J. Mech. Phys. Solids 166 (2022) 104927

Contents lists available at ScienceDirect

Journal of the Mechanics and Physics of Solids

journal homepage: www.elsevier.com/locate/jmps

Manifold embedding data-driven mechanics

Bahador Bahmani, WaiChing Sun *

Department of Civil Engineering and Engineering Mechanics, Columbia University, 614 SW Mudd, Mail Code: 4709, New York, NY 10027, United States of America

ARTICLE INFO

Keywords:
Data-driven mechanics
Manifold learning
Geodesic
Constitutive manifold

ABSTRACT

This article introduces a manifold embedding data-driven paradigm to solve small- and finite-strain elasticity problems without a conventional constitutive law. This formulation follows the classical data-driven paradigm by seeking the solution that obeys the balance of linear momentum and compatibility conditions while remaining consistent with the material data by minimizing a distance measure. Our key point of departure is the introduction of a global manifold embedding as a means to learn the geometrical trend of the constitutive data mathematically represented by a smooth manifold. By training an invertible neural network to embed the data of an underlying constitutive manifold onto a Euclidean space, we reformulate the local distance-minimization problem that requires a computationally intensive combinatorial search to identify the optimal data points closest to the conservation law with a cost-efficient projection step. Meanwhile, numerical experiments performed on path-independent elastic materials of different material symmetries suggest that the geometrical inductive bias learned by the neural network is helpful to ensure more consistent predictions when dealing with data sets of limited sizes or those with missing data.

1. Introduction

Conventional computer simulations of mechanics problems often employ discretizations to convert boundary value problems into systems of equations and obtain the discretized solution from a solver. The boundary value problems are often split into two components, i.e., the constraints for the solution in the space–time domain that often derives from balance principles, and constitutive models that relate physical quantities derived from the solutions based on a combination of phenomenological observations and constraints derived from thermodynamics. Examples of these constitutive models are abundant in the literature of the last few centuries (Timoshenko, 1983) even before the development of numerical methods for boundary value problems. Recently, the interest in deep learning has sparked a renewed interest in training neural networks to generate constitutive responses, (an idea that can track back to the last AI winter Ghaboussi et al., 1991) or replace classical solvers to generate numerical solutions of boundary value problems (Raissi et al., 2019).

Kirchdoerfer and Ortiz (2016) introduce a new paradigm in which the modeling of constitutive responses can be bypassed. Instead, a solver can be established such that one may find the data points in a material database that is closest to the manifold that obeys the conservation principle. The upshot of this data-driven approach is that it does not require finding the functional forms of the constitutive laws which could be an ambiguous, ad-hoc, and time-consuming process (Cranmer et al., 2020). Furthermore, bypassing the use of constitutive laws also eliminates the calibration process. For materials that exhibit complex physics and/or do not possess symmetry, it becomes necessary to introduce more material parameters in order to predict the material responses

* Corresponding author.
E-mail address: wsun@columbia.edu (W. Sun).

https://doi.org/10.1016/j.jmps.2022.104927
Received 17 December 2021; Received in revised form 2 May 2022; Accepted 4 May 2022
Available online 16 May 2022

with sufficient precision (Ehlers and Scholz, 2007; Wang et al., 2016; Liu et al., 2016). On-the-fly multiscale calculations, such as hierarchical upscaling methods (e.g., FEM^2) (Wang and Sun, 2016; Matouš et al., 2017) or concurrent multiscale domain coupling methods (Hughes et al., 1998; Sun and Mota, 2014; Sun et al., 2017; Badia et al., 2008) may eliminate the need to compose constitutive laws that captures the responses of the effective medium through computational homogenization. However, the computational cost of these multiscale techniques remains a major technical barrier despite years of research progress (Wang and Sun, 2018; Karapiperis et al., 2021).

The removal of the constitutive laws, therefore, brings in a new paradigm where the universal principles are enforced and the solutions are sought by finding the data points that are closest to the conservation manifold. However, one important question central to the success of this paradigm is the following—how to properly measure the "distance" between the material database and conservation manifold?

Interestingly, there have been very few researches on the choice of the distance measure for the data-driven/model-free approach. In data-driven mechanics research, such as Kirchdoerfer and Ortiz (2016, 2018), the independent components of the input–output pairs (e.g., the strain–stress pair) of the constitutive laws are often used to constitute a product space, and the measure of distance relies on the equipped weighted energy norms. This idea is utilized not only for elasticity problems but also for multiphysics (e.g., Bahmani and Sun (2021a) and higher-order continua (e.g., Karapiperis et al. (2021)). Noticeably, Leygue et al. (2018) compare simulation results obtained from different choices of distance measure and demonstrate a dependence between the energy-minimized solutions and the choice of the measure (see Fig. 6 of Leygue et al. (2018)). The underlying issue is not necessarily the choice of the distance measure but the discrete nature of data point clouds. Without any prior assumption on the intrinsic properties of the geometry, defining a unique length or distance between two data points becomes impossible. For instance, the distance between two points on a flat surface and that on a curved surface are different. Assuming that the data are on a Euclidean space such as $\mathbb{R}^{12}$ for stress–strain data, could be feasible when the data distributed in the phase space is sufficiently dense or the data set itself is close to linear (and hence the difference between the length measured from the local tangential space and the constitutive manifold is minor). However, these conditions are not necessarily realistic.

There have already been efforts that introduce resolutions to circumvent this spurious dependence. A common approach is to introduce *locally linear embedding*. In Ibañez et al. (2017), for instance, the data points used to measure the distance are factored by a weighting function such that each local linear patch around a data point can be mapped onto a lower-dimensional embedding space. In He and Chen (2020), locally convex material manifolds are also introduced to achieve locally linear exactness with dimensional reductions. Meanwhile, Kanno (2021) introduce a kernel-based method to extract the manifold of the constitutive responses *globally*. By extracting the global constitutive manifold where all the data points belong, a metric can be established via embedding. However, like the original data-driven approach where the computational cost may become significantly higher with an increasing number of data points, the complexity of the kernel method scales with the number of data points and hence can be expensive when handling a sufficiently large database even the dimensions of the data is relatively low (e.g., 1D constitutive laws).

In this paper, we introduce a neural network based **global** manifold embedding technique such that a proper distance measure corresponding to the manifold of the constitutive responses can be introduced for the data-driven mechanics simulations for path-independent materials. Our focus is, therefore, on the constitutive laws that exhibit no path dependence (e.g., elasticity and nonlinear heat transfer). Applying the global embedding technique for path-dependent materials is potentially feasible but is out of the scope of this paper.

To achieve this goal, we consider the cases where we have collected data points from experiments or direct numerical simulations. We then train an invertible neural network such that it may map all data points on the constitutive manifold of path-independent materials onto a Euclidean space equipped with a Euclidean norm.

As illustrated in Fig. 1, The distance-minimization algorithm iterates between two steps: first, a global-to-local projection $\mathcal{P}^{G\mapsto L}$ from the equilibrium state (see Definition 8) $\boldsymbol{z} \in \mathcal{C}$ (Step 1) in (a) to the corresponding material state (see Definition 6) $\boldsymbol{z}^*$ (Step 2). Then, a local-to-global projection $\mathcal{P}^{L\mapsto G}$ from material states to an updated equilibrium state is performed (Step 3). In the proposed embedded method, the local step $\hat{\mathcal{P}}^{G\mapsto L}$ in Kirchdoerfer and Ortiz (2016) is modified. We project the equilibrium state not directly onto the material database but onto the closest point on a hyperplane called mapped material database $\hat{\boldsymbol{z}} \in \hat{D}$ (the blue plane in Fig. 1). We then use the established mapping operator obtained from the invertible neural network $\mathcal{G} = \mathcal{F}^{-1}$ to convert the mapped state $\hat{\boldsymbol{z}} \in \hat{D}$ in the mapped material database back to the original material database (the gray manifold in Fig. 1), i.e. $\boldsymbol{z}^* \in D$ such that $\mathcal{P}^{G\mapsto L} = \mathcal{G}\circ\mathbb{P}\circ\mathcal{F}$. $\mathbb{P}$ is a linear operator that projects the intermediate state $\boldsymbol{z}^{\#}$ onto the closest point (with respect to the Euclidean norm) on the hyperplane, i.e., $\hat{\boldsymbol{z}}$.

This treatment is introduced to serve two purposes. First, we would like to improve the robustness of the data-driven predictions when data are not sufficiently dense in the phase space. Second, we want to remove the computational bottleneck due to the combinatorial optimization necessary for the closest point search in the data-driven paradigm. Our numerical experiments indicate that the global embedding technique may improve the robustness of the data-driven approach such that (1) it still functions reasonably well when data is sparse or not distributed with even density and (2) remains sufficiently efficient when data is abundant. Such an improvement is particularly important for high-consequence simulations where the robustness of the predictions becomes critical.

The rest of the paper is organized as follows. We will first briefly review the data-driven paradigm and outline the current state-of-the-art distance minimization techniques (Section 2). The data-driven paradigm employs global and local minimization problems sequentially to locate the data points (on the constitutive manifold) closest to the conservation manifold. Our work employs a similar design, but the local minimization problem is reformulated in the mapped Euclidean space. To make the presentation self-contained, Section 3 is provided to outline the global optimization step for the data-driven paradigm in both the small and finite

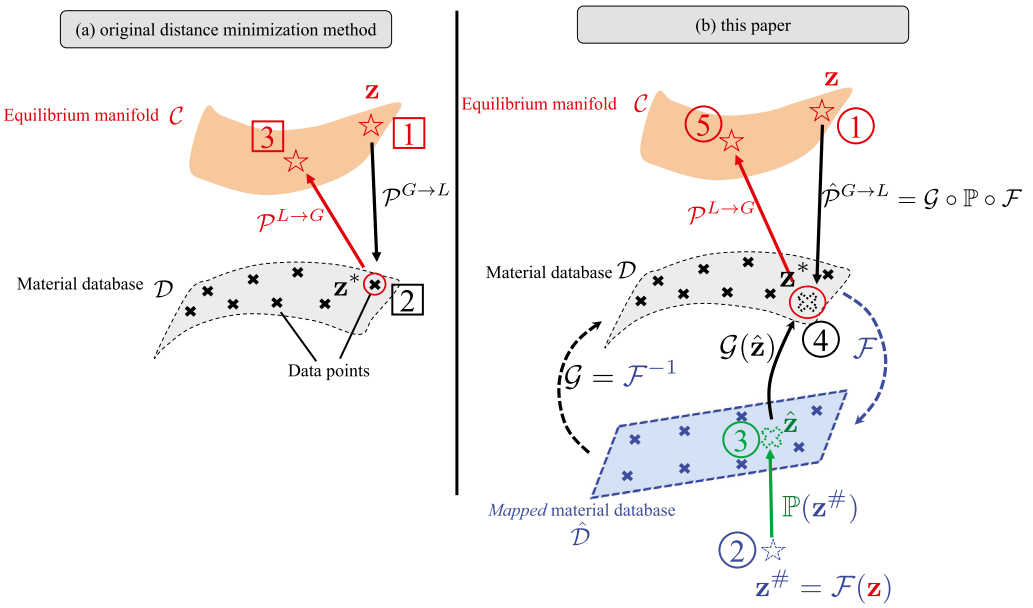


Fig. 1. A comparison between (a) the original distance minimization method (Kirchdoerfer and Ortiz, 2016) and (b) the introduced method in this paper. See Appendix B for terminology.

strain regimes. The versatility of the proposed framework for large deformation problems is provided in Section 3.1. We then revisit the local minimization problem in Section 4 and introduce the invertible neural network approach in Section 5 as the means to improve the efficiency of the local search (by replacing the combinatorial optimization with a projection performed in the mapped Euclidean space). Numerical examples are conducted to illustrate the ideas, verify the implementation and demonstrate and compare the performance of the manifold learning approach with the classical Euclidean counterpart in Section 6. A few key observations are summarized in Section 7. Unless otherwise specified, we assume that all constitutive manifolds studied in this paper are Riemannian.

In Appendix A, the manifold embedding and the manifold learning techniques commonly used for regression, classification, and computer vision problems are reviewed. Other related works on predictions made on point cloud data in manifold and metric space are outlined in Appendix A.1. Finally, we have also included the terminology commonly used in the data-driven paradigm literature for readers who might not be familiar with the data-driven approach or concepts essential to the key ideas (e.g., hyperplane) but could have been otherwise less well-known among mechanicians are provided in Appendix B.

2. Literature review on data-driven/model-free solid mechanics

The model-free distance-minimization method (cf. Kirchdoerfer and Ortiz (2016)) is a new paradigm that directly incorporates material databases as the replacements of explicit surrogate constitutive laws into the numerical simulations in a variationally consistent manner. However, due to the minimal constraints imposed to generate numerical solutions, there must be sufficient data to ensure the distance between the data points and the conversation manifold are sufficiently close. Hence, the original formulation may exhibit sensitivity to noises and outliers (Kirchdoerfer and Ortiz, 2017; Ibañez et al., 2017; He and Chen, 2020; Bahmani and Sun, 2021a; Kanno, 2021). In Ibañez et al. (2017), the manifolds of the constitutive responses are constructed in an unsupervised learning setting, and a locally linear embedding method proposed in Roweis and Saul (2000) is used to recover the low dimensional characteristics of a material database locally.

Meanwhile, the robustness against outliers can be improved by a clustering scheme based on the maximum-entropy estimation (Kirchdoerfer and Ortiz, 2017). Kanno (2018a,b) formulate the distance minimization method based on a local kernel regression method (in an offline manner) to enhance the robustness in the noisy-data and limited-data scenarios. He and Chen (2020) construct a locally linear subspace for points in a small neighborhood surrounding each material point (in an online manner). They perform convex interpolation on the constructed subspace to increase the robustness against noisy data.

However, since the locally linear embedding technique requires collecting the neighborhood data points to reconstruct the local linear space, the quality of the predictions is sensitive to how the data points are distributed. An unevenly distributed data set may lead to the constructed local space with an uneven resolution, while data set with limited data points may not be able to capture the topology of the manifold precisely.

Kanno (2021) introduces a new formulation that constructs a global embedding parameterized by a kernel method in an offline manner. One drawback of kernel methods is the scalability issue with respect to the data size (Bishop, 2006; Lan et al., 2019). Since classical kernel methods often operate on dense kernel matrices that scale cubically with the data size, a database with

many instances may therefore slow down the kernel method and make it less effective for big data applications. Eggersmann et al. (2021b) use a tensor-voting interpolation method locally inside a ball around a material point (in an offline manner) to increase the performance in the limited data regime and enhance the robustness against noise. He et al. (2021) introduce a nonlinear dimensionality reduction method based on the autoencoder architecture to perform noise-filtering.

The distance-minimization paradigm has also been extended for different applications such as elastodynamics (Kirchdoerfer and Ortiz, 2018), finite-strain elasticity (Nguyen and Keip, 2018), plasticity (Eggersmann et al., 2019), fracture mechanics (Carrara et al., 2020), geometrically exact beam theory (Gebhardt et al., 2020), poroelasticity (Bahmani and Sun, 2021a), and micro-polar continuum (Karapiperis et al., 2021). Its scalability issue with respect to the data size is addressed in Bahmani and Sun (2021a) where an isometric projection is introduced to efficiently organize the database via the ***k*-d** tree data structure that provides a logarithmic time complexity (in average) for nearest neighbor search. Other efficient data structures are also explored in Eggersmann et al. (2021a).

3. Data-driven framework

For completeness, we briefly summarize the distance-minimization data-driven paradigm (Kirchdoerfer and Ortiz, 2016). In this approach, the classical solver for a mechanics problem (or generally speaking, a boundary value problem) is reformulated as a minimization problem that finds an admissible solution satisfying the conservation laws with the minimum distance to a given database populated by discrete point clouds in the phase space. Due to its minimal model assumption about the constitutive behavior, it is known as a model-free method and categorized as a non-parametric model. As shown in Fig. 1(a), the classical model-free paradigms often consist of two iterative algorithmic steps operated in the conservation (governing equation) and constitutive manifold respectively where the equilibrium states are first determined and projected onto the material data. In the current work, to simplify the formulation and avoid the usage of a nonlinear solver, the search of the equilibrium states is based on the energy norm originating from Kirchdoerfer and Ortiz (2016) whereas material data identification is conducted in the embedded Euclidean space of the constitutive manifold.

To obtain the equilibrium state, we consider pairs of strain–stresses $\{\boldsymbol{z}_a^* = (\boldsymbol{\epsilon}_a^*, \boldsymbol{\sigma}_a^*)\}_{a=1}^{N_{\text{quad}}}$ at N_{quad} quadrature points for a spatial domain discretized by finite elements. To find the equilibrium state that has the minimum distance to the given strain–stress 2-tuple, the following constrained optimization problem is solved,

$$
\begin{aligned}
&\arg\min_{\boldsymbol{z}} \int_{\Omega} d_M^2(\boldsymbol{z}, \boldsymbol{z}^*)\, d\Omega,\\
&\text{subject to:} \qquad \nabla^x \cdot \boldsymbol{\sigma} + \boldsymbol{b} = \boldsymbol{0} \quad \text{in } \Omega,\\
&\qquad\qquad\quad \boldsymbol{\sigma}\cdot\boldsymbol{n} = \bar{\boldsymbol{t}} \quad \text{on } \Gamma_{\sigma},\\
&\qquad\qquad\quad \boldsymbol{u} = \bar{\boldsymbol{u}} \quad \text{on } \Gamma_{u},\\
&\qquad\qquad\quad \boldsymbol{\epsilon} = \frac{1}{2}\{\nabla^x \boldsymbol{u} + (\nabla^x \boldsymbol{u})^T\} \quad \text{in } \Omega,
\end{aligned}
$$

where $\boldsymbol{z} = (\boldsymbol{\epsilon}, \boldsymbol{\sigma})$, $\boldsymbol{\epsilon} \subset \mathbb{R}^m$ is the strain tensor, $\boldsymbol{\sigma} \in \mathbb{R}^m$ is the stress tensor, $d_M(\cdot,\cdot)$ is the distance function, $\boldsymbol{b} \in \mathbb{R}^{N_{\text{dim}}}$ is the body force vector, $\boldsymbol{u} \in \mathbb{R}^{N_{\text{dim}}}$ is the displacement vector, $\boldsymbol{n}$ is the normal vector to the external boundary, $\bar{\boldsymbol{t}}$ is the prescribed traction on the boundary, $\bar{\boldsymbol{u}}$ is the prescribed displacement on the boundary, $\Omega \subset \mathbb{R}^{N_{\text{dim}}}$ is the volumetric domain, $\Gamma_{\sigma} \subset \partial\Omega$ is the Dirichlet boundary surface, $\Gamma_{\sigma} \subset \partial\Omega$ is the Neumann boundary surface, $N_{\text{dim}} = \{1, 2, 3\}$ is the space dimensionality, and $m = N_{\text{dim}}(N_{\text{dim}}+1)/2$ is the number of free components in the second-order symmetric tensor. The material state at each quadrature point $\boldsymbol{z}_a^* \subset \mathcal{D}$ where $\mathcal{D}$ is the product space spanned by N_{data} data points $\mathcal{D} = \bigtimes_{b=1}^{N_{\text{data}}} (\boldsymbol{\epsilon}_b^*, \boldsymbol{\sigma}_b^*)$. The search of the equilibrium state in the mechanical phase space $\mathcal{Z} \subset \mathbb{R}^m \times \mathbb{R}^m$ is conducted by minimizing the following norm $|\boldsymbol{z}|_M$:

$$
|\boldsymbol{z}|_M^2 = \frac{1}{2}\boldsymbol{\epsilon} : \mathbb{C} : \boldsymbol{\epsilon} + \frac{1}{2}\boldsymbol{\sigma} : \mathbb{S} : \boldsymbol{\sigma}, \tag{1}
$$

where $\mathbb{C}$ and $\mathbb{S}$ are two fourth-order symmetric positive definite tensors. As suggested in Kirchdoerfer and Ortiz (2016) and He and Chen (2020), $\mathbb{S} = \mathbb{C}^{-1}$ is used herein. According to the introduced norm, the distance function reads as follows:

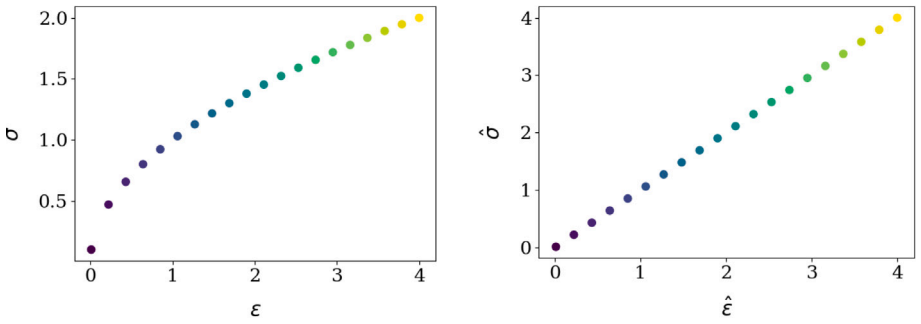
$$
d_M(\boldsymbol{z}, \boldsymbol{z}^*) = |\boldsymbol{z} - \boldsymbol{z}^*|_M = |(\boldsymbol{\epsilon} - \boldsymbol{\epsilon}^*, \boldsymbol{\sigma} - \boldsymbol{\sigma}^*)|_M \tag{2}
$$

