Scientific Machine Learning - Theoretical aspects

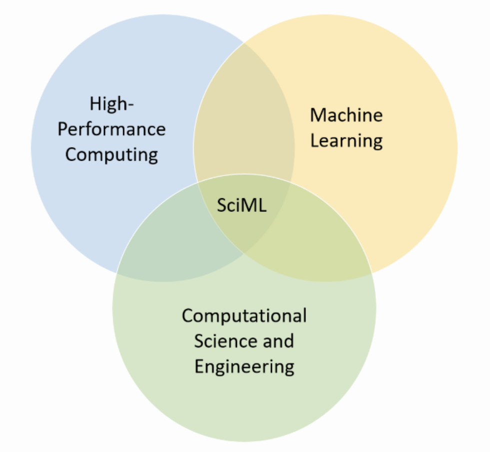
Stéphane Descombes, Stéphane Lanteri

2025-2026

Outline

Outline

Introduction to SciML



In recent years, the combination of numerical methods and machine learning has gained an ever increasing interest as a research field within Numerical Mathematics and Scientific Computing

Introduction to SciML

What is Scientific Machine Learning ?

- SciML is an emerging discipline within the data science community
- SciML seeks to address domain-specific data challenges and extract insights from scientific data sets through innovative methodological solutions
- SciML draws on tools from both Machine Learning (ML) and scientific computing to develop new methods for robust learning and data analysis
- SciML will be critical in driving the next wave of data-driven scientific discovery in the physical and engineering sciences
- SciML is multidisciplinary and leverages expertise from applied and computational mathematics, computer science, and the physical sciences

Introduction to SciML

Why Scientific Machine Learning ?

- New innovations in ML and Big Data are beginning to drive advances in scientific disciplines
- But the full potential of these techniques for data-driven discovery has yet to be fully realized
- One barrier to data-driven discovery is that existing methods often do not meet the needs of scientific users
- Application-agnostic algorithms, or those designed for more traditional ML applications such as image or natural language processing, can not typically be directly applied to scientific data sets and require non-trivial, task-specific modifications

Introduction to SciML

Why Scientific Machine Learning ?

- In many applications only limited or low-quality labels are available, while massive unlabeled (often class imbalanced) data sets are common
- Scientific data are often high-dimensional, noisy, heterogeneous, low-signal-to-noise, and multiscale
- Models should respect or incorporate physical laws, constraints, and other scientific domain knowledge
- Robust methods and an ability to quantify uncertainty are required for scientific rigor

Data-driven ML

- Machine Learning (ML) has become increasingly popular across science
- Traditionally, scientific research has revolved around theory and experiment
- One and-designs a well-defined theory and then continuously refines it using experimental data
- With rapid advances in the field of ML and the availability of increasing amounts of scientific data, data-driven approaches have become increasingly popular
- With ML, an existing theory is not required, and instead a ML algorithm can be used to analyse a scientific problem using data alone

ML for scientific problems

- But do ML algorithms actually *understand* the scientific problems they are trying to solve?
- PINNs are a powerful way of incorporating physical principles into ML

Data-driven ML

- Machine Learning (ML) has become increasingly popular across science
- Traditionally, scientific research has revolved around theory and experiment
- One and-designs a well-defined theory and then continuously refines it using experimental data
- With rapid advances in the field of ML and the availability of increasing amounts of scientific data, data-driven approaches have become increasingly popular
- With ML, an existing theory is not required, and instead a ML algorithm can be used to analyse a scientific problem using data alone

ML for scientific problems

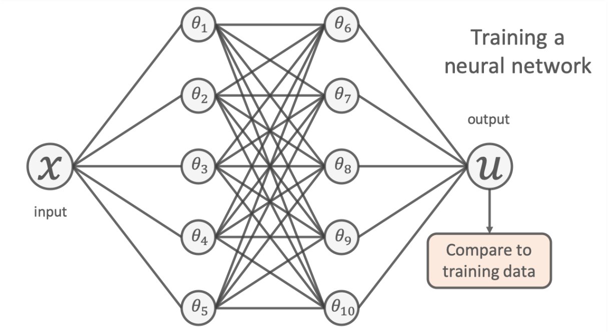
- But do ML algorithms actually *understand* the scientific problems they are trying to solve?
- PINNs are a powerful way of incorporating physical principles into ML

