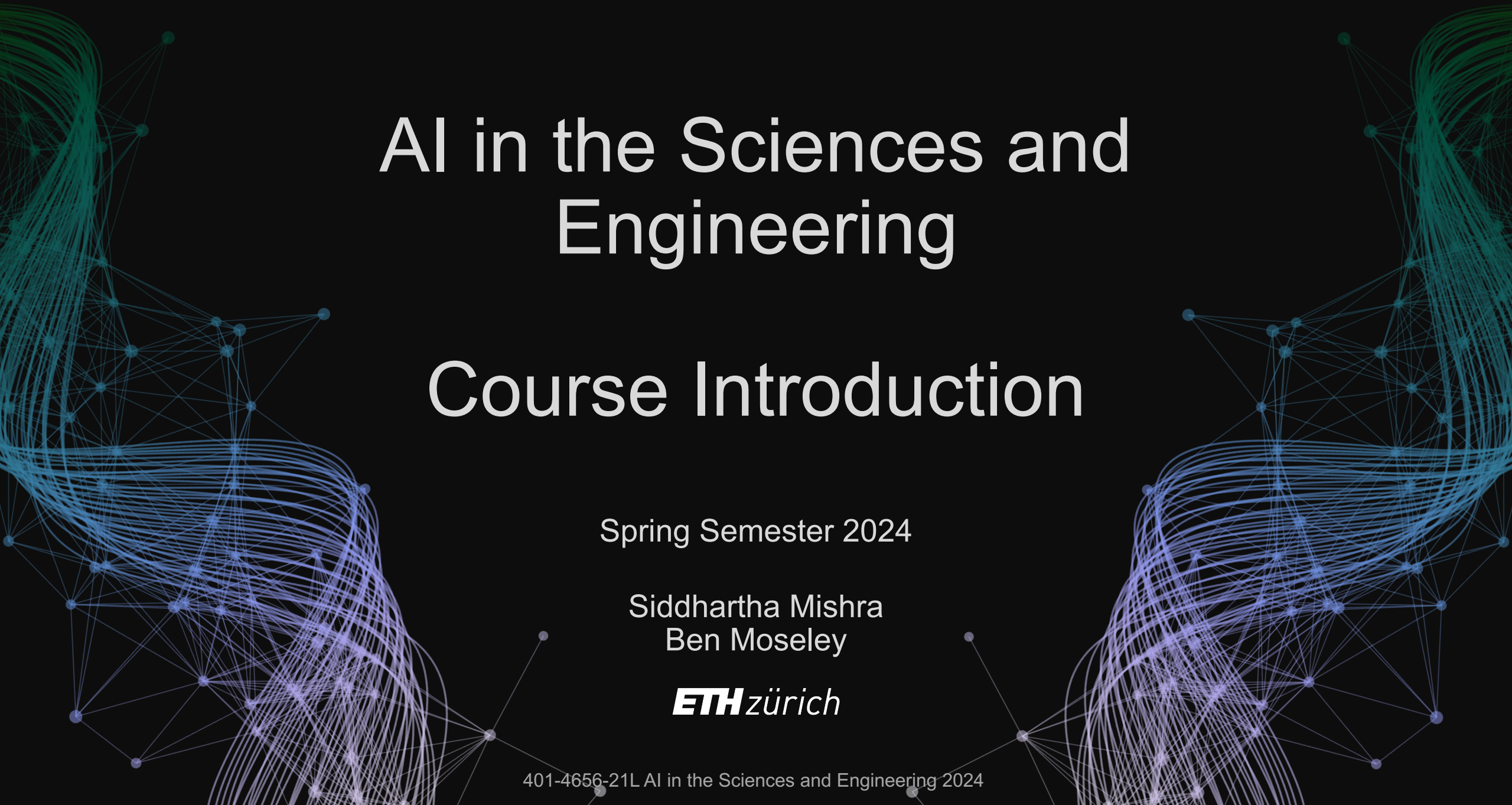


AI in the Sciences and Engineering

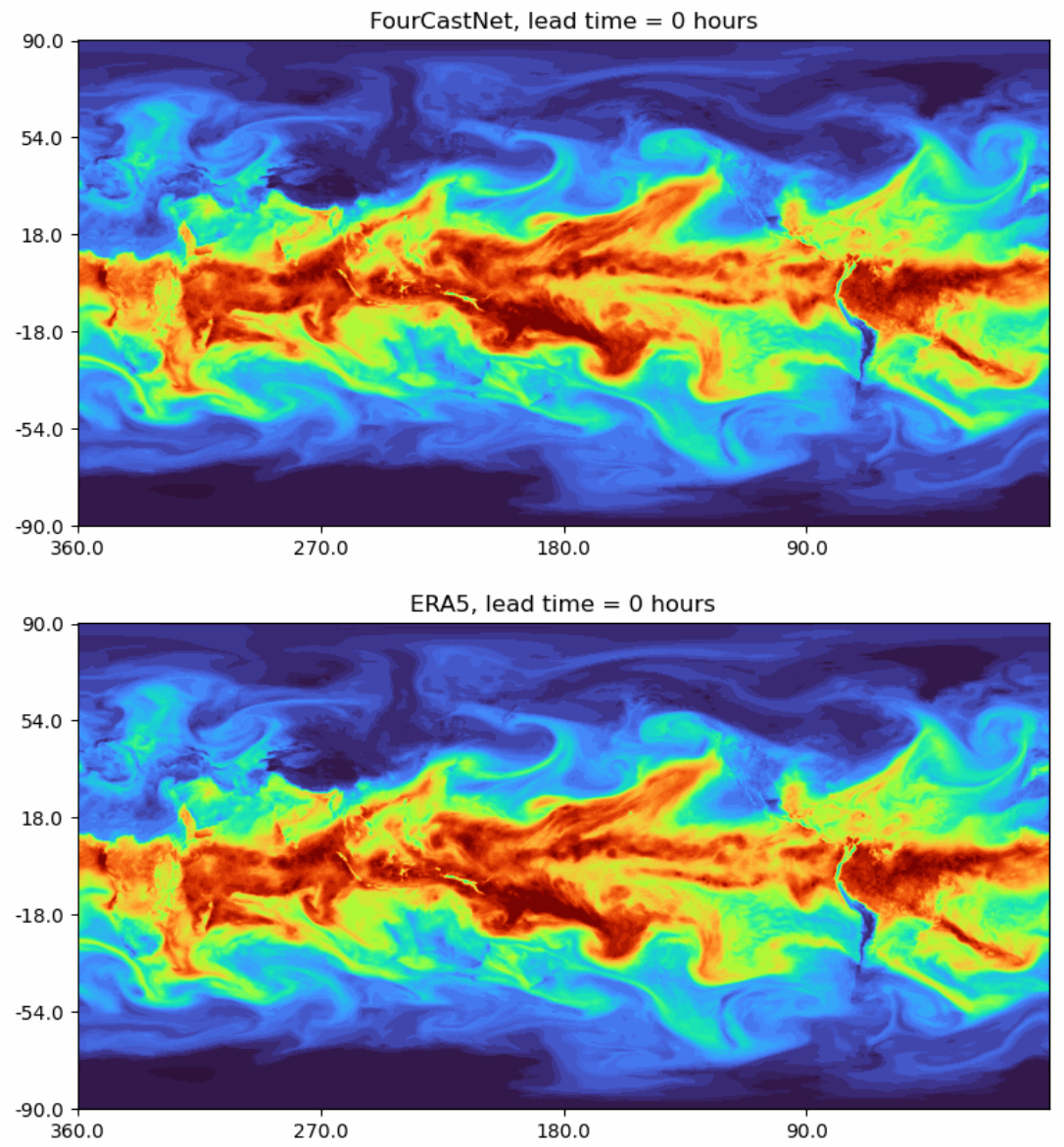
Course Introduction

Spring Semester 2024

Siddhartha Mishra
Ben Moseley

ETH zürich

AI for science: a revolution?

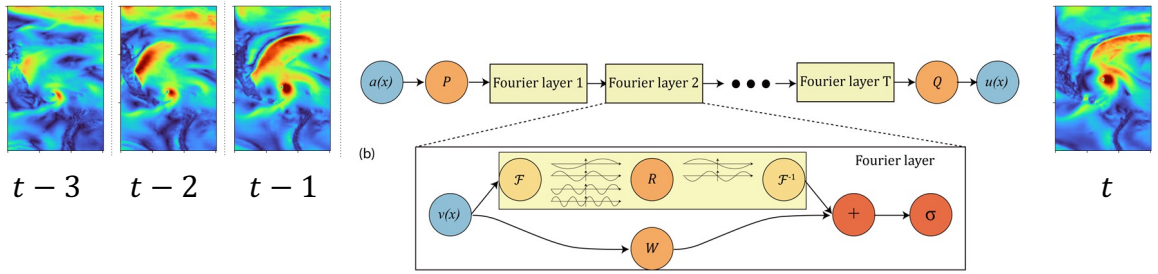


The Washington Post
Democracy Dies in Darkness

WEATHER Extreme Weather Climate Capital Weather Gang Environment Climate Lab

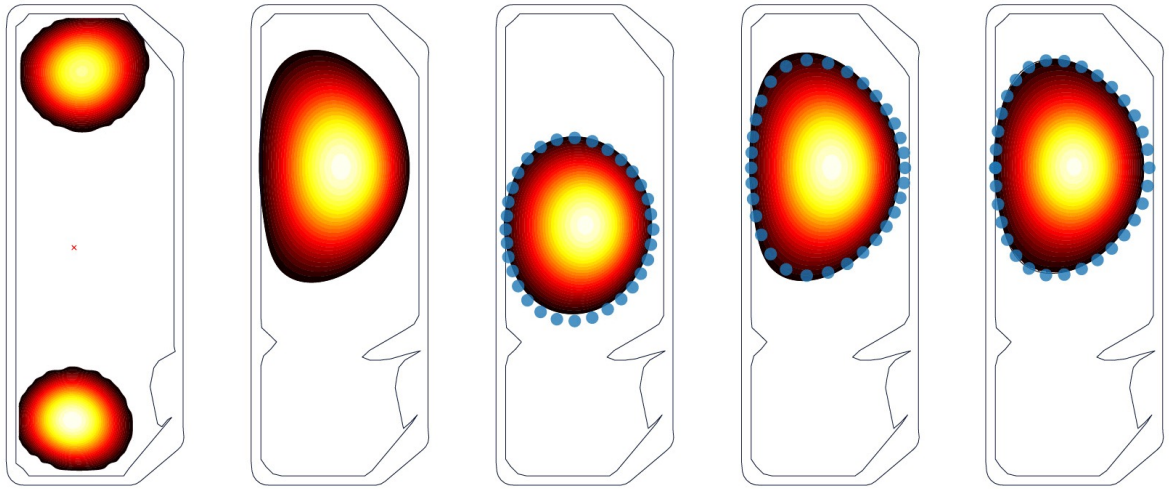
How Big Tech AI models nailed forecast for Hurricane Lee a week in advance

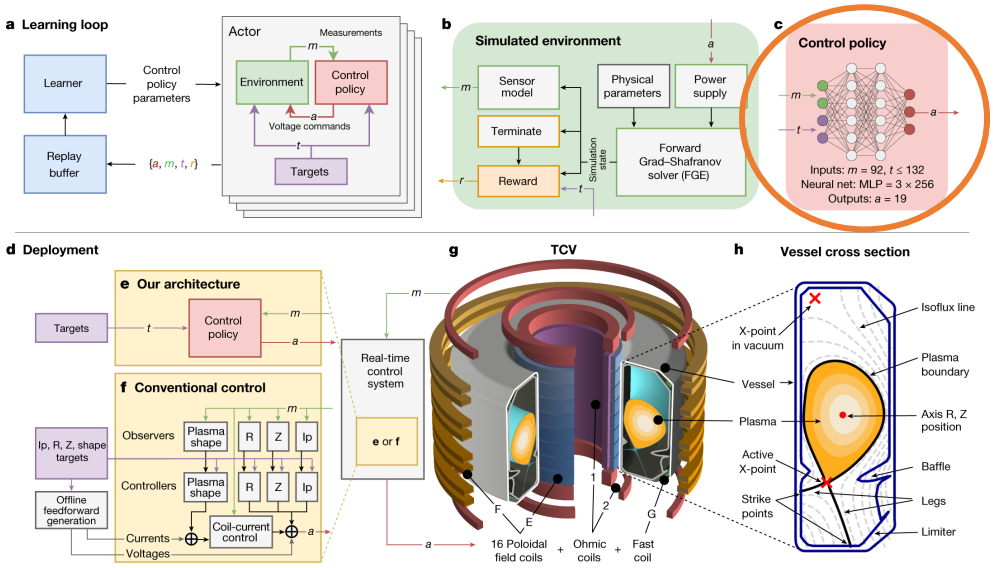
U.S. and European weather agencies are escalating their engagement with artificial intelligence as the technology rapidly advances

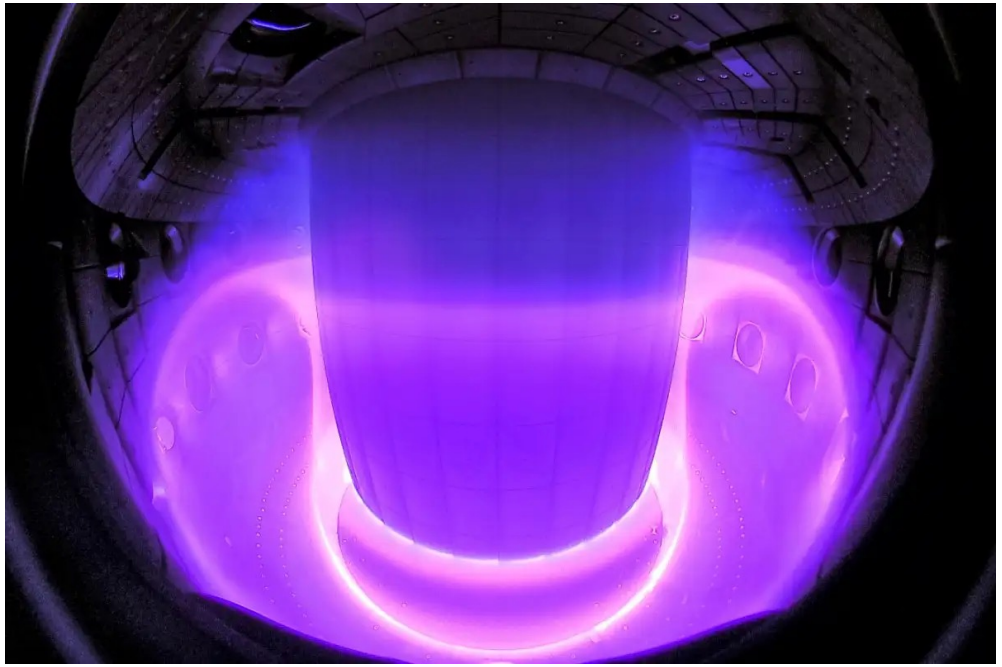


Pathak et al, FourCastNet: A Global Data-driven High-resolution Weather Model using Adaptive Fourier Neural Operators, ArXiv (2022)

AI for science: a revolution?







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

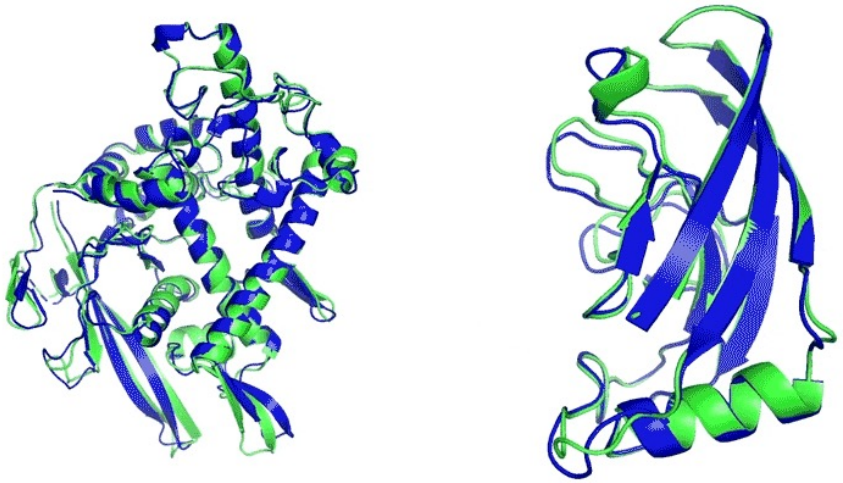
Magnetic control of tokamak plasmas through deep reinforcement learning

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

AI for science: a 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

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

The rise of AI

You
how powerful is AI today?

