DOI: 10.21857/mnlqgcpq4y

Pregledni članak / Review paper
Prihvaćeno / Accepted: 15. svibnja 2025.

ON PHYSICS INFORMED NEURAL NETWORKS – IMPOSING CONSTRAINTS IN INFINITESIMAL STRAIN CONTINUUM MECHANICS

Marko Čanađija, Martin Zlatić

Summary

The paper presents methods for implementing physical constraints from continuum mechanics in neural networks that serve as constitutive models. The focus is on infinitesimal strains in nonlinear elasticity. Basic constraints are presented with a brief overview of continuum mechanics, and then implemented in feed-forward neural networks. Among others, the topics of convexity, objectivity, thermodynamic consistency and normalization are addressed. Where appropriate, several possible techniques have been presented. Performance is demonstrated by developing a constitutive model for a single-walled carbon nanotube using physics informed neural networks.

Keywords: physics informed neural networks; physical constraints; continuum mechanics; convexity.

1. INTRODUCTION

Neural networks, one of the most widely used machine learning techniques, have been a topic of research since the first publication by neurophysiologist Warren McCulloch and mathematician Walter Pitts in 1943 [1]. Evidently, this is a topic with a long history. It took time for computer capacities to become powerful enough to enable the exponential research interest observed in the last decade. The application of machine learning is truly present today in all kinds of research areas, and, what is important for the present research, in mechanics as well.

Even within mechanics, machine learning has found a variety of applications. Probably the most popular is its ability to take on the role of a constitutive model. In short, a neural network is trained based on experimentally obtained or computed data to provide a particular output variable of interest. In this sense, the neural network is a constitutive

model. It can be classified as a data-driven method, albeit not a model-free data-driven one as originally proposed in [2]. The main advantage of a neural-network-based constitutive model over a classical model is that no assumptions about the particular constitutive law are required. To illustrate the above, consider a creep behavior. For such problems, data are usually available for stress, strain, strain rate and temperature. In a classical approach, one should think of a specific mathematical equation, in most cases a polynomial or an exponential equation, and then apply a regression method to determine the model parameters. In contrast, the general nature of the neural network does not require the specification of a particular mathematical model, making it an ideal candidate for a constitutive law. The tangent modulus, which is often required for numerical calculations, can also be easily provided by a neural network.

The initial attempts to train neural networks were based on datasets consisting of stress-strain pairs. The neural network merely learned the data and acted as a kind of table that approximated the values that were not seen during training. It soon became apparent that if one wanted to use experimental data for this purpose, the method was not very attractive due to the large amount of data required. Moreover, simply adhering to the experimental data would violate some of the constraints of continuum mechanics, although this violation is usually small. This hinders an efficient application in finite element calculations [3, 4]. For this reason, parts of physics can be built into the neural network, and such networks have been named physics informed neural networks, or simply PINNs. These will be carefully examined in the present research.

PINNs, initially introduced in [5], have nowadays found application in mechanics for a variety of problems. Nevertheless, in a very simplistic sense, these applications can be divided into three main categories. The first group of PINNs is used as a surrogate for a constitutive model. In the simplest case, NN takes the strain tensor as input, while the output is a stress tensor. Using the NN autodifferentation capability, a tangent modulus can be easily determined. PINNs are then obtained by including physical constraints. For example – objectivity by means of using invariants of the right Cauchy–Green deformation tensor as input and energy as output [6-10]. This can drastically reduce the required dataset size compared to NNs that use stress-strain pairs for training. Another consequence of the energy being the output is that training is performed on the differentiated NN, which automatically preserves an additional physical constraint – thermodynamic consistency. The polyconvexity condition is another constraint that can be included [11]; this can be achieved by resorting to the input convex neural networks (ICNN), as introduced by [12]. In such NNs, the output becomes a convex function of all inputs. Polyconvexity is enforced in the deformation gradient, its adjugate and its determinant. To ensure thermodynamic consistency, NN representing the energy is

then differentiated and trained with stress data. The loss function is then constructed using the trained and provided stresses. However, polyconvexity may be too restrictive for some problems [6]; hence, finding another way to ensure ellipticity remains an open question.

The second category builds on the above mentioned automatic differentiation capability of PINNs. This category was originally proposed in [5] and is usually associated with the term PINN. In the present article, we use the term PINN in the broader context, that is for a NN that contains any kind of physical principle. In the first group, PINN represents the energy; in the structural analysis however, PINN can provide displacements as an output. If suitable activation functions are used, PINNs can be differentiated as many times as necessary. It is well known from the beam theory that the fourth derivative corresponds to the distributed transverse load on the beam. Thus, PINN represents a partial differential equation of higher order governing mechanical behavior of beams [13, 14]. Similar considerations can be extended to plates. However, the terms of PDEs are often of multiple order, which leads to a significant loss of accuracy in the calculation of these terms. As a result, PINNs are very sensitive, leading to unstable convergence due to significant fluctuations in the weights. The recent contribution [15] solves the problem through multi-level PINNs. These are based on the separation of PINNs into multiple NNs, each of which contains only the first- or the second-order PDEs. In the end, this stabilizes the framework. So far, this approach has only been verified for the simplest problems and still needs to be verified for applications that are more complex.

The third group of PINNs used in mechanical problems are represented by NNs that can solve for complete fields. For example, [16] presents Physics-Informed Temporal Convolutional Networks (PI-TCN), a convolutional neural network (CNN) consisting of one-dimensional convolutional layers. These are integrated into a finite element NN framework for thermoelastic problems. A dataset obtained by FEA is used to train PI-TCN to learn mapping from time, coordinates and strain rates to provide a temperature field as output. The calculations are performed for a transient problem with heat conduction. In this way, NN is used to predict the temperature field, while the displacement field is calculated in the conventional way, i.e. by FEA. In this way, the time required for the calculation is greatly reduced, but still, only a part of the problem is treated.

The imposition of constraints in PINNs has recently been investigated in a number of papers [17-27]. Without going into the details of these researches, it can be summarized that they all deal with the framework of finite strains, and in most cases treat hyperelastic constitutive behavior. Analyses related to infinitesimal strains are not frequently performed. Thus, a systematic overview of the techniques presented in the cited finite strain papers – in the context of infinitesimal strains – is missing in the literature.

To this end, the present study provides a guide to enforcing constraints stemming from continuum mechanics in feed-forward neural networks. The existing literature on constitutive modeling typically deals with hyperelastic behavior with finite strain. In contrast to these results, the present study focuses on infinitesimal strains and isotropic elastic materials, which, however, are not necessarily linear. First, a brief overview of the basics of continuum mechanics and the associated constraints is given. Next, several possible techniques to incorporate these into neural networks are presented. Finally, a typical application including the development of a constitutive model of a single-walled carbon nanotube is explained using an example.

2. A BRIEF OVERVIEW OF CONTINUUM MECHANICS

2.1 Balance equations

To demonstrate the fundamental physical elements that should be included in neural networks, a very brief overview of continuum mechanics is provided for the sake of completeness [23, 24]. The balance laws are presented first. The balance of mass in its final, local form states that the rate of change of the mass density $\dot{\rho}(\boldsymbol{x}, t)$ at a spatial point $\boldsymbol{x}$ and at time t is:

$$\dot{\rho}(\mathbf{x}, t) = -\rho(\boldsymbol{x}, t)\mathrm{div}\boldsymbol{v}(\boldsymbol{x}, t), \tag{1}$$

where $\boldsymbol{v}$ is the velocity vector. In short, the law states that mass cannot be destroyed during a thermodynamic process, and that it ought to be conserved.

The balance of the linear momentum, again given in the local form is:

$$\mathrm{div}\boldsymbol{\sigma} + \rho\boldsymbol{b} = \rho\dot{\boldsymbol{v}}, \tag{2}$$

which is also known as Cauchy's first equation of motion. Above, $\boldsymbol{\sigma}$ is the Cauchy stress tensor and $\mathbf{b}$ are the volumetric forces. The inertial forces are often neglected, which leads to the Cauchy equilibrium equations: $\mathrm{div}\boldsymbol{\sigma} + \rho\boldsymbol{b} = 0$

Balance of the angular momentum results in the well-known symmetry of the Cauchy stress tensor:

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}^T. \tag{3}$$

This law is known as the second Cauchy equation of motion.