The physical constraints listed in the problem statement are enforced via the Lagrange multipliers. Then, following the conventional calculus of variations procedure, we obtain the following Euler–Lagrange equations (Nguyen et al., 2020; He and Chen, 2020; Bahmani and Sun, 2021a):

$$
\mathcal{R}^u = \int_{\Omega} \delta\boldsymbol{u} \cdot \frac{\partial \boldsymbol{\epsilon}(\boldsymbol{u})}{\partial \boldsymbol{u}} : \mathbb{C} : (\boldsymbol{\epsilon} - \boldsymbol{\epsilon}^*)\, d\Omega = 0, \tag{3}
$$

$$
\mathcal{R}^{\beta} = -\int_{\Omega} \nabla^x\, \delta\boldsymbol{\beta} : \boldsymbol{\sigma} d\Omega + \int_{\Omega} \delta\boldsymbol{\beta}\cdot\boldsymbol{b}\, d\Gamma + \int_{\Gamma_{\sigma}} \delta\boldsymbol{\beta}\cdot\bar{\boldsymbol{t}}\, d\Gamma = 0, \tag{4}
$$

$$
\mathcal{R}^{\sigma} = \int_{\Omega} \delta\boldsymbol{\sigma} : [\mathbb{S} : (\boldsymbol{\sigma} - \boldsymbol{\sigma}^*) - \nabla^x\, \boldsymbol{\beta}]\, d\Omega = 0, \tag{5}
$$



(a) material data points **(b)** mapped material data points

Fig. 2. (a) synthesized database by $\sigma = \sqrt{\epsilon}$ with 20 data points that are generated by the regular sampling along the strain axis. (b) mapped database to a vector space by the invertible neural network. Colors show the data point number.

where $\boldsymbol{\beta} \in \mathbb{R}^{N_{\text{dim}}}$ is the Lagrange multiplier to enforce the balance of linear momentum, and δ indicates an arbitrary admissible variation, i.e., $\delta \boldsymbol{u} = \delta \boldsymbol{\beta} = \boldsymbol{0}$ on the Dirichlet boundary surface. Here, we reduce the number of independent fields by a local (point-wise) satisfaction of the last equation via:

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}^* + \mathbb{S}^{-1} : \nabla^x \boldsymbol{\beta} \quad \text{in } \Omega. \tag{6}$$

Given the material states $\{\boldsymbol{z}_a^*\}_{a=1}^{N_{\text{quad}}}$ where $\boldsymbol{z}_a^* \in \mathcal{D}$, the admissible equilibrium states $\{\boldsymbol{z}_a\}_{a=1}^{N_{\text{quad}}}$ at each quadrature point can be obtained after solving the above linear, decoupled system of equations. This step can be considered, geometrically, as a global projection of material states to the conservation law manifold, which is called *global* minimization or projection; see $\mathcal{P}^{L \mapsto G}$ in Fig. 1(a). However, we do not know the optimal distribution of the material states *a priori*. The distance-minimization method iterates between the space spanned by the data points and the conservation laws manifold until the optimal solution is found. After updating the equilibrium states at each quadrature point, the new material states should be updated accordingly such that they have a minimum total distance to the equilibrium states. This step is performed locally at each quadrature point to project the equilibrium state onto the closest data point and is called *local* minimization or projection; see $\mathcal{P}^{G \mapsto L}$ in Fig. 1(a). The corresponding data-driven algorithm is provided in Algorithm 1.

Algorithm 1 Data-driven small-strain solver

Input: Database $\{(\boldsymbol{\epsilon}_b^*, \boldsymbol{\sigma}_b^*)\}_{b=1}^{N_{\text{data}}}$, and numerical parameters $\mathbb{C}$, $\mathbb{S}$
Randomly initialize the material state at quadrature points $\{(\boldsymbol{\epsilon}_a^*, \boldsymbol{\sigma}_a^*)\}_{a=1}^{N_{\text{quad}}}$ from the database
while *not converged* **do**
    Global projection by solving Eqs. (3) and (4). ▷ $\mathcal{P}^{L \mapsto G}$
    for $a = 1 : N_{\text{quad}}$ **do** ▷ local operations $\mathcal{P}^{G \mapsto L}$
        Update equilibrium state $(\boldsymbol{\epsilon}_a, \boldsymbol{\sigma}_a)$ by Eq. (6)
        Update material state $(\boldsymbol{\epsilon}_b^*, \boldsymbol{\sigma}_b^*)$ by the local projection of $(\boldsymbol{\epsilon}_a, \boldsymbol{\sigma}_a)$ via Eq. (7)

Here, we generalize the local minimization step of the original distance minimization method as follows:

$$\underset{\boldsymbol{z}^* \in \mathcal{D}}{\arg\min} \quad d_{\mathcal{D}}^2(\boldsymbol{z}_a, \boldsymbol{z}^*), \tag{7}$$

where $d_{\mathcal{D}}$ is an appropriate metric to measure the distance in the data space $\mathcal{D}$. In the original distance minimization setup (cf. Kirchdoerfer and Ortiz (2016)), we have $d_{\mathcal{D}}(\cdot,\cdot) \equiv d_{\mathcal{M}}(\cdot,\cdot)$, and the local minimization step is the nearest neighbor search in the database to find the closest data point to a given equilibrium state $\boldsymbol{z}_a$. Here we hypothesize that an efficient metric to measure the distance in each manifold could be different because the conservation and constitutive manifolds may possess different geometrical structures.

3.1. Extension to the finite-strain regime

This section extends the small-strain data-driven formulation to simulate elastic continua undergoing finite deformation. The procedure for the local projection step in the finite-strain regime is conceptually similar. The only required change is the choices of the strain and stress measures that constitute the data set and the constraints used in the global project step. For completeness, we provide the formulation of the global projection for finite-strain elasticity problems in this section.

A straightforward extension of the data-driven framework in finite-strain theory can be established for the following metric in the phase space of Green–Lagrange strain $\boldsymbol{E}$ and second Piola–Kirchhoff $\boldsymbol{S}$ tensors as follows (Nguyen and Keip, 2018; He et al., 2021):

$$|\boldsymbol{w}|_M^2 = \frac{1}{2}\boldsymbol{E} : \mathbb{C}^E : \boldsymbol{E} + \frac{1}{2}\boldsymbol{S} : \mathbb{S}^S : \boldsymbol{S}, \tag{8}$$

where $\boldsymbol{w} = (\boldsymbol{E}, \boldsymbol{S}) \in \mathcal{W} \subset \mathbb{R}^m \times \mathbb{R}^m$, $\mathbb{C}^E$ and $\mathbb{S}^S$ are two fourth-order symmetric positive definite tensors. Accordingly, we assume the material database is obtained in $\boldsymbol{E} - \boldsymbol{S}$ pairs as $\mathcal{D} = \{(\boldsymbol{E}_b^*, \boldsymbol{S}_b^*)\}_{b=1}^{N_{\text{data}}}$.

To find the equilibrium state that has the minimum distance to a given distribution of material strain–stress states $\{\boldsymbol{w}_a^* = (\boldsymbol{E}_a^*, \boldsymbol{S}_a^*)\}_{a=1}^{N_{\text{quad}}} \subset \mathcal{D}$ at N_{quad} quadrature points for a spatial domain discretized by finite elements, we pose the following constrained optimization problem in the reference coordinate system $\boldsymbol{X}$, i.e., total Lagrangian description:

$$\begin{aligned}
&\arg\min_{z} \int_\Omega d_M^2(\boldsymbol{w}, \boldsymbol{w}^*)\, d\Omega, \\
&\text{subject to:} \qquad \nabla^X \cdot \boldsymbol{P} + \boldsymbol{B} = \boldsymbol{0} \quad \text{in } \Omega, \\
&\qquad\qquad\quad \boldsymbol{P} \cdot \boldsymbol{N} = \bar{\boldsymbol{T}} \quad \text{on } \Gamma_P, \\
&\qquad\qquad\quad \boldsymbol{u} = \bar{\boldsymbol{U}} \quad \text{on } \Gamma_u, \\
&\qquad\qquad\quad \boldsymbol{E} = \frac{1}{2}\{\boldsymbol{F}^T \cdot \boldsymbol{F} - \boldsymbol{I}\} \quad \text{in } \Omega,
\end{aligned}$$

where $\boldsymbol{u}(\boldsymbol{X})$ is the displacement vector, $\boldsymbol{P} = \boldsymbol{F}\boldsymbol{S}$ is the first Piola–Kirchhoff stress, $\boldsymbol{F} = \boldsymbol{I} + \nabla^X \boldsymbol{u}$ is the deformation gradient, Γ_P is the traction boundary, $\boldsymbol{N}(\boldsymbol{X})$ is the unit outward normal vector to the boundary, $\bar{\boldsymbol{T}}(\boldsymbol{X})$ is the prescribed traction, $\bar{\boldsymbol{U}}(\boldsymbol{X})$ is the prescribed displacement, and $\boldsymbol{I}$ is the second-order identity tensor. The distance function reads as follows, according to the introduced norm:

$$d_M(\boldsymbol{w}, \boldsymbol{w}^*) = |\boldsymbol{w} - \boldsymbol{w}^*|_M = |(\boldsymbol{E} - \boldsymbol{E}^*, \boldsymbol{S} - \boldsymbol{S}^*)|_M. \tag{9}$$

Similar to the small-strain formulation, the above constrained optimization is first unconstrained via the method of the Lagrange multipliers. Then, using the calculus of variations, we obtain the following Euler–Lagrange equations:

$$\mathcal{R}^u = \int_\Omega \delta\boldsymbol{u} \cdot \frac{\partial \boldsymbol{E}(\boldsymbol{u})}{\partial \boldsymbol{u}} : \mathbb{C}^E : (\boldsymbol{E} - \boldsymbol{E}^*)\, d\Omega - \int_\Omega \delta\boldsymbol{u} \cdot \frac{\partial \boldsymbol{F}(\boldsymbol{u})}{\partial \boldsymbol{u}} : (\nabla^X \boldsymbol{\beta} \cdot \boldsymbol{S})\, d\Omega = 0, \tag{10}$$

$$\mathcal{R}^\beta = \int_\Omega \nabla^X \delta\boldsymbol{\beta} : \boldsymbol{P}\, d\Omega - \int_\Omega \delta\boldsymbol{\beta} \cdot \boldsymbol{B}\, d\Gamma + \int_{\Gamma_P} \delta\boldsymbol{\beta} \cdot \bar{\boldsymbol{T}}\, d\Gamma = 0, \tag{11}$$

$$\mathcal{R}^S = \int_\Omega \delta\boldsymbol{S} : [\mathbb{S}^S : (\boldsymbol{S} - \boldsymbol{S}^*) - \boldsymbol{F}^T \cdot \nabla^X \boldsymbol{\beta}]\, d\Omega = 0, \tag{12}$$

where $\boldsymbol{\beta} \in \mathbb{R}^{N_{\text{dim}}}$ is the Lagrange multiplier that enforces the balance of linear momentum, and δ indicates an arbitrary admissible variation, i.e., $\delta\boldsymbol{u} = \delta\boldsymbol{\beta} = \boldsymbol{0}$ on the Dirichlet boundary surface. Here, we reduce the number of independent fields by a local satisfaction of the last equation via:

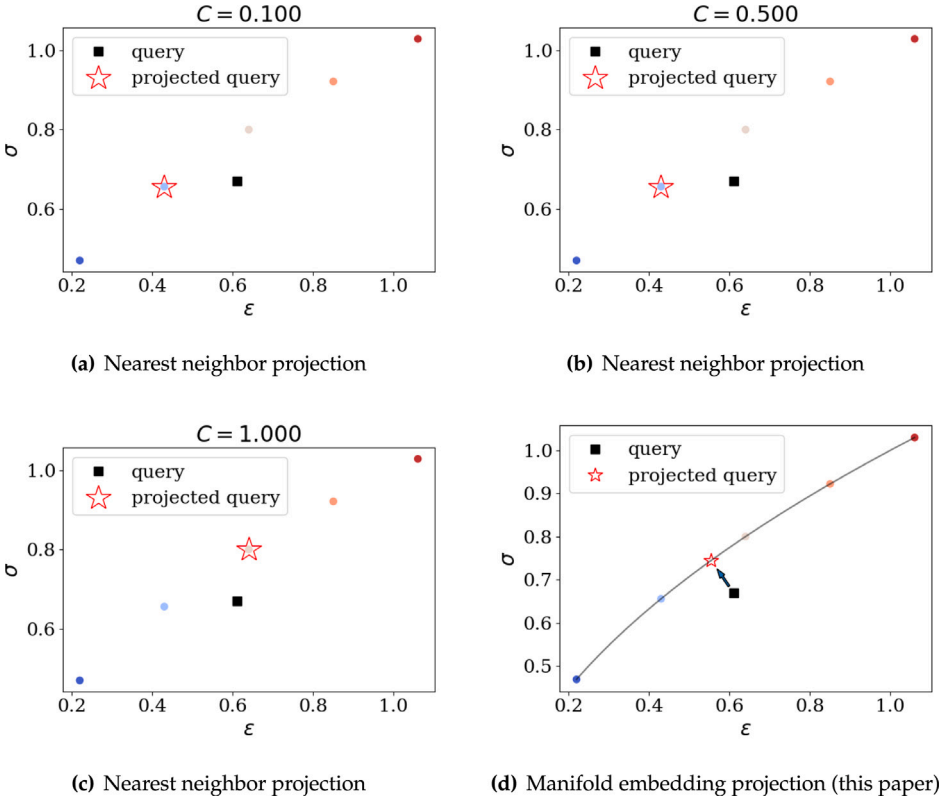
$$\boldsymbol{S} = \boldsymbol{S}^* + (\mathbb{S}^S)^{-1} : (\boldsymbol{F}^T \cdot \nabla^X \boldsymbol{\beta}) \quad \text{in } \Omega. \tag{13}$$

Due to the geometrical non-linearity, the system of equations for the global projection is nonlinear and strongly coupled. They are solved in a monolithic solver where a standard Newton–Raphson method is used. We summarize the finite-strain model-free framework in Algorithm 2.

Algorithm 2 Data-driven finite-strain solver

Input: Database $\{(\boldsymbol{E}_b^*, \boldsymbol{S}_b^*)\}_{b=1}^{N_{\text{data}}}$, and numerical parameters $\mathbb{C}^E$, $\mathbb{S}^S$
initialize all the material states to zero
while *not converged data-driven* **do**
  while *not converged Newton-Raphson* **do** ▷ Global projection $\mathcal{P}^{L\mapsto G}$
    solve linearized Eqs. (10) and (11).
  for $a = 1 : N_{\text{quad}}$ **do** ▷ Local operations $\mathcal{P}^{G\mapsto L}$
    Update equilibrium state $(\boldsymbol{E}_a, \boldsymbol{S}_a)$ by Eq. (13)
    Update material state $(\boldsymbol{E}_b^*, \boldsymbol{S}_b^*)$ by the local projection of $(\boldsymbol{E}_a, \boldsymbol{S}_a)$ via Eq. (7)

Remark 1. As pointed out by Kirchdoerfer and Ortiz (2016), the data-driven framework would require solving the Euler–Lagrange equations to update the nodal displacement and the Lagrange multiplier associated with the out-of-balance forces. This increased number of unknowns leads to a considerably larger system of equations than that of a classical displacement-based finite element solver. Moreover, the data-driven paradigm solves a double-minimization problem through an iterative scheme. As such, the distance-minimization paradigms, including the one presented in this paper, are not expected to simulate nonlinear elasticity problems faster than the conventional displacement-based finite element solver that employs an explicitly written constitutive law to update stress directly at the Gauss points.



(a) Nearest neighbor projection **(b)** Nearest neighbor projection

(c) Nearest neighbor projection **(d)** Manifold embedding projection (this paper)

Fig. 3. Comparing local projections for one arbitrary query point $z = (0.61, 0.67)$ shown by a black square marker between (a–c) the original distance minimization method and (d) manifold embedding method introduced in this paper. A query point in the local minimization step is a point that belongs to the conservation manifold, found in the global minimization step, but not necessarily has the minimum distance to the material database. In (a–c), the C parameter in Eq. (1) is varied (0.1, 0.5, 1.0) to examine how the chosen norm affects the selection of material data points. Points with colorful circular markers are material data points. The star marker shows the projected material state for the query point. The solid black curve in (d) indicates the underlying constitutive manifold used to synthesize the database.

3.2. Illustrative example 1: dependence of metrics for local projections

To show the importance of choosing a suitable metric in the local minimization step, let us consider a simple one-dimensional problem where the material database is given in Fig. 2(a). Fig. 3(a–c) studies the effect of parameter C (see Eq. (1)) on the found closest data point z^* shown by the red star marker to a query equilibrium state $z = (0.61, 0.67)$ shown by a black square marker. Notice that, in the one-dimensional setup, the fourth-order tensor $\mathbb{C}$ becomes a scalar C. As we observe, based on the value of C, there exist two different responses.

To further investigate this issue, we consider 9 random query points shown by black square markers in Fig. 4(a–c). In this experiment, we observe that the final responses can vary among *only* five different choices based on the different assigned values of the parameter C. This observation confirms that the choice of the norm may have a significant impact on the final results, which is also reported in Leygue et al. (2018). Notice that increasing the amount of data might reduce the chance of encountering this spurious issue. Here, we aim to enhance the consistency of this local projection step by introducing an algorithm that implicitly explores the underlying geometry of the data space D to find the closest interpolated data point based on a *data-driven* metric formulation.

4. Revisited local minimization in constitutive manifold

As demonstrated in, for instance, Ibañez et al. (2017) and Ibanez et al. (2018), nonlinear constitutive data often appear to be belonging to a real Riemannian manifold. While the distance between two very close points in the same tangential space of the manifold can be characterized by the Euclidean distance, the geodesics distance of data points in a constitutive manifold is generally different from the Euclidean counterpart. Our major point of departure here is the introduction of an indirect measurement of distance for the local minimization problem.

In the original work by Kirchdoerfer and Ortiz (2016), the constitutive data are thought to be in a phase space equipped with an energy norm that is used to measure the distance or discrepancy between the constitutive and equilibrium manifolds. The drawback

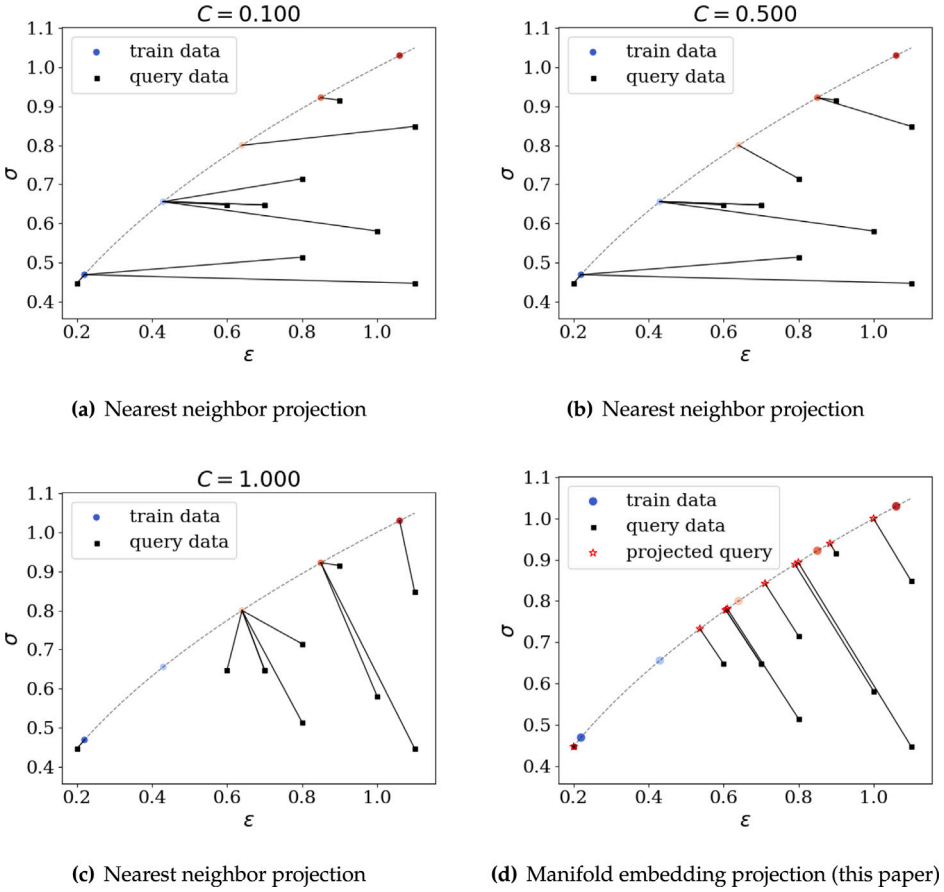


Fig. 4. Comparing local projections between (a–c) the original distance minimization method and (d) manifold embedding method introduced in this paper. Query points are shown by black square markers. In (a–c), different C parameters in Eq. (1) are used to exhibit the dependence of projected results with respect to the equipped norm. The solid lines show the projection results. The dashed line indicates the underlying constitutive manifold used to synthesize the database (colorful dot points). The results in (a–c) suggest that the norm may considerably affect the final results, especially for those points too far away from the train data points.

is that the choice of the optimal solution may be affected by the bias introduced by the different choices of the energy norms and the Euclidean distance in the phase space which has no connection to the geodesic of the constitutive manifold. This is not a significant issue for accuracy when the constitutive law is linear or when there is sufficiently high data density to guarantee that the distances measured by the norm equipped to a tangential space of the manifold and the geodesic are sufficiently close.

