Mechanics of Materials 180 (2023) 104630

Contents lists available at ScienceDirect

Mechanics of Materials

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

Mesh d-refinement: A data-based computational framework to account for complex material response

Sacha Wattel [a], Jean-François Molinari [a], Michael Ortiz [b,c], Joaquin Garcia-Suarez [a,*]

[a] Institute of Civil Engineering, Institute of Materials, École Polytechnique Fédérale de Lausanne (EPFL), CH 1015 Lausanne, Switzerland
[b] Division of Engineering and Applied Science, California Institute of Technology, 1200 E. California Blvd., Pasadena, CA 91125, USA
[c] Hausdorff Center for Mathematics, Universität Bonn, Endenicher Allee 60, 53115 Bonn, Germany

ARTICLE INFO

Keywords:
Data-driven computational mechanics
Hybrid formulation
Non-linearity
FEM-DD coupling
Model-free

ABSTRACT

Model-free data-driven computational mechanics (DDCM) is a new paradigm for simulations in solid mechanics. The modeling step associated to the definition of a material constitutive law is circumvented through the introduction of an abstract phase space in which, following a pre-defined rule, physically-admissible states are matched to observed material response data (coming from either experiments or lower-scale simulations). In terms of computational resources, the search procedure that performs these matches is the most onerous step in the algorithm. One of the main advantages of DDCM is the fact that it avoids regression-based, bias-prone constitutive modeling. However, many materials do display a simple linear response in the small-strain regime while also presenting complex behavior after a certain deformation threshold. Motivated by this fact, we present a novel refinement technique that turns regular elements (equipped with a linear-elastic constitutive law) into data-driven ones if they are expected to surpass the threshold known to trigger material non-linear response. We term this technique "data refinement", "d-refinement" for short. It works both with data-driven elements based on either DDCM or strain–stress relations learned from data using neural networks. Starting from an initially regular FEM mesh, the proposed algorithm detects where the refinement is needed and iterates until all elements presumed to display non-linearity become data-driven ones. Insertion criteria are discussed. The scheme is well-suited for simulations that feature non-linear response in relatively small portions of the domain while the rest remains linear-elastic. The method is validated against a traditional incremental solver (i.e., Newton–Raphson method) and we show that the d-refinement framework can outperform it in terms of speed at no loss of accuracy. We provide an application that showcases the advantage of the new method: bridging scales in architected metamaterials. For this application, we also succinctly outline how d-refinement can be used in conjunction with a neural network trained on microscale data.

1. Introduction

Data-driven computational mechanics (DDCM) is a rapidly-expanding research field. Data-driven solvers were originally devised to deal with problems in small-strains statics (Kirchdoerfer and Ortiz, 2016; Conti et al., 2018); extensions to dynamics (Kirchdoerfer and Ortiz, 2018; Salahshoor and Ortiz, 2023; Garcia-Suarez et al., 2022), large deformations (Platzer et al., 2021; Conti et al., 2020; Korzeniowski and Weinberg, 2022) and dissipative (inelastic) material response (Eggersmann et al., 2019; Carrara et al., 2022, 2021; Karapiperis et al., 2020, 2021) followed.

The data-driven paradigm relies on using "observed information" directly, avoiding the fitting/calibration exercise that turns these observations into a fully-defined mathematical function ("material constitutive model"). Thus, part of the efforts in the field have been focused on generating those datasets either from experimental observations (Leygue et al., 2018) or from microstructural simulations (Karapiperis et al., 2020, 2021; Korzeniowski and Weinberg, 2022). Further efforts are currently underway to maximize its numerical efficiency (Eggersmann et al., 2021a,b; Karapiperis et al., 2021; Korzeniowski and Weinberg, 2021). The mathematical foundations of the method have been solidly laid out both in small strains (Conti et al., 2018) and large deformations (Conti et al., 2020).

Arguably, the main benefit of the DDCM paradigm is that it enables circumventing the definition of intricate phenomenological laws, which are required to describe complex material response during numerical simulations (Kovachki et al., 2022). However, most materials do display a simple, consistent linear-elastic behavior as long as they remain in the

* Corresponding author.
E-mail address: joaquin.garciasuarez@epfl.ch (J. Garcia-Suarez).

https://doi.org/10.1016/j.mechmat.2023.104630
Received 15 December 2022; Received in revised form 7 March 2023; Accepted 9 March 2023
Available online 17 March 2023

small-strain regime, after which either non-linear or inelastic phenomena are expected to kick in. Here, we consider that "complex response" is anything other than linear and elastic. As linear-elastic simulations are both straightforward to implement and computationally efficient, it is therefore desirable to stick to them to solve as much of the simulation domain as possible. For this, one needs to devise a framework that can detect when an element surpasses a deformation threshold, and, once that trigger is indeed detected, proceed to enrich that element with a data-driven formulation. This would also redound on simulation speed, as it would reduce the number of phase-space searches, which remains the most time-consuming step in the algorithm, despite the fact that they are parallelizable.

A recent article of Yang et al. (2022) has been the first, to our knowledge, that combined data-driven and traditional elements in the same mesh. They used their coupling technique to analyze structures in which part of the main domain remains linear-elastic while a portion of it weakens due to an external factor (corrosion). Taking advantage of this coupling scheme, we introduce a new kind of mesh refinement technique that aims to improve both numerical efficiency and accuracy while maintaining the advantages of DDCM: "data refinement", "d-refinement" for short, that automatically turns regular elements into data-driven ones where and when necessary. This eases the computational implementation, as it avoids intricacies associated with numerically resolving explicitly either material non-linearities, inelastic mechanisms or both. D-refinement also improves accuracy: where constitutive modeling is more arduous, it replaces assumption-ridden phenomenological descriptions by data, while keeping the model in the region where it performs best (infinitesimal strain), thus also partially avoiding errors associated with dealing with a discrete material dataset (Kirchdoerfer and Ortiz, 2016). See Section 2.3.1 for further discussion on this point.

We will show that the d-refinement technique does not require load stepping in the absence of inelasticity. Since it combines both FEM and DDCM, the definition of the constants that metrize the space in DDCM (Kirchdoerfer and Ortiz, 2016; Karapiperis et al., 2021) is straightforward in d-refinement: we can make them equal to the elastic constants of the FEM linear material.

The d-refinement framework forges a better union between model-driven and data-driven ways of computing. It aspires to compete with incremental solvers based on tangent operators as the tool-of-choice when it comes to perform mechanics calculations that include either non-linear behavior or inelasticity or both. Furthermore, if the dataset comes from lower-scale simulations, d-refinement can also be regarded as an optimized FE^2 method (Feyel, 1999), in which (a) the microstructural response is obtained off-line before the simulation (Karapiperis et al., 2021; Korzeniowski and Weinberg, 2022) and (b) microstructure RVE response is directly used at the element level only when its linear-elastic range is exhausted.

The paper is structured as follows: Section 2 reviews the basics of DDCM and the "static" FEM-DD implementation, and introduces the d-refinement solving procedure. Section 3 validates the methodology via comparisons to conventional incremental solvers, we compare two kinds of systems: 3D trusses and 2D plane-stress elements. Section 4 presents an application of d-refinement to multiscale analysis. Starting from a material dataset, we use efficient data-driven simulations to characterize the mechanical response of the unit cell of an architected metamaterial, in terms of both linear-elastic constants and a dataset that includes non-linear response. Then, this information is used to study the K-field in a cracked macroscale sample (Shaikeea et al., 2022). After conclusions in Section 5, we showcase in an Appendix A how the d-refinement method can be modified to handle constitutive models learned from data via trained neural networks.

2. Methods

2.1. Data-driven computational mechanics

DDCM poses the solid mechanics boundary-value problem in an abstract phase space of work-conjugate variables (e.g., strain and stress), in which each element state is defined by a point in a "local" phase space, and the overall system phase space is the Cartesian product of that of its elements. The field equations (equilibrium and compatibility), along with the boundary conditions, represent a manifold embedded in phase space. The infinite set that contains all possible points that satisfy these constraints is termed E, and referred to as the "physically-admissible set". Traditionally, the constitutive law would amount to a graph in this space (Conti et al., 2018), classical solutions of the boundary-value problem would correspond to the intersection of the aforementioned manifold and the graph. However, in DDCM, the material behavior is represented by a "material discrete set", termed D. As the intersection of admissible set E and D is unlikely, this concept is replaced by the notion of "transversality" (Conti et al., 2018) between E and D.

The FEM-DD coupling scheme that we are to use and to introduce in the next section converges to the original DD implementation (Kirchdoerfer and Ortiz, 2016) when the whole mesh becomes data-driven; hence, for the sake of brevity, we introduce the coupling scheme algorithm directly in next section.

2.2. FEM-DD coupling

2.2.1. Introduction

We begin with a brief digression concerning terminology. Model-driven is a fair description of the current paradigm, which relies on phenomenological models of material response to close the problem: boundary conditions, force equilibrium and displacement compatibility equations need to be supplemented with a force-deformation relation. Nevertheless, using the acronym "MD" to refer to model-driven is deemed confusing, as both in the mechanics and in the material science communities it is associated with "molecular dynamics", a field where data-driven approaches are also possible (Bulin et al., 2022). We would rather use, at least in the context of solid and structural mechanics, the name "DD-FEM coupling" in lieu of "DD-MD coupling". Small caveats aside, the term is self-explanatory and, we believe, will be readily understood by researchers in any of the fields mentioned before. In a recent contribution by Korzeniowski and Weinberg (2022), the authors refer to the conventional DDCM way of computing as "DD-FEM" as it is "data-driven in a finite-element mesh". However, since we tend to associate the finite-element method not only with the discretization of the domain, but also with a way to model materials (that is why we would speak of "non-linear finite elements"), we believe our terminology to be appropriate.

The coupling implementation (Yang et al., 2022) divides the set of all elements in the numerical model (S) into two subsets: S_2 includes the indices of DD elements, while S_1 contains those of the regular ones ($S_2 \cap S_1 = \emptyset$ and $S_2 \cup S_1 = S$). The size of each subset, $|\cdot|$, is defined as the number of elements in it, thus $|S_2|+|S_1| = N_e$, where N_e is the total number of elements in the mesh. The concept of phase space distances remain unchanged for DD elements: first, a point in the local phase space Z_e (of the eth element, $e \in S_2$) is $\boldsymbol{z}_e = \{\boldsymbol{\epsilon}_e, \boldsymbol{\sigma}_e\} \in \mathbb{R}^{2N_c}$, where N_c is the number of relevant components of stress/strain (one for uni-axial elements, three for plane problems, six for 3D problems, and nine for 3D problems in generalized continua (Karapiperis et al., 2020)). This local phase space is equipped with a metric defining a norm for each point as

$$|\boldsymbol{z}_e|^2 = \frac{1}{2}\,\mathbb{C}_e\boldsymbol{\epsilon}_e \cdot \boldsymbol{\epsilon}_e + \frac{1}{2}\,\mathbb{C}_e^{-1}\boldsymbol{\sigma}_e \cdot \boldsymbol{\sigma}_e\,, \tag{1}$$

where $\mathbb{C}_e$ is a symmetric positive-definite matrix of constants not necessarily related to any material property.

Given that the global phase space is $Z = Z_1 \times Z_2 \ldots \times Z_{|S_2|}$, the norm associated to $\boldsymbol{z} \in Z$ can be taken as

$$|\boldsymbol{z}| = \left(\sum_{e \in S_2} w_e |\boldsymbol{z}_e| \right)^{1/2}, \tag{2}$$

where w_e is the volume of the eth element. The latter in turn provides a notion of distance between two points, $\boldsymbol{z}$ and $\boldsymbol{y}$, in phase space:

$$d(\boldsymbol{z}, \boldsymbol{y}) = |\boldsymbol{z} - \boldsymbol{y}|. \tag{3}$$

Note that the solution of traditional finite-element analysis also represents a point in a phase space, and hence the global solution of the system that contains both kinds of elements can be represented in a larger phase space (i.e., $Z_1 \times Z_1 \times \cdots \times Z_e$).