The balance of energy equation is:

$$\rho\dot{e} - \boldsymbol{\sigma}\colon \mathbf{d} - \rho r + \mathrm{div}\mathbf{q} = 0. \tag{4}$$

Above, $\dot{e}$ is the rate of internal specific energy per unit mass, r is the specific heat supply per unit mass, $\boldsymbol{d}$ is the symmetric part of the velocity gradient, and $\boldsymbol{q}$ is the heat flux vector. The symmetrical part of the velocity gradient $\boldsymbol{d}$ for the present infinitesimal case is equal to

$$\boldsymbol{d} = \dot{\boldsymbol{\epsilon}}, \tag{5}$$

where $\dot{\boldsymbol{\epsilon}}$ is the rate of the infinitesimal strain tensor $\boldsymbol{\epsilon}$. For a displacement vector $\boldsymbol{u}$ with components $\{u_x, u_y, u_z\}^T$, $\boldsymbol{\epsilon}$ is calculated as:

$$\begin{gathered} \epsilon_x = \frac{\partial u_x}{\partial x}, \quad \epsilon_y = \frac{\partial u_y}{\partial y}, \quad \epsilon_z = \frac{\partial u_z}{\partial z}, \\ \epsilon_{xy} = \frac{1}{2}\left(\frac{\partial u_y}{\partial x} + \frac{\partial u_x}{\partial y}\right), \epsilon_{yz} = \frac{1}{2}\left(\frac{\partial u_z}{\partial y} + \frac{\partial u_y}{\partial z}\right), \epsilon_{xz} = \frac{1}{2}\left(\frac{\partial u_x}{\partial z} + \frac{\partial u_z}{\partial x}\right). \end{gathered} \tag{6}$$

Finally, the second law of thermodynamics can be transformed into the following form:

$$\rho\theta\dot{\eta} - \rho r + \mathrm{div}\boldsymbol{q} - \frac{1}{\theta}\boldsymbol{q} \cdot \mathrm{grad}\theta \geq 0, \tag{7}$$

where Θ is the absolute temperature, and $\dot{\eta}$ is the rate of specific entropy per unit mass. The main role of the second law of thermodynamics is to place some constraints on the allowable thermodynamic processes. In particular, the part of the mechanical energy that is dissipated in the form of heat (for example through friction) cannot be reused to perform mechanical work. Another constraint restricts the spontaneous flow of heat from a colder part of a body to a warmer part. A reduced form of the second law of thermodynamics, known as the Clausius-Planck law, is of interest for this paper:

$$\boldsymbol{\sigma}\colon \boldsymbol{d} - \rho\dot{e} + \rho\theta\dot{\eta} \geq 0. \tag{8}$$

2.2 Constitutive behavior

The fundamental laws of continuum mechanics defined above still contain no information about the particular material, i.e. about the constitutive behavior. Although this can be done by defining a particular form of internal specific energy per unit mass e or even enthalpy or Gibbs free energy, it is usually done by the Helmholtz free energy $\psi(\boldsymbol{\epsilon}, \theta, \boldsymbol{\alpha})$[25]. In addition to strain and temperature, the set of internal variables $\boldsymbol{\alpha}$ represents additional variables that are used to describe the current state of the material. Internal variables can be scalars or tensors, depending on the material. A typical example is plasticity, where a scalar is used to define the isotropic strain hardening, and a second order tensor is used to define the kinematic strain hardening. The role of Helmholtz free energy can be understood when the connection between the internal energy and the free energy is defined thus:

$$e(\boldsymbol{\epsilon}, \eta, \boldsymbol{\alpha}) = \psi(\boldsymbol{\epsilon}, \theta, \boldsymbol{\alpha}) + \theta\eta. \tag{9}$$

The above equation states that the Helmholtz free energy expressed per unit mass is obtained from the internal energy by subtracting the irreversible part of the internal energy $\theta\eta$. The Helmholtz free energy is therefore a part of the energy that can be used for mechanical work at constant temperature. As already mentioned, the Helmholtz free energy is the standard method for defining the constitutive behavior of a material. Its advantage over internal energy is that it is a function of absolute temperature, in contrast to the entropy in the case of internal energy, which is not easily measurable.

2.3 Continuum mechanics constraints

2.3.1 Thermodynamic consistency

The specific format of the Helmholtz free energy $\psi(\boldsymbol{\epsilon}, \theta, \boldsymbol{\alpha})$ can be combined with the balance of energy and the second law of thermodynamics [24] in order to obtain the following inequality:

$$\rho\left(-\partial_{\boldsymbol{\epsilon}}\Psi : \dot{\boldsymbol{\epsilon}} - \partial_{\theta}\psi\dot{\theta} - \partial_{\boldsymbol{\alpha}}\psi : \dot{\boldsymbol{\alpha}}\right) - \rho\eta\dot{\theta} + \boldsymbol{\sigma} : \dot{\boldsymbol{\epsilon}} - \frac{1}{\theta}\boldsymbol{Q} \cdot \mathrm{grad}\theta \geq 0. \tag{10}$$

With the last inequality at hand, consider a purely thermal process for a start. Therefore, the deformation will be fixed. Additionally, assume a homogeneous temperature field and consider such a material behavior that requires internal variables $\boldsymbol{\alpha}$. Eq. (10) is then reduced:

$$(-\partial_\theta \psi - \eta)\dot{\theta} \geq 0. \tag{11}$$

This equation ought to be valid for every possible temperature rate $\dot{\theta}$, i.e. negative or positive, so the term in parentheses has to vanish providing the constitutive equation for the specific entropy:

$$\eta = -\partial_\theta \psi. \tag{12}$$

Second, consider now a perfect elastic material, again without the internal variables $\boldsymbol{\alpha}$, and where the temperature field is fixed and homogeneous. Eq. (10) now gives:

$$(\boldsymbol{\sigma} - \rho\, \partial_{\boldsymbol{\epsilon}} \psi) : \dot{\boldsymbol{\epsilon}} \geq 0. \tag{13}$$

Accounting for the arbitrariness of the strain rate in the same way as for the entropy above, the solution defines the constitutive law for the Cauchy stress tensor:

$$\boldsymbol{\sigma} = \rho\, \partial_{\boldsymbol{\epsilon}} \psi. \tag{14}$$

To conclude, the relationship between stress and strain tensors is not defined directly, but by the derivative of Helmholtz free energy with respect to strain. Since the balance of energy and the second law of thermodynamics are used in this derivation, they are built into this definition, and it can be said that the definition of stresses by Eq. (14) is thermodynamically consistent.

After introducing the concept of thermodynamic consistency, the remainder of the text will focus on the isothermal problems.

2.3.2 Material frame indifference and small strains

The material frame indifference is based on the concept of objectivity. To understand the concept of objectivity, one can imagine a point in space (cf. [26] sec. 17). In a reference frame defined by the origin $\boldsymbol{o}$ and the base vectors $\boldsymbol{e}_i$, the position vector of a point is denoted by $\boldsymbol{x}=x_i\boldsymbol{e}_i$. Now consider another reference frame with a different origin $\hat{\boldsymbol{o}}$ and the base vectors $\hat{\boldsymbol{e}}_i$. The position vector of the same point in the new reference frame $\hat{\boldsymbol{x}}$ will be different. Observers in these two reference frames will therefore see the position vectors differently, i.e. as $\boldsymbol{x}$ or $\hat{\boldsymbol{x}}$. These two vectors can be connected as follows:

$$\hat{\boldsymbol{x}} = \boldsymbol{u}(t) + \boldsymbol{Q}(t)\boldsymbol{x}\,, \tag{15}$$

where ***Q(t)*** is an orthogonal tensor that describes the rotation, while ***u(t)*** is the vector that connects two origins and can be understood as a translation. This relationship also includes the concept of an event, so that the observations occur in two reference frames at two different points in time, i.e. ($\boldsymbol{x}$,t) and ($\hat{\boldsymbol{x}}$, $\hat{t}$), where the points in time are related as $\hat{t} = t - a$, and a is the time shift. It should be emphasized once again that the two vectors $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$ mentioned above are two different vectors that are obtained by the transformation Eq. (15). This is fundamentally different from the transformation of a tensor quantity from one coordinate systems to another. In the latter transformation, the vector remains the same, while due to the change of the basis, only the vector components change.

Objects can be transformed from one coordinate frame to another. These transformations depend on the type of tensor:

- scalar quantities ought to remain constant, $\hat{\alpha} = \alpha$
- for a vector $\hat{\boldsymbol{x}} = \boldsymbol{Q}(t)\boldsymbol{x}$
- for a second-order tensor $\hat{\boldsymbol{T}} = \boldsymbol{Q}(t)\boldsymbol{T}\boldsymbol{Q}^T(t)$
- for a two-point tensor (for example deformation gradient) $\hat{\boldsymbol{F}} = \boldsymbol{Q}(t)\boldsymbol{F}$.

The above is now used for the following definition – an objective or a frame indifferent function will be invariant when the variables are transformed in the above manner, cf. [26] sec. 19.

Recall now that in Sec. 2.2, it was mentioned that the constitutive behavior of a material is usually defined by the Helmholtz free energy function $\psi(\boldsymbol{\epsilon}, \theta)$. The principle of the material frame indifference (or the principle of material objectivity) states that the constitutive behavior cannot depend on the choice of a reference frame. This simply means that the constitutive behavior defined by the infinitesimal strain tensors expressed in two different reference frames ought to give the same Helmholtz free energy. According to the above, the infinitesimal strain tensor should be transformed as an objective quantity.

First, the finite strain case is analyzed. Consider the objectivity of the Green-Lagrange strain tensor $\boldsymbol{E}$

$$\hat{\boldsymbol{E}} = \frac{1}{2}\left(\hat{\boldsymbol{F}}^T\hat{\boldsymbol{F}} - \boldsymbol{I}\right) = \frac{1}{2}(\boldsymbol{F}^T\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{F} - \boldsymbol{I}) = \frac{1}{2}(\boldsymbol{F}^T\boldsymbol{F} - \boldsymbol{I}) = \boldsymbol{E}, \tag{16}$$

where the $\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}$ orthogonality and the objective transformation of a two-point tensor were used. Thus, $\boldsymbol{E}$ is an objective quantity.