In the latter case, He and Chen (2020) propose a local embedding method where a locally linear patch formed among the nearest neighbors of a data point is constructed each time, and a distance is measured. Kanno (2021), on the other hand, directly uses a kernel method to establish the constitutive manifold based on the available data. Furthermore, one may also identify a finite number of affine subspaces between the data points and use them to measure the distance between points from different atlases of the manifold. In particular, the local reconstruction of the Euclidean space in He and Chen (2020) is a special case where a local space is constructed in an online manner. As an analogy, the readers may consider measuring the distance between two points by drawing a straight line through different patched surfaces of a soccer ball where the surface is locally flat and has a single normal vector within each patch but globally is of the shape of a sphere.

In this work, we refine the view that the data resides in the phase space as in Kirchdoerfer and Ortiz (2016) and restrict the data points to be elements of a sufficiently smooth manifold as in the case of Kanno (2021). However, we also want to avoid handling multiple affine subspaces to measure the distance or generate any local Euclidean subspace during the local search. As such, we introduce a new idea where we simply deform the manifold where the data reside and make it a single hyperplane (see Definition 2). This treatment then enables us to introduce a simpler perception where a single norm is sufficient to measure the distance without introducing the coordinate chart. This is done by measuring the distance of the data points of an imaginary hyperplane where all the data points lie on the nonlinear constitutive manifold are mapped onto.

We consider a constitutive response as a nonlinear mapping that maps a m-dimensional input onto a m-dimensional output. The data of the material databases are instances of this constitutive response where each instance is stored as a $2m$-dimensional vector.

For instance, the symmetric strain and stress tensors are stored as a 12-dimensional vector since both the strain and stress tensor in three-dimensional space consists of six independent components (degrees of freedom) due to their symmetries).

As such, we *globally* map the constitutive manifold data onto a hyperplane by a nonlinear mapping function $\mathcal{F} : (\mathcal{D} \subset \mathbb{R}^{2m}) \mapsto \mathbb{R}^{2m}$. Hence, if such a mapping function does exist and can be determined, the Euclidean norm is indeed a valid norm for the constructed space that embeds the constitutive manifold.

An important property we want to achieve is to ensure that the image of $\mathcal{F}$ is a hyperplane. In particular, we want to ensure that, for any data point $\hat{\boldsymbol{z}}$, the coordinates $\hat{\boldsymbol{z}}_i$ and $\hat{\boldsymbol{z}}_j$ can be linearly related by a constant matrix $\hat{\boldsymbol{K}}$, i.e.,

$$\hat{\boldsymbol{z}}_j = \hat{\boldsymbol{K}}_{ji}\hat{\boldsymbol{z}}_i \quad \text{for} \quad 1 \le i \le m, m < j \le 2\ m, \tag{14}$$

where $\hat{\boldsymbol{z}} = \mathcal{F}(\boldsymbol{z})$. To justify this claim, the hyperplane parameterized by a constant non-zero normal vector $\hat{\boldsymbol{c}} \in \mathbb{R}^{2m}$ can be found straightforwardly such that $\hat{\boldsymbol{z}} \cdot \hat{\boldsymbol{c}} = 0$. Hence, the image of the function $\mathcal{F}$ is a vector space embedded in $\mathbb{R}^{2m}$. In the rest of this subsection, we would assume such a mapping exists, and later we introduce an algorithm to find such a mapping through the expressivity power of neural networks (Hornik et al., 1989; Balestriero et al., 2021).

The image of $\mathcal{F}$ is a vector space constructed via the points sampled from the data manifold $\boldsymbol{z}^* \subset \mathcal{D}$, therefore a constitutive response outside the material data manifold$\boldsymbol{z} \notin \mathcal{D}$ will not be mapped onto the constructed hyperplane, as depicted in Fig. 1(b) where the mapped point $\boldsymbol{z}^\#$ in Step 2 does not belong to the hyperplane colored by blue. Since the normal vector of the constructed hyperplane is identical everywhere on the hyperplane, we can replace the discrete minimization statement in Eq. (7) with a continuous counterpart where a projection of the point onto the hyperplane is sufficient to determine the shortest distance defined in the mapped space. This continuous relaxation changes the initial NP-hard problem (Bahmani and Sun, 2021a) to a tractable one. We provide the details of this new local projection scheme as follows.

For the ease of interpretation and explanation, we call the first m components of the mapped variable $\hat{\boldsymbol{z}}$ pseudo-strain $\hat{\boldsymbol{\epsilon}}$ and the rest pseudo-stress $\hat{\boldsymbol{\sigma}}$, i.e., $\hat{\boldsymbol{z}}^T = [\hat{\boldsymbol{\epsilon}}^T, \hat{\boldsymbol{\sigma}}^T]$. To find a closed-form solution for the local minimization projection, we restrict that $\hat{\boldsymbol{K}}$ be symmetric positive-definite in Eq. (14), i.e., $\hat{\boldsymbol{K}} \in \mathrm{SPD}(m)$. We introduce the local minimization in the mapped space as follows:

$$\begin{aligned} &\underset{\hat{\boldsymbol{z}} \in \mathbb{R}^{2m}}{\arg\min} \quad \hat{d}^2(\boldsymbol{z}^\#, \hat{\boldsymbol{z}}), \\ &\text{subject to:} \quad \hat{\boldsymbol{\sigma}} = \hat{\boldsymbol{K}}\hat{\boldsymbol{\epsilon}}, \end{aligned}$$

where $\hat{d}(\cdot,\cdot)$ is the energy distance in the mapped domain as follows:

$$\hat{d}^2(\boldsymbol{z}^\#, \hat{\boldsymbol{z}}) = \frac{1}{2}(\boldsymbol{\epsilon}^\# - \hat{\boldsymbol{\epsilon}})^T \hat{\boldsymbol{K}}(\boldsymbol{\epsilon}^\# - \hat{\boldsymbol{\epsilon}}) + \frac{1}{2}(\boldsymbol{\sigma}^\# - \hat{\boldsymbol{\sigma}})^T \hat{\boldsymbol{K}}^{-1}(\boldsymbol{\sigma}^\# - \hat{\boldsymbol{\sigma}}). \tag{15}$$

We obtain the following unique solution for the above quadratic objective by setting the objective's derivative to zero and using basic linear algebra manipulations:

$$\hat{\boldsymbol{\epsilon}} = \frac{1}{2}(\boldsymbol{\epsilon}^\# + \hat{\boldsymbol{K}}^{-1}\boldsymbol{\sigma}^\#), \tag{16}$$

$$\hat{\boldsymbol{\sigma}} = \frac{1}{2}(\boldsymbol{\sigma}^\# + \hat{\boldsymbol{K}}\boldsymbol{\epsilon}^\#), \tag{17}$$

$$\hat{\boldsymbol{z}}^T = \mathbb{P}^T(\boldsymbol{z}^\#) = [\hat{\boldsymbol{\epsilon}}^T, \hat{\boldsymbol{\sigma}}^T], \tag{18}$$

where $\mathbb{P}$ is the projection operator to find the closest point on the hyperplane, see step $2 \rightarrow 3$ in Fig. 1(b).

Next, we should find the corresponding point in the actual data space since the governing equations are established with respect to the actual data space. Therefore, in our introduced framework, a proper map function $\mathcal{F}$ should be *globally* bijective (invertible). In the next section, we introduce a method to find an appropriate bijective map function parameterized by neural networks.

Remark 2. The energy distance in Eq. (15) is equivalent to the Euclidean distance and they can be mapped bijectively upon a linear transformation (Bahmani and Sun, 2021a). This distance can be reduced into the Euclidean distance by setting $\hat{\boldsymbol{K}} = \boldsymbol{I}_m$ where $\boldsymbol{I}_m$ is the $m \times m$ identity matrix.

Remark 3. In Eq. (14), $\hat{\boldsymbol{K}}$ controls the constructed hyperplane orientation (its normal vector). Provided that the embedding is successful such that the mapping and inverse mapping between the physical space and hyperplane both exist, the choice of $\hat{\boldsymbol{K}}$ should not have any effect on the predictions.

5. Neural network embedding of constitutive manifold

We are interested in finding an appropriate function $\mathcal{F}$ with the desired properties mentioned in the previous section to map the data points of the actual database onto a hyperplane (a Euclidean space), see Fig. 5(a). We hypothesize that such a mapping function can be found in the class of multilayer perceptrons (MLPs) due to their expressiveness power (Hornik et al., 1989). In a general setup, MLPs are not guaranteed to be invertible. However, we may use the similar idea introduced in the autoencoder architecture (cf. Hinton and Salakhutdinov (2006), Bengio et al. (2007), Vincent et al. (2008) and Baldi (2012)) to define different MLPs for the forward $\mathcal{F}$ and backward $\mathcal{G}$ functions, see Fig. 5(b). The connection between forward and backward maps, i.e., $\mathcal{G} = \mathcal{F}^{-1}$,

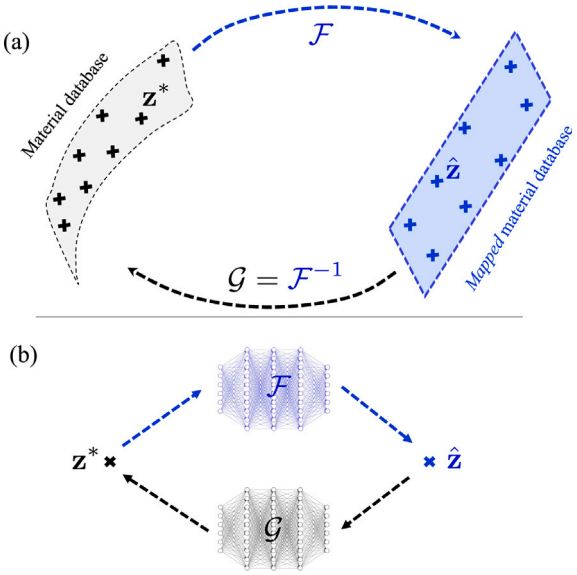


Fig. 5. (a) *global* mapping of the material data space into a vector space by a bijective function and vice-versa. (b) a vanilla autoencoder architecture to parametrize a bijective function by encoder $\mathcal{F}$ and decoder $\mathcal{G}$ deep neural networks.

is incorporated as a soft constraint in the optimization statement. Therefore, our proposed local projection can be expressed as the following optimization problem:

$$\underset{\boldsymbol{\theta}\in\mathbb{R}^{N_{\mathcal{F}}},\boldsymbol{\beta}\in\mathbb{R}^{N_{\mathcal{G}}},\hat{\boldsymbol{K}}\in\mathrm{SPD}(m)}{\arg\min} \sum_{i=1}^{N_{\text{data}}} \|\hat{\boldsymbol{\sigma}}_i - \hat{\boldsymbol{K}}\hat{\boldsymbol{\epsilon}}_i\|_2^2 + \|\boldsymbol{z}_i^* - \mathcal{G}(\mathcal{F}(\boldsymbol{z}_i^*,\boldsymbol{\theta}),\boldsymbol{\beta})\|_2^2, \tag{19}$$

where functions $\mathcal{F}$ and $\mathcal{G}$ are MLPs parameterized by unknown vectors $\boldsymbol{\theta}$ and $\boldsymbol{\beta}$ with sizes $N_{\mathcal{F}}$ and $N_{\mathcal{G}}$, respectively. In this notation, these vectors concatenate weights and biases. The first term in the above objective is designated to enforce the linearity of the target vector space. The second term enforces the bijectivity constraint also known as the reconstruction error for the autoencoder architecture where $\mathcal{F}$ and $\mathcal{G}$ are the encoder and decoder functions respectively. The above architecture is a naive formulation of the problem statement explained in the previous section, and we call it the *vanilla* autoencoder formulation.

By leveraging the expressive power of the deep neural networks (Hornik et al., 1989; Raghu et al., 2017), we argue that, as long as (1) the training is successful and (2) the unsampled data not used in the training process are indeed on the same manifold of the training data, the numerical value of $\hat{\boldsymbol{K}}_{\text{opt}}$ in (19) is not consequential. Since it is equivalent to any *fixed* $\hat{\boldsymbol{K}}_{\text{fix}} \in \mathrm{SPD}(m)$ up to a linear transformation $\hat{\boldsymbol{K}}_{\text{opt}} = \boldsymbol{T}\hat{\boldsymbol{K}}_{\text{fix}}; \boldsymbol{T} \in \mathrm{SPD}(m)$, every two members of SPD group commutes by the matrix multiplication. This linear transformation can be indirectly found as the outcome of a successfully trained neural network in which we are interested in the properties of the embedding space but not its explicit value. Therefore, in this work, we fix the matrix $\hat{\boldsymbol{K}}_{\text{fix}} \in \mathrm{SPD}(m)$ to simplify the optimization problem.

The optimization over MLPs is known to be non-convex (Goodfellow et al., 2016); hence, finding the global minimizer of the above objective is not trivial in a general setting. Even for the near-optimal local minimizers, the bijectivity constraint cannot be achieved precisely. Moreover, this optimization problem is *multi-objective*, which makes it necessary to consider the possibility of conflict situations where Pareto efficiency could be a primary concern (Yu et al., 2020; Bahmani and Sun, 2021b). To address these drawbacks, we incorporate a new architecture in the next subsection that leads to neural network predictions preserving the bijectivity condition by construction and hence bypassing the need to handle conflicting objectives.

5.1. Invertible neural network

Due to the drawbacks of the vanilla autoencoder framework mentioned in the previous section, we formulate a new machine learning strategy built upon a type of neural network that is *bijective* by its construction (as an inductive bias). The invertible neural network is built upon several coupling layers. A coupling layer, schematically shown in Fig. 6, maps the ith layer's feature vector $\boldsymbol{z}^i \in \mathbb{R}^{n_i}$ to the $(i+1)$th layer's feature vector $\boldsymbol{z}^{i+1} \in \mathbb{R}^{n_{i+1}}$ via the function $\mathcal{F}^i : \boldsymbol{z}^i \mapsto \boldsymbol{z}^{i+1}$ customized as follows Dinh et al. (2014, 2016), Ardizzone et al. (2018) and Beitler et al. (2021):

$$\boldsymbol{z}_1^{i+1} = \boldsymbol{z}_1^i \odot h_1^i(\boldsymbol{z}_2^i) + f_1^i(\boldsymbol{z}_2^i), \tag{20}$$

$$\boldsymbol{z}_2^{i+1} = \boldsymbol{z}_2^i \odot h_2^i(\boldsymbol{z}_1^{i+1}) + f_2^i(\boldsymbol{z}_1^{i+1}), \tag{21}$$

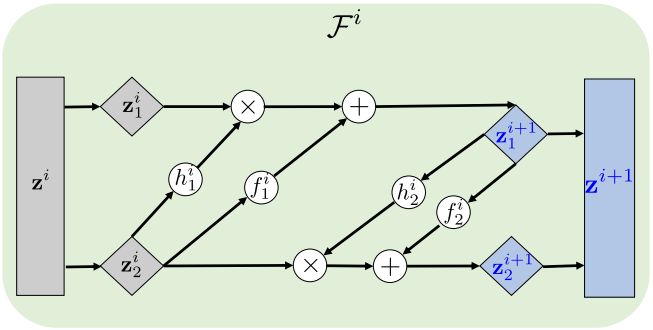


Fig. 6. Computational graph of the ith coupling layer in the invertible architecture. h_1^i, h_2^i, f_1^i, and f_2^i are any arbitrary functions such as MLP. $\boldsymbol{z}^i$ and $\boldsymbol{z}^{i+1}$ are the input and output feature vectors of the coupling layer.

where $\odot$ is the Hadamard product (element-wise multiplication), and h_1^i, h_2^i, f_1^i, and f_2^i are four arbitrary functions called internal functions. In each coupling layer, before the forward pass, the input feature $\boldsymbol{z}^i$ is divided into two arbitrary, disjoint halves $\boldsymbol{z}_1^i \in \mathbb{R}^{m_1^i}$ and $\boldsymbol{z}_2^i \in \mathbb{R}^{m_2^i}$ where $m_1^i + m_2^i = n_i$. Notice that the internal functions can be deep neural networks to increase the complexity and expressivity of each layer. In this work, we parametrize these functions via MLPs (see Fig. 6). Because of the simple yet elegant structure in each coupling layer, the backward map $(\mathcal{F}^i)^{-1} : \boldsymbol{z}^{i+1} \mapsto \boldsymbol{z}^i$ is readily available as follows:

$$\boldsymbol{z}_2^i = (\boldsymbol{z}_2^{i+1} - f_2^i(\boldsymbol{z}_1^{i+1})) \odot \frac{1}{h_2^i(\boldsymbol{z}_1^{i+1})} \tag{22}$$

$$\boldsymbol{z}_1^i = (\boldsymbol{z}_1^{i+1} - f_1^i(\boldsymbol{z}_2^i)) \odot \frac{1}{h_1^i(\boldsymbol{z}_2^i)}. \tag{23}$$

A deep bijective, invertible neural network $\mathcal{F} : (\boldsymbol{z}_{\text{in}} \in \mathbb{R}^{n_{\text{in}}}) \mapsto (\boldsymbol{z}_{\text{out}} \in \mathbb{R}^{n_{\text{out}}})$ can be constructed by the composition of L coupling layers:

$$\mathcal{F}(\boldsymbol{z}; \boldsymbol{\theta}) = \mathcal{F}^L \circ \mathcal{F}^{L-1} \circ \cdots \circ \mathcal{F}^1(\boldsymbol{z}). \tag{24}$$

where $\boldsymbol{\theta}$ concatenates all the parameters defined for deep neural networks $\{h_1^i, h_2^i, f_1^i, f_2^i\}_{i=1}^L$.

We tailor the introduced architecture for our purpose as follows. The input feature vector $\boldsymbol{z}^1 \leftarrow \boldsymbol{z}^* \in \mathcal{R}^{2m}$ can be naturally halved into $\boldsymbol{z}_1^1 \leftarrow \boldsymbol{\epsilon}^*$ and $\boldsymbol{z}_2^1 \leftarrow \boldsymbol{\sigma}^*$. In our experience, we find that a single coupling layer is sufficient for the problems solved herein. In this setup, the output feature vector becomes $(\boldsymbol{z}^2)^T = [(\boldsymbol{z}_1^2)^T, (\boldsymbol{z}_2^2)^T]$ and $\hat{\boldsymbol{\epsilon}} \leftarrow \boldsymbol{z}_1^2$ and $\hat{\boldsymbol{\sigma}} \leftarrow \boldsymbol{z}_2^2$. Also, to reduce the training computational cost, we set $h_1^1 \equiv h_2^1 \equiv \boldsymbol{J}_m$ and $f_1^1 = \boldsymbol{0}_m$ where $\boldsymbol{J}_m$ and $\boldsymbol{0}_m$ are vectors of size m filled by ones and zeros, respectively. This architecture is equivalent to the network introduced in Dinh et al. (2014).

Utilizing this architecture, the previous multi-objective loss function in Eq. (19) becomes a single objective optimization problem as follows:

$$\underset{\boldsymbol{\theta} \in \mathbb{R}^{N_{\mathcal{F}}}}{\arg\min} \sum_{i=1}^{N_{\text{data}}} \|\hat{\boldsymbol{\sigma}}_i - \hat{\boldsymbol{K}}_{\text{fix}} \hat{\boldsymbol{\epsilon}}_i\|_2^2, \tag{25}$$

where $\hat{\boldsymbol{z}} = \mathcal{F}(\boldsymbol{z}^*; \boldsymbol{\theta})$. In the following subsection, we showcase the application of the introduced local projection via a simple example.

