A benchmark for elasto-plasticity in finite strain

Louis Lesueur, Anders Thorin, Daniel Weisz-Patrault

A benchmark for elasto-plasticity in finite strain

Louis LESUEUR[1,2], Anders THORIN[1], Daniel WEISZ-PATRAULT[2]

[1] Université Paris-Saclay, CEA, List, F-91120, Palaiseau, France
[2] Laboratoire de Mécanique des Solides, CNRS UMR 7649, École Polytechnique Institut Polytechnique de Paris, F-91128 Palaiseau

ABSTRACT Finite strain elasto-plastic simulations are critical in fields such as materials science (metal forming, forging, additive manufacturing) and automotive engineering (crash simulations). These simulations are traditionally carried out using computationally intensive finite element analysis (FEA), which limits their use in optimization tasks (e.g., optimal control, design processes) and real-time applications (e.g., teleoperation, personnel training). In this work, we introduce a benchmark dedicated to highly non-linear elasto-plastic simulations, designed to evaluate and develop neural network models tailored for solving elasto-plastic problems under finite strain conditions, ultimately unlocking potentially real-time optimization and interactive simulations. The datasets include simulations of 1D and 3D elements, featuring quasi-static sequences of applied loads on complex geometries, and the resulting computed physical quantities: displacements fields, plastic flow coefficient field, stresses. To specifically evaluate the impact of plasticity on different neural networks, the datasets also feature simulations with identical inputs but employing a purely elastic constitutive law.

Link to open datasets: https://doi.org/10.57745/HQJMYA.

1. Introduction

Finite strain elasto-plastic simulations are essential in fields involving significant material deformation, where both elastic and plastic behaviors occur. Key applications include aerospace (crashworthiness, landing gear), automotive (crash simulations, forming), civil engineering (geotechnical, seismic analysis), materials science (metal forming, additive manufacturing) or biomechanics (tissue modeling, implants). Traditionally, these simulations are carried out using computationally intensive finite element analysis (FEA), which limits their use in optimization tasks (e.g., optimal control, design processes) and real-time applications (e.g., tele-operation, personnel training).

This work introduces a benchmark dedicated to highly non-linear elasto-plastic simulations, designed to evaluate and develop neural network models tailored for solving elasto-plastic problems under finite strain conditions. By leveraging the capabilities of neural networks, we aim to unlock the potential for optimization and interactive simulations, thereby addressing the computational challenges associated with traditional FEA methods.

Scientific ML: In recent years, ML has made spectacular progress, particularly in the fields of Computer Vision and language processing. These advancements now extend to "Scientific ML" which involves the use of ML algorithms in scientific contexts. Notably, the development of PINNs [6, 18] has demonstrated the ability of NNs to accurately model the solutions of differential equations.

NN in computational mechanics : More specifically in computational mechanics, NNs have been successfully used to develop surrogate models in non-linear solid mechanics, offering inference speeds that are several orders of magnitude faster than FEA. An overview of the latest techniques is presented in [9]. Three main approaches have emerged:

1. Physics-Informed Neural Networks (PINN): In the PINN approach, the solution u of a differential equation is approximated by a neural network $\hat{u}$. This approach allows the use of physics-inspired loss functions, such as minimizing a residual. PINNs are very popular and can be used to model linear elasticity [7] as well as nonlinear behaviors in solid mechanics, such

as elastoplasticity [1, 20]. Specific PINN architectures exist to enforce physical constraints like thermodynamics [15]. It is worth noting that in PINNs for PDEs, boundary conditions are incorporated into the loss function. This requires external forces to be known or parameterized beforehand, which is not always feasible.

2. Global Approaches: In these approaches, a neural network architecture is also used to approximate a solution function u of an equation but unlike PINNs, the goal is to predict its values at specific points within a domain based on input loads and boundary conditions. This approach is useful when the cost of evaluating a residual is too high [16] or when the supporting geometries are too complex. Regarding computer mechanics, a popular global use of neural networks is to learn only the constitutive law and integrate it into finite element software [8, 10].

3. Generative Neural Networks: Particularly popular in fluid mechanics [12], generative machine learning architectures, such as GANs or VAEs, can be used to generate physically realistic displacement or stress fields. Similar to image generation, controlling the outputs via prompting is a major challenge.

Datasets in computer mechanics: These advances have motivated the creation of benchmarks such as [19] and [2], which compile results from various time-dependent simulations modeling of diverse physical phenomena. Although comprehensive, these benchmarks are designed for general PDE-solving algorithms. Regarding applications in computational mechanics, specifically continuum mechanics, there is a lack of comprehensive benchmarks, and often the data used in papers are not accessible.[14] is arguably the most complete dataset in computational mechanics, but it pertains to fluid mechanics. In solid mechanics, specific datasets for certain problems can be found, such as [17], which contains semi-realistic images of porous materials. Concerning elasticity, the problem being straightforward, few papers share their data, preventing reproducibility. An exception is [3] which provide an example and database of 2D-elasticity. [21] also shares data from elastic simulations using the Finite Element Method (FEM) on plates. For non-linear solid simulations, few articles exist, and none share data to the authors' knowledge. Results of particle-based simulation are shared in [11] for reinforcement learning.

The following derives the governing equations of finite strain elasto-plasticity (section. 2). The two databases are then described (sections 3 and 4).

2. Problem statement

2.1. Large transformation plasticity

The basic set of equations of the theory of plasticity at finite strains is briefly recalled for the sake of clarity. The three-dimensional Cauchy model is first presented, and then the uni-dimensional Timoshenko beam theory, as both are used in the following datasets.

2.1.1. Cauchy model

The transformation from the reference configuration Ω_0 to the current configuration Ω_t is parametrized by $\boldsymbol{x} = \boldsymbol{\Phi}(\boldsymbol{X}, t)$, where $\boldsymbol{\Phi}$ is a bijective function called *transformation*, $\boldsymbol{X}$ denotes a particle in the reference configuration, t the time, and $\boldsymbol{x}$ the location of the particle in the current configuration. The displacement field $\boldsymbol{u}$ reads $\boldsymbol{u} = \boldsymbol{\Phi} - \boldsymbol{X}$. The Lagrangian description relies on the *transformation gradient* $\boldsymbol{F} = \boldsymbol{\nabla}_{\boldsymbol{X}}\boldsymbol{\Phi}$, which transforms a volume element $\mathrm{d}\Omega_0$ in the reference configuration into $\mathrm{d}\Omega$ in the current configuration. Classically, the Green-Lagrange deformation is defined as follows:

$$\boldsymbol{e} = \frac{1}{2}\left(\boldsymbol{F}^\top \cdot \boldsymbol{F} - \boldsymbol{1}\right) \tag{2.1}$$

Classical finite strains theory also involves a stress-free configuration called *released configuration* that is obtained by unloading the volume element $\mathrm{d}\Omega$. The transformation gradient can therefore be decomposed into the second order tensor $\boldsymbol{F}_E$ representing the elastic part of the transformation while $\boldsymbol{F}_P$ represents the plastic part:

$$\boldsymbol{F} = \boldsymbol{F}_E \cdot \boldsymbol{F}_P \tag{2.2}$$

$\boldsymbol{F}_P$ is often assumed to be isochoric, i.e., $\det \boldsymbol{F}_P = 1$ so that the volume variation reduces to $J = \det \boldsymbol{F} = \det \boldsymbol{F}_E$. Both $\boldsymbol{F}_E$ and $\boldsymbol{F}_P$ are incompatible in the sense that they are not gradients of any transformation in general. The decomposition (2.2) is not unique a priori. The macroscopic plasticity uniqueness is ensured by imposing the symmetry of $\boldsymbol{F}_E$, i.e., $\boldsymbol{F}_E^\top = \boldsymbol{F}_E$.