By means of the constitutive law, assumed linear and elastic, we can write $\boldsymbol{\sigma}_e = \mathbb{D}_e \boldsymbol{\epsilon}_e$ for every element s.t. $e \in S_1$, with $\mathbb{D}_e \in \mathbb{R}^{N_c \times N_c}$ containing the regular elastic constants of the eth element. Using the discretized compatibility relation, one can express the strain of an element in terms of the global nodal displacement vector, $\boldsymbol{u} \in \mathbb{R}^{N_{\text{dof}}}$ (N_{dof} is the total number of degrees of freedom in the mesh),

$$\boldsymbol{\epsilon}_e = \boldsymbol{B}_e \boldsymbol{u} \qquad \text{for } e = 1, \ldots, N_e, \tag{4}$$

$\boldsymbol{B}_e \in \mathbb{R}^{N_c \times N_{\text{dof}}}$ encapsulates the nodal connectivity and the shape functions used to perform nodal interpolation within each element. The global equilibrium equation is expressed as

$$\sum_{e \in S_2} w_e \boldsymbol{B}_e^\top \boldsymbol{\sigma}_e + \sum_{e \in S_1} w_e \boldsymbol{B}_e^\top \mathbb{D} \boldsymbol{B}_e \boldsymbol{u} = \boldsymbol{f}, \tag{5}$$

where $\boldsymbol{f} \in \mathbb{R}^{N_{\text{dof}}}$ is the nodal external force vector.

Notice that the displacements of the regular elements enter the nodal equilibrium equations along with the stress of the DD ones. Using Lagrange multipliers, $\boldsymbol{\eta} \in \mathbb{R}^{N_{\text{dof}}}$, these equilibrium constraints must be enforced concurrently with the distance minimization between points in the admissible set, $\boldsymbol{z} \in \text{E}$, and the material set, $\boldsymbol{z}^* \in \text{D} = \text{D}_1 \times \text{D}_2 \times \cdots \times \text{D}_{|S_2|}$, for the DD portion of the mesh (Kirchdoerfer and Ortiz, 2016). Thus, we introduce a functional

$$\Pi = d^2(\boldsymbol{z}, \boldsymbol{z}^*) + \boldsymbol{\eta} \cdot \left(\boldsymbol{f} - \sum_{e \in S_2} \boldsymbol{B}_e^\top \boldsymbol{\sigma}_e - \sum_{e \in S_1} \boldsymbol{B}_e^\top \mathbb{D} \boldsymbol{B}_e \boldsymbol{u} \right). \tag{6}$$

Compatibility is guaranteed by directly expressing, in the distance function, the strains $\boldsymbol{\epsilon}_\mathbf{e}$ with displacements (Eq. (4)). The functional obtained depends on $\boldsymbol{\sigma}_\mathbf{e}$, $\boldsymbol{u}$ and $\boldsymbol{\eta}$. Enforcing the stationarity of Π renders a problem that can be solved iteratively, until the state of the system that minimizes the distance between the solution and the admissible points in the material dataset is attained.

Setting every variation of Π to zero, we obtain

$$\delta \boldsymbol{u} \implies \sum_{e \in S_2} w_e \boldsymbol{B}_e^\top \mathbb{C}_e \left(\boldsymbol{B}_e \boldsymbol{u} - \boldsymbol{\epsilon}_e^* \right) - \sum_{e \in S_1} \boldsymbol{B}_e^\top \mathbb{D} \boldsymbol{B}_e \boldsymbol{\eta} = 0, \tag{7a}$$

$$\delta \boldsymbol{\sigma}_\mathbf{e} \implies \mathbb{C}_e^{-1} (\boldsymbol{\sigma}_\mathbf{e} - \boldsymbol{\sigma}_\mathbf{e}^*) = \boldsymbol{B}_e \boldsymbol{\eta}, \tag{7b}$$

$$\delta \boldsymbol{\eta} \implies \sum_{e \in S_2} w_e \boldsymbol{B}_e^\top \boldsymbol{\sigma}_e + \sum_{e \in S_1} w_e \boldsymbol{B}_e^\top \mathbb{D} \boldsymbol{B}_e \boldsymbol{u} = \boldsymbol{f}, \tag{7c}$$

which yields, after minor rearrangements of Eq. (7a) and introducing Eq. (7b) into Eq. (7c),

$$\sum_{e \in S_1} w_e \boldsymbol{B}_e^\top \mathbb{D}_e \boldsymbol{B}_e \boldsymbol{u} - \sum_{e \in S_2} w_e \boldsymbol{B}_e^\top \mathbb{C}_e \boldsymbol{B}_e \boldsymbol{\eta} = \sum_{e \in S_2} w_e \boldsymbol{B}_e^\top \mathbb{C}_e \boldsymbol{\epsilon}_e^*, \tag{8a}$$

$$\sum_{e \in S_1} w_e \boldsymbol{B}_e^\top \mathbb{D}_e \boldsymbol{B}_e \boldsymbol{u} + \sum_{e \in S_2} w_e \boldsymbol{B}_e^\top \mathbb{C}_e \boldsymbol{B}_e \boldsymbol{\eta} = \boldsymbol{f} - \sum_{e \in S_2} w_e \boldsymbol{B}_e^\top \boldsymbol{\sigma}_e^*, \tag{8b}$$

where we use the superindex "$*$" to highlight elements in the material dataset D. Given $\boldsymbol{z}_e^* = (\boldsymbol{\epsilon}_e^*, \boldsymbol{\sigma}_e^*) \in \text{D}_e \ \forall e \in S_2$, Eq. (8) can be solved for $\boldsymbol{u}$ and $\boldsymbol{\eta}$, and then physically-admissible strain and stresses for the DD elements can be computed as Yang et al. (2022)

$$\boldsymbol{\sigma}_e = \boldsymbol{\sigma}_e^* + \mathbb{D}_e \boldsymbol{B}_e \boldsymbol{\eta}, \tag{9a}$$

$$\boldsymbol{\epsilon}_e = \boldsymbol{B}_e \boldsymbol{u}. \tag{9b}$$

The subsequent solving of Eq. (8) and Eq. (9) is likened to a projection operation that takes the phase-space point $\boldsymbol{z}^*$ in D and projects it onto the admissible set: $P_E \boldsymbol{z}^* \in \text{E}$.

The next step of the algorithm involves projecting back onto the material set, what is achieved through an element-wise search that picks the point in D closest to the previously obtained $\boldsymbol{z} \in \text{E}$, and it is symbolically expressed as $P_D \boldsymbol{z} \in \text{D}$. The necessary notion of distance between points is provided by Eq. (3). In essence, this defines the simplest data-driven solver, which allows fixed-point iterations from the kth admissible DD solution $\boldsymbol{z}^k$ to the next one, i.e., $\boldsymbol{z}^{(k+1)} = P_E P_D \boldsymbol{z}^{(k)}$. The algorithm yields the final solution at the nth iteration if all the DD elements are assigned the same set of datapoints twice in a row, i.e., $\boldsymbol{z}^{*(n)} = P_D P_E \boldsymbol{z}^{*(n-1)} = \boldsymbol{z}^{*(n-1)}$.

The coupling is expected to converge with respect to the dataset size since the linear-elastic finite elements can be thought as DD ones in which $\boldsymbol{z} = \boldsymbol{z}^*$ always, and thus the theorems developed for pure DD meshes (Conti et al., 2018, 2020) apply.

2.2.2. Implementation

Algorithm 1 Fixed-point algorithm for DD-FEM coupling

Require: $\forall e = 1, \ldots, N_e$, compatibility matrices $\boldsymbol{B}_e$; $\forall e \in S_1$, $\mathbb{D}_e$ matrices containing elastic constants; $\forall e \in S_2$, local datasets D_e and $\mathbb{C}_e$ matrices containing distance constants; $\forall i = 1, \ldots, N_{\text{dof}}$, external forces f_i or boundary conditions.
(i) Set $k = 0$. Initial data assignation:
for $e \in S_2$ **do**
  Choose $\boldsymbol{z}_e^{*(0)} = (\boldsymbol{\sigma}_e^{*(0)}, \boldsymbol{\epsilon}_e^{*(0)}) \in \text{D}_e$ randomly
end for
(ii) Project onto E:
Solve in eq. (8) for $\boldsymbol{u}^{(k)}$ and $\boldsymbol{\eta}^{(k)}$
for $e \in S_2$ **do**
  Compute $\boldsymbol{\epsilon}_e$ and $\boldsymbol{\sigma}_e$ from eq. (9)
end for
(iii) Project onto D:
for $e \in S_2$ **do**
  Choose $\boldsymbol{z}_e^{*(k+1)} \in \text{D}_e$ closest to $\boldsymbol{z}_e^{(k)} \in \text{E}_e$
end for
(iv) Convergence test:
if $\boldsymbol{z}_e^{(k+1)} = \boldsymbol{z}_e^{(k)} \ \forall e \in S_2$ **then**
  Final displacement field $\boldsymbol{u} = \boldsymbol{u}^{(k)}$ **exit**
else
  $k \leftarrow k + 1$, **goto** (ii)
end if

Algorithm 1 presents the pseudo-code implementation of a coupled FEM-DD solver.

2.3. D-refinement

2.3.1. Motivation

As discussed in the introduction, we foresee data-driven simulations in which some elements represent material behavior that does display a meaningful, well-defined, linear-elastic range. Performing phase-space searches for elements that do not abandon the small-strain regime appears unnecessary, as no extra accuracy would be gained at the cost of potential errors associated to dataset finite size. Since DDCM requires no intersection between the sets but just transversality, there is always some potential error that is introduced every time that a material datum is assigned to an element (Kirchdoerfer and Ortiz, 2016). In summary, the phase space searches should be limited to regions that clearly depart from linearity, what will be a small portion of the total if the problem displays localization like, e.g., stark stress

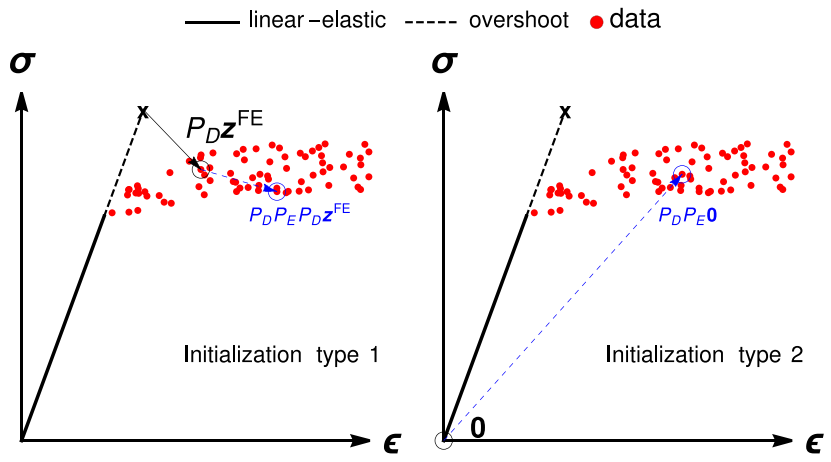

Fig. 1. Schematic representation of DD element initialization possibilities on a 2D phase space. The first circle (black) represents the initialization point and the second one (blue) the point in the dataset that the algorithm converges to in the next iteration (see that each method may pick a different datum after the iteration). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

concentrations or shear banding. Once again, let us state that "complex material response" in the context of this text refers to anything beyond linear elasticity.

Let us also remark that the searches become more involved and time-consuming as the dimensionality of the space increases. From 1-dimensional problems requiring only one component of stress and strain to define the per-element 2-dimensional phase space, to 3-dimensional elements requiring six of each in a 12-dimensional phase space, the resources invested in performing the projection onto D scale rapidly. Of course, the size of D itself also plays a major role, but this issue can be offset via smart searching procedures (Kirchdoerfer and Ortiz, 2016; Eggersmann et al., 2021a), e.g., employing the k-d tree algorithm.

2.3.2. Implementation details

For maximum efficiency, the raw material dataset should be sifted to get rid of points that represent linear response. For the upcoming simulations, we defined limit stresses σ_{lim} in terms of some combination of the stress components (e.g., mean stress) and chose to include each datum z_e whenever $f(z_e) > 0.8\,\sigma_{\text{lim}}$, where f represents the aforementioned stress combination. The reason for using a relaxed inclusion is leaving enough wiggle room in the dataset to account also for almost-non-linear response.

Likewise, to take into account the uncertainty associated with the definition of the threshold itself, we choose to switch to DD whenever $f(z) > 0.9\,\sigma_{\text{lim}}$.

When a regular element is swapped by a DD one, the strain–stress state of the latter can be initialized in different ways (Fig. 1).