Algorithm 3 Query inference with the introduced local projection

1: **Input: a query point** $(\boldsymbol{\epsilon}, \boldsymbol{\sigma})$
2: Map forward $(\boldsymbol{\epsilon}, \boldsymbol{\sigma})$ via Eqs. (20) and (21) to $(\boldsymbol{\epsilon}^\#, \boldsymbol{\sigma}^\#)$.
3: Find the closest point $(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\sigma}})$ on the constructed vector space to $(\boldsymbol{\epsilon}^\#, \boldsymbol{\sigma}^\#)$ by Eqs. (16) and (17).
4: Map backward $(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\sigma}})$ via the inverse operations in Eqs. (22) and (23) to $(\boldsymbol{\epsilon}^*, \boldsymbol{\sigma}^*)$.

Remark 4. In all of the examples solved in this paper, we set $\hat{\boldsymbol{K}}_{\text{fix}}$ equal to the identity tensor which is corresponding to the regular l2-norm projection as discussed in Remark 2.

Remark 5. Our proposed mapping operation $\mathcal{F}$ is an embedding (not immersion) operation since it is globally invertible and homeomorphism to its image.

Remark 6. The introduced method in this paper relaxes the discrete optimization step of the original distance minimization to a continuous local projection onto a constructed smooth hyperplane; hence it breaks the NP-hardness.

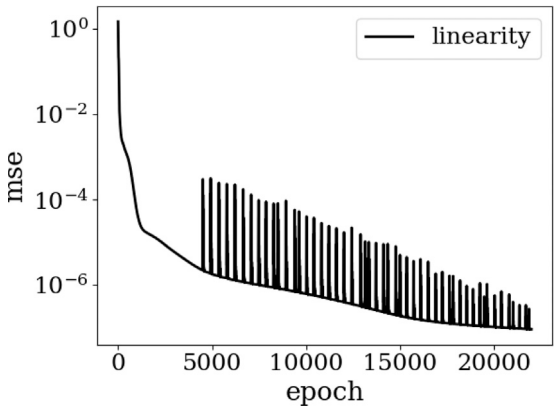


Fig. 7. Training performance of the invertible neural network trained with the database in Fig. 2(a).

Remark 7. The global embedding approach only requires one offline neural network training per data set to construct the hyperplane but does not require generations of new local space during the simulations. The trade-off is that embedding the manifold globally is, in general, more complex than creating a local linear space that preserves the convexity (cf. He and Chen (2020)). Hence, a neural network architecture that possesses sufficient expressibility is necessary to handle the global embedding case, in particular for high-dimensional cases such as micropolar elasticity or multiphysics problems. Other feasible global manifold approaches (which could potentially bypass this complexity in an offline manner but still leverage the simplicity of the Euclidean space) may require the constructions of atlas and coordinate charts (or patches) such that the co-domains of each chart, which are themselves Euclidean space intersecting with each other, can be collectively used to represent the manifold (see, for instance, Lin and Zha (2008) for a list of options). These alternatives are, nevertheless, out of the scope of this study.

5.2. Illustrative examples 2

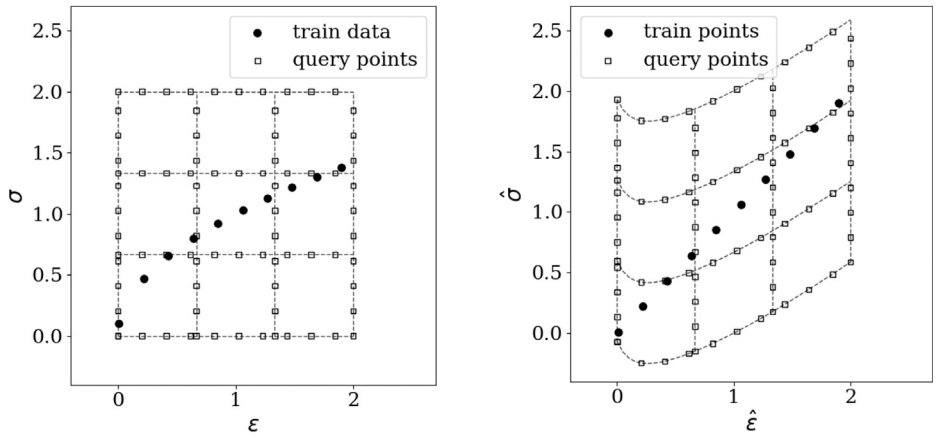
Here, we walk through the details of the introduced local projection method via the simple 1D example used in Section 3.2.

Recall that we construct the embedded vector space of the constitutive manifold in an offline manner based on the data points in an available database. The database we use is similar to the previous example, see Fig. 2(a). We utilize one invertible coupling layer with the internal function parameterized by an `elu` MLP (Clevert et al., 2015). This MLP has three hidden layers. Each layer consists of 5 units initialized by the uniform Kaiming approach (He et al., 2015). The objective is minimized by the Adam method (Kingma and Ba, 2014) with the initial learning rate of 5e−3 in a full-batch manner. To enhance the training step, we use the `ReduceLROnPlateau` learning scheduler of `PyTorch` library (Paszke et al., 2019) to reduce the learning rate by the factor of 0.91 every 50 iterations after the first 2000 iterations. The minimum learning rate is set to 1e−6. These hyperparameters are tuned manually with satisfactory results as shown in 7. A more rigorous hyperparameter tuning might, to some degree, further improve the performance, but such an endeavor is not the focus of this study (Bardenet et al., 2013; Fuchs et al., 2021; Heider et al., 2021).

Fig. 7 summarizes the training performance. As this result suggest, the invertible architecture finds a near-optimal mapping function that can approximately satisfy the introduced loss function in Eq. (25) with the error in the order of $O(-6)$. For a better comparison, the mapped database in the constructed vector space is plotted in Fig. 2(b). The linearity of the data expressed in the coordinates of the embedded space is obvious.

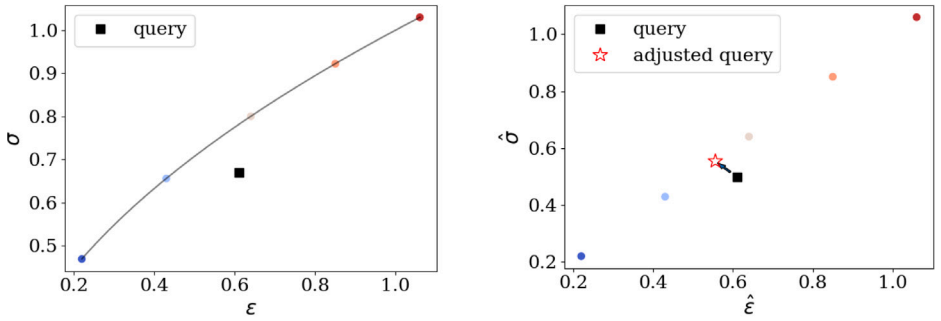
As discussed earlier, if a query point does not belong to the data manifold, it will be out of the constructed hyperplane. Hence, we define a Euclidean projection to project the mapped query point onto the constructed hyperplane, see Eqs. (16) and (17). To show this and depict the effect of the mapping function, we present the deformation of a regular grid in the strain–stress space caused by the mapping function $\mathcal{F}$ in Fig. 8. The mapping function can be thought of as a deformation of the domain of the actual data points (circular markers in Fig. 8(a)) such that the resultant deformed configuration becomes a hyperplane, i.e., a straight line in 1D setup; see circular markers in Fig. 8(b). However, points out of the data manifold, i.e., square markers in 8(a), will not be placed on the constructed hyperplane.

We show the steps of our local projection method at the query time in Fig. 9, see Algorithm 3. In Fig. 9(a) the similar equilibrium state (depicted by a black square marker) to the problem in Fig. 3 is chosen to be projected onto the data space. First, we map the query point, using Eqs. (20) and (21), onto the constructed vector space by the trained invertible neural network, see the black square marker in Fig. 9(b). Then, we find the closest point on the vector space to the mapped query point by Eqs. (16) and (17), see the red star marker in Fig. 9(b). Finally, we map back to the actual data space via the inverse operations Eqs. (22) and (23), see the red star marker in Fig. 9(c). As the final result suggests, the proposed local projection can find the closest material point in the interpolation region of the data space and admits the underlying manifold. Notice that, even qualitatively, the found closest point is closer than the found points by the nearest neighbor approach, c.f. Fig. 3, and still the found point belongs to the underlying data generator manifold (black curve line in Fig. 9).

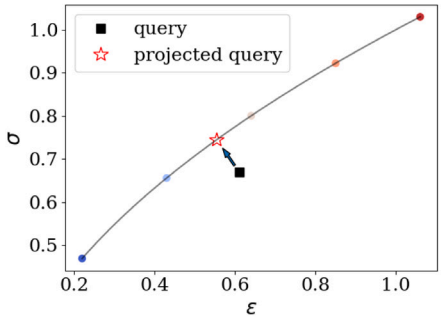


(a) regular grid points in strain-stress space **(b)** grid deformation after mapping

Fig. 8. (a) a regular grid in the strain–stress space, (b) the deformation of this grid in the mapped domain.



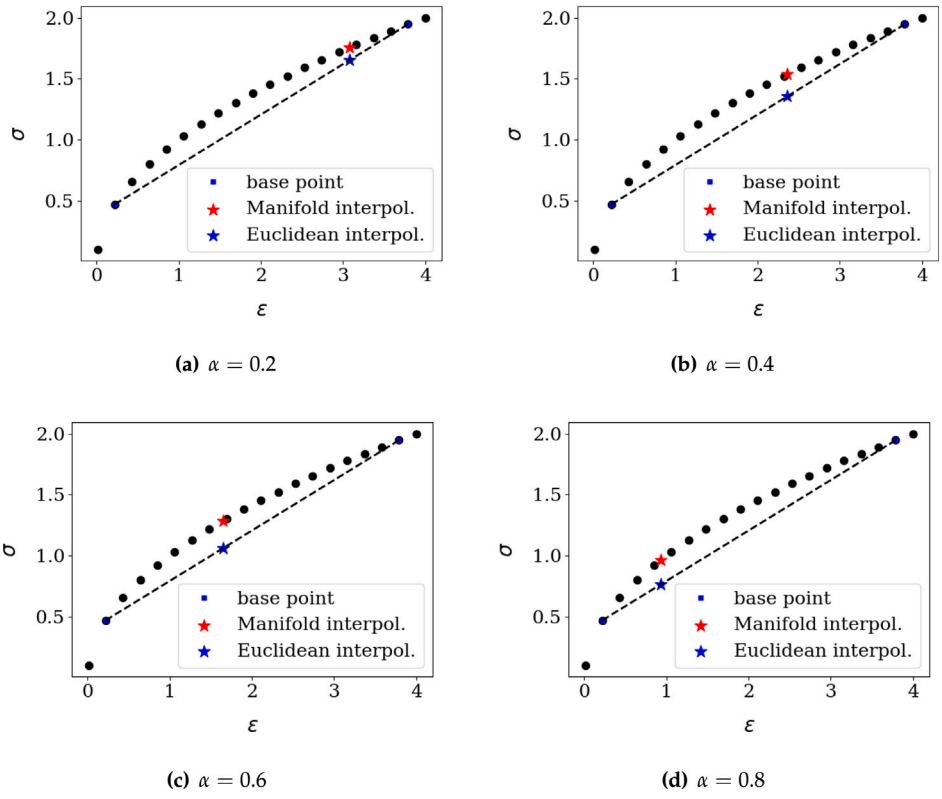
(a) step 0: requested query **(b)** step 1: forward map to a vector space



(c) step 2: backward map to the real space

Fig. 9. Summary of the steps for the introduced local projection: (a) a query point shown by a black square marker is requested to be projected onto the database. (b) the trained neural network is first applied to embed the database and the query point to the space that has vector space properties for database points. Moreover, in this step, the mapped query point should be adjusted to the constructed vector space for the database, see red star point in (b). (c) the adjusted point in (b) is finally mapped back to the real data space via the inverse operation. The black curve line shows the underlying material manifold. Colored circle points belong to the database.

For a better comparison with the nearest neighbor method, we apply our introduced local projection method to the similar query points used in Fig. 4(a–c). The result is indicated in Fig. 4(d) where the found local projections by our method provide a smooth transition from one point to the other compared with Fig. 4(a–c) where there were only five possible discrete choices. Our scheme can accurately interpolate between data points such that the projected values belong to the underlying data manifold.



(a) $\alpha = 0.2$ **(b)** $\alpha = 0.4$

(c) $\alpha = 0.6$ **(d)** $\alpha = 0.8$

Fig. 10. Interpolation capability in the regular data space and the constructed vector space for different values of α defined in Eq. (26). Interpolated points (red star markers) in the constructed vector space are on the underlying material manifold. However, not surprisingly, the interpolation in the actual data space results in out of manifold points (blue star markers).

5.3. Illustrative examples 2: a sanity check through the lens of convex interpolation on the manifold

Here, we examine the linearity of the constructed vector space by its convex interpolation capability. We claim that if the constructed space is truly vector space and the mapping function truly respects the underlying manifold, then any convex interpolation between two arbitrary points on the constructed space should result in a point on the actual manifold after the backward map. A convex interpolation between two points $\hat{\boldsymbol{z}}_s$ and $\hat{\boldsymbol{z}}_e$ is defined as follows:

$$\hat{\boldsymbol{z}}_\alpha = \alpha \hat{\boldsymbol{z}}_s + (1-\alpha)\hat{\boldsymbol{z}}_e; \quad \alpha \in (0,1). \tag{26}$$

We examine this property in Fig. 10. Start and end points $\hat{\boldsymbol{z}}_s$ and $\hat{\boldsymbol{z}}_e$ are shown by blue square markers. A red star marker depicts the interpolated point in the constructed vector space. The interpolated point in the actual data space is depicted as a blue star marker. As this experiment shows, the interpolation in the constructed vector space is equivalent to the interpolation over the underlying manifold. However, the regular interpolation in the actual data space results in points out of the underlying data manifold.

Remark 8. The performance of a trained neural network (e.g., number of neurons, number of layers, type of activation) depends on the hyperparameters. The neural network embedding is no exception. In this work, the tuning of the hyperparameters is done manually and is sufficient for our purpose, as evidenced by the loss function value at the end of the training. Other feasible and more rigorous approaches to fine-tuning the best combination of parameters can be done by, for instance, the greedy search, random search (Bergstra and Bengio, 2012), random forest (Probst et al., 2019), Bayesian optimization (Klein et al., 2017), metaheuristic or deep reinforcement learning (Fuchs et al., 2021; Heider et al., 2021). Applying state-of-the-art tuning techniques for hyperparameters is out of the scope of this study, but the relevant information can be found in the aforementioned literature.

6. Numerical examples

In this section, we benchmark four numerical problems to examine the accuracy and robustness of the proposed scheme in various scenarios. Importantly, we compare the efficiency of this method with the original distance minimization scheme in the limited data regime. First, in Section 6.1, we solve a 1D bar problem for two databases with different availability of data. Moreover, we compare

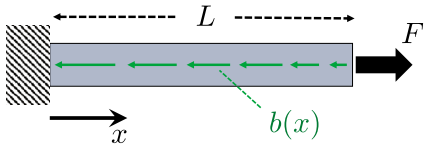


Fig. 11. One-dimensional bar with length L, body force $b(x)$, and applied force F. We use 50 uniform finite elements with linear basis functions to discretize the domain.

the robustness of the proposed invertible network with its vanilla autoencoder variant. Second, we use the same material database to solve a 3D truss system under loading–unloading conditions in Section 6.2. In this problem, we point to a circumstance where the original distance minimization may predict a spurious dissipation mechanism for an elastic material. Third, we compare the efficiency of both methods for a plane strain problem under small-strain assumption that possesses stress concentration due to the geometrical imperfection in Section 6.3. In Section 6.4, we study the application of the proposed method for an anisotropic material under the finite-strain condition. Moreover, in Appendix C, we showcase and validate the application of the proposed method for a heat conduction problem which is studied in the literature as well. The databases will be publicly available for third-party validation exercises (upon acceptance of this manuscript).

6.1. Nonlinear 1D bar code verification

The purpose of this example is to verify the implementation of the model. As pointed out by, for instance, Roache (2002), the method of manufactured solution (MMS) employed here for code verification is a purely mathematical exercise where one examines whether the computer model is capable of robustly replicating a manufactured solution of a boundary value problem. Satisfying the manufactured solution is a necessary but not sufficient condition for a successful modeling effort. In this example, our goal is to compare the accuracy and robustness of our proposed method and the original distance minimizing method (Kirchdoerfer and Ortiz, 2016) by applying them to solve the same boundary value problem.

The problem of interest is a classical 1D nonlinear elasticity problem under the quasi-static condition, as illustrated in Fig. 11. It contains a homogeneous bar composed of a nonlinear elastic material subjected to an external load. The material constitutive behavior is assumed to follow Eq. (27) defined as:

$$\sigma(\epsilon) = \alpha_m \tanh(\alpha_s \epsilon), \tag{27}$$

where α_m and α_s are material parameters. The bar length is set to $L = 1\text{m}$. The applied force at the right end is set to $F = 833.6546$ N, and the left end is fixed from the movement. The cross-sectional area of the bar is constant and equal to $1\ \text{mm}^2$. The body force applied on the 1D bar is $b(x) = -0.02\alpha_s\alpha_m[1 - \tanh^2(0.02\alpha_s x)]$. The governing equations in this problem follow the balance of linear momentum in the one-dimensional domain. The corresponding analytical MMS solution reads,

$$u(x) = 0.01x^2; x \in [0, L], \tag{28}$$

which is obtained for the constitutive law expressed in Eq. (27). This constitutive law is assumed to be the ground truth for verification purposes only.

6.1.1. Limited, complete database

The material database is a set of data points synthesized from the model in Eq. (27) with parameters $\alpha_m = 1000$ MPa and $\alpha_s = 60$. The database is populated by sampling 41 equally spaced data points in the range $\epsilon \in [-0.03, 0.03]$. For comparison purposes, we refer it as the *complete* database.

The original distance minimization method directly uses the database in its local optimization step. However, the method introduced in this paper requires an appropriate mapping $\mathcal{F} : \mathbb{R}^2 \mapsto \mathbb{R}^2$ to perform operations between the ambient data space and the mapped vector space, as explained in Sections 4 and 5. This mapping is found in an offline manner by a single invertible layer which has three hidden `elu` layers (Clevert et al., 2015) with five hidden units per layer. Neural network weights and biases are initialized by the uniform Kaiming approach (He et al., 2015).

The neural network parameters are found by the `ADAM` optimizer (Kingma and Ba, 2014) with the initial learning rate set at 0.05. We use the `ReduceLROnPlateau` learning scheduler of `PyTorch` library (Paszke et al., 2019) to adjust the learning rate every 50 iterations after the first 2000 iterations. The learning rate reduction factor is set to 0.91 with the minimum learning rate 1e−6. Before training, strain and stress data points are linearly normalized based on their maximum and minimum values to be positive and less than or equal to 1. The training performance is shown in Fig. 12.

Fig. 13 plots the raw data points and the mapped one which has the vector space properties after training the neural network map function. The results confirm that the proposed neural network architecture is able to find an appropriate bijective function to map forward and backward between the original data space and its vector space counterpart.

We compare the distribution of displacement, strain, and stress along the bar obtained from the embedding method and the original distance minimization method in Fig. 14. Generally speaking, both approaches capture the general trends of the exact solution with the embedding method showing slightly more accurate predictions on the displacement field. A major point of the

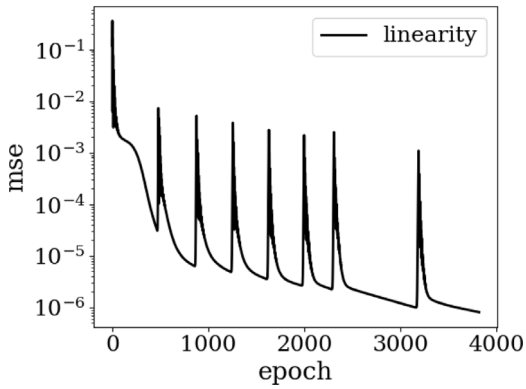
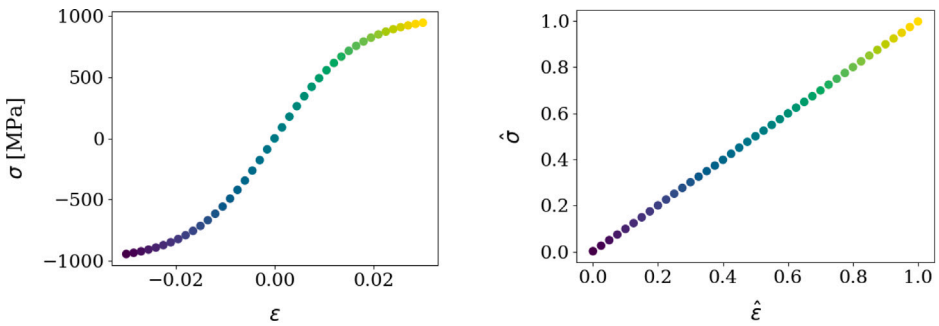


Fig. 12. The mean squared error (mse) of the linearity condition violation during the training epochs for *complete* data shown in Fig. 13(a). The training is performed in a full-batch manner.