The Eulerian description leads to define the 2nd order deformation rate tensor $\boldsymbol{d} = \mathrm{sym}\left[\dot{\boldsymbol{F}} \cdot \boldsymbol{F}^{-1}\right]$, where sym denotes the symmetric part of a tensor. By using the decomposition (2.2) one defines the plastic strain rate:

$$\boldsymbol{d}_P = \mathrm{sym}\left[\dot{\boldsymbol{F}}_P \cdot \boldsymbol{F}_P^{-1}\right] \tag{2.3}$$

Note that $\boldsymbol{d}_P$ is a 2nd order deviatoric tensor. The cumulative plastic strain rate denoted by $\dot{p}_{\mathrm{cum}}$ is then defined as follows:

$$\dot{p}_{\mathrm{cum}} = \sqrt{\frac{2}{3}\,\boldsymbol{d}_P : \boldsymbol{d}_P} \tag{2.4}$$

Introducing the virtual power of internal forces the symmetric 2nd order Cauchy stress tensor is defined and denoted by $\boldsymbol{\sigma}$ (Pa) representing the internal forces per unit surface. In virtue of the *virtual power principle* one obtains the following equilibrium equation without body and acceleration forces and and boundary conditions:

$$\begin{cases} \forall \boldsymbol{x} \in \Omega_t,\ \mathrm{div}\left[\boldsymbol{\sigma}(\boldsymbol{x},t)\right] = 0 \\ \forall \boldsymbol{x} \in \partial\Omega_{T,t},\ \boldsymbol{\sigma}(\boldsymbol{x},t) \cdot \boldsymbol{n}(\boldsymbol{x},t) = \boldsymbol{\Sigma}_{\mathrm{ext}}(\boldsymbol{x},t) \end{cases} \tag{2.5}$$

where $\partial\Omega_{T,t} \subset \partial\Omega_t$ is the part of the boundary where the external stress vector $\boldsymbol{\Sigma}_{\mathrm{ext}}$ (Pa) is applied, and $\boldsymbol{n}$ is the outer normal vector at the boundary.

Neglecting kinematic hardening, common thermodynamic assumptions lead to the following von Mises flow rule:

$$\boldsymbol{d}_P = \frac{3}{2}\,\frac{\dot{p}_{\mathrm{cum}}}{\sigma_Y(p_{\mathrm{cum}})}\mathrm{dev}\left[\boldsymbol{\kappa}\right] \tag{2.6}$$

where $\boldsymbol{\kappa} = J\,\boldsymbol{F}_E^{-1} \cdot \boldsymbol{\sigma} \cdot \boldsymbol{F}_E$ is the 2nd order Mandel stress tensor, $\sigma_Y > 0$ is the yield stress (assumed to only depends on p_{cum}) and dev denotes the deviatoric part of a tensor. In addition, the elastic domain is the convex set defined by the Von Mises yield criterion, which is the set of $\boldsymbol{\kappa}$ that satisfies the inequality:

$$\sqrt{\frac{3}{2}\,\mathrm{dev}\left[\boldsymbol{\kappa}\right] : \mathrm{dev}\left[\boldsymbol{\kappa}\right]} \leq \sigma_Y(p_{\mathrm{cum}}). \tag{2.7}$$

Furthermore, for an isotropic Neo-Hookean material, one can show that:

$$\boldsymbol{\sigma} = \frac{\mu_0}{J^{\frac{5}{3}}}\boldsymbol{F}_E^2 + \left(k_0 J(J-1) - \frac{\mu_0}{J^{\frac{5}{3}}}\frac{\mathrm{tr}\left[\boldsymbol{F}_E^2\right]}{3}\right)\boldsymbol{1} \tag{2.8}$$

where k_0 and μ_0 are the bulk and shear coefficients respectively. In the databases, $k_0 = 5.25\times10^{11}\,\mathrm{Pa}$ and $\mu_0 = 8.08 \times 10^{10}\,\mathrm{Pa}$. In addition, the hardening curve is written in the form:

$$\sigma_Y(p_{\mathrm{cum}}) = \sigma_0 + k\,p_{\mathrm{cum}}^n \tag{2.9}$$

where σ_0 (Pa) is the initial yield stress, k (Pa) a hardening coefficient and n a dimensionless exponent. In the following $\sigma_0 = 300\,\mathrm{MPa}$, $k = 0.9\,\mathrm{MPa}$ and $n = 0.13$, resulting in the hardening

curve showed in Fig. 1.

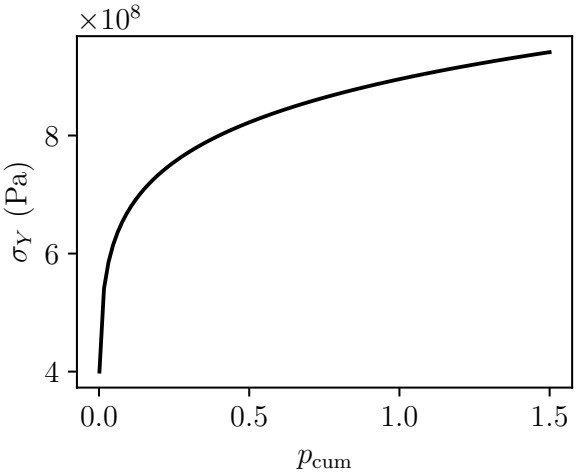


Figure 1: Hardening curve of simulations where σ_Y is in Pa and p_{cum} is dimensionless.

2.1.2. Timoshenko beam theory This subsection is dedicated to the paperclip dataset, see section 3. The Timoshenko beam theory is uni-dimensional, hence the reference and current configurations are defined by their curvilinear absissa respectively denoted by s_0 and s and the smooth mappings $[0, l_0] \ni s_0 \mapsto \boldsymbol{X}(s_0)$ and $[0, l] \ni s \mapsto \boldsymbol{x}(s, t)$ (where l_0 and l are respectively the initial and the current length of the beam, $\boldsymbol{X}$ is the particle in the reference configuration and $\boldsymbol{x}$ the location of the particle in the current configuration). The Lagrangian description relies on a bijective transformation function, which reads $\boldsymbol{x} = \boldsymbol{\Phi}(s_0, t)$. The displacement of the neutral axis reads $\boldsymbol{U} = \boldsymbol{\Phi} - \boldsymbol{X}$. The length variation reads $J = \mathrm{d}s/\mathrm{d}s_0$.

