Article

MMS

Mathematics and Mechanics of Solids
1–26
© The Author(s) 2024
Article reuse guidelines:
sagepub.com/journals-permissions
DOI: 10.1177/10812865241257850
journals.sagepub.com/home/mms

Sage

Predicting and explaining nonlinear material response using deep physically guided neural networks with internal variables

Jacobo Ayensa-Jiménez
Tissue Micronvironment Lab (TME Lab), Aragon Institute of Engineering Research (I3A), University of Zaragoza, Zaragoza, Spain; Institute for Health Research ~~Aragon~~ (IISA), University of Zaragoza, Zaragoza, Spain

Javier Orera-Echeverría
Tissue Micronvironment Lab (TME Lab), Aragon Institute of Engineering Research (I3A), University of Zaragoza, Zaragoza, Spain

Manuel Doblare
Tissue Micronvironment Lab (TME Lab), Aragon Institute of Engineering Research (I3A), University of Zaragoza, Zaragoza, Spain; Institute for Health Research ~~Aragon~~ (IISA), University of Zaragoza, Zaragoza, Spain; Nanjing Tech University, Nanjing, China

Received 18 September 2023; accepted 8 May 2024

Abstract
Nonlinear materials are often difficult to model with classical state model theory because they have a complex and sometimes inaccurate physical and mathematical description, or we simply do not know how to describe such materials in terms of relations between external and internal variables. In many disciplines, neural network methods have emerged as powerful tools to identify very complex and nonlinear correlations. In this work, we use the very recently developed concept of physically guided neural networks with internal variables (PGNNIVs) to discover constitutive laws using a model-free approach and training solely with measured force–displacement data. PGNNIVs make a particular use of the physics of the problem to enforce constraints on specific hidden layers and are able to make predictions without internal variable data. We demonstrate that PGNNIVs are capable of predicting both internal and external variables under unseen load scenarios, regardless of the nature of the material considered (linear, with hardening or softening behavior and hyperelastic), unravelling the constitutive law of the material hence explaining its nature altogether, endowing the method with some explanatory character that distances it from the traditional black box approach.

Keywords
Nonlinear computational solid mechanics, deep neural network, internal variables, inverse modelling, physics-informed machine learning, physically guided neural networks

Corresponding author:
Manuel Doblaré, Tissue Micronvironment Lab (TME Lab), Aragon Institute of Engineering Research (I3A), University of Zaragoza, Mariano Esquillor s/n, 50018 Zaragoza, Spain.
Email: mdoblare@unizar.es

1. Introduction

Machine learning (ML) methods have become ~~today~~ one of the main tools in business, science, and technology. These methodologies enable the extraction of information from data that would be intractable by means of traditional methods [1]. In particular, a new discipline that has become very popular ~~lately~~ in ~~the~~ scientific literature is scientific machine learning (SML) [2] or physics-informed machine learning (PIML) [3], where data-driven methods are combined with the scientific knowledge ~~about~~ the system for different purposes, either to gain interpretability and reliability, or to improve the performance of the different methods.

Indeed, ML has changed the way we conceive science. On one hand, it has been used to discover the hidden physical structure of the data and unravel the equations of a system [4–6]. On the other hand, the tremendous predictive power of ML has been blended with the scientific consistency of the explicit mathematical representation of physical systems through the concept of *data-driven* models for simulation-based engineering and sciences (SBES) [7]. The latter is usually performed by the combination of raw data and physical equations [8,9], by enforcing a metriplectic structure to the model, related with the fulfillment of thermodynamic laws [2] or by defining the specific structure of the model [10,11]. These novel *data-driven science* approaches arise therefore with the main purpose of turning apparently non-physically meaningful data-driven models, where approaches such as artificial neural networks (ANNs) or in particular deep learning (DL) approaches have excelled, into physics-aware models, taking advantage of the tremendous power of ANNs as universal approximators [12–16].

Physically based model ~~is~~ not only useful for making predictions but also to help in acquiring new knowledge by the interpretation of its structure, parameters, and mathematical properties. Physical interpretability is, in most cases, at least as important as predictive performance. However, it is known that interpretability is one of the main weaknesses of ANNs, as the acquired knowledge is encoded in the strength of multiple connections, rather than stored at specific locations. That explains the huge efforts that are currently being made towards "whitening" the "black-box" character of ANN [17, 18] in what has been coined as eXplainable artificial intelligence (XAI) [19, 20]. In the context of data-driven ~~simulationbased~~ engineering and sciences (DDSBES) [21], two ways of proceeding can be distinguished: building specific ANNs structures endowed with the problem equations [22, 23], also known as the *inductive bias* approach, and/or by regularizing the loss function using the physical information [24–26].

For computational mechanics, the approach to ~~be~~ follow~~ed~~ depends decisively upon the data availability and the way in which this data is used. If we follow a supervised approach, there are two possibilities.

- The first one assumes that we know the whole physics of the problem. In that situation, supervised ML is used for the sake of computational requirements. In that sense, the ML models act as reduced order models (ROMs) [27] or surrogates [28,29] that can be used ~~as a surrogate~~ in problems involving uncertainty quantification [30], optimization [31], or control. In particular, ANNs have been extensively used as surrogates in computational mechanics [32,33]. Hybridizing DL with physical information is a way of improving standard DL methods in terms of data requirements, less-expensive training, or noise filtering at the evaluation step, thanks to regularization [22,24,34]. Therefore, we are interested in the predictive capacity of the approach.
- The second assumes that we know some of the physics of the problem. In that situation, we are interested in knowing the hidden physics that remains unknown, expressed in terms of some model parameters [26] or functional relations [35]. For that reason, we are rather interested in the explanatory character of the method.

In other situations, we follow unsupervised approaches. This may be indeed due to two different possibilities:

- If we know the whole physics of the problem, the use of ML regression tools is merely instrumental and is used as an alternative way of solving numerically some system of partial differential equations (PDEs) that require an important computational effort [36]. In this case, we are taking advantage of the fact that compositional function approximation (the one followed by ANNs) has surpassed additive function approximation in terms of computational cost for back-propagation algorithms and similar [37].
- When some of the physics of the problem is unknown, the intrinsic variability of the data (for instance when measuring spatial or temporal fields) may be exploited for an unsupervised discovery of some unknown constitutive models [38,39]. This approach must be unsupervised because variables such as stresses cannot be measured, but only certain integral representations, appearing in the integral expression of the linear momentum balance equation (equilibrium) can be supervised.~~**[AQ: 1]**~~

However, in the context of the Internet of Things (IoT), where data quantity generally dominates over data quality, the data availability and variability is key, ~~and it is~~ difficult to guarantee whether, considering for instance the specific case of computational solid mechanics, "the combination of geometry and loading generates sufficiently diverse and heterogeneous strain states to train a generalizable constitutive model with just a single experiment" [40]. Therefore, there are no other alternatives rather than introducing all the control or measurable variables in the workflow, while maintaining the desirable properties of the PIML approach, namely, its ability to get fast predictions in real time (for optimization and control issues) together with its explanatory capacity.

In this work, we demonstrate, how, in the context of computational solid mechanics, physically guided neural networks with internal variables (PGNNIVs) [22, 35, 41] enable the compliance with these requirements particularly well. PGNNIVs comprise ANNs that are able to incorporate some of the known physics of the problem, expressed in terms of some measurable variables (for instance forces and displacements) and some hidden ones (for instance stresses). Their predictive character, improving many of the features of conventional ANN [22], allow for fast and accurate predictions. In addition, PGNNIVs are able to unravel the constitutive equations of different materials from unstructured data, that is, uncontrolled test data obtained from system monitorization. The method is illustrated ~~for~~ hyperelastic problems, although it can be generalized to more complex problems, provided that part of the physical nature of the problem is known and the data are rich enough to learn the unknown physics.

The content of this paper is structured as follows. First, an overview of PIML is presented, focused on computational solid mechanics. Then, the methodology is described, including the use of PGNNIVs in computational mechanics, the computational treatment of the physical tensorial fields and operators, as well as the dataset generation and training process. Then, the main results are presented, of both the predictive and explanatory capacity of the method. Finally, conclusions are summarized, together with the main limitations and a brief overview of future work.

2. PIML in computational solid mechanics

PDEs are the standard way to describe physical systems under the continuum setting thanks to their overarching capacity to model extremely different and complex phenomena. However, closed-form solutions are not known in general. That is the reason why numerical methods have become the universal tool to obtain approximate but accurate solutions to PDEs. These methods consider a given discretization in space and time, which results in an algebraic (in general nonlinear) system that is then solved by means of standard matrix manipulation.

In the last three decades, however, attempts to solve PDEs from a data-driven point of view have been numerous. First tentatives [42,43] generalized earlier ideas for ordinary differential equations (ODEs). Since then, many different approaches have been proposed, from collocation methods [44–46], variational/energy approaches [47–49], loss regularization using physical or domain knowledge [25, 50–52], to the most recent approaches using automatic differentiation [26]. The use of ANN technologies for solving PDEs related with physical problems is nowadays known as physics-informed neural networks (PINNs). Other works have extensively tried to address this challenge using the stochastic representations of high-dimensional parabolic PDEs [37, 53, 54].

In order to provide the data-driven models with a meaningful physical character, remarkable efforts have been made in the recent years to embed physical information into data-driven descriptions. The potential of solving inverse problems with linear and nonlinear behavior in solid mechanics, for example, has been explored using DL methods [55], where the forward problem is solved first to create a database, which is then used to train the ML algorithms and determine the boundary conditions from assumed measurements. Other approaches initially ~~build~~ a constitutive model into the framework by enforcing constitutive constraints, and aim at calibrating the constitutive parameters [56]. Nowadays, the use of DL models is being exploited intensively in computational mechanics [57].

In this context, clearly differentiating between external and internal variables becomes an important factor when approaching complex physical problems, but this is disregarded most times from a data-driven viewpoint. External variables are those observable, measurable variables of the system, which can be obtained directly from physical sensors such as position, temperature, or forces; internal variables are non-observable (not directly measurable) variables, which integrate locally other observable magnitudes and depend on the particular internal structure of the system [41]. This is very important to consider in the ML framework since predictions associated to an internal state model require explicitly the definition of the cloud of experimental values that identifies the internal state model [58]. This implies "measuring", or better, assuming values for non-observable variables

[59, 60]. This is for example, the case of stresses in continuum mechanics that can be determined *a priori* only after making strong assumptions such as their uniform distribution in the center section of a sample under uniform tension.

An alternative is the use of PGNNIVs. This new methodological appraisal permits to predict the values of the internal variables by mathematically constraining some hidden layers of a deep ANN (whose structural topology is appropriately predefined) by means of the fundamental laws of the continuum mechanics such as conservation of energy or mechanical momenta that relate internal non-measurable with external observable variables. With this, it is possible to transform a pure ML-based model into a physics-based one without giving up the powerful tools of DL, including the implicit correlation between observable data and the derived predictive capacity. This way, not only the real internal variables of the problem are predicted but also the data needed to train the network decreases, convergence is reached faster, data noise is better filtered, and the extrapolation capacity out of the range of the training data-set is improved, as recently demonstrated [22], when compared to standard ANN. In line with the terminology and general framework used in PIML [3], PGNNIVs showcase an intuitive interplay between an *inductive bias* approach and a *learning bias* appraisal, where physical constraints are incorporated by means of a physics-informed likelihood, that is, additional terms in the loss function.[1]

PGNNIVs are in essence a generalization of PINNs. In the latter, physical equations constrain the values of output variables to belong to a certain physical manifold that is built from the information provided by the data and the specific form of the PDE considered. One of the shortcomings of PINNs is that, in general, only simple ordinary PDEs that contain a few free parameters and are closed-form can be considered [61]. In contrast, PGNNIVs not only apply to scenarios where PDEs involve many parameters and have complex forms but also to those where a mathematical description is not available. In fact, this new paradigm embodies a unique architecture where the values of the neuron variables in some intermediate layers acquire physical meaning in an unsupervised way, providing the network with an inherent explanatory capacity. Moreover, PGNNIVs offer the possibility to train a model that is predictive for any specified boundary and initial conditions [35].

The main differentiating features that, up to the authors' knowledge, constitute a relevant contribution to the advances of data-driven physical modelling and its particular application to nonlinear mechanics are two-fold~~ed~~: first, physical constraints are applied in predefined internal layers (PILs), in contrast to previous works on PINNs. PILs are internal layers that acquire certain physical meaning, inasmuch the learning of the whole system is driven towards the fulfilling of the governing equations. Second, and even most important, PGNNIVs are able to predict and explain the nature of the system all at once, that is, predictability of the variables as well as explainability of the constitutive law are ensured and learned altogether. In all related works, modeling assumptions on the constitutive law of the correspondent materials are directly imposed, so that the material response complies with certain constraints. On the contrary, PGNNIVs only enforce universal laws of the system (e.g. balance equations) and no prior knowledge on the constitutive model is incorporated, except when specific structural parameters are of real interest, in which case this information can also be included explicitly using parametric expressions.

3. Methods

In this section, we describe the methods of this work. We follow an approach from the generalities to the detail~~s~~. First, we briefly describe the general concept of PGNNIVs and its particularization to computational solid mechanics, in particular, to solve hyperelasticity problems, and more specifically, a plane-stress case study. Next, we elaborate the computational details about the methods, that is, how the different data structures and operators are treated as well as the different algorithms used. Finally, we dedicate the last subsection to data generation and training details to ensure reproducibility of the results.

3.1. PGNNIV in computational mechanics

3.1.1. Revisiting PGNNIVs. Classical deep neural networks (DNNs) are often represented as *black boxes* that theoretically can compute and learn any kind of function correlating the input and output data. In particular, they perform very well in areas of science and technology where complex functions convey good approximations of the governing phenomena. Although there exist some heuristic rules [62], these *black boxes* are usually trained via trial and error. Adding a physical meaning to the hidden layers and constraining them by adding an extra term to the cost function has already proven to have significant advantages such as less data requirements, higher accuracy, and faster convergence in real physical problems, as well as model unravelling capacities [22]. The basic principles of PGNNIVs are briefly exposed in the next lines.