Now, the geometrical linear theory assumes that the Green-Lagrange strain tensor $\boldsymbol{E}$:

$$\boldsymbol{E} = \frac{1}{2}(\boldsymbol{F}^T\boldsymbol{F} - \boldsymbol{I}) = \frac{1}{2}(\nabla\boldsymbol{u} + \nabla\boldsymbol{u}^T + \nabla\boldsymbol{u}\nabla\boldsymbol{u}^T), \tag{17}$$

where $\boldsymbol{F} = \boldsymbol{I} + \nabla\boldsymbol{u}$ can be reduced to the infinitesimal strain tensor by neglecting the product of the gradients term $\nabla\boldsymbol{u}\ \nabla\boldsymbol{u}^T \approx 0$:

$$\boldsymbol{\epsilon} = \frac{1}{2}(\nabla\boldsymbol{u} + \nabla\boldsymbol{u}^T) = \frac{1}{2}(\boldsymbol{F} + \boldsymbol{F}^T) - \boldsymbol{I}, \tag{18}$$

i.e. displacement gradients $\nabla\boldsymbol{u}$ are so small that their products can be neglected in comparison to the gradient itself.

However, it is easy to see that the above definitions of the objectivity of second-order tensors are not valid for the infinitesimal strain tensor in Eq. (18). On the other hand, the assumption of small displacement gradients means that the rotations ought also to be small. As an illustration, consider a planar rigid body rotation:

$$x_1 = X_1\cos\alpha - X_2\sin\ \alpha\ , \quad x_2 = X_1\sin\alpha + X_2\cos\alpha. \tag{19}$$

The deformation gradient will be

$$\boldsymbol{F} = \begin{bmatrix} \cos\ \alpha & -\sin\ \alpha & 0 \\ \sin\ \alpha & \cos\ \alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{20}$$

Eq. (18) then gives the infinitesimal strain tensor as:

$$\boldsymbol{\epsilon} = \begin{bmatrix} \cos\ \alpha - 1 & 0 & 0 \\ 0 & \cos\ \alpha - 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{21}$$

The rigid body rotation should not introduce any strain, but the last equation gives vanishing strains only for $\alpha = 0$. Any angle of rotation that is not zero leads to errors, which can become quite large. Therefore, the infinitesimal strain tensor is not objective.

In conclusion, to ensure that the principle of material frame indifference is valid for small-strain setting, the rotations ought to be inherently small by assumption. In this way, objectivity is enforced by default. On the other hand, in the case of finite strains, objectivity requires a much more elaborate analysis, especially in the presence of other constraints, such as convexity [27, 28]. In this case, the convexity condition is usually too strict and relaxed to the polyconvexity constraint. It should be noted that to ensure material stability, the ellipticity condition ought to be fulfilled (cf. [26] sec. 68b and [27]), however it is rather difficult to do so. It is simpler to implement convexity, quasi-convexity and polyconvexity when defining material behavior. It has been done so for decades.

2.3.3 Convexity

The elaborations will now focus on the convexity of energy. First, recall that a function *f(x),* where $\boldsymbol{x} \in R^n$, $f \in R$, such that $f: R^n \rightarrow R$ is globally convex if any of the following conditions hold [29]:

$$\begin{gathered} f(\xi \boldsymbol{x} + (1-\xi)\boldsymbol{y}) \leq \xi f(\boldsymbol{x}) + (1-\xi) f(\boldsymbol{y}), \quad \forall \xi \in [0,1] \\ f(\boldsymbol{y}) \geq f(\boldsymbol{x}) \nabla f(\boldsymbol{x})^T (\boldsymbol{y} - \boldsymbol{x}), \quad \forall \boldsymbol{x}, \boldsymbol{y} \\ \nabla^2 f(\boldsymbol{x}) \succcurlyeq 0, \quad \forall \boldsymbol{x}. \end{gathered} \tag{22}$$

The last two conditions require that the function is first- and second-order differentiable, respectively. Furthermore, if the function $f(\boldsymbol{x})$ is convex, the function $-f(\boldsymbol{x})$ is concave.

The first condition in Eqs (22) is interpreted thus: the line that intersects the curve (right-hand side) lies above the curve (left-hand side) between these two intersection points. In other words, a linear interpolation between two points will overestimate the function. The second condition states that the linear approximation involving the first derivative underestimates the function. The third condition requires that the Hessian $\nabla^2 f(\boldsymbol{x})$ is positive-semidefinite, which means that the curvature of the function is always non-negative. For a single variable function, the condition collapses to the requirement that the second derivative is positive. Finally, a convenient consequence of global convexity is that the function also has a single global minimum. If the above considerations are only valid for a part of the domain $A \subset R^n$, the function is locally convex and has a single local minimum in A.

Some properties of a convex function useful for the present elaborations are summarized below:

1. If $f(\boldsymbol{x})$ is a convex function, and $\xi \geq 0$, the product ξf is a convex function too.
2. If $f(\boldsymbol{x})$ and $g(\boldsymbol{x})$ are convex functions, the sum $f(\boldsymbol{x}) + g(\boldsymbol{x})$ is a convex function too.
3. The combination of the above provides that for non-negative weights $\xi_i \geq 0$, the sum of the weighted convex functions f_i is convex too, i.e. $f = \sum_i^N \xi_i f_i$. This also includes integrals.
4. Linear transformation $g(\boldsymbol{x}) = f(\boldsymbol{Ax} + \boldsymbol{b})$ preserves convexity, i.e. both $f(\boldsymbol{x})$ and $g(\boldsymbol{x})$ are convex functions.
5. A composition of functions $f = h \circ g$, that is $f(\boldsymbol{x})=h(\boldsymbol{g}(\boldsymbol{x}))=h(g_1(\boldsymbol{x}),\ldots,g_N(\boldsymbol{x}))$, will be convex if h is convex and non-decreasing in each argument, and g_i are convex (or h is convex and non-increasing in each argument, and g_i are concave). Both g_i and h should be twice differentiable.

The concept of convexity is now applied in continuum mechanics. For the sake of simplicity, the dependence of the free energy on the internal variables $\boldsymbol{\alpha}$ is not considered, i.e. the nonlinear thermoelastic behavior $\psi(\boldsymbol{\epsilon},\theta)$ is analyzed. Let us now consider a one-dimensional deformation to illustrate this. The non-negativity of the first derivative in Eq. (14), i.e. $\sigma=\rho\partial_{\epsilon}\psi\geq 0$ means, for example, that the energy increases with the increase in strain. The non-negativity of the second derivative $\rho\partial_{\epsilon}^2\psi=\partial_{\epsilon}\sigma\geq 0$ means that the stress ought to increase when the strain is increased. Analogous conclusions can be drawn in connection with several variables $\psi(\boldsymbol{\epsilon},\theta)$ for the positive semi-definiteness of the second derivative $\rho\partial_{\epsilon}^2\psi=\partial_{\epsilon}\boldsymbol{\sigma}$. This observation leads to the conclusion that the energy should be a convex function of the strain tensor $\boldsymbol{\epsilon}$.

When the definition of the entropy is concerned, due to the existence of a negative sign in front of the derivative Eq. (12), the same argumentation as above will lead toward the conclusion that the free energy ψ is concave with respect to the temperature θ, or that the negative of the free energy $-\psi$ is convex with respect thereto.

As an example, consider the following form of the Helmholtz free energy used in the isotropic coupled infinitesimal thermoelasticity:

$$\rho\psi(\boldsymbol{\epsilon},\theta) = \frac{1}{2}\mathcal{C}_{ijk\ell}\epsilon_{ij}\epsilon_{kl} - 3\kappa\alpha\epsilon_{ij}\delta_{ij}(\theta-\theta_0) - \rho\frac{1}{2\theta_0}c_0(\theta-\theta_0)^2, \tag{23}$$

where $\boldsymbol{\kappa}$ is the bulk modulus, α is the linear coefficient of thermal expansion, θ_0 is the reference temperature, and $c_0=c_v$ is is the specific heat capacity at constant strain. For the ease of interpretation, the above free energy can be written in the Voigt notation as:

$$\rho\psi(\boldsymbol{\epsilon},\theta) = \frac{1}{2}\boldsymbol{\epsilon}^{\mathrm{T}}\boldsymbol{C}\boldsymbol{\epsilon} - \kappa\alpha_v(\theta-\theta_0)\mathrm{tr}\boldsymbol{\epsilon} - \rho\frac{1}{2\theta_0}c_0(\theta-\theta_0)^2. \tag{24}$$

Differentiation with respect to strain provides the following result, again written in Voigt notation:

$$\begin{aligned}\boldsymbol{\sigma} &= \rho\,\partial_{\boldsymbol{\epsilon}}\psi = \boldsymbol{C}\boldsymbol{\epsilon}\\ \partial_{\boldsymbol{\epsilon}}\boldsymbol{\sigma} &= \rho(\partial^2_{\boldsymbol{\epsilon}}\psi) = \boldsymbol{C}.\end{aligned} \tag{25}$$

The convexity condition Eq. (22) implies that the elasticity tensor $\boldsymbol{C}$ (or tangent modulus) has to be positive semi-definite. In the case of isotropic linear elasticity, this tensor has many zeros, and the enforcement of each zero can be another constraint, see [20].

In a similar way, the second derivative with respect to temperature provides the specific heat capacity as:

$$\partial^2_{\theta\theta}\psi(\boldsymbol{\epsilon},\theta) = -\frac{1}{\theta_0}c_0. \tag{26}$$

The concavity requirement now implies that the specific heat capacity c_0 ought to be positive.

Thus, while this concept is quite appropriate for infinitesimal thermoelasticity, the requirement for convexity may be too restrictive in general. For instance, there are several minima in ferroelasticity, which contradicts the convexity hypothesis [31]. There are also several minima in buckling problems, again violating the convexity framework.