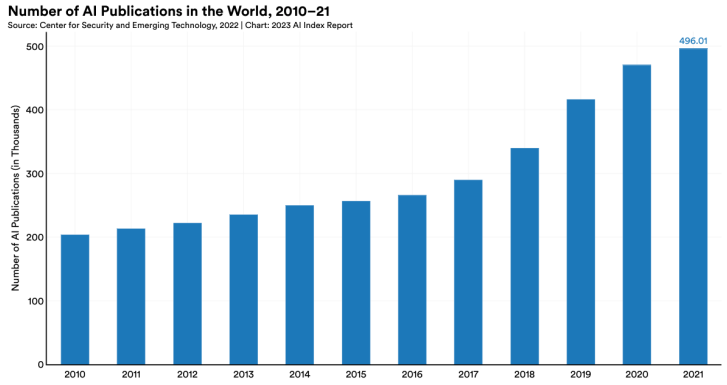
ChatGPT
As of my last update in January 2022, AI has made significant strides in various domains, demonstrating remarkable capabilities in tasks such as natural language processing, image recognition, speech synthesis, and more. Here are some aspects of AI's current power:

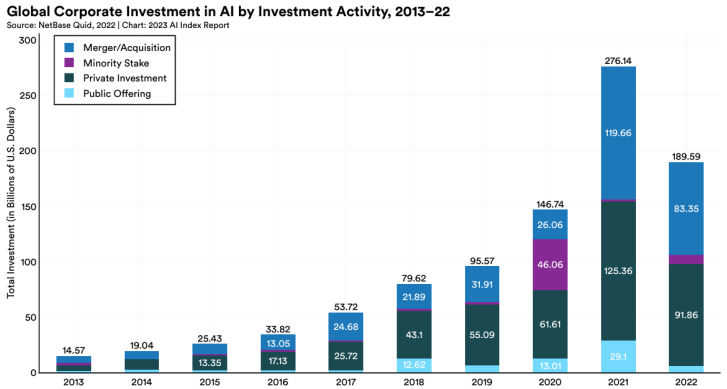
1. **Natural Language Processing (NLP)**: AI models like GPT-3 (the architecture I'm based on) have demonstrated the ability to generate human-like text, answer questions, summarize information, and even engage in basic forms of conversation. These models have advanced to the point where they can generate coherent and contextually relevant text across a wide range of topics.
2. **Computer Vision**: AI-powered systems are now capable of accurately recognizing and classifying objects, people, and activities in images and videos. They are used in various applications such as autonomous vehicles, medical imaging, surveillance, and augmented reality.
3. **Speech Recognition and Synthesis**: AI technologies can accurately transcribe spoken language into text and generate human-like speech. Virtual assistants like Siri, Alexa, and Google Assistant rely on these capabilities to understand and respond to user commands.

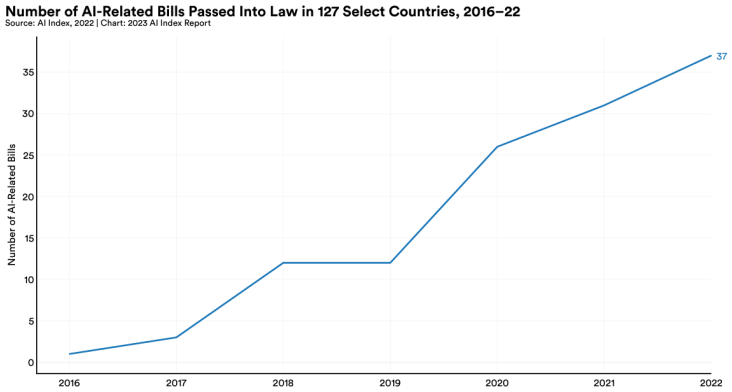
Brown et al, Language Models are Few-Shot Learners, NeurIPS (2020)

"a photograph of an astronaut riding a horse"

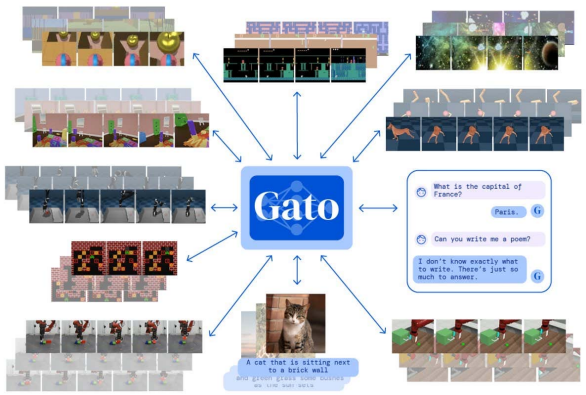
Source: Stable Diffusion Rombach et al, High-Resolution Image Synthesis with Latent Diffusion Models, CVPR (2022)



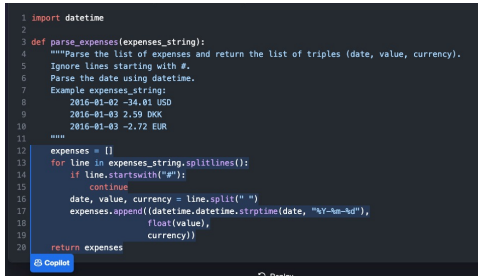




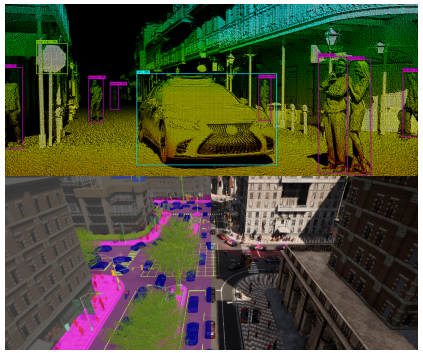
Source: AI Index Report, Stanford University



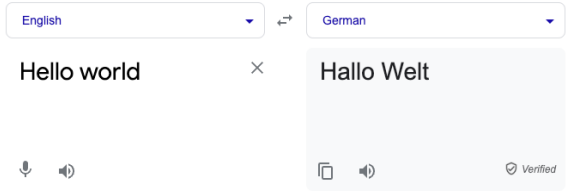
Reed et al., A Generalist Agent, TMLR (2022)



Source: GitHub Copilot

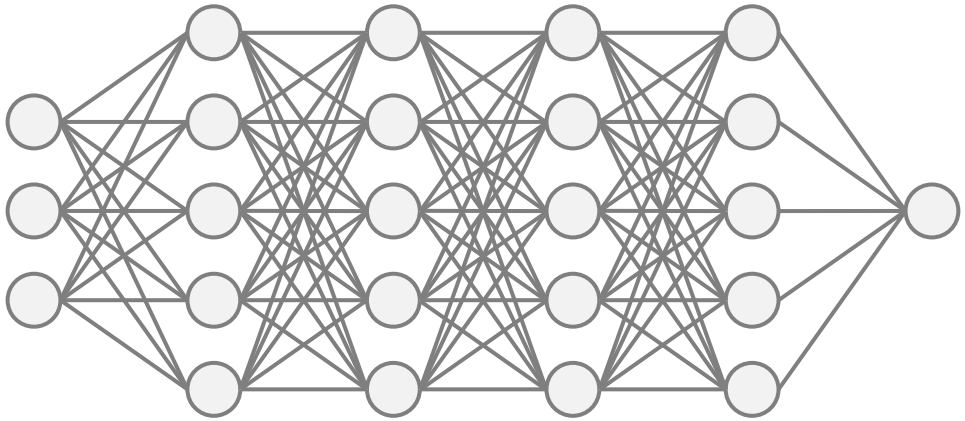


Source: Machine Learning for Autonomous Driving Workshop, NeurIPS (2023)



Source: Google Translate

What is deep learning?



Probability(Dog) = 1

Input

$\boldsymbol{x}$

Model

$NN(\boldsymbol{x}, \boldsymbol{\theta})$

Output

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

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

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

For example:

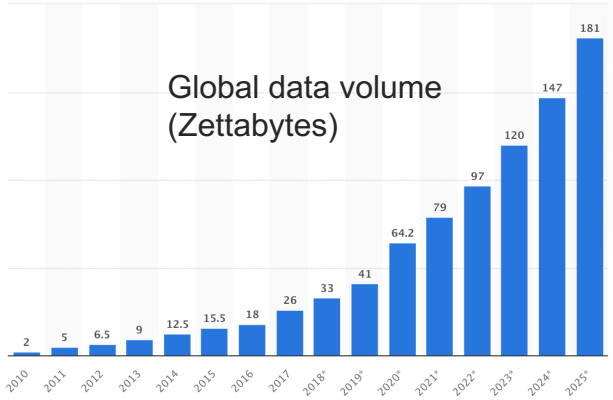
$$\boldsymbol{y} = W_2\sigma(W_1\boldsymbol{x} + \boldsymbol{b}_1) + \boldsymbol{b}_2$$

Trained ($\boldsymbol{\theta}$ learned) using:
- Many examples of inputs and outputs
- A loss function
- An optimisation algorithm, e.g. stochastic gradient descent

Why now?

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

Rapidly increasing amounts of data

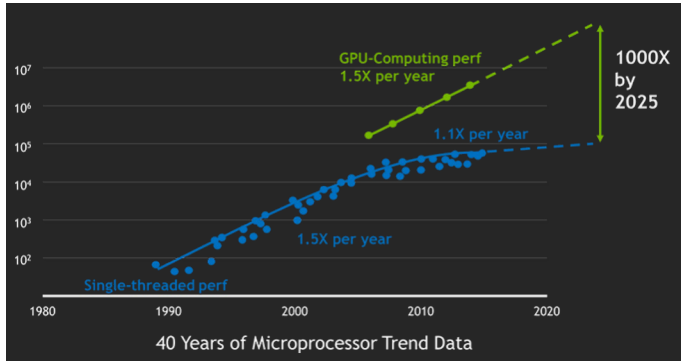


Source: Statista

IMAGENET

WIKIPEDIA
The Free Encyclopedia

Hardware improvements



Source: NVIDIA

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

Software improvements

PyTorch

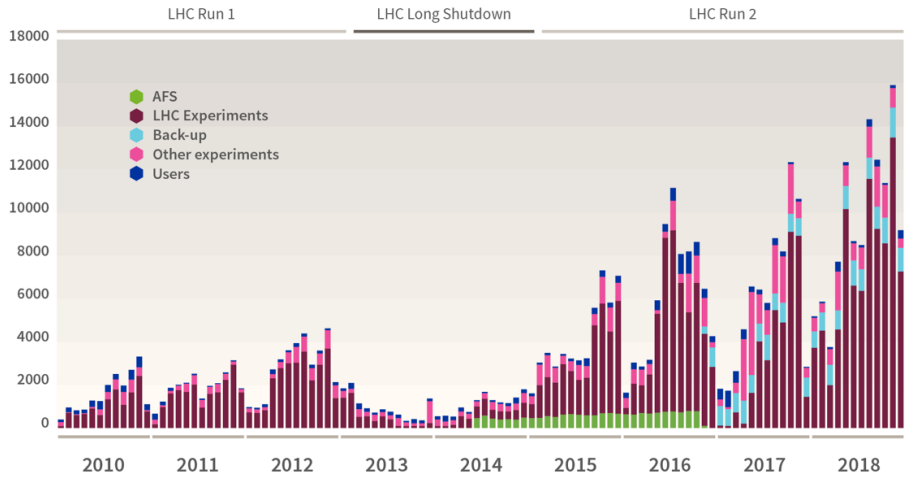
TensorFlow

Keras

- Mature deep learning frameworks
- Better training algorithms
- Deeper and more sophisticated architectures

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

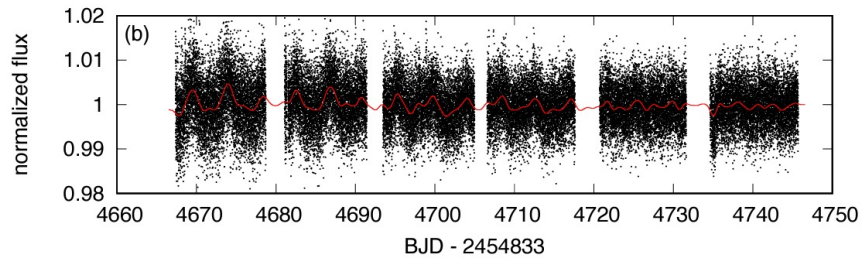
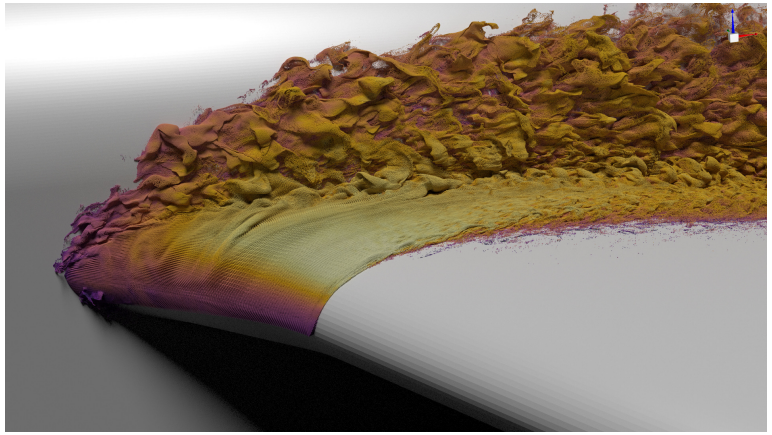


Figure 1. Light curves of K2-415 obtained by K2 (top; K2SFF) and TESS (bottom; PDC-SAP). Those data were taken at long (≈ 29 minutes) and short (2 minutes) cadences for K2 and TESS light curves, respectively. The red solid line in each panel represents the GP regression to the observed light curve (see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

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

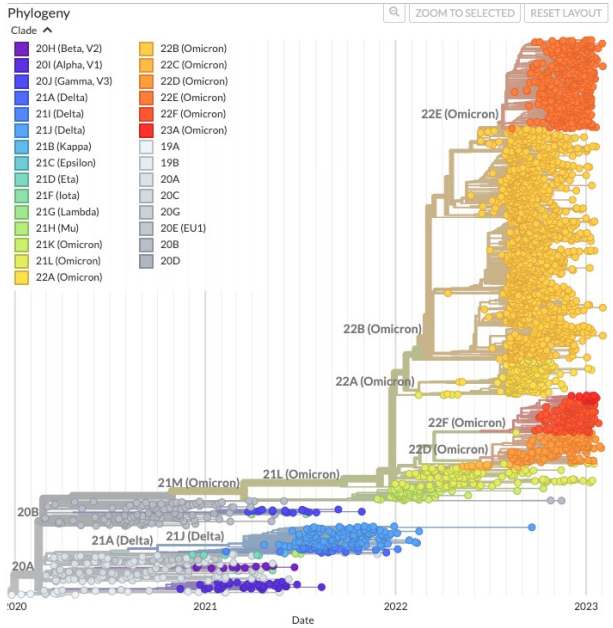
Air flow over a wing: wall-modeled large eddy simulation (~20 million core-hours) Source: NASA Ames

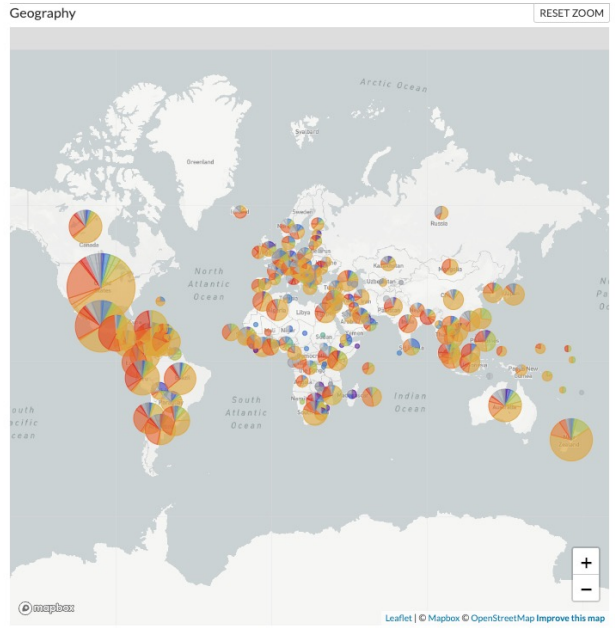


Genomic epidemiology of SARS-CoV-2 with subsampling focused globally over the past 6 months

Built with nextstrain/ncov. Maintained by the Nextstrain team. Enabled by data from GISAID.