Let us consider a set of continuous PDEs of the form

$$\mathcal{F}(u, v) = f, \text{ in } \Omega, \tag{1a}$$
$$\mathcal{G}(u, v) = g, \text{ in } \partial\Omega, \tag{1b}$$
$$\mathcal{H}(u) = v, \text{ in } \Omega, \tag{1c}$$

where u and v are the unknown fields of the problem, $\mathcal{F}$, and $\mathcal{H}$ are functionals representing the known and unknown physical equations of the specific problem, $\mathcal{G}$ is a functional that specifies the boundary conditions, and f and g are known fields.

The continuous problem has its analogous discretized representation in finite-dimensional spaces in terms of vector functions $\boldsymbol{F}$, $\boldsymbol{G}$ and $\boldsymbol{H}$, and nodal values $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{f}$, and $\boldsymbol{g}$. Particularly, $\boldsymbol{u}$ are the solution field nodal values and $\boldsymbol{v}$ are the unknown internal field variables at the different nodes.

A PGNNIV may be defined for a problem of type (1) in the following terms:

$$\begin{aligned}
\boldsymbol{y} &= \mathsf{Y}(\boldsymbol{x}),\\
\boldsymbol{v} &= \mathsf{H}(\boldsymbol{u}),\\
\boldsymbol{x} &= \boldsymbol{I}(\boldsymbol{u}, \boldsymbol{f}, \boldsymbol{g}),\\
\boldsymbol{y} &= \boldsymbol{O}(\boldsymbol{u}, \boldsymbol{f}, \boldsymbol{g}),\\
\boldsymbol{R}(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{f}, \boldsymbol{g}) &= 0,
\end{aligned}$$

where:

1. $\boldsymbol{u}, \boldsymbol{f}$, and $\boldsymbol{g}$ are the measurable variables of the problem.
2. $\boldsymbol{x}$ and $\boldsymbol{y}$ are the input and output variables respectively and will be defined depending on which relation $\boldsymbol{x} \mapsto \boldsymbol{y}$ wants to be predicted.
3. $\boldsymbol{I}$ and $\boldsymbol{O}$ are functions that compute the input $\boldsymbol{x}$ and the output $\boldsymbol{y}$ of the problem from the measurable variables. In other words, functions $\boldsymbol{I}$ and $\boldsymbol{O}$ define the data used as starting point to make predictions, $\boldsymbol{x}$, and the data that we want to predict, that is, $\boldsymbol{y}$.
4. $\boldsymbol{R}$ are the physical constraints, related to the relations given by $\boldsymbol{F}$ and $\boldsymbol{G}$.
5. Y and H are DNN models:
   - Y is the **predictive model**, whose aim is to infer accurate values for the output variables for a certain input set, that is, to surrogate the relation $\boldsymbol{x} \mapsto \boldsymbol{y}$.
   - H is the **explanatory model**, whose objective is to unravel the hidden physics of the relation $\boldsymbol{u} \mapsto \boldsymbol{v}$.

3.1.2. PGNNIVs in computational solid mechanics. Our aim now is to reframe equation (1) in the context of solid mechanics. To fix ideas, although it is not difficult to adapt the methodology to other constitutive models, we restrict this analysis to hyperelastic solids with constant and known density ρ. First, we have to consider equilibrium equations (momentum conservation) in the domain Ω.

In material coordinates, equilibrium reads

$$\mathrm{DIV}(\boldsymbol{P}) + \rho\boldsymbol{B} = \boldsymbol{0}, \tag{3}$$

where $\boldsymbol{P}$ is the first Piola–Kirchhoff stress tensor, $\boldsymbol{B}$ is the reference volumetric body force field, and DIV is the divergence operator in material coordinates.

If $\boldsymbol{\xi} = \chi(\Xi)$ is the deformation field that relates spatial ($\boldsymbol{\xi}$) and material (Ξ) coordinates, we define the deformation gradient tensor $\boldsymbol{F}$ as

$$\boldsymbol{F} = \mathrm{GRAD}(\chi) = \mathrm{GRAD} \otimes \boldsymbol{\xi}, \tag{4}$$

where GRAD is the gradient operator in material coordinates. Equation (4) shows that $\boldsymbol{F}$ is a potential tensor field so $\boldsymbol{F}$ must satisfy the following compatibility equation in each connected component of Ω:

$$\mathrm{CURL}(\boldsymbol{F}) = \boldsymbol{0}. \tag{5}$$

In the previous equations, we have used the different vector calculus operators in material coordinates, that in Cartesian coordinates and using Einstein summation convention, are expressed as

$$\left[\mathrm{GRAD}\left(f\right)\right]_k = \frac{\partial f}{\partial \Xi_k}. \tag{6a}$$

$$\left[\mathrm{DIV}\left(\boldsymbol{T}\right)\right]_k = \frac{\partial F_{ki}}{\partial \Xi_i}. \tag{6b}$$

$$\left[\mathrm{CURL}\left(\boldsymbol{T}\right)\right]_{km} = \varepsilon_{ijk}\frac{\partial F_{mj}}{\partial \Xi_i}. \tag{6c}$$

For hyperelastic materials, the first Piola–Kirchhoff stress tensor $\boldsymbol{P}$ is related to the deformation gradient tensor by means of the equation

$$\boldsymbol{P} = \frac{\partial \Psi}{\partial \boldsymbol{F}}, \tag{7}$$

where Ψ is the strain energy function expressed as a function of the deformation state given by $\boldsymbol{F}$, $\Psi = \mathfrak{F}(\boldsymbol{F})$. Obtaining this particular function is the subject of research of material sciences and many approaches are possible, ranging from phenomenological descriptions to mechanistic and statistical models.

Finally, equations (3), (4) or (5), and (7) must be supplemented with appropriate boundary conditions. We distinguish here between *essential* boundary conditions and *natural* boundary conditions. The former define the motion of the solid at some boundary points $\Gamma_E \subset \partial\Omega$:

$$\boldsymbol{\xi} = \bar{\boldsymbol{\xi}}, \tag{8}$$

whereas the latter define the traction vector at some other boundary points $\Gamma_N \subset \partial\Omega$:

$$\boldsymbol{P}\cdot\boldsymbol{N} = \bar{\boldsymbol{T}}, \tag{9}$$

where $\boldsymbol{N}$ is the material outwards normal vector and $\bar{\boldsymbol{\xi}}$ and $\bar{\boldsymbol{T}}$ are known values of the solid motion and material traction forces respectively.

Let us assume that it is possible to measure (for instance using digital image correlation techniques [63]) the system response in terms of its motion given by the map $\boldsymbol{\xi} = \chi(\Xi)$, that is, $\boldsymbol{\xi}$ is a measurable variable. Let us also assume that we can measure the volumetric loads, $\boldsymbol{B} = \boldsymbol{B}(\boldsymbol{\xi})$, as well as the prescribed traction forces $\bar{\boldsymbol{T}}$. Therefore, using equations (4) and (7), it is possible to express:

$$\boldsymbol{F} = \mathcal{A}(\boldsymbol{\xi}), \tag{10a}$$
$$\boldsymbol{P} = \mathcal{B}(\Psi,\boldsymbol{\xi}), \tag{10b}$$

for some appropriate differential operators $\mathcal{A}$ and $\mathcal{B}$. Therefore, we can recast hyperelastic solid mechanics as

$$\mathrm{DIV}(\boldsymbol{P}(\Psi,\boldsymbol{\xi})) = -\rho\boldsymbol{B} \quad \text{in} \quad \Omega, \tag{11a}$$
$$\boldsymbol{\xi} = \bar{\boldsymbol{\xi}} \quad \text{in} \quad \Gamma_E, \tag{11b}$$
$$\boldsymbol{P}\cdot\boldsymbol{N} = \bar{\boldsymbol{T}} \quad \text{in} \quad \Gamma_N, \tag{11c}$$
$$\Psi = \mathfrak{F}(\boldsymbol{F}(\boldsymbol{\xi})) \quad \text{in} \quad \Omega. \tag{11d}$$

Groupping equations (11b) and (11c), it is clear that we can express computational solid mechanics for hyperelastic materials as

$$\mathcal{F}(\boldsymbol{\xi},\boldsymbol{P}) = \boldsymbol{B}, \text{ in } \Omega, \tag{12a}$$
$$\mathcal{G}(\boldsymbol{\xi},\boldsymbol{\sigma}) = (\bar{\boldsymbol{\xi}},\bar{\boldsymbol{T}}) \text{ in } \partial\Omega, \tag{12b}$$
$$\mathcal{H}(\boldsymbol{\xi}) = \boldsymbol{P}, \text{ in } \Omega, \tag{12c}$$

that is in the form of equation (1) with $u = \boldsymbol{\xi}$, $v = \boldsymbol{P}, f = \boldsymbol{B}$, and $g = (\bar{\boldsymbol{\xi}},\bar{\boldsymbol{T}})$.

In small strains solid mechanics, it is common to work with the displacement field $\boldsymbol{U} = \boldsymbol{\xi} - \boldsymbol{\Xi}$ and to define the displacement-gradient tensor $\boldsymbol{J} = \mathrm{GRAD}(\boldsymbol{U})$. In that case, the constitutive equation is formulated as

$$\boldsymbol{\sigma} = \mathfrak{G}(\boldsymbol{\varepsilon}), \tag{13}$$

where $\boldsymbol{\sigma}$ is the Cauchy stress tensor and $\boldsymbol{\varepsilon}$ the Cauchy small strain tensor

$$\boldsymbol{\varepsilon} = \mathrm{symgrad}(\boldsymbol{U}) = \frac{1}{2}(\boldsymbol{J} + \boldsymbol{J}^{\mathsf{T}}), \tag{14}$$

and $\mathfrak{G}$ is a tensor map. With these considerations, the equations of the problem are now written as

$$\mathcal{F}(\boldsymbol{U}, \boldsymbol{\sigma}) = \boldsymbol{B}, \text{ in } \Omega, \tag{15a}$$
$$\mathcal{G}(\boldsymbol{U}, \boldsymbol{\sigma}) = (\bar{\boldsymbol{U}}; \bar{\boldsymbol{T}}), \text{ in } \partial\Omega, \tag{15b}$$
$$\mathcal{H}(\boldsymbol{U}) = \boldsymbol{\sigma}, \text{ in } \Omega, \tag{15c}$$

again in the form of equation (1) with $u = \boldsymbol{U}$, $v = \boldsymbol{\sigma}$, $f = \boldsymbol{B}$ and $g = (\bar{\boldsymbol{U}}, \bar{\boldsymbol{T}})$.

Once we have discretized the problem, our aim is to predict a motion field $\boldsymbol{\xi}$ (or a displacement field $\boldsymbol{U}$) from a particular load case, expressed in terms of the volumetric loads and the natural boundary conditions,[2] $\bar{\boldsymbol{T}}$. Therefore $\boldsymbol{x} = \boldsymbol{I}(\boldsymbol{u}, \boldsymbol{f}, \boldsymbol{g}) = (\boldsymbol{f}, \boldsymbol{g}) = (\boldsymbol{B}, \bar{\boldsymbol{T}})$ is the input vector variable and $\boldsymbol{y} = \boldsymbol{O}(\boldsymbol{u}, \boldsymbol{f}, \boldsymbol{g}) = \boldsymbol{u} = \boldsymbol{\xi}$ the output vector variable. With these last remarks, the PGNNIV problem is stated for finite solid mechanics as

$$\boldsymbol{\xi} = \mathsf{Y}(\bar{\boldsymbol{T}}), \tag{16a}$$
$$\boldsymbol{P} = \mathsf{H}(\mathrm{KIN}(\boldsymbol{\xi})), \tag{16b}$$
$$\boldsymbol{x} = (\bar{\boldsymbol{T}}, \boldsymbol{B}), \tag{16c}$$
$$\boldsymbol{y} = \boldsymbol{\xi}, \tag{16d}$$
$$\boldsymbol{R}(\boldsymbol{\xi}, \boldsymbol{P}, \bar{\boldsymbol{T}}) = (\mathrm{DIV}(\boldsymbol{P}) - \rho\boldsymbol{B}; \boldsymbol{P} \cdot \boldsymbol{N} - \bar{\boldsymbol{T}}; \boldsymbol{\xi} - \bar{\boldsymbol{\xi}}). \tag{16e}$$

where $\mathrm{KIN}(\boldsymbol{\xi})$ is a selected kinematic descriptor of the strain state, such as the deformation gradient $\boldsymbol{F} = \mathrm{GRAD}(\boldsymbol{\xi})$, the right Cauchy–Green deformation tensor $\boldsymbol{C} = \boldsymbol{F}^{\mathsf{T}}\boldsymbol{F}$, or the Green–Lagrange strain tensor $\boldsymbol{E} = \frac{1}{2}(\boldsymbol{C} - \boldsymbol{I})$, among others. For small strains solid mechanics, the methodology simplifies to

$$\boldsymbol{U} = \mathsf{Y}(\bar{\boldsymbol{T}}), \tag{17a}$$
$$\boldsymbol{\sigma} = \mathsf{H}(\mathrm{symgrad}(\boldsymbol{U})), \tag{17b}$$
$$\boldsymbol{x} = (\bar{\boldsymbol{T}}, \boldsymbol{B}), \tag{17c}$$
$$\boldsymbol{y} = \boldsymbol{U}, \tag{17d}$$
$$\boldsymbol{R}(\boldsymbol{U}, \boldsymbol{\sigma}, \bar{\boldsymbol{T}}) = (\mathrm{div}(\boldsymbol{\sigma}) - \rho\boldsymbol{B}; \boldsymbol{\sigma} \cdot \boldsymbol{N} - \bar{\boldsymbol{T}}; \boldsymbol{U} - \bar{\boldsymbol{U}}). \tag{17e}$$

The appropriate structure and architecture of H and Y depend on the complexity of the material in hands, which is discussed later. It is appropriate to point out that all the discussion uses material coordinates and fields, although the same approach can be formulated for spatial coordinates and fields, by adapting the pertinent equations.