Learning to model experimental data

- Imagine we are given some experimental data points that come from some unknown physical phenomenon, e.g. the orange points
- A common scientific task is to find a model which is able to accurately predict new experimental measurements given this data

Learning to model experimental data

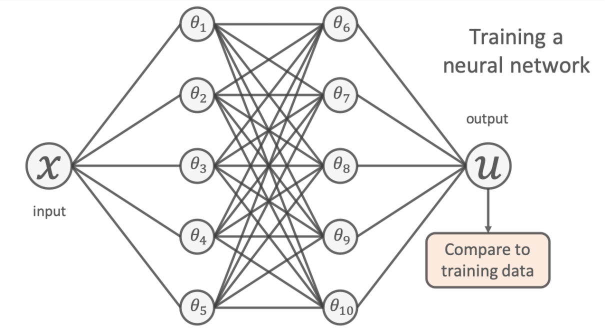
- One popular way of doing this using MLis to use a neural network (NN)
- Given the location of a data point as input (denoted x), a NN can be used to output a prediction of its value (denoted u)



Learning to model experimental data

- To learn a model, the network's free parameters (denoted by the θs) are tuned
- The goal is to have network's predictions closely match the available experimental data

$$\min \frac{1}{N} \sum_{i=1}^{N} \left(u_{\mathrm{NN}}(x_i;\theta) - u_{\mathrm{true}}(x_i) \right)^2$$



Learning to model experimental data

- The problem is, using a purely data-driven approach like this can have significant downsides
- Whilst the NN accurately models the physical process within the vicinity of the experimental data, it fails to generalise away from this training data

The rise of scientific machine learning (SciML)

- By only relying on the data, one could argue the NN has not truly *understood* the scientific problem
- What if we inform the NN with some knowledge about the physics of this process?
- For example, that the data points are actually measurements of the position of a damped harmonic oscillator

The rise of scientific machine learning (SciML)

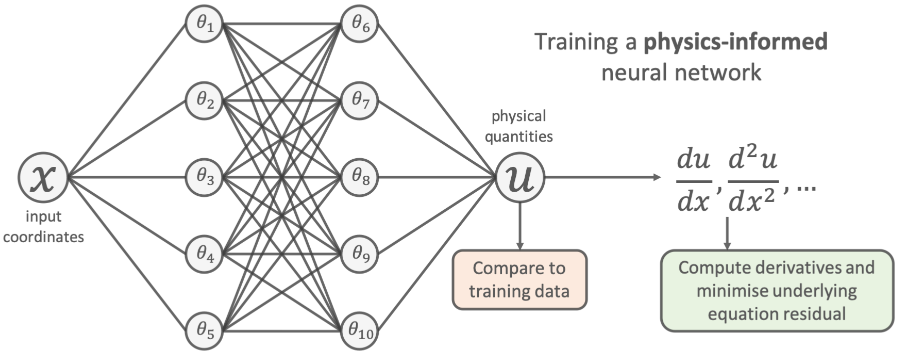
- There is a well-known differential model of this system

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

  where m is the mass of the oscillator, μ is the coefficient of friction and k is the spring constant.
- The goal is to look for ways to include this type of prior scientific knowledge into our ML workflow
- The formalism of PINNs is one way to achieve such an adapted ML workflow