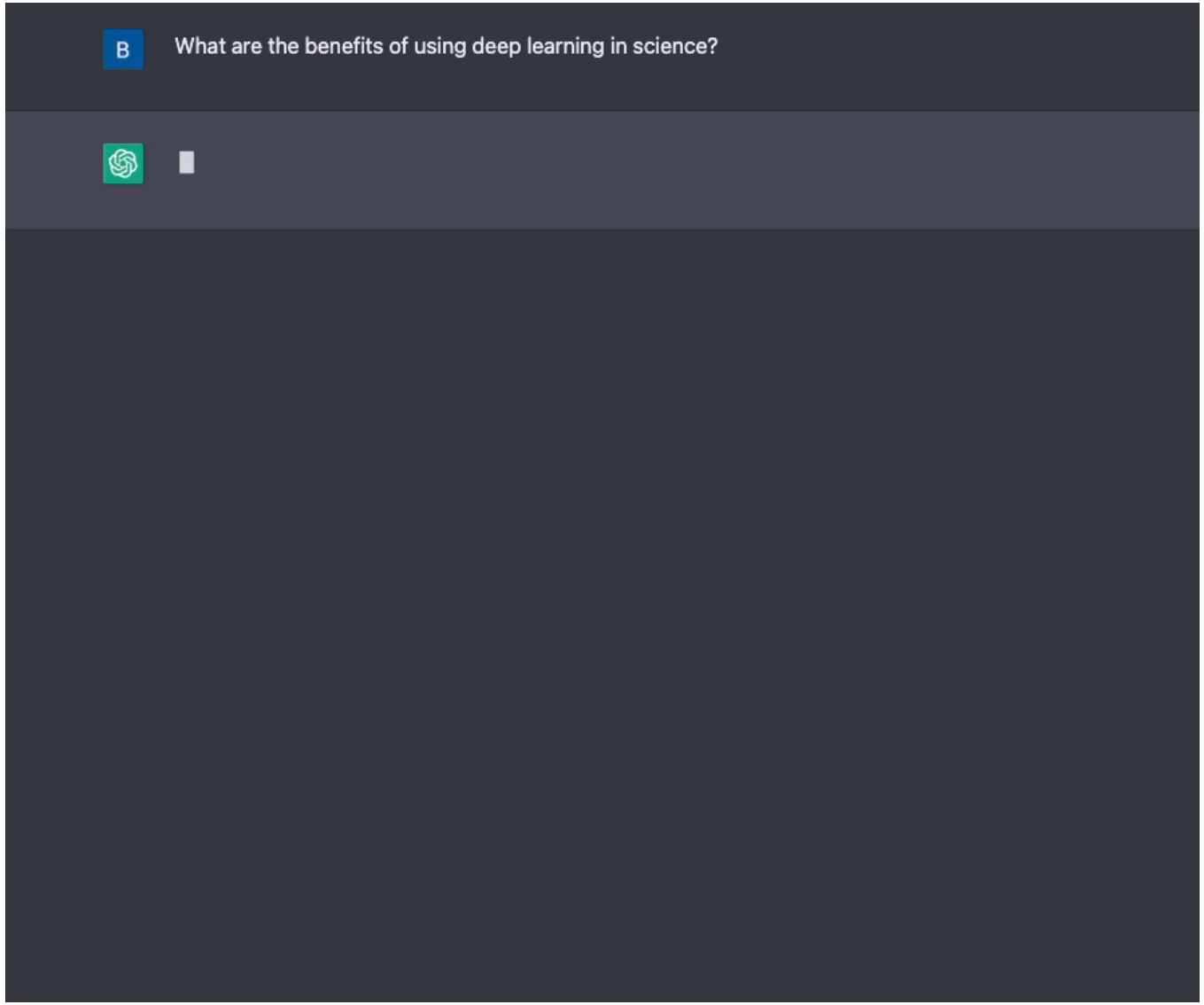
Showing 2767 of 2767 genomes sampled between Dec 2019 and Feb 2023.

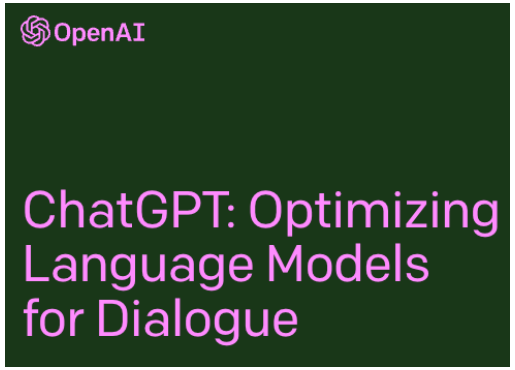




Source: Nextstrain

Deep learning for science

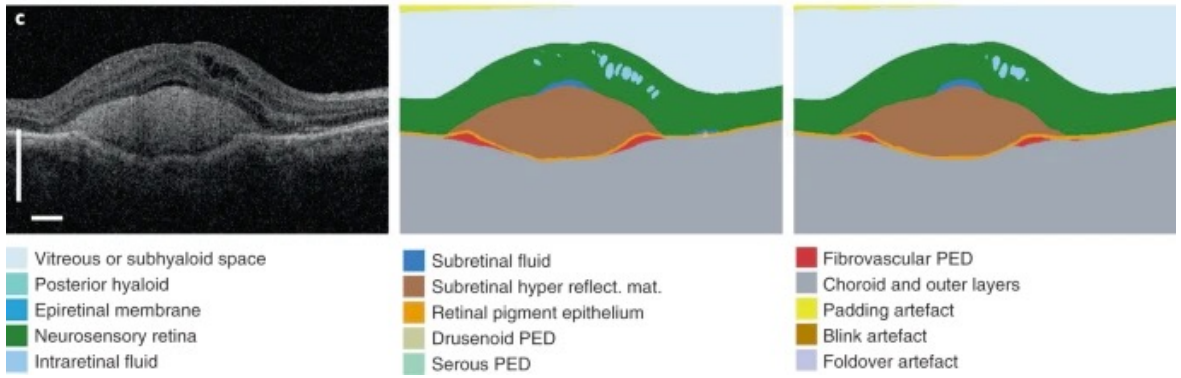




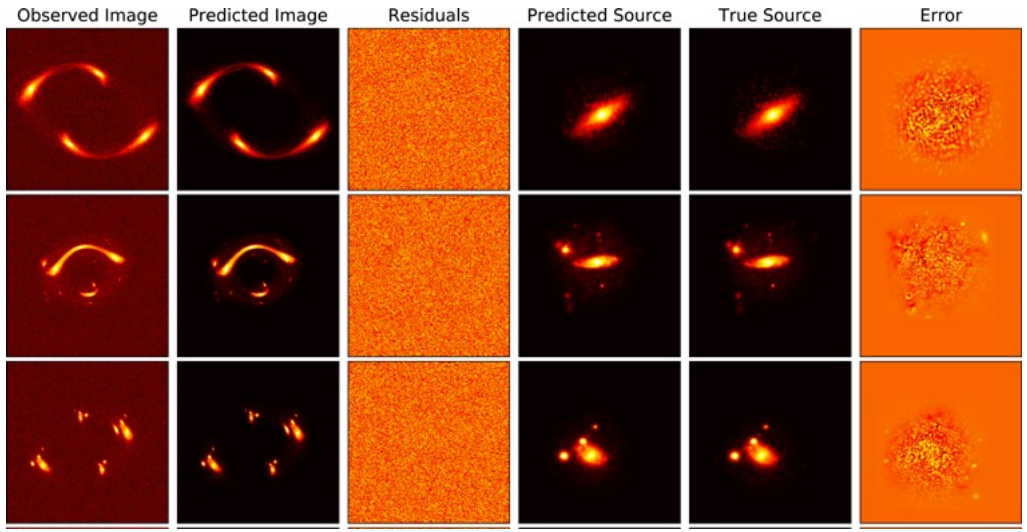
Source: OpenAI

A scientific revolution?

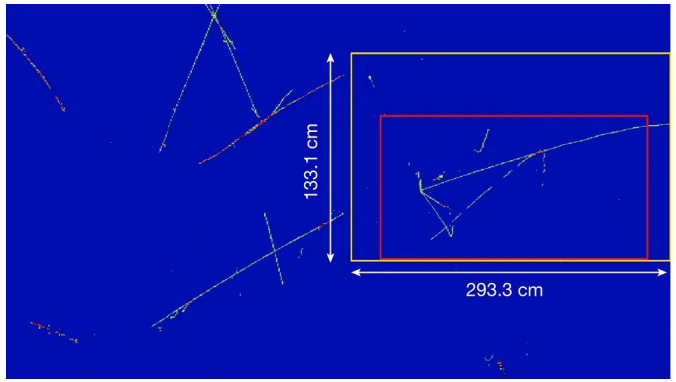
De Fauw, et al. Clinically applicable deep learning for diagnosis and referral in retinal disease, Nature Med (2018)

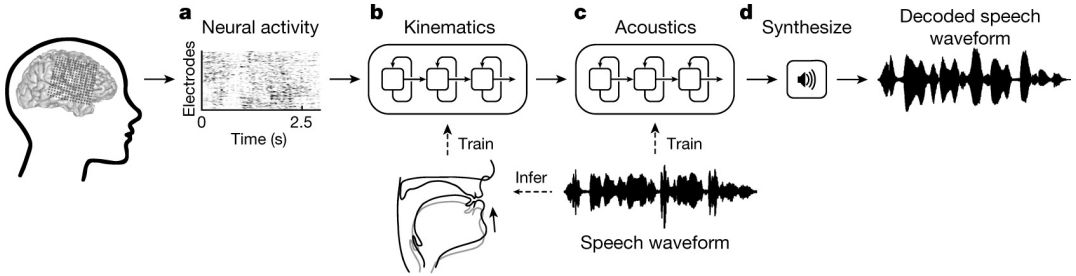


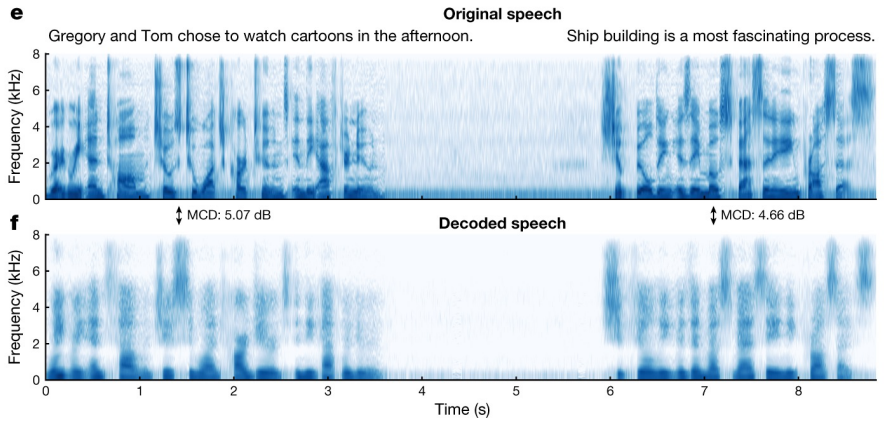
Morningstar et al, Data-driven Reconstruction of Gravitationally Lensed Galaxies Using Recurrent Inference Machines, ApJ (2019)



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

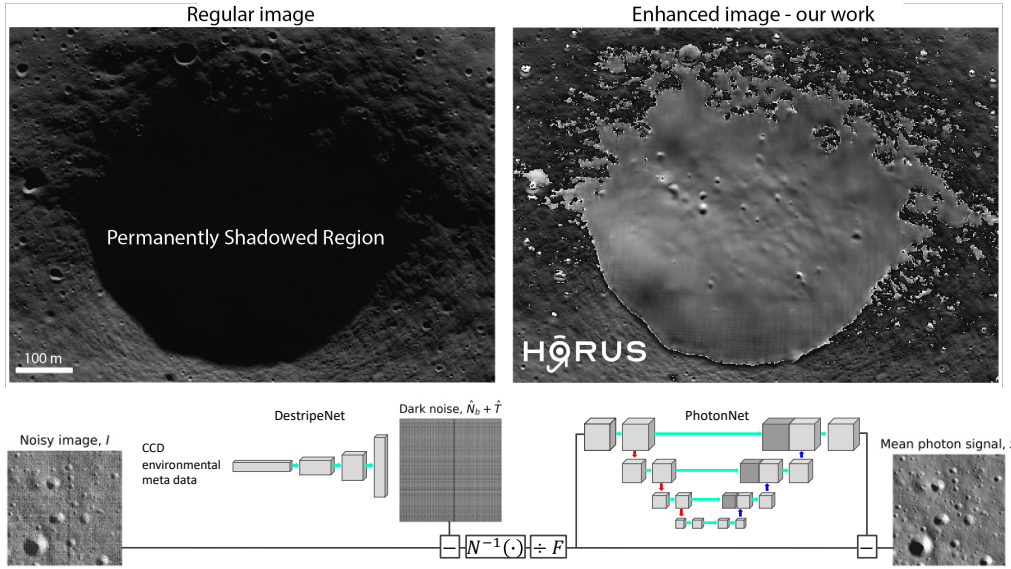




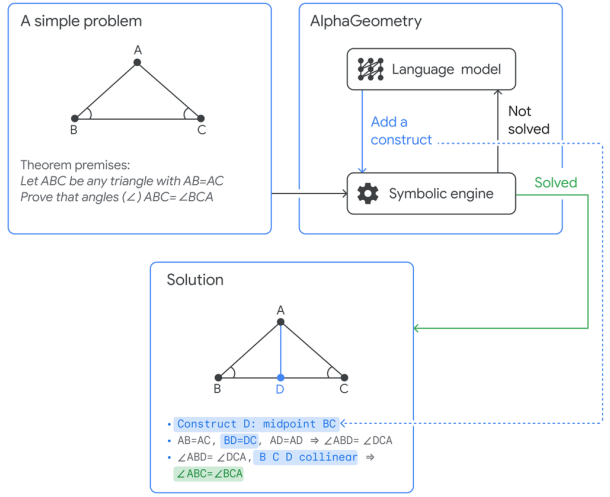


Anumanchipalli et al, Speech synthesis from neural decoding of spoken sentences. Nature (2019)

A scientific revolution?



Moseley et al, Extreme Low-Light Environment-Driven Image Denoising Over Permanently Shadowed Lunar Regions With a Physical Noise Model, CVPR (2021)

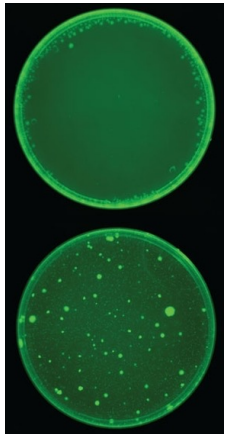


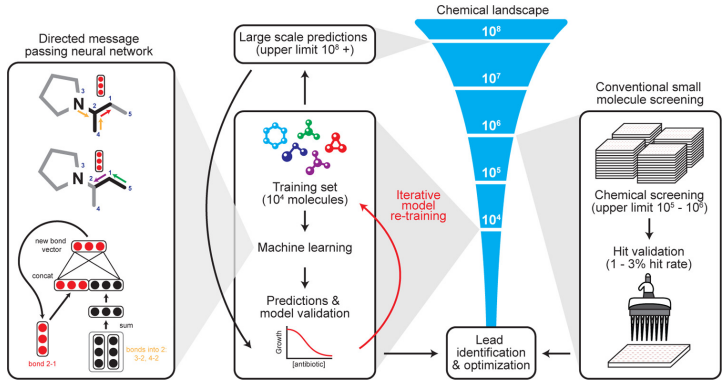
Trinh et al, Solving olympiad geometry without human demonstrations, Nature (2024)

E. coli resistance

Halicin (discovered)

Ciprofloxacin (ineffective)





Stokes et al, A deep learning approach to antibiotic discovery, Cell (2020)

A Write a Python code for solving the 2D diffusion equation in a square box using the FTCS (Forward Time Centered Space) method. Use Dirichlet boundary conditions such that the value of the variable is 0 on the right and left boundaries, while they are 1 on the top and bottom.