- Method # 1: The last state of the regular element, call it z_e^{FE}, can be swapped by its closest point in D, i.e., by $z_e^* = \text{argmin}\, d(z_e^{FE}, \text{D})$, or, using projection notation $z_e^* = P_D z_e^{FE}$. This approach initializes the element at a proper location in phase space, at the expense of performing an extra search.
- Method # 2: Simply initialize the new DD element state at the origin, i.e., $z_e^* = 0 \in \mathbb{R}^{2N_c}$. This method has the advantage of avoiding the initial projection, but could cause the element to follow a phase-space trajectory different from the one in method #1 in subsequent steps. This approach appears more adequate for history-dependent materials, but these are out of the scope of this text. Herein, we will show that this initialization procedure is faster (since it avoids extra phase-space searches) but can lead to excessive refinement when used without adequate incremental loading.

Incremental loading is unavoidable in some circumstances as, e.g., large deformation analysis (to recompute geometrical stiffness changes associated to shape evolution), in the presence of dynamics (time marching), or inelasticity (history-dependent material response). Nevertheless, when dealing with quasi-static elastic yet non-linear response, the load could be applied at once and the phase-space searches would take care of finding the most convenient element states. We will show that this is indeed the case, but that the final solution can depend on the choice of aforementioned initialization methods.

We are assuming monotonously-increasing loading throughout this text. If the material exhibits inelasticity, elements that have been refined should not be swapped back to linear FE as the deformation they suffered is irreversible. However, if the material is non-linear elastic, one could define a threshold under which a DD-informed element is converted back to a linear FE element.

The general version of the d-refinement solution procedure is developed in Algorithm 2.

3. Verification

Two verification exercises are presented in this section. For both, an underlying constitutive law – from which a synthetic database is generated – is assumed to exist, as this allows reaching solutions with a traditional iterative solver, here specifically Newton–Raphson (NR). This iterative solution is then considered as exact in order to estimate the accuracy of the data-driven or d-refined solutions. The two cases are used to study different behaviors of the method. In the first one, constant-stress triangular (CST) elements are used under the plane stress assumption. The main topics of interest are the wall time speed-up, the spread of the refinement and the influence of the initiation of newly refined elements. The second one is a 3D truss-beam discretized with axial bar elements. This is used to study the accuracy and convergence of the method with respect to the number of datapoints, the influence of noise in the dataset and the impact of the load stepping procedure.

3.1. 2D plane-stress elements: plate with hole

We study first the capabilities of the one-load-step d-refinement procedure. We create a dataset using the per-element phase-space trajectories of the incremental NR solver, and then (a) we compare to NR solver in terms of simulation speed assuming no incremental loading in the d-refinement case. The effect of changing the number of elements in the dataset is considered. (b) We also verify the accuracy

Algorithm 2 D-refinement framework

Require: $\forall e = 1, \dots, N_e$, compatibility matrices $\boldsymbol{B}_e$, $\mathbb{D}_e$ matrices containing elastic constants, local datasets D_e and $\mathbb{C}_e$ matrices containing distance constants; $\forall i = 1, \dots, N_{\mathrm{dof}}$, final external forces F_i or boundary conditions.
(i) Set $j = 1$, $S_2^{(1)} = \emptyset$ and $S_1^{(1)} = \{1, 2, 3, \dots, N_e\}$.
(ii) Incremental loading. Initialize $l = 1$
for $l \leq N_{\mathrm{steps}}$ **do**
Set loading level $\boldsymbol{f} = l/N_{\mathrm{steps}}\boldsymbol{F}$
  if $|S_2^{(j)}| = 0$ **then**
    Set $\boldsymbol{\eta} = 0$ and solve for $\boldsymbol{u}$ in eq. (8b).
  else
    (ii.a) Initialize $k = 0$
    (ii.b) Project onto E:
     Solve in eq. (8) for $\boldsymbol{u}^{(k)}$ and $\boldsymbol{\eta}^{(k)}$
    for $e \in S_2{}^{(j)}$ **do**
      Compute $\boldsymbol{\epsilon}_e$ and $\boldsymbol{\sigma}_e$ from eq. (9)
    end for
    (ii.c) Project onto D:
    for $e \in S_2^{(j)}$ **do**
      Choose $\boldsymbol{z}_e^{*(k+1)} \in \mathrm{D}_e$ closest to $\boldsymbol{z}_e^{(k)} \in \mathrm{E}_e$
    end for
    (ii.d) Convergence test:
    if $\boldsymbol{z}_e^{(k+1)} = \boldsymbol{z}_e^{(k)}$ $\forall e \in S_2^{(j)}$ **then**
      Final displacement field $\boldsymbol{u} = \boldsymbol{u}^{(k)}$ **goto** (iii)
    else
      $k \leftarrow k + 1$, **goto** (ii.b)
    end if
  end if
  (iii) Check for linear elements above threshold:
   Set $S_1^{(j+1)} = S_1^{(j)}$, $S_2^{(j+1)} = S_2^{(j)}$
  for $e \in S_1^{(j)}$ **do**
   Compute element strain and/or stress: $\boldsymbol{\epsilon}_e$ and $\boldsymbol{\sigma}_e$
    if threshold criterion **then**
      Append e to $S_2^{(j+1)}$, drop it from $S_1^{(j+1)}$
      Initial assignation: either choose $\boldsymbol{z}_e^{*(k)} \in \mathrm{D}_e$ closest to $(\boldsymbol{\epsilon}_e, \boldsymbol{\sigma}_e)$ or pre-defined $\boldsymbol{z}_e^* \in \mathrm{D}_e$
    end if
  end for
  (vi) Global convergence condition
  if $|S_2^{(j)}| = |S_2^{(j-1)}|$ **then**
    All DD elements converged (or no need for DD), no need of further refinement **exit**
  else
    $j \leftarrow j + 1$
  end if
  $l \leftarrow l + 1$
end for

of d-refinement via comparison to NR, and (c) compare the two initialization methods (no incremental loading in the d-refinement case) in terms of accuracy (comparison to NR) and computational time. For each initialization method, the proportion of elements that become DD and how many of them reach the same state as in the NR solution is considered. Lastly, (d) we assess the impact of widely distributed non-linearity (i.e., higher load, and hence less linear elements and more refinement).

The phase-space searches are carried out in parallel using six processors as long as the total number of DD elements is above twelve. This problem is implemented in a Mathematica notebook (Wolfram, 2000), which can be downloaded from on-line repositories (see supplementary material section).

3.1.1. System

We resort to a classic 2D benchmark exercise: a thin square plate (side length $L = 0.2\,\mathrm{m}$) with a circular hole (radius $r = 0.02\,\mathrm{m}$) loaded in tension on two opposite edges, see Fig. 3(a). The applied traction p equals 100 MPa.

Material. The material the plate is made of is assumed to change (decrease) stiffness when the mean stress ($\sigma_m = (\sigma_{\mathrm{I}} + \sigma_{\mathrm{II}})/2 = (\sigma_{\mathrm{xx}} + \sigma_{\mathrm{yy}})/2$) surpasses a limit value σ_{lim}. No inelastic behavior is considered in these simulations, deformations are considered recoverable; this does not play out anyway since we study a quasi-static monotonously-increasing loading scenario. The Young's modulus of the material is assumed to change according to an underlying model (Fig. 2):

$$E(\sigma_m) = \begin{cases} E_0 & \text{if} \quad \sigma_m < \sigma_{\mathrm{lim}} \\ \max\left[\left(\frac{\sigma_{\mathrm{lim}}}{\sigma_m}\right) E_0\,,\, 0.5 E_0\right] & \text{if} \quad \sigma_m \geq \sigma_{\mathrm{lim}} \end{cases}, \tag{10}$$

E_0 being the zero-strain modulus, while the Poisson's ratio is assumed to remain constant. For the simulations we are to show, $E_0 = 200\,\mathrm{GPa}$, $\nu = 0.33$ and $\sigma_{\mathrm{lim}} = 75\,\mathrm{MPa}$. This model is "invisible" to the data-based solvers, but not for the NR one. We use $\mathbb{C}_e = \mathbb{D}_e = \mathbb{D}$ $\forall e$, where

$$\mathbb{D} = \frac{E_0}{1-\nu^2}\begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{bmatrix}. \tag{11}$$

We note that models of this type have been proposed as a means of quantifying the effect of crack shielding by microcracking in brittle materials such as ceramics (Hutchinson, 1987; Ortiz, 1987).

The mesh is generated with CST elements. The system is solved first with linear-elasticity elements to check that at least some elements, but not all, surpass the limit stress. A simulation where every element remains linear or becomes non-linear would not be an interesting case study for d-refinement. After this first verification, we move to solve it also using pure DDCM, with d-refinement and with the Newton–Raphson (NR) solver.

At each loading level, the incremental NR solver convergence condition is $|\boldsymbol{f}_{\mathrm{ext}} - \boldsymbol{f}_{\mathrm{int}}|/|\boldsymbol{f}_{\mathrm{ext}}| < \mathrm{tol}$, where $\boldsymbol{f}_{\mathrm{ext}}$ is the nodal external force vector and $\boldsymbol{f}_{\mathrm{int}}$ the internal one for unrestrained degrees of freedom. The total external force is applied linearly over ten steps.

The datasets used for this first comparison proceed from the solution of the incremental solver: at the end of each loading step, the strain and stress state of each element is recorded as a point in phase space that becomes part of the material set D according to the criterion in Section 2.3.2.

3.1.2. Solution analysis

Qualitatively, Fig. 3(b), the method performs as expected: the top frame displays the elements that overcome the mean-stress limit in the linear-elastic simulation (vertically-hatched elements), while solid-color elements in the same figure show the spread of non-linearity predicted by the NR solution. The latter set is larger and contains most of the former. The stress is more widely distributed over the domain once the non-linear softer material is accounted for, what pulls more elements past the threshold than those presumed by the linear-elastic solver. The lower frame shows, in addition to the aforementioned NR elements that go non-linear, the elements that become data-driven according to d-refinement (horizontal hatching). See how these subsume completely the others, meaning that this solver does predict where non-linear behavior needs to be accounted for and proceeds in accordance to implement the DD formulation therein.

Wall time speed-up. Regarding the results presented next, let us remark that the time difference between realizations of the same simulation differ by about than a ±10% (maximum), in particular those solved with pure DDCM, in which the initial assignation is random.

We begin by comparing the performance of the new approach against NR solver for different values of tol while raising the load up to

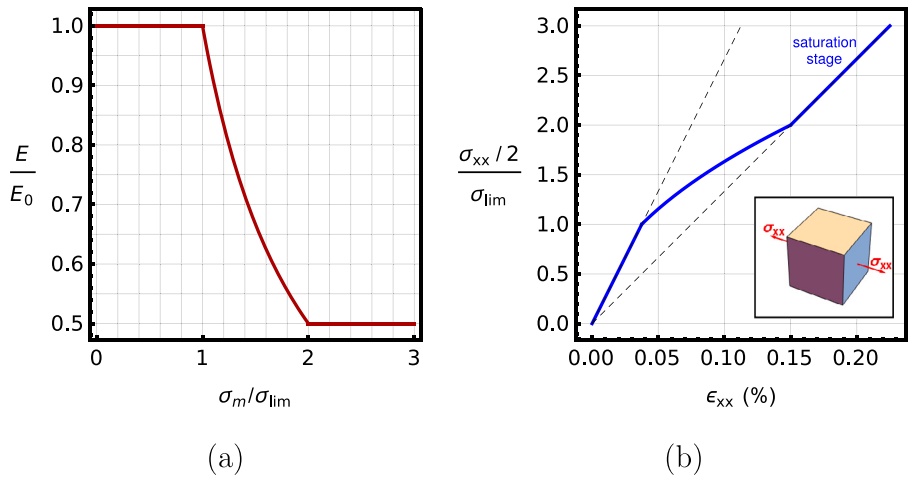

Fig. 2. Visualizing the material response of model Eq. (10). (a) Stiffness loss as mean stress increases. (b) Stress–strain relation for uniaxial loading (cf. Ref. Ortiz (1987)).