3.1.3. Case study: geometry and external forces. For illustration purposes, the case study considered in this work consists in a non-uniform biaxial test on a rectangular plate of height $L_1 = 16$ cm and width $L_2 = 20$ cm, under plane stress. No volumetric loads are incorporated, that is, $\boldsymbol{B} = \boldsymbol{0}$. We impose a certain arbitrary compression load profile $p = p(s)$ (where s is the coordinate along the right and top contour). To accelerate computations, we consider a load profile that is symmetric with respect to the vertical and horizontal axis and acts perpendicularly to the plate contour, as shown in Figure 1. The symmetry of the problem allows therefore for the analysis of an equivalent problem by extracting the upper-right portion of the plate and applying the corresponding symmetry boundary conditions. For the discretization, we considered a mesh of $n_x \times n_y = 11 \times 9$ nodes.

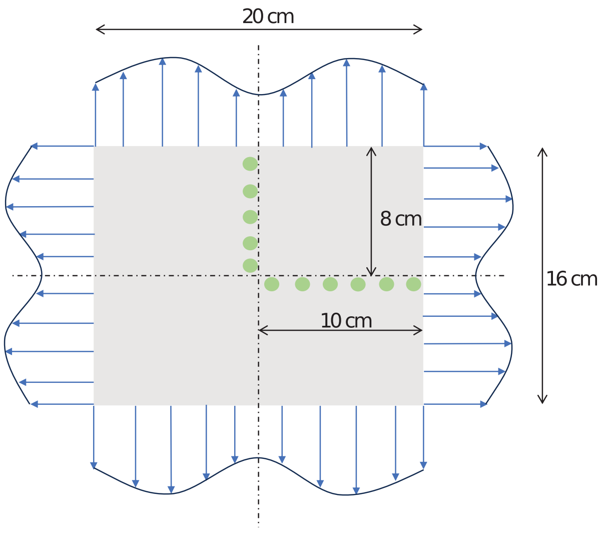

Figure 1. Dimensions and representation of the non-uniform biaxial test on a 2D plate. Load $p = p(s)$ acting perpendicularly to the contour is arbitrary, provided that it is compatible with the symmetry of the problem. We locate the origin $(x, y) = (0, 0)$ at the geometrical center of the plate.

3.2. Data and operator representation

In this section, we discuss how the different mathematical objects in our use case problem (scalar, vectorial, and tensorial fields and operators) are represented. First, we describe how the different fields are encoded and related to the measured data. Then, we explain how the different operators involved are built. This includes both the known operators (equilibrium and compatibility) as well as the unknown relationships comprising the predictive and explanatory networks, Y and H respectively. Then, we discuss how physical constraints are hardwired into the ANN, so that the built PGNNIV is tailored towards the discovery of constitutive models that comply with the physics of the solid mechanics problem in the context of elastic and hyperelastic theory, constraining therefore the learning space and bypassing the parametrization of the constitutive law.

3.2.1. Data structures. The data that contain the nodal and element-wise variables (that is, displacements and stresses/strains respectively) is stored in array structures. Now we introduce the notation that will be used for referring to a given tensor field, that is represented by an array $\mathtt{I}$ containing the information of the tensorial field itself. The different dimensions of the data are represented using the indexation $\mathtt{I}[I|J|K]$, where I is a multi-index associated with the discretization of the problem, J with the tensorial character and K with the data instance, that is, the piece of data (including all scalar, vectorial and tensorial fields) corresponding to one specific load case. Thus, considering that we have a dataset of size N and a discretization of size $n_x \times n_y$, the displacement field is represented by $\mathtt{U}[i,j|k|l]$ where $i = 1, \ldots, n_x, j = 1, \ldots, n_y$, and where $k = 1, 2$ (2D problem) and $l = 1 \ldots, N$, so that

$$\mathtt{U}[i,j|k|l] = u_k^{(l)}(x_i, y_j)$$

is the k component of the displacement field evaluated at (x_i, y_j) (that is, the node (i,j)) corresponding to the data l. Analogously, the strain and stress fields are represented respectively by $\mathtt{E}[i,j|k,l|m]$ and $\mathtt{S}[i,j|k,l|m]$, where $i = 1, \ldots, n_x - 1, j = 1, \ldots, n_y - 1, k, l = 1, 2$, and $m = 1 \ldots, N$, such that, for instance,

$$\mathtt{E}[i,j|k,l|m] = E_{kl}^{(m)}\left(\frac{1}{2}(x_i + x_{i+1}), \frac{1}{2}(y_j + y_{j+1})\right)$$

is the k, l component of the strain tensor evaluated at the element (i,j) corresponding to the data m.

Finally, the traction forces, $\bar{\boldsymbol{T}}^{\text{top}}$ and $\bar{\boldsymbol{T}}^{\text{right}}$, which are treated as the inputs of our problem (provided the volume forces are not considered), are represented by $\mathtt{T}^{\text{top}}[i|j|k]$ where $i = 1, \ldots, n_x, j = 1, 2$, and $k = 1 \ldots, N$, so that

$$\mathtt{T}^{\text{top}}[i|j|k] = T_j^{\text{top},(k)}(x_i, L_1/2),$$

and $\texttt{T}^{\text{right}}[i|j|k]$ where $i = 1, \ldots, n_y, j = 1, 2$, and $k = 1 \ldots, N$, ~~so that~~

$$\texttt{T}^{\text{right}}[i|j|k] = T_j^{\text{right},(k)}(L_2/2, y_i),$$

both associated with the data instance k.

Note that the computational details of the method do not depend on whether the finite or infinitesimal framework is used. We only need to adapt the interpretation of the strain and stress fields, $\texttt{E}$ and $\texttt{S}$, accordingly and adapt the definition of $\texttt{E}$ in terms of the displacement field $\texttt{U}$, $E = \text{KIN}(\texttt{U})$.

3.2.2. Operator construction. In this section, we specify the details to build the predictive and explanatory networks (Y and H), as well as the constraint operator $\boldsymbol{R}$. The definition of the complexity of a PGNNIV adapted to solid mechanics stems from the architecture of the predictive and explanatory networks, Y and H, as they must be able to learn the nonlinearities between variables $\bar{\boldsymbol{t}} \mapsto \boldsymbol{U}$ or $\bar{\boldsymbol{t}} \mapsto \boldsymbol{\xi}$ and $\boldsymbol{E} \mapsto \boldsymbol{P}$ (or $\boldsymbol{\varepsilon} \mapsto \boldsymbol{\sigma}$).

When knowing the input data structure, the predictive network must be able to represent the data variability, so typically it has an autoencoder-like structure. Its complexity is therefore associated with the latent dimensionality and structure of the volumetric loads and boundary conditions. If not, a deep-enough ANN is able to reproduce the input–output relation as a universal approximator. Here we build an ANN that is sufficiently accurate when predicting the output from a given input.[3]

Thus, since we consider a biaxial quadratic load applied to the plate, the complexity of the network depends on the data variability. For non-uniform loads, the values of the elemental loads are the input of a *vanilla* DNN whose output are the nodal displacements. For the uniform load, we use an autoencoder-like DNN.[4]

Now we discuss the architecture of the explanatory network. As in this work, we restrict ourselves to the elastic regime, the input variable is the given strain state at an arbitrary node $\texttt{E}[i,j|k,l|m]$ (E represents the Cauchy deformation tensor for infinitesimal theory and the Green–Lagrange deformation tensor for finite strains theory), and the output variable is the associated stress state at that same element, $\texttt{S}[i,j|k,l|m]$ (again, S represents the Cauchy stress tensor or the first Piola–Kirchhoff tensor depending on whether we are in the infinitesimal or finite strains theory). Note that under the homogeneity assumption (and postulating that the stress state depends only on the value of the deformation at the same point), the explanatory network is a nonlinear map $\mathbb{R}^3 \to \mathbb{R}^3$, due the symmetry of both tensors, which may be expressed symbolically as

$$\texttt{S}[i,j|\cdot,\cdot|m] = \mathsf{H}\left(\texttt{E}[i,j|\cdot,\cdot|m]\right).$$

In particular, H has to be able to capture the highly nonlinear dependencies that may exist between variables. This is in theory possible thanks to the universal approximation theorem (see the introduction section for some references): by adding more internal layers (also known as hidden layers) to the DNN model, we can provide the network with the learning capability and complexity that a particular nonlinear constitutive law might require. The implementation of the explanatory network for homogeneous materials is efficiently implemented using a pointwise ~~nonliner~~ operator, that is, a DNN to move across the domain element by element, but expanding the features in a higher dimensional spaces, as illustrated in Figure 2. We call this type of architecture a moving multilayer perceptron (mMLP).[5]

For the very particular case of linear elasticity, we can explicitly parameterize the explanatory network, $\mathsf{H}(\boldsymbol{\sigma}) = \boldsymbol{H}(\boldsymbol{\sigma}; \boldsymbol{D})$, where $\boldsymbol{D}$ is the elastic tensor and $\sigma_{ij} = D_{ijkl}\varepsilon_{kl}$. Using Voigt convention, the tensor $\boldsymbol{D}$ is expressed, under plane stress conditions and in the most general case, as a 3×3 matrix.

$$\boldsymbol{D} = \begin{pmatrix} d_{11} & d_{12} & d_{13} \\ d_{12} & d_{22} & d_{23} \\ d_{13} & d_{23} & d_{33} \end{pmatrix}, \tag{18}$$

~~where~~ d_{ij} ~~are~~ free model parameters, whereas for an isotropic elastic material, we have

$$\boldsymbol{D} = \frac{E}{1-\nu^2} \begin{pmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & 1-\nu \end{pmatrix}, \tag{19}$$

where the elastic modulus E and the Poisson ratio ν are the only free fitting parameters learned during the training process.

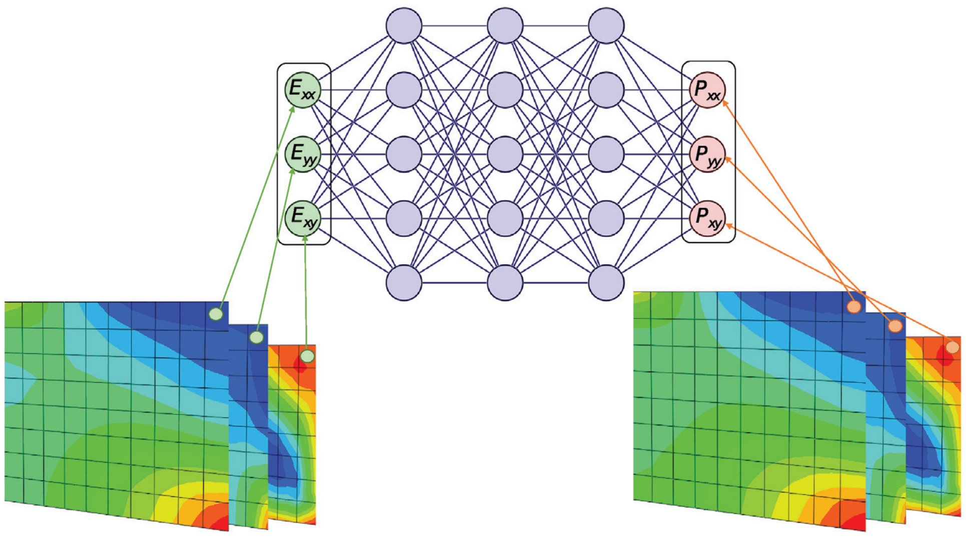

Figure 2. Representation of the explanatory network for the 2D plane stress problem. On the left, the strain field computed from the displacement field predicted by Y is represented on the plate. In an homogeneous material, the DNN is fed with the strain state on each element, and the correspondent stresses are obtained. The set of weights of this DNN moves across the elements of the plate and updates after each iteration of the optimization (see Figure 3 for the whole picture), similar to a 2D-convolutional filter. We call this type of architecture a moving multilayer perceptron (mMLP).

Table 1. Detailed architecture of the PGNNIV for the different illustrated cases.

Small strains problem

Network	Class of architecture	Detailed layers (input and output neurons included)
Predictive network Y	MLP	36/50/tanh/50/tanh/50/tanh/50/tanh/198
Explanatory network H	mMLP	3/100/tanh/100/tanh/100/tanh/100/tanh/100/tanh/100/tanh/100/tanh/3

Large strains problem

Network	Class of architecture	Detailed layers (input and output neurons included)
Predictive network Y	autoencoder	18/9/tanh/3/tanh/3/tanh/9/tanh/198
Explanatory network H	mMLP	3/60/ReLU/60/ReLU/60/ReLU/60/ReLU/60/ReLU/3

An analogous reasoning holds for more complex parametric dependencies. It is possible to express the explanatory network H as a parametric model relating the strain and the stress states, that is

$$\mathtt{S}[i,j|\cdot,\cdot|m] = \mathbf{H}\left(\mathtt{E}[i,j|\cdot,\cdot|m];\boldsymbol{\Lambda}\right),$$

where $\boldsymbol{\Lambda}$ are some pre-defined fitting parameters that are learned during the training step. In particular, an homogeneous material is described as

$$\mathtt{S}[i,j|\cdot,\cdot|m] = \mathbf{H}\left(\mathtt{E}[i,j|\cdot,\cdot|m];\boldsymbol{\Lambda}_{ij}\right).$$

Table 1 shows the architecture details for the unique PGNNIV trained on the linear, softening and hardening materials in small strains theory and the unique PGNNIV trained on the Ogden material in finite strain theory, in both cases when the parametric structure of the constitutive equation is not prescribed. When the constitutive equation is prescribed, the explanatory network is replaced by the constitutive relation expressed in terms of the different parameters. For the small strains case, the Y network layer includes empty elements (null value) corresponding to the most general natural boundary conditions where shear loads are applied on the contour. Therefore, the input layer comprises $2 \times (n_x - 1) \times (n_y - 1) = 2 \times (10 + 8) = 36$ neurons. The number of neurons of the output layer corresponds to $2 \times n_x \times n_y = 2 \times 11 \times 9 = 198$.