Here is a Python code for solving the 2D diffusion equation in a square box using the FTCS method:

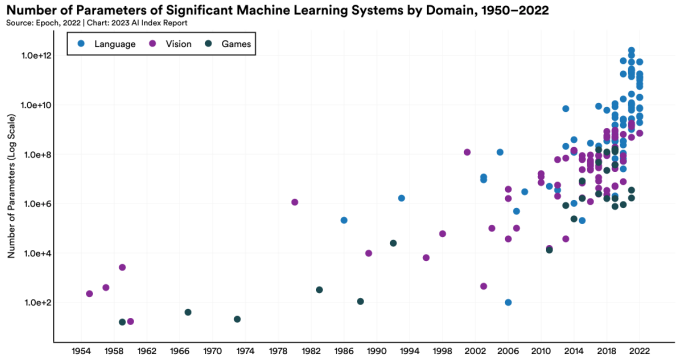
```
import numpy as np
import matplotlib.pyplot as plt

# Define the parameters
D = 0.1  # Diffusion coefficient
L = 1.0  # Length of the square box
N = 100  # Number of grid points in each direction
dx = L / (N - 1)  # Grid spacing
dy = L / (N - 1)
dt = 0.0001  # Time step
t_final = 1.0  # Final time

# Define the initial condition
u = np.zeros((N, N))
u[:, 0] = 1.0  # Bottom boundary
u[:, -1] = 1.0  # Top boundary
u[0, :] = 0.0  # Left boundary
u[-1, :] = 0.0  # Right boundary

# Define the coefficients for the FTCS method
alpha = D * dt / dx**2
beta = D * dt / dy**2

# Iterate in time
t = 0.0
while t < t_final:
    # Compute the next time step
    u[1:-1, 1:-1] += alpha * (u[2:, 1:-1] - 2*u[1:-1, 1:-1] + u
    [:-2, 1:-1]) + beta * (u[1:-1, 2:] - 2*u[1:-1, 1:-1] + u[1:-1,
     :-2])

    # Update the boundary conditions
    u[:, 0] = 1.0  # Bottom boundary
    u[:, -1] = 1.0  # Top boundary
    u[0, :] = 0.0  # Left boundary
    u[-1, :] = 0.0  # Right boundary
```

Kashefi and Mukerji, ChatGPT for Programming Numerical Methods, Journal of Machine Learning for Modeling and Computing (2023)

Challenges of deep learning

OpenAI

ChatGPT: Optimizing Language Models for Dialogue



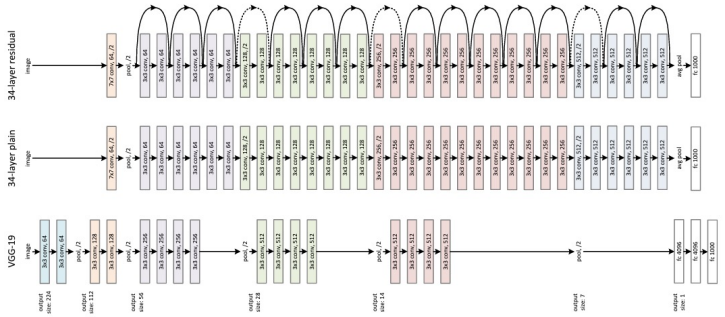
2.1 Model and Architectures

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)



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

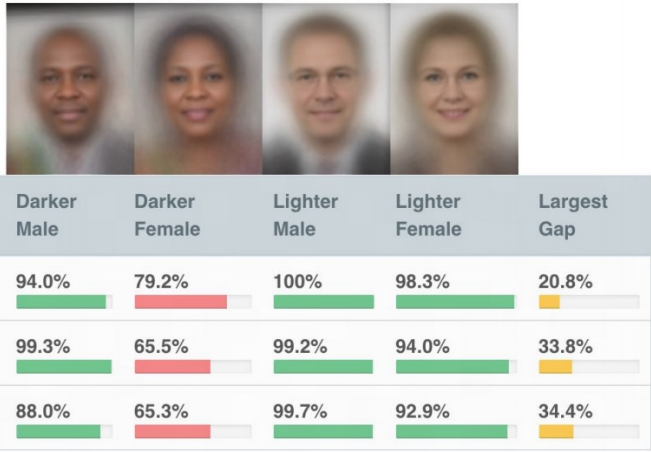
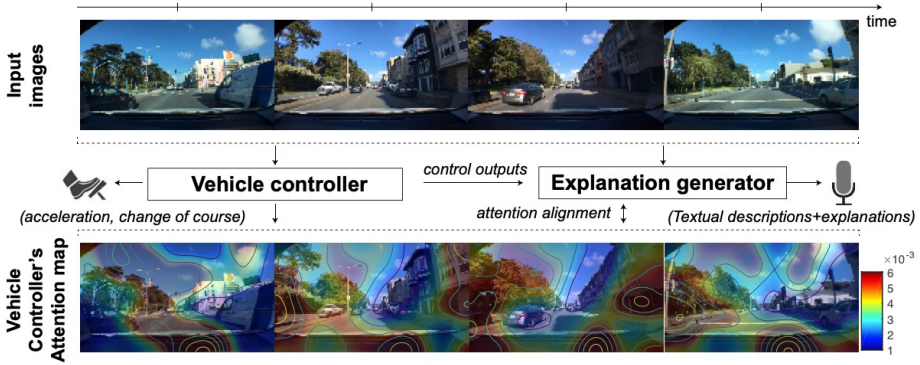


Figure 1. Intersectional Skin Type and Gender Classification Accuracy Disparities.

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

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"*

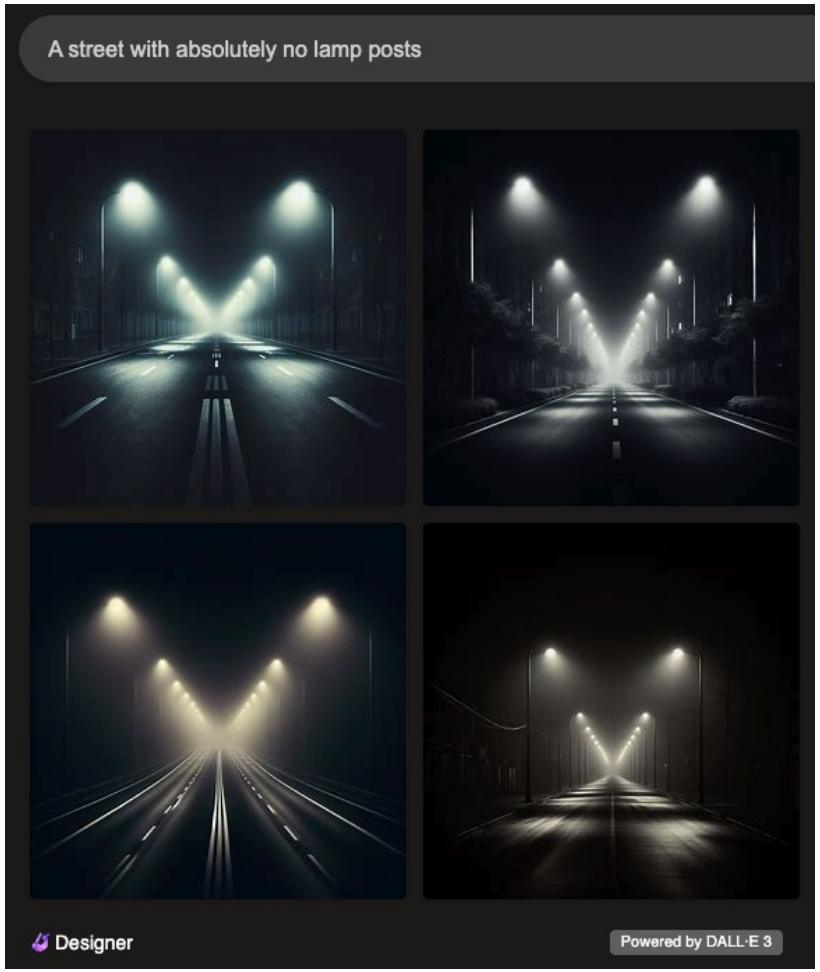
Human annotator: *"The car heads down the street + because the street is clear."*

The challenge of generalisation

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

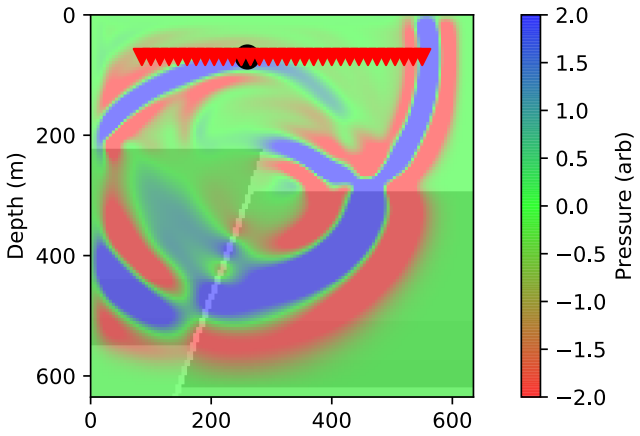
laboratory coat 40%
jeweler's loupe 8%
English foxhound 6%
soccer ball 4%
neck brace 3%

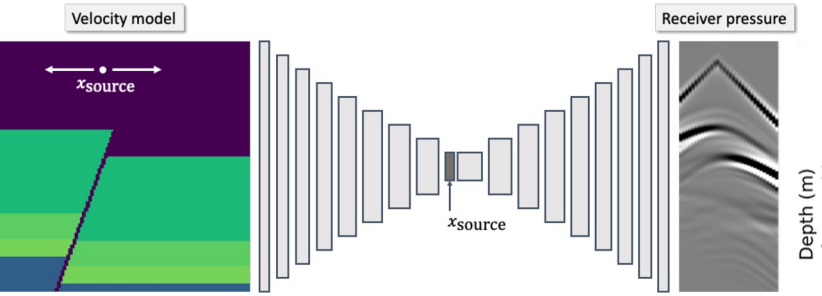
The challenge of reasoning

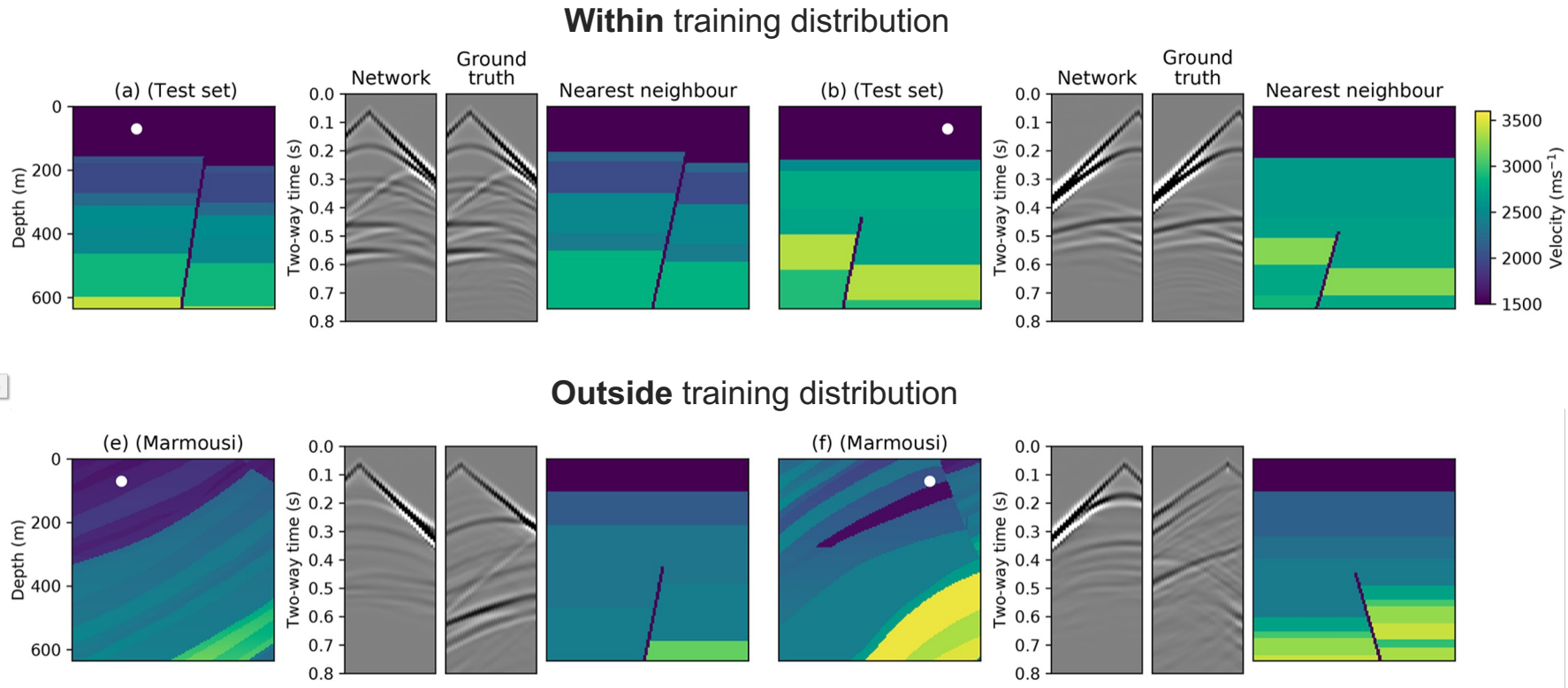


Source: DALL·E 3

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

Scientific machine learning (SciML)

Major problem

Naively using deep learning for scientific tasks usually leads to:

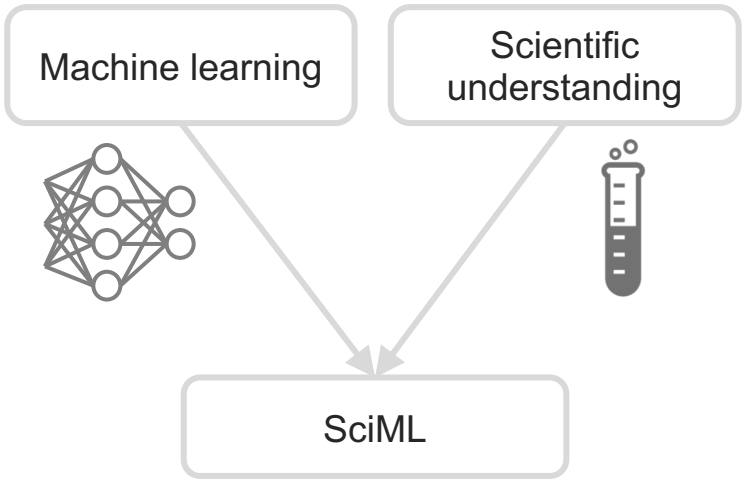
- Lack of interpretability
- Poor generalisation
- Lots of training data required

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

Traditional scientific method:

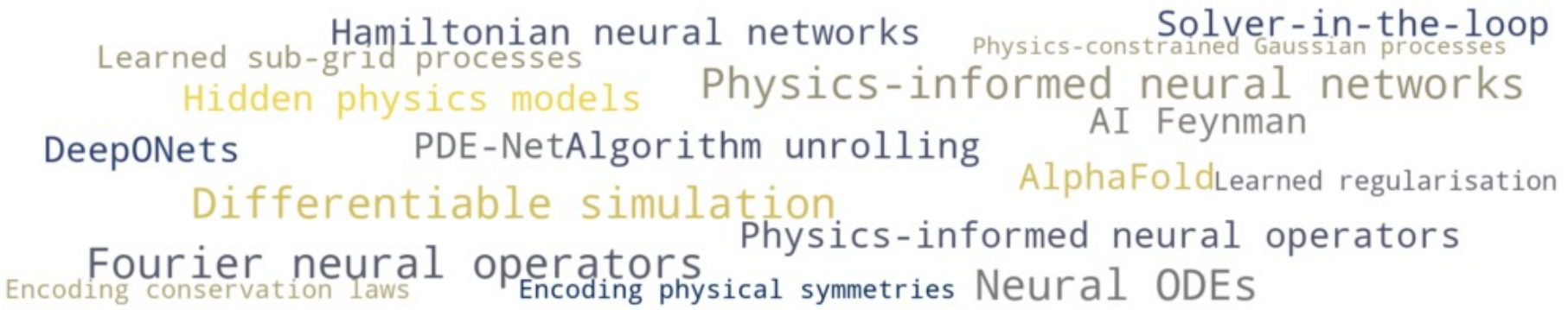
- Revolves around theory and experiment
- a good theory should be explainable and make **novel** predictions

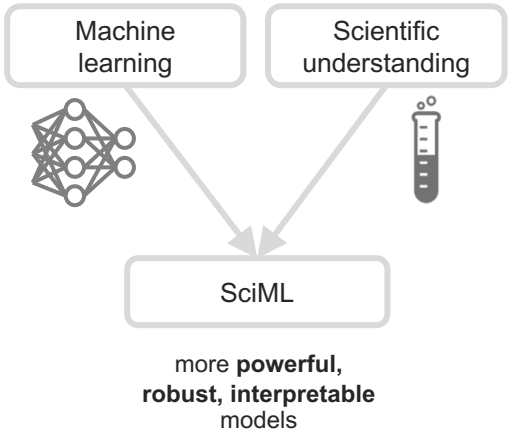
Solution



more **powerful, robust, interpretable** models

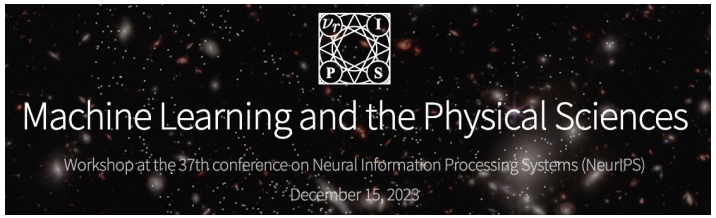
Scientific machine learning

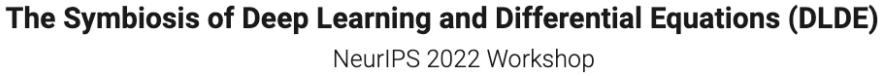


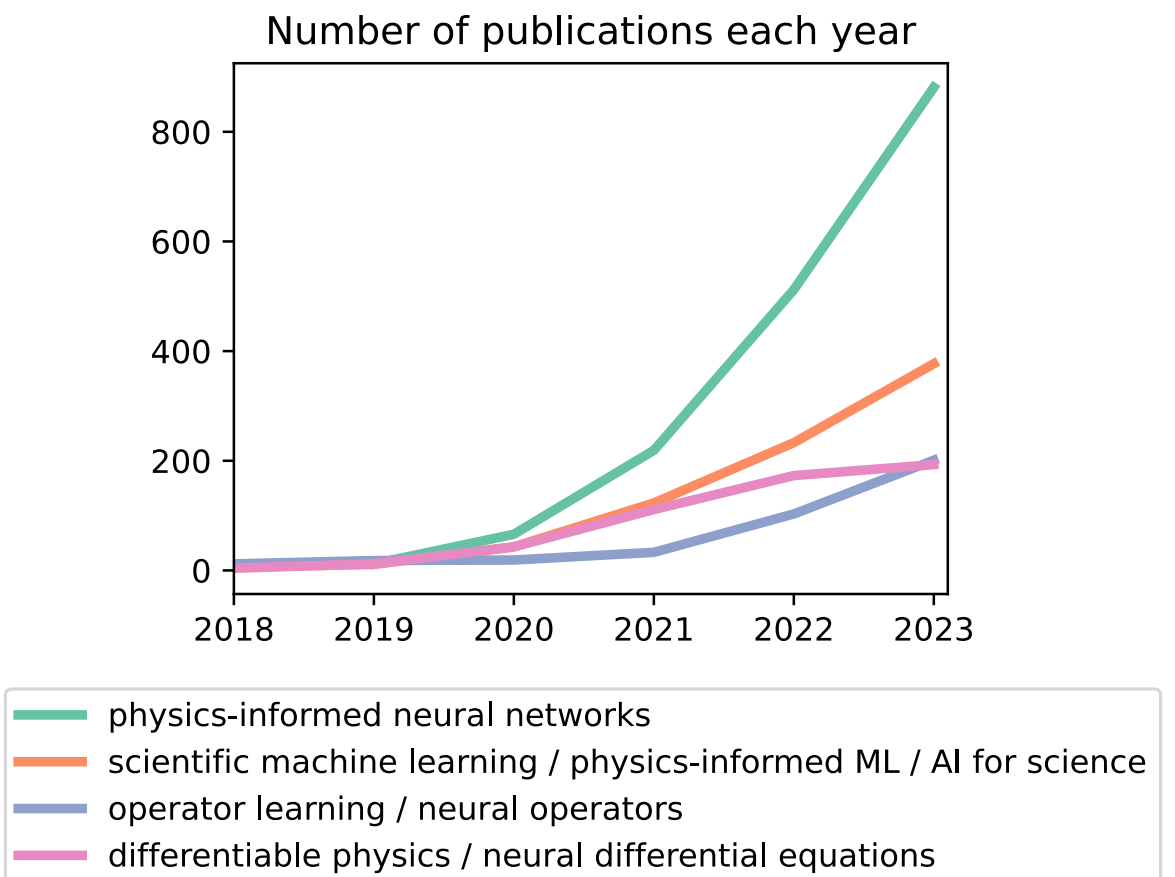


more **powerful, robust, interpretable** models

A rapidly growing field







Source: Scopus keyword search (Feb 2024)

Course learning objectives

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

Course timeline

Tutorials		Lectures			
Mon 12:15-14:00 HG E 5		*Wed 08:15-10:00 ML H 44*		*Fri 12:15-13:00 ML H 44*	
19.02.		21.02.	**Course introduction**	23.02.	Introduction to deep learning I
26.02.	Introduction to PyTorch	28.02.	Introduction to deep learning II	01.03.	Importance of PDEs in science
04.03.	CNNs and surrogate modelling	06.03.	Introduction to physics-informed neural networks	08.03.	PINNs – limitations and extensions
11.03.	Implementing PINNs I	13.03.	PINNs – extensions and theory	15.03.	PINNs – theory
18.03.	Implementing PINNs II	20.03.	Introduction to operator learning	22.03.	DeepONets and spectral neural operators
25.03.	Operator learning I	27.03.	Fourier- and convolutional- neural operators	29.03.	
01.04.		03.04.		05.04.	
08.04.	Operator learning II	10.04.	Operator learning – limitations and extensions	12.04.	Introduction to transformers
15.04.		17.04.	Foundational models for operator learning	19.04.	Graph neural networks for PDEs
22.04.	GNNs	24.04.	GNNs for PDEs / introduction to diffusion models	26.04.	Introduction to diffusion models
29.04.	Transformers	01.05.		03.05.	Diffusion models - applications
06.05.	Diffusion models	08.05.	Introduction to differentiable physics	10.05.	Hybrid workflows
13.05.	Coding autodiff from scratch	15.05.	Neural differential equations	17.05.	Introduction to JAX
20.05.		22.05.	Symbolic regression and equation discovery	24.05.	Course summary and future trends
27.05.	Introduction to JAX / NDEs	29.05.	Guest lecture: ML in chemistry and biology	31.05.	Guest lecture: ML in chemistry and biology

Practical information

- All materials will be available on course Moodle page:
  - Lecture slides + recordings
  - Tutorial exercises
- Performance assessment:
  - 3 x individual homework assignments, due every 4 weeks
    - Will be related to content covered in the lectures and tutorials
    - First assignment will be released on 8 March (deadline for submission: 5 April)
  - No exams!
- Teaching Assistants:
  - Bogdan Raonic, 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

Lecture overview

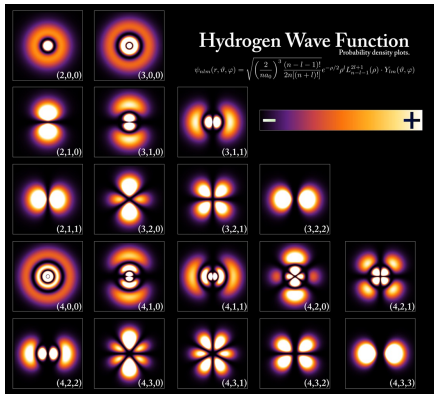
- Course motivation
- Key scientific tasks and how machine learning can help
  - Simulation
  - Inverse, control, and data assimilation problems
  - Equation discovery and anomaly detection
- Overview of scientific machine learning
  - Ways to incorporate scientific principles into deep learning

Learning objectives

- Understand why AI is useful in science
- Understand key scientific tasks we want to solve
- Explain the motivation behind SciML

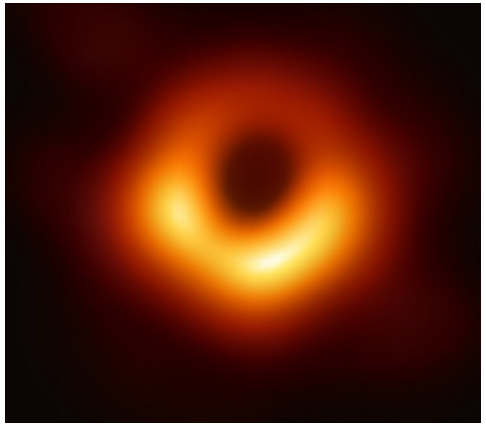
Key scientific tasks

PDEs are the building blocks of science



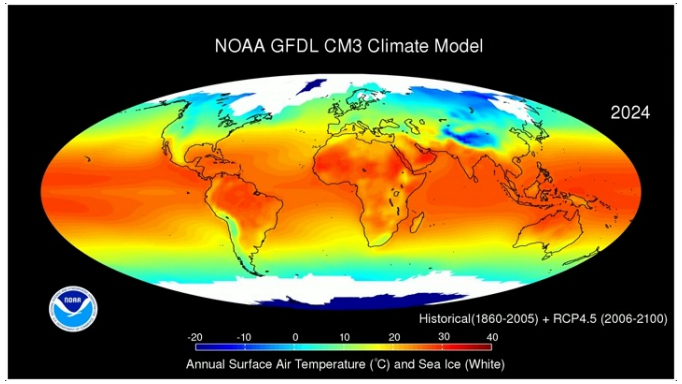
Source: Wikipedia

Schrödinger equation



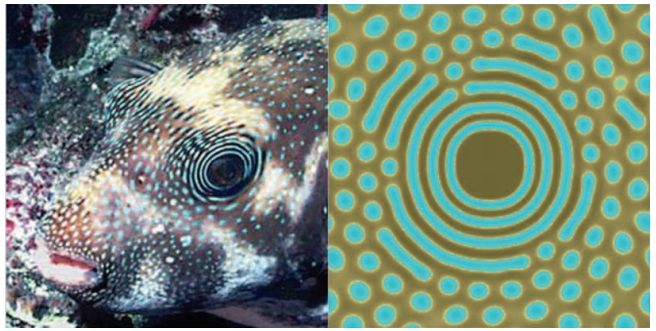
Source: The Event Horizon Telescope (2019)

Einstein field equations



Source: NOAA

Navier-Stokes equations



Source: Kondo and Miura, Science (2010)

Reaction-diffusion equation

Key scientific tasks

Key scientific tasks: simulation

$$b = F(a)$$

a = set of input conditions

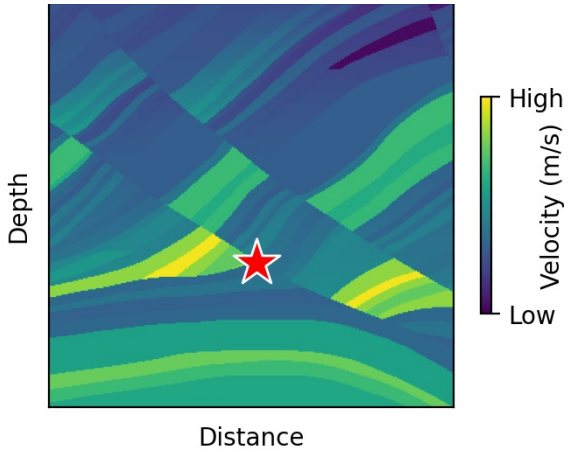
F = physical model of the system (usually a PDE)

b = resulting properties given F and a

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)

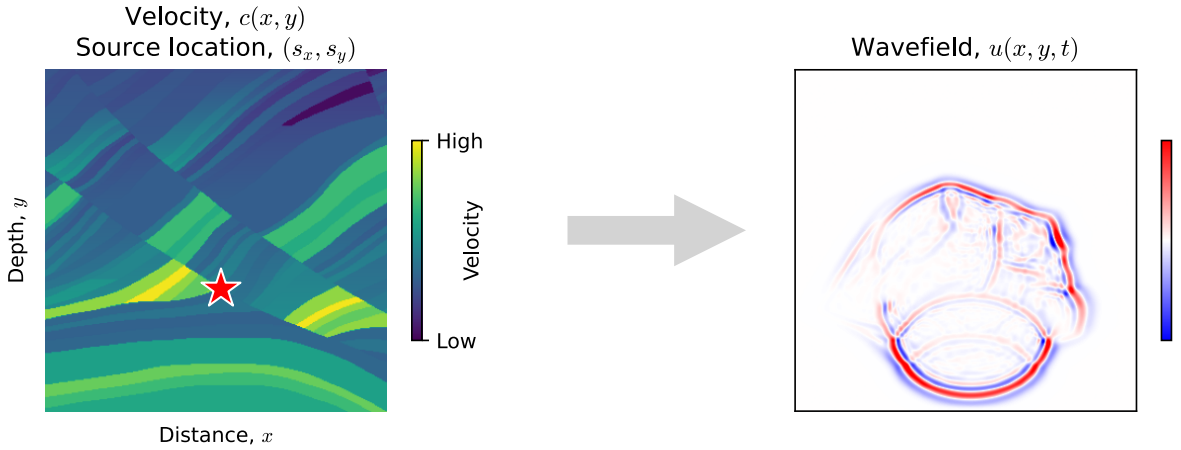


$$\nabla^2 u - \frac{1}{c(x)^2}\frac{\partial^2 u}{\partial t^2} = s$$

Wave equation

Key scientific tasks: simulation

$$b = F(a)$$



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

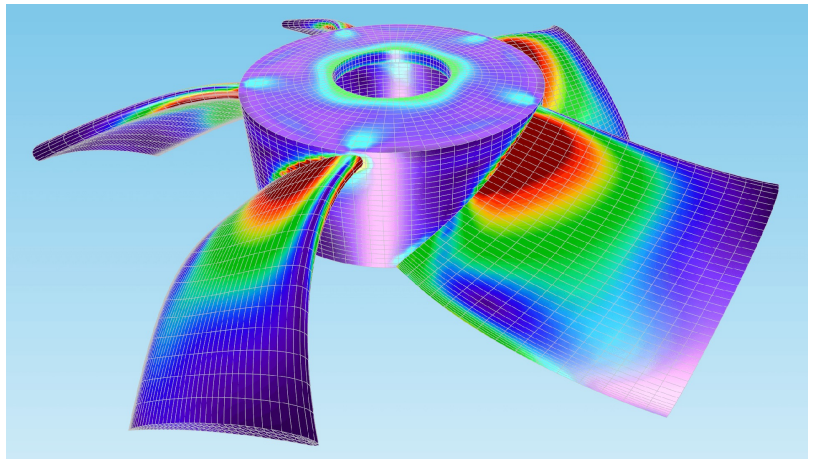
$a =$ velocity model, $c(x, y)$, source location, s

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

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

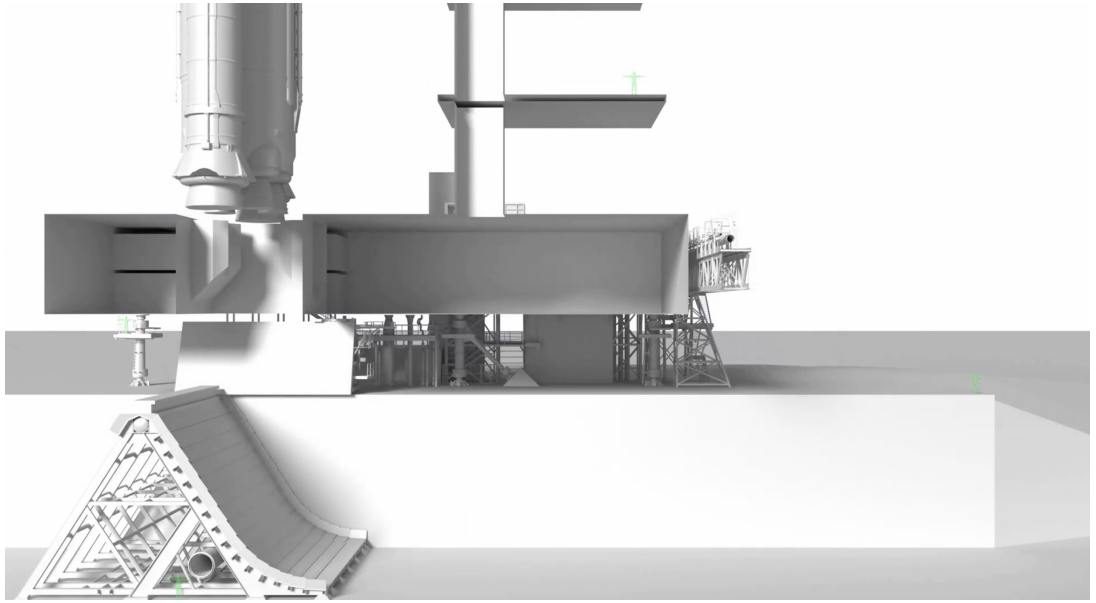
How can we solve simulation problems?