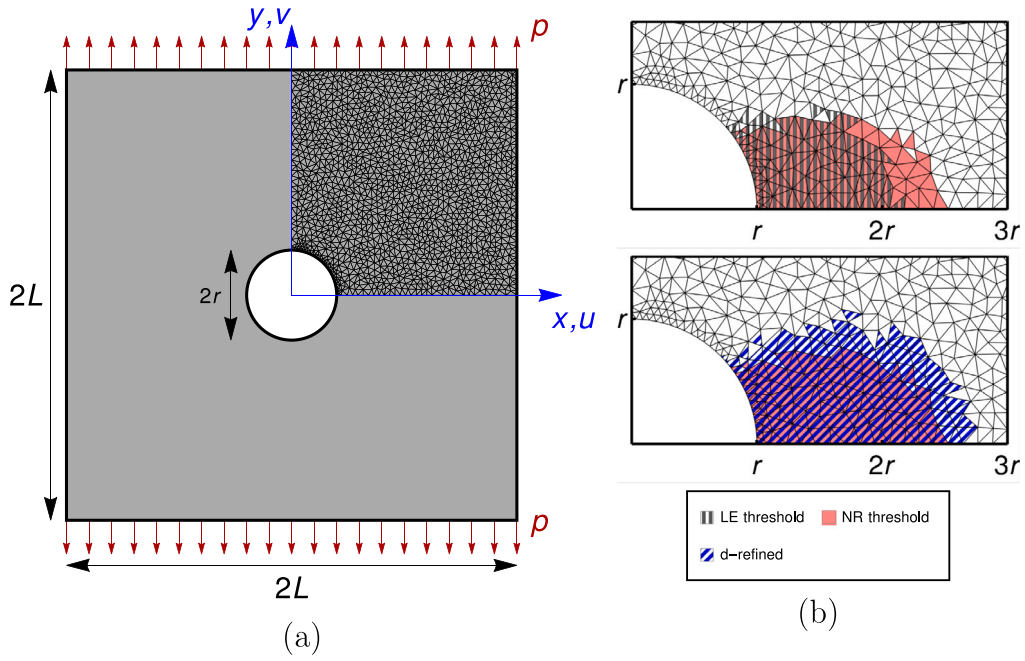

Fig. 3. System and simulation's main outcome (results corresponding to tol = 10^{-3} and |D| = 886 elements). (a) Scheme of the system, including loading and reduced mesh (taking advantage of symmetry). (b) Results: elements above σ_{lim} in linear-elastic simulations (vertically-hatched gray), those above the same threshold in Newton–Raphson simulations (solid pink) and elements that switched to DD in the course of the simulation (diagonally-hatched blue). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

its final value, $p = 100\,\text{MPa}$, over ten increments at constant rate. Note that this final value would yield a state of mean stress $\sigma_m \approx 50\,\text{MPa} < \sigma_{\text{lim}}$ away from the hole.

We take advantage of the most accurate NR simulation (tol = 10^{-5}) to generate the dataset as described above and of the linear-elastic solution to set the method constants: $\mathbb{C}_e = \mathbb{D}\ \forall e \in S_2$. The set D contains at first 29 240 data (including information about elastic loading), and it reduces to just 886 (that only include information within the non-linear regime) after the sifting (Section 2.3.2). Let us re-iterate that this convenient reduction is uncontroversial under the assumption of monotonously-increasing load, but new considerations enter the picture if one wants to consider non-monotonous loading (either elastic or inelastic); for instance, one would have to take decisions as to how to deal with DD elements that revert to the elastic regime.

The results concerning computational speed-up are consigned in Table 1. The d-refinement technique is faster than both NR solver (for all the precision tolerances tried) and pure DD solution. For the latter, a simulation with a reduced material set (|D| = 4598, including linear response) was sampled at random from the NR phase-space trajectories (29 240 points).

The evolution of this particular d-refinement simulation is represented in Fig. 4. All the elements that were presumed by the linear-elastic solver to be above the mean-stress threshold were turned into DD after the first iteration. Of course, all these elements were assigned new dataset points, what is reflected in the other (red) curve first point. As the simulation moved forth, the number of DD elements converged quickly, while the material point assigned to each one of them did so more slowly.

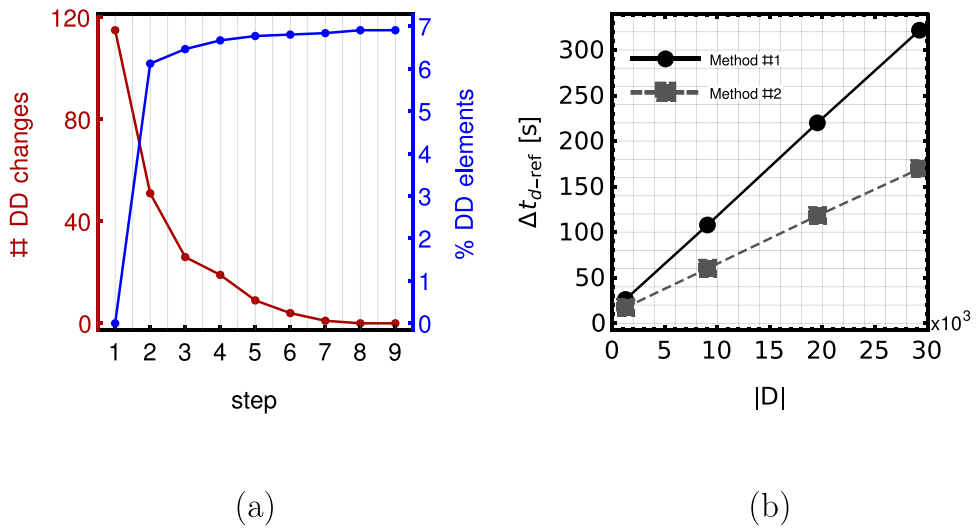

Fig. 4. (a) Simulation progress in terms of portion of the mesh being refined and number of DD elements that changed material data by the end of each step. (b) Linear scaling of d-refinement wall time (Δt_{d-ref}) with size of dataset (for a similar number of refined elements): the solid line (Method #1) represents initial assignation $\boldsymbol{z}_e^* = \mathrm{argmin}\ d(\boldsymbol{z}_e^{FE}, \mathrm{D})$ (202 refined elements by the end of the simulation), the dashed line (Method #2) $\boldsymbol{z}_e^* = \boldsymbol{0}$ (270 refined elements on average). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Accuracy of d-refinement. To assess the difference between the solution using mesh d-refinement and the most accurate incremental solver (tol = 10^{-5}), we choose to compare the phase-space distance between the points corresponding to the Newton–Raphson solution ($\boldsymbol{z}^{\mathrm{NR}} \in \mathbb{R}^{N_e \times 2N_c}$, $N_c = 3$ for plane stress) and the ones of the d-refinement solution ($\boldsymbol{z}^{\text{d-ref}} \in \mathbb{R}^{N_e \times 2N_c}$):

$$\frac{|\boldsymbol{z}^{\text{d-ref}} - \boldsymbol{z}^{\mathrm{NR}}|}{|\boldsymbol{z}^{\text{d-ref}}|} = \frac{\left(\sum_{e=1}^{N_e} w_e |\boldsymbol{z}_e^{\text{d-ref}} - \boldsymbol{z}_e^{\mathrm{NR}}|\right)^{1/2}}{\left(\sum_{e=1}^{N_e} w_e |\boldsymbol{z}_e^{\mathrm{NR}}|\right)^{1/2}} \approx 0.034\,, \tag{12}$$

that is, the distance between solutions in phase space is less than 4% of the one from the origin to the d-refinement solution. This is in spite of the material set containing exactly the same points as the non-linear portion of the Newton–Raphson trajectory. Actually, if we compare where the refined elements land, only 23 out of 202 arrive at their final configuration in the incremental solver. This does not translate in error nonetheless (recall that the error we have estimated using distances is less than a 4%), which necessarily means that the local density of the material set in phase space is such that some elements can converge to nearby points at virtually no cost.

For this kind of initialization ($\boldsymbol{z}_e^* = P_D \boldsymbol{z}^{FE}$, labeled "method #1"), it is interesting to note that the proportion of elements that become data-driven is either completely independent of or very weakly-dependent on the phase-space density. Increasing the precision of the incremental solver by raising the number of load steps from 10 to 25 increases the material dataset size to |D| = 1878 points, and this has no effect either over the number of elements that become DD (202) or on the proportion of these that reach the same destination (23). In terms of computational time, Table 2, d-refinement again outperforms the most accurate NR (tol = 10^{-5}).

Effect of dataset size on d-refinement performance. This also motivates a sensitivity study as to the influence of the size of the material set D (number of material data): the incremental solver is kept at ten load increments and tol = 10^{-3}, but we grow D by lowering the threshold to facilitate points in the Newton–Raphson trajectory entering D. We find the expected linear proportionality between total wall time and dataset size, see solid line in Fig. 4. We discuss the influence of the other initial data assignation (which happens to reduce time, see dashed line in Fig. 4, at the expense of accuracy) in the next section, but both methods display this linear scaling. To further decrease the computational time, one could use some of the techniques discussed in the literature (Eggersmann et al., 2021a,b).

Influence of initial data assignation. Until now, reported results used phase-space searches to assign an initial datum to each element that evolves from FEM to DD. This involves additional searches that could be avoided by always assigning a pre-defined initial datum, i.e., the origin.

This creates a somewhat disjoint material set D, since there is a single point away from the cluster that represents non-linear behavior.

The approach brings about a substantial speed improvement: based on the results presented in Fig. 4, we find that this initial assignation method ("method #2") reduces the computation time by about 45%. This comes at the expense of increasing the error when compared to NR results: the distance ratio Eq. (12) doubled from 3.5% to 7%. So, as often, there is here a trade-off between accuracy and time efficiency. Depending on the priority, one would favor one method over the other. However, this conclusion is contingent on localized non-linear behavior, what leads to having similar final number of DD elements, see next point.

Impact of spreading non-linear behavior. As the load increases, the actual solution departs more meaningfully from the linear-elastic one. As elements excursion deeper past the threshold, richer datasets are also necessary to capture the response.

For instance, keeping the dataset inclusion criterion and increasing by 20% the load we have been using leads to an increase in |D| from 886 to 3419 phase-space points.

This also increases both the NR computation time and the d-refinement's, as well as the relative error between them (Eq. (12)), see Table 3. The spreading of the non-linearity induced by the intenser load means more DD elements and hence more phase-space searches, what slows d-refinement down to the point of NR being faster in this case. The final number of refined elements has escalated from 202 to 620. Interestingly, initialization type 2 ($\boldsymbol{z}_e^* = \boldsymbol{0}$) ends up defaulting to a pure data-driven mesh due to accumulation of errors and the total time skyrockets.

We remarked that the precision of either approach can be increased by sharpening the refinement threshold criterion. In any case, such an approach is a barren one, since in practice the actual threshold will not be well-defined; the difference between d-refinement and the incremental solver could be regarded not as a demerit of the former but as modeling bias infused in the latter.

The main conclusion is that in quasi-static, monotonous-loading cases where non-linearity is expected to be present well over the domain, d-refinement must be used with initialization #1, since this yields

Table 1
Solving the plane-stress problem with different methods: wall time comparison (p = 100 MPa, initialization type 1, 10 NR increments).

	$\Delta t_{\text{d-ref}}$ ($\lvert D\rvert$ = 886)	Δt_{NR} (tol = 10^{-3})	Δt_{NR} (tol = 10^{-4})	Δt_{NR} (tol = 10^{-5})	Δt_{DD} ($\lvert D\rvert$ = 4588)
[s]	20.3	38.5	51.2	72.8	928.7
Increment	–	× 1.9	× 2.5	× 3.6	× 45.8

better first guesses, thus boosting convergence and limiting the number of extra elements to be refined. To gauge the extent of the non-linear portion beforehand, comparing the results of a linear-elastic calculation to the threshold may suffice. We have shown that initialization method #2 is adequate and efficient when there is localized non-linearity and, in the next section, we will also prove its suitability when used in tandem with appropriate load increments.

3.2. 1D bar elements in 3D space: octet-truss structure

In this study, we create a dataset by sampling a constitutive law, and compare d-refinement and NR solver (a) changing the density of the dataset, and (b) changing the number of loading steps.

This application is implemented in a Jupyter notebook, which is provided as supplementary material.