In the context of finite strains, the issue of convexity is complicated by the requirement of the material frame indifference, as already explained in Sec. 2.3.2. In particular, it is shown in [32] that enforcing the convexity of the energy by Eq. (22) violates the objectivity principle in the presence of compressive stresses [26, 27]. The concept of quasiconvexity, as introduced in [33], circumvents the constraint on existing of a single minimum by allowing other (local) minima, which, however, have a higher energy value. As discussed in [27, 34], this approach includes other problems as well. It seems that the more general approach is polyconvexity as introduced in [27], which has weaker growth conditions than quasiconvexity. In the context of this approach, the problem is redefined by using invariants as variables instead of a strain tensor or deformation gradient, see example [6]. While neo-Hookean materials do not fall into this framework, the Mooney-Rivlin and Ogden materials do [27], as does anisotropic hyperelasticity [35].

2.3.4 Other constraints

Normalization of stresses and energy

Remaining physical constraints are introduced in this section. The fact that the stress and the energy have to vanish at the undeformed state, i.e.

$$\begin{aligned} \boldsymbol{\sigma}(\boldsymbol{\epsilon} = \mathbf{0}) &= \mathbf{0} \\ \psi(\boldsymbol{\epsilon} = \mathbf{0}, \theta) &= 0 \end{aligned} \tag{27}$$

is known as the normalization of stress/energy constraint.

Non-negativity of the energy

The above normalization constraints are usually accompanied by the non-negativity of the energy condition:

$$\psi(\boldsymbol{\epsilon}, \theta) \geq 0. \tag{28}$$

Combining conditions Eqs (27) & (28) gives a conclusion that the minimum of the energy ought to be at the undeformed state $\boldsymbol{\epsilon} = \mathbf{0}$.

Volumetric growth condition

The volumetric growth condition or coercivity condition should reflect the fact that compressing a volume to a point requires infinite energy. In particular, for the infinitesimal strains:

$$\begin{aligned} \psi &\longrightarrow \infty \quad \text{as} \quad \epsilon_v \rightarrow -1 \\ \psi &\longrightarrow \infty \quad \text{as} \quad \epsilon_v \rightarrow \infty. \end{aligned} \tag{29}$$

The second condition reflects the analogous behavior in tension.

The reader is reminded that in the case where a volume is compressed to a point, the volumetric strain is $\epsilon_v = \Delta V/V_0 = -1$. However, since in the infinitesimal strain context, the volumetric strain is excluded to grow to infinity anyway, this condition is substituted by the convexity requirement, and the above conditions are not relevant. If, for some reason, one still wants to enforce these, it ought to resort to the true (logarithmic) strain, where the volumetric strain can grow to negative infinity.

Symmetry of the stress tensor

The symmetry of the stress tensor follows from Eq. (3). This also requires that the tangent modulus defined in Eq. $(25)_2$ has major symmetry property. This is written in the abbreviated Voigt notation as a condition that its components are related as $C_{ij}=C_{ji}$.

Other constraints could include the incompressibility constraint, which is particularly important in the finite strain hyperelasticity of rubbers [6]; it will be not addressed here. Similarly, the material orthotropy aspect and alike will also not be addressed.

3. IMPLEMENTATIONS OF THE PHYSICAL CONSTRAINTS INTO NEURAL NETWORKS

3.1 Underlying principles of simple feed-forward neural networks

To understand the concept of incorporating the physical constraints described above into a neutral network, the basics of a feed-forward neural network are briefly explained.

Assume that a dataset D compromised out of N points is available:

$$D = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) \in R^p \times R^q, |i = 1, .., N\}, \tag{30}$$

where $=\dim(\boldsymbol{x}_i)$, $q=\dim(\boldsymbol{y}_i)$. The task is to determine function f:

$$\boldsymbol{y}_i = f(\boldsymbol{x}_i), \forall i \in D. \tag{31}$$

For example, assume p=2 and q=1. Let us introduce two weight matrices $\boldsymbol{W}^{(1)}$ with dimensions (r,p), and $\boldsymbol{W}^{(2)}$ with dimensions (q,r). The dimension r = 3 will be used for vector $\boldsymbol{h}^{(1)}$– for illustration purposes. The input and the output can be related through simple matrix multiplication:

$$\begin{aligned} \boldsymbol{h}^{(1)} &= \left(\boldsymbol{W}^{(1)}\right)^T \boldsymbol{x} \\ \hat{\boldsymbol{y}} &= \left(\boldsymbol{W}^{(2)}\right)^T \boldsymbol{h}^{(1)}. \end{aligned} \tag{32}$$

A slightly different notation for the output is introduced to differentiate the result of the above function $\hat{y}$ from the $\boldsymbol{y}$ values in the dataset D. This series of multiplications can clearly be substituted in the following manner:

$$\hat{\boldsymbol{y}} = \left(\boldsymbol{W}^{(2)}\right)^T \left(\boldsymbol{W}^{(1)}\right)^T \boldsymbol{x}, \tag{33}$$

or with $\left(\boldsymbol{W}^{(0)}\right)^T = \left(\boldsymbol{W}^{(2)}\right)^T \left(\boldsymbol{W}^{(1)}\right)^T$

$$\hat{\boldsymbol{y}} = \left(\boldsymbol{W}^{(0)}\right)^T \boldsymbol{x}. \tag{34}$$

The last result is, of course, a linear relationship between the input and the output through the weight matrix $(\boldsymbol{W}^{(0)})^T$. Due to its linear nature, the above result is of limited use in mechanics. Note that the coefficients of the weight matrices are unknown at this point, and should be determined in some manner. This issue will be dealt with later.

This operation can be represented schematically, Fig. 1. In this representation, the vectors $\boldsymbol{x}, \boldsymbol{h}^{(1)}$ and $\hat{y}$ are referred to as the input, hidden and output layers, while the coefficients $w_{ij}^{(k)}$ are referred to as weights. The vector components are called neurons, and the whole scheme can be thought of as resembling a neural network as part of the nervous system of humans or animals.

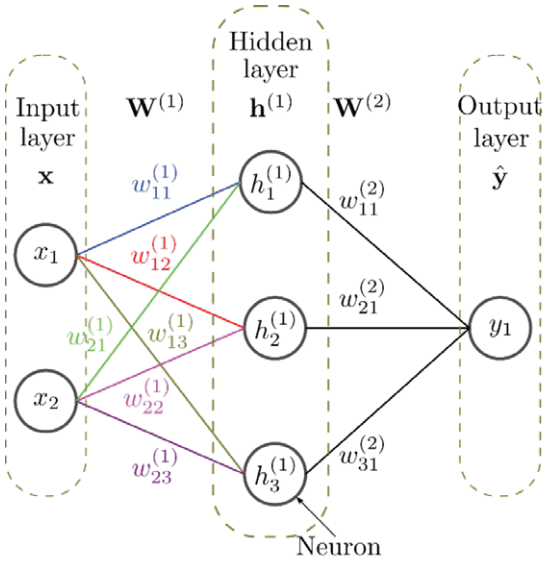

Figure 1. A feed-forward neural network.
Slika 1. Unaprijedna neuronska mreža.

Wider applicability of the approach requires introduction of nonlinearity. Eq. (32) can be modified in the following way:

$$\begin{aligned} \boldsymbol{h}^{(1)} &= g^{(1)}\left(\left(\boldsymbol{W}^{(1)}\right)^T \boldsymbol{x} + \boldsymbol{b}^{(1)}\right) \\ \hat{\boldsymbol{y}} &= g^{(2)}\left(\left(\boldsymbol{W}^{(2)}\right)^T \boldsymbol{h}^{(1)} + \boldsymbol{b}^{(2)}\right), \end{aligned} \tag{35}$$

where $\boldsymbol{b}^{(k)}$ are bias vectors and $g^{(k)}$ is a nonlinear function taking $\boldsymbol{W}^T \boldsymbol{x} + \boldsymbol{b}$ as argument. This function is called the activation function. One of the simplest activation functions is the ReLU function $g(\xi)=\max(0,\xi)$, which returns zero for all negative inputs, Fig. 2. If each element of the vector representing the hidden layer is imagined as a neuron, the activation function determines whether the information is passed on (in the case of positive inputs to the neuron) or not (in the case of negative inputs to the neuron). Another activation function is softplus function:

$$\begin{aligned} g(x) &= \log(\exp(x) + 1) \\ g'(x) &= \frac{1}{1+\exp(x)} \\ g''(x) &= \frac{\exp(x)}{(1+\exp(x))^2} = g'(x)\big(1 - g'(x)\big). \end{aligned} \tag{36}$$

Softplus can also be seen as a smooth approximation of the ReLU activation function (Fig. 2). Unlike the ReLU function, it is twice differentiable and integrable. The first derivative is the logistic or sigmoid function, which is also an activation function, and a desirable property in problems of this type. There are also a number of other standard activation functions [36], but custom activation functions are commonly used as well.

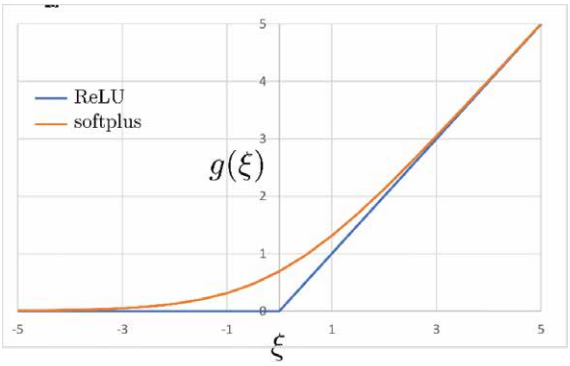

Figure 2. ReLU and softplus activation functions.
Slika 2. ReLU i softplus aktivacijska funkcija.