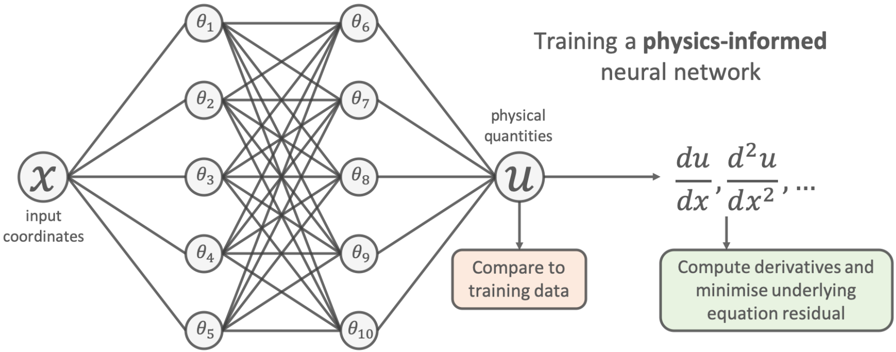
Principles

- Add the known differential equations directly into the loss function when training the neural network



Principles

1. Sample a set of input training locations $\{x_j\}$ and pass them through the network
2. Compute gradients of the network's output with respect to its input at the $\{x_j\}$s
3. Compute residual of the underlying differential equation using these gradients
4. Add residual as an extra term in the loss function



Principles

- Definition of the loss function

$$\min \frac{1}{N}\sum_{i=1}^{N}\left(u_{\mathrm{NN}}(x_i;\theta) - u_{\mathrm{true}}(x_i)\right)^2 + \frac{1}{M}\sum_{j=1}^{M}\left([m\frac{d^2}{dx^2} + \mu\frac{d}{dx} + k]u_{\mathrm{NN}}(x_j;\theta)\right)^2$$

- The additional *physics loss* tries to ensure that the solution learned by the network is consistent with the known physics

- M. Raissi, P. Perdikaris and G.E. Karniadakis
  Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations
  Journal of Computational Physics, Vol. 378, pp. 686-707 (2019)
  https://doi.org/10.1016/j.jcp.2018.10.045
- Authors introduce the concept of PINNs
- Exploit Deep Neural Networks (DNN) and leverage their well known capability as universal function approximators
- PINNs are NN that are trained to solve supervised learning tasks while respecting any given laws of physics described by general nonlinear ODEs or PDEs
- For this, differentiate NN with respect to their input coordinates and model parameters
- PINNS can be used for solving two main classes of problems: data-driven solution and data-driven discovery of PDEs
- They can deal with time-dependent problems with continuous time and discrete time PINNs

- PINNs are a type of universal function approximator that can embed the knowledge of any physical laws
- They overcome the low data availability of some biological and engineering systems that makes most state-of-the-art ML techniques lack robustness, rendering them ineffective in these scenarios
- The prior knowledge of general physical laws acts in the training of NNs as a regularization agent that limits the space of admissible solutions, increasing the correctness of the function approximation
- This way, embedding this prior information into a NN results in enhancing the information content of the available data
- This in turn facilitates the learning algorithm to capture the right solution and to generalize well even with a low amount of training examples

- Most of the physical laws that govern the dynamics of a system can be described by systems of ODEs or PDEs
- In general, these governing equations cannot be solved exactly
- Numerical methods must be used (such as finite differences, finite elements and finite volumes)
- In this setting, these governing equations must be solved while accounting for prior assumptions, linearization, and adequate time and space discretization
- Solving the governing equations of physical phenomena using DL has emerged as a new field of SciML, leveraging the universal approximation and high expressivity of neural networks
- In general, DNNs could approximate any high-dimensional function given that sufficient training data are supplied
- However, such networks do not consider the physical characteristics underlying the problem
- The level of approximation accuracy provided by these networks is still heavily dependent on careful specifications of the problem geometry as well as the initial and boundary conditions
- Without this preliminary information, the solution is not unique and may lose physical correctness

