

RESEARCH ARTICLE 

# Parametrized polyconvex hyperelasticity with physics-augmented neural networks

Dominik K. Klein , Fabian J. Roth , Iman Valizadeh  and Oliver Weeger 

Cyber-Physical Simulation, Technical University of Darmstadt, Darmstadt, Germany

**Corresponding author:** Dominik K. Klein; Email: [klein@cps.tu-darmstadt.de](mailto:klein@cps.tu-darmstadt.de)

**Received:** 15 June 2023; **Revised:** 19 September 2023; **Accepted:** 02 October 2023

**Keywords:** constitutive modeling; hyperelasticity; parametrized material; partially input convex neural networks; physics-augmented neural networks

## Abstract


In the present work, neural networks are applied to formulate parametrized hyperelastic constitutive models. The models fulfill all common mechanical conditions of hyperelasticity by construction. In particular, partially input convex neural network (pICNN) architectures are applied based on feed-forward neural networks. Receiving two different sets of input arguments, pICNNs are convex in one of them, while for the other, they represent arbitrary relationships which are not necessarily convex. In this way, the model can fulfill convexity conditions stemming from mechanical considerations without being too restrictive on the functional relationship in additional parameters, which may not necessarily be convex. Two different models are introduced, where one can represent arbitrary functional relationships in the additional parameters, while the other is monotonic in the additional parameters. As a first proof of concept, the model is calibrated to data generated with two differently parametrized analytical potentials, whereby three different pICNN architectures are investigated. In all cases, the proposed model shows excellent performance.

## Impact Statement

Constitutive models relate the strain inside a material body to the stress it evokes. In this work, the excellent flexibility that neural networks offer is exploited to formulate hyperelastic constitutive models, which describe large, reversible deformations. The models are physics-augmented, that is, they fulfill all common mechanical conditions of hyperelasticity by construction. This results in highly flexible yet physically sensible neural network models. These models will be applicable to a wide range of materials, particularly to the representation of microstructured materials, for example, fiber-reinforced composites, metamaterials, textiles, or tissues. By that, computationally more expensive methods can be avoided, thus accelerating the simulation and optimization of engineering components made of microstructured materials.

## 1. Introduction

Convexity is a convenient property of mathematical functions in many applications. However, it also constrains the function space a model can represent. While for some applications, this constraint is well motivated, it is too restrictive for other use cases. Moreover, there are applications where a function can be

 This research article was awarded an Open Data badge for transparent practices. See the Data Availability Statement for details.

© The Author(s), 2023. Published by Cambridge University Press. This is an Open Access article, distributed under the terms of the Creative Commons Attribution licence (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted re-use, distribution and reproduction, provided the original article is properly cited.



motivated to be convex in some of its arguments, while it should not necessarily be convex in the other arguments. The latter is usually the case for hyperelastic material models with parametric dependencies, such as process parameters in 3D printing which influence material properties (Valizadeh et al., 2021), or microstructured materials with a parametrized geometry (Fernández et al., 2022). In the framework of hyperelasticity, the polyconvexity condition introduced by Ball (Ball, 1976, 1977) requires the associated energy potentials to be convex functions in several strain measures. However, there is generally no mechanical motivation for a hyperelastic potential to be convex in additional parameters on which it might depend. To reflect this, a modeling framework for parametrized polyconvex hyperelasticity should provide potentials which are convex in the arguments of the polyconvexity condition and can represent more general functional relationships in the additional parameters. Finally, in some cases, further conditions such as monotonicity of the hyperelastic potential in some parameters can be motivated by physical considerations (Valizadeh et al., 2021).

In finite elasticity theory, convexity of the energy potential in the primary deformation measure alone (the deformation gradient  $F$ ) would be too restrictive. In particular, this would be incompatible with growth and objectivity conditions, and it would not allow to represent certain phenomena such as buckling (Ebbing, 2010; Section 5.2). Polyconvexity circumvents these problems by formulating energy potentials which are convex in an extended set of arguments, namely the deformation gradient, its cofactor, and its determinant, making this convexity condition compatible with aforementioned physical considerations. When the energy potential is convex in these strain measures (and thus polyconvex) and an additional coercivity condition is fulfilled,<sup>1</sup> the existence of minimizers of the underlying variational functionals of finite elasticity theory is guaranteed (Kružík and Roubíček, 2019). Indeed, this coercivity condition makes assumptions on the hyperelastic potential which lie far outside a practically relevant deformation range, making the practical relevance of this existence theorem questionable (Klein et al., 2022a).

Apart from that, from an engineering perspective, polyconvexity is desirable since it implies ellipticity (or rank-one convexity) of hyperelastic potentials (Zee and Sternberg, 1983; Neff et al., 2015). Ellipticity, in turn, is important for a stable behavior of numerical applications such as the finite element method. Without polyconvexity, the ellipticity of a hyperelastic potential is cumbersome to check, and practically impossible to fulfill by construction. Overall, from an engineering perspective, polyconvexity is desirable since it implies ellipticity, rather than for its significance in existence theorems.

In constitutive modeling, neural networks (NNs) can be applied to represent hyperelastic potentials. These highly flexible models are usually formulated to fulfill mechanical conditions relevant to hyperelasticity, thus combining the extraordinary flexibility that NNs offer with a sound mechanical basis. Such models are precious in fields where highly flexible yet physically sensible models are required, such as the simulation of microstructured materials (Gärtner et al., 2021; Kumar and Kochmann, 2022; Kalina et al., 2023). Furthermore, including mechanical conditions improves the model generalization (Klein et al., 2022b), allowing for model calibrations with sparse data usually available from real-world experiments (Linka et al., 2023). For the construction of polyconvex potentials, several approaches exist (Chen and Guilleminot, 2022; Klein et al., 2022a; Tac et al., 2022), where the most noteworthy approaches are based on input-convex neural networks (ICNNs). Proposed by Amos et al. (2017), this special network architecture has not only been successfully applied in the framework of polyconvexity, but is also very attractive in, for example, other physical applications which require convexity (Huang et al., 2021; As'ad and Farhat, 2023; Rosenkranz et al., 2023) and convex optimization (Calafiore et al., 2020). Besides this particular choice of network architecture, using invariants as strain measures ensures the fulfillment of several mechanical conditions at once, for example, objectivity and material symmetry. This is well-known from analytical constitutive modeling (Schröder and Neff, 2003; Ebbing, 2010) and also commonly applied in NN-based models (Klein et al., 2022b; Kalina et al., 2023; Linka and Kuhl, 2023; Tac et al., 2023). Finally, by embedding the NN-potential into a larger modeling framework, that is,

<sup>1</sup> For a coercive function,  $f(x) \rightarrow \infty$  as  $\|x\| \rightarrow \infty$  holds.

adding additional analytical terms, all common constitutive conditions of hyperelasticity can be fulfilled by construction, which was at first introduced for compressible material behavior by Linden et al. (2023). Therein, models that fulfill all mechanical conditions by construction are denoted as *physics-augmented neural networks* (PANNs).

In the literature, also parametrized models were proposed for different applications, both in the analytical (Wu et al., 2018; Valizadeh and Weeger, 2022) and in the NN context (Baldi et al., 2016; Shojaee et al., 2023). In particular, also parametrized hyperelastic constitutive models were proposed. In Valizadeh et al. (2021), an analytical model is proposed which maps process parameters of a 3D printing process to material properties, by formulating parametrized hyperelastic potentials. In Linka et al. (2021) and Fernández et al. (2022), parametrized hyperelastic potentials based on NNs are proposed and applied to different homogenized microstructures. However, to the best of the authors' knowledge, none of the existing parametrized hyperelastic models based on NNs fulfills all constitutive conditions at the same time. In particular, no model fulfills the polyconvexity condition.

To conclude, while parametrized and polyconvex models are well-established in the framework of NN-based constitutive modeling, the link between both still needs to be made. In the present work, this is done by applying partially input convex neural networks (pICNNs) as proposed by Amos et al. (2017). Receiving two sets of input arguments, pICNNs are convex in one while representing arbitrary relationships for the other. With the model proposed in this work being an extension of Linden et al. (2023), all common constitutive conditions of hyperelasticity are fulfilled by construction. In particular, the model fulfills several mechanical conditions by using polyconvex strain invariants as inputs, while the pICNN preserves the polyconvexity of the invariants. Furthermore, growth and normalization terms ensure a physically sensible stress behavior of the model. Two cases are considered, one with an arbitrary functional relationship in the additional parameters and the other being monotonic in the additional parameters. To formulate the functional relationships, three different pICNN architectures with different complexities are applied. The proposed model will be valuable for the representation of microstructured materials. In particular, the model can represent materials with a parametrized microstructure, for example, lattice-metamaterials with varying radii (Fernández et al., 2022), fiber-reinforced elastomers where the volume fraction of the fibers might vary (Kalina et al., 2023), microstructures with spherical inclusions, where the stiffness of the inclusions might vary (Klein et al., 2022b), or knitted textiles with graded stitch types and knitting parameters (Do et al., 2020). The parametrization allows for both simulation and optimization of such materials, while the polyconvexity of the model ensures a stable behavior of the numerical simulations required for this.

The outline of the manuscript is as follows. In Section 2, the convexity of function compositions is discussed. In Section 3, the fundamentals of parametrized hyperelasticity are briefly introduced, which are then applied to the proposed PANN model in Section 4. The applicability of the parametric architectures is demonstrated by calibrating it to data generated with two differently parametrized analytical potentials in Section 5, followed by the conclusion in Section 6.

### 1.1. Notation

Throughout this work, scalars, vectors, and second-order tensors are indicated by  $a$ ,  $\mathbf{a}$ , and  $\mathbf{A}$ , respectively. The second-order identity tensor is denoted as  $\mathbf{I}$ . Transpose and inverse are denoted as  $\mathbf{A}^T$  and  $\mathbf{A}^{-1}$ , respectively. Furthermore, trace, determinant, and cofactor are denoted by  $\text{tr } \mathbf{A}$ ,  $\det \mathbf{A}$ , and  $\text{cof } \mathbf{A} := \det(\mathbf{A})\mathbf{A}^{-T}$ . The set of invertible second-order tensors with positive determinant is denoted by  $\text{GL}^+(3) := \{\mathbf{X} \in \mathbb{R}^{3 \times 3} \mid \det \mathbf{X} > 0\}$  and the special orthogonal group in  $\mathbb{R}^3$  by  $\text{SO}(3) := \{\mathbf{X} \in \mathbb{R}^{3 \times 3} \mid \mathbf{X}^T \mathbf{X} = \mathbf{I}, \det \mathbf{X} = 1\}$ . For the function composition  $f(g(x))$  the compact notation  $(f \circ g)(x)$  is applied. The Softplus, Sigmoid, and ReLU functions are denoted by  $s(x) = \ln(1 + e^x)$ ,  $\text{sm}(x) = \frac{1}{1 + e^{-x}}$ , and  $[x]_+ = \max(x, 0)$ , respectively. The element-wise product between vectors is denoted as  $*$ .

### 2. Convexity of function compositions

To lay the foundational intuition for constructing convex neural networks, we first consider the univariate function

$$f : \mathbb{R} \rightarrow \mathbb{R}, \quad x \mapsto f(x) := (g \circ h)(x), \tag{1}$$

where  $f$  is composed of two functions  $g, h : \mathbb{R} \rightarrow \mathbb{R}$ . Given that all of these functions are twice continuously differentiable, convexity of  $f$  in  $x$  is equivalent to the nonnegativity of the second derivative

$$f''(x) = (g'' \circ h)(x) h'(x)^2 + (g' \circ h)(x) h''(x) \geq 0. \tag{2}$$

A sufficient, albeit not necessary condition for this is that the function  $h$  is convex ( $h'' \geq 0$ ), while the function  $g$  is convex and nondecreasing ( $g' \geq 0, g'' \geq 0$ ). Conversely, if a function acting on a convex function does not fulfill these conditions, the resulting function is not necessarily convex, see Figure 1 for an example. The recursive application of equation (2) yields conditions for arbitrary many function compositions. The innermost function, here  $h$ , must only be convex, while every following function must be convex and nondecreasing to preserve convexity.