3.2.1. System

The second study features a truss made of octet unit cells (Deshpande et al., 2001). The model is made up by axially-loaded bar elements (no bending), with linearized kinematics, and a total of $6 \times 2 \times 2$ unit octet truss cells submitted to a 3-point bending test as represented in Fig. 5. The parameters of the unit cell octet truss were inspired by the work in Ref. Shaikeea et al. (2022) and are summarized in Table 4. It should be noted that the goal is not to reproduce their results: to do so, either buckling in the case of slender member or joint stiffness for thick member should have been accounted for. This model is intended to test and explore the performance of the d-refinement solver. It has two main advantages: bar elements are straightforward to discretize and their local phase-space is two-dimensional: $\boldsymbol{z}_e = (\epsilon_e, \sigma_e) \in \mathbb{R}^2$.

The beam displacements are fixed on both lower edges and a displacement-controlled deflection is imposed on the top middle nodes. While the load is applied in steps, the problem is considered to be quasi-static and thus solved with the static solver presented above. See Fig. 5 for a graphical representation.

Material. The behavior of the bulk material is also inspired from Ref. Shaikeea et al. (2022), in which trimethylolpropane triacrylate (TMPTA) is used. The material displays similar Young's modulus at zero-strain $E_0 = 430\,\text{MPa}$ and limit strength $\sigma_f = 11\,\text{MPa}$. However, this artificial material is assumed to be non-linear elastic, as opposed to brittle, and is represented by a hyperbolic-tangent stress–strain relation (see Fig. 6):

$$E(\epsilon) = E_0 \left[1 - \tanh^2\left(\frac{E_0}{\sigma_l}\epsilon\right)\right], \tag{13}$$

$$\sigma(\epsilon) = \sigma_f \tanh\left(\frac{E_0}{\sigma_l}\epsilon\right). \tag{14}$$

A synthetic database of 2700 points is generated by sampling uniformly the constitutive law between $\epsilon = -0.2$ and $\epsilon = 0.2$. For the purpose of d-refinement, the trigger to leave the linear elastic domain is assumed to be $\sigma > \sigma_l = 7\,\text{MPa}$. Fig. 6 summarizes the material behavior, with the underlying constitutive law, the dataset and the assumed linear elastic domain.

Table 2
Solving the plane-stress problem: wall time comparison (p = 100 MPa, initialization type 1, 25 NR increments).

	$\Delta t_{\text{d-ref}}$ ($\lvert D\rvert$ = 1858)	Δt_{NR} (tol = 10^{-5})
[s]	55.3	122.3
Increment	–	× 2.2

Table 3
Solving the plane-stress problem: wall time comparison and accuracy (p = 120 MPa, different initialization types, 10 NR increments). The NR solution is used as reference to compare in this case.

	$\Delta t_{\text{d-ref}}$ ($\lvert D\rvert$ = 3419)		Δt_{NR} (tol = 10^{-5})
	Method #1	Method #2	
[s]	154.8	924.3	105.3
Increment	× 1.5	× 8.8	–
Error (Eq. (12))	0.05	0.14	–

Table 4
Summary of the different parameters used in the truss models with both the geometric parameters and the bulk material properties.

Notation	Value	
l	530 μm	Strut length
d	65 μm	Strut diameter
E_0	430 MPa	TMPTA Young's modulus at zero strain
σ_f	11 MPa	TMPTA tensile strength

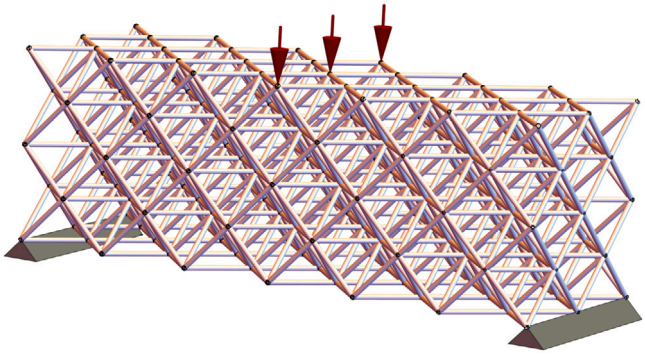

Fig. 5. Scheme of three-point bending configuration. Arrows represent the imposed displacement at the top middle nodes.

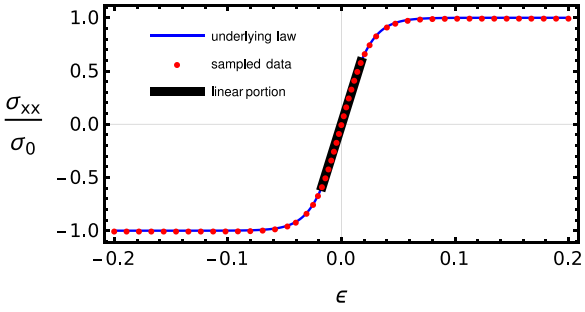

Fig. 6. Constitutive law, synthetic dataset and the limits of the linear elastic domain. For the dataset, only 1% of the points are plotted to increase legibility.

3.2.2. Solution analysis

The following is a comparative study of the results obtained with three different solvers: a Newton–Raphson informed by the constitutive law (NR), a pure data-driven one informed by the dataset from the constitutive law (DDCM) and a d-refinement one informed by the dataset as well as the zero-strain Young's modulus and linear elastic limit (DR). Initialization method #2 is chosen in this case. Unlike the

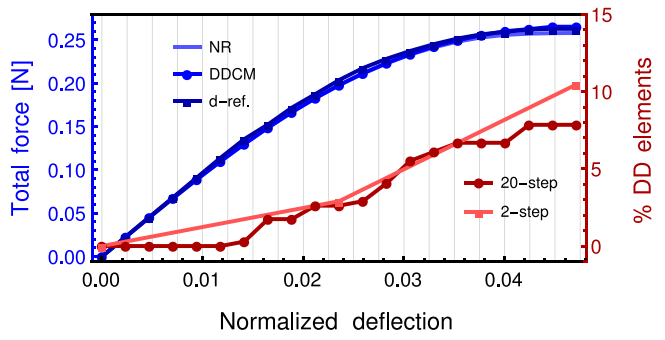

Fig. 7. Load–deflection curves (upper) obtained with the three solvers and a 20-step loading protocol (left vertical axis). The deflection is normalized by the beam length. The total force is computed through the internal forces in the bars that converge at the nodes where the displacement is enforced. The lower curves (right vertical axis) represent the fraction of rod elements that have been refined, i.e. switched to DD, for both the 20-step loading and the 2-step loading. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

previous plane-stress study, the incremental version of d-refinement is used here.

Error analysis. The load response of the beam for each solver is recorded to be then compared. With a 2700-point dataset ($|\mathrm{D}| = 2700$) and taking the Newton–Raphson solution as a reference, the relative errors in load response are 2.8% and 1.8% for the DD and DR solvers, respectively.

Convergence with noiseless and noisy dataset. Then, the number of points in the dataset is increased. The errors, as defined by Eq. (12), of both the DD and DR solvers reduce at a rate given by a power law with exponent one. This $\sim|\mathrm{D}|^{-1}$ scaling is coherent with the literature (Kirchdoerfer and Ortiz, 2016). However, for the DR solver, the convergence to the NR solution is ultimately limited by the quality of the linear assumption at small strains. The linear approximation does not overlap perfectly with the underlying hyperbolic-tangent law, hence some minor difference in the linear-elastic elements persists independently of the number of datapoints. We stress that this is not a flaw in the d-refinement method but a matter of modeling bias.

When random Gaussian noise is added to the datapoints (both in strain and stress) with standard deviation inversely proportional to the square root of the number of datapoints (Kirchdoerfer and Ortiz, 2016), the same behavior is observed but the convergence rate slows down to $|D|^{-1/2}$ which is again in agreement with the literature results (Kirchdoerfer and Ortiz, 2018, 2017). For data whose noise is independent of the number of datapoints, max-ent data-driven solvers (Kirchdoerfer and Ortiz, 2017) are more adequate than the fixed-point one presented here. As for a more formal proof of convergence, the linear elastic approximation can be seen as being part of the dataset (as a graph or a sampling of a line with infinite density for example) and thus the convergence proof derived in Ref. Conti et al. (2018) can be extended to the d-refinement solver: if the linear approximation and the dataset converge to the real underlying material behavior, so will do the d-refinement solver.

Influence of load-stepping. As the stepping procedure unfolds and the load increases, more elements go over the threshold and are switched to the DD solver. The size of the loading step influences which elements are switched: a large step may cause the refinement to overshoot, i.e., to refine elements that should have remained within the linear elastic regime. The red lines (lower) in Fig. 7 highlight this phenomenon: the fraction of refined elements at the end of the loading is 7% with 20 steps and 10% with 2 steps. In Fig. 8, with 20-step loading, the refined elements correspond exactly to the ones that undergo large straining according to the NR solution. In 2-step loading, more elements are refined. This overshooting, associated to using initialization at the origin and not enough load steps, does not influence the accuracy of the final solution provided that (a) the dataset has points in the elastic zone, at least in the proximity of the limit as suggested before, and (b) the refined elements represent a fraction of the total (recall the results obtained in the previous case study when increasing the load). The only cost of the overshooting in refinement is computational, no meaningful extra errors are induced. However, adding more loading steps also have the cost of computing extra intermediate steps that might not be of interest, so a balance should be met depending on the type and purpose of the simulation. It should be noted that the standard Newton–Raphson solver could diverge if the load steps were too large, thus another advantage of the d-refinement method is its stability.

4. Application: bridging scales to study the fracture process zone in architected metamaterials

While DDCM allows to use experimental data to make predictions without resorting to phenomenological models, it can also be applied as an efficient method to solve multiscale problems (Karapiperis et al., 2021, 2020; Korzeniowski and Weinberg, 2022). It is particularly suited in the case where a convincing microscale model exists, which, for any reason, cannot be run at the larger scale of interest. In such a case, a dataset consisting of RVE microscale simulations over different loading paths can be computed offline and then passed to a macroscale DDCM model. Since the overhead associated to running microscale simulations is much lower than the one of performing experiments, the development of the dataset can be done in several iterations: areas of the phase space which have been found to not be dense enough can be repopulated (Karapiperis et al., 2021); one can even foresee launching the RVE microscale simulations on the fly (Karapiperis et al., 2021).

To showcase the merits of DDCM for multiscale modeling, the two models presented in Section 3 are combined to devise an architected metamaterial (Meza et al., 2015). Starting from the synthetic dataset of TMPTA material behavior, microscale simulations of an RVE unit cell of octet truss were run using the DDCM solver. Then, this dataset was fed to a continuous macroscale model with the goal of resolving the stress distribution around a crack tip with the d-refinement solver.

In summary, we will go from a dataset of the underlying material to the response of the microstructure, to finally reach the macroscale and perform mechanical analysis where the non-linear behavior of the octet-truss architected metamaterial is taken into consideration with no recourse to any constitutive modeling whatsoever.

This family of artificial materials has commanded the attention of the research community in recent decades (Fleck et al., 2010). Due to their manufacturability and remarkable properties, they are poised to challenge conventional materials in a bevy of applications, from thermal insulation (Dou et al., 2018) to impact absorption (Portela et al., 2021).

4.1. System

Microscale. The microscale model is the same used in Section 3.2. The RVE is a single unit octet-truss cell (represented in the middle of Fig. 9). The axes of the frame of reference $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ are aligned with the lattice planes ([100], [010], [001]), see Fig. 10(a). The cross-sections of the bar elements that lie on the boundary are reduced as they are shared with the neighboring cells.

To create the dataset, the stress response to specified strain states $(\epsilon_{xx}\ \epsilon_{yy}\ \epsilon_{xy})$ are simulated. For this, the strain state is converted into imposed displacements at the ten boundary nodes. To compute the homogenized stress states, the sum of the forces, on each facet, for each direction, is divided by the apparent area of the cell facet. The material behavior is again considered elastic. Since it was verified previously that the accuracy of the final solution does not depend on the number of loading steps when using initialization method #2, no incremental loading is employed. The material response of the unit cell is illustrated in Fig. 10 where strain–stress paths are plotted for uni-axial elongation, pure shear and combined shear-stretching.

