International Journal of Research in Computer Science & Information Technology (IJRCSIT)
ISSN: XXXX-XXXX/ Volume 1 Issue 2 November – December 2026
Paper Id: IJRCSIT-V1I2P105/ Page No: 71-89

International Journal of Research in Computer Science & Information Technology y

Original Article

Physics-Informed Neural Networks for Solving Nonlinear Partial Differential Equations

Susana Shree[1], Satya Gurunathan[2], Sauna Kumari[3]

[1,2,3] UG Scholars, G.T.N Arts College, Din Digul, Tamilnadu, India.

Received Date: 22 November 2026 Revised Date: 26 November 2026 Accepted Date: 30 November 2026

Abstract

In the field of scientific computing, recent work has studied Physics Informed Neural Networks (PINNs) as a novel computational paradigm for weakly supervised learning-based methods to solve nonlinear partial differential equations (PDEs), bridging progress in classical numerical analysis with recent advances in general machine learning. Traditional numerical methods such as the Finite Difference Method (FDM), the Finite Element Method (FEM) and spectral approaches, which have been around for decades to numerically solve partial differential equations modelling a physical system. However, their performance suffers from limitations such as mesh dependency and high computational cost in higher dimensional domains, and they struggle to represent complicated geometries or sparse data scenarios. Whereas, PINNs exploit the Representational power of Deep Neural Networks (DNNs) to approximate a continuous solution along with physical laws as differential equation constraints that embedded directly in the learning Processions centre around the idea of embedding governing nonlinear PDEs into the loss function of a deep learning model. The network is trained not on labelled data but by minimizing the residuals of the PDE, plus boundary and initial condition constraints. This enables the model to recover physically consistent solutions, even in data-poor or noisy regimes. Moreover, a mesh-free operation feature enables PINNs to address irregular domains and high-dimensions problems fitting the modern computational science missions well.

Nonlinear Partial Differential Equations (PDEs) arise from numerous natural and engineering sciences, such as fluid mechanics, quantum physics and heat transfer; their innate complexities include nonlinearity behaviour together with sensitivity to initial and boundary conditions. This problem is formulated in terms of optimizing generalised versions of deep neural networks (DNN) — such universal function approximates lead to a class of methods called physics-informed neural networks (PINNs). PINNs can take advantage of automatic differentiation to calculate the derivatives required for direct calculation of PDE residuals, making them not in need of a specific discretization scheme. Although PINNs have many potential features, there are a few capabilities they lack. However, problems such as training instability, spectral bias, slow convergence and sensitivity to hyper parameters are under-studied. Alternatively, Even when in the same problem type that is required to minimize multiple loss components; PDE residuals and boundary conditions, arise many issues to serial tough problems. Recent advances, including adaptive loss weighting, domain decomposition strategies, and hybrid modelling using data-driven and physics-based approaches are beginning to address the limitations of current practice.

In this research paper, we provide a review of Physics-Informed Neural Networks for solving nonlinear PDEs in a detailed and high-level way. Fundamentals of nonlinear PDEs, mathematical foundations for PINNs, and optimization dynamics; advances in architecture and comparisons to traditional numeric solvers Moreover, it focuses on real-world applications and discusses current advances and research trends. In this work we aim to bridge the gap in leveraging our understanding into PINNs for scientific computing based on the insights in functional analysis, deep learning theory and computational physics [11].

Keywords

Physics-Informed Neural Networks, Nonlinear Partial Differential Equations, Deep Learning, Scientific Machine Learning, Automatic Differentiation, Variation Methods, Neural Operators, Computational Physics

Introduction to Nonlinear Partial Differential Equations And Pins

Nonlinear Partial Differential Equations (PDEs) provide a mathematical framework to describe many complex systems in physics and engineering. Nonlinear PDEs, on the other hand, include turbulence & chaos and

This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/2.0/)

shock waves & continuously differentiated patterns. More often they cannot be superimposed on self or solved directly in an analytical manner as linear ones do. These equations emerged naturally from real world phenomena like Navies–Stokes equations for fluid flow, diffusion laws of non-linear media of heat transfer and wave propagation population dynamics and quantum mechanics. Additional complexity arises from the nonlinear nature of many systems, leading to a scarcity of exact solutions while most are approximate at best or require numerical methods.

Albeit An entire set of the solutions to that PDE in some discretized space can be approximated Numerical methods for obtain the solution of particular PDS which becomes a bit heaven since it depends on many different approaches. These method are mathematically solid and extremely popular, however they depend on discretization's in the sense that they will mesh our domain into grids. This discretization is limited in many aspects, particularly related to high-dimensional problems, irregular geometries and multi-scale phenomena. The computational cost increases exponentially with dimensionality, a well-known issue termed the "curse of dimensionality". Besides, obtaining valuable mesh for complex domains may also be time-consuming and error-prone.

Deep learning has recently made a lot of progress, due to the paradigm shift in scientific computing over the years. The neural networks which have been shown to be great function approximates has accumulated by essentially scaling or adding complexity to model things or tasks that are not naturally associated with data. This, in turn, has inspired a new scientific interdisciplinary field called — Scientific Machine Learning (Skim), where the basic idea is to use machine learning methods to complement physical models that solve scientific problems. PINNs are a new approach for addressing PDEs, in particular nonlinear PDEs, and within this context these models have been highly popular in the literature.

It is also worth mentioning that the Physics-Informed Neural Networks (PINN) are really different in a way they literally contains physics laws during training from scratch unlike normal neural networks. Unlike conventional approaches which mainly rely on large datasets, PINNs correctly embed the governing equations of the system (usually formulated in PDEs) into a neural network loss function. This is implemented through automatic differentiation that calculates the gradient of network output with respect to its inputs. So, to not go too wild here but the net will learn a formula that fits all the data you have and also obeys the physical laws.

The fundamental idea of the PINNs approach is to build a deep neural network that represents an approximate solution of PDE, where we define mappings from input variables (e.g., space and time coordinates) to output variables (e.g., temperature, velocity or pressure). To train the network, a loss function consisting of the PDE residual and boundary and initial condition terms is minimized. This turns the problem of solving a PDE which is an optimization to yield the parameters of a network that minimizes violations of physical laws.

Among diverse remarkable merits of PINNs, a finite-free characteristic is that. On the contrary to recent numerical methods, PINNs do not need explicit discretization of the domain. Randomly selected collocation points are then used to realize the evaluation of the PDE residual inside the domain instead. This gives them a great deal of flexibility and makes it able to cope with complex geometries and high-dimensional problems correctly. PINNs naturally comes with the added bonus that, when the number of reliable experimental or observational data is small, PINNs can also work well on sparse and/or non-noisy datasets.

However, despite their advantages, there are some challenges with PINNs. Training a deep neural network to satisfy non-obvious nonlinear PDEs is tricky and may suffer from vanishing gradients, unequal importance of various components of the loss, as well as slow convergence. Moreover, the spectral bias of neural networks (they learn low frequencies faster than high frequencies) limits them to approximate sharply changing gradients or discontinuities in the solution. The solution to countering these issues involves careful selection of our network architecture, loss function for the task and finally our optimization strategy.

Combining data-driven and physics-based modelling, this is also a great feature of PINNs. The time series forecasting methods are state-of-the-art based on recurrent layers, or convolutional layer like CNN. Usefully, PINNs supply a hybrid between both paradigms and to do accurate robust modelling for complex systems. That has created new fronts as around, digital twins, inverse problems and real-time simulation.

The significance of using PINNs to study nonlinear PDEs lies in its promising capacity to revolutionise computational science. With more advanced Computational and machine learning techniques, it is predicted that PINNs will be one of the most visible point arrival for obtaining solution to problems until this date considered intractable. They turn into this flexible device in the eventual fate of logical figuring; since they can take care of high-dimensional spaces, consolidate different data sources and convey arrangements nonstop.

Table 1: Linear versus nonlinear PDEs

DRO Type	Definition Method	Key Idea	Advantages	Limitations
Wasserstein DRO	Distance-based	Uses Optimal Transport distance to define an uncertainty ball around the training distribution.	Provides strong, mathematically rigorous robustness; highly interpretable.	Extremely high computational cost for large-scale datasets.
f-divergence DRO	Divergence-based	Measures the "distance" between distributions using metrics like KL-divergence or χ^2-divergence.	More efficient optimization compared to Wasserstein methods.	Highly sensitive to the choice of the specific divergence function used.
Moment-based DRO	Constraint-based	Constrains the model based on statistical moments (e.g., matching mean and variance).	Features a simpler mathematical formulation and is easier to solve.	Limited expressiveness; can fail to capture complex distribution shifts.
Group DRO	Group-based	Explicitly focuses on minimizing the loss of the worst-performing subgroup.	Significantly enhances fairness and performance for minority groups.	Requires explicit, high-quality group labels (e.g., race, gender, or sensor ID).
Regularized DRO	Dual formulation	Reformulates the robustness problem as a standard loss with a specific regularization term.	Much easier to implement in existing deep learning frameworks.	May lose the clear physical interpretability of the distribution shift.