(a) material database **(b)** mapped material database

Fig. 13. (a) Material database and (b) mapped material database embedded onto a vector space using the invertible neural network. Colors show the data number.

departure between the two methods is the predicted strain field where the original distance minimization method exhibits spurious oscillations in the strain field, while the embedding method does not. This spurious non-smoothness has been observed previously in the numerical solutions obtained from the model-free paradigm, such as Carrara et al. (2020) and Bahmani and Sun (2021a), and is related to the discrete nature of the point cloud. Note that the spurious non-smoothness exhibited in the original distance minimization can also be suppressed by increasing the density of the data points in the parametric space. However, this non-smoothness may also become more severe when the available data is sparse. While limiting the data may increase the difficulty in generating the correct inductive bias (see Williams et al. (2021) for a more in-depth discussion), the manifold embedding may overcome the spurious non-smoothness due to the continuous nature of the hyperplane.

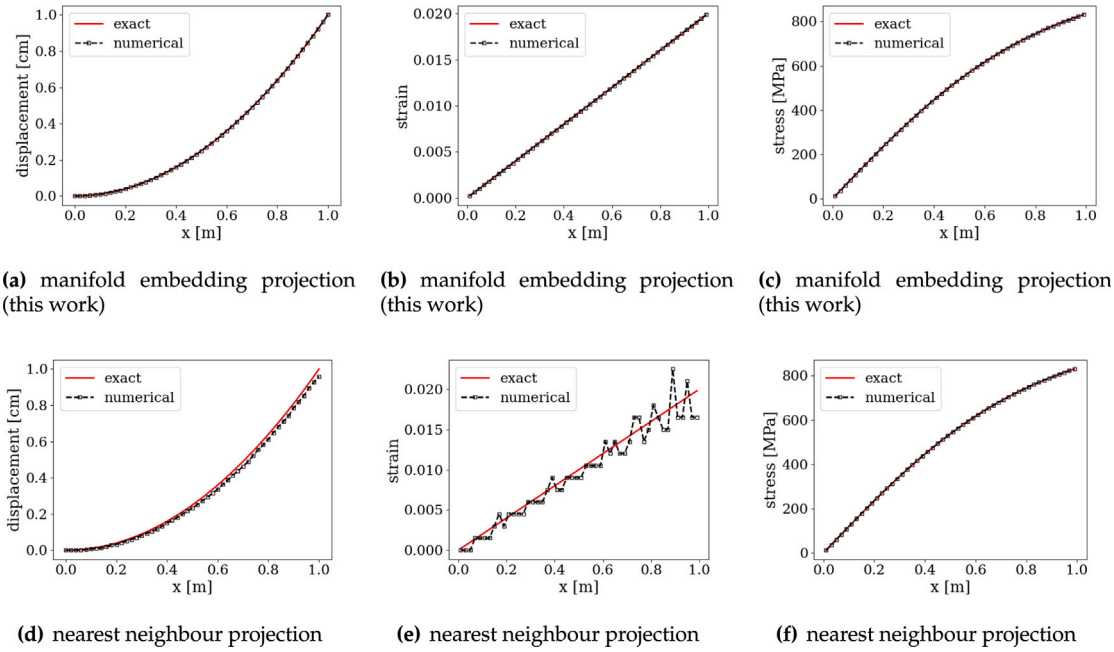
As a support for the previous claim, we plot the strain–stress state at each bar element for both schemes in Fig. 15. The found material states by the proposed scheme accurately resemble the underlying generative function that populates the material database. This observation indicates that the proposed scheme is able to indirectly learn and recover the underlying geometric structure of the material behavior.

6.1.2. Limited, incomplete database

To validate the robustness of the proposed scheme, we eliminate some data points from the nonlinear regions of the previous database and call the new database *incomplete*, as shown in Fig. 16(a). We retrain the neural network map function for the new database, and the training performance is reported in Fig. 17; the architecture and hyper-parameters are kept the same as the ones used for the *complete* database.

The predicted displacement, strain, and stress fields along the bar are shown in Fig. 18 for both methods. As the results suggest, the accuracy of the proposed method in this paper is still satisfactory even for such an incomplete and limited database. However, as expected, the accuracy of the original distance minimization is considerably reduced, especially for displacement and strain fields (c.f. Fig. 14(d–e)).

Similarly, the found strain–stress states in the entire domain are plotted as dot points in Fig. 19 to demonstrate the capabilities of the proposed manifold learning algorithm in recovering the embedded space of the underlying hidden constitutive manifold. Comparisons of results in Fig. 19(a) with Fig. 19(a) reveal that the performance of the proposed manifold method is not significantly



(a) manifold embedding projection (this work)

(b) manifold embedding projection (this work)

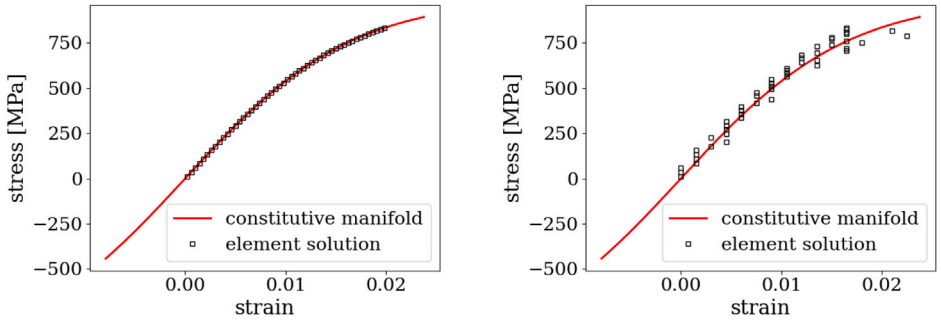
(c) manifold embedding projection (this work)

(d) nearest neighbour projection

(e) nearest neighbour projection

(f) nearest neighbour projection

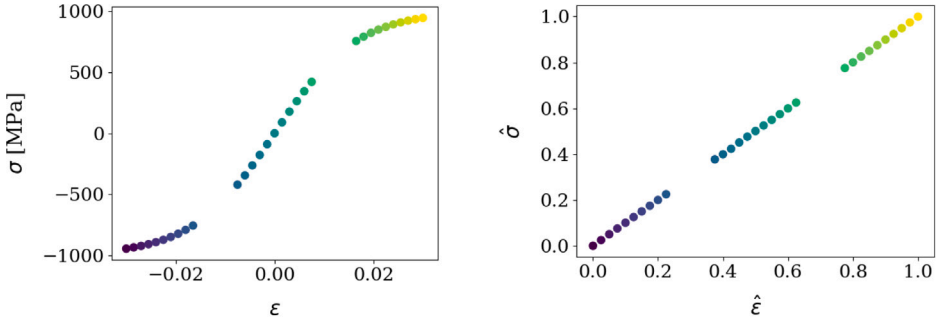
Fig. 14. A comparison between solution fields obtained by the proposed scheme and nearest neighbor projection onto the database. The strain field predictions by the proposed method are more accurate and smooth; see (b) and (e).



(a) manifold embedding projection (this work)

(b) nearest neighbour projection

Fig. 15. Strain–stress values obtained via (a) the introduced projection and (b) the nearest neighbor projection. The proposed scheme may identify material states belonging to the hidden material manifold.



(a) material database

(b) mapped material database

Fig. 16. (a) *incomplete* material database (b) mapped material database into a vector space using the invertible neural network. Colors show the data number.

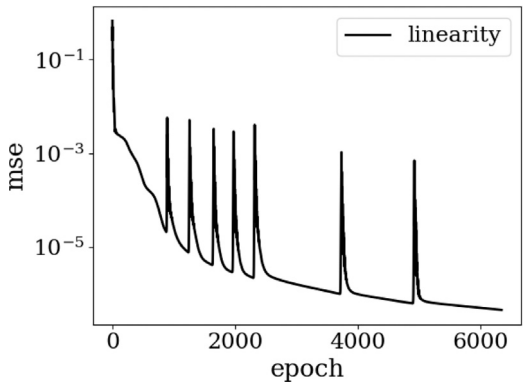
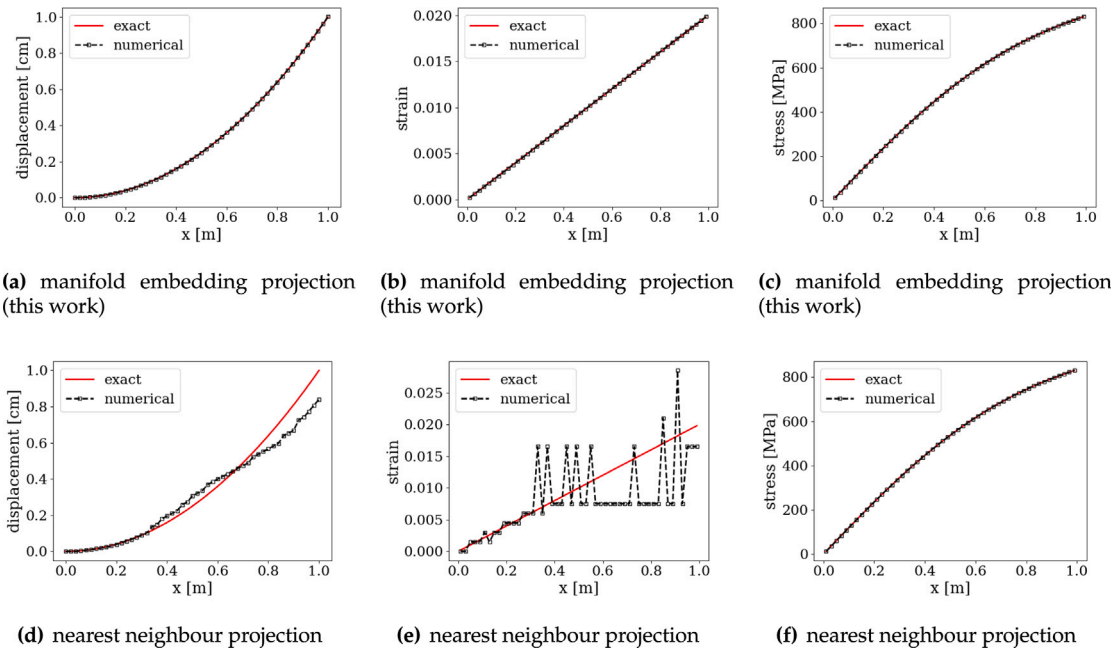


Fig. 17. The mean squared error (mse) of the linearity condition violation during the training epochs for *incomplete* data shown in Fig. 16(a). The training is performed in a full-batch manner.



(a) manifold embedding projection (this work) **(b)** manifold embedding projection (this work) **(c)** manifold embedding projection (this work)

(d) nearest neighbour projection **(e)** nearest neighbour projection **(f)** nearest neighbour projection

Fig. 18. A comparison between solution fields obtained by the proposed scheme and nearest neighbor projection. Even in the *incomplete* data scenario, the proposed scheme can find more accurate responses compared to the original distance minimization scheme.

affected by the missing data. On the other hand, the predicted state values obtained via the original distance minimization method are shown to be very sensitive to the missing data as evidenced in Figs. 19(b) and 15(b).

This numerical experiment indicates that the knowledge of the geometry of the manifold gained from the neural network may improve the robustness of the data-driven predictions. Of course, the quality of the projections may still depend on how the missing data distributes, the smoothness of the data, and whether losing this data may otherwise significantly alter the learned mapping between the constitutive manifold and the embedded hyperplane. It should be noticed that, while we did not implement the manifold learning alternatives, we believe that the spurious oscillation pattern in the strain field exhibited in Fig. 18(e) can also be suppressed when a global manifold is constructed (Kanno, 2021), a locally embedding space is identified (He et al., 2021) or the data density is sufficiently high such that the projection is within a very small distance.

6.1.3. Comparison with the vanilla autoencoder architecture

Here, we aim to check the reconstruction error and the bijectivity constraint of the proposed invertible architecture compared with a vanilla autoencoder architecture. The details of the autoencoder map function and its optimization statement are described at the beginning of Section 5.

Encoder and decoder branches of the autoencoder framework have three hidden layers with three hidden units per layer. For a fair comparison between these two frameworks, this particular network configuration is chosen to be similar to the invertible internal

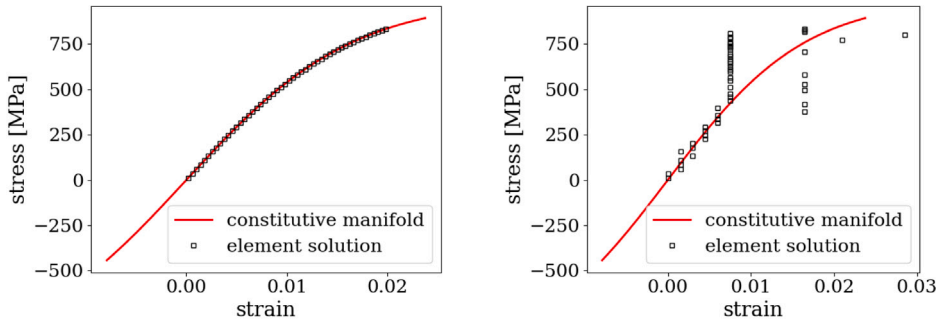


Fig. 19. Strain–stress values obtained at each finite element via (a) the introduced projection and (b) the nearest neighbor projection. The proposed scheme may identify material states that belonging to the hidden material manifold, even in the incomplete data-set scenario.

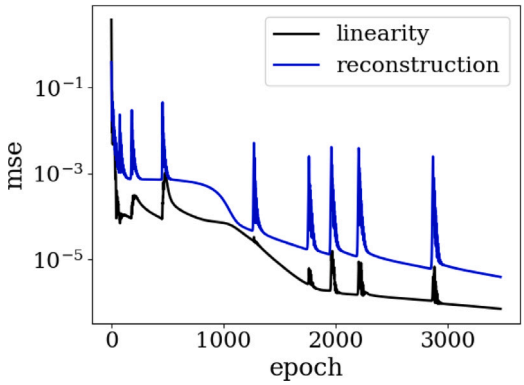
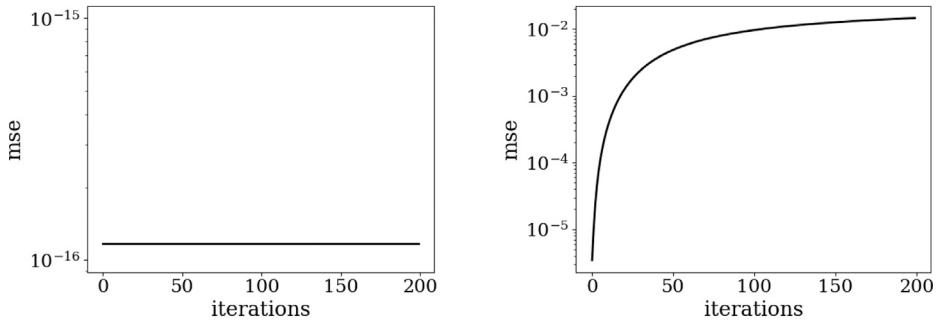


Fig. 20. Mean squared error (mse) of the reconstruction and linearity constraint errors during the training epochs for the *complete* data shown in Fig. 13(a) using the vanilla autoencoder framework. The training is performed in a full-batch manner. The number of training iterations is similar to the invertible network training shown in Fig. 12.

network with almost the same number of unknown parameters. The total number of unknown parameters for the autoencoder and invertible networks are 82 and 76, respectively. The rest of the hyper-parameters are similar to the invertible network reported earlier in Section 6.1.1.

Fig. 20 shows the autoencoder training performance. With the same number of gradient descent iterations, the obtained error of the linearity constraint is almost similar to the invertible architecture (c.f. Fig. 12). However, the autoencoder network's reconstruction error (backward map) is significant relative to the machine precision error of the invertible network, although it is small in the absolute sense. To compare the bijectivity property and possible accumulated round of errors, we study the reconstruction stability and accuracy of the trained networks during 200 consecutive forward and backward mappings for a batch of 100 data points generated randomly from the constitutive manifold and not seen in the training. The results are plotted in Fig. 21. Not surprisingly, the accuracy of the invertible network remains constant around the machine precision, while the autoencoder accuracy rapidly decreases up until a saturation level. Notice that this experiment of consecutive backward–forward mappings is designed as a validation exercise to test the robustness of the forward and backward mappings and to detect any potential information loss when the forward and backward mapping occurs.

Since the invertibility of the forward and backward mapping may prevent the information loss, it is a property desirable for the cases where the data set does not contain noise or error. This could be the case when the data are generated from direct numerical simulations, and the prediction is made for the constitutive responses of the corresponding effective medium. On the other hand, there are also circumstances in which the invertibility is purposely disabled in order to denoising data (e.g., data compression via the singular value decomposition). In the case where data are acquired from experiments or sensors may contain both noise and outliers, these noises and outliers may increase the difficulty of embedding through the neural network training. Presumably, this issue can be circumvented by de-noising or filtering procedures performed before the embedding (cf. Lyu et al. (2019)). Interested readers may also be referred to, for instance, He et al. (2021) in which autoencoder is used. In this setting, the encoder is used to compress data into latent space and therefore filter noises, while the decoder is used to reconstruct the de-noised data such that the processed data is smooth and free of oscillation.



(a) introduced invertible architecture **(b)** vanilla autoencoder architecture

Fig. 21. Comparing the stability of two architectures after 200 forward and backward mappings for 100 randomly generated data points not used in the network training. The reconstruction error of the tailored architecture (a) is almost negligible and remains stable during continuously forward–backward mappings.

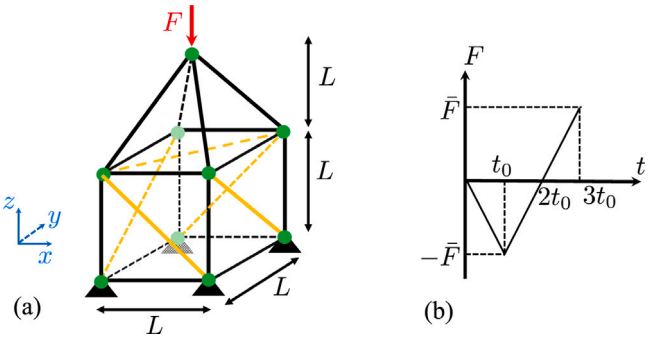


Fig. 22. (a) Truss system under (b) quasi-static cyclic loading F. The horizontal axis in (b) indicates the step number. We set $\bar{F} = 3$ kN, $t_0 = 20$, and the entire loading–unloading takes 60 steps.

6.2. Nonlinear truss system

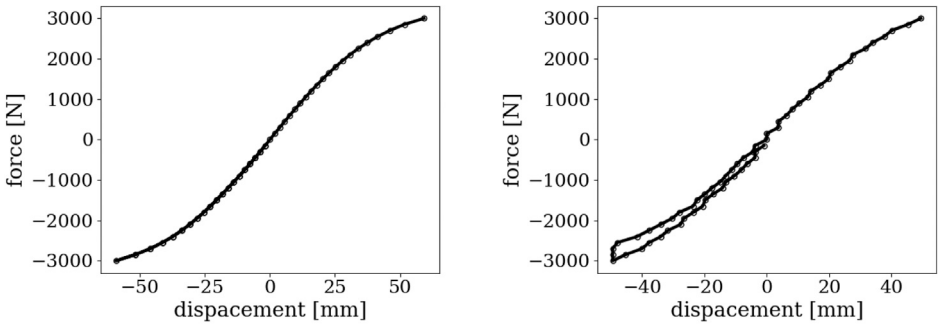
In this problem, we compare the efficiency, robustness, and accuracy of the proposed method against the nearest neighbor projection for a 3D truss structure that undergoes quasi-static, elastic loading–unloading, as shown in Fig. 22. The structure is loading at the top node by a linearly increasing compressive force to reach the maximum force $\bar{F}$, then the compressive force is linearly unloaded up to a maximum tensile force $\bar{F}$ as shown in Fig. 22(b). The continuum elasticity equations can be straightforwardly simplified for 3D truss elements, hence the details of the derivation are omitted for the sake of brevity; interested readers can follow reference books such as Fish and Belytschko (2007) and Belytschko et al. (2014).

The material database is similar to the complete data used in the previous problem (see Fig. 13), and we use the same neural network map function as the previous problem. The global optimization parameters and local ones (in the case of the original distance minimization method) are the same as the previous problem, i.e., $C = S^{-1} = 42694.67$ MPa.

Fig. 23 depicts the force–displacement plots obtained by the proposed data-driven scheme and the original distance minimization method. Interestingly, the original distance minimization method shows spurious hysteresis behavior in the unloading stage which could be wrongly considered as an irreversible mechanism symptom by a not experienced practitioner, while the proposed scheme in this paper accurately predicts the smooth elastic behavior.