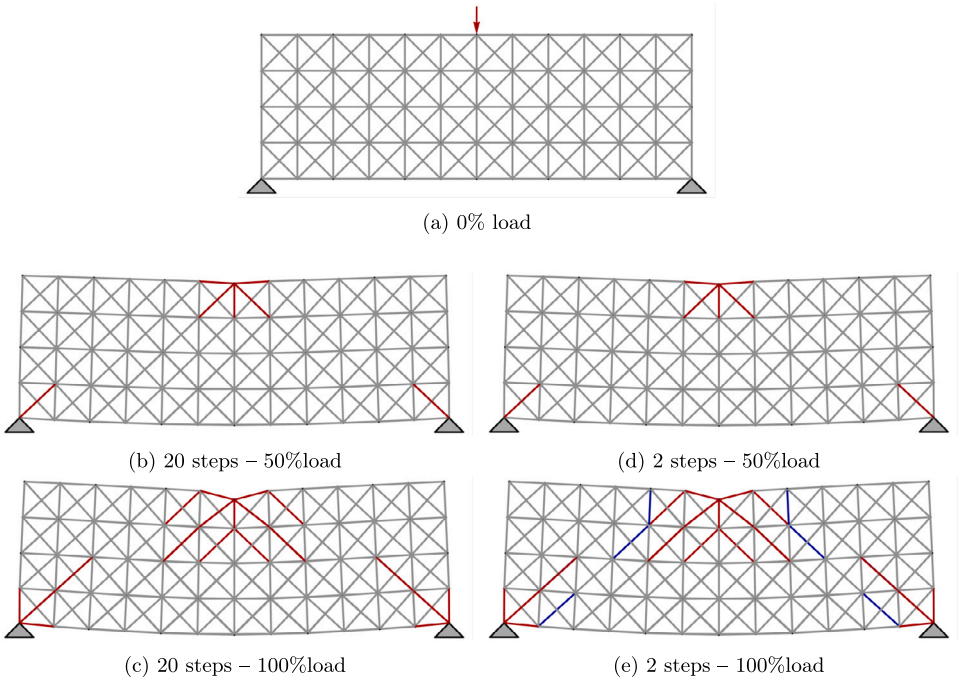

Fig. 8. Projection of the beam truss on a 2D plane. The right column depicts results obtained with a 2-step loading and the left one with a 20-step loading. In red, elements that should be and are refined. In blue, elements that are unnecessarily refined. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

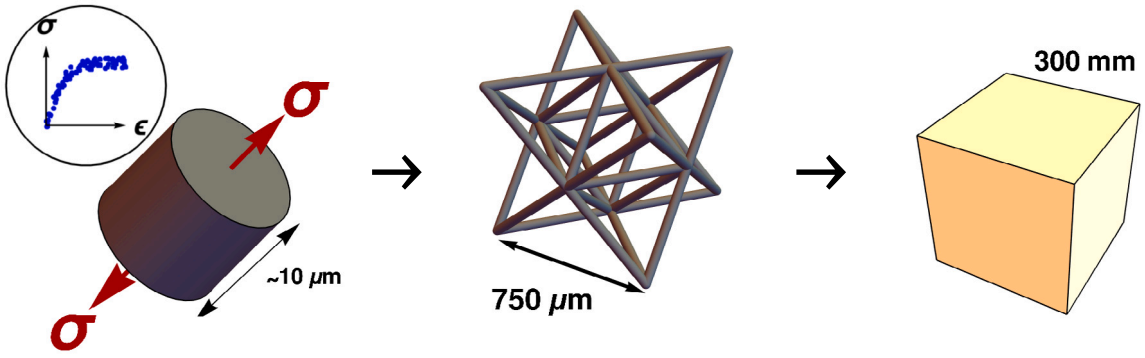

Fig. 9. Scheme of the scale bridging procedure (from left to right): rod material dataset is used to characterize the response of octet-truss unit cell using DDCM, these RVE simulations yield linear-elastic constants and a non-linear material-response dataset used for macroscale simulations of a cubic sample of architected material using d-refinement.

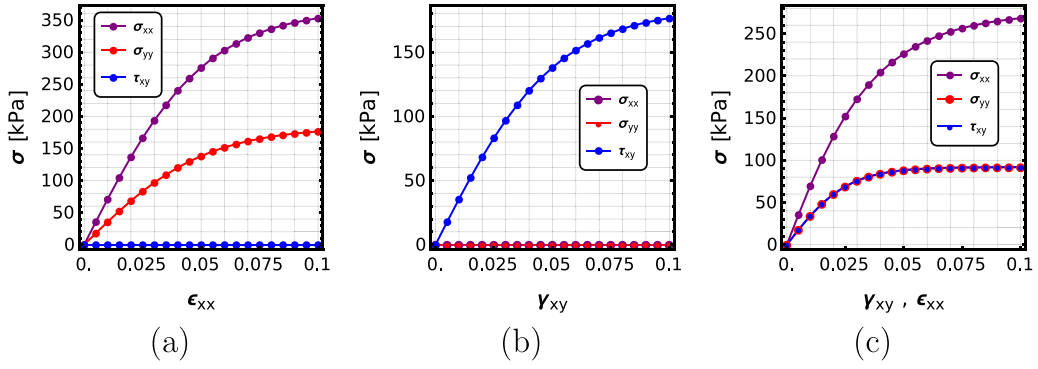

Fig. 10. Strain–stress paths of microscale RVE simulations. (a) Pure uniaxial elongation. (b) Pure shear. (c) Simultaneous elongation and shear.

Unit-cell elastic response. First, as the d-refinement method necessitates information about the small-strain behavior, the elastic constants should be determined. For this, the small-strain response of the unit cell to different deformation scenarios was solved with DDCM. The

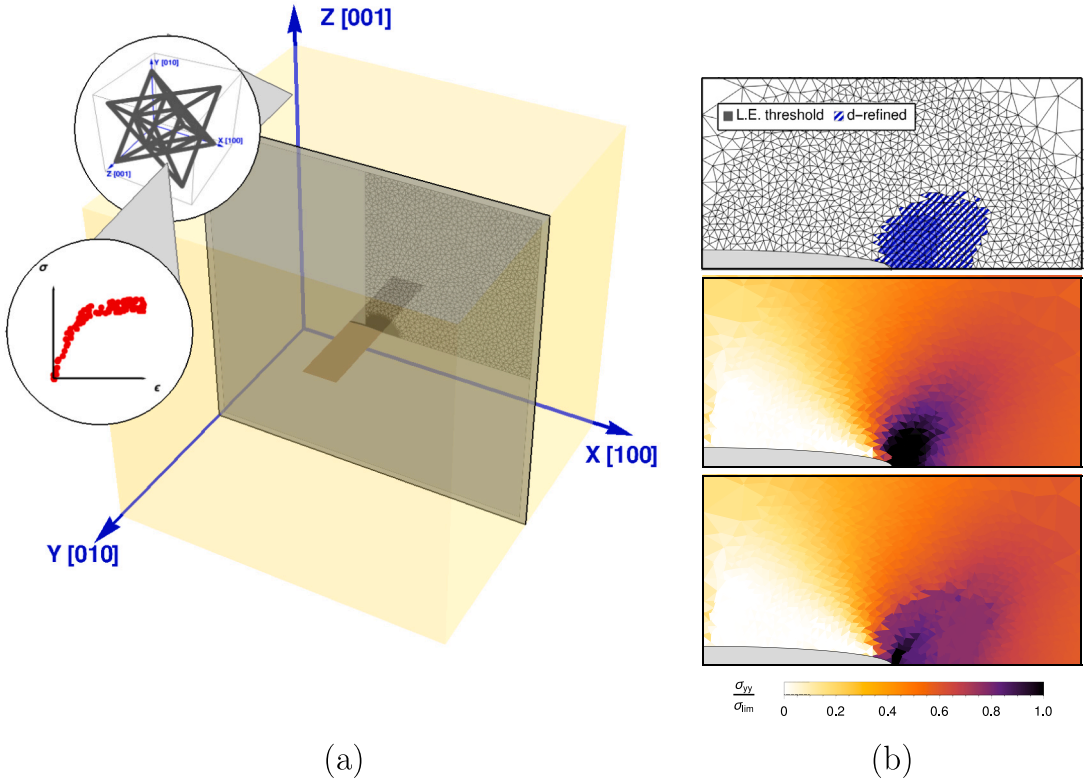

Fig. 11. Macroscale simulation using d-refinement with data generated using RVE probing with DDCM. (a) Scheme of cubic block of architected metamaterial containing a long crack within (cf. Ref. Shaikeea et al. (2022)). The middle plane includes the reduced mesh used to generate plane-strain results. It also represents TMPTA dataset used to characterize the unit cell and a scheme of the octet-truss. (b) Results. Top: detail of the mesh around the crack tip including area above threshold according to linear-elastic pre-analysis (solid gray) and elements that became DD during the d-refinement solution (blue, diagonal hatching). Middle: dimensionless vertical normal stress σ_{yy} (normalized by $\sigma_{max} \approx 2p$) obtained with linear-elastic simulation (darkest zone is close to or above the critical value). Bottom: dimensionless vertical normal stress obtained with d-refinement simulation. See how the stress concentration at the tip region is blurred. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

following compliance matrix is obtained:

$$\begin{pmatrix} \epsilon_{xx} \\ \epsilon_{yy} \\ \epsilon_{zz} \\ \gamma_{yz} \\ \gamma_{xz} \\ \gamma_{xy} \end{pmatrix} = \frac{1}{0.1E_0} \begin{pmatrix} 9 & -3 & -3 & 0 & 0 & 0 \\ -3 & 9 & -3 & 0 & 0 & 0 \\ -3 & -3 & 9 & 0 & 0 & 0 \\ 0 & 0 & 0 & 12 & 0 & 0 \\ 0 & 0 & 0 & 0 & 12 & 0 \\ 0 & 0 & 0 & 0 & 0 & 12 \end{pmatrix} \begin{pmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \sigma_{yz} \\ \sigma_{xz} \\ \sigma_{xy} \end{pmatrix}, \tag{15}$$

with $\gamma_{ij} = 2\epsilon_{ij} = u_{i,j} + u_{j,i}$. These results are consistent with the analytical prediction of homogenized octet-truss elastic constant with facets aligned with the principal directions (Deshpande et al., 2001). The octet-truss architected material is known to be a cubic orthotropic material with the same Young's modulus along the three orthotropy axes (Deshpande et al., 2001).

Let us, finally, remark that the inelastic behavior of the unit cell has been a subject of constitutive modeling in the past. In Ref. Deshpande et al. (2001), the authors worked out an anisotropic yield criterion, whose limitations were openly admitted. In these circumstances, d-refinement can provide an efficient pathway to macroscale simulations while a fully-satisfactory constitutive model is being devised.

Macroscale. The goal of this model is to study how the stress around a crack tip is distributed for an architected metamaterial. Linear-elastic fracture mechanics (LEFM) yields infinite stress at the crack tip (Griffith, 1921). In reality, there is a region of damaged material around the tip, the "fracture process zone" (Pineau and Pardoen, 2007), that prevents the appearance of this singularity by redistributing stress over a greater area. The relative size thereof defines the range of validity of LEFM: if this region characteristic length is of the order of the geometrical features of the system (Hutchinson, 1983), then a ductile fracture viewpoint should be adopted. Assuming a long crack and uniform loading, only a cross-section is considered by taking the plane strain assumption, as pictured in Fig. 11. The unit cell is a cube of side $l = 750\,\mu m$. The model is qualitatively similar to the one presented in Section 3.1 (Fig. 3(a)) but with the shape of the hole changed to an elongated ellipse in order to better represent a thin crack tip. Specifically, the short radius is $0.1a$, $2a = 40l$ being the crack length. The total area of the plate is $2H \times 2H$, with $H = 10a$.

The mesh is refined around the crack tip, up to an element size of the order of the octet-truss unit cell. The problem was solved with the d-refinement method and each element was equipped with a dataset sifted according to the criterion $\sigma_{yy} > \sigma_{lim} = 280\,kPa$, as this value ensures that the unit cell remains elastic (value obtained after inspecting uniaxial traction results of the unit cell, assuming it to be the most unfavorable case). The traction magnitude in this case is $p = 160\,kPa$.