In addition, the Timoshenko beam theory assumes that a rigid cross-section is attached to each particle $\boldsymbol{X}$. Therefore one can define a direct orthonormal reference frame $(\boldsymbol{n}_0(s_0), \boldsymbol{i}_0(s_0), \boldsymbol{j}_0(s_0))$ for the cross-section in the reference configuration, where $\boldsymbol{n}_0$ is the outer normal vector of the cross-section. Consider, the cross-section frame in the current configuration denoted by $(\boldsymbol{n}(s, t), \boldsymbol{i}(s, t), \boldsymbol{j}(s, t))$, where $\boldsymbol{n}$ is the outer normal vector of the cross section in the current configuration. There exists an orthogonal second-order tensor $\boldsymbol{O}(s_0, t)$ such that $\boldsymbol{n} = \boldsymbol{O} \cdot \boldsymbol{n}_0$ (and similarly for $\boldsymbol{i}, \boldsymbol{j}$). One can for instance consider the three Euler angles of $\boldsymbol{O}$ in the reference configuration denoted by $(\varphi_{n_0}, \varphi_{i_0}, \varphi_{j_0})$. The associated curvature variation reads:

$$\boldsymbol{\chi} = \boldsymbol{O}^{-1} \cdot \frac{\partial \boldsymbol{O}}{\partial s_0} \tag{2.10}$$

which is an anti-symmetric 2nd order tensor, which can be re-written as a three components vector.

Consider the tangent unit vector in the reference configuration $\boldsymbol{t}_0(s_0) = \partial \boldsymbol{X}/\partial s_0$. It is assumed that in the reference configuration the cross section outer normal vector aligns with the tangent vector of the neutral axis, i.e $\boldsymbol{n}_0 = \boldsymbol{t}_0$. Consider the tangent unit vector in the current configuration $\boldsymbol{t}(s, t) = \partial \boldsymbol{x}/\partial s$. Unlike the Euler-Bernoulli beam theory $\boldsymbol{n} \neq \boldsymbol{t}$ in general. Consider the rotation angles θ^u, θ^v such as $\boldsymbol{t} = \cos(\theta^v)\cos(\theta^u)\boldsymbol{n} + \cos(\theta^v)\sin(\theta^u)\boldsymbol{i} - \sin(\theta^v)\boldsymbol{j}$, which is completed by unit vectors $\boldsymbol{u}, \boldsymbol{v}$ to form a direct orthonormal frame, where $\boldsymbol{u} = -\sin(\theta^u)\boldsymbol{n} + \cos(\theta^u)\boldsymbol{i}$ and $\boldsymbol{v} = \sin(\theta^v)\cos(\theta^u)\boldsymbol{n} + \sin(\theta^v)\sin(\theta^u)\boldsymbol{i} + \cos(\theta^v)\boldsymbol{j}$.

The Lagrangian analysis of the transformation of a beam element $\mathrm{d}\Omega_0$ into $\mathrm{d}\Omega$ leads to the

following elastic-plastic decomposition:

$$\left\{\begin{array}{l} J = J_E\, J_P \\ \boldsymbol{\chi} = \boldsymbol{\chi}_E + \boldsymbol{\chi}_P \\ \theta^u = \theta^u_E + \theta^u_P \\ \theta^v = \theta^v_E + \theta^v_P \end{array}\right. \tag{2.11}$$

where the subscript E sands for *elastic* and P for *plastic*. Note that $\boldsymbol{\chi}_E$ and $\boldsymbol{\chi}_P$ are three components vectors (or alternatively anti-symmetric 2nd order tensors).

The Eulerian description involves the generalized deformation rate vector $\boldsymbol{d} = \partial \boldsymbol{v}/\partial s - \boldsymbol{w} \cdot \boldsymbol{t}$ where $\boldsymbol{w} = \dot{\boldsymbol{O}} \cdot \boldsymbol{O}^{-1}$ is an anti-symmetric 2nd order tensor, which can be re-written as a three components vector, and the curvature rate $\partial \boldsymbol{\omega}/\partial s$, which leads to define the plastic strain rate vectors:

$$\boldsymbol{d}_P = \left(\dot{J}_P\, J_P^{-1}, \dot{\theta}^u_P, \dot{\theta}^v_P\right) \text{ and } \boldsymbol{\omega}_P = \dot{\boldsymbol{\chi}}_P. \tag{2.12}$$

Hence the cumulative plastic strain rate reads:

$$\dot{p}_d = \sqrt{\boldsymbol{d}_P \cdot \boldsymbol{d}_P} \text{ and } \dot{p}_\omega = \sqrt{\boldsymbol{\omega}_P \cdot \boldsymbol{\omega}_P} \tag{2.13}$$

and the cumulative plastic strain reads:

$$p_{\text{cum}} = \int_t (\dot{p}_d + \dot{p}_w)\ \mathrm{d}t. \tag{2.14}$$

Introducing the virtual power of internal forces the generalized forces in the current configuration: $\boldsymbol{R}$ (N) representing the resultant internal force and $\boldsymbol{M}$ (N.m) representing the internal bending moment. In virtue of the *virtual power principle* one obtains the following equilibrium equations and boundary conditions:

$$\left\{\begin{array}{l} \dfrac{\partial \boldsymbol{R}(s,t)}{\partial s} + \boldsymbol{f}_{\text{ext}}(s,t) = 0 \\ \boldsymbol{R}(s,t) \times \boldsymbol{t}(s,t) - \dfrac{\partial \boldsymbol{M}(s,t)}{\partial s} + \boldsymbol{c}_{\text{ext}}(s,t) = 0 \\ \boldsymbol{R}(0,t) = -\boldsymbol{R}_{\text{ext},0}(t) \text{ and } \boldsymbol{R}(l,t) = \boldsymbol{R}_{\text{ext},l}(t) \\ \boldsymbol{M}(0,t) = -\boldsymbol{M}_{\text{ext},0}(t) \text{ and } \boldsymbol{M}(l,t) = \boldsymbol{M}_{\text{ext},l}(t) \end{array}\right. \tag{2.15}$$

where $\boldsymbol{R}_{\text{ext},0}, \boldsymbol{R}_{\text{ext},l}$ (N) are the external forces applied in $s = 0$ and $s = l$ respectively, $\boldsymbol{M}_{\text{ext},0}, \boldsymbol{M}_{\text{ext},l}$ (N.m) are the external bending moments applied in $s = 0$ and $s = l$ respectively, and $\boldsymbol{f}_{\text{ext}}$ (N/m) and $\boldsymbol{c}_{\text{ext}}$ (N.m/m) are the external force per unit length and external bending moment per unit length respectively.