3.2.3. Coupling the two networks using physical constraints. The definition and subsequent formulation of the PGN-NIV framework implies that the loss function includes a term proportional to the quadratic error between the predictions and true values of the output variable (minimization of the maximum likelihood of the data given the parameters) and other penalty terms related to some (physical) equations, that is, equilibrium constraints. Therefore the different terms involved are as follows:

1. Loss term associated with the measurement of the displacement field:

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N} ||\bar{\boldsymbol{U}}^{(i)} - \Upsilon(\boldsymbol{t}^{(i)})||^2, \tag{20}$$

   where $\bar{\boldsymbol{U}}^i$ is the observed displacement corresponding to sample i.
2. Constraint associated with the equilibrium equation.

$$\nabla \cdot \boldsymbol{P} = \boldsymbol{0}, \text{or} \quad \nabla \cdot \boldsymbol{\sigma} = \boldsymbol{0}. \tag{21}$$

3. Constraint associated with the compatibility in the domain.

$$\boldsymbol{E} - \frac{1}{2}\left(\boldsymbol{F}^{\mathsf{T}}\boldsymbol{F} - \boldsymbol{I}\right) = \boldsymbol{0}, \text{or} \quad \boldsymbol{\varepsilon} - \mathrm{symgrad}(\boldsymbol{U}) = \boldsymbol{0}. \tag{22}$$

4. Constraint associated with the equilibrium of the stresses in the boundary.

$$\boldsymbol{P} \cdot \boldsymbol{N} - \boldsymbol{T} = \boldsymbol{0}, \text{ in } \Gamma_N. \tag{23}$$

5. Constraints associated with the compatibility of the displacements in the boundary.

$$U_x(x = 0, y) = 0, \quad U_y(x, y = 0) = 0. \tag{24}$$

The global cost function (which turns out to be a *virtual* physics-informed likelihood in a Bayesian formulation or, equivalently, a regularized cost function in the most common terminology) can be computed as a weighted sum of MSE and PEN, with PEN referring to the physical terms, that is, equations (21), (22), (23), and (24). As equation (22) may be expressed as an explicit relation between $\boldsymbol{E}$ (or $\boldsymbol{\varepsilon}$) and $\boldsymbol{U}$, it is directly embedded in the network architecture, as illustrated in Figure 3. Therefore, the loss is expressed as:

$$\mathrm{CF} = \mathrm{MSE} + \mathrm{PEN}, \tag{25}$$

where

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}\left[p_1 \|\bar{\boldsymbol{U}}^{(i)} - \Upsilon\left(\bar{\boldsymbol{T}}^{(i)}\right)\|^2\right], \tag{26}$$

and

$$\begin{aligned}\mathrm{PEN} = {} & \frac{1}{N}\sum_{i=1}^{N}\left[p_2 \|\nabla \cdot \boldsymbol{P}^{(i)}\|^2\right] \\ & + \frac{1}{N}\sum_{i=1}^{N}\left[p_3 \|\boldsymbol{P}^{(i)} \cdot \boldsymbol{N} - \bar{\boldsymbol{T}}^{(i)}\|^2\right] \\ & + \frac{1}{N}\sum_{i=1}^{N}\left[p_4\left(\|U_x^{(i)}(0, y)\|^2 + \|U_y^{(i)}(x, 0)\|^2\right)\right],\end{aligned} \tag{27}$$

where with the superscript (i), we refer to the ith piece of data, and $p_j, j = 1, 2, 3, 4$ are penalty coefficients that account for the relative importance of each term in the global CF (and may be seen as Lagrange multipliers that softly enforce the constraints). Recall that no penalty for the compatibility in the domain is included since $\boldsymbol{E} - \frac{1}{2}(\boldsymbol{F}^{\mathsf{T}}\boldsymbol{F} - \boldsymbol{I})$ (or $\boldsymbol{\varepsilon} - \frac{1}{2}(\nabla \otimes \boldsymbol{U} + \boldsymbol{U} \otimes \nabla)$) is identically $\boldsymbol{0}$.

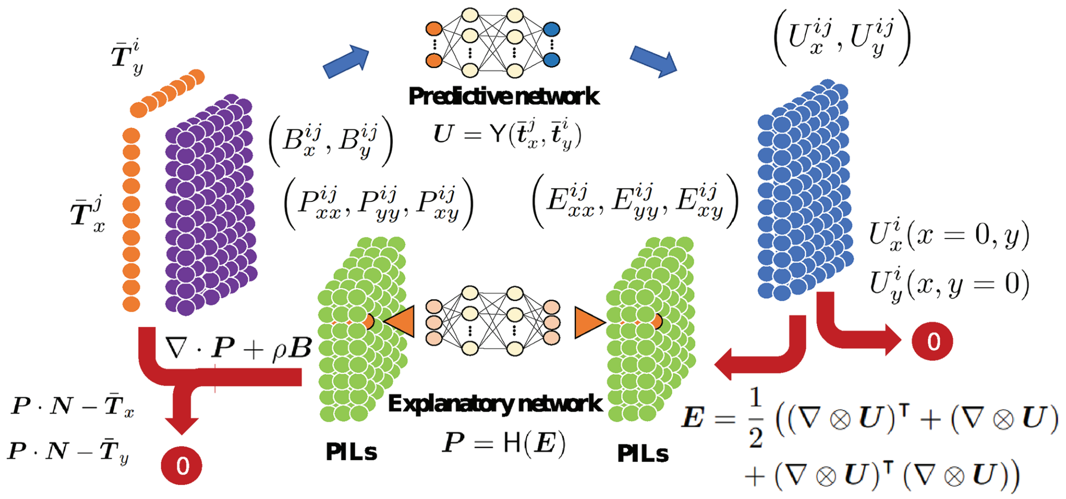
$\bar{T}_y^i$
$\bar{T}_x^j$
(B_x^{ij}, B_y^{ij})
$(P_{xx}^{ij}, P_{yy}^{ij}, P_{xy}^{ij})$
Predictive network
$U = \Upsilon(\bar{t}_x^j, \bar{t}_y^i)$
$(E_{xx}^{ij}, E_{yy}^{ij}, E_{xy}^{ij})$
(U_x^{ij}, U_y^{ij})
$U_x^i(x=0,y)$
$U_y^i(x,y=0)$
0
$\nabla \cdot P + \rho B$
$P \cdot N - \bar{T}_x$
$P \cdot N - \bar{T}_y$
0
PILs
Explanatory network
$P = \mathrm{H}(E)$
PILs
$E = \frac{1}{2}((\nabla \otimes U)^\top + (\nabla \otimes U) + (\nabla \otimes U)^\top (\nabla \otimes U))$

Figure 3. Graphical representation of the designed 2D-planar stress PGNNIV. All significant tensorial fiels of the problem are represented: the input variables (top and right tractions and volume forces, where the latter are assumed to be null so removed formally from the input), the output variables (displacement field at each nodal value), as well as the internal variables of the problem (stress and strain fields, represented in Voigt notation).

The ANN minimization problem reads therefore:

$$\min_{W} \mathrm{CF}(\mathcal{E}; W), \tag{28}$$

where W are the network parameters and $\mathcal{E} = \{\bar{T}^i, \bar{U}^i | i = 1, \cdots, N\}$ is a given training data-set. By minimizing this function (and assuring that ~~not~~ overfitting is observed by examining the predictions for test data), we will obtain predictions of displacement, stresses, and strains. For simplicity, Algorithm 1 details a stochastic gradient descent version of the optimization, even if in this work we always used the Adam optimizer.

From the theoretical point of view, equation (28) presents a complex constrained optimization problem that has been widely studied in the context of applied mathematics, i.e., Langrange multipliers. However, when NNs come into play along with PDEs, the optimization becomes more involved because the complex nature of the Pareto front, extensively studied for PINNs [64], determines that the optimum is a state where an individual loss cannot be further decreased without increasing at least one of the others, and therefore, the optimal set of weighting hyperparameters p_j cannot be inferred in advance. This weighting hyperparameters p_j, commonly referred as penalties, arise in a natural way if they are regarded as real numbers scaling the covariance matrix of the variables' *virtual* maximum likelihood probability distribution. This concept was introduced in the context of state-space particle dynamics [65,66].

Figure 3 shows a graphical representation of the different structures involved (tensorial fields) and the links between them (known and unknown operators) for finite strains solid mechanics.

3.2.4. Details about the discretization. The discretization of both space and time domains lies on the basis of numerical methods. In the particular case of solid mechanics under the hypotheses considered here, time discretization turns out not to be relevant for the overall computations because loads are applied in a quasi-static way, and the sole discretization of the geometry provides a very good approximation of how the continuum solid behaves.

Traditional FEMs follow a matrix-based approach to have algebraic systems whose solution is an approximate solution. By subdividing the whole domain into small parts (*finite elements*), PDEs governing the physical phenomena occurring in the particular geometry can be approximated by means of computable functions to generate algebraic systems even for complex geometries. However, FEMs require exact knowledge of the properties of the material and are usually time-consuming.

On the contrary, PGNNIVs require no information about the material properties since these are learned during the training process of the network, and the calculation time for the forward problem is reduced to seconds at prediction time in the online loop. The discrete nature of methods such as FEM, which subdivide the spatial domain in small elements, very closely resembles that of PGNNIVs, which comprise a number of

Algorithm 1: PGNNIV learning algorithm

```
Input: PGNNIV architecture, batch size $n_b$, penalties $p_k$, $k = 1, 2, 3, 4$, and number of iterations $M$;
Data: external forces $\bar{\mathtt{T}}^{(k)}$, measured displacements $\bar{\mathtt{U}}^{(k)}$, $k = 1, \dots, N$;
Initialization of PGNNIV parameters, $\boldsymbol{w} = \boldsymbol{w}^0$, $j = 0$;
repeat
    for $i = 1, \dots, n_b$ do
        $\mathtt{U}^{(i)} \leftarrow \mathsf{Y}\left(\bar{\mathtt{T}}^{(i)}; \boldsymbol{w}\right)$;/* Predictive network */
        $\mathtt{E}^{(i)} = \mathsf{KIN}\left(\mathtt{U}^{(i)}\right)$;/* Green-Lagrange or Cauchy strain tensor */
        $\mathtt{S}^{(i)} \leftarrow \mathsf{H}\left(\mathtt{E}^{(i)}; \boldsymbol{w}\right)$;/* Explanatory network */
    end
    $R_1^{(i)} = \sum_{i=1}^{n_b} \|\bar{\mathtt{U}}^{(i)} - \mathtt{U}^{(i)}\|^2$;
    $R_2^{(i)} = \sum_{i=1}^{n_b} \|\mathsf{DIV}(\mathtt{S}^{(i)})\|^2$;
    $R_3^{(i)} = \sum_{i=1}^{n_b} \|\mathtt{S}^{(i)} \cdot \boldsymbol{N} - \bar{\mathtt{T}}^{(i)}\|^2$;
    $R_4^{(i)} = \sum_{i=1}^{n_b} \|\mathtt{U}_x^{(i)}(0, y)\|^2 + \|\mathtt{U}_y^{(i)}(x, 0)\|^2$;
    $\mathrm{CF} = p_1 R_1^{(i)} + p_2 R_2^{(i)} + p_3 R_3^{(i)} + p_4 R_4^{(i)}$;
    $\boldsymbol{w} \leftarrow \boldsymbol{w} - \nabla_w \mathrm{CF}$; /* Stochastic gradient descent step */
    $j \leftarrow j + 1$;
until $j = M$;
Output: Optimal parameters $\boldsymbol{w}^* = \boldsymbol{w}$ for Y and H;
```

discrete units (neurons) to represent field variables. Moreover, commonly used differential operators in these methods are also subject of a suitable description in the PGNNIV framework using convolutional filters.

In particular, as illustrated in Figure 3, each neuron of the input or output layers of the predictive network corresponds to a node in the FEM framework, such that the specific layer is built by the concatenation of the neurons corresponding to the value of the variable in different discretization nodes and associated with the different spatial dimensions (x and y in that case). This way, data can be naturally considered in the neural structure and flow throughout the different models and loss functions. Similarly, the neurons in Figure 2 are associated with the values of strains and stresses in the different elements (with an identical structure, but this time using Voigt notation for tensor representation). The value of each neuron associated with a given element may be derived, using an appropriate convolution filter, from the values of the neurons associated to the element nodes.