Having in mind the above, the function $\hat{\boldsymbol{y}}=\hat{\boldsymbol{y}}(\boldsymbol{x};\boldsymbol{W},\boldsymbol{b})$ represents a neural network, where $\boldsymbol{W}=\{\boldsymbol{W}^{(1)},\boldsymbol{W}^{(2)}\}$ and $\boldsymbol{b}=\{\boldsymbol{b}^{(1)},\boldsymbol{b}^{(2)}\}$ are the parameters of the neural network. Consequently, a feed-forward neural network is nothing more than a composition of functions, including activation functions that rely on matrix/vector multiplication and vector summation.

The only thing needed to be determined are the values of weights and biases. These follow from the minimization of the cost function:

$$J_{MSE}(\boldsymbol{W},\boldsymbol{b};\boldsymbol{x},\boldsymbol{y}) = J_{MSE}(\hat{\boldsymbol{y}},\boldsymbol{y}) = \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{y}_i - \hat{\boldsymbol{y}}_i)^2. \tag{37}$$

The above function is called the mean squared error function. It represents a measure how different the true $\boldsymbol{y}$ dataset outputs are from the prediction of the neural network $\hat{\boldsymbol{y}}$. Other popular error measures are the root mean squared error J_{RMSE}, and the mean absolute error J_{MAE}:

$$\begin{aligned} J_{RMSE} &= \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{y}_i - \hat{\boldsymbol{y}}_i)^2} \\ J_{MAE} &= \frac{1}{N}\sum_{i=1}^{N}|\boldsymbol{y}_i - \hat{\boldsymbol{y}}_i|. \end{aligned} \tag{38}$$

The required parameters follow as the solution of the minimization problem:

$$\boldsymbol{W},\boldsymbol{b} = \arg\min_{\boldsymbol{W}^*,\boldsymbol{b}^*} J_{MSE}\,(\boldsymbol{W}^*,\boldsymbol{b}^*). \tag{39}$$

This optimization requires the gradients of the cost function, with respect to the weights and biases. These follow through the so-called backpropagation procedure based on the simple chain rule. In particular, using Eq. $(35)_2$ provides:

$$\begin{aligned} \partial_{\boldsymbol{W}^{(2)}} J_{MSE} &= \partial_{\hat{\boldsymbol{y}}} J_{MSE}\,\partial_{\boldsymbol{W}^{(2)}}\hat{\boldsymbol{y}} = \partial_{\hat{\boldsymbol{y}}} J_{MSE}\big(g^{(2)}\big)'\,\boldsymbol{h}^{(1)} \\ \partial_{\boldsymbol{W}^{(1)}} J_{MSE} &= \partial_{\hat{\boldsymbol{y}}} J_{MSE}\,\partial_{\boldsymbol{h}^{(1)}}\hat{\boldsymbol{y}}\,\partial_{\boldsymbol{W}^{(1)}}\boldsymbol{h}^{(1)} = \partial_{\hat{\boldsymbol{y}}} J_{MSE}\big(g^{(2)}\big)'\boldsymbol{W}^{(2)}\big(g^{(1)}\big)'\boldsymbol{x}\,. \end{aligned} \tag{40}$$

Since the particular forms of the activation functions $g^{(k)}$ are known in advance, the whole process can be carried out effortlessly. If necessary, the same principles are used to differentiate the output with respect to the input:

$$\partial_{\boldsymbol{x}}\hat{\boldsymbol{y}} = \partial_{\boldsymbol{h}^{(1)}}\hat{\boldsymbol{y}}\,\partial_{\boldsymbol{x}}\boldsymbol{h}^{(1)} = \left(g^{(2)}\right)'\boldsymbol{W}^{(2)}\left(g^{(1)}\right)'\boldsymbol{W}^{(1)}. \tag{41}$$

This is usually referred to as the automatic differentiation of the neural network.

3.2 Partially Input Convex Integrable Neural Networks (PICINNs)

The discussion in Sec. 2.3.3 indicated that the constraints stemming from continuum mechanics may require that a function is convex in some (or all) variables. Since the neural network assumes the role of a constitutive model, such restrictions should also apply to the network itself: The output of the neural network is a convex function of some of the input variables. The details of enforcing thermodynamic consistency, Sec 2.3.1, will be addressed later. At this point, we only emphasize that the network should be integrable and derivable at least up to the second order.

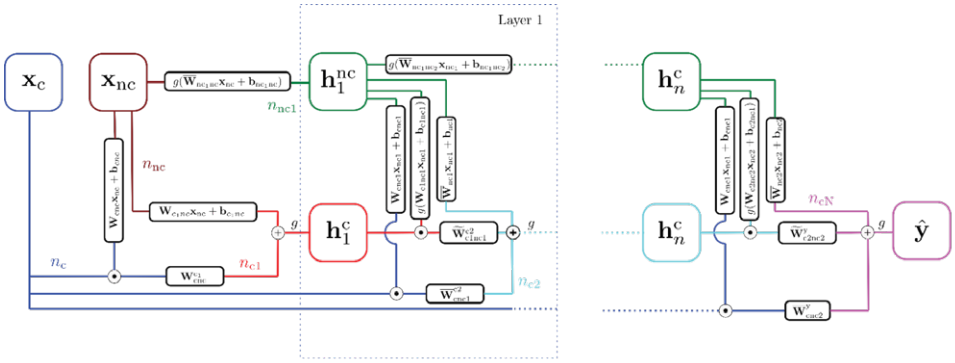

Figure 3. General architecture of PICINN. Colors indicate dimensionality of inputs/outputs. $\overline{\boldsymbol{W}}$ indicates normalized weights in neural network. $\widetilde{\boldsymbol{W}}$ denotes normalized weights with non-negativity constraints.

Slika 3. Općenita arhitektura PICINN. Boje označavaju dimenzionalnost ulaza/izlaza. $\overline{\boldsymbol{W}}$ predstavlja normalizirane težine, dok $\widetilde{\boldsymbol{W}}$ predstavlja težine s ne-negativnim ograničenjima.

At this point, we would like to prove that the neural network in Fig. 3 can be designed so that the output is convex in some input variables, while in the remaining input variables, it is not [4, 8]. The vector of the input variables is $\boldsymbol{x}=\{\boldsymbol{x}_c,\boldsymbol{x}_{nc}\}^T$, whereby the variables in which the output is convex are $\boldsymbol{x}_c$, and where it is not, $\boldsymbol{x}_{nc}$. The output of the network is vector $\hat{\boldsymbol{y}}$. Vectors representing a hidden layer i will be denoted in the similar way as $\boldsymbol{h}_i^c$ and $\boldsymbol{h}_i^{nc}$.

We would like to show that each component of the output vector $\hat{\boldsymbol{y}}_l$ is a convex function, with respect to the components of the input vector $\boldsymbol{x}_c$. The starting point is that the composition of the functions defining the neural network is manifested by the relationship between the hidden layer i and the next hidden layer $j=i+1$, compactly written in vectorial form:

$$\begin{aligned}\boldsymbol{h}_j^c &= \mathrm{g}\Big\{\widetilde{\boldsymbol{W}}_{c_i nc_i}^{c_j}\Big[\boldsymbol{h}_i^c \odot g\left(\boldsymbol{W}_{c_i nc_i}\boldsymbol{h}_i^{nc} + \boldsymbol{b}_{c_i nc_i}\right)\Big] \\ &+\overline{\boldsymbol{W}}_{c_0 nc_i}^{c_j}\big[\boldsymbol{x}_c \odot \left(\boldsymbol{W}_{c_0 nc_i}\boldsymbol{h}_i^{nc} + \boldsymbol{b}_{c_0 nc_i}\right)\big] + \overline{\boldsymbol{W}}_{nc_i}\boldsymbol{h}_i^{nc} + \boldsymbol{b}_{nc_i}\Big\} \\ \boldsymbol{h}_j^{nc} &= g\left(\overline{\boldsymbol{W}}_{nc_i nc_j}\boldsymbol{h}_i^{nc} + \boldsymbol{b}_{nc_i nc_j}\right).\end{aligned} \tag{42}$$

Above, $\boldsymbol{W}$ and $\boldsymbol{b}$ are the weight matrices and bias vectors, respectively. Element-wise multiplication, i.e. the Hadamard product, is denoted by $\odot$.

In order to find out under which conditions each element of the output vector of the neural network $\hat{\boldsymbol{y}}_l$ is convex with respect to the input vector $\boldsymbol{x}_c$, consider the following:

- The neural network can be considered as a composition of functions, where each layer is a composition of the output of the previous layer, including the input $\boldsymbol{x}$ and the output layer $\hat{\boldsymbol{y}}$.
- According to Eq. $(42)_1$ and property 5 listed in Sec. 2.3.3, the activation function g ought to be convex and non-decreasing. Taking into account the fact that the network should also be integrable and derivable (at least twice differentiable), this is fulfilled by the softplus function among the standard activation functions Eq. (36).
- Likewise, property 5 requires that the argument of the activation function g is a convex function. In the present case, the element of the hidden layer j ought to be convex with respect to the output of the previous layer $\boldsymbol{h}_i^c$. Note that only the first term in curly brackets involves $\boldsymbol{h}_i^c$, so that the other terms can be considered as constant vectors with no influence on the convexity of the current layer (property 4).
- The $\boldsymbol{h}_i^c$ is multiplied component-wise with the activation function, and the activation function softplus is always positive. The result of this Hadamard multiplication is multiplied by the weight matrix $\widetilde{\boldsymbol{W}}_{c_i nc_i}^{c_j}$, which can be regarded as a sum of weighted convex functions of $\boldsymbol{h}_i^c$. According to property 3, the weights ought to be non-negative, which imposes the constraint $\boldsymbol{W}_{c_i nc_i}^{c_j} \geq 0$. Note that

all other weights and biases are not constrained in the above illustration. Note also that the weights connecting the input layer and the first hidden layer do not need to be non-negative to preserve convexity.