Consider the generalized force $\boldsymbol{R}_0$ (N) and bending moment $\boldsymbol{M}_0$ (N.m) defined in the reference configuration:

$$\begin{array}{l} \boldsymbol{R}_0 = (N, Q^u, Q^v) \\ \boldsymbol{M}_0 = J_E\, \boldsymbol{O}^{-1}.\boldsymbol{M}.\boldsymbol{O} \end{array} \tag{2.16}$$

where $N = J_E \boldsymbol{R} \cdot \boldsymbol{t}$ is the normal force, and $Q^u = J_E \cos(\theta^v) \boldsymbol{R} \cdot \boldsymbol{u}$ and $Q^v = -J_E \boldsymbol{R} \cdot \boldsymbol{v}$ are the shear forces.

Neglecting kinematic hardening, common thermodynamic assumption lead to the following flow rule:

$$\boldsymbol{d}_P = \frac{\dot{p}_d}{S\, \sigma_Y}\, \boldsymbol{R}_0 \text{ and } \boldsymbol{\omega}_P = \frac{\dot{p}_\omega}{S\, \sigma_Y}\, \boldsymbol{M}_0 \tag{2.17}$$

where S is the area of the cross section. In addition, the elastic domain is the convex set defined by:

$$\left\{\begin{array}{l} \sqrt{\boldsymbol{R}_0.\boldsymbol{R}_0} - S\, \sigma_Y \leq 0 \\ \sqrt{\boldsymbol{M}_0.\boldsymbol{M}_0} - SL\, \sigma_Y \leq 0 \end{array}\right. \tag{2.18}$$

Furthermore for isotropic material the following relations hold:

$$\begin{cases} N = ES\, e_E \\ Q^u = GS\, \theta_E^u \\ Q^v = GS\, \theta_E^v \\ \boldsymbol{M}_0 = E\boldsymbol{I} \cdot \boldsymbol{\chi}_E \end{cases} \tag{2.19}$$

where $e_E = 1/2(J_E^2 - 1)$ is the Green-Lagrange elastic strain, E (MPa) the Young modulus, G (MPa) the shear coefficient, S the area of the section, and $\boldsymbol{I}$ (m^4) the 2nd order tensor of second moment of area of the section.

3. Paperclip dataset

The "paperclip" dataset consists of a set of quasi-static simulations that model the action of human fingers on a paperclip modeled as a Timoshenko beam. It is openly accessible [13].

3.1. Input: Mesh and boundary conditions

The mesh of the paperclip is presented in Fig. 2.

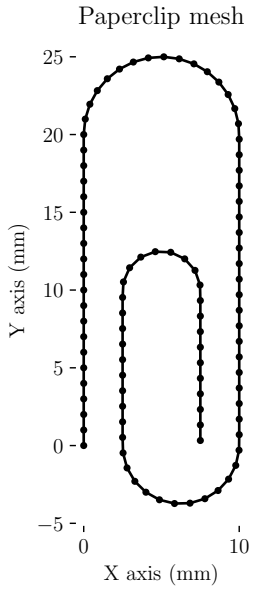


Figure 2: Mesh of the paperclip used the first dataset.

To model the action of hands deforming the paperclip, two wrenches are applied at two points on the paperclip $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, identified by their curvilinear abscissas $s = s_1$ and $s = s_2$. For the static equilibrium to be satisfied, the two wrenches are balanced:

$$\begin{cases} \boldsymbol{R}_{1,t} + \boldsymbol{R}_{2,t} = 0 \\ \boldsymbol{M}_{2,t} + \boldsymbol{M}_{1,t} + \boldsymbol{x}_1 \times \boldsymbol{R}_{1,t} + \boldsymbol{x}_2 \times \boldsymbol{R}_{2,t} = 0 \end{cases} \tag{3.1}$$

where $\times$ denotes the cross product, $\boldsymbol{R}_{1,t}, \boldsymbol{R}_{2,t}$ are forces and $\boldsymbol{M}_{1,t}, \boldsymbol{M}_{2,t}$ are bending moments applied to $\boldsymbol{x}_1, \boldsymbol{x}_2$ at any time t. Using the equilibrium (3.1) for each series, it is sufficient to choose the points of application and generate two sequences $\boldsymbol{R}_{1,t}$ and $\boldsymbol{M}_{1,t}$. The procedure is as follows:

1. We randomly choose a pair of curvilinear abscissas (s_1, s_2). For symmetry reasons, we can choose $s_1 > s_2 + m$, where $m > 0$ is a margin that models the spacing between the fingers.

2. We randomly choose the initial values $\boldsymbol{R}_{1,0}$ and $\boldsymbol{M}_{1,0}$, such that each component is within $[-R_{\max}, R_{\max}]$ and $[-M_{\max}, M_{\max}]$ respectively, where $R_{\max} > 0$ and $M_{\max} > 0$.

3. The sequence is divided into n_s steps (n_s is randomly chosen between 1 and a number $n_{\max}$). In the proposed dataset: $R_{\max} = 0.05\,\mathrm{N}$, $M_{\max} = 1 \times 10^{-4}\,\mathrm{N\,m}$ and $n_{\max} = 3$.

4. Due to non-linearity, for each step, the external forces and bending moments are applied progressively. Classically, identical loading increments are considered, but in this contribution loading increments progressively rotate to follow various loading paths.

Such sequences are presented in Tab. 1.

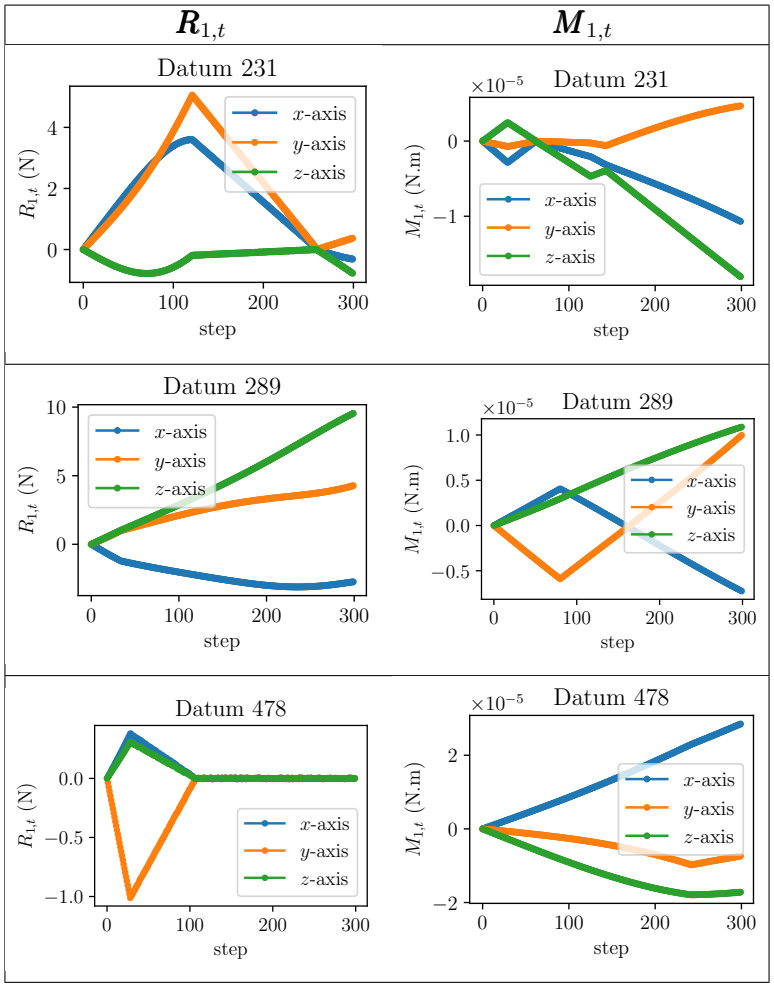


Table 1: Examples of loading sequences from the dataset, where x, y, z are global coordinates.