For instance, it is possible to define the discrete gradient filter GRAD acting on the nodes for obtaining values on the elements or acting on the elements for obtaining values on the nodes. For example, if $\mathtt{W} = \mathsf{GRAD} \otimes \mathtt{U}$, then, for $k = 1, 2$,

$$\begin{aligned}\mathtt{W}[i,j|k,1|m] = \frac{1}{2h_x}\big(&\Delta_x\mathtt{U}[i,j+1|k|m]+\\ &+\Delta_x\mathtt{U}[i,j-1|k|m]\big),\end{aligned}$$

$$\begin{aligned}\mathtt{W}[i,j|k,2|m] = \frac{1}{2h_y}\big(&\Delta_y\mathtt{U}[i+1,j|1|m]+\\ &+\Delta_y\mathtt{U}[i-1,j|1|m]\big),\end{aligned}$$

where

$$\Delta_x\mathtt{U}[i,\cdot|1|m] = \mathtt{U}[i+1,\cdot|1|m] - \mathtt{U}[i-1,\cdot|1|m],$$

and

$$\Delta_y\mathtt{U}[\cdot,j|1|m] = \mathtt{U}[\cdot,j+1|1|m] - \mathtt{U}[\cdot,j-1|1|m].$$

Analogously, if $\mathtt{R} = \mathsf{GRAD} \cdot \mathtt{T}$, for $k = 1, 2$,

$$\begin{aligned}\mathtt{R}[i,j|k|m] = \frac{1}{2h_x}\big(&\Delta_x \mathtt{T}[i,j+1|k,1|m]+\\ &+\Delta_x \mathtt{T}[i,j-1|k,1|m]\big)+\\ +\frac{1}{2h_y}\big(&\Delta_y \mathtt{T}[i+1,j|k,2|m]+\\ &+\Delta_y \mathtt{T}[i-1,j|k,2|m]\big)\,.\end{aligned}$$

Now we can define the different discretized differentials. For instance the 2D-discretized gradient of a vector field $\boldsymbol{V}$, represented by the array $\mathtt{V}$, is $\mathtt{DV} = \mathsf{GRAD} \otimes \mathtt{V}$, where GRAD is the discrete gradient operator. Therefore, we compute $\mathtt{DU} = \mathsf{GRAD} \otimes \mathtt{U}$ and for large strains, we have $\mathtt{E} = \frac{1}{2}(\mathtt{DU} + \mathtt{DU}^\mathsf{T} + \mathtt{DU}^\mathsf{T}\mathtt{DU})$ and for small strains $\mathtt{E} = \frac{1}{2}(\mathtt{DU} + \mathtt{DU}^\mathsf{T})$.

Similarly, the 2D-discretized divergence of a tensor field $\boldsymbol{T}$, represented by $\mathtt{T}$, is defined as: $\mathsf{DIV}(\mathtt{T}) = \mathsf{GRAD} \cdot \mathtt{T}$, so we can express the equilibrium equation both in infinitesimal as well as finite strains theory as $\mathsf{DIV}(\sigma) = \mathtt{0}$ or $\mathsf{DIV}(\mathtt{P}) = \mathtt{0}$, respectively.

3.3. Data generation and training process

In this work, we generate synthetic data although the methodology is the same for real experimental data. Synthetic data generation is more practical, as it allows for error validation by comparing to the real solutions and inexpensive, given that an accurate numerical solution is available via FEM analysis. Therefore, each instance has to be interpreted as a particular load case sampled randomly and independently, following a given multivariate distribution.

3.3.1. Small strains: linear, softening, and hardening elastic materials.

For the creation of the linear, softening, and hardening materials, we used Matlab, in co-simulation with Abaqus CAE/6.14-2. Matlab was used to automatically and iteratively generate an Abaqus input file that contained the geometry and load profiles. Once each numerical simulation is completed, the results are stored back into Matlab. For all ~~the~~ test cases, the geometry is the same, whereas the variability in the data-set is achieved by randomly changing the load profiles so that all the examples correspond with different experiments. For this first example, the load profiles are parabolic and are generated for both the right and top contours.

We consider three elastic materials with different constitutive laws. On one hand, we choose an isotropic linear elastic material with elastic modulus $E = 1000$ Pa and Poisson's coefficient $\nu = 0.3$. On the other hand, we considered two nonlinear materials, one with softening properties and another one with hardening properties.[6] A relation of the type $\sigma = K\varepsilon^n$ was used, with $K = 1.869 \times 10^1$ Pa and $n = 0.45$ for the softening material and $K = 1.869 \times 10^{12}$ Pa and $n = 3.5$ for the hardening material. The data-set comprises $N = 10^3$ FEM-simulations for the linear material and $N = 10^4$ FEM-simulations for the hardening and softening materials. The details about the different data-sets used are summarized in Table 3.

3.3.2. Finite strains: Ogden-like hyperelastic material.

For the finite strains case, an incompressible Ogden-like hyperelastic material of order 3 is used for the dataset generation. The incompressible Ogden [67] hyperelastic material of order m is defined in terms of its strain-energy density function:

$$\Psi(\boldsymbol{C}) = \Psi(\lambda_1, \lambda_2, \lambda_3) = \sum_{i=p}^{m} \frac{\mu_p}{\alpha_p}\left(\lambda_1^{\alpha_p} + \lambda_2^{\alpha_p} + \lambda_3^{\alpha_p} - 3\right). \tag{29}$$

For reproducibility purposes, in Table 2, we report the material parameters used for the data generation. For this second example, we produce $N = 10^4$ examples corresponding to uniform biaxial tests (Table 3), where $\lambda_1, \lambda_2 \in [1; 1.10]$. For an incompressible membrane under biaxial deformation, assuming a plane stress state, the

Table 2. Parameter values for the Ogden hyperelastic material.

Parameter	Value	Units
μ_1	281	Pa
μ_2	−280	Pa
μ_3	0.31	Pa
α_1	1.66	-
α_2	1.61	-
α_3	38.28	-

Table 3. Summary of the different data-sets used.

Problem	Load profile	N	Train/test
Small strains (linear)	Parabolic	10^3	80%/20%
Small strains (hardening)	Parabolic	10^4	80%/20%
Small strains (softening)	Parabolic	10^4	80%/20%
Large strains (Ogden)	Uniform	10^4	80%/20%

solution of the problem using an Ogden's strain-energy function has analytical solution [68]. The deformation field corresponding to uniform biaxial deformations is

$$\begin{aligned}\xi_1 &= \chi_1(\Xi_1, \Xi_2, \Xi_3) = \lambda_1 \Xi_1,\\ \xi_2 &= \chi_2(\Xi_1, \Xi_2, \Xi_3) = \lambda_2 \Xi_2,\\ \xi_3 &= \chi_3(\Xi_1, \Xi_2, \Xi_3) = \frac{1}{\lambda_1 \lambda_2}\Xi_3,\end{aligned}$$

where we have denoted $(\Xi_1, \Xi_2, \Xi_3) = (X, Y, Z)$ and $(\xi_1, \xi_2, \xi_3) = (x, y, z)$ according to the previously introduced notations, so, the non-vanishing components of the Green–Lagrange deformation tensor, $\boldsymbol{E}$, are

$$\begin{aligned}E_{xx} &= \frac{1}{2}(\lambda_1^2 - 1),\\ E_{yy} &= \frac{1}{2}(\lambda_2^2 - 1),\\ E_{zz} &= \frac{1}{2}((\lambda_1\lambda_2)^{-2} - 1).\end{aligned}$$

As we assume plane stress, ($\sigma_{zz} = \sigma_{xz} = \sigma_{yz} = 0$), the non-vanishing components of the first-order Piola–Kirchhoff stress tensor are

$$\begin{aligned}P_{xx} &= \frac{1}{\lambda_1}\sum_{k=1}^{3}\mu_k\left(\lambda_1^{\alpha_k} - (\lambda_1\lambda_2)^{-\alpha_k}\right),\\ P_{yy} &= \frac{1}{\lambda_2}\sum_{k=1}^{3}\mu_k\left(\lambda_2^{\alpha_k} - (\lambda_1\lambda_2)^{-\alpha_k}\right).\end{aligned}$$

3.3.3. Training process. For the evaluation of the methodology, we have trained different PGNNIVs corresponding to four cases. Each case is presented as a different numerical experiment (E).

- Linear material with parametric explanatory network (anisotropic (E1) and isotropic (E2)).
- Linear (E3), softening (E4), and hardening (E5) materials with non-parametric explanatory network.
- Ogden-like material with parametric explanatory network (E6).
- Ogden-like material with non-parametric explanatory network (E7).

Table 4. Value of the hyperparameters for all the training processes.

Description	Experiment	Loss function coefficients				Learning algorithm	
		p_1	p_2	p_3	p_4	Learning rate α	Batch size n_b
Linear (anisotropic model)	E1	1×10^3	1×10^{-3}	1×10^{-3}	1×10^3	5×10^{-5}	100
Linear (isotropic model)	E2	1×10^3	1×10^{-3}	1×10^{-3}	1×10^3	5×10^{-5}	100
Linear (model-free)	E3	1×10^3	1×10^{-3}	1×10^{-3}	1×10^3	5×10^{-5}	100
Softening (model-free)	E4	1×10^3	1×10^{-3}	1×10^{-3}	1×10^3	5×10^{-5}	100
Hardening (model-free)	E5	1×10^4	1×10^{-8}	1×10^{-8}	1×10^4	5×10^{-4}	100
Ogden (model-free)	E6	1.0×10^0	1×10^{-3}	1×10^{-3}	1×10^{-3}	1×10^{-3}	128
Ogden (model-based)	E7	1.0×10^0	1×10^{-3}	1×10^{-3}	1×10^{-3}	1×10^{-3}	128

For all the datasets considered, there is a number of hyperparameters that have been tuned for obtaining the proceeding results with the different networks, namely the learning rate β and the four penalty coefficients p_j, $j = 1, \dots, 4$. For reproducibility purposes, Table 4 shows a summary of the hyperparameters that were found to give optimal results for the optimization process.

4. Results

When used as forward solvers, PGNNIVs can either predict measurable variables if force–displacement data are available, for example, through digital image correlation (DIC) techniques, or explain the internal state of the solid if this is needed for a certain application. In this section, we validate the performance of PGNNIVs acting as a forward-solver against standard FEM solutions for the plate using the different materials proposed and also as a method for constitutive equation discovery.

4.1. Predictive capacity

4.1.1. Infinitesimal strains case. We first evaluate the prediction capacity of PGNNIVs for the different tested materials under random parabolic loads. For a quantitative evaluation of the predictive capacity of the PGNNIV, we define the relative error (ϵ_r) of an array field $\mathtt{I}$ as:

$$\epsilon_r(\mathtt{I}) = \frac{\sum_{I,J,K} \left(\hat{\mathtt{I}}[I|J|K] - \mathtt{I}[I|J|K]\right)^2}{\sum_{I,J,K} \mathtt{I}[I|J|K]^2}, \tag{30}$$

where $\hat{\mathtt{I}}$ is the predicted value, and $\mathtt{I}$ is the value obtained using FEM. For instance, for the displacement field $\boldsymbol{U}$ represented by the array $\mathtt{U}$:

$$\epsilon_r(\mathtt{U}) = \sqrt{\frac{\sum_{i,j,k,l} \left(\hat{\mathtt{U}}[i,j|k|l] - \mathtt{U}[i,j|k|l]\right)^2}{\sum_{i,j,k,l} \mathtt{U}[i,j|k|l]^2}}, \tag{31}$$

or using standard field notation:

$$\epsilon_r(\boldsymbol{U}) = \sqrt{\frac{\sum_{i,j,k,l} \left(\hat{U}_k^{(l)}(x_i, y_j) - U_k^{(l)}(x_i, y_j)\right)^2}{\sum_{i,j,k,l} (U_k^{(l)}(x_i, y_j))^2}}, \tag{32}$$

where $U_k^{(l)}(x, y)$ represents the value of $U_k(x, y)$ corresponding to the data sample l. Table 5 shows the relative error obtained by the PGNNIV for the softening, hardening, and linear materials. We observe predictive errors below 6% for all the cases.

4.1.2. Finite strains case. We evaluate now the Ogden material under finite strains. Table 6 shows the relative error obtained by the PGNNIV for the Ogden material, resulting in that case in errors below 2% (due to the lower dataset variability).

Table 5. PGNNIV predictive errors for the linear, softening, and hardening materials (experiments E3, E4 and E5).

Type of material	$\epsilon_r(\boldsymbol{U})$	$\epsilon_r(\boldsymbol{\sigma})$	$\epsilon_r(\boldsymbol{\varepsilon})$
Linear	0.0053	0.0321	0.0361
Softening	0.0092	0.0207	0.0455
Hardening	0.0117	0.0448	0.0551

Table 6. PGNNIV predictive errors for the hyperelastic material (experiments E6 and E7).

Type of material	$\epsilon_r(\boldsymbol{U})$	$\epsilon_r(\boldsymbol{\sigma})$	$\epsilon_r(\boldsymbol{\varepsilon})$
Ogden material	0.0034	0.0156	0.0037

It should be noted that, unlike other methods such as PINNS or the EUCLID method, the error has been calculated for variable and varying boundary conditions.

4.2. Explanatory capacity

The explanatory capacity of the method is evaluated here in two levels: parametric identification and state model unveiling.[7]

4.2.1. Infinitesimal strains. First, we train and evaluate a linearized version of the PGNNIV (where Y and H are replaced with weight matrices with no activation functions) with the dataset corresponding to the isotropic linear elastic material parametrized with elastic modulus E and Poisson ratio ν. It is possible to evaluate the relative error when learning an elastic tensor $\boldsymbol{D}$ by means of

$$\epsilon_r(\boldsymbol{D}) = \frac{\|\hat{\boldsymbol{D}} - \boldsymbol{D}\|_\mathrm{F}}{\|\boldsymbol{D}\|_\mathrm{F}}, \tag{33}$$

where $\hat{\boldsymbol{D}}$ is the predicted value, $\boldsymbol{D}$ the real one, and $\|\cdot\|_\mathrm{F}$ depicts the Fröbenius norm, $\|\boldsymbol{A}\|_F = \sqrt{\boldsymbol{A}:\boldsymbol{A}}$. For assessing the identifiability of the structural parameters-, it is possible to evaluate the prediction of the model parameters associated with equations (18) and (19) by computing the relative errors for a given parameter λ, that are defined as:

$$\epsilon_r(\lambda) = \frac{|\hat{\lambda} - \lambda|}{\lambda}, \tag{34}$$

where $\hat{\lambda}$ and λ are the predicted and real values, respectively. The learned anisotropic and isotropic tensors, up to a precision of 1 Pa are:

$$\boldsymbol{D}_\mathrm{aniso} = \begin{pmatrix} 1099 & 330 & 0 \\ 330 & 1099 & 0 \\ 0 & 0 & 511 \end{pmatrix},$$
$$\boldsymbol{D}_\mathrm{iso} = \begin{pmatrix} 1098 & 331 & 0 \\ 331 & 1098 & 0 \\ 0 & 0 & 767 \end{pmatrix}.$$

The errors of the elastic tensor when using the anisotropic or the isotropic elastic material model are respectively $\varepsilon_r(\boldsymbol{D}_\mathrm{aniso}) = 0.0144$ and $\varepsilon_r(\boldsymbol{D}_\mathrm{iso}) = 0.002$, i.e, when the assumed hypotheses are true, the explanatory capacity of the network increases. Notwithstanding, the general anisotropic model has a certain explanatory capacity, as we may detect the supplementary structural symmetries in the resulting elastic tensor, that is, $d_{13}, d_{23} \ll d_{11}, d_{12}, d_{22}, d_{33}$, $d_{11} \simeq d_{22}$. The last symmetry condition $d_{33} = d_{11} - d_{12}$ is not fulfilled (and the value of the parameter d_{33} itself is not accurately predicted), as the dataset is not sufficiently rich in large pure shear-stress states, so the model is not able to detect this symmetry. This last remark is important and illustrates one of the weaknesses of the method (indeed, of any ML approach): an excess of model degrees of

Table 7. Predicted and real values of the model parameters (experiments E1 and E2).