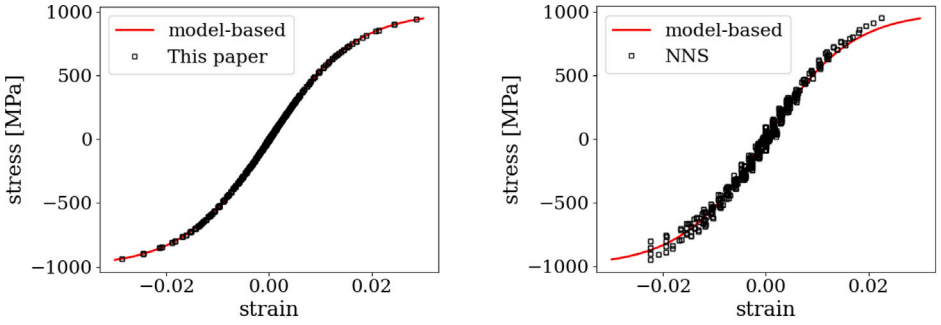
The strain–stress pairs in each bar element in the truss system during the entire loading history are plotted in Fig. 24 to investigate the offset between the estimated material states (dot points) and the underlying material constitutive manifold (red line). As these plots suggest, the proposed scheme finds the material states that are truly on the underlying material manifold. However, the original distance minimization method cannot precisely recover the nonlinear behavior. Notice that this performance issue of the original distance minimization is not unexpected as it is known as a data demanding method due to its minimal assumption about the constitutive model (Bahmani and Sun, 2021a). In other words, it can converge to the true behavior in the big data regimes (Kirchdoerfer and Ortiz, 2016; Conti et al., 2018).

The stress values in each bar element for the maximum compressive and tensile applied forces are shown in Fig. 25 which are obtained by three solvers: the classical model-based method, the introduced projection method herein, and the original distance minimization method. There is a good agreement between the model-based method and the method introduced in this paper, while the original distance minimization method predicts inaccurate stress values in some elements; e.g., see left and right elements of the top Pyramid.



(a) manifold embedding projection (this work) **(b)** nearest neighbour projection

Fig. 23. Force–displacement relation for the top node during the loading history via (a) the introduced projection and (b) the nearest neighbor projection. The introduced approach does not suffer from a possible spurious history dependence for an elastic material, in the limited data regime.



(a) introduced projection **(b)** nearest neighbour projection

Fig. 24. Strain–stress values obtained at each finite element during the loading history via (a) the introduced projection and (b) the nearest neighbor projection. The founded material states by the introduced scheme (a) follow the underlying, hidden material manifold, especially in the nonlinear regions.

6.3. Nonlinear elasticity

Here, we show the effectiveness of the proposed method compared with the original distance minimization method in dealing with 2D nonlinear elasticity problems. The problem domain and boundary conditions are depicted in Fig. 26 where the bottom is fixed from horizontal and vertical movements and a uniform vertical displacement $u_y = 0.1$m is applied over the top edge.

We synthesize a database consisting of 1000 data points sampled from the following strain energy (Nguyen et al., 2020; Bahmani and Sun, 2021a):

$$\psi(\boldsymbol{\epsilon}) = \frac{1}{2}(1 + 2\epsilon_{kk} - 2\log(1 + \epsilon_{kk})) + \frac{1}{2}(\log(1 + \epsilon_{kk}))^2 + \epsilon_{ki}\epsilon_{ik}. \tag{29}$$

Plane strain condition is assumed and a regular grid is used to sample the strain components in the ranges $-0.335 \leq \epsilon_{11} \leq 0.0155$, $0.12 \leq \epsilon_{22} \leq 1$, and $-0.03 \leq \epsilon_{12} \leq 0.03$.

The mapping $\mathcal{F} : \mathbb{R}^6 \mapsto \mathbb{R}^6$ from the ambient space to the mapped space is performed by a single invertible layer consisting of three hidden `elu` layers with ten hidden units per layer. Neural network weights and biases are initialized by the uniform Kaiming approach.

The neural network parameters are found by the `ADAM` optimizer with an initial learning rate = 0.002. We use the `ReduceLROnPlateau` learning scheduler to adjust the learning rate every 50 iterations after the first 1000 iterations. The learning rate reduction factor is set to 0.91 with the minimum learning rate 1e−6. Before training, strain and stress data points are linearly normalized based on their maximum and minimum values to be positive and less than or equal to 1. The training performance is shown in Fig. 27.

The $\mathbb{C}$ parameter in the global optimization step is set to the Hessian of the strain energy at zero strain, and $\mathbb{S}$ tensor is set to its inverse. The same parameters are used in the local optimization for the original distance minimization method.

In Fig. 28, we show the predicted displacement, strain, and stress contours by the introduced manifold method in this paper, the original distance minimization method, and the classical model-based method. Although the improvement in the displacement field predictions of the manifold approach over the nearest neighbor projection is marginal, the manifold method considerably enhances the strain and stress field predictions, both in terms of the magnitude of the errors and the symmetry of the solutions.

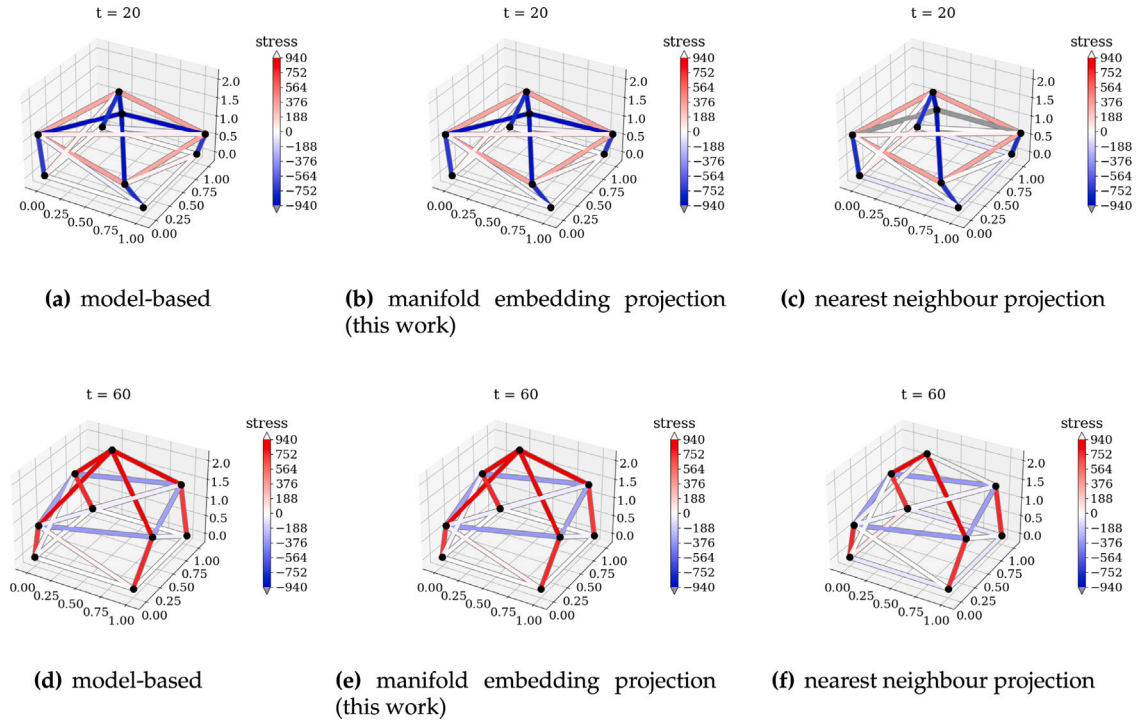


Fig. 25. A comparison between solution fields obtained by the proposed scheme and nearest neighbor projection onto the database. In (c) and (f), the brute-force distance minimization scheme provides wrong stress states for the top left and right bars.

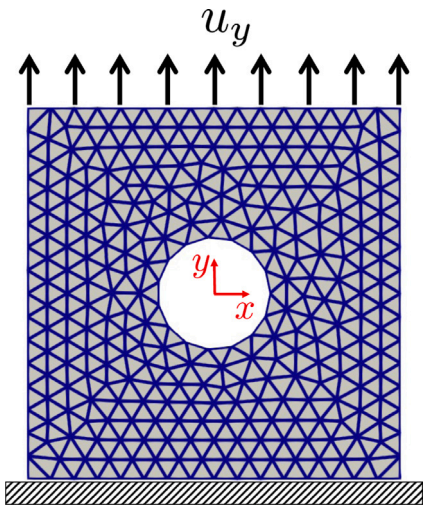


Fig. 26. A 1m-by-1m plane strain square with a circular void of 0.15 m radius at the center. The domain is discretized by 545 triangular finite elements.

The strain field obtained via the original distance minimization method (shown in Fig. 28) indicates that the discrete search does not yield a strain field that even qualitatively describes the overall characteristics of the problem, e.g., strain concentration around the hole imperfection. Notice that this is not unexpected since the amount of data used in this problem is very limited; 1000 data points to sample a 6-dimensional phase space is insufficient for the original distance minimization scheme. Furthermore, it is also not difficult to see that the solutions obtained from the nearest neighbor projection may lose the symmetry due to the bias induced by the limited choice of data pointed. This limitation can be overcome or at least suppressed by the usage of embedded space for projection where each query point is projected onto a hyperplane instead of a point from the material database.

The proposed method is also conceptually different from the classical constitutive laws approach in which we merely *implicitly* leverage the generalization power of deep neural networks in high-dimensional space (cf. Balestriero et al. (2021) and Belkin et al.

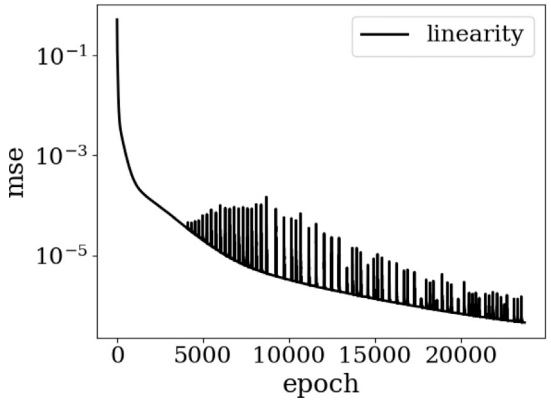


Fig. 27. Mean squared error (mse) of the linearity condition violation during the training epochs for the 2D nonlinear elasticity data. The training is performed in a full-batch manner with random shuffling at each epoch.

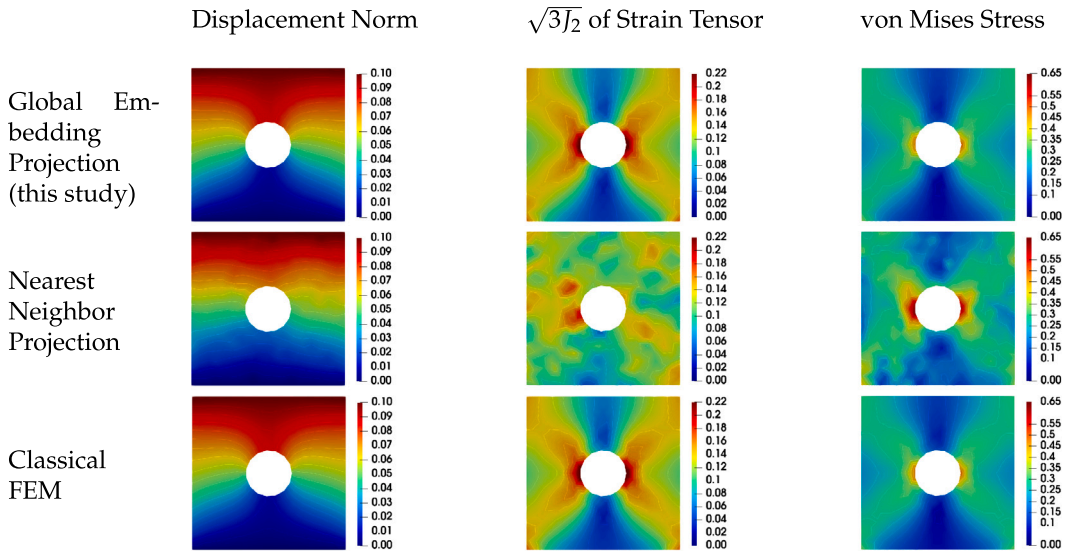


Fig. 28. A comparison between solution fields obtained by the two data-driven methods, i.e., global embedding (this study) and nearest neighbor projections, versus classical model-based nonlinear FEM (as ground truth). The simulations are performed under the plane strain condition.

(2019)) to gain knowledge on the manifold and utilize this acquired knowledge to generate a more precise notion of distance. We did not explicitly introduce a specific form or equations to curve-fitting the data. As such, the resultant paradigm remains general and applicable for different mechanics problems.

In Fig. 29, for the nearest neighbor projection method, the amount of data is gradually increased until we reach to almost similar error (for nodal displacements compared with those obtained from classical model-based finite element as the ground-truth) obtained by the global embedding projection method that utilizes a database consisting of only 10^3 data points. Notice that (1) the classical approach needs 3 orders of magnitude more data than the proposed scheme to obtain a similar error in the displacement field prediction. (2) assuming the availability of this amount of data, the simulation time (see Remark 9) is 1 order of magnitude more than the proposed scheme here. (3) the proposed scheme is almost insensitive to the random initialization, which increases robustness.

Remark 9. The simulation time in this problem refers to the actual boundary value problem simulation (Algorithm 1) obtained by the same finite element solver. The only difference is the scheme used in the local projection step. The solver is written in Python and was run on a MacBook laptop with a Quad-Core Intel Core i5 processor running at 1.4 GHz using 16 GB of RAM.

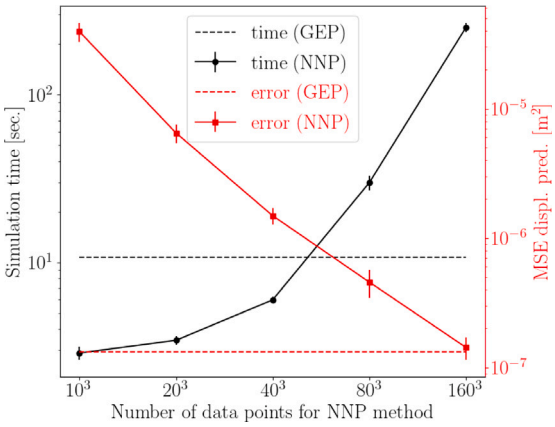


Fig. 29. Simulation time (left axis) and displacement error (right axis) comparisons between the global embedding projection (GEP) and nearest neighbor projection (NNP) methods. The horizontal axis demonstrates the different amounts of data used in the NNP simulations. Vertical bars at each point show 1.5 standard deviations among 10 randomly initialized simulations for a fixed database. The markers indicate the average among these 10 simulations. For the GEP method, we use the database with 10^3 data points, however, 10 randomly initialized simulations are also executed for this method. The average simulation time and displacement error for the GEP method are plotted by dashed black and red lines, respectively. The final solution by the GEP method is almost insensitive to the initialization (zero standard deviation in calculated errors). However, the mean and standard deviation values of simulation time for the GEP method are 10.7 and 0.3 s, respectively.

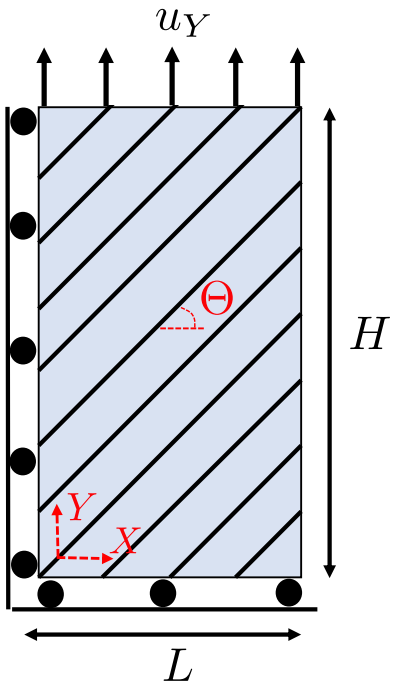


Fig. 30. Uni-axial test domain with length $L = 1$ mm and height $H = 2$ mm. The domain is assumed to be reinforced by fibrous materials orientated at Θ degree inclined from the horizontal axis. The applied load is set to $u_Y = 0.7$ mm, and the sample remains in plane-strain condition during the experiment.

6.4. Finite-strain anisotropic elasticity

In this example, we showcase the application of the proposed method for capturing the transverse isotropic elastic behavior of a solid in the finite-strain regime. As shown in Fig. 30, the uni-axial test is simulated for a synthesized material database populated from the transversely isotropic St. Venant hyperelasticity model (Bonet and Burton, 1998) with the following strain energy functional:

$$\psi = \frac{1}{2}\lambda E_{II}E_{JJ} + \mu E_{IJ}E_{IJ} + \left(\alpha + \beta(I_C - 3) + \gamma(IV_C - 1)\right)(IV_C - 1) - \frac{1}{2}\alpha(V_C - 1) \tag{30}$$

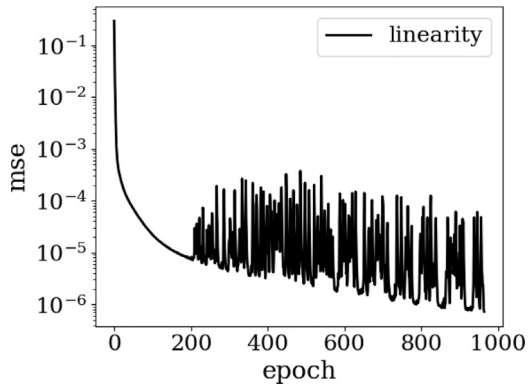


Fig. 31. The mean squared error (mse) of the linearity condition violation during the training epochs for the anisotropic hyperelastic data when the fiber orientation is 45 degrees. The training is performed with batch sizes of 2000 and random shuffling at each epoch.

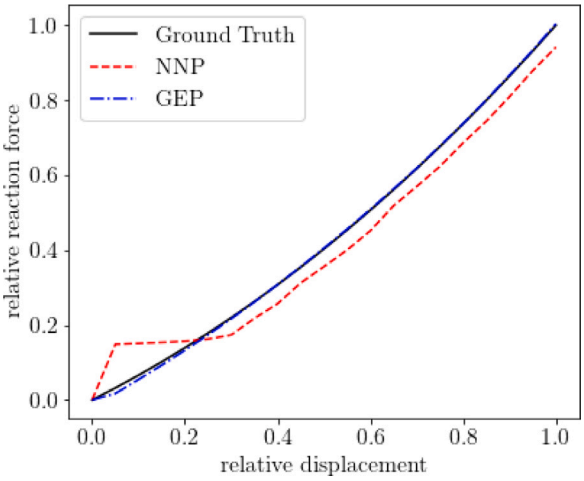


Fig. 32. Displacement–force curve for different methods during 20 loading steps in the uni-axial test.

where $I_C = C_{KK}$ is the first invariant of the right Cauchy–Green deformation tensor $\boldsymbol{C} = \boldsymbol{F}^T \cdot \boldsymbol{F}$, and λ, μ, α, β, and γ are material properties. The fourth and fifth pseudo invariants of $\boldsymbol{C}$ are defined as $IV_C = \boldsymbol{A} \cdot \boldsymbol{C}\boldsymbol{A}$, and $V_C = \boldsymbol{A} \cdot \boldsymbol{C}^2\boldsymbol{A}$, respectively, where the unit vector $\boldsymbol{A}$ represents the principal fiber orientation of orthotropy in the reference undeformed configuration. The database is generated by uniformly sampling of strains in ranges $E_{XX} \in [-0.36, 0.33]$, $E_{YY} \in [0.21, 0.75]$, and $E_{XY} \in [-0.85, 0.1]$ with the number of samples $41 \times 41 \times 21$. In the data generation process, we use $\alpha = -0.4$ GPa, $\beta = -0.025$ GPa, $\gamma = 0.025$ GPa, $\mu = 0.4$ GPa, $\lambda = 0.6$ GPa; these material properties correspond to an orthotropic material with the Elastic modulus in the fiber direction two times the host matrix counterpart and the same Poisson's ratio 0.25 for both fiber and matrix constituents (Bonet and Burton, 1998). The fiber orientation is set to 45 degrees.

The invertible neural network has a coupling layer with three internal layers and 30 hidden neurons per layer. The initial ADAM learning rate is set to 0.005, and the learning decay scheduler is activated after 500 epochs. The rest of the hyperparameters is kept the same as in the previous example. The training performance plots are shown in Fig. 31.

Displacement and Lagrange multiplier fields in the weak statements, i.e., Eqs. (10) and (11), are discretized with the standard linear basis functions via the Bubnov–Galerkin method. 812 uniform triangular elements discretize the undeformed reference domain, and the single point quadrature rule is used to calculate the underlying integrations. The simulations are conducted for 20 loading steps starting from zero to 0.7 mm applied vertical displacement. The parameter $\mathbb{C}^E = (\mathbb{S}^S)^{-1}$ is set equal to the Hessian tensor of the isotropic St. Venant model with $\lambda = 0.6$ GP and $\mu = 0.4$ GP.