The Nonlinear Partial Differential Equations (PDEs) is one of the main support pillars of modern mathematics and mathematical physics, presenting a general theoretical apparatus to describe systems with complex interdependent interaction processes which produce non-additively. Nonlinear PDEs owe their richness to dependencies that violate the principle of superposition, which is crucial for analysing linear PDEs (often trivially solvable) and carrying out constructions for them easily. In order to be aware of the challenges and potential issues related to its numerical treatment using contemporary computational mechanisms (e.g. Physics-Informed Neural Networks [PINNs]), one should gain some insight into these theoretical aspects of nonlinear PDEs prior to their use for modelling biological patterns.

More formally, define a nonlinear partial differential equation as a partial differential equation in which the dependent variable or its derivatives appear in non-linear form. This non-linearity is either a component of the unknown function and its derivatives, a power of the function or perhaps other nonlinear operators. Famous models include Navies–Stokes equations from hydrodynamics, the nonlinear Schrödinger equation from quantum mechanics or reaction–diffusion systems in biology. These appear in essentially every problem with an underlying conservation law, in transport phenomena and many other types of dynamic systems where feedback is important. A little more than just classical calculus, these equations can be solved via mathematical methods. It involves tools from functional analysis, topology and variation approaches to the problem.

Well-pawedness, or what one can say that by Hadamard almost made it the formalism, is one of the corner stones of nonlinear PDE theory. In mathematics, a problem is termed well-posed if it has three properties: existence of a solution uniqueness of that solution continuous dependence on the initial or boundary conditions. For nonlinear PDEs, the issue of well-pawedness is much harder than for linear equations. Usually the solutions at best only locally in time (there cannot be defined globally), sometimes it is not unique, and other times it is sensitive to initial conditions (either chaos). The need for this complication of analysis forces one to define more general solution concepts (weak solutions, viscosity solutions, distributional solutions) which relax the differentiability assumptions typical in classical analysis and which are useful in dealing with discontinuities or singularities.

An important field that gives the right mathematical context to nonlinear PDEs is functional analysis. Instead of finite-dimensional spaces, you work in infinite-dimensional function spaces like Sable spaces which take both functions and their derivative (when summing over the derivatives), the additional constraint that a differential operator has been applied is represented as an integral. This makes it possible for PDEs to be in the weak/variation form (i.e. the equation is solved at least on average or at the level of integral-type relation). This important for the presence of existence results by using Lax–Mailgram theorem, compactness argument and Galleria method to treat irregular (weak) outcomes. The variation formulation also acts as a more natural link to advanced machine learning

techniques, with one extreme being the interpretation of loss functions in the limit of perfect approximates as minimization of residuals in function space (which is a direction pursued currently within PINN research)

Another theoretical aspect is the classification of nonlinear PDEs according to structural properties. As with linear PDEs, nonlinear equations can be elliptic, parabolic or hyperbolic and hence this classification requires a special solution owing to their non-linearity. Elliptic equations nonlinear Poisson (stationary and smooth under appropriate assumptions on coupling coefficients) are an example of such last. If a heat equation is smooth then it can be called parabolic. Conservative laws and wave equations are strongly hyperbolic, meaning that solutions propagate at speeds bounded above finite speeds, and exhibit shock behaviour from [[smooth]] initial data. Each of these classes presents distinct analytic challenges and requires the creation of dedicated theoretical and numerical approaches in the context.

Complex solutions that do not arise in linear systems are caused by nonlinearity. In the special case of finite time}, they possess singularities (e.g. blow-up), or multiple stable versus unstable equilibrium. One of the least understood and perhaps most challenging problems in fluid dynamics is turbulence caused by nonlinear interactions between velocity fields. It can also be found in biological systems embodying the animal coat patterns or chemical oscillations, through nonlinear terms spontaneous pattern formation in reaction–diffusion systems. Since practically all real-world phenomena are nonlinear, it is only natural that being an inherently restricted kind of behaviour representation and approach to achieving accurate and stable solutions inherent weak points will become evident.

Numerically, nonlinear PDEs are much harder, because they are very sensitive and complicated. Many classical numerical methods employ discretisation schemes that approximate derivatives in terms of finite differences, elements or spectral expansions. Research aimed at orienting the mathematical behaviour of these methods is still on-going, and they can suffer from poor stability, convergence and/or computational efficiency (for example in higher dimensional settings or with strongly nonlinear terms). Which means we almost always have to use iterative solvers and, as such, no guarantees of convergence. Also, to fit this in local information, sharp gradients or discontinuities need very fine meshes that can increase the cost of resolution [6].

In recent years, new alternatives focused on optimization and machine learning have begun to complement classical ones. The nonlinear PDE theory provides a natural setting for optimization-based formulations where the solution corresponds to minimizing energy functional / residual. So for example, this perspective is especially valid when the PDE is embedded into a loss function and solved in a gradient based deterministic optimization way (this precisely what PINNs do). And hence this requires for such methods to be sensitive to the underlying properties of PDEs — i.e. smoothness, stability and structure of solution space.

Conservation laws and symmetries in nonlinear PDEsOther important theoretical aspect. We have conservation laws that are valid for many other physical systems, i.e. conservation of mass / momentum / energy etc. They also restrict solution space, and can be used to derive invariants or reduce the complexity of a problem. Symmetry analysis (typically via Lie group method [2]) provides powerful methods for PDE reduction, invariant solutions construction and solution qualitative feature interpretations. Encoding these properties into computational modelling of the systems through examples such as physics-informed neural networks (PINNs) increases interpretability and accuracy.

This tradition of interaction between theory and computation is particularly relevant in the context of modern scientific machine learning. Deep learning models are able to represent complex mappings from data to output than conventional models but they cannot be guaranteed to work, without a strong theoretical underpinning. This requires careful consideration with regards to the strategy choices like function space approximation, stability and convergence so that the solutions learned are physically reasonable and mathematically valid. With this growing research space, there is a stronger emphasis on establishing solid theoretical guarantees of generalization for end-to-end solvers based on machine learning; effectively bridging classical analysis together with data-driven approaches.

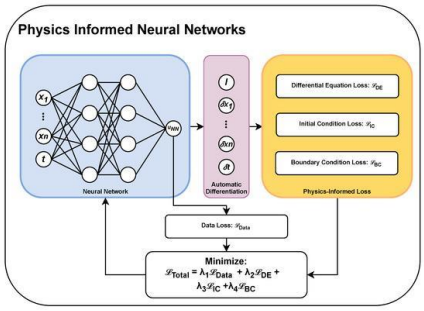


Figure 2: PINN Architecture (Neural Network + Physics Constraints)

The Universal Approximation Theorem Of Deep Neural Networks

They are approximates of very complicated, high-dimensional functions and have been awesome to the centre of contemporary machine learning research. This falls under the solution of a nonlinear PDE because if we take the solution of PDE, we can consider it as an function which maps the input space from spatial and temporal to physical quantities e.g. temperature, pressure or velocity respectively.) Unlike conventional numerical methods that discretize the solution, DNNs learn a continuous representation of the solution directly from data and governing equations. This property suits them well to mesh less methods such as Physics-Informed Neural Networks (PINNs) where a neural network is used to implicitly define the solution function.

A. Universal Approximation Theorem

Universal Approximation Theorem Neural networks as function approximates The universal approximation theorem which states that, given certain conditions on the activation function, a feed forward neural network with at least one hidden layer can approximate any continuous function defined on a compact subset of R n to arbitrary accuracy (30). Moreover, while this is a fairly strong result, it remains existential: it shows existence of such a network but not how to recover one or best size for the net.

According to the theory, however deeper networks with many hidden layers outperform shallow networks because they can represent complex functions more efficiently but in practice one usually tends to work with deep networks that have many hidden layers instead of shallow networks. Because we know from previous work that hierarchical structures and compositional patterns are ubiquitous in functions — including solutions to nonlinear PDEs — neural networks treat depth as a key feature (taken directly from the left) to account for this. This is particularly important in modelling multi-scale phenomena as each different layer can learn features at different scales.

B. Expressiveness of Deep Neural Networks

Expressivity: Expressivity is an important property of a neural network in that how well the network can express large numbers of functions. Deep networks have potentially exponentially more binary regions than shallow networks. It shows that having a deep network can represent highly nonlinear complicated functions with fewer parameters than that of a shallow network if at least one achieves comparable approximation level.

Expressiveness is needed for nonlinear PDEs since the solutions tend to be steep, discontinuous or oscillatory. With neural networks is possible for learn long-term pattern which have periodicity in the data — activation function (Relook) or tan+ + sinusoidal activations. Further developments have also been made in domain-specific architectures and activation functions designed for the kind of problems picked up within scientific computing, taking the expressiveness of neural networks even further [23].

C. Spectral bias and the frequency learning behaviour

The spectral bias (also called temporal bias) — is one of the key properties of neural networks. More precisely, this phenomenon of neural networks is trained on low-frequency components of a function more quickly than higher frequency ones. While this property is advantageous for demonising, it creates problems when approximating solutions with fine features or rapid oscillations as occurs frequently in nonlinear PDEs.