- If the above constraints are met, each element of layer *j* is convex with respect to the output of the previous layer *i*. Having in mind the recursive nature of the neutral network, this also means that the element of the output vector of the neural network $\hat{\boldsymbol{y}}_l$ is convex with respect to the input vector $\boldsymbol{x}_c$.

PICINN could actually be simpler and still preserve the convexity, however connecting the input layer $\boldsymbol{x}$ to the hidden layers $\boldsymbol{h}_i$ alleviates possible numerical problems [12, 37]. For more details see [4, 8, 30, 38].

FInally, convexity can be preserved by resorting to the custom activation functions. Some examples can be found in [7] and the follow-up works [6, 17].

3.3 Other continuum mechanics constraints in neural networks

The previous section has shown how convexity can be directly incorporated into the architecture of the neural network. This section now presents a suitable approach to enforcing the remaining constraints.

First of all, the neural network at hand will use components of the strain tensor and possibly other variables as input, while the strain energy is the output. Having the network designed in the manner described in Sec. 3.2 assures the convexity of the energy with respect to the strains. Training such a network would require the energy as part of the dataset. While the energy could be estimated somehow, the usual dataset consists only of stress-strain pairs. Therefore, using the automatic differentiation feature that relies on Eq. (41), the network designed as an energy-strain network can be readily transformed into a network that takes strains as input and gives the stresses as output. This modification automatically enforces the thermodynamic consistency Eq. (14). An additional automatic differentiation provides the tangent module:

$$\mathbf{C} = \rho \frac{\partial^2 \widehat{\psi}}{\partial \boldsymbol{\epsilon}^2} = \frac{\partial \boldsymbol{\sigma}}{\partial \boldsymbol{\epsilon}} \tag{43}$$

which is typically required in any calculation.

Although normalization constraints Eq. (27) can also be enforced in several ways, the approach described below enforces constraints exactly. To address these constraints, the strain energy and the stresses are redefined as [8, 18]:

$$\hat{\psi}(\boldsymbol{\epsilon}) = \tilde{\psi}(\boldsymbol{\epsilon}) - \tilde{\psi}(\mathbf{0}) - \partial_{\boldsymbol{\epsilon}}\tilde{\psi}(\boldsymbol{\epsilon})\big|_{\boldsymbol{\epsilon}=0}\boldsymbol{\epsilon}$$
$$\boldsymbol{\sigma}/\rho = \partial_{\boldsymbol{\epsilon}}\hat{\psi}(\boldsymbol{\epsilon}) = \partial_{\boldsymbol{\epsilon}}\tilde{\psi}(\boldsymbol{\epsilon}) - \partial_{\boldsymbol{\epsilon}}\tilde{\psi}(\boldsymbol{\epsilon})\big|_{\boldsymbol{\epsilon}=0}. \tag{44}$$

This modification is used during the training.

The normalization condition for the stresses in conjunction with the thermodynamic consistency Eq. (14) ensures that the strain energy has an extremum at $\boldsymbol{\sigma} = \mathbf{0}$. As the strain energy is a convex function in this case, this extremum is a minimum. As a result, the strain energy cannot be negative, as required by Eq. (28).

Due to the infinitesimal strains assumption, the role of the growth condition Eq. (29) is taken over by the convexity property of the network.

As a final constraint, the symmetry of the Cauchy stress tensor Eq. (3) can be enforced simply by training with only six different stress components, while the remaining three shear stress components are considered equal to the calculated three shear stress components.

With the normalization introduced above, the loss function Eq. (37) now is:

$$J_{\mathrm{MSE}} = \frac{1}{N}\sum_{i=1}^{N}(\partial_{\boldsymbol{\epsilon}}\tilde{\psi}(\boldsymbol{\epsilon}_i) - \partial_{\boldsymbol{\epsilon}}\tilde{\psi}(\boldsymbol{\epsilon})\big|_{\boldsymbol{\epsilon}=0} - \boldsymbol{\sigma}_i/\rho)^2. \tag{45}$$

If standardization is required, a straightforward modification of the above is required, see for an example in a one-dimensional context [4]. Weights and biases follow as the solution of the minimization problem Eq. (39).

As the first alternative approach, any type of constraint can be included via standard techniques readily available in optimization. In this sense, the penalty method is proposed in [20]. The method relies on an extension of the standard loss function, i.e. Eq. (37) or (38):

$$J^{\mathrm{C}}(\boldsymbol{W}, \boldsymbol{b}; \boldsymbol{x}, \boldsymbol{y}) = J + \tilde{J}, \tag{46}$$

where the new term $\tilde{J}$ is the term that includes constraints $c_j(\boldsymbol{x})=0$, $j = 1, \ldots, n_p$. Two possibilities are available at this point. In the classical penalty method, it is:

$$\tilde{J} = \frac{p}{N}\sum_{i=1}^{N}\sum_{j=1}^{n_p} c_j(\boldsymbol{x}_i)^2, \tag{47}$$

where p is the penalty factor. Alternatively, one can use the exact penalty method:

$$\tilde{J} = \frac{p}{N}\sum_{i=1}^{N}\sum_{j=1}^{n_P}\left|c_j(\boldsymbol{x}_i)\right|. \tag{48}$$

While the classical penalty method cannot enforce the equality constraints exactly, the exact penalty method has no gradient defined at c_j=0. This issue can be circumvented [20], although the gradient will still be discontinuous at c_j=0. While considering the choice of the penalty method, note that the neural network is not exact either, but only an approximation. This technique can be used to enforce inequalities too.

The third option is to simply add new points to the original dataset, with the new points being the actual constraints [20]. This, of course, only approximately enforces the constraints. An extension of this simple technique is known as oversampling, see for example [3]. As an illustration, in a stress normalization $\boldsymbol{\sigma}=\mathbf{0}$, these points usually already form part of the dataset. In oversampling, these points are simply copied multiple times in order to enlarge the dataset. This can also be understood as multiplying only the terms in the loss function with points in questions by a certain weighting factor, which increases the influence of these points in the standard loss functions of type Eq. (37) or (38).

4. EXAMPLE

To demonstrate the above concepts, an open access dataset already published in the literature [4] will be considered. The dataset contains numerical values of stress-strain curves obtained by molecular dynamics (MD) simulations of uniaxial tests of single-walled carbon nanotubes (SWCNTs). SWCNTs were tested both in tension and compression at a temperature of 300 K. 818 different SWCNTs were tested, covering all possible chiralities up to a SWCNT diameter of 4.0 nm. The length to diameter ratio was about 5. Further details on the calculations can be found in [3, 4, 38, 39]. Out of all the SWCNT configurations analyzed, only the armchair SWCNT(10,10) is considered in this example. The failure of this nanotube, as observed in the MD simulations under compression and tension, is shown in Figs 4, 5.

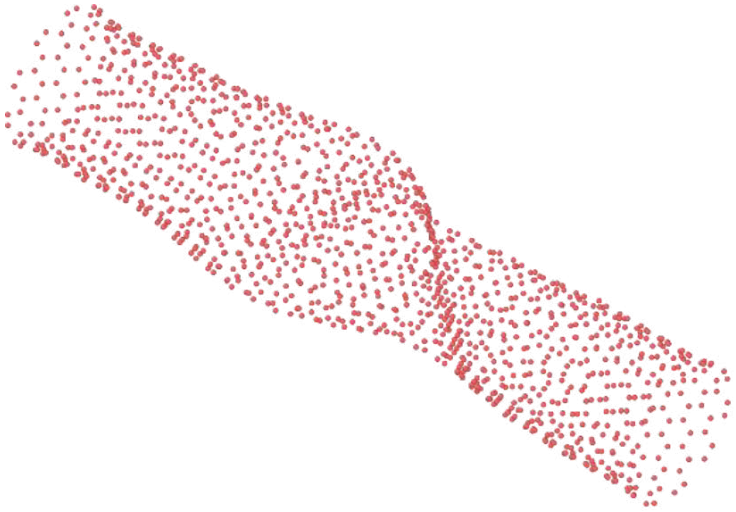

Figure 4. Shell-like buckling failure of a SWCNT(10,10).
Slika 4. Gubitak stabilnosti tlačno opterećene ugljikove nanocijevi SWCNT(10,10).

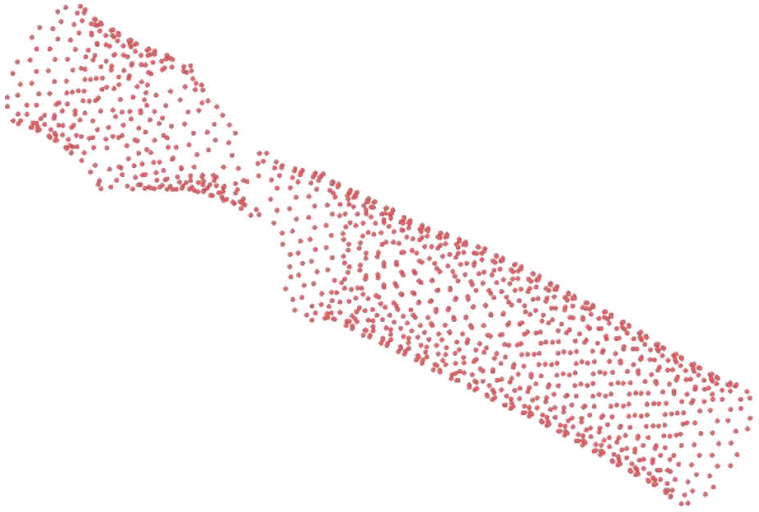

Figure 5. Fracture of a SWCNT(10,10) in tension.
Slika 5. Lom ugljikove nanocijevi SWCNT(10,10) pri vlačnom opterećenju.