Parameter, λ	Relative error, $\epsilon_r(\lambda)$
Anisotropic model	
d_{11}	0.0002
d_{12}	0.0008
d_{13}	∞
d_{22}	0.0002
d_{23}	∞
d_{33}	0.3359
Isotropic model	
E	0.0020
ν	0.0052

Table 8. Explanatory errors for the H model subjected to uniform uniaxial test (experiments E3, E4, and E5).

Type of material	$\epsilon_r(\mathrm{H})$
Linear	**0.0302**
Softening	**0.0096**
Hardening	**0.0592**

freedom (parameters) can result in poor explanatory capacity especially if the dataset does not have sufficient variability [69].

Moving to specific structural parameter identification, Table 7 describes how the model is able to predict accurately the different elastic parameters. As commented before, the worst prediction is observed for the anisotropic model and the parameter that correlates shear stresses and strains. If, using the anisotropic model, we were interested in finding the value of the isotropic model parameters E and ν, it is possible to compute ν using the relations $\nu = \frac{d_{12}}{d_{11}} = \frac{d_{12}}{d_{22}}$, and then to compute E using $E = d_{11}(1-\nu^2) = d_{22}(1-\nu^2) = d_{12}\frac{1-\nu^2}{\nu} = d_{33}(1+\nu)$. Of course, we obtain a different accuracy for each of these expressions.

Using the anisotropic model, we obtain values of 0.001 and 0.001 for $\epsilon_r(\nu)$ and values of 0.0004, 0.0004, 0.004, and 0.34 for $\epsilon_r(E)$, in agreement with the previous observations.

While the predictive capacity of PGNNIVs does not necessarily surpass that of a classical (unconstrained) NN, the significant improvement is visible when assessing the explanatory capacity of the PGNNIV, which can be evaluated by its ability to learn the material constitutive law. This also distinguishes the proposed method from others such as using PINNs for parameter identification. We perform a virtual uniaxial test using the explanatory network, which corresponds to the functional representation of $\sigma_{xx} = \mathsf{H}_1(\varepsilon, 0, 0)$ for $\varepsilon \in [\varepsilon_{\min}; \varepsilon_{\max}]$ and we compare the PGNNIV predictions with a virtual uniaxial test produced with FEM. Results are shown in Figure 4 and Table 8.

The explanatory error is quantified as the normalized area confined between the real uniaxial test curve and the PGNNIV-predicted one in Figure 4. It is expressed as:

$$\epsilon_r(\mathsf{H}) = \sqrt{\frac{\int_{\varepsilon_{\min}}^{\varepsilon_{\max}} (\hat{\sigma}_{xx}(\varepsilon) - \sigma_{xx}(\varepsilon))^2 \, \mathrm{d}\varepsilon}{\int_{\varepsilon_{\min}}^{\varepsilon_{\max}} \sigma_{xx}^2(\varepsilon) \, \mathrm{d}\varepsilon}}, \tag{35}$$

where $\hat{\sigma}_{xx}$ and σ_{xx} are the predicted and FEM stresses, respectively, which result from strains $\varepsilon \in [\varepsilon_{min}; \varepsilon_{max}]$.

4.2.2. Finite strains. We explore now the explanatory capacity for the finite strains case. If $\lambda \in [\lambda_{\min}; \lambda_{\max}]$ is the longitudinal stretch, λ_1, the relative explanatory errors for a given transversal stretch λ_2 are defined as:

$$\epsilon_{r,ij}(\mathsf{H}) = \sqrt{\frac{\int_{\lambda_{\min}}^{\lambda_{\max}} (\hat{P}_{ij}(\lambda, \lambda_2) - P_{ij}(\lambda, \lambda_2))^2 \, \mathrm{d}\lambda}{\int_{\lambda_{\min}}^{\lambda_{\max}} P_{ij}^2(\lambda, \lambda_2) \, \mathrm{d}\lambda}}. \tag{36}$$

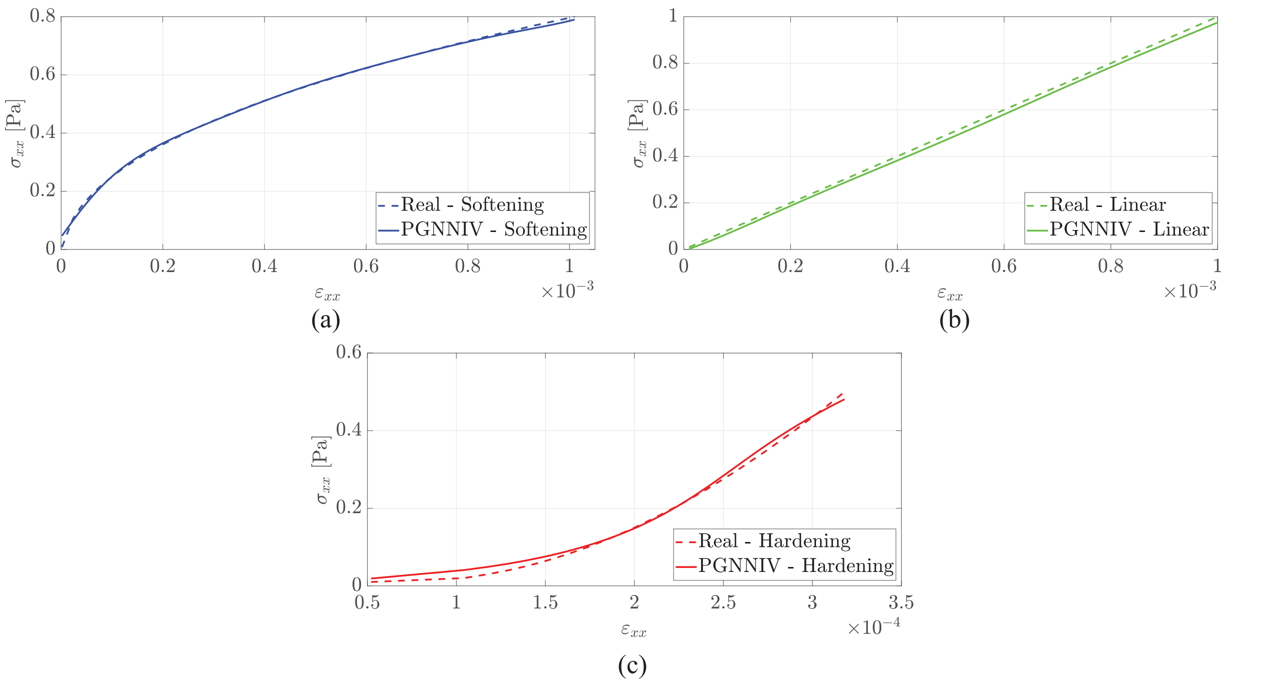

Figure 4. PGNNIV prediction versus FEM solution of the uniaxial test curve for the different datasets (experiments E3, E4, and E5). We observe good agreement between the FEM solution (continuous line) and the PGNNIV prediction (dashed line), for the softening (a), linear (b), and hardening (c) materials.

Note that, as the roles of x and y (represented by i and j) are symmetrical in the considered biaxial test, the indicated errors are sufficient for illustrating the explanatory capacity of the method. For structural parameter identification, the formula used for error quantification is equation (34).

As explained previously, we first explicitly state the parametric shape of the constitutive equation, that is, we prescribe the material to be Ogden-like. Under these assumptions and for the uniform biaxial test considered, the constitutive relation writes

$$P_{xx} = \frac{1}{\sqrt{2E_{xx}+1}} \sum_{p=1}^{3} \mu_k \left[(2E_{xx}+1)^{\alpha_k/2} - \kappa^{-\alpha_k/2} \right],$$

$$P_{yy} = \frac{1}{\sqrt{2E_{yy}+1}} \sum_{p=1}^{3} \mu_k \left[\left(2E_{yy}+1\right)^{\alpha_k/2} - \kappa^{-\alpha_k/2} \right].$$

where $\kappa = (2E_{xx}+1)(2E_{yy}+1)$. Therefore, the parameters α_k, μ_k, $k = 1, 2, 3$ are, in principle, the ones that ought to be learned by the explanatory network. We obtain values for the parameters of $\mu_1 = 276\,\text{Pa}$, $\mu_2 = -277\,\text{Pa}$, $\mu_3 = 0.31\,\text{Pa}$, $\alpha_1 = 1.53$, $\alpha_2 = 1.47$, and $\alpha_3 = 38.32$. The relative errors for the different parameters are shown in Table 9. It is important to note that there are some parameters that are more accurately predicted than others, i.e., they are superfluous. This fact relies on the capacity of each of the parameters for explaining the material response, as observed in Figure 5, where we compare the theoretical constitutive relation with the one obtained using the learned parameters. The explanatory errors are reported in Table 10, adding evidence of the explanatory power of the method despite the discrepancies in some parameters.

We now evaluate the model discovered by the PGNNIV with a virtual biaxial test using the explanatory network. The functional representation is now $(P_{xx}, P_{yy}, P_{xy}) = \mathsf{H}(E_{xx}, E_{yy}, 0)$ for $E_{xx}.E_{yy} \in [E_{\min}; E_{\max}]$, and we compare the PGNNIV predictions with the model used for the data generation. The results are shown in Figure 6 for three different values of λ_2, and the errors, computed according to equation (36) are displayed in Table 11.

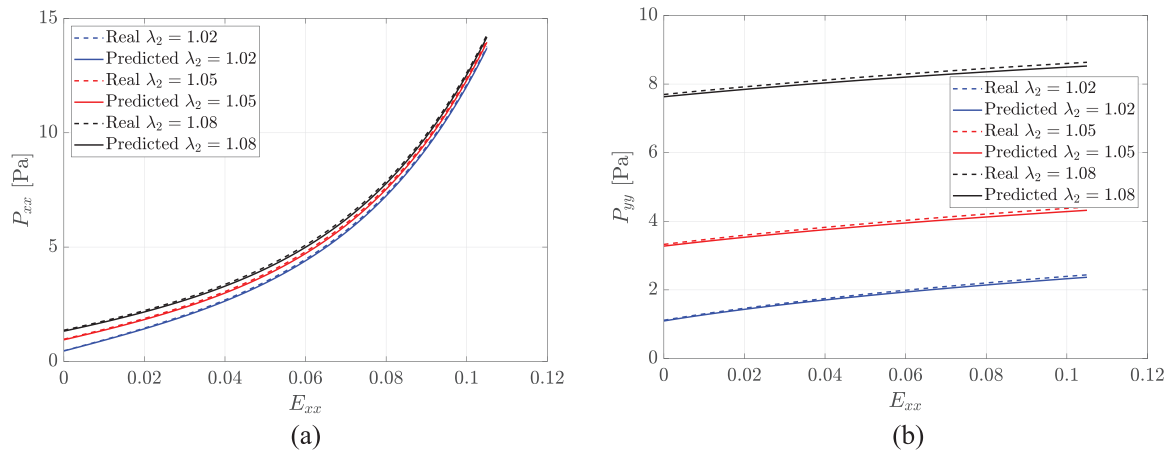

Figure 5. Parametric PGNNIV prediction (experiment E6) versus analytic solution of the uniform biaxial test curve for the Ogden material. We observe good agreement between the analytical solution (continuous line) and the PGNNIV prediction (dashed line), for the different values of λ_2. This indicates that the network has a good ~~explicability~~ capacity even though some superflous model parameters are not accurately fitted. (a) Component yy. (b) Component xx.

Table 9. Predicted and real values of the model parameters for the Ogden material (experiment E6).

Parameter, λ	Relative error, $\epsilon_r(\lambda)$
μ_1	0.017
μ_2	−0.011
μ_3	0.005
α_1	0.077
α_2	0.086
α_3	0.001

Table 10. Explanatory errors $\epsilon_{r,ij}(\mathrm{H})$ for the finite strains problem subjected to uniform biaxial test (parametric discovery, experiment E5).

Y stretch ratio	Relative error, $\epsilon_r(\mathrm{H})$	
	$\epsilon_{r,xx}$	$\epsilon_{r,yy}$
$\lambda_2 = 1.02$	0.0098	0.0265
$\lambda_2 = 1.05$	0.0115	0.0193
$\lambda_2 = 1.08$	0.0132	0.0110

Table 11. Explanatory errors $\epsilon_{r,ij}(\mathrm{H})$ for the finite strains problem subjected to uniform biaxial test (non-parametric discovery, experiment E7).