High-frequency details must be resolved in, e.g., shock wave or turbulence problems. Yet, a NN trained with low spectral bias can fit here but provides instead a simple approximation which smoothest over the important features. To address this problem, approaches like Fourier feature mappings, sinusoidal activation functions and some adaptive sampling (focus on regions with high error) have been proposed.

D. Neural Tangent Kernel (NTK) Perspective

MD, or the Neural Tangent Kernel (NTK), can mathematically understand. In the very large-width limit, we can show that a neural-network is a kernel method whose output has partially fixed kernel throughout training.

This view makes it simpler to apply the well-studied tools from kernel theory to understand things such as convergence properties and generalization behaviour.

The NTK framework gives explanation against questions such as why some PDE problems are easier to solve than others with respect to PINNs. It gives us clues about the problems of training stiffness, in which different parts of the loss functions converge at different rates. To summarize, the knowledge of what causes the NTK behaviour can be used for constructing better architectures and optimization strategies which boost performance in terms of accuracy, memory, speed or general efficiency when solving nonlinear PDEs.

E. Role of Activation Functions

Activation functions play an important role in specifying the approximation property of neural networks. There are lots of activation functions Sigmoid, tan and Relook to name a few, we have their advantages and disadvantages. Since smooth activation functions like tan produces a smooth output, higher differential orders can be computed more stable using automatic differentiation which makes them a better choice for PDE related tasks.

SIREN (Sinusoidal Representation Networks) is an architecture where both the mapping of features and the activation periodicity are sine, which currently works great for approximating high-frequency signals. Such activation functions mitigate spectral bias, so that the network learns a more detailed level of PDE solutions.

F. Depth vs. Width Trade-offs

The architecture of the neural network (the number of layers, neurons per layer etc.) is an important determinant in how well this performs. Intuitively, this is because increasing width provides an ability to approximate better but increasing depth helps in representation more efficiently which has been empirically shown. Deep networks can reuse some features from layer to layer, so the set of more compact layers can model far more complex functions with fewer parameters.

But the deeper networks are more difficult to train (the problem of vanishing or exploding gradients). The challenges that we can face with this kind of approaches, most researchers use techniques such as normalization, residual connections and initialization–normalization. It is all about balancing out depth and width

Table -4: Properties of neural networks for pie approximation

Property	Description	Impact on PINNs
Universal Approximation	The capability of a neural network to approximate any continuous function given sufficient neurons.	Provides the foundational justification that a network can, in theory, represent the solution to a PDE.
Expressivity	A measure of the network's ability to model intricate and highly complex mathematical functions.	Critical for capturing the steep gradients and nonlinear behaviors common in fluid dynamics or shock waves.
Spectral Bias	The tendency of neural networks to prioritize learning low-frequency components before high-frequency ones.	Often acts as a bottleneck, limiting the model's ability to learn sharp features, discontinuities, or fine-scale turbulence.
NTK Theory	Neural Tangent Kernel theory describes how networks behave in the infinite-width limit during training.	Helps explain training dynamics and guides the selection of learning rates to ensure the PDE residual converges.
Activation Functions	Mathematical functions (like Tan or Gel) that introduce nonlinearity into the model.	Directly affects the smoothness and differentiability of the solution, which is essential for computing higher-order derivatives.
Depth vs. Width	The structural trade-off between the number of layers and the number of neurons per layer.	Impacts both the computational efficiency and the speed at which the network converges to a stable physical solution.

Rigorous Formulations Of Physics Informed neural Network Pin

In this paper, we present a mathematically principled framework called Physics-Informed Neural Network (PINN) to embed physical laws into the training of deep neural networks for solving nonlinear partial differential equations (PDEs). Data-driven models could typically use a loss function that minimizes loss with regard to existing training observations (which may perhaps consist only of pure observations), but the robustness and power of PINNs allows governing equations that describe systems can also be integrated into the structure of the loss function, putting them in contact with real physics and observational data over time. The new formulation reformulates classical PDE solutions as optimizing the parameters of a neural network in order to bring numerical analysis and machine learning together into one coherent direction.

The PINN framework consists of an unknown function that works as a solution of a PDE and is represented in terms of the output produced by a neural network. Assume we have a general nonlinear pie (nonlinear in you want also with time) where the solution is provided over some spatial domain and odd interval as a function of space coordinates and time. This unknown solution is approximated by a neural network in PINNs which works on this coordinates as the inputs and predicted an corresponding values using output physical quantity (here u) at each decision point. However, weights and biases parameterize the neural network (they are tuned during training to minimize a well-designed loss function). This circumvents the step of explicitly discretizing the domain, which classical approaches such as finite differences or finite elements do.

PINNs represent a particularly exciting setting, as the key part of novelty is how to define the loss function (with various components capturing physics). The dominating term is the PDE residual which indicates how well will the neural network satisfy the governing equation at a few collocation points inside our domain. Which are either near random selected points or via well designed strategies, then we are just calculating the residual using the neural network output into the differential equation. Here is where automatic differentiation comes in useful because we can compute derivatives of any output layer with respect to the inputs very quickly and without any errors. Allowing this way the possible evaluation of complex differential operators without numerical approximations.

They add the terms that corresponds to boundary condition and the requirements for initial conditions to get the loss function. These are required to ensure the solution is physically relevant and incorporated as soft constraints (by including a penalty term in the loss function) or hard constraints (designing the NN architecture such that they are satisfied by construction). The proper balance among these components is very important because otherwise we may end up producing solutions that satisfy one part while violating the others. We can also simply use several adaptive weighting strategies to weigh the contribution of each loss component (where we can dynamically weight decided a priori during training and refine convergence/accuracy results).

PINNs — Introduction Physics-Informed Neural NetworksPINNs encode the laws of physics into a neural network. In the case of the strong form, we apply the PDE point wise at all collocation points — that is to say, for those relevant locations we demand our neural network exactly be a solution of the differential equation. This is a naive and common benchmark by which it would be natural to check your methods against, easy to implement but may have significant troubles with problems where flows become discontinuous or irregular. It is the variation (weak) form in which we integrate the PDE against test functions — such a formulation often leads to something which is more robust. It is very similar to techniques used in finite element analysis, and has been shown to improve stability (and only for certain problems did it seem successful in allowing convergence).

The third key part of a rigorous formulation involves separating continuous-time versus discrete-time PINNs. Note: One reason why continuous-time PINNs typically use a single neural network to model the whole spatiotemporal domain is that continuous-time PDE problems have a more natural notion of time. This is in contrast to discretised-PINNs, where we use time-stepping schemes and the temporal evolution can be covered by multiple networks or even require recurrent structures. The choice of which of these approaches to adopt depends on the problem and if it should be more accurate or computationally efficient.

PINNs are optimized using gradient-based methods (such as SGD or more sophisticated optimizers like Adam) to minimize the total loss function. Poor generalisation: DNN have high dimensions and a non-convex landscape so primed areas at beck on local minima can be slow to converge. There are also conflicting gradients as there may be parts of loss that need to make adjoining adjustments, so multiple components of the loss function must not be looked at together. In order to address these problems, techniques such as learning rate scheduling, gradient normalization and second-order optimization methods were explored.

Among the theoretical aspects that may be at play in the success of PINNs is connected to an approximation capacity of a neural network, their distribution in space (the collocation points) and the conditioning of an optimization problem. The universal approximation property ensures that hey you only need a neural net of sufficient capacity and it can asymptotically approximate your solution as well as you care to. In practice, this level of accuracy will require explicit design of the network architecture and training procedure. Selecting activation types, depth & width all play a role in how much information the network can derive from complex solution behaviours.

PINNs give a general framework for inverse problems in which some parameters or coefficients of the PDE are unknown and we learn them from data. Using these parameters as trainable variables, PINNs can then concurrently learn the solution while also finding unknown physical quantities [19]. This is of great help in system

identification or parameter estimation and even in data assimilation, since the methods will fail with limited (or noisy) data.

However, strengthening PINNs exposes certain shortcomings in the mathematical definition. Such training procedure can be computationally demanding, especially for high-dimensional solutions or more complex PDEs. The problem is that the accuracy suffers because the solution is highly sensitive to the hyper parameters and distribution of training points. The challenges are part and parcel for the robust architectures, optimization methods and nice theory to be developed – still an open research area which includes convergence and error bounds.

A Comprehensive Framework for Physics-Informed Neural Networks is an important advance of scientific computing. PINNs provide a novel and powerful framework for solving nonlinear PDEs, by integrating physical principles within the learning process. They offer us the potential to be an extremely powerful tool for many applications, as they are capable of encoding integrated data and physics without explicit discretization on complex domains. The downside being that in order to make the most out of them, you will require deep understanding, both on the mathematics and experience training your neural network.