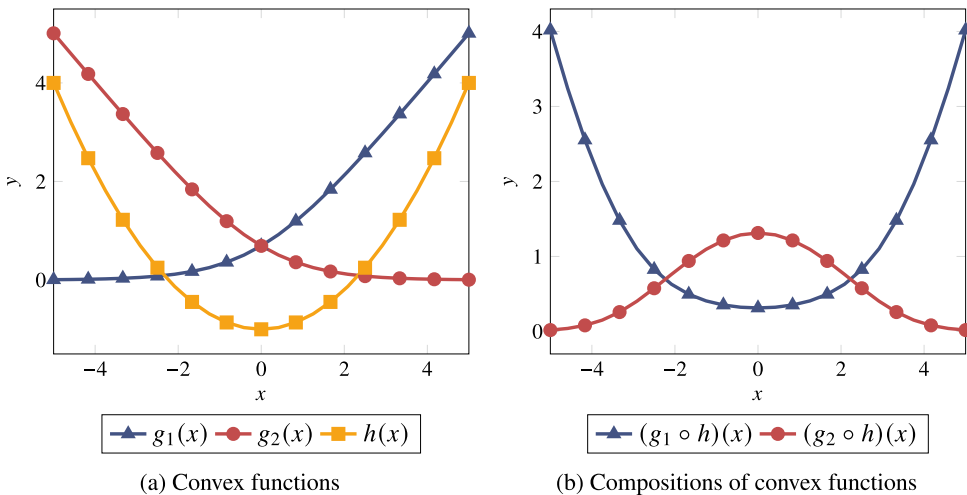
The generalization to compositions of multivariate functions is also straightforward. For this, we consider the function

$$f : \mathbb{R}^m \rightarrow \mathbb{R}, \quad \mathbf{x} \mapsto f(\mathbf{x}) := (g \circ \mathbf{h})(\mathbf{x}), \tag{3}$$

with  $\mathbf{h} : \mathbb{R}^m \rightarrow \mathbb{R}^n$  and  $g : \mathbb{R}^n \rightarrow \mathbb{R}$ . Given that all of these functions are twice continuously differentiable, convexity of  $f$  in  $\mathbf{x}$  is equivalent to the positive semi-definiteness of its Hessian. Similar reasoning as above leads to the sufficient condition that  $\mathbf{h}$  must be component-wise convex, while  $g$  must be convex and nondecreasing, see Klein et al. (2022a) for an explicit proof. Again, the recursive application of this yields conditions for arbitrary many function compositions. Here, the innermost function must be component-wise convex, while every following function must be component-wise convex and nondecreasing to preserve convexity.

In the same manner, the composite function  $f$ , compare equation (1), is monotonically increasing (or nondecreasing) when its first derivative

$$f'(x) = (g' \circ h)(x) h'(x) \geq 0 \tag{4}$$



**Figure 1.** Compositions of univariate convex functions.  $h(x) = 0.2x^2 - 1$ ,  $g_1(x) = s(x)$ ,  $g_2(x) = s(-x)$ . Note that  $g_1(x)$  is convex and nondecreasing, thus the composite function  $(g_1 \circ h)(x)$  is convex.  $g_2(x)$  is convex but decreasing, and the composite function  $(g_2 \circ h)(x)$  is not convex.

is nonnegative, which is fulfilled when both  $g$  and  $h$  are nondecreasing functions ( $g' \geq 0, h' \geq 0$ ). The recursive application of this yields again conditions for arbitrary many function compositions. When all functions within a composite function are nondecreasing, the overall function is nondecreasing, see  $(g_1 \circ h)(x)$  for  $x \geq 0$  in Figure 1 for an example. In this case, the generalization to compositions of vector-valued functions leads to the condition that all functions must be component-wise nondecreasing.

These basic ideas will be applied in both the mechanical requirements of the proposed model, compare Section 3.2, and in the construction of suitable network architectures, compare Section 4.2.

### 3. Parametrized hyperelastic constitutive modeling

Hyperelastic constitutive models describe the behavior of materials such as rubber for large, reversible deformations. For this, a hyperelastic potential is formulated which corresponds to the strain energy density stored in the body due to deformation. In this work, the hyperelastic potential depends both on the strain and additional parameters characterizing the material. In Section 3.1, the mechanical conditions that the constitutive model should fulfill are introduced. The general framework for a model formulated in strain invariants which fulfills these conditions is introduced in Section 3.2.

#### 3.1. Constitutive requirements for parametrized hyperelasticity

The mechanical conditions of hyperelasticity are now briefly discussed. For a detailed introduction, the reader is referred to Holzapfel (2000) and Ebbing (2010). The parametrized hyperelastic potential

$$\psi : \text{GL}^+(3) \times \mathbb{R}^n \rightarrow \mathbb{R}, \quad (\mathbf{F}; \mathbf{t}) \mapsto \psi(\mathbf{F}; \mathbf{t}) \tag{5}$$

corresponds to the strain energy density stored in the body  $\mathcal{B} \subset \mathbb{R}^3$  due to the deformation  $\varphi : \mathcal{B} \rightarrow \mathbb{R}^3$ . It depends on the deformation gradient  $\mathbf{F} = D\varphi$  and the parameter vector  $\mathbf{t} \in \mathbb{R}^n$ . With the stress being defined as the gradient field

$$\mathbf{P} = \frac{\partial \psi(\mathbf{F}; \mathbf{t})}{\partial \mathbf{F}}, \tag{6}$$

the (i) *second law of thermodynamics* is fulfilled by construction. The principle of (ii) *objectivity* states that a model should be independent of the choice of observer, which is formalized as

$$\psi(\mathbf{Q}\mathbf{F}; \mathbf{t}) = \psi(\mathbf{F}; \mathbf{t}) \quad \forall \mathbf{F} \in \text{GL}^+(3), \mathbf{Q} \in \text{SO}(3), \mathbf{t} \in \mathbb{R}^n. \tag{7}$$

Also, the model should reflect the materials underlying (an-)isotropy, which corresponds to the (iii) *material symmetry condition*

$$\psi(\mathbf{F}\mathbf{Q}^T; \mathbf{t}) = \psi(\mathbf{F}; \mathbf{t}) \quad \forall \mathbf{F} \in \text{GL}^+(3), \mathbf{Q} \in \mathcal{G} \subseteq \text{SO}(3), \mathbf{t} \in \mathbb{R}^n, \tag{8}$$

where  $\mathcal{G}$  denotes the symmetry group under consideration. The (iv) *balance of angular momentum* implies that

$$\frac{\partial \psi(\mathbf{F}; \mathbf{t})}{\partial \mathbf{F}} \mathbf{F}^T = \mathbf{F} \frac{\partial \psi(\mathbf{F}; \mathbf{t})}{\partial \mathbf{F}^T} \quad \forall \mathbf{F} \in \text{GL}^+(3), \mathbf{t} \in \mathbb{R}^n. \tag{9}$$

Furthermore, we consider (v) *polyconvex* potentials which allow for a representation

$$\psi(\mathbf{F}; \mathbf{t}) = \mathcal{P}(\boldsymbol{\xi}; \mathbf{t}) \text{ with } \boldsymbol{\xi} := (\mathbf{F}, \text{cof} \mathbf{F}, \det \mathbf{F}), \tag{10}$$

where  $\mathcal{P}$  is a convex function in  $\boldsymbol{\xi}$ . Note that polyconvexity does not restrict the potential's functional dependency on  $\mathbf{t}$ . While the notion of polyconvexity stems from a rather theoretical context, it is also of practical relevance as it is the most straightforward way of fulfilling the ellipticity condition

$$(\mathbf{a} \otimes \mathbf{b}) : \frac{\partial^2 \psi(\mathbf{F}; \mathbf{t})}{\partial \mathbf{F} \partial \mathbf{F}} : (\mathbf{a} \otimes \mathbf{b}) \geq 0 \quad \forall \mathbf{a}, \mathbf{b} \in \mathbb{R}^3. \tag{11}$$

Also known as material stability, this condition leads to a favorable behavior in numerical applications. Finally, a physically sensible stress behavior requires fulfillment of the (vi) *growth condition*

$$\psi \rightarrow \infty \text{ as } (\det \mathbf{F} \rightarrow 0^+ \vee \det \mathbf{F} \rightarrow \infty), \tag{12}$$

as well as a stress-free reference configuration  $\mathbf{F} = \mathbf{I}$ , also referred to as (vii) *normalization*

$$\mathbf{P}(\mathbf{I}; \mathbf{t}) = \mathbf{0} \quad \forall \mathbf{t} \in \mathbb{R}^n. \tag{13}$$

In the most general case, no mechanical condition restricts the functional dependency of the potential  $\psi(\mathbf{F}; \mathbf{t})$  in the parameters  $\mathbf{t}$ . However, for *some* applications, it may be well motivated to assume that the potential is a monotonically increasing function in the parameters. This (viii) *monotonicity condition* is formalized as

$$\frac{\partial \psi(\mathbf{F}; \mathbf{t})}{\partial t_i} \geq 0 \quad \forall i \in \mathbb{N}_{\leq n}, \mathbf{F} \in \text{GL}^+(3), \mathbf{t} \in \mathbb{R}^n. \tag{14}$$

Note that this does not imply monotonicity of the components of  $\mathbf{P}(\mathbf{F}; \mathbf{t})$  in  $\mathbf{t}$ , which would mean that every component of the mixed second derivative

$$\frac{\partial \mathbf{P}(\mathbf{F}; \mathbf{t})}{\partial \mathbf{t}} = \frac{\partial^2 \psi(\mathbf{F}; \mathbf{t})}{\partial \mathbf{F} \partial \mathbf{t}} \tag{15}$$

would have to be nonnegative. However, formulations which fulfill [equation \(15\)](#) could easily become too restrictive. For example, they might lead to potentials which are convex in  $\mathbf{F}$  alone instead of the extended set of arguments of the polyconvexity condition, compare [equation \(10\)](#). However, convexity of the potential in  $\mathbf{F}$  is not compatible with a physically sensible material behavior (Yang Gao et al., 2017). Thus, the monotonicity condition [equation \(14\)](#) is applied throughout this work.

Note that additional conditions on a physically sensible behavior of the hyperelastic potential can be formulated, for example, the energy normalization  $\psi(\mathbf{I}; \mathbf{t}) = 0 \quad \forall \mathbf{t} \in \mathbb{R}^n$  (Linden et al., 2023). However, throughout this work, we focus on the representation of the stress, meaning the gradient of the potential. Still, most conditions presented in this section are formulated in the hyperelastic potential, mainly for a convenient, brief notation.

### 3.2. Invariant-based modeling

By formulating the potential  $\psi$  in terms of invariants of the right Cauchy–Green tensor  $\mathbf{C} = \mathbf{F}^T \mathbf{F}$ , conditions (ii–iv) can be fulfilled. Throughout this work, isotropic material behavior is assumed, that is,  $\mathcal{G} = \text{SO}(3)$  in [equation \(8\)](#). In this case, three polyconvex invariants

$$I_1 = \text{tr } \mathbf{C}, \quad I_2 = \text{tr}(\text{cof } \mathbf{C}), \quad I_3 = \det \mathbf{C}, \tag{16}$$

are considered. Then, the potential can be reformulated as<sup>2</sup>

$$\psi : \mathbb{R}^m \times \mathbb{R}^n \rightarrow \mathbb{R}, \quad (\mathcal{J}; \mathbf{t}) \mapsto \psi(\mathcal{J}; \mathbf{t}), \tag{17}$$

with

$$\mathcal{J} = (I_1, I_2, I_3, I_3^*) \in \mathbb{R}^4, \quad I_3^* = -\sqrt{I_3}, \tag{18}$$

where the additional polyconvex invariant  $I_3^*$  is important for the model to represent negative stress values, compare Klein et al. (2022a). The invariants are nonlinear functions in the arguments of the polyconvexity condition, compare [equation \(10\)](#). Thus, following [Section 2](#), the potential  $\psi$  must be *convex* and component-wise *nondecreasing* in  $\mathcal{J}$  to preserve the polyconvexity of the invariants. By this, the overall potential fulfills the (v) *polyconvexity condition*. Note that this general form of the potential does not yet fulfill conditions (vi–vii), which ensure a physically sensible stress behavior of the model.