The vertical displacement versus reaction force curve on the loading side during the simulations is plotted in Fig. 32. There is a good agreement between both data-driven methods and the benchmark. However, the GEP method predictions are slightly better than the NNP method, especially in the initial loading steps. The benchmark solution is obtained by the classical model-based FEM.

Fig. 33 compares strain and stress responses at the last loading step. The non-homogeneous deformation of this anisotropic material is captured accurately for both methods compared with the benchmark. However, the stress field prediction is less accurate

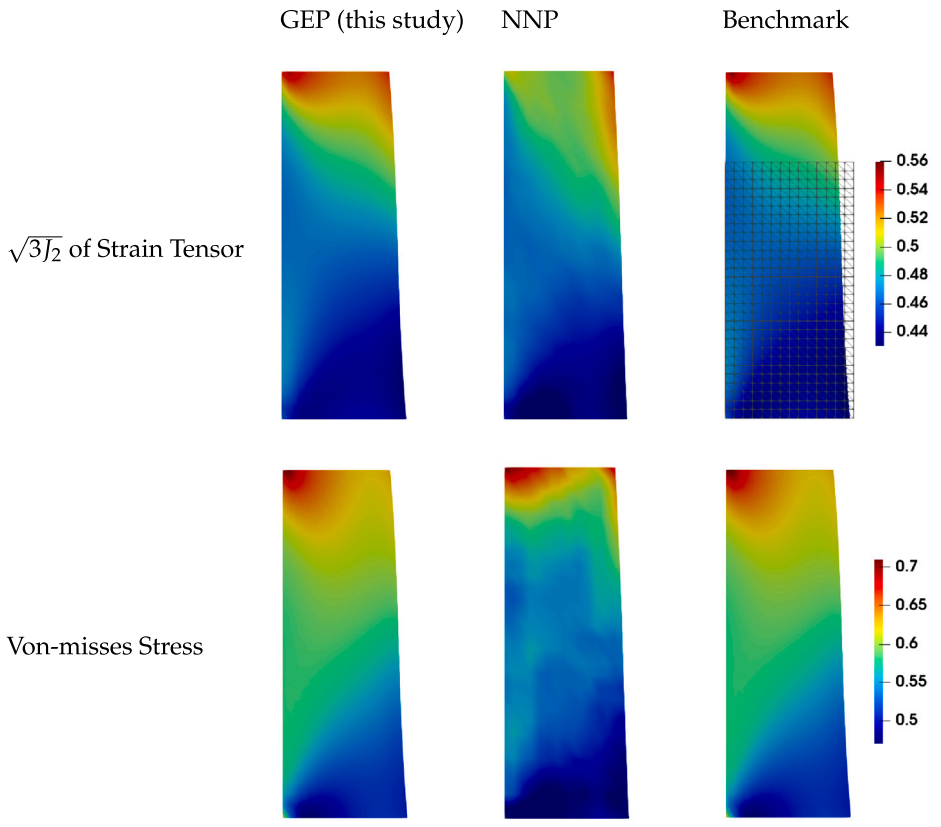


Fig. 33. A comparison between solution fields obtained by the two data-driven methods, i.e., global embedding (this study) and nearest neighbor projections, versus classical model-based nonlinear FEM (as the benchmark). The simulations are performed under the plane strain condition. The computational domain is the same for all methods and is shown in the top right contour image.

for the NNP method. Accurate stress and strain fields are necessary for failure or buckling analysis since it directly affects the localization zone and stress or strain redistribution in the post localization regime. Notice that the observed deficiency can be resolved if the amount of data is increased.

7. Conclusions

We introduce a data-driven approach that employs the invertible neural network to embed nonlinear constitutive manifolds onto a hyperplane. This treatment provides a distance measure more consistent with the intrinsic property of the material data and therefore enables more robust data-driven predictions when available data is limited. By mapping data points onto a hyperplane, the distance minimization algorithm may leverage the flatness and the continuous nature of the hyperplane to significantly simplify the projection step of the data-driven paradigm and bypass the cumbersome combinatorial search that may bear an increasing cost with the increasing amount of data.

The specific choice of norms equipped by the phase space of material data can be considered as an inductive bias (an assumption that is used to predict the output of given input it has not encountered) that affects the quality of the predictions. Not surprisingly, inductive bias does not only persists in training neural network constitutive law, symbolic regression but is often introduced in hand-crafting a constitutive law (e.g., Occam's razor, assumptions of smoothness, convexity), even though it is not always precisely recognized and consciously understood in the literature. The manifold embedding enables us to refocus on generating the inductive bias that helps us interpret and examine the **geometry** of the data while taking advantage of the simplicity of the Euclidean distance measure only available in the Euclidean space. This apparently subtle difference is nevertheless a crucial step for a wide spectrum of tasks highly relevant for physics and mechanics predictions because insights on the geometry of the data may play an important role in the quality and even correctness of these predictions. Examples of these applications include uncertainty quantification (Giovanis and Shields, 2018), de-noising (Lyu et al., 2019; Fefferman et al., 2020), and predictions with missing data (Mishne et al., 2019).

There are also a few other research directions that are important but have not yet been explored in this paper, such as the construction of isometric embedding to directly measure the geodesic distance between data points, the search of optimal hyperparameters for the offline training, and the de-noising strategy on manifolds. Note that the proposed global embedding method is not only useful for delivering model-free simulation results, but can be used as a means to encrypt material data in the mapped

database. In such a case, the trained neural network can be used as a key to unlock the material data into interpretable data whereas the true identity of the materials can be masked in the mapped database. The manifold embedding can be used for applications that require proprietary data, or data of privacy concerns. These research directions will be pursued in the future but are considered out of the scope of this work.

CRediT authorship contribution statement

Bahador Bahmani: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing – original draft. **WaiChing Sun:** Conceptualization, Methodology, Validation, Resource, Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We thank the two anonymous reviewers for their insightful feedback and helpful suggestions. Fruitful discussions with Ran Ma are gratefully acknowledged. The authors are supported by the National Science Foundation, USA under grant contracts CMMI-1846875 and OAC-1940203, and the Dynamic Materials and Interactions Program from the Air Force Office of Scientific Research, USA under grant contracts FA9550-21-1-0391 and FA9550-21-1-0027. These supports are gratefully acknowledged. The views and conclusions contained in this document are those of the authors, and should not be interpreted as representing the official policies, either expressed or implied, of the sponsors, including the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

Appendix A. Manifold embedding in machine learning

We aim to leverage the expressibility of neural networks to find a global Euclidean embedding with desirable properties. Hence, the first important question is the existence of such an embedding.

Whitney embedding theorem (Whitney, 1936, 1944) proves the existence of a function $\mathcal{F} : \mathcal{M} \mapsto \mathbb{R}^n$ that maps a sufficiently smooth manifold $\mathcal{M}$ with intrinsic dimension m to a vector (Euclidean) space of dimension $n \geq n_{\text{whitney}}$ where $n_{\text{whitney}} = 2\,\text{m}$, provided that $\mathcal{M}$ is not a real projective space. Nash proves the existence of such a mapping even with the isometry (distance-preserving) restriction for $n \geq n_{\text{nash}}$ where $n_{\text{nash}} > n_{\text{whitney}}$, known as Nash–Kuipe embedding theorem (Nash, 1954; Kuiper, 1955). Günther finds a tighter lower bound on the dimensionality of the Euclidean space $n_{\text{nash}} \geq \max\{m(m+1)/2, m(m+3)/2+5\}$ in the Nash embedding set-up (Günther, 1991).

Although these theoretical results show the existence of such mappings, the remaining question is the feasibility of finding such mappings from an algorithmic perspective. Baraniuk and Wakin (2009) and Clarkson (2008) propose algorithms (with theoretical guarantees) based on the random projection methods to achieve an approximate isometric mapping. Also, some previous studies empirically show the feasibility of finding such mapping functions such as Isomap (Tenenbaum et al., 2000), Locally Linear Embedding (Roweis and Saul, 2000), Hessian Eigenmapping (Donoho and Grimes, 2003), Laplacian Eigenmaps (Belkin and Niyogi, 2003), Local Tangent Space Alignment (Zhang and Zha, 2004), Local Fisher's Discriminant Method (Sugiyama, 2007), Riemannian Manifold Learning (Lin and Zha, 2008), and Local Manifold Learning-Based k-Nearest-Neighbor (Ma et al., 2010) among many others.

The scope of our work remains in the Whitney embedding setup, which is less restrictive than Nash–Kuipe embedding that requires the isometry condition. Notice that, although the isometry property is an essential condition to preserve the geometrical structure, there is a trade-off between the dimensionality of the target Euclidean space and the isometry restriction, i.e., $n_{\text{nash}} > n_{\text{whitney}}$. Also, constructing a proper mapping function that preserves metrics between two spaces is more challenging and ongoing research.

A.1. Related works on machine learning for manifold data

Courty et al. (2017) propose a method based on the Siamese architecture (Chopra et al., 2005) to approximately embed the Wasserstein metric into the Euclidean metric. This method finds the backward map from the Euclidean metric to the initial Wasserstein metric in an autoencoder fashion (Hinton and Salakhutdinov, 2006) that is approximately invertible. A similar idea is used for point cloud embedding to speed up the Wasserstein distance calculations (Kawano et al., 2020). Xiao et al. (2018) embed sub-samples of datasets with an arbitrary metric into the Euclidean metric to address mode collapse for generative adversarial networks (GANs) (Goodfellow et al., 2014). Bramburger et al. (2021) aim to parametrize the Poincaré map by a deep autoencoder architecture that enforces invertibility as a soft constraint.

Otto and Rowley (2019), Lusch et al. (2018) and Gin et al. (2021) use deep autoencoder architecture to find a coordinate transformation that linearizes a nonlinear partial differential equation (PDE). (Lee and Carlberg, 2020) use deep convolutional

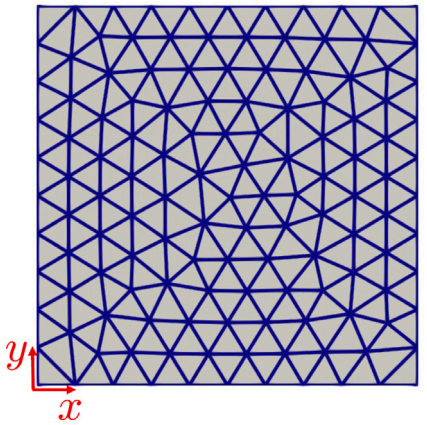


Fig. 34. The square domain of length 1 m for the heat conduction problem with pure Dirichlet boundary and non-zero heat source. The domain is discretized by 226 triangular finite elements.

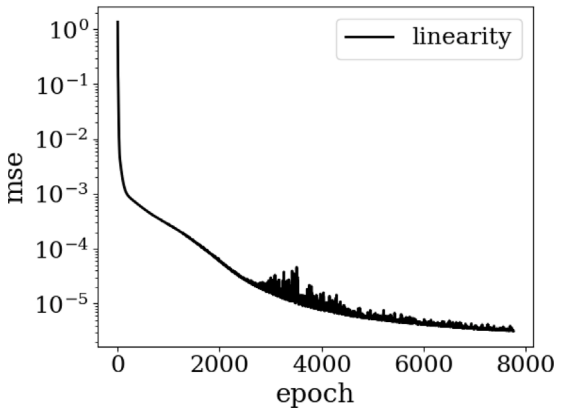


Fig. 35. Mean squared error (mse) of the linearity condition violation during the training epochs for the 2D nonlinear heat transfer data. Random mini-batches of the size 200 are used during the training.

autoencoders to project a dynamical system onto a nonlinear manifold of fewer dimensions instead of the classical approach of finding a linear subspace. Kim et al. (2021) construct a reduced-order model based on the autoencoder architecture with shallow multilayer perceptrons (MLPs) to find a proper linear subspace for a nonlinear PDE that describes advection-dominated phenomena. To the best of our knowledge, the current work is the first paper that leverages the bijectivity of the invertible neural network to generate the desired embedding space for the data-driven paradigm.

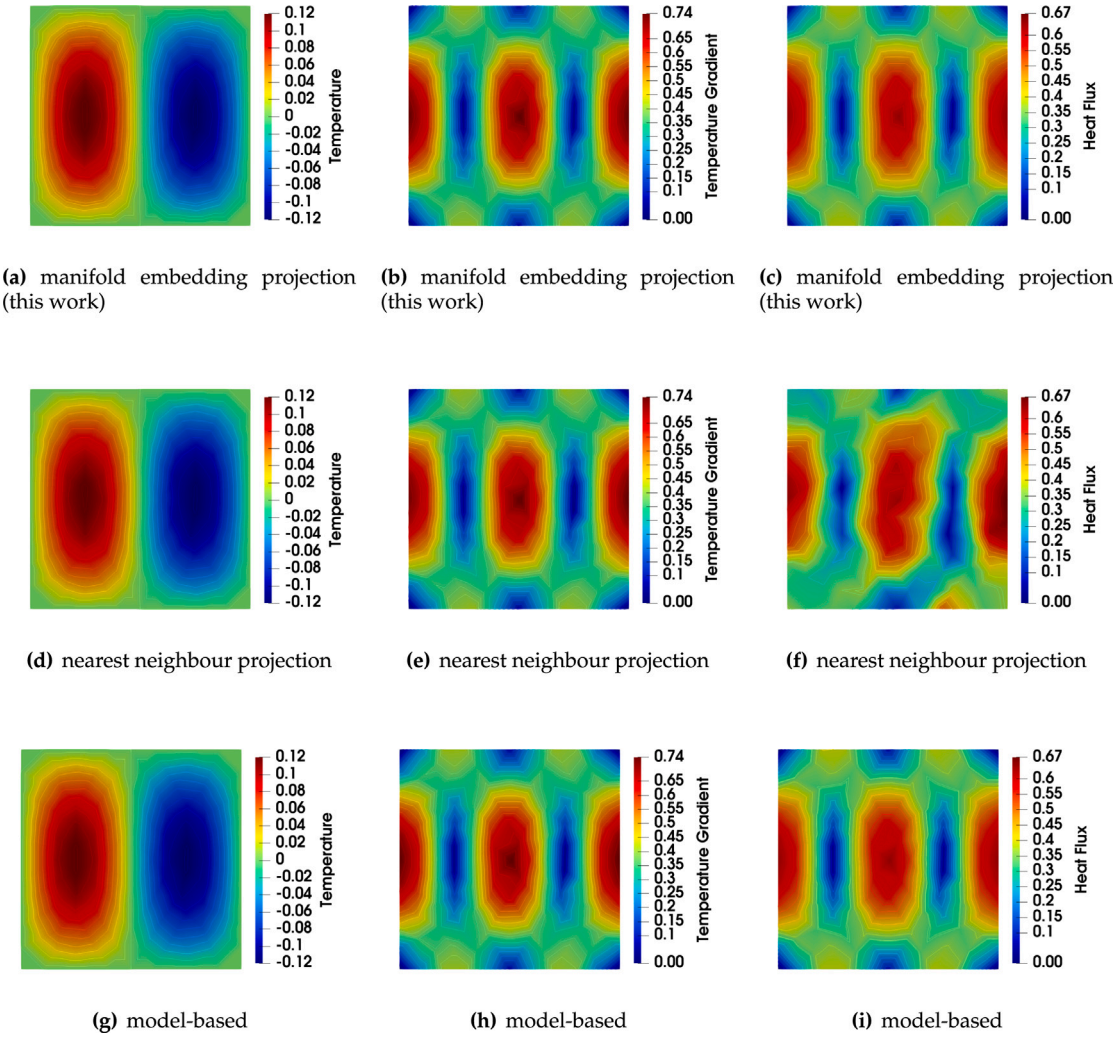
Appendix B. Terminology for data-driven mechanics

Definition 1 (*Vector Space*)**.** A vector space is a set of objects that are closed under vector addition and scalar multiplication with commutativity, associativity, additive identity, additive inverse, multiplicative identity, and distributive properties axioms Axler (1997).

Definition 2 (*Hyperplane*)**.** A hyperplane is a set of points (objects) $\boldsymbol{p} \in \mathbb{R}^n$ that satisfy $\boldsymbol{c} \cdot \boldsymbol{p} = 0$ for a non-zero point $\boldsymbol{c}$ and any arbitrary point $\boldsymbol{p}$ belongs to the hyperplane. The vector space's axioms (properties) can be easily verified for hyperplane; hence, it is a vector subspace. The vector $\boldsymbol{c}$ indicates the normal direction to the hyperplane.

Definition 3 (*Equilibrium Manifold*)**.** The set of all equilibrium states belongs to a manifold called equilibrium manifold. Particularly, in this work, we assume this manifold (denoted as C) is smooth and continuous.

Definition 4 (*Material Database*)**.** A set of multi-dimensional data points (e.g., pair of strain–stress tensors in solid mechanics) collected from experiments or finer-scale simulations, denoted as D that records the local constitutive responses.



(a) manifold embedding projection (this work) **(b)** manifold embedding projection (this work) **(c)** manifold embedding projection (this work)

(d) nearest neighbour projection **(e)** nearest neighbour projection **(f)** nearest neighbour projection

(g) model-based **(h)** model-based **(i)** model-based

Fig. 36. A comparison between solution fields (temperature, temperature gradient magnitude, and heat flux magnitude) obtained by (a–c) the proposed projection, (d–f) nearest neighbor projection, and (g–i) classical model-based nonlinear finite element solver. The introduced scheme improves heat flux prediction.

Definition 5 (*Mapped Material Database*)**.** A set of new points obtained after transforming (mapping) the points from the material database onto a hyperplane, which is denoted as $\hat{\mathcal{D}}$. We use the terms transformation and mapping interchangeably throughout this work.

Definition 6 (*Material State*)**.** A state (denoted as z^*) (e.g., a particular pair of strain–stress data) that belongs to the material database, i.e. $z^* \in \mathcal{D}$. In general, z^* is not necessarily admissible in the equilibrium manifold $\mathcal{C}$.

Definition 7 (*Mapped Material State*)**.** A state (denoted as $\hat{z}$) that belongs to the mapped material database, i.e. $\hat{z} \in \hat{\mathcal{D}}$. $\hat{z}$ is admissible to a hyperplane generated from an invertible neural network that maps the material data point cloud onto a hyperplane, i.e. $\mathcal{F} : \mathcal{D} \to \hat{\mathcal{D}}$ with the corresponding inverse mapping $\mathcal{G} : \hat{\mathcal{D}} \to \mathcal{D}$.

Definition 8 (*Equilibrium State*)**.** A state (denoted as z) that satisfies the equilibrium equation, i.e. $z \in \mathcal{C}$. In general, z is not necessarily an element in the material database.

Appendix C. Nonlinear heat conduction benchmark

In this section, we provide an additional nonlinear heat condition problem as a benchmark. In this problem, a square domain is composed of a material of nonlinear thermal conductivity where the Dirichlet boundary condition is prescribed, as shown in Fig. 34.

Due to the simplicity of this boundary value problem, it was used in previous work such as Nguyen et al. (2020) as to verify the data-free paradigm. As such, we adopt this example in the appendix to examine the performance of the manifold embedding solver in the limited data scenario and compare it with the classical one based on the nearest neighbor projection.

At the steady-state, the heat conduction governing equation reads

$$\nabla^x \cdot \boldsymbol{q}(x, y) + s(x, y) = 0, \tag{31}$$

where $\boldsymbol{q}(x, y)$ and $s(x, y)$ are the heat flux, and the heat source/sink term, respectively. The manufactured solution for the temperature field (Nguyen et al., 2020)

$$T(x, y) = \frac{1}{2}\sin(2\pi x)y(1 - y), \tag{32}$$

satisfies the governing equations with the following source term

$$s(x, y) = 2\pi^2 y(1 - y)\sin(2\pi x)[1 - \tanh^2(\pi y(1 - y)\cos(2\pi x))] + \sin(2\pi x)[1 - \tanh^2(\frac{1}{2}\sin(2\pi x)(1 - 2y))], \tag{33}$$

when the constitutive model is chosen as

$$\boldsymbol{q} = \tanh(\nabla^x T). \tag{34}$$

For the global projection formulation, we follow the model-free derivation of the heat conduction problem (Nguyen et al., 2020). The extension of manifold embedding projection for the heat conduction problem in the phase space of $(\nabla^x T, \boldsymbol{q})$ is straightforward and omitted for the sake of brevity.

To populate the database from the assumed constitutive model in Eq. (34), a regular grid with size 20^2 is used for temperature gradient components in the ranges $-1 \leq \partial T/\partial x \leq 1$ and $-1 \leq \partial T/\partial y \leq 1$. The $\boldsymbol{C}$ parameter for global optimization is set to $0.42\boldsymbol{I}$ where $\boldsymbol{I}$ is the second-order identity tensor, and $\boldsymbol{S}$ tensor is set to its inverse as suggested in Nguyen et al. (2020) and Bahmani and Sun (2021a). The original distance minimization scheme requires similar parameters for the local optimization which is set equal to the global optimization parameters.