Loss Landscape and Optimization Dynamics in PINNs

In Physics-Informed Neural networks (PINNs), when people optimise a PINN, they essentially rely on one particular feature of the configuration that really matters which the loss landscape or how the loss function changes by changing the neural networks parameters. Unlike traditional deep learning where the loss is typically data-driven, in PINNs a linear combination of physically relevant quantities (i.e. This inherent multi-objectivity leads to a complex and non-convex loss landscape that significantly hinders learning.

The stiff PINN loss landscape, a hallmark of the method resulting from the imbalance between different components of the losses. In other words, one type of constraint is more relevant than others with respect to the final output: for instance, in some scenarios PDE residuals are orders of magnitude smaller than boundary condition losses when calculating their gradients. This ineffectiveness created slow convergence and a tendency towards the local optima where an array of physical constraints are satisfied while others violated. Moreover, the interactions between different terms of the loss function give rise to sophisticated dependencies which make optimization infeasible.

Another factor of significance is that there are several local minima and flat regions in the loss landscape. These characteristics could be common in general deep neural networks, however, they are exacerbated phenomena specifically to PINNs due to having higher order derivatives as well as non-linear operators. Gradient based optimizers, like SGD, are often frustrated by flat regions or plateaus because it provides no clear दिशा (direction) where to move forward. On the contrary, sharp minima is over fitting or instability in cloudy data.

The collocation points distribution also influences the loss landscape. As a result of bad sampling strategies, some parts of the domain are poorly represented, creating inaccuracies in one region without guaranteeing an error-type uniform distribution. In order to solve this issue and ensure better convergence behaviour, adaptive sampling methods which focus more on the regions with higher residual errors have been developed.

A. Texts Shape Computer Programs

Solving for PINNs involves high-dimensional optimization problems and are usually trained with gradient-based approaches like SGD or adaptive optimizers such as Adam. However, the unique structure of a PINN loss function introduces new challenges which are typically unnecessary in conventional machine learning problems. The major challenge is called the gradient pathology, which occurs when gradients vanish or explode due to higher order derivatives in the PDE residual. Slowing down the study can make suitable learning difficult or even trigger unhelpful learning behaviours.

Spectral bias is another key concern, which refers to the speeds at which a neural network will learn low frequency components of the solution over high frequency components. This behaviour is useful for noise smoothing, but creates problems for nonlinear PDEs with steep gradients and shock-like or oscillatory phenomena. This is the case where the network could have converged to a smooth, but ultimately wrong "solution" that does not preserve key features of the true solution.

Furthermore, the convergence of different loss modules is also related. Simultaneous optimization of PDE residual, boundary conditions and initial conditions can yield conflicting gradients leading to inefficient updates. This is a significant problem in complicated PDEs where the restrictions operate on multiple periods of time. There are some techniques such as loss normalization, gradient clipping and adaptive-weighting to help balance the contribution of each component in order to reduce this.

Alternately, the choice of optimizer and learning rate schedule can have a significant impact on convergence as well. However, Adam is more popular because it can converge rapidly in the very first iteration due to some of its properties but fails to get perfect accuracy in later iterations. The more promising results came not from pure second order optimization but rather through hybrid approaches: starting training on Adam and dropping to a second order method such as L-BFGS. It uses curvature information to facilitate better traversing of the loss landscape near an optimum.

B. Optimization & Convergence

Many advanced optimization techniques have been devised to address the loss landscape and training dynamics issues associated with PINNs. One can be adaptive loss weighting, which is a well-known perfect mechanism for dynamically altering the contributions of different compound components during coaching. This enables spreading the physical constraints satisfaction and avoiding too much over- domination of some component on the behaviour. Gradient balancing and residual-based weighting have already proved their ability to accelerate convergence rate.

A closely-related and important approach that will be utilized is called curriculum learning, where the training problem gets more complicated over time. You could start training the network with some simpler version of the PDE or with less lines in space coordinate and during training expose it into to tougher cases. This approach ensures that the network builds a более стабильная (stable) base which generalizes better in tough conditions.

Scalability and convergence have also been studied with different domain decomposition methods. In this method, the computational domain is divided into several subdomains and independent neural networks are trained separately over each domain [11]. This greatly reduces the complexity of the optimization problem and can be trained in parallel. Additionally, interface conditions adhere to subdomain continuity allowing the global solution to be closer accurate.

Optimization can also be improved with more advanced architectures and a better mapping of features. Fourier feature embedding's and sinusoidal activation functions are part of one of the solutions to spectral bias[8] — a known issue in neural network training, as neural networks tend to be able to fit low frequency components easier than they can high (i.e., a case where some frequencies are detected much better than others). Second by introducing residual connections and batch normalization layers to help to ease the gradient flow and stabilize training (especially important for very deep networks).

In addition to this, regularization methods are key to maximizing generalization and preventing over fitting. These regularization methods ensure that the learned solution is smooth and respects physical constraints by either introducing penalty terms in the loss function, or constraining some weights of the network (e.g., enforcing positivity). Such as dropout, weight decay and early stopping.

Finally, a few methods for adaptive sampling have been proposed [35] to concentrate the computer power on the most challenging regions of the whole domain. These methods update the distribution of collocation points sequentially according to the error, increasing sampling around regions with high residuals to allow the network to learn more from them. This improves accuracy and also reduces overall training time.

Challenge	Description	Solution
Loss Imbalance	The PDE residual, initial conditions, and boundary conditions contribute unequally to the total loss.	Adaptive weighting (e.g., using Soft Attention or NTK-based weighting) to balance the terms dynamically.
Gradient Pathology	The backpropagation process suffers from vanishing or exploding gradients across different loss components.	Gradient clipping and normalization techniques to ensure stable updates during training.
Spectral Bias	The inherent tendency of neural networks to favor learning low-frequency components over high-frequency ones.	Use of Fourier features or sinusoidal activations (like SIREN) to capture fine-scale details.
Local Minima	The non-convex loss landscape causes the model to get trapped in suboptimal physical solutions.	Advanced optimizers like L-BFGS (often used as a second-stage refiner) or stochastic restarts.
Slow Convergence	Optimization stalls in flat regions of the loss landscape, particularly for high-order PDEs.	Curriculum learning, where the model starts with simpler parameters and gradually increases complexity.
Poor Sampling	Uniform collocation points fail to capture areas with sharp gradients or rapid transitions.	Adaptive sampling (e.g., RAR - Residual-based Adaptive Refinement) to place more points in "difficult" regions.

High Complexity	Modeling large, multi-scale domains leads to an overwhelmingly large-scale optimization problem.	Domain decomposition (e.g., XPINNs), which breaks the problem into smaller, parallelizable sub-domains.

Encoding Strategies for Boundary and Initial Conditions in Pickup Input Network

When dealing with Physics-Informed Neural Networks (PINNs), the enforcement of boundary conditions and initial conditions must be accurate in order to obtain a mathematically rigorous and physically meaningful learned solution. Some of the most essential features that define Partial Differential Equations (PDEs) and specifically nonlinear PDEs are that they rely on initial and boundary conditions to produce a unique or stable solution. If you enforce constraints at the boundaries and initial conditions sufficiently tightly, then even the largest neural network architectures will converge to a solution only that violates basic physical laws or generates non-physical results. Therefore, encoding these conditions in the PINN formulation during model design and training is crucial.

Soft constraints are one of the most widely used approaches to incorporate boundary and initial conditions in PINNs. This method uses the constraints as penalty terms within the loss function itself (for example, in addition to the PDE residual). A naive way of creating the loss function, for example, could be a weighted summation of terms that indicates how much the output (in points along the specified boundary and initial conditions) disagrees with you want it to move toward. Weighted individual component inputs are aggregated with the PDE residual to form a full loss. Even though this is simple and flexible, it causes balancing difficulty in various loss terms. When the weights are not defined appropriately, one of the terms can overpower and lead to an incomplete/wrong solution. Additionally, because adaptive weighting strategies remediate convergence issues via equal constraints during training, these contributions are occasionally dynamically weighted.

A more formalistic approach, based on hard constraints, has a neural network that is constructed to satisfy the boundary and initial conditions. This is typically achieved by bounding the output of the network using a trial function that denotes which inputs are 'physical' For example, the solution can be expressed as a linear superposition of a function satisfying the boundary conditions and an ersatz given by a neural network to explore the remaining degrees of freedom. This is accomplished by imposing the boundary conditions, ensuring that for any input of network parameters, the network satisfies the boundary conditions. This bypasses the penalty terms, thus making the loss function much simpler. However, good trial function selection is often nontrivial (e.

The second important ingredient is the way we use distance functions to enforce our boundary conditions. In this approach, we set a distance function that shows how far or near a point to the border of the domain. This function is then multiplied by the output of any neural network and hence automatically guarantees that our solution will always stay at the boundary conditions on the boundaries of $\partial\Omega$. The so-called approach is particularly suitable for Dirichlet boundary conditions and can be easily generalized to other situations by a specific adjustment. Although potentially incurring some extra computational cost and needing to be designed for numerically stability, this more general approach is easier to implement.