- 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) Source: 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 (usually a PDE)

b = resulting properties given F and a

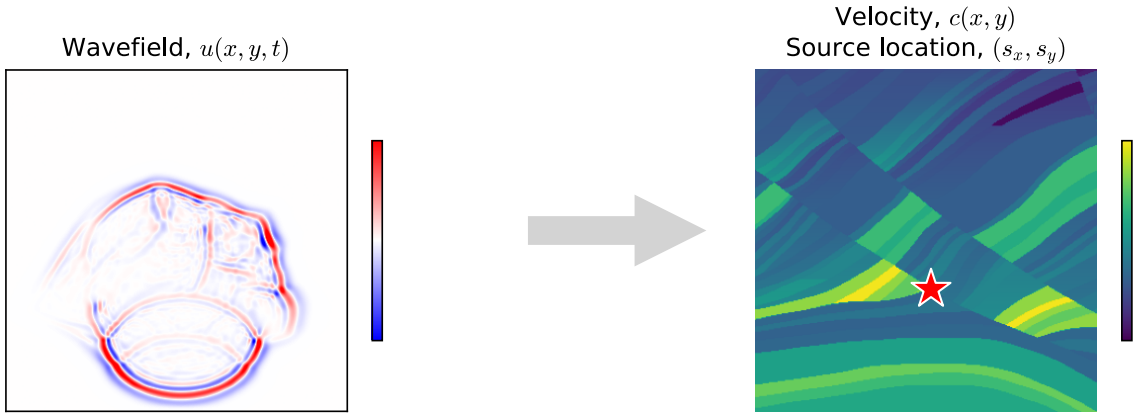
Key scientific tasks: inverse problems

$$b = F(a)$$

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?

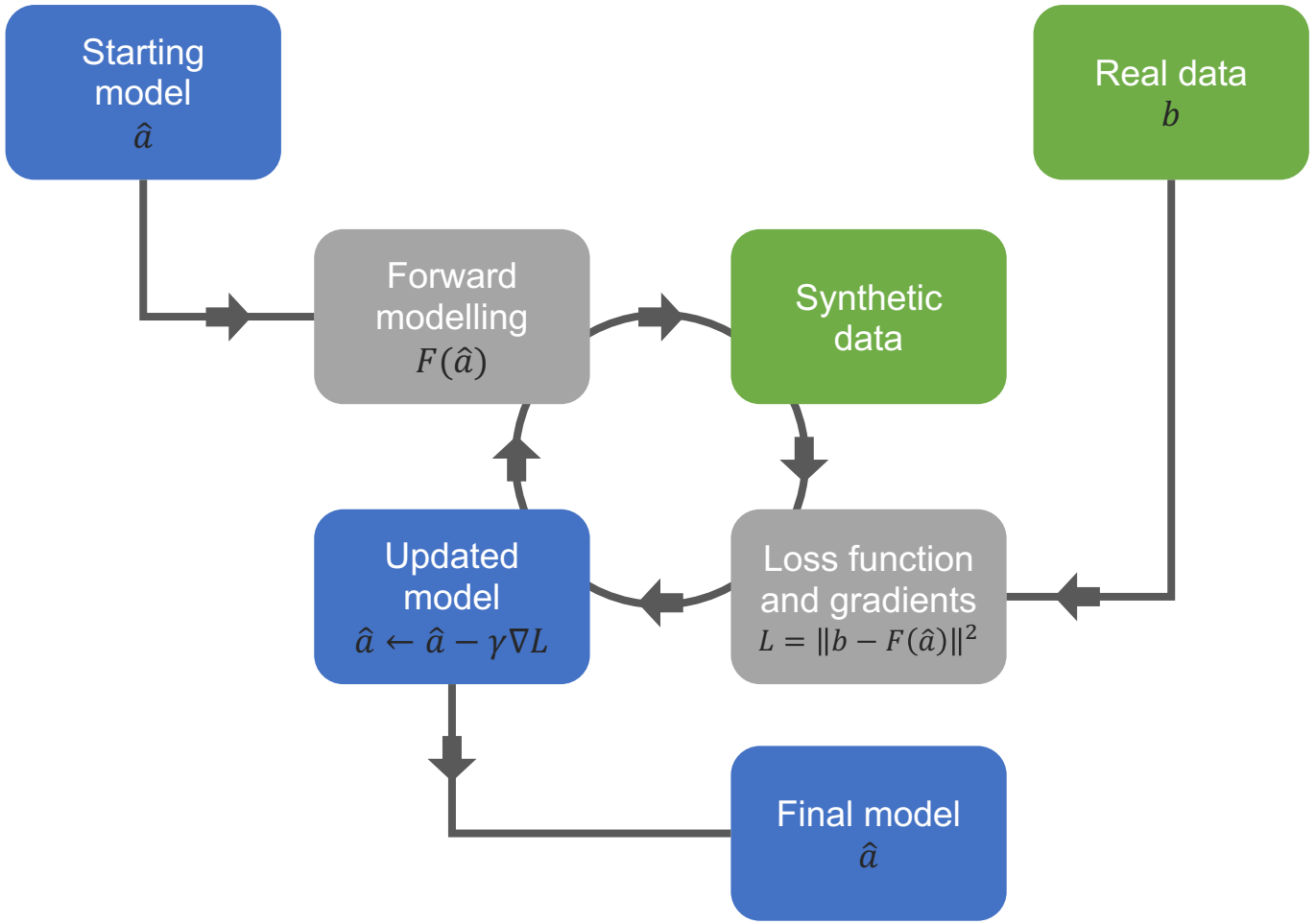
$$b = F(a)$$

- 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

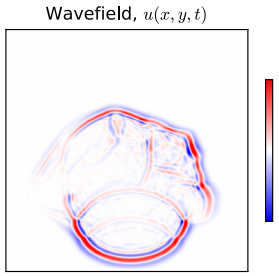
Solving inverse problems with gradient descent

$\hat{a}$ = starting velocity model

Starting model $\hat{a}$

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

Real data b



b = wavefield

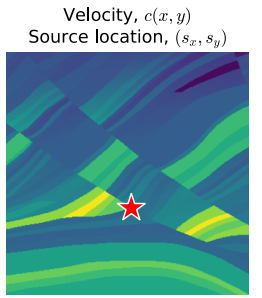
Forward modelling $F(\hat{a})$

Synthetic data

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

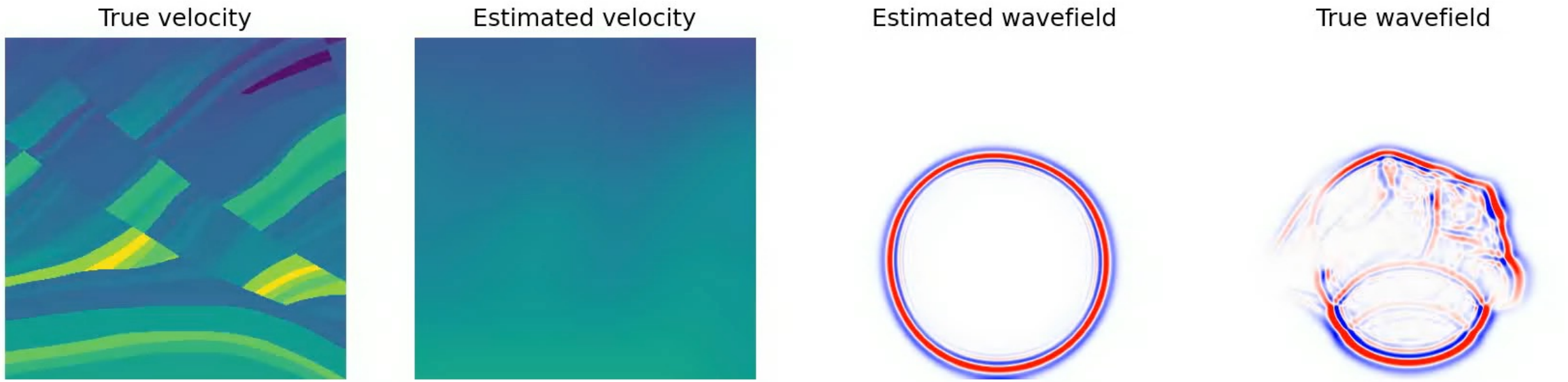
Loss function and gradients $L = \|b - F(\hat{a})\|^2$

Final model $\hat{a}$



a = true velocity model

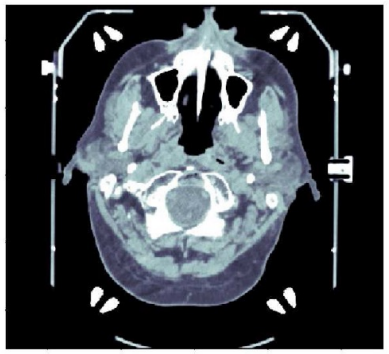
Solving inverse problems with gradient descent



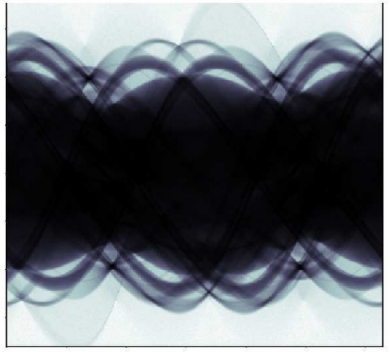
$$\min_{\hat{a}} \|b - F(\hat{a})\|^2$$

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

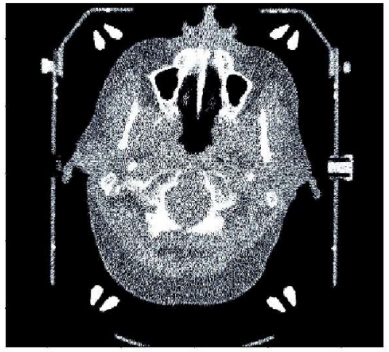
Challenges of inverse problems



Ground truth computed tomography image

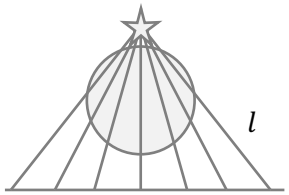


Resulting tomographic data (sinogram)



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)

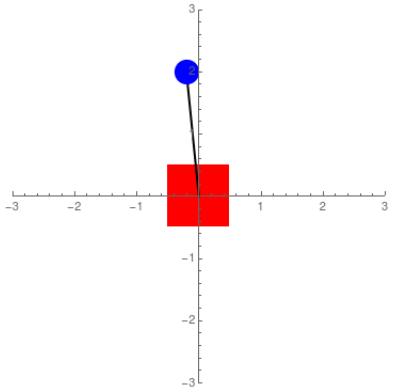
Typically, inverse problems are **incredibly challenging** to solve because:

- 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



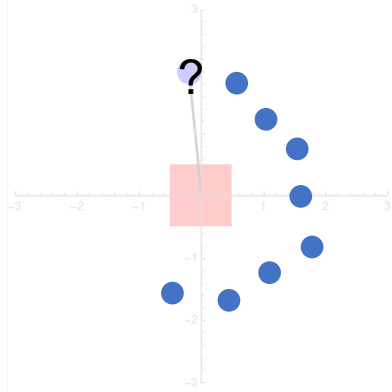
Source: Neil Gershenfeld / MIT

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



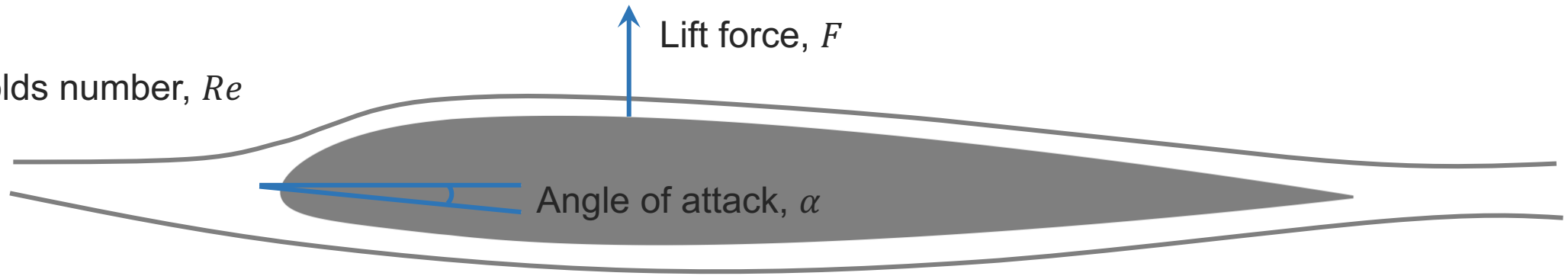
$$b = F(a)$$

$a = f(t)$, 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: design

- Also a type of inverse problem



Aerofoil shape (set of points), $\{x_i, y_i\}_{i=1}^{N}$

Design task:

Find $\{x_i, y_i\}_{i=1}^{N}$, α which maximize F

Key scientific tasks: discovering unknown physics

$$b = F(a)$$

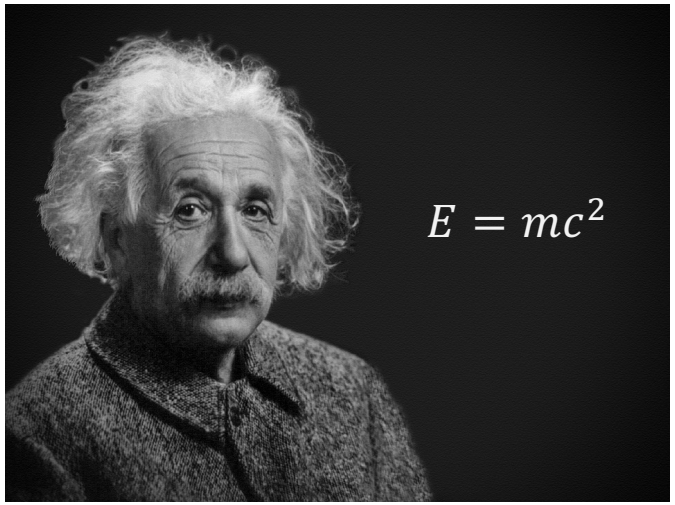
a = set of input conditions

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

b = resulting properties given F and a

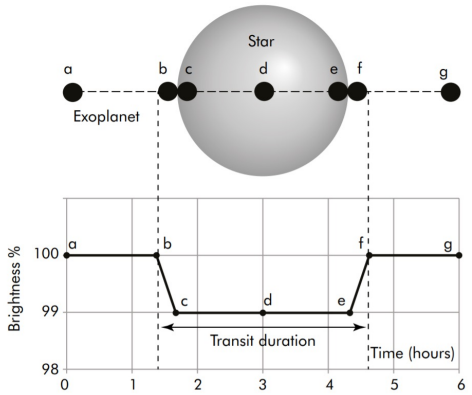
Challenges of discovering unknown physics

$$b = F(a)$$



- 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

Key scientific tasks: anomaly detection



Source: IOP

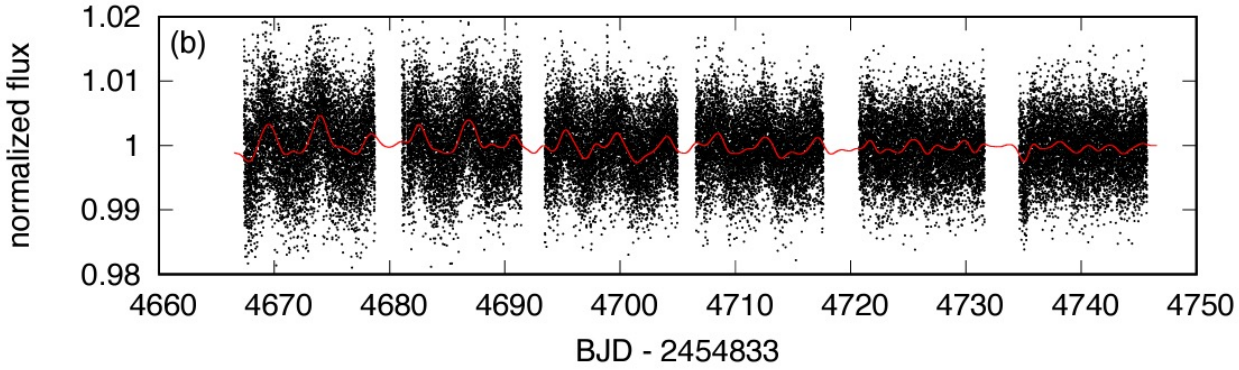


Figure 1. Light curves of K2-415 obtained by K2 (top; K2SFF) and TESS (bottom; PDC-SAP). Those data were taken at long (≈ 29 minutes) and short (2 minutes) cadences for K2 and TESS light curves, respectively. The red solid line in each panel represents the GP regression to the observed light curve (see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

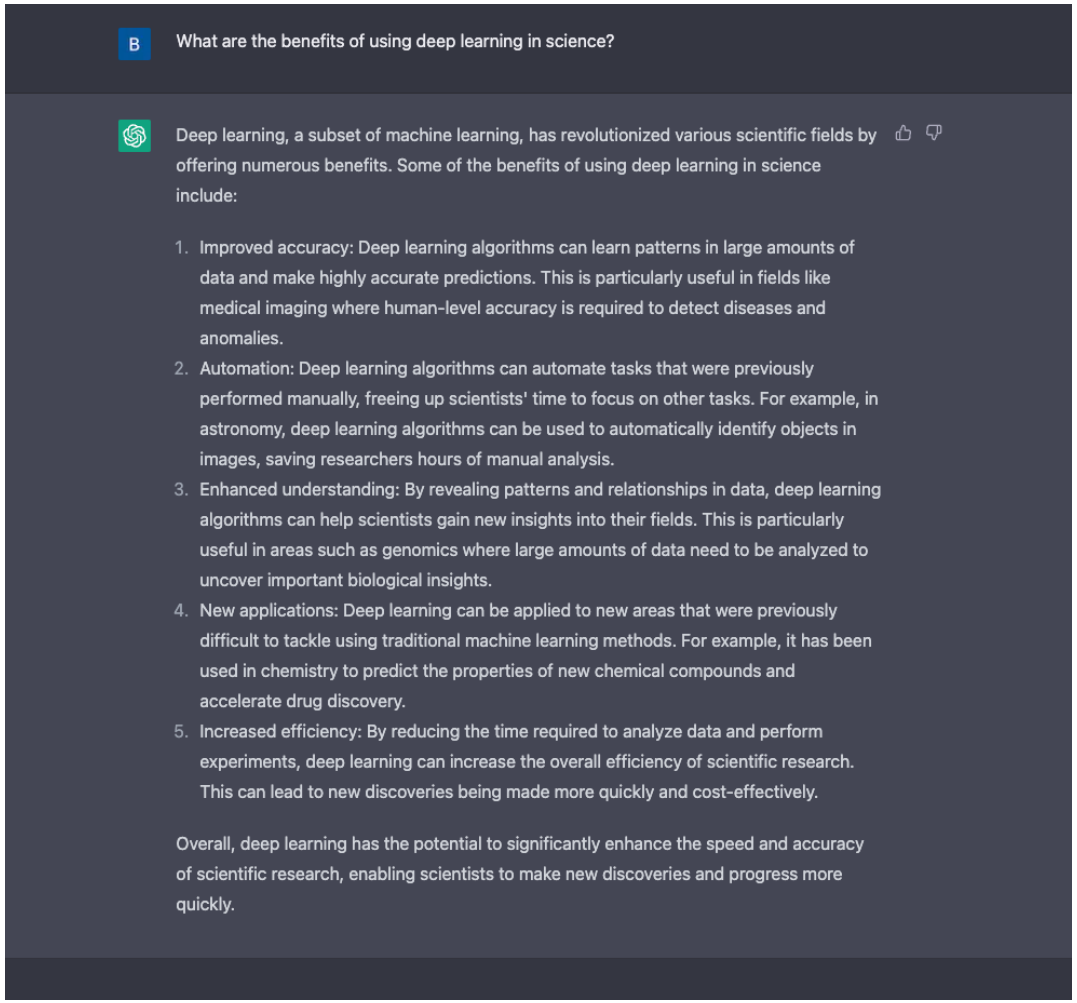
Hirano et al, An Earth-sized Planet around an M5 Dwarf Star at 22 pc, The Astronomical Journal (2023)

Anomalies can be hard to detect because:

- Large amounts of noise
- Out-of-distribution / unexplained physics
- Infrequent

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:

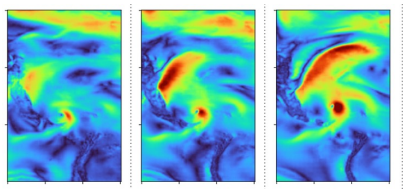
1. Improved accuracy: Deep learning algorithms can learn patterns in large amounts of data and make highly accurate predictions. This is particularly useful in fields like medical imaging where human-level accuracy is required to detect diseases and anomalies.
2. Automation: Deep learning algorithms 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.

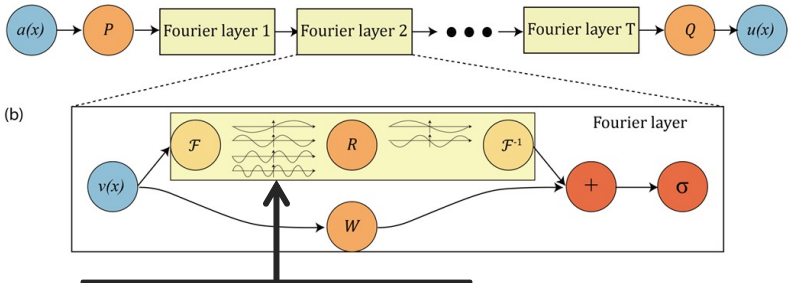
- Accuracy
- Efficiency
- Automation
- Discovery

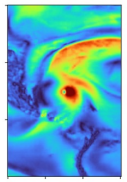
Example - simulation

Fourier Neural Operator



$t-3,\ t-2,\ t-1$



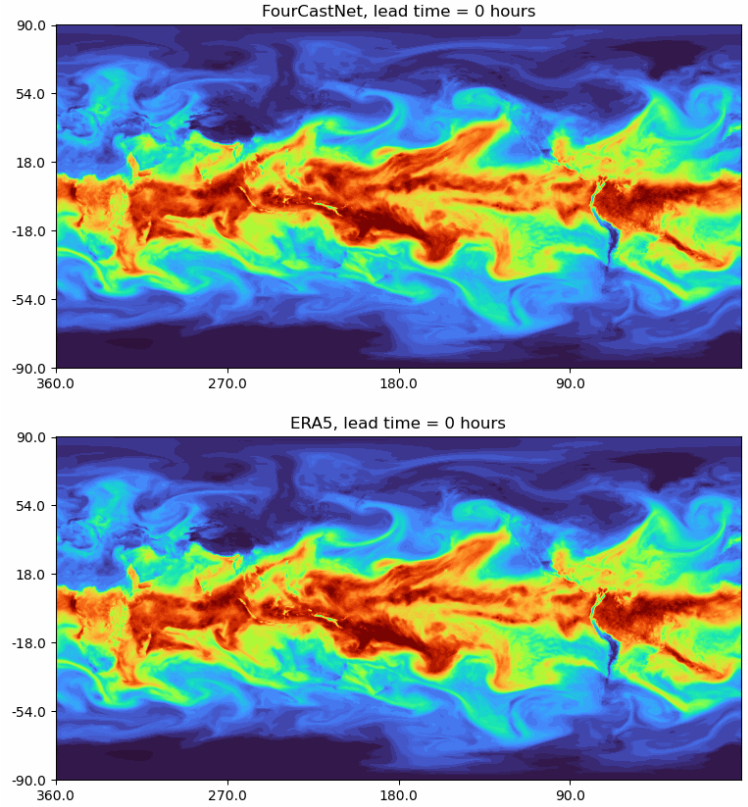


t

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

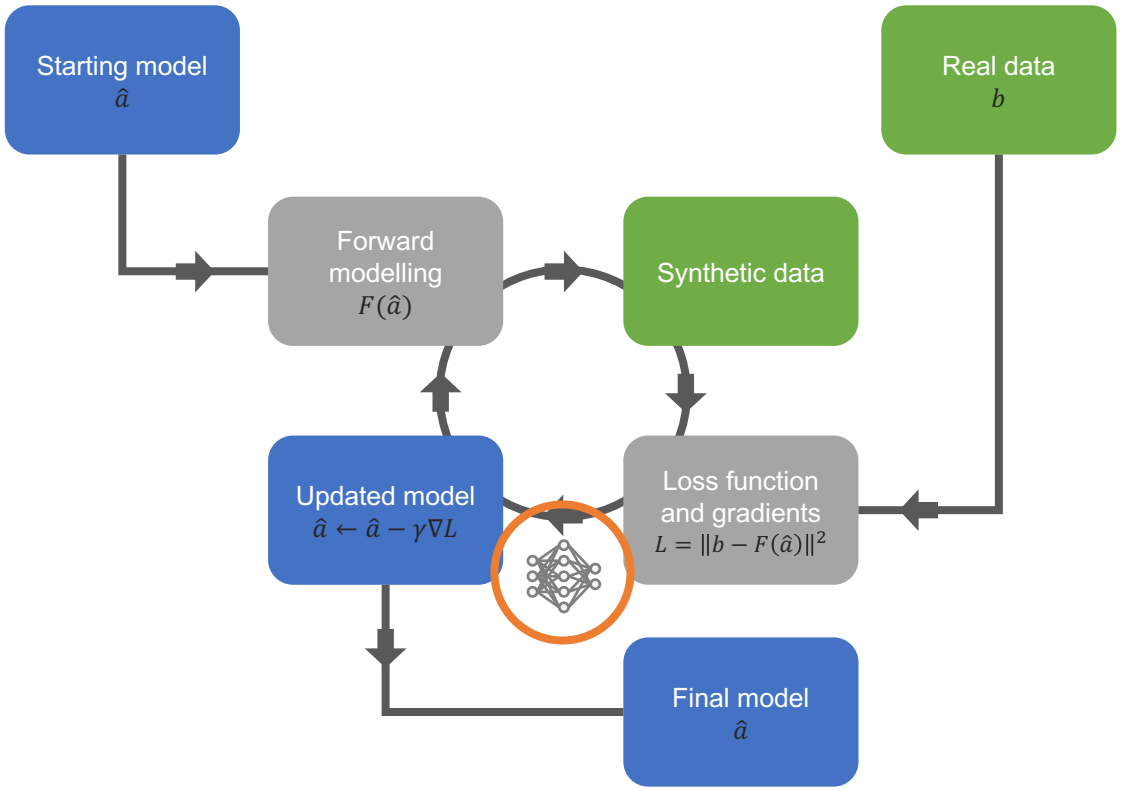


Pathak et al, FourCastNet: A Global Data-driven High-resolution Weather Model using Adaptive Fourier Neural Operators, ArXiv (2022)

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

Example - inverse problems

Learned gradient descent

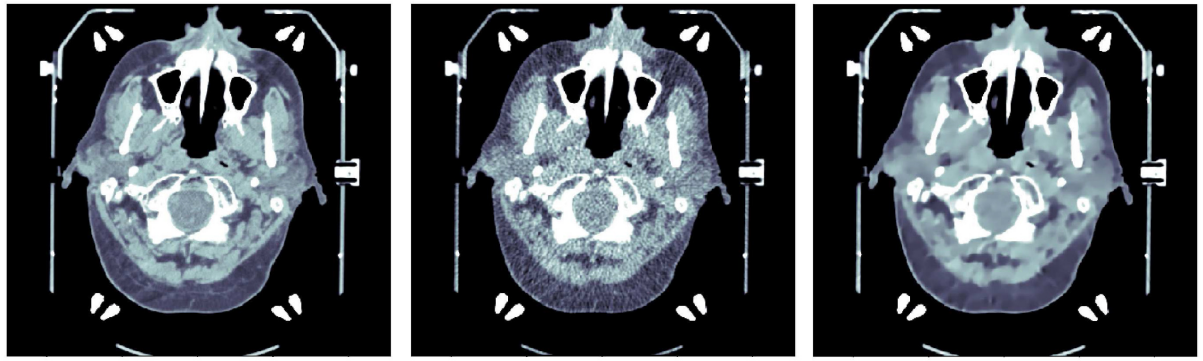


Adler et al, Solving ill-posed inverse problems using iterative deep neural networks, Inverse Problems (2017)

Idea: use a neural network to **update** gradient before descent step

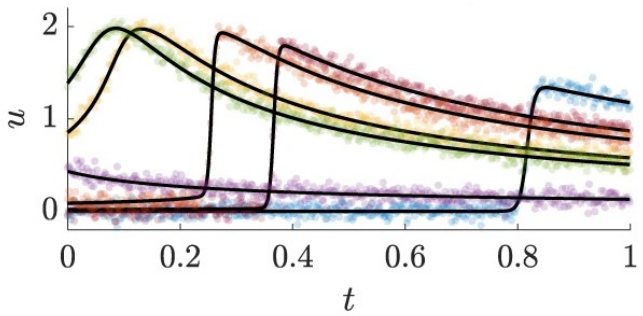
1. Initialise $\hat{a}$
2. Loop:
    1. Compute gradient, $g = \frac{\partial L(\hat{a})}{\partial \hat{a}}$
    2. Update gradient, $g \leftarrow NN(g, \hat{a}, b, \theta)$
    3. Take step, $\hat{a} \leftarrow \hat{a} - \gamma g$

Network is trained using many example inverse problems, and differentiating through entire 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: $u_t + 1.002uu_x - 0.0032u_{xx} = 0$

$$u_t = \Lambda\phi$$

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

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

1. Fit a neural network to data, $\hat{u} = NN(x,t;\theta) \approx \tilde{u}$, using supervised learning
2. 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
3. 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

Naively using deep learning for scientific tasks usually leads to:

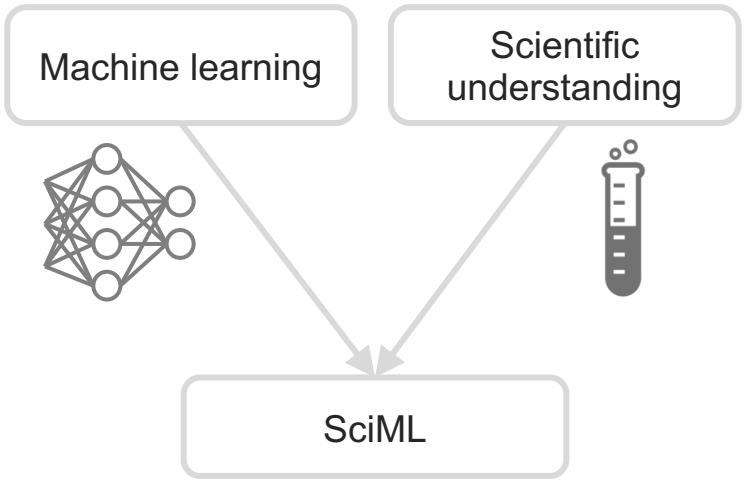
- Lack of interpretability
- Poor generalisation
- Lots of training data required

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

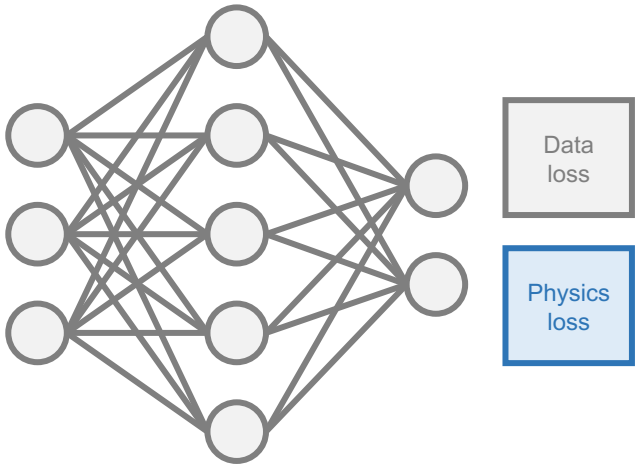
Solution



more **powerful, robust, interpretable** models

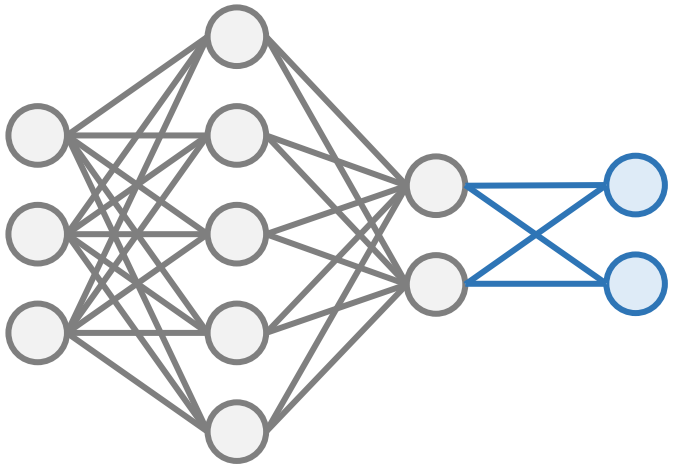
Ways to incorporate scientific principles into machine learning

Loss function



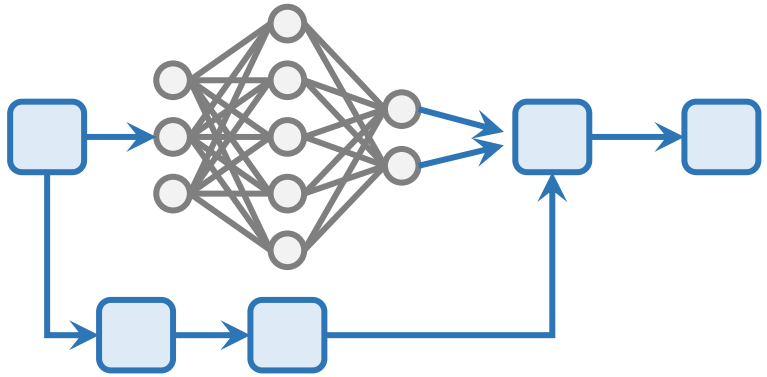
Example:
Physics-informed neural networks
(add governing equations to loss function)

Architecture



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

Hybrid approaches



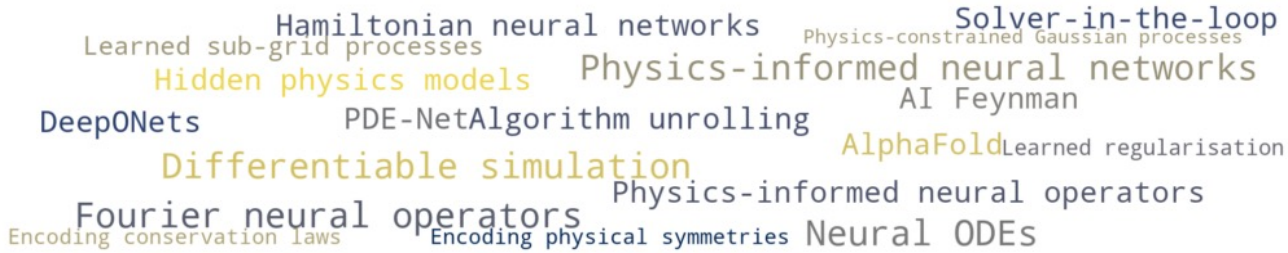
Example:
Neural differential equations
(incorporating neural networks into PDE models)

A plethora of SciML techniques

SciML technique

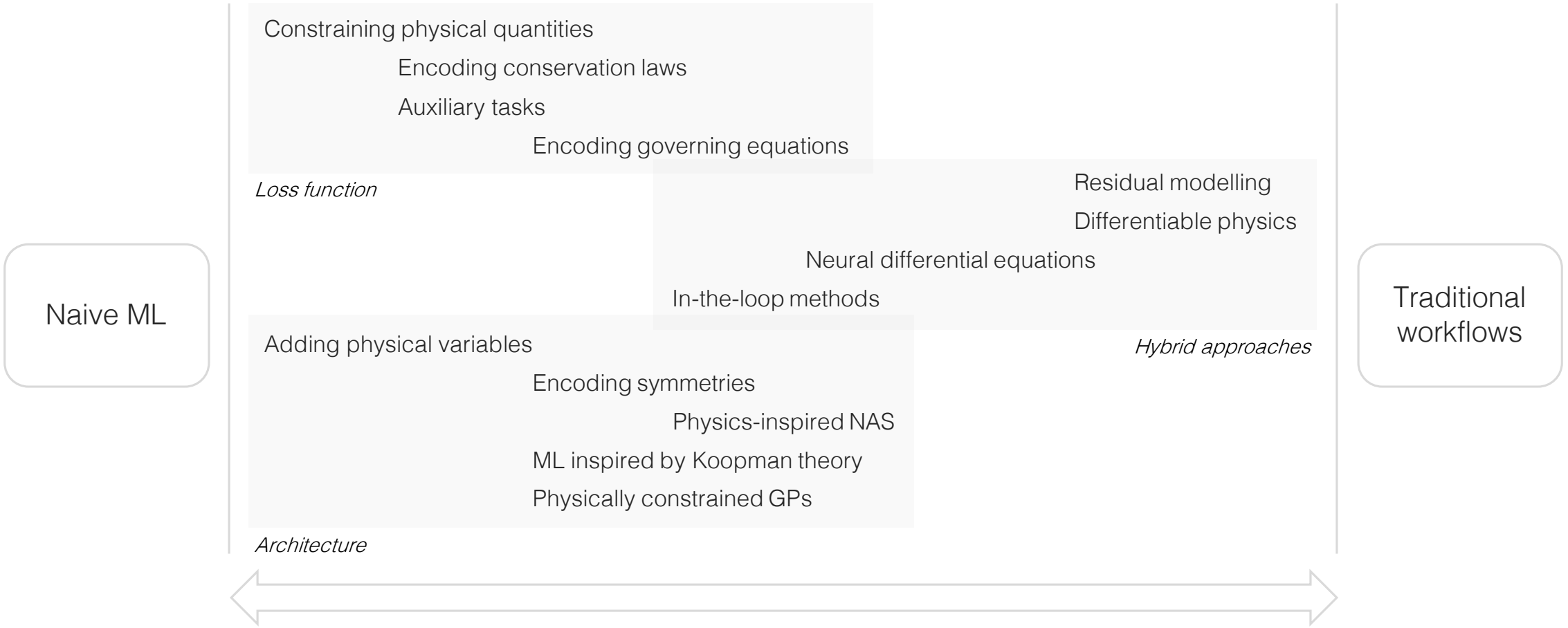
Scientific task

	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.		



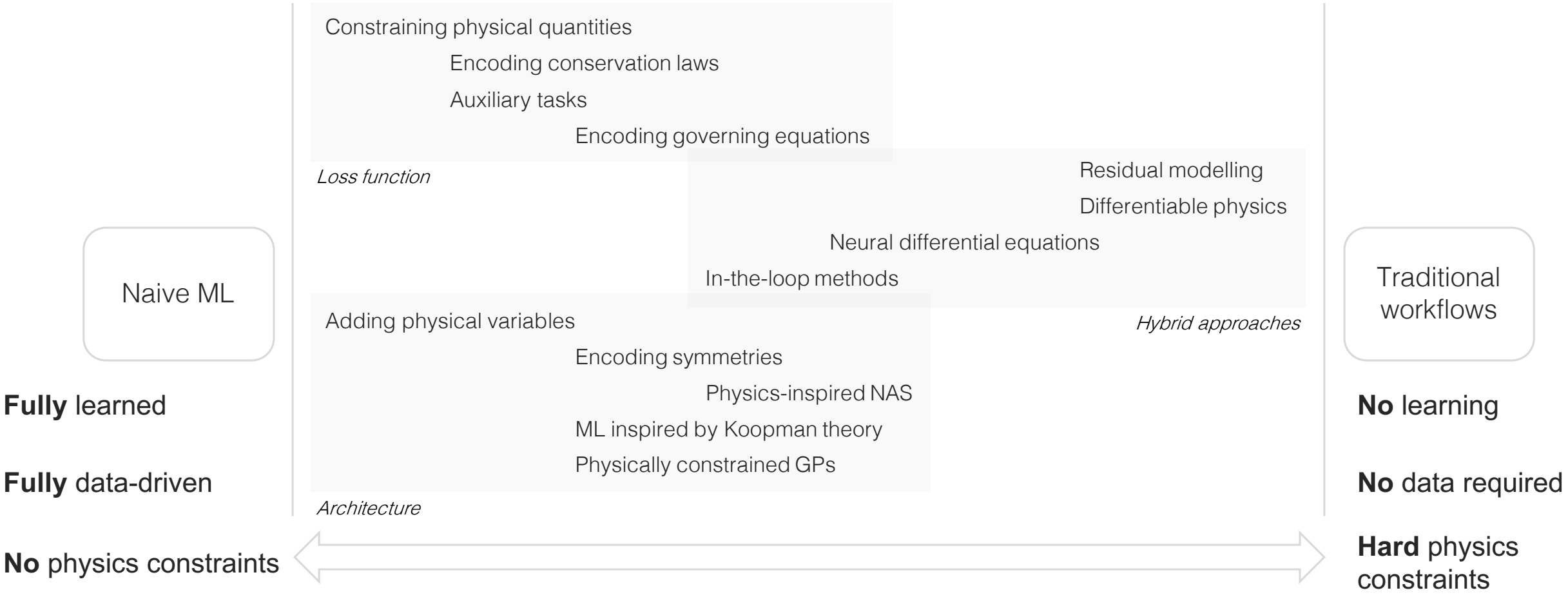
Source: B Moseley, Physics-informed machine learning: from concepts to real-world applications, PhD thesis, 2022

A plethora of SciML techniques



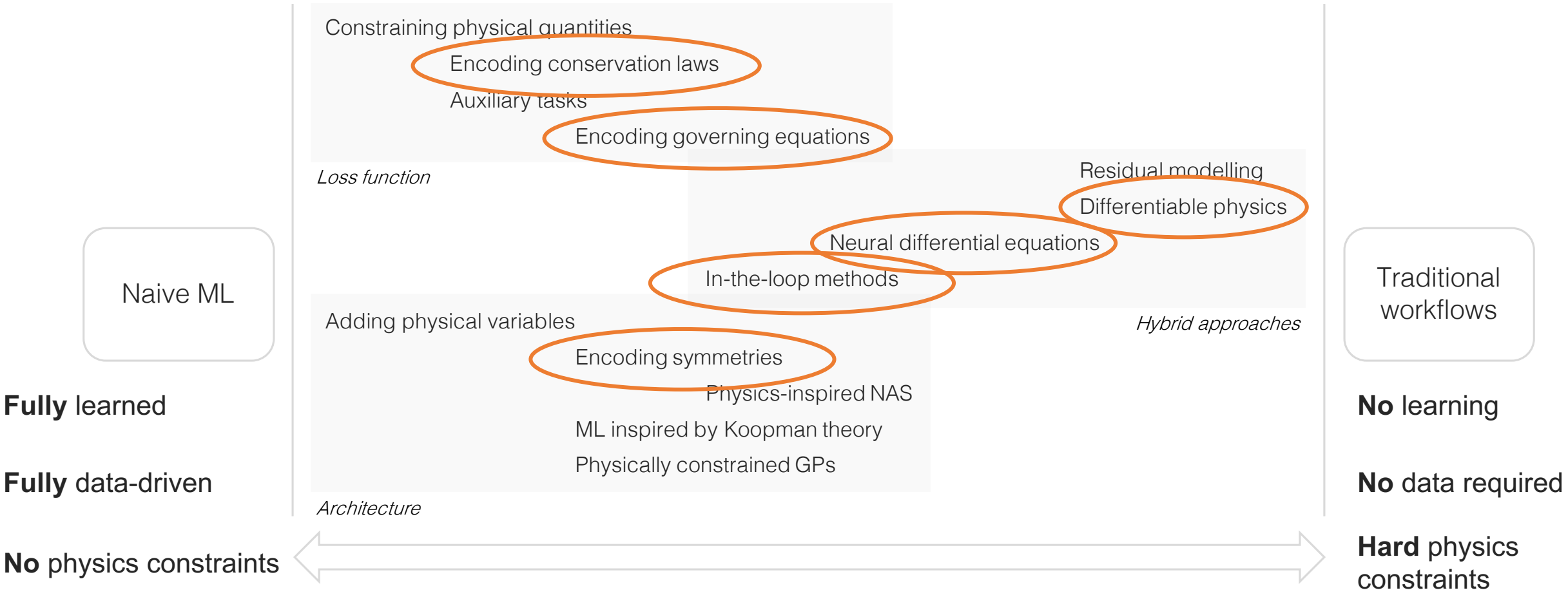
Source: B Moseley, Physics-informed machine learning: from concepts to real-world applications, PhD thesis, 2022

A plethora of SciML techniques



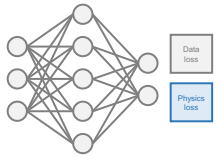
Source: B Moseley, Physics-informed machine learning: from concepts to real-world applications, PhD thesis, 2022

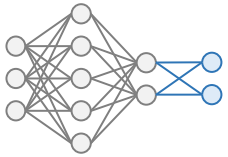
A plethora of SciML techniques



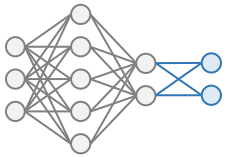
Source: B Moseley, Physics-informed machine learning: from concepts to real-world applications, PhD thesis, 2022

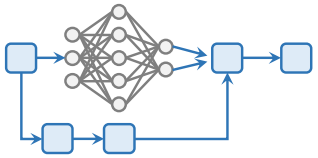
Course overview





Course introduction → Introduction to deep learning → Physics-informed neural networks → Operator learning → Transformers, graph neural networks, diffusion models, and their applications in science → Differentiable physics, neural differential equations, and equation discovery → Guest lectures: ML in chemistry and biology





Lecture summary

- AI has grown exponentially in the last decade
- It is driving many breakthroughs in **science,** and **changing** how we carry out scientific research
- SciML tightly **combines** scientific understanding with machine learning