Database generation. In both verification cases, the dataset was generated synthetically, assuming knowledge of the underlying constitutive law. For the rod element, the phase-space is only two-dimensional and thus it can easily be populated using only the estimated minimum and maximum strain sustained by a bar as bounds. For the 2D elements, the phase space is six-dimensional, meaning that to populate it with a similar density as the two-dimensional phase space, an exponent of three must be applied to the number of datapoints (i.e. a 10^9 datapoints in a unit hypercube in 6D have the same density as 1000 datapoints in a unit 2D square). As this is hardly tractable, some knowledge of the expected strain paths must be obtained to constrain the region of interest in phase space. Previously, this information came from the resolution of the problem with non-linear FEM, thus giving the exact required strain paths. For the architected metamaterial, no underlying constitutive law is available. The chosen approach was to first solve

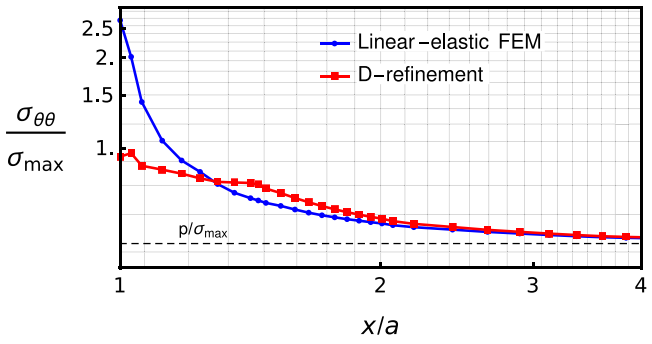

Fig. 12. Hoop stress along the x-axis, from the crack tip ($x = a$) to the middle of the mesh (total horizontal extent of the plate from crack tip is $9a$).

the problem with a linear elastic FEM solver using the cubic material linear-elastic constants. The strain state of each element was recorded. Microscale RVE simulations were then run on 30 strain states linearly interpolated between the zero strain and 1.5 times the recorded strain states. Given that the simulation is composed of 4940 elements, this resulted in $4940 \times 29 + 1 = 143\,261$ points (the zero-strain states were not reproduced for each element), but only 173 surpasses the aforementioned threshold.

4.2. Results

The results are summarized in Fig. 11(b). The top image depicts the mesh around the crack, those elements that the linear-elastic solver predicted to go above the threshold appear in solid color. The refined elements are superimposed with hatched diagonal filling. As expected, the refinement subsumes the above-threshold elements; the extent of the former can be thought as an estimation of the fracture process zone in the architected metamaterial.

The first solve using linear FEM yields the results on the middle plot, with a sharp stress concentration at the crack tip. Then, on the bottom plot, the results with the multiscale d-refinement approach show a more diffuse stress field, with a zone of near-max stress ($\sigma_{\max}$) around the crack tip.

The re-distribution of stress due to material non-linearity is also evident in Fig. 12, where the hoop stress along the horizontal symmetry plane is visualized. Both solutions converge to the remote loading conditions, but, as the crack is approached, the stress is raised faster in the multiscale d-refinement model than in the linear solution, until the latter overtakes the former around the tip. The logic behind this behavior is well understood: the non-linear material cannot carry as much load around the tip, so more material is engaged in high-stress away from it. The stress does not plateau as in the elasto-plastic case, as the beyond-elastic stress states in the octect-truss can be diverse (Fig. 10).

D-refinement could also allow to speedily assess fracture process zone sensitivity in terms of underlying unit-cell material. In recent experiments, the fracture initiation process was characterized for a truss made of similar yet brittle rods (Shaikeea et al., 2022). Considering this brittle micro-constituent behavior is straightforward in d-refinement, as the infinitesimal-strain moduli would not change and the non-linear behavior could be accounted for by erasing some rods from the unit cell, what in turn would translate in a different homogenized non-linear behavior.

5. Conclusion

The d-refinement method brings together the best of two computational paradigms: the speed and ease of definition and implementation of linear FEM method along with the ability of DDCM to faithfully capture complex material behavior. By resorting to DDCM only where and when necessary, efficient data-driven simulations can be run at little accuracy cost — when the material at hand displays a well-defined linear elastic regime at small-strain. The computational cost compares favorably to traditional incremental solvers such as Newton–Raphson without the need to define non-linear constitutive relations.

For multiscale simulations, the offline database generation and data reuse are clear advantages against FE^2 methods. The application to architected meta-materials illustrates the potential use of efficient data-driven methods in multiscale scenarios. Now, thanks to DDCM and d-refinement, efficient simulations of systems featuring architected metamaterials are not contingent on prior development of a constitutive law informed by micromechanical behavior.

While the current implementation is restricted to path-independent (elastic) behavior in statics, the application of the d-refinement method could be readily extended to the formulation of DDCM in dynamics, large deformation or path-dependent material behavior.

Finally, we note that the d-refinement concept can be used also if the constitutive law for the non-linear regime is replaced, not by DDCM, but by a data-trained neural network (see, e.g., Ref. Bonatti and Mohr (2021), Masi et al. (2021) and He and Chen (2022)). A demonstration is included in the Appendix A.

Supplementary material

Mathematica notebooks that implement the calculations described in the text are available from the last author GitHub page github.com/jgarciasuarez, in the repository named "d-refinement". Likewise, Jupyter notebooks are available from the first author Gitlab page github.com/jgarciasuarez, in the repository named "d-refinement".

CRediT authorship contribution statement

Sacha Wattel: Conceptualization, Formal analysis, Writing – original draft, Validation. **Jean-François Molinari:** Supervision, Writing – reviewing and editing. **Michael Ortiz:** Supervision, Writing – reviewing and editing. **Joaquin Garcia-Suarez:** Supervision, Conceptualization, Formal analysis, Writing – original draft, Validation.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Joaquin Garcia-Suarez reports financial support was provided by Swiss National Science Foundation.

Data availability

No data was used for the research described in the article.

Acknowledgments

S.W., J.-F. M. and J. G.-S. gratefully acknowledge the support of the Swiss National Science Foundation via grant "Wear across scales" (200021_197152). The suggestions of two anonymous reviewers have helped to improve the quality and scope of this text.

Table 5
Summary of the parameters defining network architecture.

Layer	Type	# neurons
1(*input*)	–	3
2	Linear	6
3	ReLu	6
4	Linear	18
5	ReLu	18
6	Linear	6
7(*output*)	Linear	3

Appendix A. d-refinement with neural networks

We briefly present how the concept of d-refinement is adapted when the response of the elements turned data-driven is informed, not through projections between sets as it has been done already (see Section 2.3), but through a constitutive law learned from data.

In this case, we will use a very simple neural network architecture, trained on the same data used for the macroscale simulations in Section 4. We are aware of the excellent work being done as to constitutive modeling with physics-informed neural networks (Bonatti and Mohr, 2021; Masi et al., 2021; He and Chen, 2022), yet implementing and training these complex architectures is out of the scope of this appendix and it is left as future work.

The network used is made by 7 layers, numbers 2,4,6 being linear ones and 3 and 5 non-linear with ReLu activation (see Ref. Masi et al. (2021) for a discussion concerning the choice of activation functions), for details, see Table 5. We converged on this architecture after a trial-and-error process guided by the training loss.

The whole data (143 260 set of phase-space points, containing each three strains and corresponding three stresses) produced via RVE probing, as described in Section 4, is used to train the network, no threshold is used as there is no data sifting. The three stress components make up the output of the net and are normalized by the pressure applied as remote loading ($p = 160\,\text{kPa}$) while the input data is presented as the three strain components normalized by a characteristic strain $\epsilon_c = p/E_c$, where the characteristic value of stiffness is $E_c = 3.592\,\text{MPa}$. 80% of the data are used for training, 10% are used for validation during training and the remaining 10% are used as test data once the network has been trained. Both the training and the validation loss (mean-squared error) are below 10^{-4}. The error during test is less than 1%. The network is implemented using Mathematica's specialized functions (Wolfram, 2000) (see supplementary materials to download the notebook). The default learning rate is used.

The use of neural networks as replacement for constitutive laws in non-linear simulations is not new, and has already been successfully implemented (see, e.g., Masi and Stefanou (2022) and Lefik and Schrefler (2003)); still, tangent operators at the element level need to be generated. This "tangent-operator" approach is not suitable for d-refinement, as it would render back a Newton–Raphson method, the only difference being that only the operators of the elements surpassing the threshold need to be recomputed as they correspond to each element linear-elastic matrix. Contrariwise, the traditional DDCM "phase-space iterations approach" can be leveraged to devise a fixed-point iteration scheme similar to the one presented in Section 2.1: when minimizing the distance, elementwise, between a material-admissible state $\boldsymbol{z}_e^* \in Z_e$ and a physics-admissible state $\boldsymbol{z}_e \in Z_e$, instead of considering that $\boldsymbol{z}_e^* \in \mathrm{D}_e$, we can assume that for each element $\boldsymbol{z}_e^* = \{\boldsymbol{\epsilon}_e^*, \boldsymbol{\sigma}_e^*\} = \{\boldsymbol{\epsilon}_e, \mathrm{NN}(\boldsymbol{\epsilon}_e)\} \in \mathcal{M} \subset Z$, where $\mathcal{M}$ represents a manifold embedded withing the phase space that is learned from the data by the neural network and parameterized with the strains (in our case, $\mathcal{M}$ is the same for all elements), $\boldsymbol{\epsilon}_e^*$ and $\boldsymbol{\epsilon}_e$ are the element's material strain and physics-admissible one (and can be taken to be equal) and $\mathrm{NN}(\boldsymbol{\epsilon}_e)$ means applying the neural network operations to the input $\boldsymbol{\epsilon}_e$ to obtain the stress state compatible with the material data.

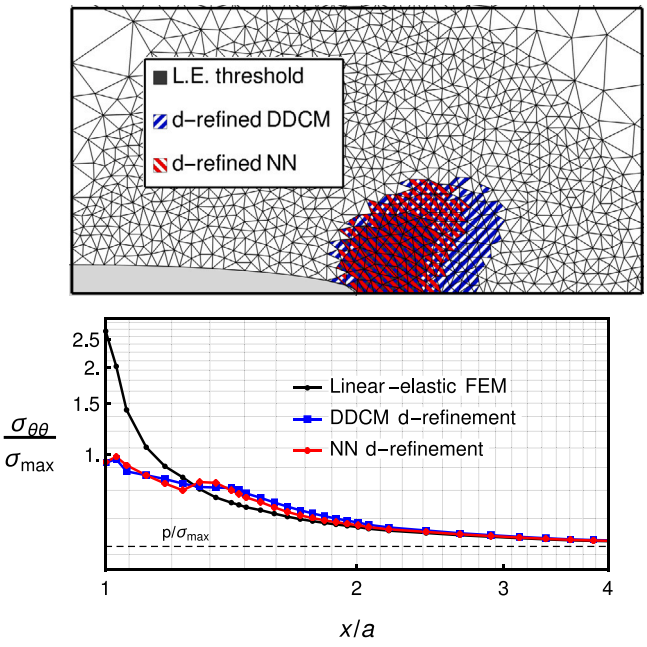

Fig. 13. Results for NN d-refinement. Top: comparison of refined regions. Bottom: comparison hoop stresses along the vertical axis starting from the crack tip ($x = a$).

Adopting this viewpoint, the variational structure presented in Section 2.2.1 is preserved, the Euler–Lagrange equations Eq. (8) being the same. The fixed-point iterative method must change, though (Algorithm 3). Now, once the NN data-driven elements are introduced, their stresses are recomputed based on the current strain-state (i.e., $\boldsymbol{\epsilon}_e^* = \boldsymbol{\epsilon}_e$, $\boldsymbol{\sigma}_e^* = \mathrm{NN}(\boldsymbol{\epsilon}_e^*)$); these recomputed stresses will not satisfy the nodal equilibrium equations, Eq. (8b), what gives rise to non-zero value of the Lagrange multipliers in the next iteration and the necessity to "project onto E" (Eq. (9)), as in the original DDCM method.

This is an acute difference with respect to the original implementation: the manifold $\mathcal{M}$ is assumed to intersect the physically-admissible set E, since the original data satisfies transversality, and to be continuous, so the set of elements to choose during the "projection-onto-material-set" step is infinite (unlike the discrete finite dataset in the traditional approach), hence the norm of the Lagrange multipliers can become arbitrarily small if sufficient fixed-point iterations accrue. Thus, we argue that the natural stop condition for these simulations can be expressed as $\|\boldsymbol{\eta}\|/\|\boldsymbol{u}\| < \mathrm{tol}$, i.e., the vector norm of the multipliers is smaller than a certain percentage of the norm of the nodal displacements. For the following simulation, we chose $\mathrm{tol} = 0.01$ and not to use incremental loading.