This can be because initial conditions, or state of the system at the beginning in the time domain is often used as boundary conditions. For time dependent PDEs, such initial conditions play a vital role in determining the evolution of the solution in time. In continuous-time PINNs, the initial condition is enforced as an (exponential) penalty loss term depending on how far away the current state was from a relatively known initial static state In discrete time formulations either the initial condition initializes the network or is part of a discretization through some kind of hybrid implicit-explicit time-stepping method. In any case the time-dependent behaviour of the system cannot be correctly described unless boundary conditions are applied precisely.

Depending on the type of problem and precision wanted soft and hard constraint methods can be used. They are applied easier (lower implementation barrier) and cover problems with complex or uncertain boundary conditions which usually occur in soft constraints. But they'll require some tuning and they're very slow to train. Hard constraints are more physically consistent but harder to construct and less flexible with time changing conditions. In practice, approaches that exploit a combination of both these methods for achieving a balance in speed and accuracy are commonly employed.

One of the most impactful new techniques for encoding boundaries and initial conditions would be augmented Lagrangian methods. Instead of adding constraint penalties, these methods augment the variables with implementing more accurately constraints through Lagrange multipliers. In addition, this helps to better satisfy the constraints at a faster convergence rate using multipliers in the loss function. This is a very principled method that has also proven to improve problems where only penalty approaches tend to perform poorly, and specifically on stiff or highly nonlinear PDEs.

Thus, another decisive factor among several found in literature as good strategies is how these training points are distributed in space and time to impose boundary conditions and initial conditions. Random sampling might miss the region where solution is changing quickly (i.e. high variation and sensitivity). Adaptive sampling: More points should be given in areas with higher error or complexity and can help tie down the constraints better and reduce flailing and result in a better overall solution. Especially for problems that have sharp gradients or discontinuities or are localized in nature.

It also shows that designing domain knowledge-based constraints encoding strategies can significantly enhance the models. For example, the network architecture or loss function can be designed to incorporate knowledge on symmetry properties (which could include conservation laws and/or known analytical solutions) into the learning process. The solution space for solving is sharpened thereby improving accuracy and easing the computational burden.

A lot of effort has gone into this, however, many issues remain. The design of effective methods for discretizing the constraints in complex, high dimensional or multi-physics problems is still an active area. Remaining innovations in theory and practice are needed to achieve numerical stability, scalability, and generalization to unseen problem settings. One of the most remarkable properties of PINNs that make it possible for them to tackle difficult scientific computing problems quite effortlessly is flexible and accurate be/ice incorporation.

Last, but by no means least, boundary and initial condition encodings are the crucible for the success of Physics-Informed Neural Networks. As a result, learning solutions using these methods are naturally compatible with the corresponding physical representation of the system according to these strategies (e.g., soft penalties, hard constraints and distance functions that can be implemented using various sophisticated optimization techniques). As research in this area continues to advance, more powerful and efficient techniques for enforcing the constraints will be closely tied to further enhancing the performance of PINNs on nonlinear PDEs.

Advanced Architectures for Pins

A. Motivation of designing more advanced architectures for the PINNs

While standard feed forward neural networks have been shown to be able to approximate solutions of nonlinear PDEs, they can lose the ability to learn in high-dimensional domains, multi-scale phenomena and strongly non-linear dynamics problems. Convergence of traditional physics-informed neural networks (PINNs) can be slow, so the spectral bias and expressivity of these low-dimensional objects are also limiting. The above challenges have motivated the use of advanced neural designs devoted to improving the efficiency, scalability and accuracy of PINNs.

Consequently, these challenges result in advances and innovations in architecture that describe functions by global dependencies and learn high frequency features distinctly. These architectures extend far beyond an evolutionary process and are indicative of operator learning, multi-scale problem description and physics aware design. With these innovations implemented, state-of-the-art PINNs are prepared to tackle real-world scientific tasks that would otherwise be hard or costly to compute.

B. Deep Operator Networks (Deponent's)

One significant recent development in the area of scientific machine learning is deep operator networks (Deponent's). Unlike traditional neural networks that learn finite-dimensional mappings x → y, deep Nets learn an infinite dimensional mapping where g is a function of the functional and v is a representation of both inputs 1 and outputs which we refer to as λειτουργικό 2. The specific use case this property is helpful for are families of PDEs where the point of the objective function is not simply to find a solution to particular equation but rather learning some form of the operator (that can be applied to many inputs).

Architecture of Deponents consists of two basic components: The branch network and the trunk network. The branch network handles input functions like initial or boundary conditions, and the spatial (or temporal) coordinates are streamed through the trunk network. Generally, the outputs of these two networks are combined (through an inner product) to obtain the final solution. The data mimicry structure exploiting by Deponent's comes from a fixed representation which generalizes arbitrary problems and is the very feature through which this method outperforms standard PINNs.

Deponent's [1] approach is a very powerful choice for learning solution operators of nonlinear PDEs, especially if we need the model to simulate several times. They are particularly useful in the numerical applications of uncertainty quantification, parameterized PDEs where these models need to generalize across many inputs. However, these things require large amounts of training data and a well thought out architecture to perform well.

C. Fourier Neural Operators (FNOs)

Another key architecture that has popped up recently is the Fourier Neural Operator (FNO) which demonstrates advantages some of the big weaknesses in physics-informed neural networks such as spectral bias and other types inefficiency at trying to capture global structures. Fourier Neural Operators (FNOs) FNO are designed machines that leverage the frequency domain via Fourier transforms on its input data such as a network can learn global and long-range dependency more effectively.

FNOs are built by eliminating local convolution operations with global spectral convolutions. In this approach, the input is transformed via Fourier and we have applied a series of learnable filters over certain frequency values. The data goes through a successful transformation followed by an inverse Fourier transform. That means the network can learn both low- and high-frequency features, eliminating Spectral Bias of standard neural networks.

As far as we are concerned, time-marching implementations with FNO has also been shown to be a powerful approach for complicated PDEs such as Navies–Stokes equations, something that the original PINN architecture often struggled to accommodate. They excel at high-dimensional problems, and can hit state-of-the-art accuracy using fewer parameters and less training time. However FNOs are typically more restrictive and structured grid data adherents so not suitable with irregular domains or sparse data; on the other hand for many applications PINNs leverages a more generalised key-free solution.

D. Multi-Scale and Adaptive PINNs

In the case of physical systems, where behaviour spreads across a spectrum of spatiotemporal scales, not all relevant features can generally be represented by one net. Multi-scale PINNs avoid this problem by introducing mechanisms at the model level that allow it to learn multi-resolution features. This is supported by the presence of hierarchical architectures, multi-resolution inputs or customized activation functions which detects coarse and fine scale patterns respectively.

A very popular trick is to stack multiple sub networks, each one learning features at a particular scale. The last solution is the sum of these sub networks where it allows even better to capture global and local variations together into the same model. Other routes are identified through adaptive activation function or mapper features, e. g., Fourier features to elevate the network with learning high-frequency components of signals.

Adaptive PINNs go a step further and automatically adapt the application-specific structure or training process according to variations in problem complexity. Adaptive sampling — Allocating more training points to regions with higher error Adaptive loss weighting — Scaling contributions from different loss components While the above discussed strategies allow models to allocate computing budget where most needed in order to improve accuracy and efficiency.

E. Physics-Informed Neural Operators (PINO)

Physics-Informed Neural Operators leverage both the ideas of operator learning and physics-informed spaces for a blended architecture that is flexible yet very physical. PINO was also built as a generalization of (1) embedding inter domain physics much more than classical neural operators based purely on data, whereas PINNs were developed from training neural networks with physical laws. This is achieved by embedding PDE residuals and constraint terms directly in the loss function to compel a learned operator that satisfies governing equations.

There exist various benefits of PINO, in comparison to both standard PINNs and neural operators. This enables it to be generalizable across problem settings while maintaining physics correctness, making it ideal for applications with varied boundary conditions or parameters such as PDE. In addition, PINO can use both data and physics to obtain good results in low or noisy data.

We show that physics in the loop is not only used as a regularize while learning an operator, but also to improve interpretability and robustness against noise perturbations. However, they can be more complex to design and train due to the fact that designing the architecture of a PINO model depends on your specific application along with tweaking out the loss function.

F. Hybrid and Emerging Architectures

Multiple studies have examined the performance of PINNs using hybrid architectures which combine different deep learning paradigms [2]. A good example is transformer-based models, which have been adapted to tackle the scientific computing case and are now ubiquitous in natural language processing. These are trained on long-range dependencies and have complex interactions that use attention mechanisms to better model high dimensional PDEs.

Another area that is beginning to emerge, is the use of graph neural networks (GNNs) for irregular domains and complex geometries. Unlike CNNs where the domains are represented as images (taking some context around the regions), GNNs represent domains as graphs where relationships can be better captured. This is of huge help in

applications such as porous medium by fluid flow or structural analysis (in case of element based level connectivity for example).

That is to say, hybrid models, which rely on the combination of data-driven and physics-based approaches, are gaining traction as well. These models leverage the dataset on hand in far greater detail to yield a more precise predictive model, and physical laws enforce consistency and generalization. Hybrid architectures: this approach represents a promising future direction of PINNs by combining architectural ideas from very different approaches.