<sup>2</sup> Note that  $\psi(\mathbf{F}; \mathbf{t})$  and  $\psi(\mathcal{J}; \mathbf{t})$  are different functions, but in the interest of readability, the same symbols are used.

In the analytical case, an explicit choice of functional relationship for the hyperelastic potential has to be made, which fulfills all above-introduced conditions. One such choice is the Neo–Hookean model

$$\psi^{\text{nh}}(I_1, I_3; t) = \frac{\mu(t)}{2}(I_1 - 3 - 2 \ln \sqrt{I_3}) + \frac{\lambda(t)}{2}(\sqrt{I_3} - 1)^2. \tag{19}$$

Here, the Lamé parameters  $\lambda(t), \mu(t)$  are parametrized in terms of  $t \in \mathbb{R}$ . We remark that there exist different representations of material parameters, for example, the Lamé parameters can be calculated by the Young’s modulus  $E$  and the Poisson’s ratio  $\nu$  by

$$\mu = \frac{E}{2(1 + \nu)}, \quad \lambda = \frac{E\nu}{(1 + \nu)(1 - 2\nu)}. \tag{20}$$

While some analytical models base their functional relationship on physical reasoning, such as the Hencky model (Hencky, 1928; Neff et al., 2016), most constitutive models are of heuristic nature. Simply put, the fulfillment of the objectivity condition by the Neo–Hookean model has a solid mechanical motivation, while its linear dependency on  $I_1$  has not and is simply a man-made choice. The following section discusses how such limitations can be circumvented by applying NNs as highly flexible functions.

#### 4. Parameterized, physics-augmented neural network model

As discussed in the previous section, the formulation of parametrized polyconvex potentials requires functions that are convex and nondecreasing in several strain invariants. At the same time, the functional relationship in the additional parameters should be either a general one or monotonically increasing, respectively, compare equation (14). Instead of making an explicit choice for such a formulation, we represent it by a neural network (NN), which can generally represent arbitrary functions (Hornik, 1991).

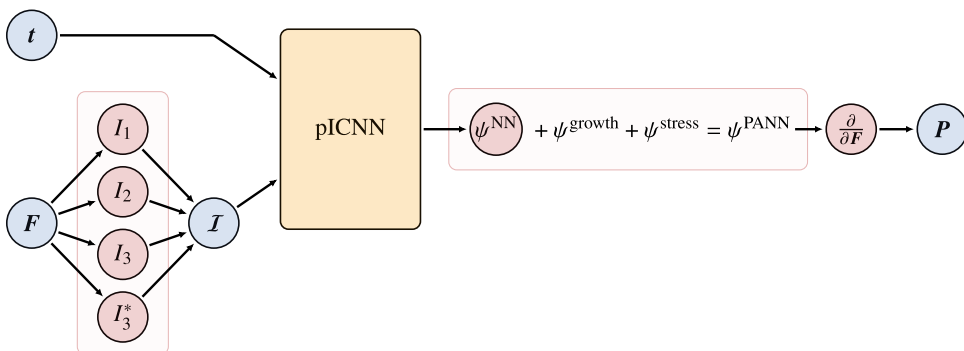
##### 4.1. Physics-augmented model formulation

To incorporate the constitutive requirements introduced above in Section 3, the NN is only a part of the overall PANN material model given by

$$\psi^{\text{PANN}}(\mathcal{J}; t) = \psi^{\text{NN}}(\mathcal{J}; t) + \psi^{\text{growth}}(J) + \psi^{\text{stress}}(J; t), \tag{21}$$

which is an extension of the model proposed by Linden et al. (2023) with parametric dependencies. The overall flow and structure of the model are visualized in Figure 2.

In equation (21),  $\psi^{\text{NN}}(\mathcal{J}; t)$  denotes the partially input-convex neural network (pICNN), which is convex and nondecreasing in  $\mathcal{J}$  and arbitrary (or monotonically increasing) in  $t$ . In Section 4.2,



**Figure 2.** Illustration of the PANN-based constitutive model. The pICNN is convex and nondecreasing in the invariants  $\mathcal{J}$  while representing arbitrary (or monotonically increasing) functional relationships in the additional parameters  $t$ .



different pICNN architectures are discussed. To this point,  $\psi^{\text{NN}}$  is treated as a general, sufficiently smooth function. The remaining terms in equation (21) ensure a physically sensible stress behavior of the model. In particular, they ensure the growth and normalization conditions, compare equations (12) and (13). With the analytical growth term

$$\psi^{\text{growth}}(J) := \left( J + \frac{1}{J} - 2 \right)^2 \tag{22}$$

and the normalization term introduced by Linden et al. (2023)

$$\psi^{\text{stress}}(J; \mathbf{t}) := -\mathbf{n}(\mathbf{t}) J, \tag{23}$$

the polyconvexity of the model is preserved, compare Linden et al. (2023) for a discussion. Here,

$$\mathbf{n}(\mathbf{t}) := 2 \left( \frac{\partial \psi^{\text{NN}}}{\partial I_1}(\mathbf{t}) + 2 \frac{\partial \psi^{\text{NN}}}{\partial I_2}(\mathbf{t}) + \frac{\partial \psi^{\text{NN}}}{\partial I_3}(\mathbf{t}) - \frac{\partial \psi^{\text{NN}}}{\partial I_3^*}(\mathbf{t}) \right) \Bigg|_{\mathbf{F}=\mathbf{I}} \in \mathbb{R} \tag{24}$$

is a weighted sum of derivatives of the pICNN potential with respect to the invariants for the undeformed state  $\mathbf{F} = \mathbf{I}$ .

In most applications, the stress, meaning the gradient of the potential, compare equation (6), is of interest rather than the potential itself. Here, the gradient of the potential can be evaluated either by using automatic differentiation, or by calculating the derivatives of the NN potential in an explicit way, compare Franke et al. (2023).

#### 4.2. pICNN architectures

Different pICNN architectures applicable to the model are now discussed, which are all based on feed-forward neural networks (FFNNs). From a formal point of view, FFNNs are multiple compositions of vector-valued functions (Aggarwal, 2018). The components are referred to as nodes or neurons, and the function acting in each node is referred to as activation function. The simple structure and recursive definition of FFNNs make them a very natural choice for constructing convex functions. In a nutshell, when the first layer is component-wise convex and every subsequent layer is component-wise convex and nondecreasing, the overall function is convex in its input, compare Section 2. This can also be adapted to partially convex functions, as proposed by Amos et al. (2017).

**Definition 1** (pICNNs). The FFNN

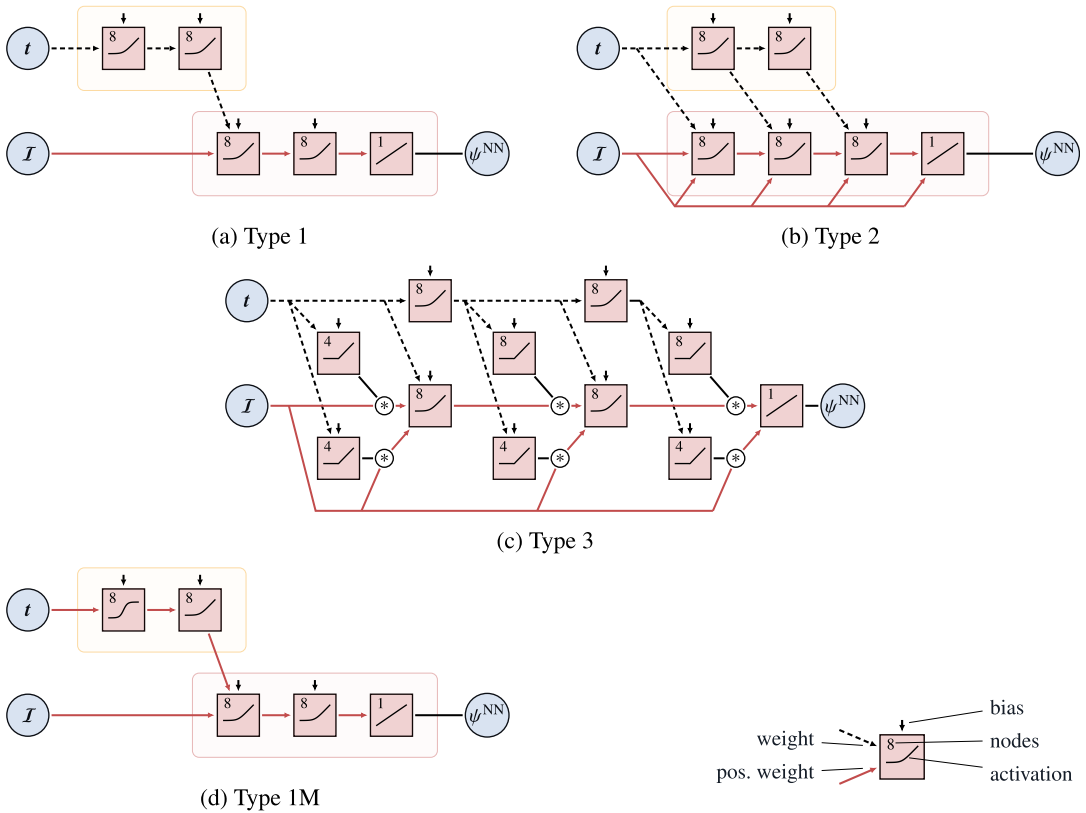
$$\mathcal{P} : \mathbb{R}^m \times \mathbb{R}^n \rightarrow \mathbb{R}, (\mathbf{x}, \mathbf{y}) \mapsto \mathcal{P}(\mathbf{x}, \mathbf{y}) \tag{25}$$

is called a pICNN, if  $\mathcal{P}$  is convex w.r.t.  $\mathbf{x}$ .

In the following, three different pICNN architectures are described. The interrelation between the two inputs and the overall complexity gets gradually more pronounced from Type 1 to Type 3, with Type 3 being a slightly adapted version of the architecture proposed by Amos et al. (2017). The more complex pICNN architectures can be reduced to the simpler ones by constraining a subset of their parameters to take on specific values. For explicit proofs of convexity, the reader is referred to Klein et al. (2022a) and made aware of the fact that, when investigating convexity in  $\mathbf{x}$ , the influence of the nonconvex input  $\mathbf{y}$  can be seen as an additional bias which does not influence convexity in  $\mathbf{x}$ . In addition, an adapted version of the simplest pICNN architecture, which is monotonically increasing in  $\mathbf{y}$ , is discussed. In general, the other two pICNN architectures could be adapted to be monotonically increasing in  $\mathbf{y}$ .

Note that for representing a parametrized polyconvex potential, the pICNN must be convex and nondecreasing in  $\mathbf{x}$ , as discussed in Section 3.2. This requires some adaptations to the general pICNN





**Figure 3.** Different pICNN architectures for the representation of the neural network potential  $\psi^{\text{NN}}$ . For Type 1–3, the NN is convex and nondecreasing in  $\mathcal{J}$ , and can take arbitrary functional relationships in  $\mathbf{t}$ . In addition, for Type 1 M, the NN is monotonically increasing in  $\mathbf{t}$ .

architectures. The adaptations are discussed after introducing the general architectures, and the adapted architectures are visualized in Figure 3 architectures for one specific choice of nodes and layers.