- PINNs are designed to be trained to satisfy the given training data as well as the imposed governing equations
- In this fashion, a NN can be guided with training data that do not necessarily need to be large and complete
- PINNs allow for addressing a wide range of problems in computational science and represent a pioneering technology leading to the development of new classes of numerical solvers for PDEs
- PINNs can be thought of as a meshfree alternative to traditional approaches and new data-driven approaches for model inversion and system identification
- Notably, the trained PINN network can be used for predicting the values on simulation grids of different resolutions without the need to be retrained

Outline

Poisson equation

- Dirichlet boundary conditions

$$\begin{cases} -u''(x) + c(x)u(x) = f(x), \; x \in]-1,1[, \\ \quad u(-1) = 0, \; u(1) = 0. \end{cases}$$

Exact solution for $c = 0$ and $f(x) = \pi^2 \sin(\pi x)$, $-1 \le x \le 1$,

$$u(x) = \sin(\pi x), \; -1 \le x \le 1.$$

- Dirichlet/Neumann boundary conditions

$$\begin{cases} -u''(x) + c(x)u(x) = f(x), \; x \in]-1,1[, \\ \quad u(-1) = 0, \; u'(1) = 4. \end{cases}$$

Exact solution for $c = 0$ and $f(x) = -2$, $-1 \le x \le 1$,

$$u(x) = (x+1)^2, \; -1 \le x \le 1.$$

- Dirichlet/Robin boundary conditions

$$\begin{cases} -u''(x) = f(x), \; x \in]0,1[, \\ \quad u(-1) = 0, \; u'(1) = u(1). \end{cases}$$

Same exact solution for $c = 0$ and $f(x) = -2$, $-1 \le x \le 1$.

Heat equation

$$\begin{cases} \dfrac{\partial u}{\partial t}(t,x) - \dfrac{\partial^2 u}{\partial x^2}(t,x) = 0, x \in]-1,1[,\ t > 0, \\ u(t,-1) = u(t,1) = 0, t > 0, \\ u(0,x) = u_0(x),\ x \in]-1,1[. \end{cases}$$

u_0 is the initial condition

Advection equation

$$\begin{cases} \dfrac{\partial u}{\partial t}(t,x) + a\dfrac{\partial u}{\partial x}(t,x) = 0,\ (t,x) \in \mathbb{R}^+ \times \mathbb{R} \\ u(0,x) = u_0(x),\ x \in \mathbb{R}, \end{cases}$$

u_0 is the initial condition, a is a constant, $a > 0$.

- If u_0 belongs to $C^1(\mathbb{R})$, existence and uniqueness of a classical solution u belonging to $C^1(\mathbb{R}^+ \times \mathbb{R})$

$$u(t,x) = u_0(x - at).$$

Inviscid Burgers equation / Viscid Burgers equation

$$\begin{cases} \dfrac{\partial u}{\partial t}(t,x) + u(t,x)\dfrac{\partial u}{\partial x}(t,x) = 0,\ (t,x) \in \mathbb{R}^+ \times \mathbb{R} \\ u(0,x) = u_0(x),\ x \in \mathbb{R}, \end{cases}$$

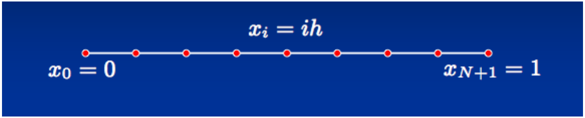
u_0 is the initial condition.

- If u_0 belongs to $C^1(\mathbb{R})$, bounded over $\mathbb{R}$ with a derivative bounded over $\mathbb{R}$, if f belongs to $C^2(\mathbb{R})$ and $f''(u_0(x))u_0'(x) \geq 0$: existence and uniqueness of a classical solution u belonging to $C^1(\mathbb{R}^+ \times \mathbb{R})$.

$$\begin{cases} \dfrac{\partial u}{\partial t}(t,x) + u(t,x)\dfrac{\partial u}{\partial x}(t,x) = \nu\dfrac{\partial^2 u}{\partial x^2}(t,x),\ (t,x) \in \mathbb{R}^+ \times \mathbb{R} \\ u(0,x) = u_0(x),\ x \in \mathbb{R}, \end{cases}$$

u_0 is the initial condition, ν is a constant, $\nu > 0$.
Hopf-Cole transformation, transform the viscid Burgers equation into a linear heat equation.