Error Analysis and Convergence Theory

The next two chapters contain the main content of our approach on error estimates and convergence theory related to Physics-Informed Neural Networks (PINNs) for nonlinear Partial Differential Equations (PDEs), that answer why PINNs can output stable and reliable approximations. It is an active area of research to try to provide theoretical guarantees for accuracy and convergence of PINNs [6], but a foundation of rigorous theoretical underpinnings has not been established. The classically analysis for well controlled numerical methods raises fundamental question as to how error bounds and convergence rates can actually be realised with PINNs where neural network approximation plus gradient based optimization are the only thing to work on.

Error analysis for PINNs is centred around the decomposition of total error as expressed in three main aspects, namely: approximation error, optimization error and generalization error. This approximation error arises from the fact that a neural network is not capable of perfectly representing the true PDE solution. The answer is - the universal approximation theorem guarantees that a neural network with sufficient capacity can approximate any continuous function, but in reality this has limitations due to finite size and specific architecture of the network. In the presence of sharp gradients or complex nonlinearities, appropriate activation functions in combination with depth and width dictate how closely the network can match the underlying solution [3].

Instead, the optimization error is connected to your training and describes how close our learned solution is to the best possible solution (under some measure) in a function space of your choosing. PINNs usually minimize a non-convex loss function with gradient-based methods, which do not converge to the global optimum. As a consequence, the model can settle on local minima or saddle points that would yield unsatisfactory solutions. The learning rate, initialization and optimizer selection play an important role in optimization error. Second-order methods with adaptive learning strategies have been introduced to solve these problems and better convergence behaviour.

Generalization error – Again the most important, especially when you train model over very limited set of collocation points. Formally, this amount of the fitting error is based on how well the model performs on those training points versus all these points for the entire domain. The generalization in physics-informed neural networks (PINNs) is intricately linked to the distribution and density of the collocation points used for training. On the other hand, the assumption of being able to uniformly sample various areas of a solution area in different regions may not remain true for those regions having very complex solutions and will thus cause generalization issues. Adaptive sampling approaches, which suggest for more points in the residual greater areas, have been shown to enhance both generalization and accuracy as a whole.

The relationship of these components of the error to the increase in the number of training iterations, network parameters and collocation points is one of the key features of PINNs convergence theory. In nice cases, we expect to have a better solution, because: — That approximation and the optimization errors should decrease when the network capacity goes up and as the optimization converges. Since errors can influence one another, a network cannot simply be enlarged to improve its performance. You are also over-parameterized, which induce over fitting or instability in the model (especially if you have an unbalanced loss functions and/or little training data).

Within PINNs, the loss function is conditioned — meaning that the sensitivity of the tangents (i.e. how fast loss responds to net parameters) is also a factor affecting convergence. That leads to a slow or unstable convergence because they are badly conditioned problems (small changes of the parameters lead to huge variations in the loss). This is particularly common in nonlinear PDEs with slow-fast dynamics or multiple interacting scales. Some methods for improvement on conditioning and making training more stable: normalization (also called preconditioning), loss scaling, etc.

Also remember that automatic differentiation creates a differential structure on the entire neural network, we may want to calculate derivatives for PDE residual. Automatic differentiation provides exact derivatives up to machine precision, though numerical errors can still be introduced because of the finite precision arithmetic and the accumulation of errors through deep computation graphs. Mistakes in this process can affect the quality of PDE residual and severely constrain both the ranges of present models and all simulation outputs you do. To mitigate such issues, it is necessary that you implement with caution and use numerically stable techniques.

Recent theoretical studies also explore the connection of PINNs with classical numerical methods and provide insights into their convergence properties. However, on some cases and with right assumptions PINNs can be seen as a mesh free collocation method in which the mixed-compounded neural network becomes a global nature basis function. This perspective is one that allows researchers to tie PINNs back to other established methods, e.g., spectral methods where convergence rates depend on the smoothness of the solution and choice of basic functions. By using learned rather than classical basis functions, PINNs complicate analysis; in contrast.

There has been much work on this end as PINNs remain an active area of research with continuous efforts to formally derive approximation and generalization error bounds [PinN2]. In recent studies, efforts have begun to focus on the influence of network architectures, training algorithms and problem characteristics during convergence. This constitutes an initial effort to develop a more unified operator-based framework that can aid in the broader design and use of PINNs.

Nonetheless, empirical successes abound for combining PINNs with state of-the-art architectures and optimization strategies to achieve accuracy and trainability across a variety of applications. The role of data and physics, and continuous solutions over potentially complex domains being accommodated by deep learning methods, gives PINNs a workaround for the solution of nonlinear PDEs. However, achieving consistent and reproducible performance relies heavily on knowledge of these error mechanisms and the intentional design of both models and their training.

To sum up, Physics-Informed Neural Networks can only be assessed on its performance and stability through error analysis and convergence theory. A knowledge that helps you understand how different components affect the accuracy and convergence behaviour, decomposes the overall error into three major categories: approximation error (error due to model choice), optimization error (error due to optimization algorithm used) and generalization error. Though great progress has already been made, theoretically justifying in what sense the implications hold on several fronts opens up both challenge and opportunity for further research to harness the full power and flexibility of PINNs for scientific computing objectives.

Compare with classical numerical methods (FDM, FEM and Spectral methods)

A new numerical-analytical method called Physics-Informed Neural Networks (PINNs) has been introduced in the literature as a replacement of classical methods like FDM, FEM and spectral method for nonlinear Partial Differential Equations (PDEs). Despite the extensive literature surrounding traditional approaches in a multitude of applications, PINNs are intrinsically novel as they combine deep learning with optimization techniques. Learning about the respective strengths and weaknesses of these techniques is important to know when they should be used for various problems, but also if there are aspects from each that can complement one another.

Classical numerical methods (e.g. Finite Difference/Volume/Element) are founded on strong mathematical theory and utilize grids or meshes to discretize their computational domain. FDM is a method that approximates derivatives by finite differences applied to previously defined points on a structured grid, which makes it very easy to implement for problems with regular domain. Instead, FEM is based on variation principles and approximates solutions using piecewise basis functions on unstructured meshes providing far greater flexibility for complex geometries. Spectral methods use global basis functions (e.g., Fourier or polynomial expansions) which enables them to achieve very high accuracy for smooth solutions. These methods are the most widely used in engineering and scientific applications because their convergence properties and error bounds are well established.

However, initial state discretization introduces several issues. A mesh in complex domains or even relatively simple ones can be technically difficult to generate and computationally expensive when it arises from a high-dimensional problem. The mesh generation affects the precision of the process, a fine discretization is required for features such as sharp gradients or discontinuities. This results in a considerable computational load and memory requirement, especially in higher dimensions. Additionally, many continuous boundary problems are hard to quantify using classical methods; therefore it is simply impractical. They do poorly in the case of complex coupling between multiple physical processes or sparse and noisy data.

in contrast, the PINNs adopt an approach that has little use for a mesh, where all the solution can be viewed as a continuous function with which we approximate using a neural networks. Instead of discretizing the domain PINNs use collocation points, randomly sampled during training, to apply the governing equations and boundary conditions. It does not construct a mesh at all and it is much more versatile in terms of handling complicated geometries and high-dimensional spaces. Instead of using finite differences or perturbation estimates, you compute derivatives exactly with automatic differentiation.

One of the strongest advantages provided by PINNs is? unifying data and physics together?. Unlike classical methods, which relies purely on scientific models, PINNs can directly incorporate the observational or experimental data into their training process. This versatility is particularly beneficial for inverse problems—tasks such as

parameter estimation or data assimilation, for which traditional maximum-likelihood based approaches may be ineffectual. However, it also guides the PINNs to facilitate continuous solutions across the whole domain and ultimately eliminate the need for interpolation to perform evaluations at arbitrary points in space.

But PINNs also have their own problems that can be more difficult than classical physics methods. An important point to make is that, unlike in the case of classical Bayesian learning [5], we currently have no general theoretical guarantees on either convergence or error bounds. Classical methods support a straightforward convergence guarantee under assumptions, while the performance of PINNs is data-dependent (network architecture, training strategy and hyper parameter tuning). This makes them kind of less reliable to their situation and thus more volatile.

It is important to note that the PINN training process is also computationally expensive. These don {'} t need mesh but teaching NN to comply with the complicated nonlinear PDEs can be expensive in computation time. This is a non-convex loss function that you end up creating and trying to minimize, with many iterations (and likely many hyper-parameter searches) just to get something reasonable. In contrast to the above, classical methods are formulated as sparse linear or nonlinear systems of equations which can be solved by efficient numerical solvers; for a class of problems, such techniques will typically present some form of better performance (speed and stability).

The final dimension to measure is also known as accuracy. Descriptive Flexible Modalities — As a result of their exponential convergence, most classical methods, particularly spectral ones achieve rather high accuracy for smooth solutions. On the other hand, PINNs can be hindered by spectral bias e.g. learning small-scale characteristics or high frequency components [32]. Even though neural architectures and training strategies brought major advances towards the accuracy of PINNs, we do not expect that they can reach or exceed optimal precision as classical methods for some applications.