**Proposition 1** (pICNN—Type 1). The pICNN with input  $\mathbf{x} =: \mathbf{x}_0, \mathbf{y} =: \mathbf{y}_0$ , output  $\mathcal{P}(\mathbf{x}, \mathbf{y}) := \mathbf{x}_{H_x+1} \in \mathbb{R}$ , and  $H_x, H_y$  hidden layers

$$\begin{aligned}
 \mathbf{y}_{h+1} &= \sigma_h \left( \mathbf{W}_h^{[yy]} \mathbf{y}_h + \mathbf{b}_h^{[y]} \right) && \in \mathbb{R}^{n_h}, \quad h = 0, \dots, H_y, \\
 \mathbf{x}_1 &= \tilde{\sigma}_0 \left( \mathbf{W}_0^{[xx]} \mathbf{x}_0 + \mathbf{b}_0^{[x]} + \mathbf{W}^{[xy]} \mathbf{y}_{H_y+1} \right) && \in \mathbb{R}^{m_0}, \\
 \mathbf{x}_{h+1} &= \tilde{\sigma}_h \left( \mathbf{W}_h^{[xx]} \mathbf{x}_h + \mathbf{b}_h^{[x]} \right) && \in \mathbb{R}^{m_h}, \quad h = 1, \dots, H_x
 \end{aligned} \tag{26}$$

is convex in  $\mathbf{x}$  given that the weights  $\mathbf{W}_h^{[xx]}$  are nonnegative for  $h \geq 1$  and the activation functions  $\tilde{\sigma}_h$  are convex and nondecreasing for  $h \geq 0$ . The remaining weights, all biases  $\mathbf{b}$ , and the activation functions  $\sigma_h$  can be chosen arbitrarily.

**Proposition 2** (pICNN—Type 2). The pICNN with input  $\mathbf{x} =: \mathbf{x}_0, \mathbf{y} =: \mathbf{y}_0$ , output  $\mathcal{P}(\mathbf{x}, \mathbf{y}) := \mathbf{x}_{H+1} \in \mathbb{R}$ , and  $H$  hidden layers

$$\begin{aligned} \mathbf{y}_{h+1} &= \sigma_h \left( \mathbf{W}_h^{[yy]} \mathbf{y}_h + \mathbf{b}_h^{[y]} \right) && \in \mathbb{R}^{n_h}, \quad h = 0, \dots, H, \\ \mathbf{x}_{h+1} &= \tilde{\sigma}_h \left( \mathbf{W}_h^{[xx]} \mathbf{x}_h + \mathbf{W}_h^{[x_0]} \mathbf{x}_0 + \mathbf{W}_h^{[xy]} \mathbf{y}_h + \mathbf{b}_h^{[x]} \right) && \in \mathbb{R}^{m_h}, \quad h = 0, \dots, H \end{aligned} \tag{27}$$

is convex in  $\mathbf{x}$  given that the weights  $\mathbf{W}_h^{[xx]}$ ,  $h \geq 1$  are nonnegative and the activation functions  $\tilde{\sigma}_h$  are convex and nondecreasing for  $h \geq 0$ . The remaining weights, all biases  $\mathbf{b}$ , and the activation functions  $\sigma_h$  can be chosen arbitrarily.

**Proposition 3** (pICNN—Type 3). The pICNN with input  $\mathbf{x} =: \mathbf{x}_0, \mathbf{y} =: \mathbf{y}_0$ , output  $\mathcal{P}(\mathbf{x}, \mathbf{y}) := \mathbf{x}_{H+1} \in \mathbb{R}$ , and  $H$  hidden layers

$$\begin{aligned} \mathbf{y}_{h+1} &= \sigma_h \left( \mathbf{W}_h^{[yy]} \mathbf{y}_h + \mathbf{b}_h^{[y]} \right) && \in \mathbb{R}^{n_h}, \quad h = 0, \dots, H, \\ \mathbf{x}_{h+1} &= \tilde{\sigma}_h \left( \mathbf{W}_h^{[xx]} \left( \mathbf{x}_h * \left[ \tilde{\mathbf{W}}_h^{[xy]} \mathbf{y}_h + \tilde{\mathbf{b}}_h^{[x]} \right]_+ \right) + \right. && \\ &\quad \left. \mathbf{W}_h^{[x_0]} \left( \mathbf{x}_0 * \left[ \tilde{\mathbf{W}}_h^{[x_0y]} \mathbf{y}_h + \tilde{\mathbf{b}}_h^{[x_0]} \right]_+ \right) + \right. && \\ &\quad \left. \mathbf{W}_h^{[xy]} \mathbf{y}_h + \mathbf{b}_h^{[x]} \right) && \in \mathbb{R}^{m_h}, \quad h = 0, \dots, H \end{aligned} \tag{28}$$

is convex in  $\mathbf{x}$  given that the weights  $\mathbf{W}_h^{[xx]}$ ,  $h \geq 1$  are nonnegative and the activation functions  $\tilde{\sigma}_h$  are convex and nondecreasing for  $h \geq 0$ . The remaining weights, all biases  $\mathbf{b}$ , and the activation functions  $\sigma_h$  can be chosen arbitrarily.

**Proposition 4** (pICNN—Type 1 M). The pICNN with input  $\mathbf{x} =: \mathbf{x}_0, \mathbf{y} =: \mathbf{y}_0$ , output  $\mathcal{P}(\mathbf{x}, \mathbf{y}) := \mathbf{x}_{H_x+1} \in \mathbb{R}$ , and  $H_x, H_y$  hidden layers

$$\begin{aligned} \mathbf{y}_{h+1} &= \sigma_h \left( \mathbf{W}_h^{[yy]} \mathbf{y}_h + \mathbf{b}_h^{[y]} \right) && \in \mathbb{R}^{n_h}, \quad h = 0, \dots, H_y, \\ \mathbf{x}_1 &= \tilde{\sigma}_0 \left( \mathbf{W}_0^{[xx]} \mathbf{x}_0 + \mathbf{b}_0^{[x]} + \mathbf{W}^{[xy]} \mathbf{y}_{H_y+1} \right) && \in \mathbb{R}^{m_0}, \\ \mathbf{x}_{h+1} &= \tilde{\sigma}_h \left( \mathbf{W}_h^{[xx]} \mathbf{x}_h + \mathbf{b}_h^{[x]} \right) && \in \mathbb{R}^{m_h}, \quad h = 1, \dots, H_x \end{aligned} \tag{29}$$

is convex in  $\mathbf{x}$  and monotonically increasing in  $\mathbf{y}$  given that the weights  $\mathbf{W}_h^{[xy]}, \mathbf{W}_h^{[xx]}$ ,  $h \geq 1$ , and  $\mathbf{W}_h^{[yy]}$ ,  $h \geq 0$ , are nonnegative, the activation functions  $\tilde{\sigma}_h, h \geq 0$ , are convex and nondecreasing, and the activation functions  $\sigma_h, h \geq 0$ , are nondecreasing. If at least one activation function  $\sigma_h$  is not convex, the pICNN is not convex in  $\mathbf{y}$ . The remaining weights and all biases  $\mathbf{b}$  can be chosen arbitrarily and the activation functions  $\sigma_h$ .

To construct pICNNs which are convex and *nondecreasing* in  $\mathbf{x}$ , also the weights acting directly on  $\mathbf{x}$  must be nonnegative. This means that for all types,  $\mathbf{W}_h^{[xx]}$  has to be nonnegative for all  $h$ . Both Type 2 and Type 3 use so-called passthrough layers, which pass the argument  $\mathbf{x}$  into every hidden layer. In conventional (p)ICNNs, passthrough layers have a significant benefit. Here, the NN must not necessarily be nondecreasing in the input, as naturally, convex functions can also be decreasing, compare Section 2. Thus, the weights acting directly on the input may take positive or negative values. Using passthrough

layers exploits this benefit in every layer of the NN. However, as in the application to polyconvexity, also the weights of the passthrough layer  $W_h^{[x_0]}$  must be nonnegative, their benefit is limited.

Furthermore, as only the gradient of the potential is considered in this work, compare equation (6), all contributions to the output layer which are independent of the invariants are omitted, such as the bias in the output layer and the last two parameter layers of Type 2.

Throughout this work, the convex and nondecreasing Softplus activation function, compare Figure 1, is applied in all hidden layers for both  $\sigma_h$  and  $\tilde{\sigma}_h$ , except for Type 1 M, where the monotonically increasing but nonconvex Sigmoid activation function is applied in the first layer of the parameter input. In the output layer, a linear activation function is applied. By this, the potential is infinitely continuously differentiable in  $\mathbf{x}$ . Type 1 and Type 2 are also infinitely continuously differentiable in  $\mathbf{y}$ . However, due to the application of the ReLu function in Type 3, this architecture is not continuously differentiable in  $\mathbf{y}$ . This could be circumvented by applying any positive and continuously differentiable function instead of ReLu, for example, the Softplus function. Note again that the adapted architectures are visualized in Figure 3 for one specific choice of nodes and layers.

### 5. Numerical examples

In this section, the models proposed in this work are calibrated to data generated with analytical parametrized potentials. In this way, generating datasets with a large variety of parameter combinations and deformation scenarios is straightforward, which helps providing first insights to the behavior of the parametrized PANN models. In Section 5.1, the models are calibrated to data generated with a Neo-Hookean potential including one parameter. In Section 5.2, the models are calibrated to data generated with a Neo-Hookean potential which includes two parameters which are inspired by a 3D-printing process.

#### 5.1. Scalar-valued parametrization

##### 5.1.1. Data generation

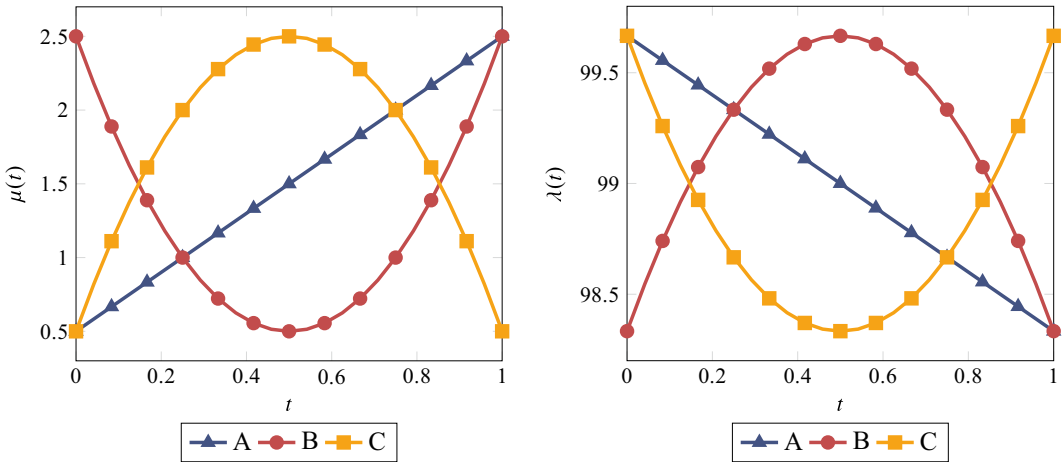
As a first proof of concept, the models proposed in Section 4 are calibrated to data generated with the parametrized Neo-Hookean potential introduced in equation (19). For this, three different parametrizations

$$\mu(t) = \begin{cases} 0.5 + 2t, & \text{Case A} \\ 8t^2 - 8t + 2.5, & \text{Case B,} \\ -8t^2 + 8t + 0.5, & \text{Case C} \end{cases} \quad \lambda(t) = \kappa - \frac{2}{3}\mu(t), \quad t \in [0, 1], \quad (30)$$

of the Lamé parameters  $\mu, \lambda$  with a constant bulk modulus  $\kappa = 100$  are applied. The different parametrizations are chosen such that the hyperelastic potential has both convex and concave dependencies on the parameter  $t$ , compare Figure 4. Thus, the pICNN Types 1–3 are examined, meaning the architectures which represent arbitrary functional relationships in the parameter. Overall, discrete values for both the deformation gradient  $\mathbf{F}$  and the scalar parameter  $t$  have to be sampled for the data generation, resulting in datasets of the form

$$\mathcal{D} = \{ ({}^1\mathbf{F}, {}^1t; {}^1\mathbf{P}), \dots, ({}^n\mathbf{F}, {}^nt; {}^n\mathbf{P}) \}, \quad (31)$$

where in each tuple, the prescribed deformation gradient  ${}^i\mathbf{F}$  and the parameter  ${}^it$  have a corresponding first Piola–Kirchhoff stress  ${}^i\mathbf{P}$ . As the data is generated with an analytical potential, also the values of the potential  $\psi({}^i\mathbf{F}, {}^it)$  are available and could be included in the dataset. However, as real-world experiments only provide stress values, this would be a less general approach. Also, even when data on the potential is available, including it in the calibration process does barely improve the model quality (Klein et al., 2022a). Thus, the potential is calibrated only through its gradients, which is referred to as Sobolev training (Vlassis and Sun, 2021).