3.2. Output The free Finite Element software Cast3m [4] was used. For each simulation, the following quantities are computed and are provided in the database (or can be easily retrieve using relationships presented in section 4.2): the displacement $\boldsymbol{U}$ and Euler rotation angles of the cross section $(\varphi_1, \varphi_2, \varphi_3)$ in $\boldsymbol{O}$, the generalized strains: $e = 1/2(J^2 - 1)$, θ^u, θ^v, and the three component curvature vector $\boldsymbol{\chi}$, the internal resultant force $\boldsymbol{R}_0$ and bending moment $\boldsymbol{M}_0$ all defined in section 2.1.2. For elastoplastic simulations, the cumulative plastic strain p_{cum} and plastic generalized deformations: $e_P = 1/2(J_P^2 - 1)$, θ_P^u, θ_P^v, $\boldsymbol{\chi}_P$ are also computed.

To reduce the size of the dataset, these data are projected onto a coarser mesh, which is generated by regularly subsampling the initial mesh. Each simulation corresponding to one sequence takes an average of $3\,\mathrm{min}$ using Cast3m, on a standard laptop.

3.3. Visualization of Results and Analysis

A total of 1000 sequences were generated. An example is presented in Fig. 3. The minimum, maximum, and average values of the outputs are presented in Table 2.

	min	avg	max
u_t (mm)	-34	-1.6	41
u_u (mm)	-43	-0.6	36
u_v (mm)	-37	-0.35	36
e	-2.1	0.03	2.1
θ^u	-1.8	-0.06	1.6
θ^v	-1.8	0.08	1.6
χ_t (mm^{-1})	-3.1×10^{-3}	6.5×10^{-7}	3.4×10^{-3}
χ_u (mm^{-1})	-2.2×10^{-4}	1.2×10^{-6}	2.1×10^{-4}
χ_v (mm^{-1})	-1.9×10^{-4}	-4.0×10^{-6}	2.0×10^{-4}
R_t (N)	-14	0.2	14
R_u (N)	-14	0.07	13
R_v (N)	-12	-0.2	13
M_t (N mm)	-0.24	6.5×10^{-4}	0.33
M_u (N mm)	-0.74	2.1×10^{-4}	0.50
M_v (N mm)	-0.36	-1.4×10^{-3}	0.60

Table 2: Outputs value ranges for an elastic-plastic computation.

4. Car hood dataset

The car hood dataset consists of a set of quasi-static simulations that simulate hammering of the car hood using the classic 3D Cauchy medium. It is in open access [13].

4.1. Input: Mesh and boundary conditions

The original mesh used is a CAD mesh of the hood of a Renault Twizy (Fig. 4a). In order to run accurate finite element simulations with Cast3m, a refined version is constructed using MeshLab (Fig. 4b).

The (X, Y, Z) coordinate system is defined by the principal components of the mesh (in the sense of PCA), so that the width and length of the hood are aligned with the (X, Y) plane. The origin of the coordinate system is located at the centroid of the mesh.

Arbitrarily, the hood is clamped in three zones, corresponding to Dirichlet boundary conditions (zero displacement imposed). They are highlighted in red in Fig. 5

The loading sequences simulate a certain number of hammer blows. It is constructed using the following steps:

1. The number of hammer blows is randomly selected, ranging between 1 and n_h. Here, $n_h = 10$.

2. For each hammer blow, a strike point p is randomly chosen in the (X, Y) plane. The strike area is then defined by the set of triangles located at the intersection between the mesh and a cylinder centered at p with radius R. Here, $R = 0.2\,\mathrm{m}$.

3. A sequence has a determined duration of n_t points. The durations of each blow are then calculated by partitioning $[\![1, n_t]\!]$ into n_h elements. Here, $n_t = 200$.

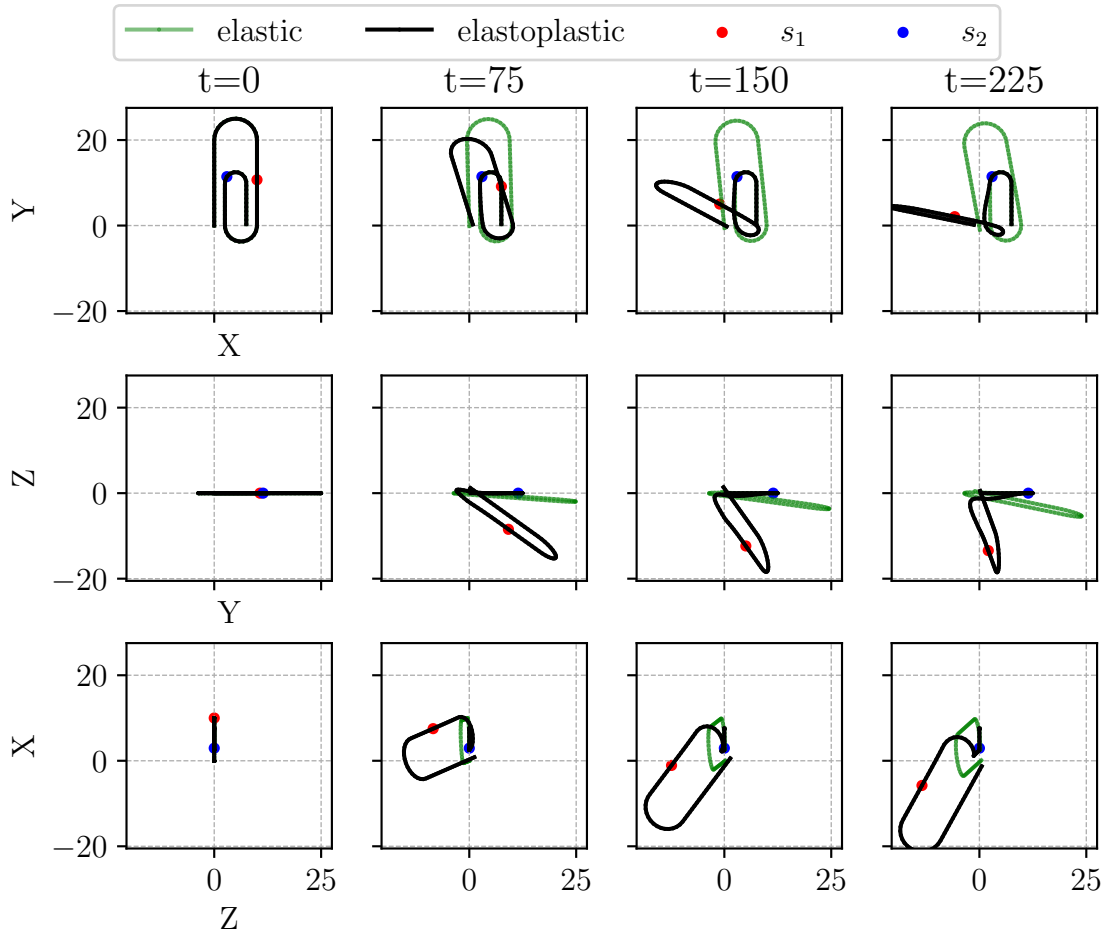
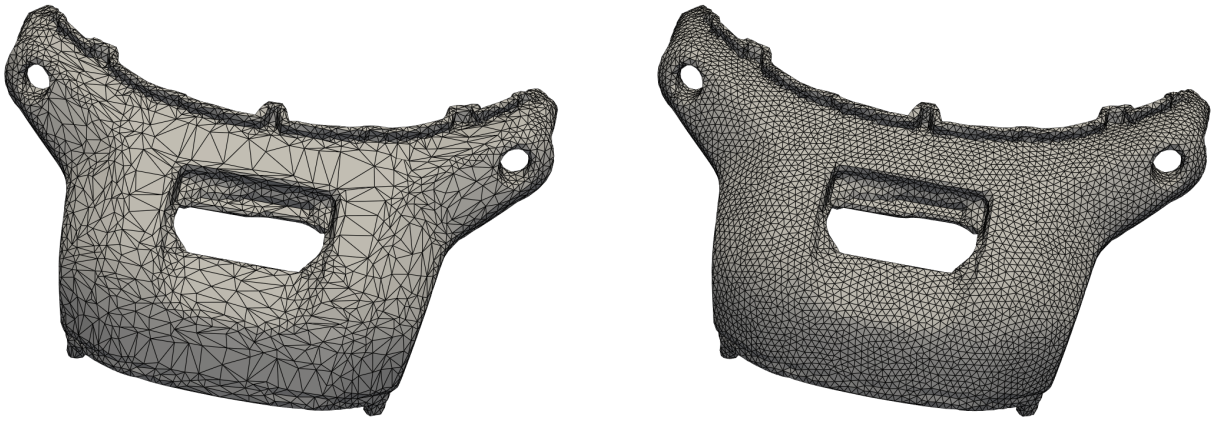