For the proposed scheme, we should first find an appropriate map function $\mathcal{F} : \mathbb{R}^4 \mapsto \mathbb{R}^4$ to perform operations between the original data space and the mapped space with the vector space properties. This step is done offline by training an invertible neural network map function. For this database, we design an invertible neural network consisting of four `elu` layers, each has ten hidden units. The uniform Kaiming approach initializes the network weights and biases.

`ADAM` optimizer iterates to find close to optimal parameters for the mapping function with an initial learning rate = 0.002. We use `ReduceLROnPlateau` learning scheduler to adjust the learning rate every 50 iterations after the first 500 iterations. The learning rate reduction factor is set to 0.91 with the minimum learning rate 1e−6. Before training, temperature gradient and heat flux data points are linearly normalized based on their maximum and minimum values to be positive and less than or equal to 1. The training performance is shown in Fig. 35.

As shown in Fig. 36, there is a good agreement between the proposed scheme and the ground-truth solution. The heat flux predictions by the original distance minimization method are less accurate than the proposed method in the limited data regime. Notice that, the spurious oscillation of the heat flux and the break of symmetry exhibited in the solution obtained from the original distance minimization scheme is attributed to the limited data available in the database. If the database is sufficiently large, both issues may become less severe and may converge to the benchmark solution. Kirchdoerfer and Ortiz (2016), Leygue et al. (2018), Nguyen et al. (2020), and Bahmani and Sun (2021a).

References

Ardizzone, Lynton, Kruse, Jakob, Wirkert, Sebastian, Rahner, Daniel, Pellegrini, Eric W., Klessen, Ralf S., Maier-Hein, Lena, Rother, Carsten, Köthe, Ullrich, 2018. Analyzing inverse problems with invertible neural networks. arXiv preprint arXiv:1808.04730.

Axler, Sheldon Jay, 1997. Linear Algebra Done Right, Vol. 2. Springer.

Badia, Santiago, Parks, Michael, Bochev, Pavel, Gunzburger, Max, Lehoucq, Richard, 2008. On atomistic-to-continuum coupling by blending. Multiscale Model. Simul. 7 (1), 381–406.

Bahmani, Bahador, Sun, WaiChing, 2021a. A kd-tree-accelerated hybrid data-driven/model-based approach for poroelasticity problems with multi-fidelity multi-physics data. Comput. Methods Appl. Mech. Engrg. 382, 113868.

Bahmani, Bahador, Sun, WaiChing, 2021b. Training multi-objective/multi-task collocation physics-informed neural network with student/teachers transfer learnings. arXiv preprint arXiv:2107.11496.

Baldi, Pierre, 2012. Autoencoders, unsupervised learning, and deep architectures. In: Proceedings of ICML Workshop on Unsupervised and Transfer Learning. JMLR Workshop and Conference Proceedings, pp. 37–49.

Balestriero, Randall, Pesenti, Jerome, LeCun, Yann, 2021. Learning in high dimension always amounts to extrapolation. arXiv preprint arXiv:2110.09485.

Baraniuk, Richard G., Wakin, Michael B., 2009. Random projections of smooth manifolds. Found. Comput. Math. 9 (1), 51–77.

Bardenet, Rémi, Brendel, Mátyás, Kégl, Balázs, Sebag, Michele, 2013. Collaborative hyperparameter tuning. In: International Conference on Machine Learning. PMLR, pp. 199–207.

Beitler, Jan Jetze, Sosnovik, Ivan, Smeulders, Arnold, 2021. PIE: Pseudo-invertible encoder. arXiv preprint arXiv:2111.00619.

Belkin, Mikhail, Hsu, Daniel, Ma, Siyuan, Mandal, Soumik, 2019. Reconciling modern machine-learning practice and the classical bias–variance trade-off. Proc. Natl. Acad. Sci. 116 (32), 15849–15854.

Belkin, Mikhail, Niyogi, Partha, 2003. Laplacian eigenmaps for dimensionality reduction and data representation. Neural Comput. 15 (6), 1373–1396.

Belytschko, Ted, Liu, Wing Kam, Moran, Brian, Elkhodary, Khalil, 2014. Nonlinear Finite Elements for Continua and Structures. John wiley & sons.

Bengio, Yoshua, LeCun, Yann, et al., 2007. Scaling learning algorithms towards AI. Large-Scale Kernel Mach. 34 (5), 1–41.

Bergstra, James, Bengio, Yoshua, 2012. Random search for hyper-parameter optimization. J. Mach. Learn. Res. 13 (2).

Bishop, Christopher M., 2006. Pattern recognition. Mach. Learn. 128 (9).
Bonet, J., Burton, A.J.0944, 1998. A simple orthotropic, transversely isotropic hyperelastic constitutive equation for large strain computations. Comput. Methods Appl. Mech. Engrg. 162 (1–4), 151–164.
Bramburger, Jason J., Brunton, Steven L., Kutz, J. Nathan, 2021. Deep learning of conjugate mappings. arXiv preprint arXiv:2104.01874.
Carrara, Pietro, De Lorenzis, Laura, Stainier, Laurent, Ortiz, Michael, 2020. Data-driven fracture mechanics. Comput. Methods Appl. Mech. Engrg. 372, 113390.
Chopra, Sumit, Hadsell, Raia, LeCun, Yann, 2005. Learning a similarity metric discriminatively, with application to face verification. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). 1, IEEE, pp. 539–546.
Clarkson, Kenneth L., 2008. Tighter bounds for random projections of manifolds. In: Proceedings of the Twenty-Fourth Annual Symposium on Computational Geometry. pp. 39–48.
Clevert, Djork-Arné, Unterthiner, Thomas, Hochreiter, Sepp, 2015. Fast and accurate deep network learning by exponential linear units (elus). arXiv preprint arXiv:1511.07289.
Conti, Sergio, Müller, Stefan, Ortiz, Michael, 2018. Data-driven problems in elasticity. Arch. Ration. Mech. Anal. 229 (1), 79–123.
Courty, Nicolas, Flamary, Rémi, Ducoffe, Mélanie, 2017. Learning wasserstein embeddings. arXiv preprint arXiv:1710.07457.
Cranmer, Miles, Sanchez Gonzalez, Alvaro, Battaglia, Peter, Xu, Rui, Cranmer, Kyle, Spergel, David, Ho, Shirley, 2020. Discovering symbolic models from deep learning with inductive biases. Adv. Neural Inf. Process. Syst. 33, 17429–17442.
Dinh, Laurent, Krueger, David, Bengio, Yoshua, 2014. Nice: Non-linear independent components estimation. arXiv preprint arXiv:1410.8516.
Dinh, Laurent, Sohl-Dickstein, Jascha, Bengio, Samy, 2016. Density estimation using real nvp. arXiv preprint arXiv:1605.08803.
Donoho, David L., Grimes, Carrie, 2003. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. Proc. Natl. Acad. Sci. 100 (10), 5591–5596.
Eggersmann, Robert, Kirchdoerfer, Trenton, Reese, Stefanie, Stainier, Laurent, Ortiz, Michael, 2019. Model-free data-driven inelasticity. Comput. Methods Appl. Mech. Engrg. 350, 81–99.
Eggersmann, Robert, Stainier, Laurent, Ortiz, Michael, Reese, Stefanie, 2021a. Efficient data structures for model-free data-driven computational mechanics. Comput. Methods Appl. Mech. Engrg. 382, 113855.
Eggersmann, Robert, Stainier, Laurent, Ortiz, Michael, Reese, Stefanie, 2021b. Model-free data-driven computational mechanics enhanced by tensor voting. Comput. Methods Appl. Mech. Engrg. 373, 113499.
Ehlers, Wolfgang, Scholz, Bernd, 2007. An inverse algorithm for the identification and the sensitivity analysis of the parameters governing micropolar elasto-plastic granular material. Arch. Appl. Mech. 77 (12), 911–931.
Fefferman, Charles, Ivanov, Sergei, Lassas, Matti, Narayanan, Hariharan, 2020. Reconstruction of a Riemannian manifold from noisy intrinsic distances. SIAM J. Math. Data Sci. 2 (3), 770–808.
Fish, Jacob, Belytschko, Ted, 2007. A First Course in Finite Elements. Wiley.
Fuchs, Alexander, Heider, Yousef, Wang, Kun, Sun, WaiChing, Kaliske, Michael, 2021. DNN2: A hyper-parameter reinforcement learning game for self-design of neural network based elasto-plastic constitutive descriptions. Comput. Struct. 249, 106505.
Gebhardt, Cristian Guillermo, Schillinger, Dominik, Steinbach, Marc Christian, Rolfes, Raimund, 2020. A framework for data-driven structural analysis in general elasticity based on nonlinear optimization: The static case. Comput. Methods Appl. Mech. Engrg. 365, 112993.
Ghaboussi, J., Garrett, Jr., J.H., Wu, Xiping, 1991. Knowledge-based modeling of material behavior with neural networks. J. Eng. Mech. 117 (1), 132–153.
Gin, Craig, Lusch, Bethany, Brunton, Steven L., Kutz, J. Nathan, 2021. Deep learning models for global coordinate transformations that linearise PDEs. European J. Appl. Math. 32 (3), 515–539.
Giovanis, Dimitris G., Shields, Michael D., 2018. Uncertainty quantification for complex systems with very high dimensional response using grassmann manifold variations. J. Comput. Phys. 364, 393–415.
Goodfellow, Ian, Bengio, Yoshua, Courville, Aaron, 2016. Deep Learning. MIT Press.
Goodfellow, Ian, Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron, Bengio, Yoshua, 2014. Generative adversarial nets. Adv. Neural Inf. Process. Syst. 27.
Günther, Matthias, 1991. Isometric embeddings of Riemannian manifolds. In: Proceedings of the International Congress of Mathematicians, Vols. I, II (Kyoto, 1990). Math. Soc. Japan, Tokyo, pp. 1137–1143.
He, Qizhi, Chen, Jiun-Shyan, 2020. A physics-constrained data-driven approach based on locally convex reconstruction for noisy database. Comput. Methods Appl. Mech. Engrg. 363, 112791.
He, Xiaolong, He, Qizhi, Chen, Jiun-Shyan, 2021. Deep autoencoders for physics-constrained data-driven nonlinear materials modeling. Comput. Methods Appl. Mech. Engrg. 385, 114034.
He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, Sun, Jian, 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1026–1034.
Heider, Yousef, Suh, Hyoung Suk, Sun, WaiChing, 2021. An offline multi-scale unsaturated poromechanics model enabled by self-designed/self-improved neural networks. Int. J. Numer. Anal. Methods Geomech. 45 (9), 1212–1237.
Hinton, Geoffrey E., Salakhutdinov, Ruslan R., 2006. Reducing the dimensionality of data with neural networks. Science 313 (5786), 504–507.
Hornik, Kurt, Stinchcombe, Maxwell, White, Halbert, 1989. Multilayer feedforward networks are universal approximators. Neural Netw. 2 (5), 359–366.
Hughes, Thomas J.R., Feijóo, Gonzalo R., Mazzei, Luca, Quincy, Jean-Baptiste, 1998. The variational multiscale method—a paradigm for computational mechanics. Comput. Methods Appl. Mech. Engrg. 166 (1–2), 3–24.
Ibanez, Rubén, Abisset-Chavanne, Emmanuelle, Aguado, Jose Vicente, Gonzalez, David, Cueto, Elias, Chinesta, Francisco, 2018. A manifold learning approach to data-driven computational elasticity and inelasticity. Arch. Comput. Methods Eng. 25 (1), 47–57.
Ibañez, Ruben, Borzacchiello, Domenico, Aguado, Jose Vicente, Abisset-Chavanne, Emmanuelle, Cueto, Elias, Ladeveze, Pierre, Chinesta, Francisco, 2017. Data-driven non-linear elasticity: constitutive manifold construction and problem discretization. Comput. Mech. 60 (5), 813–826.
Kanno, Yoshihiro, 2018a. Data-driven computing in elasticity via kernel regression. Theoret. Appl. Mech. Lett. 8 (6), 361–365.
Kanno, Yoshihiro, 2018b. Simple heuristic for data-driven computational elasticity with material data involving noise and outliers: a local robust regression approach. Jpn. J. Ind. Appl. Math. 35 (3), 1085–1101.
Kanno, Yoshihiro, 2021. A kernel method for learning constitutive relation in data-driven computational elasticity. Jpn. J. Ind. Appl. Math. 38 (1), 39–77.
Karapiperis, Konstantinos, Ortiz, Michael, Andrade, José E., 2021. Data-driven nonlocal mechanics: Discovering the internal length scales of materials. Comput. Methods Appl. Mech. Engrg. 386, 114039.
Kawano, Keisuke, Koide, Satoshi, Kutsuna, Takuro, 2020. Learning wasserstein isometric embedding for point clouds. In: 2020 International Conference on 3D Vision (3DV). IEEE, pp. 473–482.
Kim, Youngkyu, Choi, Youngsoo, Widemann, David, Zohdi, Tarek, 2021. A fast and accurate physics-informed neural network reduced order model with shallow masked autoencoder. J. Comput. Phys. 110841.
Kingma, Diederik P., Ba, Jimmy, 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
Kirchdoerfer, Trenton, Ortiz, Michael, 2016. Data-driven computational mechanics. Comput. Methods Appl. Mech. Engrg. 304, 81–101.
Kirchdoerfer, Trenton, Ortiz, Michael, 2017. Data driven computing with noisy material data sets. Comput. Methods Appl. Mech. Engrg. 326, 622–641.
Kirchdoerfer, Trenton, Ortiz, Michael, 2018. Data-driven computing in dynamics. Internat. J. Numer. Methods Engrg. 113 (11), 1697–1710.

Klein, Aaron, Falkner, Stefan, Bartels, Simon, Hennig, Philipp, Hutter, Frank, 2017. Fast bayesian optimization of machine learning hyperparameters on large datasets. In: Artificial Intelligence and Statistics. PMLR, pp. 528–536.
Kuiper, Nicolaas H., 1955. On C^1-isometric imbeddings. I. Indag. Math. (Proc.) 58, 545–556.
Lan, Liang, Wang, Zhuang, Zhe, Shandian, Cheng, Wei, Wang, Jun, Zhang, Kai, 2019. Scaling up kernel SVM on limited resources: A low-rank linearization approach. IEEE Trans. Neural Netw. Learn. Syst. 30 (2), 369–378.
Lee, Kookjin, Carlberg, Kevin T., 2020. Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders. J. Comput. Phys. 404, 108973.
Leygue, Adrien, Coret, Michel, Réthoré, Julien, Stainier, Laurent, Verron, Erwan, 2018. Data-based derivation of material response. Comput. Methods Appl. Mech. Engrg. 331, 184–196.
Lin, Tong, Zha, Hongbin, 2008. Riemannian manifold learning. IEEE Trans. Pattern Anal. Mach. Intell. 30 (5), 796–809.
Liu, Yang, Sun, WaiChing, Fish, Jacob, 2016. Determining material parameters for critical state plasticity models based on multilevel extended digital database. J. Appl. Mech. 83 (1).
Lusch, Bethany, Kutz, J. Nathan, Brunton, Steven L., 2018. Deep learning for universal linear embeddings of nonlinear dynamics. Nature Commun. 9 (1), 1–10.
Lyu, He, Sha, Ningyu, Qin, Shuyang, Yan, Ming, Xie, Yuying, Wang, Rongrong, 2019. Manifold denoising by nonlinear robust principal component analysis. Adv. Neural Inf. Process. Syst. 32.
Ma, Li, Crawford, Melba M., Tian, Jinwen, 2010. Local manifold learning-based k-nearest-neighbor for hyperspectral image classification. IEEE Trans. Geosci. Remote Sens. 48 (11), 4099–4109.
Matouš, Karel, Geers, Marc G.D., Kouznetsova, Varvara G., Gillman, Andrew, 2017. A review of predictive nonlinear theories for multiscale modeling of heterogeneous materials. J. Comput. Phys. 330, 192–220.
Mishne, Gal, Chi, Eric, Coifman, Ronald, 2019. Co-manifold learning with missing data. In: International Conference on Machine Learning. PMLR, pp. 4605–4614.
Nash, John, 1954. C^1 isometric imbeddings. Ann. of Math. 383–396.
Nguyen, Lu Trong Khiem, Keip, Marc-André, 2018. A data-driven approach to nonlinear elasticity. Comput. Struct. 194, 97–115.
Nguyen, Lu Trong Khiem, Rambausek, Matthias, Keip, Marc-André, 2020. Variational framework for distance-minimizing method in data-driven computational mechanics. Comput. Methods Appl. Mech. Engrg. 365, 112898.
Otto, Samuel E., Rowley, Clarence W., 2019. Linearly recurrent autoencoder networks for learning dynamics. SIAM J. Appl. Dyn. Syst. 18 (1), 558–593.
Paszke, Adam, Gross, Sam, Massa, Francisco, Lerer, Adam, Bradbury, James, Chanan, Gregory, Killeen, Trevor, Lin, Zeming, Gimelshein, Natalia, Antiga, Luca, et al., 2019. Pytorch: An imperative style, high-performance deep learning library. Adv. Neural Inf. Process. Syst. 32, 8026–8037.
Probst, Philipp, Wright, Marvin N., Boulesteix, Anne-Laure, 2019. Hyperparameters and tuning strategies for random forest. Wiley Interdiscip. Rev. Data Min. Knowl. Discov. 9 (3), e1301.
Raghu, Maithra, Poole, Ben, Kleinberg, Jon, Ganguli, Surya, Sohl-Dickstein, Jascha, 2017. On the expressive power of deep neural networks. In: International Conference on Machine Learning. PMLR, pp. 2847–2854.
Raissi, Maziar, Perdikaris, Paris, Karniadakis, George E., 2019. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J. Comput. Phys. 378, 686–707.
Roache, Patrick J., 2002. Code verification by the method of manufactured solutions. J. Fluids Eng. 124 (1), 4–10.
Roweis, Sam T., Saul, Lawrence K., 2000. Nonlinear dimensionality reduction by locally linear embedding. Science 290 (5500), 2323–2326.
Sugiyama, Masashi, 2007. Dimensionality reduction of multimodal labeled data by local fisher discriminant analysis. J. Mach. Learn. Res. 8 (5).
Sun, WaiChing, Cai, Zhijun, Choo, Jinhyun, 2017. Mixed Arlequin method for multiscale poromechanics problems. Internat. J. Numer. Methods Engrg. 111 (7), 624–659.
Sun, WaiChing, Mota, Alejandro, 2014. A multiscale overlapped coupling formulation for large-deformation strain localization. Comput. Mech. 54 (3), 803–820.
Tenenbaum, Joshua B., De Silva, Vin, Langford, John C., 2000. A global geometric framework for nonlinear dimensionality reduction. Science 290 (5500), 2319–2323.
Timoshenko, Stephen, 1983. History of Strength of Materials: With a Brief Account of the History of Theory of Elasticity and Theory of Structures. Courier Corporation.
Vincent, Pascal, Larochelle, Hugo, Bengio, Yoshua, Manzagol, Pierre-Antoine, 2008. Extracting and composing robust features with denoising autoencoders. In: Proceedings of the 25th International Conference on Machine Learning. pp. 1096–1103.
Wang, Kun, Sun, WaiChing, 2016. A semi-implicit discrete-continuum coupling method for porous media based on the effective stress principle at finite strain. Comput. Methods Appl. Mech. Engrg. 304, 546–583.
Wang, Kun, Sun, WaiChing, 2018. A multiscale multi-permeability poroplasticity model linked by recursive homogenizations and deep learning. Comput. Methods Appl. Mech. Engrg. 334, 337–380.
Wang, Kun, Sun, WaiChing, Salager, Simon, Na, SeonHong, Khaddour, Ghonwa, 2016. Identifying material parameters for a micro-polar plasticity model via X-ray micro-computed tomographic (CT) images: lessons learned from the curve-fitting exercises. Int. J. Multiscale Comput. Eng. 14 (4).
Whitney, Hassler, 1936. Differentiable manifolds. Ann. of Math. 645–680.
Whitney, Hassler, 1944. The self-intersections of a smooth n-manifold in 2n-space. Ann. of Math. 220–246.
Williams, Francis, Gojcic, Zan, Khamis, Sameh, Zorin, Denis, Bruna, Joan, Fidler, Sanja, Litany, Or, 2021. Neural fields as learnable kernels for 3D reconstruction. arXiv preprint arXiv:2111.13674.
Xiao, Chang, Zhong, Peilin, Zheng, Changxi, 2018. Bourgan: Generative networks with metric embeddings. arXiv preprint arXiv:1805.07674.
Yu, Tianhe, Kumar, Saurabh, Gupta, Abhishek, Levine, Sergey, Hausman, Karol, Finn, Chelsea, 2020. Gradient surgery for multi-task learning. arXiv preprint arXiv:2001.06782.
Zhang, Zhenyue, Zha, Hongyuan, 2004. Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. SIAM J. Sci. Comput. 26 (1), 313–338.