**Figure 4.** Three different parametrizations (Cases A, B, C) of the Lamé coefficients  $\mu(t)$  and  $\lambda(t)$  in the Neo-Hookean model.

Following Fernández et al. (2022), the sampling of the stress–strain states is motivated by physical experiments which could also be applied in experimental investigations. In particular, a uniaxial tension stress state, a biaxial tension stress state, and a shear deformation state are applied, where each load case consists of 101 datapoints, and the data is generated by numerically solving the underlying equation systems for each load case. The uniaxial tension is applied in  $x$ -direction with  $F_{11} \in [0.5, 1.5]$ , the equibiaxial tension is applied in  $x - y$ -direction with  $F_{11} = F_{22} \in [0.5, 1.5]$ , and simple shear is applied with  $F_{12} \in [-0.5, 0.5]$ . Since hyperelastic potentials are usually formulated in terms of strain invariants, compare Section 3.2, for the test dataset to be representative, the space of invariants should be considered rather than the space of deformation gradients. Thus, for testing purposes, two general deformation modes are used, corresponding to an interpolation of the training cases in the invariant space. For this, the deformation gradient

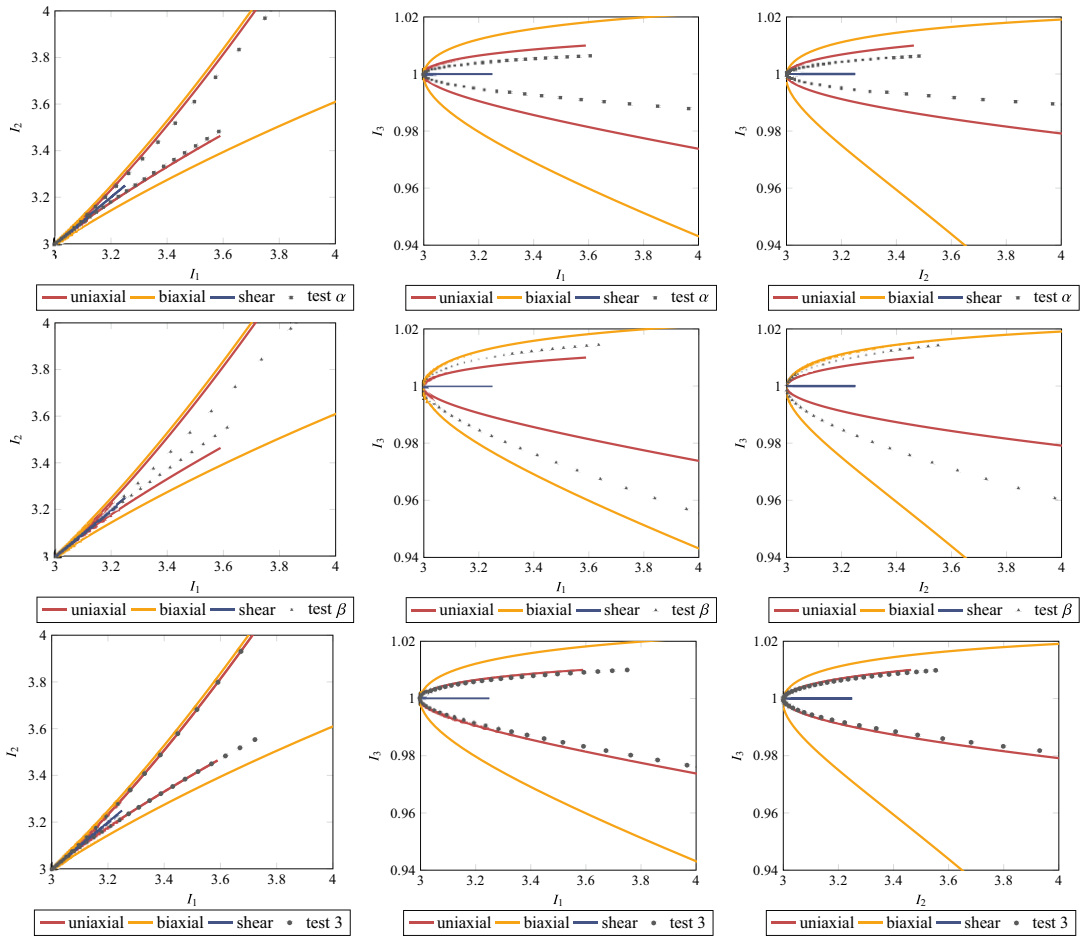
$$\mathbf{F} = \begin{bmatrix} \lambda & 0 & 0 \\ 0 & \lambda^m & 0 \\ 0 & 0 & F_{33} \end{bmatrix}, \quad P_{33} = 0, \quad \lambda \in [0.5, 1.5] \tag{32}$$

is applied, where  $F_{33}$  is calculated by solving the corresponding system of equations. This deformation gradient is inspired by Baaser et al. (2013), where a similar deformation is applied to sample the space of isotropic invariants at incompressibility. For  $m = -0.7$ , the deformation state, denoted as test  $\alpha$ , represents an interpolation of uniaxial tension and shear (cf. Figure 5, first column). Corresponding to  $m = -0.18$ , test  $\beta$  interpolates uniaxial and biaxial tension (cf. Figure 5, second column).

Furthermore, a mixed shear-tension deformation mode is investigated, compare “test 3” in Fernández et al. (2021). Here, the deformation gradient

$$\mathbf{F} = \begin{bmatrix} 1 + 0.5\lambda & 0.4\lambda & 0 \\ 0 & F_{22} & 0 \\ 0 & 0 & F_{33} \end{bmatrix}, \quad P_{22} = P_{33} = 0, \quad \lambda \in [-1, 1] \tag{33}$$

is considered, where  $F_{22}, F_{33}$  are calculated by solving the corresponding system of equations. Surprisingly, despite seeming like a more general deformation gradient than equation (32), in the invariant space, this load case is almost identical to uniaxial tension (cf. Figure 5, third column). Thus, the mixed shear-tension test as introduced in Fernández et al. (2021) for a metamaterial model with cubic symmetry, is not as general for isotropic materials, and for most investigations to follow, test  $\alpha$  and  $\beta$  are applied.



**Figure 5.** Load paths of the test cases in the invariant space for  $\mu = 1.5$ . First row: test  $\alpha$ , second row: test  $\beta$ , and third row: mixed shear-tension test. First column:  $I_1 - I_2$  plane, second column:  $I_1 - I_3$  plane, third column:  $I_2 - I_3$  plane.

It should be noted that in this work, only isotropic material behavior is considered, which allows for the generation of general deformation modes using only main diagonal components of the deformation gradient. For general anisotropic materials, compare Ebbing (2010), this strategy might not be applicable. Then, other data generation strategies such as extracting deformation modes out of complex FE simulations (Kalina et al., 2023), or sampling the space of physically sensible deformation gradients (Kunc and Fritzen, 2019) could be applied. Nevertheless, the test cases applied in this work and particularly Figure 5 demonstrates that the generality of a deformation mode should always be investigated in the invariant space, rather than only investigating deformation gradients, compare Kalina et al. (2022) and Kalina et al. (2023).

### 5.1.2. Model preparation and calibration

In this example, the pICNN architectures with an arbitrary functional relationship in the parameter  $t$  are applied, that is, Type 1–3. The hyperparameters, that is, the number of nodes and layers, of the different pICNN architectures described in Section 4.2 are chosen such that they are in the same order of magnitude for all models. The number of nodes and layers are visualized in Figure 3. The total number of trainable

parameters for the models using the Type 1–3 pICNNs are 272, 516, and 580, respectively. For the model calibration, the loss function given as the mean squared error

$$MSE = \frac{1}{9lmn} \sum_{i=1}^l \sum_{j=1}^m \frac{1}{w_{ij}} \sum_{k=1}^n \left\| {}^{ijk}P - P^{(ik}F; j_t) \right\|^2 \tag{34}$$

is minimized. Here, the outer loop over  $i$  corresponds to the  $l$  load paths in the calibration dataset. Each load path is combined with  $m$  different, fixed  $t$  values, where the sum over  $j$  corresponds to the values of  $t$ . Finally, for one fixed combination of load path and parameter  $t$ , the weight is calculated according to the norm

$$w_{ij} = \frac{1}{n} \sum_{k=1}^n \left\| {}^{ijk}P \right\|, \tag{35}$$

and the innermost sum over  $k$  corresponds to the  $n$  different deformation gradients. For the evaluation of the loss after the model calibration, all weights are set to one, that is,  $w_{ij} = 1$ . The models are implemented in TensorFlow 2.10.0, using Python 3.10.9. For the optimization, the Adam optimizer is used with a learning rate of 0.002 and 7,000 epochs. The full batch of training data is used with TensorFlow’s default batch size. Although the number of parameters is in the same order of magnitude for all model architectures, it should be noted that the calibration process is affected by many hyperparameters (Nakkiran et al., 2021), which suggests that the optimal calibration process for each pICNN architecture might be different. Nevertheless, with the applied calibration strategy, all model architectures show excellent results in the examples to follow.

*Study I:* For the calibration dataset, the uniaxial, biaxial, and shear loads are combined with  $t \in \{0, 0.2, 0.4, 0.6, 0.8, 1\}$ . Thus, the calibration dataset consists of 1,818 tuples. For the test dataset, the test load cases (test  $\alpha$  and test  $\beta$ ) are combined with all remaining 195 values for  $t$  not included in the calibration, thus consisting of 19,695 tuples. For this study, the PANN model as described in Section 4 is applied, with three different versions using the pICNN architectures Type 1–3 as described in 4.2. Each model is calibrated five times to each parametrization case, where the model with the worst test loss is excluded.

*Study II:* For the calibration dataset, the uniaxial, biaxial, and shear-tension loads are combined with  $t \in \{0, 0.1, 0.9, 1\}$ , yielding a calibration dataset with 1,212 tuples. For the test dataset, the mixed shear-tension test is combined with the 197 remaining values for  $t$  not included in the calibration, thus consisting of 19,897 tuples. For this study, the model as described in Section 4 is adapted in such a way that it does not include the normalization term equation (13). Here, the pICNN architecture Type 1 is applied. One model instance with the normalization condition, and one without is calibrated. The models are calibrated one time to the parametrization case A.