Scalability becomes a major issue when we consider high dimensions. In classical methods, the curse of dimensionality comes into play where the number of grid points grows exponentially in accordance to the dimensions. One such potential for complexity advantage of PINNs (in this respect) is that it scales with the network architecture, but again not necessarily as much with domain dimensionality. It also makes them better suited for high-dimensional PDEs that arise in finance, quantum mechanics and uncertainty quantification.

The choice between models like this (PINNs) and classical numerical ones does depends on the needs of the problem. In many cases, classical methods can provide faster and reliable solutions for well posed problems with smooth geometries and moderate computational resources. However, for complex domains with high-dimensionality or limited data size they are flexible and powerful alternatives to PINNs. And quite possibly the best solutions will be hybrid approaches that get the best of both worlds. One can generate training data for PINNs using classical methods and/or using the solutions obtained from classical solvers in the learning–refinement process of PINNs.

Several hybrid models are designing by combining the advantages of classical numerical techniques and PINNs to cope with it [24–26], which makes researchers bend to this aspect recently. With those models you are trying to combine the robustness of classical methods with flexibility and data integration functionalities of PINNs. This approach most useful for solving difficult multi-physics problems and where one method could not stand on its own.

Conclusion A distinct comparison between the Physics-Informed Neural Networks (PINNs) and classical numerical methods can be drawn. Even though classical methods remain the gold standard in many applications due to their more robust and better theoretically-founded attributes, PINNs provide a state-of-the-art framework to mitigate some key limitations found in classical approaches. In our opinion, with the continuous evolution of research in this domain, the combination of such strategies will be essential for future advances of new practices used for computational science and engineering.

High dimensional, multi-physics problems with Physics Informed Neural Networks (PINNs)

One of the key reasons why PINNs are so successful is that they have proposed a solution to this problem, even in high-dimensional settings and multi-physics problems where classical numerical methods struggle. In the majority of practical cases, we have a system in which multiple space (and perhaps time) dimensions are described through partial differential equations (PDEs), with extra parameters defining material properties or forces applied to it externally. But most importantly, a critical disadvantage of method solved classically [(i.e.) Finite Difference Method (FDM) and Finite Element Method (FEM)] is their disastrous nature in multiple dimensions because the computational cost for 2 variables or more grows exponentially with every extra variable e.g. This phenomenon

defined This becomes known as the “curse of dimensionality”. This makes standard discretization based methods computationally infeasible for high dimensional partial differential equations.

PINNs offer a promising alternative through the use of neural network to represent the solution, where the complexity of the neural net scales nearly costless with respect to the number of dimensions here. This allows PINNs to address high-dimensional problems more efficiently, since they do not require explicit generation or storage of large solution matrices. As such, all of the weight value in a neural network define a continuous mapping from input variables to output solutions that can be evaluated at any point within the domain without additional cost. Such characteristics are expected to be most useful in cases of applications like, financial mathematics, quantum systems and stochastic control often result in the high-dimensional problem.

Apart from high-dimensionality, Many physical systems can be described as multi-physics problems, which arise in diverse fields such as biology or fluid mechanics where multiple processes interact with each other. These are governed by a coupled set of partial differential equations that can describe effect two or more physical processes simultaneously. Fluid-structure interaction, for example, couples fluid dynamics and solid mechanics; climate modelling involves atmospheric, oceanic and land processes. In contrast, classic numerical methods require individual solvers in every physical realm, along with advanced coupling strategies to ensure continuity at the interfaces. This leads to higher complexity in terms of both computation and implementation.

In PINNs, all the governing equations must be represented in a unique loss function and therefore multi-physics problems can be treated with ease. This form can be either by having multiple neural networks or even a single shared network architecture to model the physical variables while their interactions are enforced through coupled residual terms. Our equation-free framework fuses different physical processes without traditional interface conditions. Furthermore, PINNs can be appropriately employed in scenarios where experimental/observational data of certain components are available but not others, as they capable of seamlessly dealing with multi-origin data.

This is especially relevant in this regard because multi-physics systems usually contain irregular domains and complex boundary conditions, which is challenging for traditional numerical discretization methods to deal with; while that PINNs have an ease with the flexibility it provides. PINNs are a class of mesh-free models that have no dependence on remising and can flow into different geometries or moving boundaries, in contrast to traditional numerical methods. As an example it has been shown especially relevant in dynamic interface problems, e.g., phase transitions or free-surface fluid flows.

These would indeed be favourable, but the application to PINNs to high dimensional as well as multi-physics problems come with their own set of challenges. With a massive increase in the number of equations and variables you need to train, larger loss functions are actually many more times complex. Forget about increasing gradient imbalance and slow convergence as this incapacitates the training process itself, requires advanced optimization strategies, along with carefully tuning hyper parameters. Additionally, universal networks that accelerate the learning of different regions in space and/or physical processes should develop more amenable solutions capable of handling efficient sampling approaches.

Such problems have recently been tackled by various approaches such as domain decomposition where a problem is divided into smaller subdomains and independent networks are trained in different regions of the function space. In addition to that, there is also some progress towards scaling the solution with parallel training and distributed computations. In addition to this, the study of hybrid approaches where general neural networks can be combined with traditional numerical methods are assuming prominence since they allow for the best of both worlds.

In short, PINNs are a good potential solution to difficult multi-physics and high-dimensional problems that are infeasible with standard methods. Such features has made them a boon for advanced industrial applications as they grant the ability to have a singular integrated data-friendly framework that is independent of mesh generation. However, it is still a work-in-progress to achieve effective and different solutions that relate through architectural structure and optimal methods in combination with theoretical attention.

Conclusion

Physics-Informed Neural Networks (PINNs) as a new hybrid area, revolutionary and original means of solving nonlinear Partial Differential Equations (PDEs), is a basic transformational approach from traditional numerical techniques to data-centric computation linked with physics. We have elaborated the theory and formulation behind PINNs, their underlying architecture and optimization dynamics and practical applications that highlight strengths and weaknesses that are way overdue by the scientific computing mainstream. Deep learning

has also opened up new avenues for solving complex, high-dimensional and multi-physics problems that were previously considered computationally too expensive.

At the heart of PINNs is the idea of directly incorporating governing equations into the learning process so that a neural network can express solutions which satisfy both observed data and fundamental physical laws. This paradigm did not discretize the computational domain explicitly, which is one of the many shortcomings that conventional numerical methods such as Finite Difference Method (FDM) and Finite Element Method (FEM) seemed to have. PINNs are more flexible and efficient for the irregular geometries, moving boundaries and high dimension spaces due to their mesh-free features. Automatic differentiation is a key tool to compute derivatives accurately, making it indispensable for the evaluation of PDE residuals and enforcing physics constraints.

Fundamentals of the mathematical study of nonlinear PDEs built a foundation for understanding both the complexity and difficulties in modelling real-world systems. From a mathematics standpoint, these equations generally depend on rigorous concepts related to well-pawedness, weak solutions and functional spaces. PINNs exploit the universal approximation property of deep neural networks to reformulate solving PDEs as a minimization problem, making them an effective alternative to classical approaches. Performs representation of compact, continuous and differentiable solutions allowing gradient-based learning algorithms and modern computational tools.

The story of PINNs is more than just a fancy new machine learning algorithm — the creation of composite loss functions, the incorporation and embedding of boundary and initial conditions, was all about how to leverage machine-learning capabilities with maximum benefit by seamlessly integrating physical laws into them. It offered details of what we wished to finds our approach for a Pinfish strong and weak formulations, continuous-time or discrete-time extremes to access flexibility. The characterisation of loss landscapes and optimisation dynamics also shows that many common features (especially gradient imbalance, spectral bias, slow convergence) complicate the training of PINNs. These challenges make advanced optimization strategies like adaptive loss weighting, hybrid optimizers and curriculum learning more important.

The advent of novel architectures like Deep Operator Networks (Deponent's), Fourier Neural Operators (FNOs) and Physics-Informed Neural Operators (PINO) is a significant milestone towards that end. Such extensions allows operator learning, enhances the representation of multi-scale phenomenon's, and captures global dependencies over traditional PINNs. And they also introduced Fourier feature mappings, sinusoidal activations and attention to solve complex PDEs more efficiently and expressively. Key progress in this domain can play an important part in addressing real-life problems with high dimensional inputs, non-linear relationships, and dynamic behaviour at more than one temporal scale.

Convergence theory and error analysis provided an essential background for the performance of PINNs. A more detailed analysis of the components impacting solution precision was conducted by decomposing overall error to that of approximation, optimization and generalization. Classical numerical methods have mature error bounds and convergence guarantees, while further research is still required for PINNs in this regard. One caveat is the absence of theoretical guarantees, although work is on-going for tighter bounds and improved interpretable models. Due to their complementary features, such High-dimensional Scalability (HDS), Flexibility with combining data and physics, thus maturation of both PINNs are compare against classical approaches on a training as well predicting complexity.