The input variables of the dataset are the true strain and the parameters n, m, which define the chirality of a SWCNT, while the output variable is the true stress. The total size of the dataset is about 1.5 million points.

The central problem is that MD simulations performed at temperatures above absolute zero involve stochastic oscillations of the atoms, in this case the carbon atoms. These oscillations are related to the temperature at which the simulations are performed, since the temperature correlates with the kinetic energy of the atoms, but only with the part of the energy that comes from the stochastic part of the velocity. All atoms are therefore constantly oscillating, and these serve as a trigger for the self-oscillation of a SWCNT. Since the lowest vibrational mode of such a one-dimensional structure is bending, the SWCNT will be slightly bent. Continuous bending oscillations cause small stress fluctuations that are superimposed on the stresses generated by the tensile or compressive loading, see the inset in Fig. 7.

This represents a fundamental problem in developing a model that could be used as a surrogate to these MD simulations. In particular, if one tries to approximate the tangent modulus as

$$E_T = \frac{\Delta\sigma}{\Delta\epsilon}, \tag{49}$$

it is clear that between some neighboring points for the increase in strain $\Delta\epsilon$, a decrease in stresses $\Delta\sigma$ is observed. As a result, the tangent modulus will be negative (Fig. 9), which is physically unjustifiable. Although the problem can be solved in various ways, for example by reducing the number of points, we would like to apply the PICINN presented above to solve this issue.

The neural network is designed so that the output is the strain energy, which is convex in the strain as the input variable, while in the chirality parameters, it is not. The number of layers of the network shown in Fig. 3 is set to N=2. Other parameters of this neural network are: $n_c=n_{c3}$=1, n_{nc}=2, n_{cN}=15, $n_{c1}=n_{c2}$=10. The softplus function is used as the activation function. Some of the weights, as shown in Fig. 3, were normalized. The dataset was randomly split into the training-validation-test subsets, and the ratio used was 60:20:20 (or in terms of points 881,996: 293,999: 293,999). A batch size of 1,024 was used, and the training was performed for 500 epochs. The Glorot uniform initializer was used [40]. As already emphasized in Sec. 2.3.1, the network is designed to output the strain energy, whereas training is performed on the differentiated network to obtain the ability to train on stress-strain pairs while enforcing thermodynamic consistency. Eq. (45) was used as the loss function.

Training was repeated five times. The network that gave the lowest test loss was selected as a reference one. The loss change during the process is shown in Fig. 6. An overview of losses is given in Tab. 1.

Table 1. Training, validation, and test losses $\cdot 10^{-4}$ after 500 epochs for trainings 1-5. The best losses are indicated by the bold numerals.
Tablica 1. Vrijednost funkcije gubitka s trening, validacijskim i testnim podacima $\cdot 10^{-4}$ nakon 500 epoha, za treninge 1-5. Najbolje vrijednosti označene su masnim brojkama.

	ID 1	ID 2	ID 3	ID 4	ID 5
training	1.523	**0.739**	0.955	1.775	2.283
validation	1.566	**0.784**	0.996	1.688	2.274
test	1.566	**0.795**	1.009	1.690	2.271

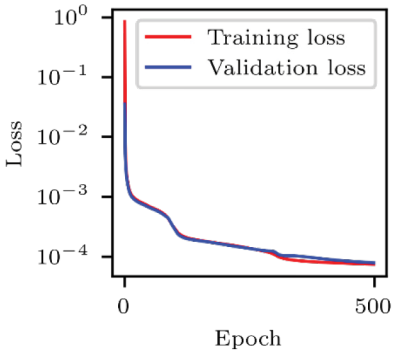

Figure 6. Reduction of loss during training.
Slika 6. Smanjenje funkcije gubitka tijekom treniranja.

Probably the most important result is the stress-strain curve. Fig. 7 shows the comparison between the result of the molecular dynamics and the one of the neural network. Both curves are almost identical where the overall curves are considered. However, the inset shows enlarged parts of the curves where the oscillatory nature of the MD curve becomes visible, while the convexity constraint obviously smoothes the curve. It should be repeatedly emphasized that if the physical constraints mentioned above are not followed, the neural network can be designed to capture the oscillations as well, compare these results with those in [3]. It is also visible that the stress vanishes for ϵ=0.

Once trained, the neural network is highly efficient in terms of computing time. This makes it an excellent candidate for a constitutive model that can be coupled with the finite element method or another numerical method. A stress or tangent modulus for a given

strain is evaluated in a fraction of a second, while an MD simulation providing the same data takes hours to complete. However, one should keep in mind that the generation of a training dataset by MD is computationally very intensive, although performed only once. The full power of a neural network comes therefore into play in problems where repeated calculation of the stress or tangential modulus is required.

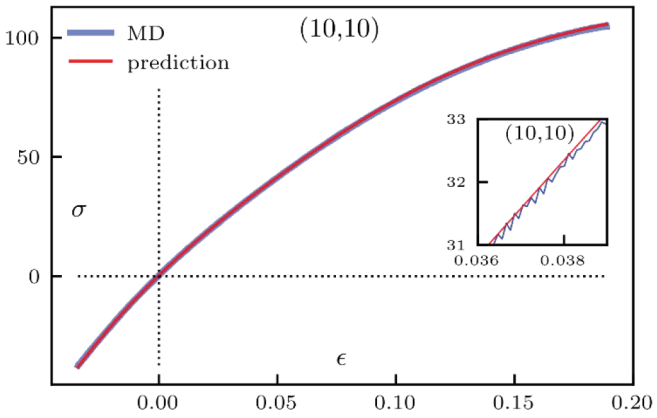

Figure 7. Comparison of MD and stress (GPa) -strain curves predicted by the neural network.
Slika 7. Usporedba krivulja naprezanje (GPa) – deformacija dobivenih molekularnom dinamikom i neuronskom mrežom.

The strain energy is obviously convex, Fig. 8. Since the nanotube loaded in compression fails due to the shell-like buckling, lower absolute minimum strains can be sustained than in tension, so the curve is shorter in the compression part. Also, the strain energy vanishes for

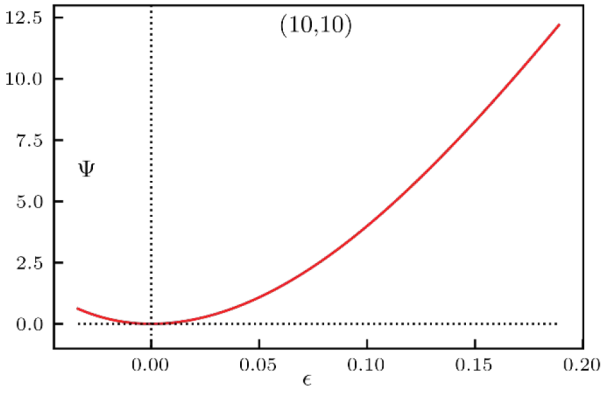

Figure 8. Predicted strain energy (GPa) vs. true strain.
Slika 8. Predviđana energija deformiranja (GPa) – stvarna deformacija.

The comparison of the tangent modulus is shown in Fig. 9. The one obtained from the MD is calculated using Eq. (49). Clearly, this modulus undergoes large fluctuations and assumes negative values. However, the curve of the tangent modulus obtained by PICINN is smooth and always positive, Fig. 10.

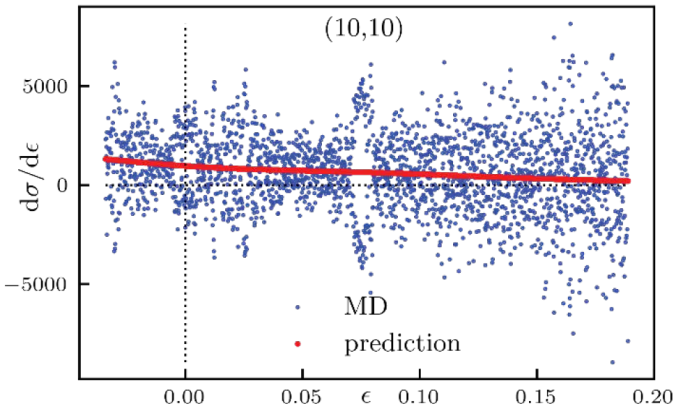

Figure 9. Comparison of MD and predicted tangent modulus (GPa).
Slika 9. Usporedba predviđanog tangentnog modula s modulom dobivenim molekularnom dinamikom, (GPa).

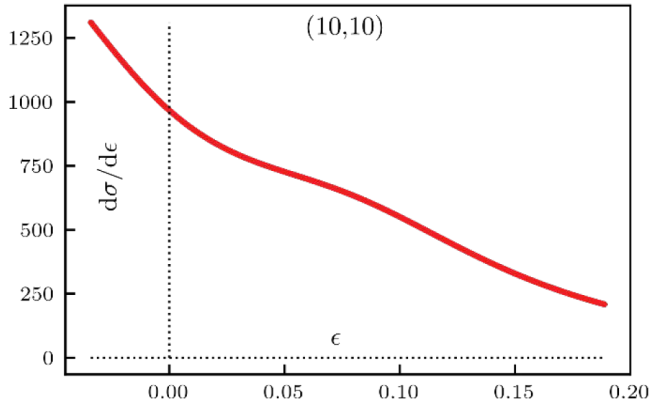

Figure 10. Predicted tangent modulus (GPa).
Slika 10. Predviđani tangentni modul (GPa).