Figure 3: Sample of the paperclip dataset providing strongly nonlinear deformations.



(a) Initial CAD Mesh

(b) Refined mesh

Figure 4: Car hood meshes

4. For each blow, its maximum intensity (peak of the triangle) (f, t) is randomly chosen within $[0, f_m] \times [t_{s,i}, t_{s,f}]$, where $t_{s,i}$ represents the initial step of the blow, and $t_{s,f}$ its final step. In the dataset, $f_m = 400\,\mathrm{Pa}$.

Examples of loading sequences are shown in Fig. 6. In average, a sequence is generated in

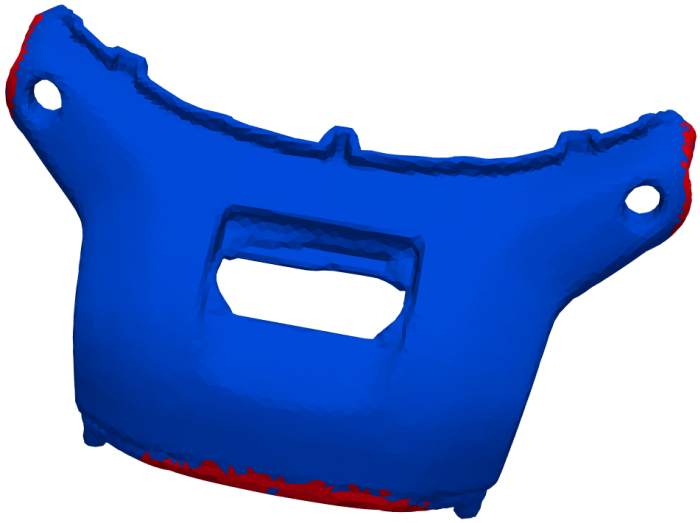


Figure 5: Boundary conditions applied on the hood: Dirichlet condition (zero displacement) in red, Neumann condition (imposed zero force almost everywhere) in blue.

12 min. Examples of paths are shown on Fig. 7.

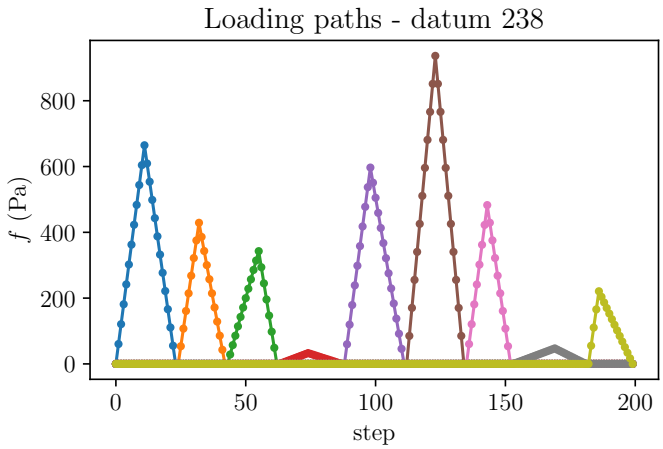


Figure 6: Amplitudes of a few loading sequences. Each color represents a hit.

4.2. Output

The governing equations are detailed in section 2.1.1.

Only displacements $\boldsymbol{u}$, the Cauchy stress tensor $\boldsymbol{\sigma}$ and the cumulative plastic strain p_{cum} are recorded. Indeed, all the other computed quantities (i.e., transformation gradient, elastic and plastic Green-Lagrange strain tensors) can be retrieved only using $\boldsymbol{u}$ and $\boldsymbol{\sigma}$.

Indeed, the transformation $\boldsymbol{\Phi}$ and the transformation gradient $\boldsymbol{F}$ can be computed from $\boldsymbol{u}$. One can also compute $J = \det[\boldsymbol{F}]$.