5.1.3. Results

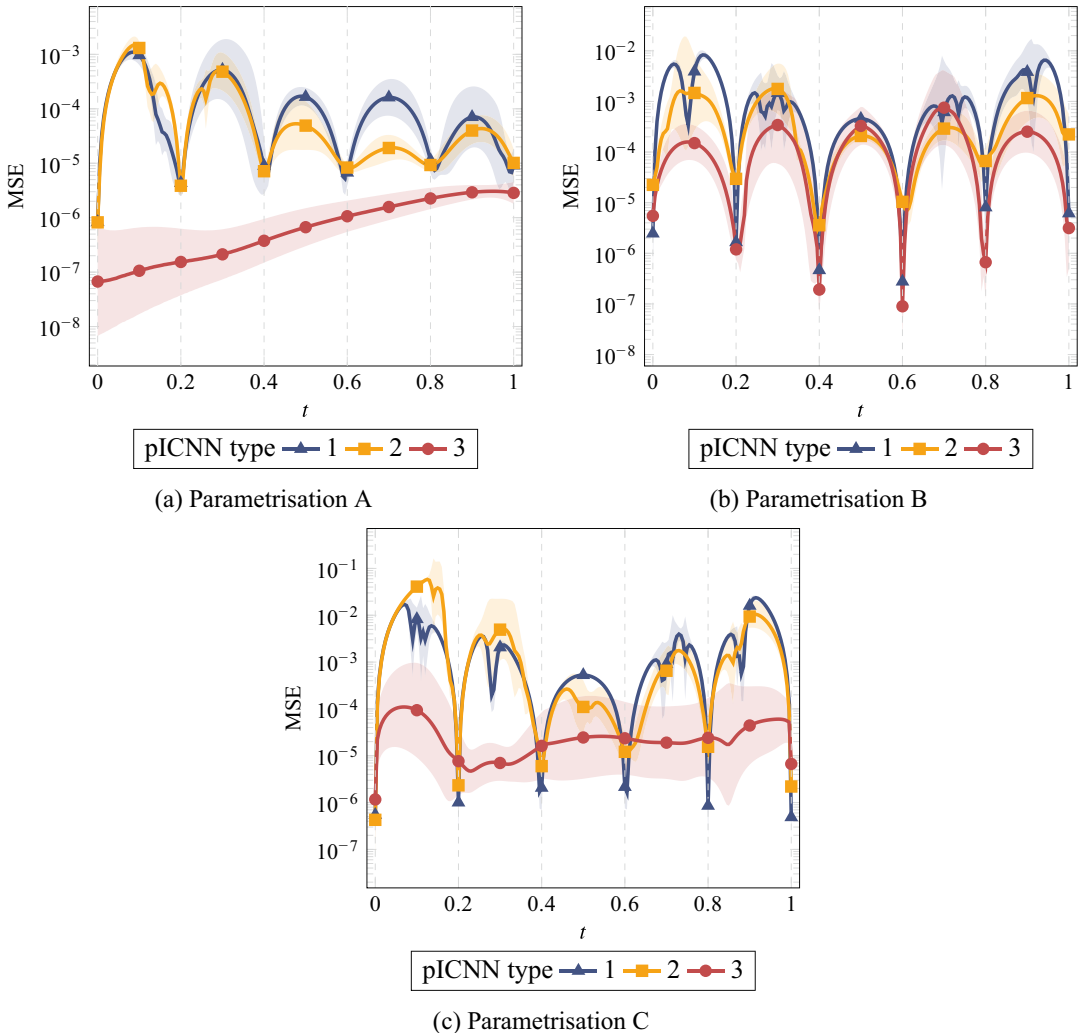
*Study I:* In Table 1, the MSE values of the calibrated models are presented for all pICNN architectures and all parametrization cases. In general, all pICNN architectures are able to interpolate the data for all

**Table 1.** Average  $\log_{10}$  MSE for the scalar-valued parametrization

|        | Calibration |        |        | Test   |        |        |
|--------|-------------|--------|--------|--------|--------|--------|
|        | Case A      | Case B | Case C | Case A | Case B | Case C |
| Type 1 | −4.54       | −4.44  | −5.24  | −3.63  | −2.72  | −2.47  |
| Type 2 | −4.55       | −3.26  | −4.31  | −3.74  | −2.99  | −2.15  |
| Type 3 | −5.42       | −4.74  | −3.62  | −5.90  | −3.49  | −3.75  |

Note. Four best-calibrated model instances for study I for all parametrization cases and pICNN types.

parametrization cases and also show excellent performance on the test dataset for all parametrizations. In general, the pICNN Type 3 performs better on the test dataset than the remaining architectures. However, due to the simplicity of the examined data, no premature conclusions about the general performance of the different architectures should be drawn. Even the architecture with the lowest complexity might be sufficiently flexible in practical applications. Furthermore, the reduced complexity of Type 1 might be advantageous in some applications, for example, when implementing the model in a finite-element code. The MSEs were evaluated for the test load cases (test  $\alpha$  and  $\beta$ ) and all values of  $t$  are visualized in Figure 6. Not surprisingly, the models perform better for values of  $t$  which were included in the model calibration. In particular for parametrization A and C, the pICNN Type 3 performs way better than the other architectures. This superior performance of pICNN Type 3 might be explained by the multiplicative operations between invariants and parameters in its architecture, which enables it to more accurately and easily describe the Neo-Hookean potential used for data generation, compare equation (19). Note that, when leaving the training values of  $t$ , the MSE increases quite quickly for Type 1 and 2, which could be a sign of overfitting in the parameter  $t$ . For parametrization B, the pICNN architecture 3 shows a similar behavior

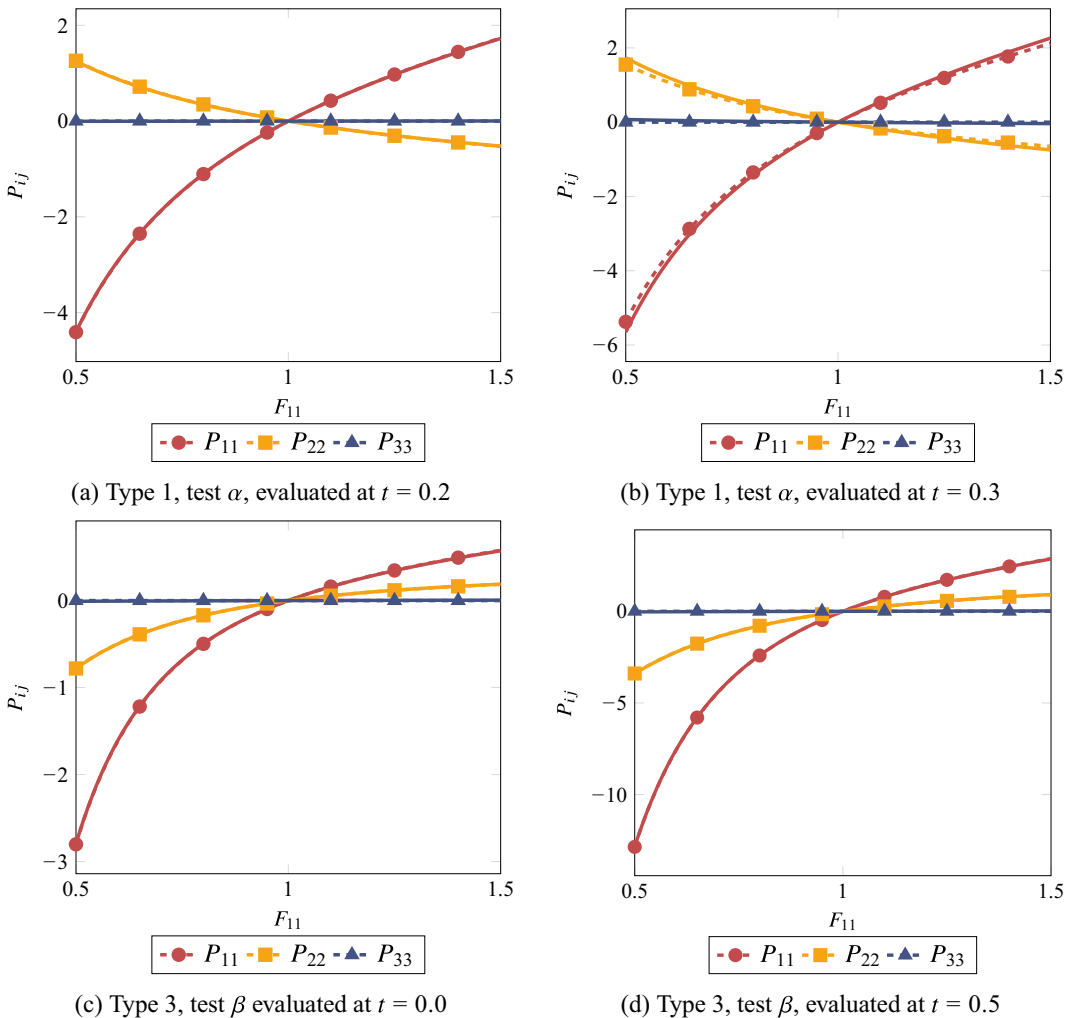


**Figure 6.** Evaluation of the test cases. Continuous lines denote the average of  $\log_{10}$  MSE, while shaded areas denote the standard deviation of  $\log_{10}$  MSE.

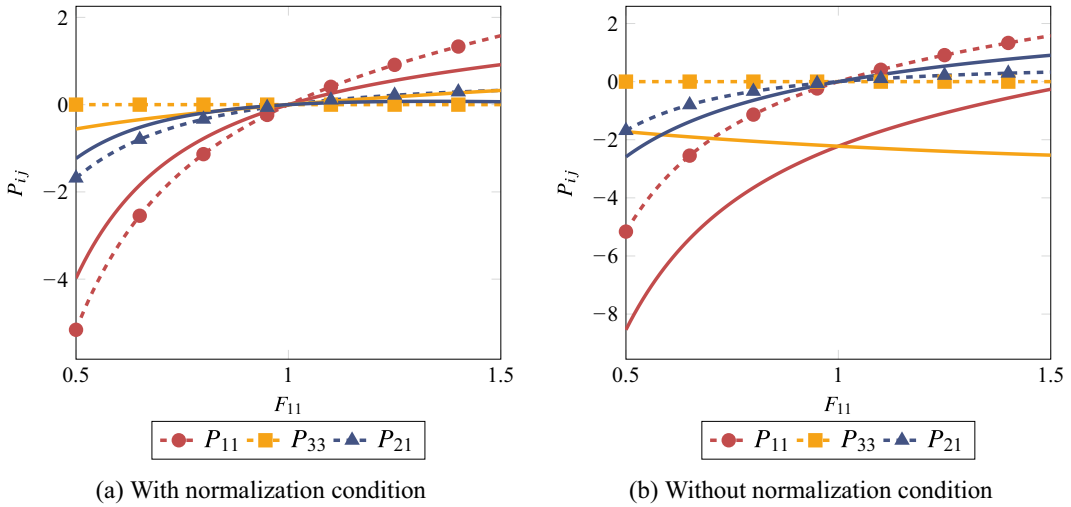


compared to the other pICNN architectures. While pICNN Type 1 and 2 have quite similar prediction qualities between different calibration instances, pICNN Type 3 has a higher discrepancy between different calibration instances, as indicated by the shaded areas.

For the following investigation, a random instance of the four trained models was chosen. In Figure 7, some stress predictions of the models are visualized for the parametrization case C. On the top row, a model using a Type 1 pICNN is evaluated on test  $\alpha$ , while on the bottom row, a model using a Type 3 pICNN is evaluated on test  $\beta$ . On the left-hand side, the models are evaluated for values of  $t$  used in the calibration, while on the right-hand side, the models are evaluated for values of  $t$  not used in the calibration. In all cases, the model has to extrapolate in the load case. The interpolation is excellent for both evaluated models. For Type 1, the evaluation for  $t = 0.3$  shows some visible deviations from the ground truth. This case has a  $\log_{10}$  MSE of  $-2.66$ . Thus, with the MSEs of Figure 6 in mind, this is a representative case for the less good model predictions. And still, the prediction quality might be good enough for most practical applications. The pICNN Type 3 is able to also perfectly make predictions at  $t = 0.5$ , although the magnitude of the stress components differs by a factor of  $\approx 4$  for different values of  $t$ .