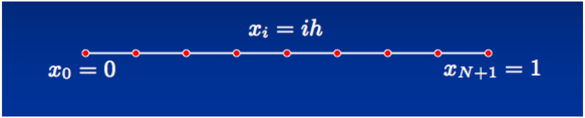
Forward Euler scheme



Forward Euler scheme

1.
$$\frac{u_i^{n+1} - u_i^n}{\Delta t} - \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{h^2} = 0,\ i = 1, \cdots, N,\ n = 0, \cdots, M-1,$$

Forward Euler scheme

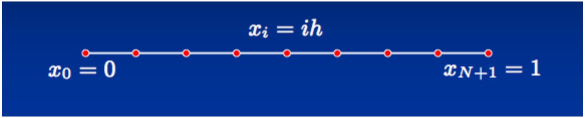


Forward Euler scheme

1.
$$\frac{u_i^{n+1} - u_i^n}{\Delta t} - \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{h^2} = 0,\ i = 1, \cdots, N,\ n = 0, \cdots, M-1,$$

2.
$$u_i^0 = u_0(x_i), i = 1, \cdots, N,$$

Forward Euler scheme



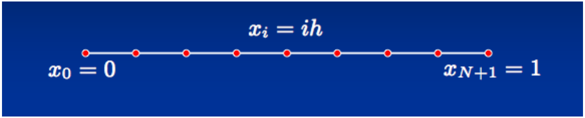
Forward Euler scheme

1.
$$\frac{u_i^{n+1} - u_i^n}{\Delta t} - \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{h^2} = 0,\ i = 1, \cdots, N,\ n = 0, \cdots, M-1,$$

2.
$$u_i^0 = u_0(x_i), i = 1, \cdots, N,$$

3.
$$u_0^n = u_{N+1}^n = 0, n = 0, \cdots, M.$$

Forward Euler scheme



Forward Euler scheme

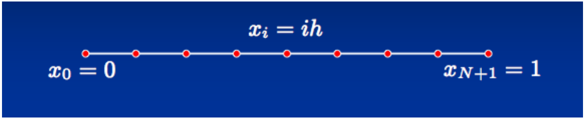
1. $$\frac{u_i^{n+1} - u_i^n}{\Delta t} - \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{h^2} = 0,\ i = 1, \cdots, N,\ n = 0, \cdots, M-1,$$

2. $$u_i^0 = u_0(x_i), i = 1, \cdots, N,$$

3. $$u_0^n = u_{N+1}^n = 0, n = 0, \cdots, M.$$

$$\lambda = \frac{\Delta t}{h^2}$$

Backward Euler scheme

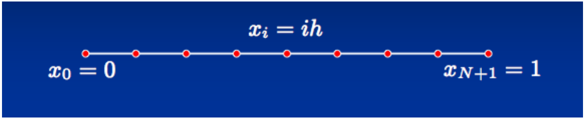


Backward Euler scheme

1

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} - \frac{u_{i+1}^{n+1} - 2u_i^{n+1} + u_{i-1}^{n+1}}{h^2} = 0,\ i = 1, \cdots, N,\ n = 1, \cdots, M-1,$$

Backward Euler scheme

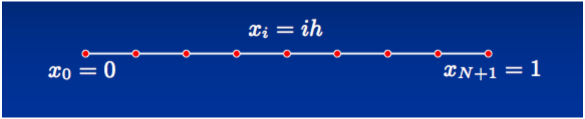


Backward Euler scheme

1. $$\frac{u_i^{n+1} - u_i^n}{\Delta t} - \frac{u_{i+1}^{n+1} - 2u_i^{n+1} + u_{i-1}^{n+1}}{h^2} = 0,\ i = 1, \cdots, N,\ n = 1, \cdots, M-1,$$