The total Green-Lagrange strain $\boldsymbol{E} = 1/2(\boldsymbol{F}^\top \cdot \boldsymbol{F} - \boldsymbol{1})$ is then computed. $\boldsymbol{F}_E^2$ can be obtained from $\boldsymbol{\sigma}$ through (2.8), hence the elastic Green-Lagrange tensor $\boldsymbol{E}_E = 1/2\left(\boldsymbol{F}_E^2 - \boldsymbol{1}\right)$ can also computed. Therefore using $\boldsymbol{F} = \boldsymbol{F}_E \cdot \boldsymbol{F}_P$ one can compute $\boldsymbol{F}_P^\top \cdot \boldsymbol{F}_P = \boldsymbol{F}^\top \cdot \left(\boldsymbol{F}_E^2\right)^{-1} \cdot \boldsymbol{F}$ hence the plastic Green-Lagrange tensor $\boldsymbol{E}_P = 1/2(\boldsymbol{F}_P^\top \cdot \boldsymbol{F}_P - \boldsymbol{1})$ is retrieved. From this simple observation, it becomes clear that a learning algorithm focused on stresses and displacements is sufficient to recover

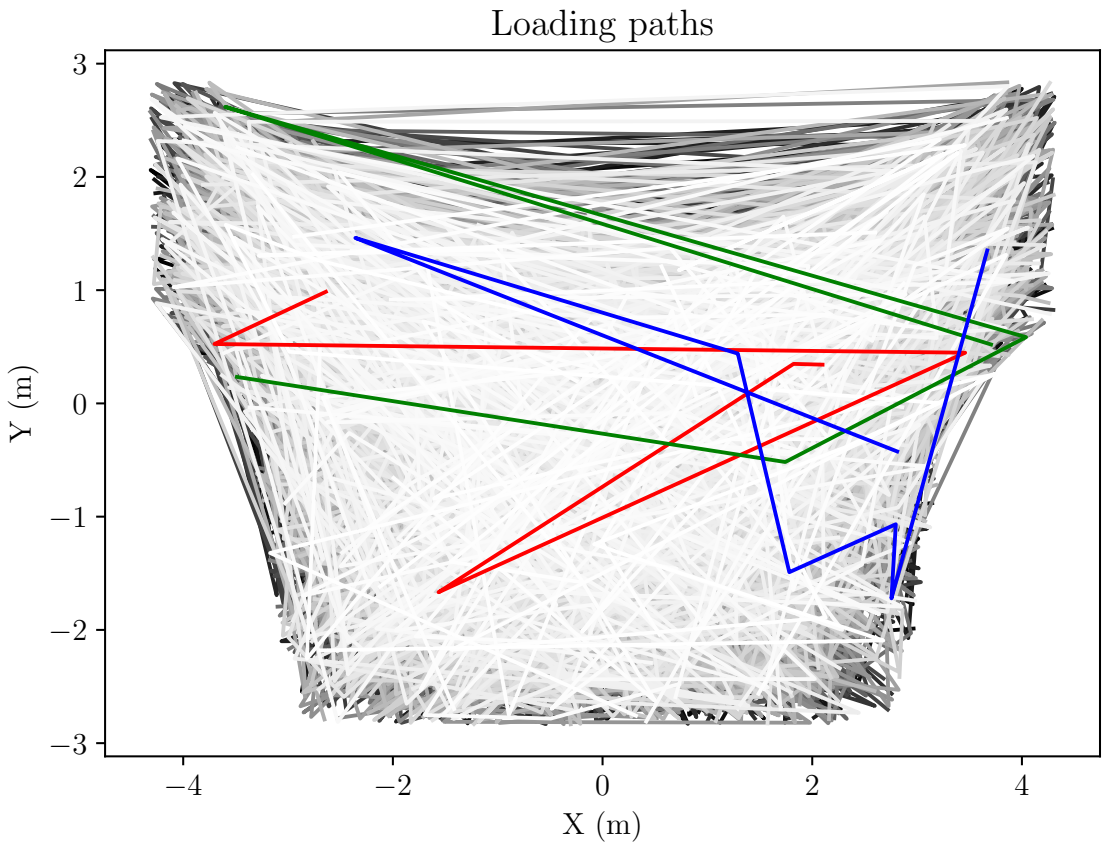


Figure 7: Examples of loading paths. Each colored line represents an example path taken by the hammer in a sequence from the database. All sequences in the database are represented in gray scale.

all the essential quantities of the problem—but whether also incorporating other fields during the training process or in the architecture may help the training is an open problem.

All these results are projected onto a coarser mesh to reduce the database size. This coarser mesh was generated using the quadric edge decimation algorithm in MeshLab [5], so that the points of the coarse mesh are shared with those of the fine mesh, to ensure correct subsampling. Some examples of coarse meshes are shown in Fig. 8.

(a) decimation = 0.05 **(b)** decimation = 0.25 **(c)** decimation = 0.5 **(d)** decimation = 0.75 **(e)** decimation = 1.0

Figure 8: Different submeshes extracted with Quadric edge decimation.

4.3. Visualization of Results and Analysis

A total of 1000 sequences were generated. A few examples are visible on Fig. 9. The minimum, maximum, and average values of the inputs and outputs are presented in Table 3.

5. Conclusion

In conclusion, we believe that this dataset will be valuable to the Scientific ML community for several reasons:

	min	avg	max
u_X (mm)	-3040	-2	2433
u_Y (mm)	-1458	-4	3264
u_Z (mm)	-5000	-134	599
σ_{xx} (Pa)	-33×10^9	-26×10^6	16×10^9
σ_{yy} (Pa)	-33×10^9	-20×10^6	20×10^9
σ_{zz} (Pa)	-33×10^9	-11×10^6	18×10^9
σ_{xy} (Pa)	-3.6×10^9	1.8×10^6	3.5×10^9
σ_{xz} (Pa)	-2.4×10^9	0.1×10^6	4.4×10^9
σ_{yz} (Pa)	-5.4×10^9	-1.4×10^6	2.4×10^9
p_{cum}	0.0	0.01	2.9

Table 3: Samples of the second database: deformation of a car hood in response to multiple hammer taps.