**Figure 7.** Results for parametrization case C, evaluated for the test cases. Dashed lines and points denote the data, while continuous lines denote the model prediction.



**Figure 8.** Results for parametrization case C, evaluated for the mixed shear-tension case. In this case, the model was only calibrated on the edges of the parameter domain of  $t$ , and evaluated in the middle. Dashed lines and points denote the data, while continuous lines denote the model prediction.

*Study II:* In Figure 8, a comparison is made between a model which fulfills the normalization condition of equation (13) by construction and one which only learns to approximate the condition through the calibration dataset. This case includes values of  $t$  only on the edges of its parameter domain and is here evaluated for the value in the middle. While for the model which fulfills the normalization condition by construction, there are some significant deviations from the ground truth, it still fulfills the normalization condition in an exact way. The model which does not fulfill the normalization condition by construction has to learn it by the data which fulfill this property. While this indeed works out for the values of  $t$  included in the calibration dataset, for the case visualized here it is violated quite obviously. This is despite the used deformation mode, the mixed shear-tension test, being very close to the uniaxial deformation mode contained in the training set, compare Figure 5. This example demonstrates the benefit of fulfilling mechanical conditions by construction, in particular for the generalization of the model.

**5.2. Vector-valued parametrization with monotonicity condition**

**5.2.1. Data generation**

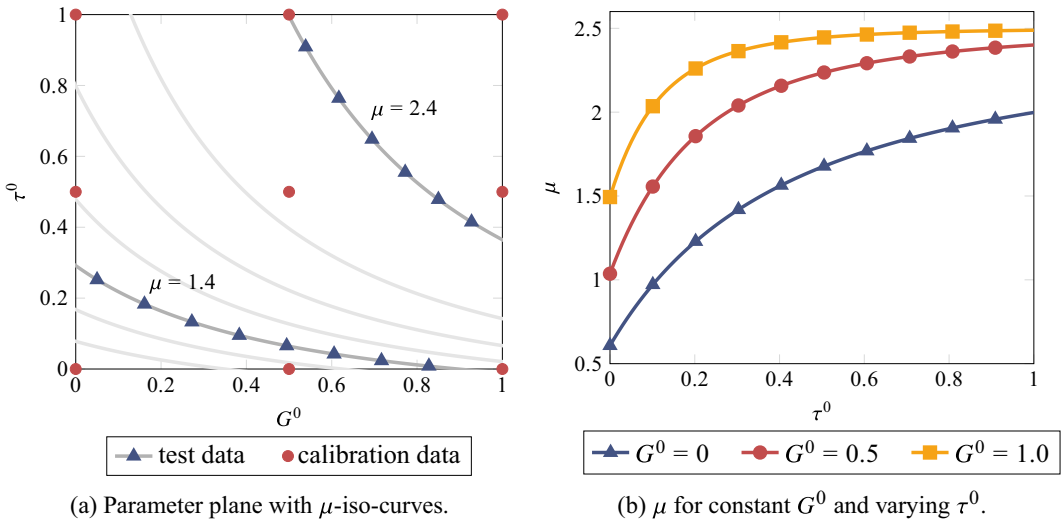
In the next example, the Neo-Hookean potential

$$\psi^{nh}(I_1, I_3; \mathbf{t}) = \frac{\mu(\mathbf{t})}{2} (I_1 - 3 - 2 \ln \sqrt{I_3}) + \frac{\lambda}{2} (\sqrt{I_3} - 1)^2 \tag{36}$$

is considered, where  $\mu(\mathbf{t})$  is parametrized in  $\mathbf{t} = (G^0, \tau^0) \in [0, 1]^2$  and  $\lambda = 100 = \text{const}$ . The parametrization

$$\begin{aligned} \mu(\mathbf{t}) &= 2.5 \tanh H^v, & H^v &= 1.7 G^2 \ln \hat{\tau}, \\ \hat{G} &= 0.6 + 0.4G^0, & \hat{\tau} &= 1.5 + 4.5\tau^0, \end{aligned} \tag{37}$$

is inspired by a 3D printing process, where a liquid photopolymer resin is hardened by exposing it to ultraviolet light for a given time, compare Valizadeh et al. (2021). The properties of the final solid material depend on both the light intensity, which is determined through the greyscale value  $\hat{G}$ , and the time  $\hat{\tau}$  the light is applied on the resin. Here, these two parameters are parameterized in a physically sensible range through  $G^0$ , which is associated with the light intensity, and  $\tau^0$ , which is associated with the exposure time for which the light is applied on the resin. The parametrization consists of three ideas. First



**Figure 9.** Dependency of the shear modulus  $\mu$  on the vector-valued, 3D printing-inspired parametrization in terms of  $(G^0, \tau^0)$ .

of all, the shear modulus  $\mu$  is influenced by both  $G^0$  and  $\tau^0$ . In particular, the same shear modulus can be achieved by different combinations of  $(G^0, \tau^0)$ . This is reflected by the intermediate quantity  $H^v$ , compare equation (37). Secondly, the shear modulus is bounded from above, which is reflected by the tanh function which receives  $H^v$  as an input. Lastly, the shear modulus is a monotonically increasing function in  $(G^0, \tau^0)$ , which reflects the physical observation that the shear modulus increases when increasing the light intensity or the light exposure time. In Figure 9, these characteristics are visualized.

The deformation gradients are sampled as described in Section 5.1.1. For the calibration dataset, nine parameter combinations  $\mathbf{t} = (G^0, \tau^0)$  are sampled, compare Figure 9. For the test dataset, two  $\mu$ -iso-curves are considered for  $\mu \in \{1.4, 2.4\}$ , compare Figure 9. For each iso-curve, 100  $(G^0, \tau^0)$  tuples are sampled. Overall, this results in a calibration dataset with 2,727 tuples and a test dataset with 20,200 tuples.

### 5.2.2. Model preparation and calibration

In this example, the pICNN architecture with a monotonically increasing functional relationship in the parameter  $\mathbf{t}$  is applied, that is, Type 1 M. The number of nodes and layers is visualized in Figure 3, where the total number of trainable parameters is 280. For the model calibration, all stress values are normalized by the inverse mean Frobenius norm of all tuples in the calibration dataset. Then, the loss function is given as the mean squared error

$$\text{MSE} = \frac{1}{9n} \sum_{i=1}^n \frac{1}{w_i} \left\| \mathbf{P} - \mathbf{P}(^i\mathbf{F}; \mathbf{t}) \right\|^2 \tag{38}$$

is minimized, where  $n$  is the number of tuples in the calibration dataset. The sample weight

$$w_i = \left\| \mathbf{P} \right\| + 1 \geq 1 \tag{39}$$

is calculated for each single tuple and used to encourage better accuracy of the model when predicting small stress values. For the optimization, the SLSQP optimizer is used.

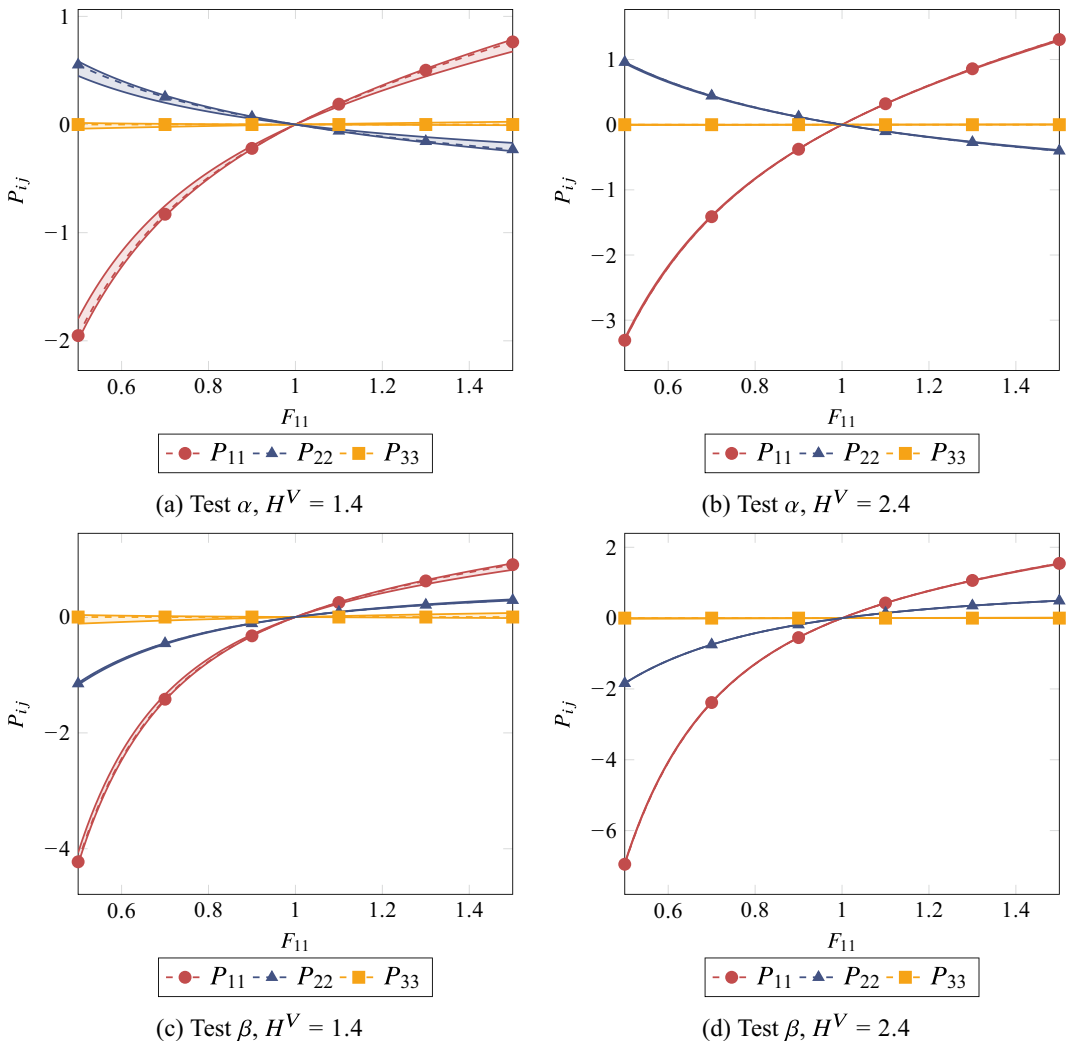
### 5.2.3. Results

In Table 2, the average  $\log_{10}$  MSE of the four model instances with the best test MSE, as well as the  $\log_{10}$  MSE of the model instance with the best test MSE are presented. The performance on both the calibration and the test dataset is excellent. In Figure 10, the stress predictions for the test cases for  $H^v \in \{1.4, 2.4\}$  are

**Table 2.** Average  $\log_{10}$  MSE for the vector-valued parametrization

|                      | Calibration | Test  |
|----------------------|-------------|-------|
| Average of best four | -4.37       | -3.26 |
| Best model           | -5.18       | -3.73 |

Note. Four best-calibrated model instances.



**Figure 10.** Model prediction for the test cases. Dashed lines and points denote the data, while lines and shaded areas depict the calibrated model evaluated for different parameter combinations  $(G^0, \tau^0)$  on  $H^v$  iso-curves.

visualized for the best model instance. The calibrated model is evaluated for 100 different  $(G^0, \tau^0)$  combinations on each  $\mu$ -iso-curve. For  $H^v = 2.4$ , the model perfectly learns the invariance of  $H^v$  in  $(G^0, \tau^0)$ . Thus, the model predictions for different  $(G^0, \tau^0)$  combinations are practically identical. For  $H^v = 1.4$ , the model predictions slightly differ for different  $(G^0, \tau^0)$  combinations, which is indicated by the (fairly small) shaded areas in Figure 10. This can be traced back to the larger sensitivity of the ground

truth models material parameter  $\mu$  at smaller  $H^V$  values, compare equation (36) and Figure 9. Overall, the model performs excellent, in particular with regard to the low amount of  $(G^0, \tau^0)$  samples in the calibration dataset.