CONCLUSIONS

The paper presents elementary physical constraints that can be built into a neural network. More specifically, feed-forward neural networks are the focus of this work. At the beginning, the continuum mechanics constraints were presented, followed by their implementation in neural networks. Several possible methods for incorporating these principles into the neural network architecture are presented. The most important constraints considered were convexity, objectivity, thermodynamic consistency and normalization.

The above constraints are typically used in the context of finite strains. In contrast, in the present study, the implementation was adapted to infinitesimal strains, more precisely to nonlinear elasticity. It was shown that while some rather strict constraints cannot be fulfilled simultaneously in finite strains, namely convexity and objectivity, this might simply be applied in the infinitesimal framework.

A demonstration of the efficiency of the method can be found in the example. It is shown that convexity indeed eliminates unwanted stochastic effects, and enforces other physical constraints exactly.

ACKNOWLEDGMENTS

The University of Rijeka, under project number uniri-iskusni-tehnic-23-37, has supported this work. We gratefully acknowledge this support.

References

[1] W. S. McCulloch, W. Pitts, A logical calculus of the ideas immanent in nervous activity, The bulletin of mathematical biophysics 5 (1943) 115–133.

[2] T. Kirchdoerfer, M. Ortiz, Data-driven computational mechanics, Computer Methods in Applied Mechanics and Engineering 304 (2016) 81–101.

[3] V. Košmerl, I. Štajduhar, M. Čanađija, Predicting stress–strain behavior of carbon nanotubes using neural networks, Neural Computing and Applications (2022) 1–16.

[4] M. Čanađija, V. Košmerl, M. Zlatić, D. Vrtovšnik, N. Munjas, A computational framework for nanotrusses: Input convex neural networks approach, European Journal of Mechanics-A/Solids 103 (2024) 105195.

[5] M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, *Journal of Computational Physics*, 378 (2019) 686-707.

[6] M. Zlatić, M. Čanađija, Recovering Mullins damage hyperelastic behavior with physics augmented neural networks, Journal of the Mechanics and Physics of Solids, (2024) 105839.

[7] K. Linka, E. Kuhl, A new family of constitutive artificial neural networks towards automated model discovery, Computer Methods in Applied Mechanics and Engineering 403 (2023) 115731.

[8] F. As' ad, P. Avery, C. Farhat, A mechanics-informed artificial neural network approach in data-driven constitutive modeling, International Journal for Numerical Methods in Engineering 123 (12) (2022) 2738–2759.

[9] K. A. Kalina, P. Gebhart, J. Brummund, L. Linden, W. Sun, M. Kästner, Neural network-based multiscale modeling of finite strain magnetoelasticity with relaxed convexity criteria, Computer Methods in Applied Mechanics and Engineering 421 (2024) 116739.

[10] M. Zlatić, M. Čanađija, Incompressible rubber thermoelasticity: a neural network approach, *Computational Mechanics*, 71 (2023), 895-916.

[11] J. N. Fuhg, A. Jadoon, O. Weeger, D. T. Seidl, R. E. Jones, Polyconvex neural network models of thermoelasticity, *arXiv preprint* (2024) arXiv:2404.15562.

[12] B. Amos, L. Xu, J. Z. Kolter, Input convex neural networks, in: International Conference on Machine Learning, PMLR, 2017, 146–155.

[13] T. Kapoor, H. Wang, A. Núñez, R. Dollevoet, Physics-informed neural networks for solving forward and inverse problems in complex beam systems, *IEEE Transactions on Neural Networks and Learning Systems* 35 (2023) 5981-5995.

[14] O. Kianian, S. Sarrami, B. Movahedian, M. Azhari, PINN-based forward and inverse bending analysis of nanobeams on a three-parameter nonlinear elastic foundation including hardening and softening effect using nonlocal elasticity theory, *Engineering with Computers* (2024) 1-27.

[15] W. He, J. Li, X. Kong, L. Deng, Multi-level physics informed deep learning for solving partial differential equations in computational structural mechanics, *Communications Engineering* (2024), 3:151.

[16] D. W. Abueidda, M.E. Mobasher, Variational temporal convolutional networks for I-FENN thermoelasticity, Computer Methods in Applied Mechanics and Engineering, 429 (2024), p.117122.

[17] M. Zlatić, F. Rocha, L. Stainier, M. Čanađija, Data-driven methods for computational mechanics: a fair comparison between neural networks based and model-free approaches 431 (2024) 117289.

[18] S. Huang, Z. He, C. Reina, Variational Onsager Neural Networks (VONNs): a thermodynamics-based variational learning strategy for non-equilibrium PDEs, Journal of the Mechanics and Physics of Solids 163 (2022) 104856.

[19] P. Thakolkaran, A. Joshi, Y. Zheng, M. Flaschel, L. De Lorenzis, S. Kumar, NN-EUCLID: deep-learning hyperelasticity without stress data, Journal of the Mechanics and Physics of Solids 169 (2022) 105076.
[20] P. Weber, J. Geiger, W. Wagner, Constrained neural network training and its application to hyperelastic material modeling, Computational Mechanics 68 (2021) 1179–1204.
[21] D. K. Klein, M. Fernández, R. J. Martin, P. Neff, O. Weeger, Polyconvex anisotropic hyperelasticity with neural networks, Journal of the Mechanics and Physics of Solids 159 (2022) 104703.
[22] L. Linden, D. K. Klein, K. A. Kalina, J. Brummund, O. Weeger, M. Kästner, Neural networks meet hyperelasticity: A guide to enforcing physics, Journal of the Mechanics and Physics of Solids 179 (2023) 105363.
[23] G. A. Holzapfel, Nonlinear Solid Mechanics: a Continuum Approach for Engineering Science, Kluwer Academic Publishers, 2002.
[24] M. Čanađija, Thermomechanics of Solids and Structures, Elsevier, 2023.
[25] V. Lubarda, On thermodynamic potentials in linear thermoelasticity, Int. J. Solids Struct. 41 (26) (2004) 7377–7398.
[26] C. Truesdell, W. Noll, The Non-Linear Field Theories of Mechanics, Springer, 2004.
[27] J. M. Ball, Convexity conditions and existence theorems in nonlinear elasticity, Archive for rational mechanics and Analysis 63 (1976) 337–403.
[28] M. Silhavy, The mechanics and thermodynamics of continuous media, Springer Science & Business Media, 2013.
[29] D. G. Luenberger, Y. Ye, Linear and Nonlinear Programming, Springer, 2008.
[30] S. P. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, 2004.
[31] G. A. Maugin, The thermomechanics of nonlinear irreversible behaviours, Vol. 27, World scientific, 1999.
[32] B. D. Coleman, W. Noll, On the thermostatics of continuous media, Archive for rational mechanics and analysis 4 (1959) 97–128.
[33] C. B. Morrey Jr, Quasi-convexity and the lower semicontinuity of multiple integrals. (1952).
[34] V. Ebbing, Design of polyconvex energy functions for all anisotropy classes, Inst. für Mechanik, Abt. Bauwissenschaften, 2010.
[35] J. Schröder, P. Neff, Invariant formulation of hyperelastic transverse isotropy based on polyconvex free energy functions, International journal of solids and structures 40 (2) (2003) 401–445.

[36] I. Goodfellow, Y. Bengio, A. Courville, Deep learning, MIT press, 2016.
[37] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.
[38] M. Čanađija, Deep learning framework for carbon nanotubes: Mechanical properties and modeling strategies, Carbon 184 (2021) 891–901.
[39] M. Čanađija, S. Ivić, Carbon nanotubes as a basis of metamaterials and nanostructures: Crafting via design optimization, Mechanics of Materials 197 (2024) 105105.
[40] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, PMLR 9:249-256, 2010.

O FIZIKALNO INFORMIRANIM NEURONSKIM MREŽAMA – UKLJUČIVANJE OGRANIČENJA MEHANIKE KONTINUUMA PRI INFINITEZIMALNIM DEFORMACIJAMA

Sažetak

Članak predstavlja metode uključivanja fizikalnih ograničenja koja potiču iz mehanike kontinuuma. Fokus je na infinitezimalnim deformacijama u nelinearnoj elastičnosti. Predstavljaju se osnovna ograničenja s kratkim pregledom mehanike kontinuuma, koja se zatim primjenjuju u unaprijednim neuronskim mrežama. Između ostalih, uključuju se konveksnost, objekitvnost, termodinamička konzistentnost i normalizacija. Kada je to prikladno, predstavlja se nekoliko različitih metoda za isto ograničenje. Ponašanje ovakvih fizikalno informiranih neuronskih mreža pokazano je razvoju konstitutivnog modela ugljikovih nanocijevi s jednom stijenkom.

Ključne riječi: fizikalno informirane neuronske mreže; fizikalna ograničenja; mehanika kontinuuma; konveksnost.

Marko Čanađija
Faculty of Engineering
University of Rijeka
Vukovarska 58, 51000 Rijeka, Croatia
e-mail: marko.canadija@riteh.uniri.hr

Martin Zlatić
Faculty of Engineering
University of Rijeka
Vukovarska 58, 51000 Rijeka, Croatia
e-mail: martin.zlatic@riteh.uniri.hr