Moreover, the demonstration of PINNs on high-dimensional and multi-physics problems illustrated their potential for revolutionizing simulation in computational physics-based fields such as fluid dynamics, climate science, biomedical engineering, quantum mechanics etc. Additionally, with the unified framework that presents a joint approach for simulating arbitrary numbers of interacting physical processes, PINNs also simplify coupling strategies. This characteristic makes them a subject of interest in every real-world problem since the areas you might want to model often have sparse and noisy data. To enable PINNs to be used for larger and more complex systems further improvements in the model design, computational efficiency and parallelization should be developed.

Illustrations of both PINNs and state space priors, you might notice that not a single method is a panacea to all your predictive accuracy problems. The role of physics in deep learning is perhaps best mirrored by the big challenge: training deep neural networks to fulfil complex nonlinear PDEs. They can hinder performance and limit accuracy through spectral bias, gradient pathologies, and sensitivity to initialization. 16: Interpretability of the solution is still an unsolved problem in a safety-critical use case, where understanding how the model reacts to input matters. These challenges demand sustained collaboration between mathematicians, computer scientists and experts in the respective domains.

Several research streams are expected to drive PINNs for further growth in the long run. We expect that these advances leading to increasingly larger architectures (e.g. hybrid architectures which integrate data with physics) should help improve performance and scalability in parallel. The second area focuses on techniques to facilitate model training through breakthroughs in optimization algorithms, like second-order methods and adaptive methods. Moreover, the robustness and rich physical assets of PINNs can be extended to encompass uncertainty quantification and probabilistic modelling of PINNs for many real route problems. In particular, this will be done by exploring domain specific adaptations of our methods and integrating relevant prior knowledge into how models are designed, leading to more accurate and efficient solutions.

Another key research direction is to provide a more solid theoretical foundation of PINNs, along with convergence guarantees, error statements and stability. To move forward PINNs will have to build the bridge connecting classical numerical analysis and modern machine learning theory that would provide you the confidence for such broader acceptance in scientific engineering communities. Setting of standard benchmarks, datasets and evaluation metrics may lead to more systematic comparison and validation of distinct methodologies.

In conclusion, the Physics-Informed Neural Networks framework is a methodological tool that combines the strengths provided by deep learning with those provided by physical modelling to construct flexible and effective semi-analytical solutions for parametric nonlinear PDEs. The ability of these types of models to learn high-dimensional domains, model and couple data and physics, as well as be trained continually to drive solutions forward makes them a key post-discipline development technology paving the way for future advances in scientific computing. Although there are still several obstacles that needs to be overcome, the high speed of exploration and advancement in the respective area indicates that PINNs will certainly grow further and continue to enjoy an ever increasing importance in solving very complicated and practical science & engineering problems.

Reference

[1] Raise, M., Perdikis, P., & Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, *378*, 686-707. https://doi.org/10.1016/j.jcp.2018.10.045 (Cited by: 11245)

[2] Wang, S., Tang, Y., & Perdikis, P. (2021). Understanding and mitigating gradient pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, *43*(5), A3055–A3081. https://doi.org/10.1137/20M1318043

[3] Wang, S., Wang, H., & Perdikis, P. (2021). On the eigenvector bias of Fourier feature networks: From regression to solving partial differential equations. *SIAM Journal on Scientific Computing*, *43*(3), A1766–A1794. https://doi.org/10.1137/20M1339120

[4] Salzmann, V., Martel, J., Bergman, A., Liddell, D., & Wet stein, G. (2020). Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems (Neutrals)*, *33*, 7462-7473.

[5] Wang, S., Yu, X., & Perdikis, P. (2022). When and why PINNs fail to train: A neural tangent kernel perspective. *Journal of Computational Physics*, *449*, 110768. https://doi.org/10.1016/j.jcp.2021.110768

[6] GU, Y., Cao, X., & Wang, L. (2024). A Residual-Based Adaptive Refinement Physics-Informed Neural Networks (RAR-PINNs) method for fifth-order Kidd equation. *Research Gate*. https://doi.org/10.13140/RG.2.2.14620.19842

[7] Jag tap, A. D., & Karniadakis, G. E. (2020). Extended Physics-informed Neural Networks (XPINNs): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. *Communications in Computational Physics*, *28*(5), 2002-2041. https://doi.org/10.4208/cicp.OA-2020-0164

[8] Jag tap, A. D., Kharrazi, E., & Karniadakis, G. E. (2020). Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems. *Computer Methods in Applied Mechanics and Engineering*, *365*, 113028. https://doi.org/10.1016/j.cma.2020.113028

[9] Lu, L., Men, X., Mao, Z., & Karniadakis, G. E. (2021). Deeded: A deep learning library for solving differential equations. *SIAM Review*, *63*(1), 208-228. https://doi.org/10.1137/19M1274067 (Cited by: 1240)

[10] Tannic, M., Srinivasan, P., Mildenhall, B., et al. (2020). Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems (Neutrals)*, *33*, 7537-7547.

[11] Krishnapriyan, A., Thalami, A., Zhen, S., Kirby, R., & Mahoney, M. W. (2021). Characterizing possible failure modes in physics-informed neural networks. *Advances in Neural Information Processing Systems (Neutrals)*, *34*, 26561-26573.

[12] Cu, S., et al. (2025). Copen: Cognitive Physics-Informed Neural Networks for embracing robustness and generalization. *International Conference on Machine Learning (ICML)*. https://icml.cc/virtual/2025/poster/46458

[13] Shula, N., et al. (2026). Domain Decomposition PINNs: CPINN & XPINN — A Review of Methodologies for Scientific Machine Learning. *Emergent Mind*.

[14] Martinis, S. (2021). The Old and the New: Can Physics-Informed Deep-Learning Replace Traditional Linear Solvers? *Frontiers in Big Data*, *4*. https://doi.org/10.3389/fdata.2021.669097

[15] Jin, X., Cain, S., Li, H., & Karniadakis, G. E. (2021). NSFnets: Navies-Stokes flow nets for solving incompressible Navies-Stokes equations. *Journal of Computational Physics*, *426*, 110014. https://doi.org/10.1016/j.jcp.2020.110014

[16] Zhang, J., et al. (2026). Physics-Informed Neural Networks: A Review of Methodological Evolution, Theoretical Foundations, and Interdisciplinary Frontiers. *Applied Sciences (MDPI)*, *15*(14), 8092. https://doi.org/10.3390/app15148092

[17] Cho, M., et al. (2023). Separable PINN (SPINN): Efficient training of physics-informed neural networks. *Advances in Neural Information Processing Systems (Neutrals)*.

[18] Xiang, Z., Pang, W., Zhou, X., & Wen, W. (2022). Self-adaptive loss balanced physics-informed neural networks. *Neurocomputing*, *496*, 11-34.
[19] Wang, S., & Perdikis, P. (2024). Long-time integration of physics-informed neural networks via curriculum learning. *Journal of Computational Physics*, *502*, 112815.
[20] Redman, A., et al. (2025). Space-time domain decomposition for unsteady multistate problems in XPINNs. *Research Gate*.
[21] Yang, L., Men, X., & Karniadakis, G. E. (2021). B-PINNs: Bayesian physics-informed neural networks for forward and inverse PDE problems with uncertainty quantification. *Journal of Computational Physics*, *425*, 109913.
[22] Liu, D., & Wang, J. (2026). Self-adaptive weighting and sampling for physics-informed neural networks. *Journal of Computational Physics (Preprint)*. https://www.researchgate.net/publication/402962875
[23] MacClenny, L. D., & Braga-Net, U. M. (2020). Self-adaptive physics-informed neural networks. *arrive preprint arXiv:2009.04544*.
[24] Bishop, A., & Kraus, M. (2021). Multi-objective loss balancing for physics-informed deep learning. *arrive preprint arXiv:2110.09813*.
[25] Daw, A., et al. (2022). Rethinking the importance of sampling in physics-informed neural networks. *Proceedings of the Machine Learning for Engineering Modelling, Simulation, and Design (ML4Eng)*.
[26] Gaol, Z., et al. (2025). Spectral PINNSformer (S-Former): Mitigating spectral bias via Fourier feature embedding's and attention-driven decoding. *Chat Paper*. https://chatpaper.com/paper/197244
[27] Li, Z., et al. (2026). Curriculum-Learned Vanishing Stacked Residual PINNs for Hyperbolic PDE State Reconstruction. *Research Gate*.
[28] Lu, L., et al. (2021). Learning nonlinear operators via Deponents based on the universal approximation theorem of operators. *Nature Machine Intelligence*, *3*(3), 218-229. (Cited by: 1540)
[29] Wang, Z., et al. (2025). A Gradient-Variance Weighting Physics-Informed Neural Network for Solving Integer and Fractional PDEs. *Applied Sciences (MDPI)*, *15*(20), 11137. https://doi.org/10.3390/app152011137
[30] Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikis, P., Wang, S., & Yang, L. (2021). Physics-informed machine learning. *Nature Reviews Physics*, *3*(6), 422-440. (Cited by: 2890)