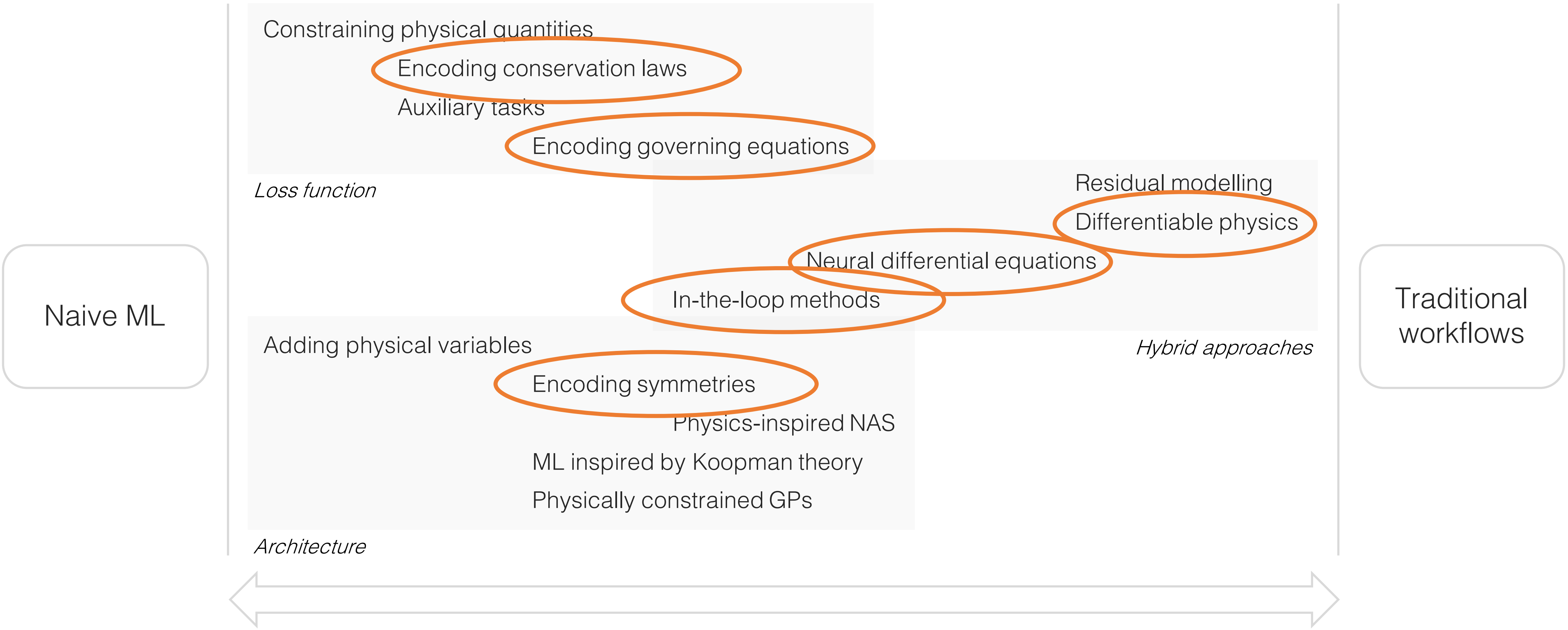
Source: B Moseley, Physics-informed machine learning: from concepts to real-world applications, PhD thesis, 2022

A plethora of SciML techniques



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

A plethora of SciML techniques



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

Lecture summary

- **Deep learning** has grown exponentially in the last decade
- It has huge potential for **revolutionising** science
- The latest research is in the field of scientific machine learning, which **blends** together scientific knowledge and machine learning
- A **plethora** of approaches exist which use deep learning for scientific computing