2. $$u_i^0 = u_0(x_i), i = 1, \cdots, N,$$

Backward Euler scheme



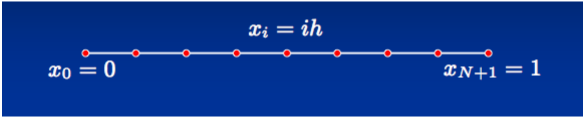
Backward Euler scheme

1.
$$\frac{u_i^{n+1} - u_i^n}{\Delta t} - \frac{u_{i+1}^{n+1} - 2u_i^{n+1} + u_{i-1}^{n+1}}{h^2} = 0,\ i = 1, \cdots, N,\ n = 1, \cdots, M-1,$$

2.
$$u_i^0 = u_0(x_i), i = 1, \cdots, N,$$

3.
$$u_0^n = u_{N+1}^n = 0, n = 0, \cdots, M.$$

Upwind scheme / Lax-Wendroff scheme



1

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} + a\frac{u_j^n - u_{j-1}^n}{h} = 0,$$

2

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} + a\frac{u_{j+1}^{n+1} - u_{j-1}^{n+1}}{2h} - \frac{a^2\Delta t}{2}\frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{h^2} = 0.$$

L^2 stability analysis

1. Let u be a 1-periodic function, the Fourier coefficients $\hat{u}(k)$, $k \in \mathbb{Z}$ are complex numbers defined by the integrals

$$\hat{u}(k) = \int_0^1 u(x) e^{-2\pi ikx} dx.$$

2. Parseval formula

$$\|u\|_2^2 = \sum_{k \in \mathbb{Z}} |\hat{u}(k)|^2.$$

3. Von Neumann analysis, there exists a constant $K > 0$ independent of Δt and h such that for all n, $1 \leq n \leq M$,

$$\|u^n\|_2 = \left(\sum_{j=1}^{N} h |u_j^n|^2 \right)^{1/2} \leq K \|u^0\|_2 = \left(\sum_{j=1}^{N} h |u_j^0|^2 \right)^{1/2}.$$

4. Stability, consistency, convergence - Lax-Richtmyer theorem.

Outline

Back to the future

Let n in $\mathbb{N}^*$, σ a function from $\mathbb{R}$ to $\mathbb{R}$, $M([0,1]^n)$ the space of finite, signed regular Borel measures and $C([0,1]^n)$ the space of continuous functions on $[0,1]^n$.

Definition.

We say that σ is **sigmoidal** if

$$\lim_{t \to -\infty} \sigma(t) = 0, \lim_{t \to +\infty} \sigma(t) = 1,$$

we say that σ is **discriminatory** if for a measure μ in $M([0,1]^n)$

$$\int_{[0,1]^n} \sigma\left(y^t x + \theta\right) d\mu(x) = 0$$

for all y in $\mathbb{R}^n$ and θ in $\mathbb{R}$ implies that $\mu = 0$. □

A sigmoidal function can not be a polynomial function.

Back to the future

Let n in $\mathbb{N}^*$, σ a function from $\mathbb{R}$ to $\mathbb{R}$, $M([0,1]^n)$ the space of finite, signed regular Borel measures and $C([0,1]^n)$ the space of continuous functions on $[0,1]^n$.

Theorem 1, Cybenko (1989).

Let σ be any continuous discriminatory function. The finite sums of the form

$$G(x) = \sum_{j=1}^{N} \alpha_j \sigma\left(y_j^t x + \theta_j\right)$$

are dense in $C([0,1]^n)$. □

Lemma 1, Cybenko (1989).

Any bounded, measurable sigmoidal function, σ, is discriminatory. In particular, any continuous sigmoidal function is discriminatory. □

Approximation by ReLU Neural Networks

$$g(x) = \begin{cases} 2x, & 0 \le x \le 1/2, \\ 2(1-x), & 1/2 < x < 1. \end{cases}$$