Y stretch ratio	Relative error, $\epsilon_r(\mathrm{H})$	
	$\epsilon_{r,xx}$	$\epsilon_{r,yy}$
$\lambda_2 = 1.02$	0.0034	0.0357
$\lambda_2 = 1.05$	0.0033	0.0177
$\lambda_2 = 1.08$	0.0039	0.0152

5. Discussion, conclusions, and future work

Throughout this work, we have presented the mathematical foundations of PGNNIVs in the field of computational solid mechanics, which demonstrates to be a particularly interesting niche for the use of such methodology. We have demonstrated that PGNNIVs have both predictive and explanatory capacity:

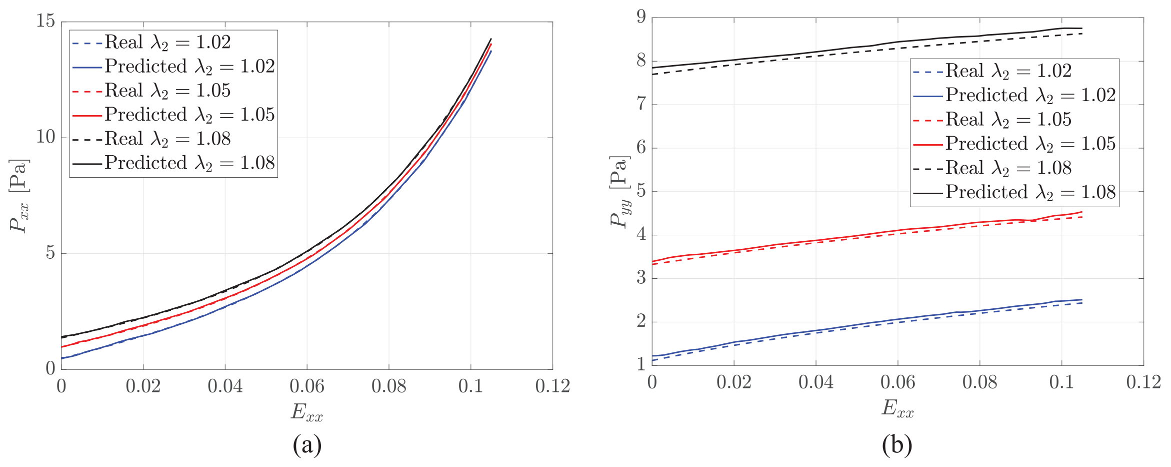

Figure 6. Non-parametric PGNNIV prediction (experiment E7) versus analytic solution of the biaxial test curve for the Ogden material. We observe good agreement between the FEM solution (continuous line) and the PGNNIV prediction (dashed line), for the different values of λ_2. (a) Component xx. (b) Component yy.

- **Predictive capacity**: PGNNIVs are able to accurately predict the solid response to new external stimuli in real time, something fundamental for optimization, control and probabilistic problems. They are also able to predict not only the solid response in terms of displacement field but also the deformation and stress fields, without the need of any extra post-processing. This has been demonstrated ~~in~~ when evaluating predictive capacity, where we have obtained relative errors always below 10%. Controlling non-primary fields is sometimes important in engineering problems, as high stresses cause damage, plasticity, or structural failure. As the explanatory network, once trained, encodes all the information about the material properties, it can be used for the prediction of stresses directly from the displacement fields, if necessary.
- **Explanatory capacity:** PGNNIVs are able to unveil hidden state model equations, that is, the constitutive equations of computational solid mechanics. First, for parameter identification and fitting, PGNNIV are able to identify inherent material symmetries (such as isotropy) and also to predict the value of the structural model parameters with high accuracy. In the latter, PGNNIVs are in a certain sense an alternative to conventional least-square minimization problems [70] (e.g. using standard methods such as Levenberg–Marquardt algorithm) but making use of the software and hardware tools associated with ANN technology: graphical processor units (GPUs) and tensor processor units (TPUs), distributed and cloud computation, scalability, transfer, and federate learning strategies among others. In addition, PGNNIVs address the more challenging problem of model-free unravelling of nonlinear materials constitutive laws. We have demonstrated the explanatory capacity of PGNNIVs both for parameter identification and state model discovery with many examples (linear and nonlinear materials both in the infinitesimal and finite strains framework). The relative error when predicting structural parameters is always below 2% except if the dataset does not contain information about the material response to stimuli associated with a certain parameter or the parameter itself has not a direct impact in the explanatory capacity (superfluous parameters). Besides, the explanatory relative error is also below 10% for all the cases analyzed.

One important characteristic of PGNNIVs related to this double capacity is the fact that it is possible to decouple two sources of variability in the data obtained from a mechanical system that can be measured using any kind of sensor, namely, the stimuli variability and the variability related to system response.

- The stimuli variability is stored in the predictive network, which acts as an autoencoder. The encoder is able to map data that lives in a space of dimension D, to a latent space of dimension $d \ll D$. The size of the latent space is therefore informative about the data variability, and the values of the latent variables are a compressed representation of data. For instance, in the large deformations problem, the data input was in a space of dimension $D = 18$ (see Table 1), whereas the latent space has dimension $d = 3$. In principle, with a latent space of dimension $d = 2$ would have been enough, as the data variability has two components (horizontal and vertical pressure), but we kept a different latent space for illustrating that the methodology would be satisfactory for any d greater than this intrinsic number of degrees of freedom, generally unknown. In addition to theoretical considerations, this fact has important practical consequences: if we know the sources of variability of our system, it is possible to design the predictive network accordingly.

- The physical interpretable knowledge, that is, the constitutive equation of the material, is delocalized, spread and diluted in the weights of the predictive network but is also encoded in the relation $\boldsymbol{\varepsilon} \mapsto \boldsymbol{\sigma}$ or $\boldsymbol{E} \mapsto \boldsymbol{P}$ that is learned by the explanatory network, in a much more structured way. Indeed, if it is intended to identify and adjust some structural parameters, this is possible. Otherwise, the constitutive model as a whole may be also unravelled using the expressiveness of ANN methods.

In that sense, PGNNIVs are knowledge generators from data as most of ML techniques are, with the difference that for this particular method, the physical knowledge is directly distilled in a separated component. This particularity is what makes the difference between PINNs and PGNNIVs. In PINNs, data and mathematical physics models are seamlessly integrated for solving parametric PDEs of a given problem [3, 26], but, by construction, the information cannot be extrapolated to other situations. Which means that, in the context of solid mechanics, the network trained for a given problem using PINNs cannot be used for predicting the response of the system under different volume loads or boundary conditions, which greatly weakens its ability as a predictive method. PGNNIVs overcome this difficulty precisely by distilling the physical information from the intrinsic variability of the stimuli.

There is another paramount characteristic of PGNNIVs for computational solid mechanics that has been largely discussed and has explained their emergence. There is no need to have access to the values of internal variables, that are, strictly speaking, non-measurable as they are mathematical constructs coming from a scientific theory. In that sense, even if the bases for thermodynamically appropriated ANN for constitutive equations in solid mechanics have been investigated [71], it is important to recall that stress fields, such as $\boldsymbol{P}$ or $\boldsymbol{\sigma}$ are not accessible without the need of extra hypothesis (geometry or load specific configurations). This fundamental issue should not be overlooked and many recent works have worked for this purpose. Efficient Unsupervised Constitutive Law Identification and Discovery (EUCLID) method is one of the most acclaimed efforts in that direction, either using sparse identification [38, 72, 73], clustering [74], Bayesian methods [75], or ANN [40]. However, EUCLID paradigm relies on the fact that the geometry and loads are appropriate enough to ensure strain-stress fields variability in a single specimen. When the geometry or the data acquisition capabilities do not satisfy this requirement, the only possibility is to take action on the dataset or the network, or at least to track the different load conditions and incorporate them to the computational pipeline.

Finally, among the different PIML methods, PGNNIVs are more transparent than other approaches that have demonstrated to be very performant for computing the evolution of dynamical systems by incorporating thermodynamical constraints. Structure preserving neural networks enforce first and second laws of thermodynamics as regularization term [76]. Even if in the cited work the GENERIC structure allows for a split between reversible and non-reversible (dissipative) components, the physical information is again diluted in all the network weights, rather than in some specific components. A dimensionality reduction of the dynamic data was also explored [77]. This may be interpreted also as an information reduction to distil physical knowledge, although the interpretability is still dark. In PGNNIVs, however, interpretable physical information (that is, knowledge) is located in specific ANN components or, if explicit parametric relations are formulated, in specific model parameters.

Nevertheless, the presented methodology still has some limitations and there exists room for exploration in several directions:

1. The data requirements for the problem in hands are high. In this work, we have generated data synthetically, but in reality, sensors usually collect noisy data from experimental tests and there exist also important limitations concerning the size of the datasets. A probabilistic (Bayesian) viewpoint will enable a new interpretation of PGNNIVs in the *small data* regime, although this methodology is rather thought for systems where intensive data mining is possible, in which data quantity prevails over data quality.
2. Defining a suitable architecture for the Y and H networks is not a simple task and requires an iterative process, including the tuning of many hyperparameters. In addition, PGNNIVs require extra hyperparameters, i.e., penalty coefficients p_i, making the process even more involved and time consuming. We have presented some insights into to the complexity and architecture of both predictive and explanatory networks, related with both their prediction and explanation character but either a great intuition for network design, or time-intensive trial-and-error iterative testing are needed.

3. In this work, we have made used of relatively coarse discretizations, but finer meshes will result in more expensive training processes due to the exponentially larger number of parameters required. More powerful computational strategies (distributed computing, parallelization) as well as more advanced hardware (GPUs and TPUs) will enable the acceleration of the PGNNIVs training processes although the problem of dealing with multidimensional and unstructured meshes still remains open.

Many challenges lie ahead of the development of a more general PGNNIV framework. Next lines of research will address the formulation of PGNNIVs under finite strain assumptions using a much more theoretical basis, such as the presented in some recent works [71, 78, 79], leveraging its predictive and explanatory power. Furthermore, extensions of the 2D planar stress architecture to general 3D problems with more complex geometries and load scenarios as well as to more complex constitutive laws that might depend on time (visco-elasticity), or heterogeneous conditions, still pose major challenges for the future.

Finally, the usage of real data from sensors, for example, through digital image or volume correlation tests (DIC, DVC) [63] or even more advanced methods such as finite-element model updating (FEMU) [80] or virtual fields methods (VFM) [81] will put to the test the applicability of PGNNIVs to real scientific problems in the field of engineering.

In conclusion, we have demonstrated that in the context of computational solid mechanics, PGNNIVs are a family of ANN for accurately predicting measurable and non-measurable variables such as displacement and stress fields, in real time, and are also able to describe or unravel the constitutive model with high accuracy for different linear and nonlinear (hyper-)elastic materials. Even if this work is preliminary, the ingredients that it comprises correspond to a general approach and the methodology can be applied to cases of scientific interest with the necessary adaptations of the network architectures.

Acknowledgements

The authors gratefully acknowledge Eduardo Orera-Echeverría for his contributions to the figure visualization and code preparation. J. A. J. gratefully acknowledges the financial support from the Government of Aragon (DGA).

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

ORCID iD

Jacobo Ayensa-Jiménez https://orcid.org/0000-0003-2564-6038

Notes

1. These terms are also known as as *collocation* or *regularization losses* in the scientific literature.
2. It is important to recall that, as essential boundary conditions can be measured as an output variable, it is not necessary to include them as inputs of our problem.
3. More sophisticated approaches coming from Manifold Learning theory are possible for analyzing data dimensionality and structure [82], but this is not the main interest of this work and therefore is out of the scope of this particular study.
4. It is important to note that a single, complex enough, autoencoder-like DNN would be able to represent the data variability even in the more complex scenario (the case with the largest latent space in the process of data generation). However, we have decided to use two different network architectures for illustrating this particular feature of PGNNIV: the predictive network is associated to data variability, rather than data nature. Therefore, we can adapt the network architecture to our problem characteristics, aiming either at a better network performance (avoiding overfitting) or at a lower computational cost.
5. Note that, in TensorFlow framework, this function was implemented as a convolutional filter using a window of size 1, for the sake of computational efficiency.
6. In Abaqus, they were modeled as plastic materials with no discharging effects caused by the removal of the load and strain ranges confined within very small values, that allowing for the compliance of the nonlinear constitutive law with the infinitesimal strains hypothesis.

7. As discussed later, even if parametric identification can be achieved with classical fitting methods, optimization algorithms are not robust when dealing with noisy and unstructured data, and ANN-specific hardware and software tools are not available.