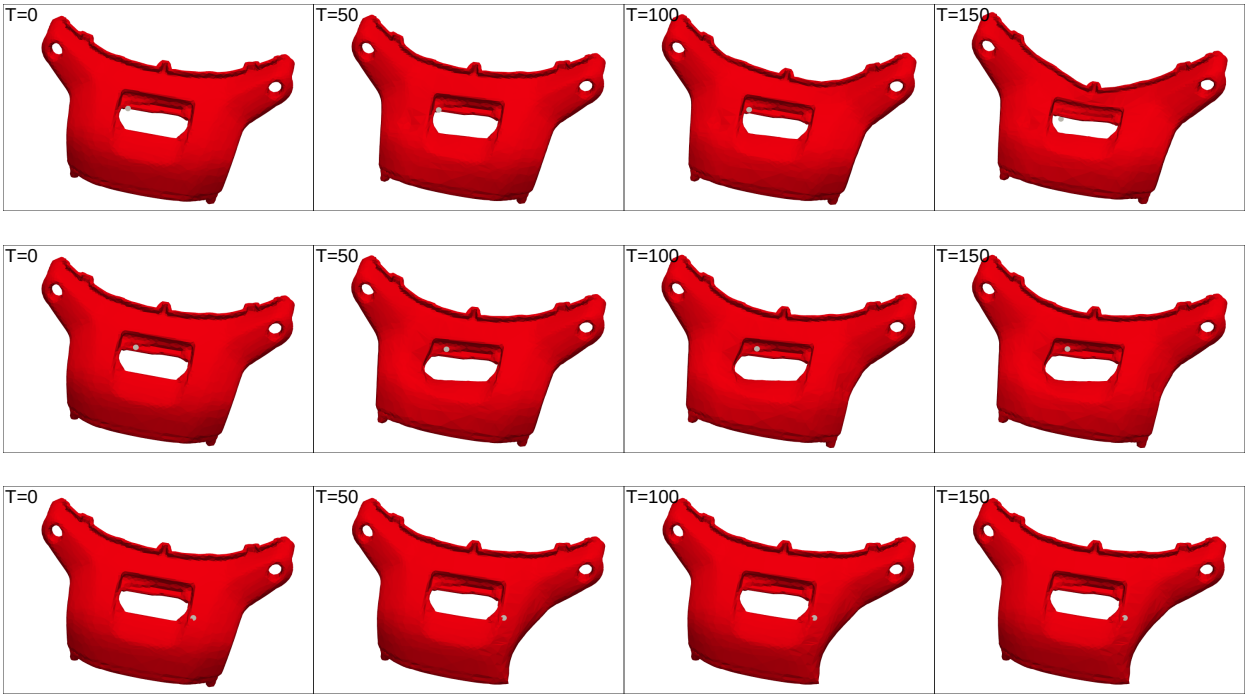


Figure 9: Some example of generated data.

- **Training material**: This database is the first of its kind to contain such precise simulations in a quantity suitable for training data-driven models, particularly neural networks. It models physically complex problems that are challenging and can guide the development of new architectures tailored to Scientific ML.
- **Benchmark**: One of the main limitations of previous works was the lack of a reference dataset for benchmarking proposed architectures, especially in the field of finite strain elasto-plasticity. We believe that this dataset can serve as a benchmark for future architectures.
- **Speeding-up Simulations**: This database paves the way for significant acceleration in simulation times, with numerous applications in engineering, such as real-time interactive simulations and optimization of manufacturing processes.

References

[1] ARORA, Rajat, Pratik KAKKAR, Biswadip DEY, Amit CHAKRABORTY. *Physics-informed neural networks for modeling rate- and temperature-dependent plasticity*. 23, 2022. arXiv: 2201.08363.

[2] BENATO, Lisa et al. *Shared data and algorithms for deep learning in fundamental physics*. Computing and Software for Big Science 6(1):9, 2022. arXiv: 2107.00656.

[3] CASENAVE, Fabien, Brian STABER, Xavier ROYNARD. *Mmpg: a mesh morphing gaussian process-based machine learning method for regression of physical problems under nonparametrized geometrical variability*. Advances in Neural Information Processing Systems 36, 2024. arXiv: 2305.12871.

[4] CEA. *Cast3m*. http://www-cast3m.cea.fr/. 2020.

[5] CIGNONI, Paolo, Marco CALLIERI, Massimiliano CORSINI, Matteo DELLEPIANE, Fabio GANOVELLI, Guido RANZUGLIA. "MeshLab: an Open-Source Mesh Processing Tool". Eurographics Italian chapter conference, 2008.

[6] HABIB, Ahed, Ausamah AL HOURI, M Talha JUNAID, Samer BARAKAT. "A systematic and bibliometric review on physics-based neural networks applications as a solution for structural engineering partial differential equations". *Structures*. Volume 69. Elsevier. 2024: 107361.

[7] HAGHIGHAT, Ehsan, Maziar RAISSI, Adrian MOURE, Hector GOMEZ, Ruben JUANES. *A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics*. Computer Methods in Applied Mechanics and Engineering 379:113741, 2021.

[8] HASHASH, YMA, Sungmoon JUNG, Jamshid GHABOUSSI. *Numerical implementation of a neural network based material model in finite element analysis*. International Journal for numerical methods in engineering 59(7):989–1005, 2004.

[9] HERRMANN, Leon, Stefan KOLLMANNSBERGER. *Deep learning in computational mechanics: a review*. Computational Mechanics:1–51, 2024.

[10] HUANG, Daniel Z, Kailai XU, Charbel FARHAT, Eric DARVE. *Learning constitutive relations from indirect observations using deep neural networks*. Journal of Computational Physics 416:109491, 2020. arXiv: 1905.12530.

[11] HUANG, Zhiao, Yuanming HU, Tao DU, Siyuan ZHOU, Hao SU, Joshua B. TENENBAUM, Chuang GAN. *PlasticineLab: a soft-body manipulation benchmark with differentiable physics*. 7, 2021. arXiv: 2104.03311.

[12] KIM, Byungsoo, Vinicius C AZEVEDO, Nils THUEREY, Theodore KIM, Markus GROSS, Barbara SOLENTHALER. "Deep fluids: a generative network for parameterized fluid simulations". *Computer graphics forum*. Volume 38. Wiley Online Library. 2019: 59–70. arXiv: 1806.02071.

[13] LESUEUR, Louis, Anders THORIN, Daniel WEISZ-PATRAULT. *A benchmark for elasto-plasticity in finite strain*. Version V1. 2025. [10.57745/HQJMYA].

[14] LIENEN, Marten, David LÜDKE, Jan HANSEN-PALMUS, Stephan GÜNNEMANN. *From zero to turbulence: generative modeling for 3d flow simulation*. arXiv preprint arXiv:2306.01776, 2023. arXiv: 2306.01776.

[15] MASI, Filippo, Ioannis STEFANOU, Paolo VANNUCCI, Victor MAFFI-BERTHIER. *Thermodynamics-based artificial neural networks for constitutive modeling*. Journal of the Mechanics and Physics of Solids 147:104277, 2021. arXiv: 2005.12183.

[16] MENDIZABAL, Andrea, Pablo MÁRQUEZ-NEILA, Stéphane COTIN. *Simulation of hyperelastic materials in real-time using deep learning*. Medical image analysis 59:101569, 2020. arXiv: 1904.06197.

[17] RABBANI, Arash, Masoud BABAEI, Reza SHAMS, Ying DA WANG, Traiwit CHUNG. *Deepore: a deep learning workflow for rapid and comprehensive characterization of porous materials*. Advances in Water Resources 146:103787, 2020. arXiv: 2005.03759.

[18] RAISSI, Maziar, Paris PERDIKARIS, George 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 378:686–707, 2019.

[19] TAKAMOTO, Makoto, Timothy PRADITIA, Raphael LEITERITZ, Daniel MACKINLAY, Francesco ALESIANI, Dirk PFLÜGER, Mathias NIEPERT. "Pdebench: an extensive benchmark for scientific machine learning". *Advances in Neural Information Processing Systems*. Edited by S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, A. Oh. Volume 35. Curran Associates, Inc., 2022: 1596–1611. arXiv: 2210.07182.

[20] VLASSIS, Nikolaos N, WaiChing SUN. *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, 2021. arXiv: 2010.11265.

[21] ZHONG, Weiheng, Hadi MEIDANI. *Physics-informed discretization-independent deep compositional operator network*. Computer Methods in Applied Mechanics and Engineering 431:117274, 2024. arXiv: 2404.13646.