The algorithm for this kind of "data-refinement with neural networks" simulations is presented in Algorithm 3. Once again, we highlight that it relies on direct computation of absolute stress values instead of the usual incremental changes necessary to define tangent operators. These computations are easily parallelizable (do not require domain partitions), just like the phase-space searches. In the following simulation, we use again six kernels (although it is also acknowledged that for the relatively small number of refined elements we obtain the speed-up associated to parallelization is not substantial). The threshold to switch elements to data-driven is also the same across d-refinement versions ($\sigma_{yy} > 0.9\sigma_{\text{lim}}$).

Once the NN is trained and the methodology has been established, we solve the stress concentration at the crack tip of the material problem, and compare with the DDCM implementation of d-refinement.

- The NN d-refinement solver is about 3x faster: 2.5 s compared to 8 s in DDCM d-refinement (for the tolerance being used, $\mathrm{tol} = 0.01$).

- It ends up refining less elements: 162 versus 233. Both refined regions subsume the elements that the linear solver predicts to undergo non-linearity, but their geometry is different (Fig. 13 top), even though the threshold to become DD is left unchanged $\sigma_m > 0.9\sigma_{\text{lim}}$. This is attributed to the fact of the greater range of choice associated to the continuous manifold.
- The relative difference between d-refinement implementations computed using distances as in Eq. (12) is less than 2%, i.e.,

$$\frac{|\boldsymbol{z}^{\text{DDCM}} - \boldsymbol{z}^{\text{NN}}|}{|\boldsymbol{z}^{\text{DDCM}}|} \approx 0.016\,. \tag{16}$$

- The hoop stress distribution along the symmetry edge (Fig. 13 bottom) is very similar in both cases.

Algorithm 3 D-refinement framework with neural network modeling ("NN d-refinement")

Require: tol for convergence condition, $\forall e = 1, \dots, N_e$, compatibility matrices $\boldsymbol{B}_e$, $\mathbb{D}_e$ matrices containing elastic constants, local datasets D_e and $\mathbb{C}_e$ matrices containing distance constants; $\forall i = 1, \dots, N_{\text{dof}}$, final external forces F_i or boundary conditions.
(i) Set $j = 1$, $S_2^{(1)} = \emptyset$ and $S_1^{(1)} = \{1, 2, 3, \dots, N_e\}$.
(ii) Incremental loading. Initialize $l = 1$
for $l \leq N_{\text{steps}}$ **do**
Set loading level $\boldsymbol{f} = l/N_{\text{steps}}\boldsymbol{F}$
  if $|S_2^{(j)}| = 0$ **then**
    Set $\boldsymbol{\eta} = 0$ and solve for $\boldsymbol{u}$ in eq. (8b).
  else
    (ii.a) Initialize $k = 0$
    (ii.b) Project onto E:
     Solve in eq. (8) for $\boldsymbol{u}^{(k)}$ and $\boldsymbol{\eta}^{(k)}$
    for $e \in S_2{}^{(j)}$ **do**
      Compute $\boldsymbol{\epsilon}_e$ and $\boldsymbol{\sigma}_e$ from eq. (9)
    end for
    (ii.c) Local convergence condition:
    if $||\boldsymbol{\eta}^{(k)}||/||\boldsymbol{u}^{(k)}|| <$ tol **then**
      Fixed-point iterations converged, **goto** (iii)
    end if
    (ii.d) Project onto D:
    for $e \in S_2^{(j)}$ **do**
      Make $\boldsymbol{\epsilon}_e^{*(k+1)} = \boldsymbol{\epsilon}_e^{(k)}$ and then compute $\boldsymbol{\sigma}_e^{*(k+1)} = \text{NN}(\boldsymbol{\epsilon}_e^{*(k+1)})$
    end for
     $k \leftarrow k + 1$, **goto** (ii.b)
  end if
  (iii) Check for linear elements above threshold:
   Set $S_1^{(j+1)} = S_1^{(j)}$, $S_2^{(j+1)} = S_2^{(j)}$
  for $e \in S_1^{(j)}$ **do**
   Compute element strain and/or stress: $\boldsymbol{\epsilon}_e$ and $\boldsymbol{\sigma}_e$
    if threshold criterion **then**
      Append e to $S_2^{(j+1)}$, drop it from $S_1^{(j+1)}$
      Initial assignation: make $\boldsymbol{\epsilon}_e^{*(k+1)} = \boldsymbol{\epsilon}_e^{(k)}$ and then compute $\boldsymbol{\sigma}_e^{*(k+1)} = \text{NN}(\boldsymbol{\epsilon}_e^{*(k+1)})$
    end if
  end for
  (vi) Global convergence condition
  if $|S_2^{(j)}| = |S_2^{(j-1)}|$ **then**
    All DD elements converged (or no need for DD), no need of further refinement **exit**
  else
    $j \leftarrow j + 1$
  end if
   $l \leftarrow l + 1$
end for

A thorough comparison between these and other potential approaches is a subject of ongoing research.

References

Bonatti, Colin, Mohr, Dirk, 2021. One for all: Universal material model based on minimal state-space neural networks. Sci. Adv. 7 (26), eabf3658.

Bulin, Johannes, Hamaekers, Jan, Ariza, Pilar, Ortiz, Michael, 2022. Interatomic-potential-free, data-driven molecular dynamics. https://arxiv.org/abs/2208.04937.

Carrara, Pietro, Ortiz, Michael, De Lorenzis, Laura, 2021. Data-driven rate-dependent fracture mechanics. J. Mech. Phys. Solids 155, 104559.

Carrara, Pietro, Ortiz, Michael, De Lorenzis, Laura, 2022. Model-free fracture mechanics and fatigue. In: Aldakheel, Fadi, Hudobivnik, Blaž, Soleimani, Meisam, Wessels, Henning, Weißenfels, Christian, Marino, Michele (Eds.), Current Trends and Open Problems in Computational Mechanics. Springer International Publishing, Cham, pp. 75–82.

Conti, Sergio, Müller, Stefan, Ortiz, Michael, 2018. Data-driven problems in elasticity. Arch. Ration. Mech. Anal. 229 (1), 79–123.

Conti, Sergio, Müller, Stefan, Ortiz, Michael, 2020. Data-driven finite elasticity. Arch. Ration. Mech. Anal. 237 (1), 1–33.

Deshpande, Vikram, Fleck, Norman, Ashby, Michael, 2001. Effective properties of the octet-truss lattice material. J. Mech. Phys. Solids 49 (8), 1747–1769.

Dou, Nicholas G., Jagt, Robert A., Portela, Carlos M., Greer, Julia R., Minnich, Austin J., 2018. Ultralow thermal conductivity and mechanical resilience of architected nanolattices. Nano Lett. 18 (8), 4755–4761, PMID: 30022671.

Eggersmann, Robert, Kirchdoerfer, Trenton, Reese, Stephanie, 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.

Feyel, Frédéric, 1999. Multiscale FE2 elastoviscoplastic analysis of composite structures. Comput. Mater. Sci. 16 (1), 344–354.

Fleck, Norman, Deshpande, Vikram, Ashby, Michael, 2010. Micro-architectured materials: past, present and future. Proc. R. Soc. A 466 (2121), 2495–2516.

Garcia-Suarez, Joaquin, Cornet, Arthur, Wattel, Sacha, Molinari, Jean-François, 2022. Data-driven numerical site response. arxiv.org/abs/2209.12800.

Griffith, Alan A., 1921. VI. The phenomena of rupture and flow in solids. Philos. Trans. R. Soc. Lond. Ser. A Contain. Pap. Math. Phys. Character 221 (582–593), 163–198.

He, Xiaolong, Chen, Jiun-Shyan, 2022. Thermodynamically consistent machine-learned internal state variable approach for data-driven modeling of path-dependent materials. Comput. Methods Appl. Mech. Engrg. 402, 115348, A Special Issue in Honor of the Lifetime Achievements of J. Tinsley Oden.

Hutchinson, John W., 1983. Fundamentals of the phenomenological theory of nonlinear fracture mechanics. J. Appl. Mech. 50 (4b), 1042–1051.

Hutchinson, John W., 1987. Crack tip shielding by micro-cracking in brittle solids. Acta Metall. 35 (7), 1605–1619.

Karapiperis, Konstantinos, Harmon, John, Andò, Edward, Viggiani, Gioacchino, Andrade, José E., 2020. Investigating the incremental behavior of granular materials with the level-set discrete element method. J. Mech. Phys. Solids 144, 104103.

Karapiperis, Konstantinos, Stainier, Laurent, Ortiz, Michael, Andrade, José E., 2021. Data-Driven multiscale modeling in mechanics. J. Mech. Phys. Solids 147, 104239.

Kirchdoerfer, Trenton, Ortiz, Michael, 2016. Data-driven computational mechanics. Comput. Methods Appl. Mech. Engrg. 304, 81–101.

Kirchdoerfer, Trenton, Ortiz, M., 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.

Korzeniowski, Tim Fabian, Weinberg, Kerstin, 2021. A multi-level method for data-driven finite element computations. Comput. Methods Appl. Mech. Engrg. 379, 113740.

Korzeniowski, Tim Fabian, Weinberg, Kerstin, 2022. Data-driven finite element computation of open-cell foam structures. Comput. Methods Appl. Mech. Engrg. 400, 115487.

Kovachki, Nikola, Liu, Burigede, Sun, Xingsheng, Zhou, Hao, Bhattacharya, Kaushik, Ortiz, Michael, Stuart, Andrew, 2022. Multiscale modeling of materials: Computing, data science, uncertainty and goal-oriented optimization. Mech. Mater. 165, 104156.

Lefik, M., Schrefler, B.A., 2003. Artificial neural network as an incremental non-linear constitutive model for a finite element code. Comput. Methods Appl. Mech. Engrg. 192 (28), 3265–3283, Multiscale Computational Mechanics for Materials and Structures.

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.

Masi, Filippo, Stefanou, Ioannis, 2022. Multiscale modeling of inelastic materials with Thermodynamics-based Artificial Neural Networks (TANN). Comput. Methods Appl. Mech. Engrg. 398, 115190.

Masi, Filippo, Stefanou, Ioannis, Vannucci, Paolo, Maffi-Berthier, Victor, 2021. Thermodynamics-based Artificial Neural Networks for constitutive modeling. J. Mech. Phys. Solids 147, 104277.

Meza, Lucas R., Zelhofer, Alex J., Clarke, Nigel, Mateos, Arturo J., Kochmann, Dennis M., Greer, Julia R., 2015. Resilient 3D hierarchical architected metamaterials. Proc. Natl. Acad. Sci. 112 (37), 11502–11507.

Ortiz, Michael, 1987. A continuum theory of crack shielding in ceramics. J. Appl. Mech. 54 (1), 54–58.

Pineau, Andre, Pardoen, Thomas, 2007. 2.06 - Failure of metals. In: Milne, I., Ritchie, R.O., Karihaloo, B. (Eds.), Comprehensive Structural Integrity. Pergamon, Oxford, pp. 684–797.

Platzer, Auriane, Leygue, Adrien, Stainier, Laurent, Ortiz, Michael, 2021. Finite element solver for data-driven finite strain elasticity. Comput. Methods Appl. Mech. Engrg. 379, 113756.

Portela, Carlos M., Edwards, Bryce W., Veysset, David, Sun, Yuchen, Nelson, Keith A., Kochmann, Dennis M., Greer, Julia R., 2021. Supersonic impact resilience of nanoarchitected carbon. Nature Mater. 20 (11), 1491–1497.

Salahshoor, Hossein, Ortiz, Michael, 2023. Model-free Data-Driven viscoelasticity in the frequency domain. Comput. Methods Appl. Mech. Engrg. 403, 115657.

Shaikeea, Angkur Jyoti Dipanka, Cui, Huachen, O'Masta, Mark, Zheng, Xiaoyu Rayne, Deshpande, Vikram Sudhir, 2022. The toughness of mechanical metamaterials. Nature Mater. 21 (3), 297–304.

Wolfram, Stephen, 2000. The Mathematica Book, Vol. 4. Cambridge University Press Cambridge.

Yang, Jie, Huang, Wei, Huang, Qun, Hu, Heng, 2022. An investigation on the coupling of data-driven computing and model-driven computing. Comput. Methods Appl. Mech. Engrg. 393, 114798.