References

[1] Bishop, CM, and Nasrabadi, NM. *Pattern recognition and machine learning* (Vol. 4). Singapore: Springer, 2006.
[2] Cueto, E, and Chinesta, F. Thermodynamics of learning physical phenomena. *Arch Comput Methods Eng* 2023; 30 : 4653–4666.
[3] Karniadakis, GE, Kevrekidis, IG, Lu, L, et al. Physics-informed machine learning. *Nat Rev Phys* 2021; 3(6): 422–440.
[4] Schmidt, M, and Lipson, H. Distilling free-form natural laws from experimental data. *Science* 2009; 324(5923): 81–85.
[5] Brunton, SL, Proctor, JL, and Kutz, JN. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proc Natl Acad Sci USA* 2016; 113(15): 3932–3937.
[6] Udrescu, SM, and Tegmark, M. AI Feynman: a physics-inspired method for symbolic regression. *Sci Adv* 2020; 6(16): eaay2631.
[7] Chinesta, F, and Cueto, E. Empowering engineering with data, machine learning and artificial intelligence: a short introductive review. *Adv Model Simul Eng Sci* 2022; 9(1): 21.
[8] Kirchdoerfer, T, and Ortiz, M. Data-driven computational mechanics. *Comput Methods Appl Mech Eng* 2016; 304: 81–101.
[9] Ayensa-Jiménez, J, Doweidar, MH, Sanz-Herrera, JA, et al. An unsupervised data completion method for physically-based data-driven models. *Comput Methods Appl Mech Eng* 2019; 344: 120–143.
[10] Greydanus, S, Dzamba, M, and Yosinski, J. Hamiltonian neural networks. *Adv Neural Inf Process Syst* 2019; 32.
[11] Jin, P, Zhang, Z, Zhu, A, et al. Sympnets: intrinsic structure-preserving symplectic networks for identifying hamiltonian systems. *Neural Netw* 2020; 132: 166–179.
[12] Cybenko, G. Approximations by superpositions of a sigmoidal function. *Math Control Signals Syst* 1989; 2: 183–192.
[13] Hornik, K. Approximation capabilities of multilayer feedforward networks. *Neural Netw* 1991; 4(2): 251–257.
[14] Pinkus, A. Approximation theory of the MLP model in neural networks. *Acta Numer* 1999; 8: 143–195.
[15] Lu, Z, Pu, H, Wang, F, et al. The expressive power of neural networks: a view from the width. In: *Advances in neural information processing systems*, pp. 6231–6239.**[AQ: 2]**
[16] Hanin, B. Universal function approximation by deep neural nets with bounded width and ReLU activations. *Mathematics* 2019; 7(10): 992.
[17] Mahendran, A, and Vedaldi, A. Understanding deep image representations by inverting them. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 5188–5196.**[AQ: 3]**
[18] Shwartz-Ziv, R, and Tishby, N. Opening the black box of deep neural networks via information. *arXiv* [preprint]. arXiv:170300810, 2017.
[19] Adadi, A, and Berrada, M. Peeking inside the black-box: a survey on explainable artificial intelligence (XAI). *IEEE Access* 2018; 6: 52138–52160.
[20] Xu, F, Uszkoreit, H, Du, Y, et al. Explainable AI: a brief survey on history, research areas, approaches and challenges. In: *Natural language processing and Chinese computing: 8th CCF international conference, NLPCC 2019*, Dunhuang, China, 9–14 October 2019, pp. 563–574. Berlin: Springer.
[21] Ibañez, R, Borzacchiello, D, Aguado, JV, et al. Data-driven nonlinear elasticity: constitutive manifold construction and problem discretization. *Comput Mech* 2017; 60: 813–826.
[22] Ayensa-Jimenez, J, Doweidar, MH, Sanz-Herrera, JA et al. Prediction and identification of physical systems by means of physically-guided neural networks with meaningful internal layers. *Comput Methods Appl Mech Eng* 2021; 381: 113816.
[23] Masi, F, Stefanou, I, Vannucci, P, et al. Thermodynamics-based artificial neural networks for constitutive modeling. *J Mech Phys Solids* 2021; 147: 104277.
[24] Karpatne, A, Atluri, G, Faghmous, JH, et al. Theory-guided data science: a new paradigm for scientific discovery from data. *IEEE Trans Knowl Data Eng* 2017; 29(10): 2318–2331.
[25] Muralidhar, N, Islam, MR, Marwah, M, et al. Incorporating prior domain knowledge into deep neural networks. In: *2018 IEEE international conference on big data (big data)*, Seattle, WA, 10–13 December 2018, pp. 36–45. New York: IEEE.
[26] Raissi, M, Perdikaris, P, and Karniadakis, GE. Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J Comput Phys* 2019; 378: 686–707.
[27] Lucia, DJ, Beran, PS, and Silva, WA. Reduced-order modeling: new approaches for computational physics. *Prog Aerosp Sci* 2004; 40(1–2): 51–117.
[28] Forrester, AI, and Keane, AJ. Recent advances in surrogate-based optimization. *Prog Aerosp Sci* 2009; 45(1–3): 50–79.
[29] Cozad, A, Sahinidis, NV, and Miller, DC. Learning surrogate models for simulation-based optimization. *AIChE J* 2014; 60(6): 2211–2227.
[30] Tripathy, RK, and Bilionis, I. Deep UQ: learning deep neural network surrogate models for high dimensional uncertainty quantification. *J Comput Phys* 2018; 375: 565–588.
[31] Frangos, M, Marzouk, Y, Willcox, K, et al. Surrogate and reduced-order modeling: a comparison of approaches for large-scale statistical inverse problems. In: *Large-scale inverse problems and quantification of uncertainty*. 2010, pp. 123–149.**[AQ: 4]**
[32] Yagawa, G, and Okuda, H. Neural networks in computational mechanics. *Arch Comput Methods Eng* 1996; 3(4): 435–512.
[33] Ghaboussi, J. Advances in neural networks in computational mechanics and engineering. In: Waszczyszyn, Z (ed.) *Advances of soft computing in engineering*. Berlin: Springer, 2010, pp. 191–236.

[34] Kollmannsberger, S, D'Angella, D, Jokeit, M, et al. *Deep learning in computational mechanics*. Berlin: Springer, 2021.

[35] Ayensa-Jiménez, J, Doweidar, MH, Sanz-Herrera, JA, et al. Understanding glioblastoma invasion using physically-guided neural networks with internal variables. *PLoS Comput Biol* 2022; 18(4): e1010019.

[36] Lagaris, IE, Likas, AC, and Papageorgiou, DG. Neural-network methods for boundary value problems with irregular boundaries. *IEEE Trans Neural Netw* 2000; 11(5): 1041–1049.

[37] Han, J, Jentzen, A, and E, W. Solving high-dimensional partial differential equations using deep learning. *Proc Natl Acad Sci USA* 2018; 115(34): 8505–8510.

[38] Flaschel, M, Kumar, S, and De Lorenzis, L. Discovering plasticity models without stress data. *npj Comput Mater* 2022; 8(1): 91.

[39] Tartakovsky, AM, Marrero, CO, Perdikaris, P, et al. Learning parameters and constitutive relationships with physics informed deep neural networks. *arXiv* [preprint]. arXiv:180803398, 2018.

[40] Thakolkaran, P, Joshi, A, Zheng, Y, et al. NN-EUCLID: deep-learning hyperelasticity without stress data. *J Mech Phys Solids* 2022; 169: 105076.

[41] Ayensa-Jiménez, J, Doweidar, MH, Sanz-Herrera, JA, et al. On the application of physically-guided neural networks with internal variables to continuum problems. *arXiv* [preprint]. arXiv:201111376, 2020.

[42] van Milligen, BP, Tribaldos, V, and Jiménez, J. Neural network differential equation and plasma equilibrium solver. *Phys Rev Lett* 1995; 75(20): 3594.

[43] Lagaris, IE, Likas, A, and Fotiadis, DI. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Trans Neural Netw* 1998; 9(5): 987–1000.

[44] Rudd, K, and Ferrari, S. A constrained integration (CINT) approach to solving partial differential equations using artificial neural networks. *Neurocomputing* 2015; 155: 277–285.

[45] Sirignano, J, and Spiliopoulos, K. DGM: a deep learning algorithm for solving partial differential equations. *J Comput Phys* 2018; 375: 1339–1364.

[46] Abueidda, DW, Lu, Q, and Koric, S. Deep learning collocation method for solid mechanics: linear elasticity, hyperelasticity, and plasticity as examples. *arXiv* [e-prints]. arXiv-2012, 2020.

[47] E, W, and Yu, B. The Deep Ritz method: a deep learning-based numerical algorithm for solving variational problems. *Commun Math Stat* 2018; 6(1): 1–12.

[48] Samaniego, E, Anitescu, C, Goswami, S, et al. An energy approach to the solution of partial differential equations in computational mechanics via machine learning: concepts, implementation and applications. *Comput Methods Appl Mech Eng* 2020; 362: 112790.

[49] Nguyen-Thanh, VM, Zhuang, X, and Rabczuk, T. A deep energy method for finite deformation hyperelasticity. *Eur J Mech A/Solids* 2020; 80: 103874.

[50] Karpatne, A, Watkins, W, Read, J, et al. Physics-guided neural networks (PGNN): an application in lake temperature modeling. *arXiv* [preprint]. arXiv:171011431, 2017.

[51] Stewart, R, and Ermon, S. Label-free supervision of neural networks with physics and domain knowledge. In: *Thirty-first AAAI conference on artificial intelligence*, 2016, pp. 2576–2582.**[AQ: 5]**

[52] Magiera, J, Ray, D, Hesthaven, JS, et al. Constraint-aware neural networks for riemann problems. *J Comput Phys* 2020; 409: 109345.

[53] E, W, Han, J, and Jentzen, A. Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations. *Commun Math Stat* 2017; 5(4): 349–380.

[54] Huré, C, Pham, H, and Warin, X. Deep backward schemes for high-dimensional nonlinear PDEs. *Math Comput* 2020; 89(324): 1547–1579.

[55] Tamaddon-Jahromi, HR, Chakshu, NK, Sazonov, I, et al. Data-driven inverse modelling through neural network (deep learning) and computational heat transfer. *Comput Methods Appl Mech Eng* 2020; 369: 113217.

[56] Hamel, CM, Long, KN, and Kramer, SL. Calibrating constitutive models with full-field data via physics informed neural networks. *Strain* 2022; 59: e12431.

[57] Herrmann, L, and Kollmannsberger, S. Deep learning in computational mechanics: a review. *Comput Mech*. Epub ahead of print January 2024. DOI: 10.1007/s00466-023-02434-4.

[58] Karapiperis, K, Stainier, L, Ortiz, M, et al. Data-driven multiscale modeling in mechanics. *J Mech Phys Solids* 2021; 147: 104239.

[59] Rao, C, Sun, H, and Liu, Y. Physics-informed deep learning for computational elastodynamics without labeled data. *J Eng Mech* 2021; 147(8): 04021043.

[60] Bharadwaja, B, Nabian, MA, Sharma, B, et al. Physics-informed machine learning and uncertainty quantification for mechanics of heterogeneous materials. *Integr Mater Manuf Innov* 2022; 11: 607–627.

[61] Haghighat, E, Abouali, S, and Vaziri, R. Constitutive model characterization and discovery using physics-informed deep learning. *Eng Appl Artif Intell* 2023; 120: 105828.

[62] Walczak, S, and Cerpa, N. Heuristic principles for the design of artificial neural networks. *Inf Softw Tech* 1999; 41(2): 107–117.

[63] Chu, T, Ranson, W, and Sutton, MA. Applications of digital-image-correlation techniques to experimental mechanics. *Exp Mech* 1985; 25(3): 232–244.

[64] Rohrhofer, FM, Posch, S, and Geiger, BC. On the Pareto front of physics-informed neural networks. *arXiv* [preprint]. arXiv:210500862, 2021.

[65] Kaltenbach, S, and Koutsourelakis, PS. Incorporating physical constraints in a deep probabilistic machine learning framework for coarse-graining dynamical systems. *J Comput Phys* 2020; 419: 109673.

[66] Kaltenbach, S, and Koutsourelakis, PS. Physics-aware, deep probabilistic modeling of multiscale dynamics in the small data regime. *arXiv* [preprint]. arXiv:210204269, 2021.

[67] Ogden, RW. Large deformation isotropic elasticity–on the correlation of theory and experiment for incompressible rubberlike solids. *Proc R Soc Lond A Math Phys Sci* 1972; 326(1567): 565–584.

[68] Holzapfel, GA. *Nonlinear solid mechanics: a continuum approach for engineering science*, 2002.**[AQ: 6]**

[69] Herm, LV, Heinrich, K, Wanner, J, et al. Stop ordering machine learning algorithms by their explainability! a user-centered investigation of performance and explainability. *Int J Inf Manag* 2023; 69: 102538.

[70] Dennis Jr, JE, and Schnabel, RB. *Numerical methods for unconstrained optimization and nonlinear equations*. Philadelphia, PA: SIAM, 1996.

[71] Linka, K, and Kuhl, E. A new family of constitutive artificial neural networks towards automated model discovery. *Comput Methods Appl Mech Eng* 2023; 403: 115731.

[72] Flaschel, M, Kumar, S, and De Lorenzis, L. Unsupervised discovery of interpretable hyperelastic constitutive laws. *Comput Methods Appl Mech Eng* 2021; 381: 113852.

[73] Flaschel, M, Kumar, S, and De Lorenzis, L. Automated discovery of generalized standard material models with EUCLID. *Comput Methods Appl Mech Eng* 2023; 405: 115867.

[74] Marino, E, Flaschel, M, Kumar, S, et al. Automated identification of linear viscoelastic constitutive laws with EUCLID. *Mech Mater* 2023; 181: 104643.

[75] Joshi, A, Thakolkaran, P, Zheng, Y, et al. Bayesian-EUCLID: discovering hyperelastic material laws with uncertainties. *Comput Methods Appl Mech Eng* 2022; 398: 115225.

[76] Hernández, Q, Badías, A, González, D, et al. Structure-preserving neural networks. *J Comput Phys* 2021; 426: 109950.

[77] Hernandez, Q, Badias, A, Gonzalez, D, et al. Deep learning of thermodynamics-aware reduced-order models from data. *Comput Methods Appl Mech Eng* 2021; 379: 113763.

[78] Linka, K, Pierre, SRS, and Kuhl, E. Automated model discovery for human brain using constitutive artificial neural networks. *Acta Biomater* 2023; 160: 134–151.

[79] Linden, L, Klein, DK, Kalina, KA, et al. Neural networks meet hyperelasticity: a guide to enforcing physics. *J Mech Phys Solids* 2023; 179: 105363.

[80] Mottershead, JE, Link, M, and Friswell, MI. The sensitivity method in finite element model updating: a tutorial. *Mech Syst Signal Process* 2011; 25(7): 2275–2296.

[81] Grediac, M, Pierron, F, Avril, S, et al. The virtual fields method for extracting constitutive parameters from full-field measurements: a review. *Strain* 2006; 42(4): 233–253.

[82] Ma, Y, and Fu, Y. *Manifold learning theory and applications* (Vol. 434). Boca Raton, FL: CRC Press, 2012.