## 6. Conclusion

In the present work, a NN-based constitutive model for parametrized hyperelasticity is proposed. The model is formulated in such a way that it fulfills all common constitutive conditions of hyperelasticity by construction, without being too restrictive in the parametric dependencies of the model. In particular, by applying pICNNs, the model fulfills the polyconvexity condition, while still being able to represent arbitrary functional relationships in the additional parameter. In addition, a polyconvex potential is proposed which is monotonic in the additional parameters.

As a first proof of concept, the model is calibrated to data generated with an analytical potential which depends on one scalar-valued parameter. Different pICNN architectures with different complexities are examined, where all architectures performed excellent. However, due to the simplicity of the examined data, no premature conclusions about the general performance of the different architectures should be drawn. Even the architecture with the lowest complexity might be sufficiently flexible in practical applications, while its easier and thus computationally more efficient model structure can be advantageous. Furthermore, the proposed model is calibrated to data generated with an analytical potential which depends on multiple parameters. In this case, the dependency of the ground truth potential in the parameters is monotonic, and the NN-based potential which by construction is monotonic in the additional parameters is applied. Again, the model shows excellent performance.

The extension of the proposed framework to multiphysical constitutive models, such as electroelasticity (Klein et al., 2022b), will be straightforward, as well as the application in finite-element analysis (Franke et al., 2023) and optimization of microstructured materials (Ortigosa et al., 2023). Furthermore, the application to real-world experimental data of composites or polymer materials with varying constituents is targeted.

**Data availability statement.** The authors provide access to the simulation data required to reproduce the results through the public GitHub repository at <https://github.com/CPSHub/sim-data>.

**Author contribution.** Conceptualization: D.K.K., F.J.R., I.V., O.W.; Formal analysis: D.K.K., F.J.R.; Funding acquisition: O.W.; Investigation: F.J.R.; Methodology: D.K.K., F.J.R., I.V., O.W.; Resources: O.W.; Software: D.K.K., F.J.R.; Validation: F.J.R.; Visualization: D.K.K., F.J.R.; Writing—original draft: D.K.K.; Writing—review and editing: D.K.K., F.J.R., I.V., O.W.

**Funding statement.** This research was supported by the Deutsche Forschungsgemeinschaft (DFG—German Research Foundation)—Grant No. 492770117 and the Graduate School of Computational Engineering within the Centre of Computational Engineering at TU Darmstadt.

**Competing interest.** The authors declare none.

**Ethical standard.** The research meets all ethical guidelines, including adherence to the legal requirements of the study country.

## References

- Aggarwal CC (2018) *Neural Networks and Deep Learning*, 1st Edn. Cham: Springer International Publishing.
- Amos B, Xu L and Kolter JZ (2017) Input convex neural networks. *Proceedings of the 34th International Conference on Machine Learning* 70, 146–155.
- As'ad F and Farhat C (2023) A mechanics-informed neural network framework for data-driven nonlinear viscoelasticity. AIAA SCITECH 2023 Forum.
- Baaser H, Hopmann C and Schobel A (2013) Reformulation of strain invariants at incompressibility. *Archive of Applied Mechanics* 83(2), 273–280.
- Baldi P, Cranmer K, Faucett T, Sadowski P and Whiteson D (2016) Parametrized neural networks for high-energy physics. *The European Physical Journal C* 76, 235.
- Ball JM (1976) Convexity conditions and existence theorems in nonlinear elasticity. *Archive for Rational Mechanics and Analysis* 63(4), 337–403.

- Ball JM** (1977) Constitutive inequalities and existence theorems in nonlinear elasto-statics. *Herriot Watt Symposium: Nonlinear Analysis and Mechanics 1*, 187–241.
- Calafiore GC, Gaubert S and Corrado P** (2020) Log-sum-exp neural networks and posynomial models for convex and log-log-convex data. *IEEE Transactions on Neural Networks and Learning Systems* 31(3), 827–838.
- Chen G and Guilleminot J** (2022) Polyconvex neural networks for hyperelastic constitutive models: A rectification approach. *Mechanics Research Communications* 125, 103993.
- Do H, Tan YY, Ramos N, Kiendl J and Weeger O** (2020) Nonlinear isogeometric multiscale simulation for design and fabrication of functionally graded knitted textiles. *Composites Part B: Engineering* 202, 108416.
- Ebbing V** (2010) *Design of Polyconvex Energy Functions for All Anisotropy Classes*. PhD Thesis, Universität Duisburg-Essen.
- Fernández M, Fritzen F and Weeger O** (2022) Material modeling for parametric, anisotropic finite strain hyperelasticity based on machine learning with application in optimization of metamaterials. *International Journal for Numerical Methods in Engineering* 123, 577–609.
- Fernández M, Jamshidian M, Böhlke T, Kersting K and Weeger O** (2021) Anisotropic hyperelastic constitutive models for finite deformations combining material theory and data-driven approaches with application to cubic lattice metamaterials. *Computational Mechanics* 67 (2), 653–677.
- Franke M, Klein DK, Weeger O and Betsch P** (2023) Advanced discretization techniques for hyperelastic physics-augmented neural networks. *Computer Methods in Applied Mechanics and Engineering* 416, 116333.
- Gärtner T, Fernández M and Weeger O** (2021) Nonlinear multiscale simulation of elastic beam lattices with anisotropic homogenized constitutive models based on artificial neural networks. *Computational Mechanics* 68, 1111–1130.
- Hencky H** (1928) Über die form des Elastizitätsgesetzes bei ideal elastischen Stoffen. *Zeitschrift für Technische Physik* 9, 215–220.
- Holzäpfel GA** (2000) *Nonlinear Solid Mechanics: A Continuum Approach for Engineering*, 2nd Edn. Chichester: Wiley.
- Hornik K** (1991) Approximation capabilities of multilayer feedforward networks. *Neural Networks* 4(2), 251–257.
- Huang S, He Z, Chem B and Reina C** (2021) Variational Onsager neural networks (VONNs): A thermodynamics-based variational learning strategy for non-equilibrium PDEs. *Journal of the Mechanics and Physics of Solids* 163, 104856.
- Kalina KA, Linden L, Brummund J and Kästner M** (2023) FEANN: An efficient data-driven multiscale approach based on physics-constrained neural networks and automated data mining. *Computational Mechanics* 71, 827–851.
- Kalina KA, Linden L, Brummund J, Metsch P and Kästner M** (2022) Automated constitutive modeling of isotropic hyperelasticity based on artificial neural networks. *Computational Mechanics* 69, 213–232.
- Klein DK, Fernández M, Martin RJ, Neff P and Weeger O** (2022a) Polyconvex anisotropic hyperelasticity with neural networks. *Journal of the Mechanics and Physics of Solids* 159, 104703.
- Klein DK, Ortigosa R, Martínez-Frutos J and Weeger O** (2022b) Finite electro-elasticity with physics-augmented neural networks. *Computer Methods in Applied Mechanics and Engineering* 400, 115501.
- Kružík M and Roubíček T** (2019) *Mathematical Methods in Continuum Mechanics of Solids*, 1st Edn. New York: Springer International Publishing.
- Kumar S and Kochmann DM** (2022) What machine learning can do for computational solid mechanics. In *Current Trends and Open Problems in Computational Mechanics*. Cham: Springer, pp. 275–285.
- Kunc O and Fritzen F** (2019) Finite strain homogenization using a reduced basis and efficient sampling. *Mathematical and Computational Applications* 24(2), 56.
- Linden L, Klein DK, Kalina KA, Brummund J, Weeger O and Kästner M** (2023) Neural networks meet hyperelasticity: A guide to enforcing physics. *Journal of the Mechanics and Physics of Solids* 179, 105363.
- Linka K, Hillgärtner M, Abdolazizi KP, Aydin RC, Itskov M and Cyron CJ** (2021) Constitutive artificial neural networks: A fast and general approach to predictive data-driven constitutive modeling by deep learning. *Journal of Computational Physics* 429, 110010.
- Linka K and Kuhl E** (2023) A new family of constitutive artificial neural networks towards automated model discovery. *Computer Methods in Applied Mechanics and Engineering* 403 A, 115731.
- Linka K, St. Pierre SR and Kuhl E** (2023) Automated model discovery for human brain using constitutive artificial neural networks. *Acta Biomaterialia* 160, 134–151.
- Nakkiran P, Kaplan G, Bansal Y, Yang T, Barak B and Sutskever I** (2021) Deep double descent: Where bigger models and more data hurt. *Journal of Statistical Mechanics: Theory and Experiment* 2021(12), 124003.
- Neff P, Eidel B and Martin RJ** (2016) Geometry of logarithmic strain measures in solid mechanics. *Archive for Rational Mechanics and Analysis* 222, 507–572.
- Neff P, Ghiba I-D and Lankeit J** (2015) The exponentiated Hencky-logarithmic strain energy. Part I: Constitutive issues and rank-one convexity. *Journal of Elasticity* 121, 143–234.
- Ortigosa R, Martínez-Frutos J and Gil AJ** (2023) Programming shape-morphing electroactive polymers through multi-material topology optimisation. *Applied Mathematical Modelling* 118, 346–369.
- Rosenkranz M, Kalina KA, Brummund J and Kästner M** (2023) A comparative study on different neural network architectures to model inelasticity. *International Journal for Numerical Methods in Engineering* 124, 4802–4840.
- Schröder J and Neff P** (2003) Invariant formulation of hyperelastic transverse isotropy based on polyconvex free energy functions. *International Journal of Solids and Structures* 40, 401–445.

- Shojaee M, Valizadeh I, Klein DK, Sharifi P and Weeger O** (2023) Multiscale modeling of functionally graded shell lattice metamaterials for additive manufacturing (pre-print under review). *Engineering with Computers*. DOI: [10.1007/s00366-023-01906-8](https://doi.org/10.1007/s00366-023-01906-8)
- Tac V, Linka K, Sahli-Costabal F, Kuhl E and Tepole AB** (2023) Benchmarking physics-informed frameworks for data-driven hyperelasticity. *Computational Mechanics*.
- Tac V, Sahli-Costabal F and Tepole AB** (2022) Data-driven tissue mechanics with polyconvex neural ordinary differential equations. *Computer Methods in Applied Mechanics and Engineering* 398, 115248.
- Valizadeh I, Al Aboud A, Dörsam E and Weeger O** (2021) Tailoring of functionally graded hyperelastic materials via grayscale mask stereolithography 3D printing. *Additive Manufacturing* 47, 102108.
- Valizadeh I and Weeger O** (2022) Parametric visco-hyperelastic constitutive modeling of functionally graded 3D printed polymers. *International Journal of Mechanical Sciences* 226, 107335.
- Vlassis NN and Sun WC** (2021) Sobolev training of thermodynamic-informed neural networks for interpretable elasto-plasticity models with level set hardening. *Computer Methods in Applied Mechanics and Engineering* 377, 113695.
- Wu J, Zhao Z, Hamel CM, Mu X, Kuang X, Guo Z and Qi HJ** (2018) Evolution of material properties during free radical photopolymerization. *Journal of the Mechanics and Physics of Solids* 112, 25–49.
- Yang Gao D, Neff P, Roventa I and Thiel C** (2017) On the convexity of nonlinear elastic energies in the right Cauchy-green tensor. *Journal of Elasticity* 127, 303–308.
- Zee L and Sternberg ER** (1983) Ordinary and strong ellipticity in the equilibrium theory of incompressible hyperelastic solids. *Archive for Rational Mechanics and Analysis* 83, 53–90.