Received: 19 January 2018 | Revised: 17 January 2019 | Accepted: 3 February 2019
DOI: 10.1002/nme.6029

RESEARCH ARTICLE

WILEY

SoftFEM: The Soft Finite Element Method

Jose M. Peña[1,2,3] | Antonio LaTorre[3] | Antoine Jérusalem[2]

[1]Lurtis Ltd., Oxford, UK
[2]Department of Engineering Science, University of Oxford, Oxford, UK
[3]Center for Computational Simulation, Universidad Politécnica de Madrid, Campus de Montegancedo, Boadilla, Spain

Correspondence
Jose M. Peña, Lurtis Ltd., Oxford OX3 7AN, UK.
Email: jm.penya@lurtis.com

Antoine Jérusalem, Department of Engineering Science, University of Oxford, Oxford OX1 3PJ, UK.
Email: antoine.jerusalem@eng.ox.ac.uk

Funding information
Spanish Ministry of Science and Innovation, Grant/Award Number: TIN2017-83132-C2-2-R; Universidad Politécnica de Madrid, Grant/Award Number: PINV-18-XEOGHQ-19-4QTEBP

Summary

While the finite element method (FEM) has now reached full maturity both in academy and industry, its use in optimization pipelines remains either computationally intensive or cumbersome. In particular, currently used optimization schemes leveraging FEM still require the choice of dedicated optimization algorithms for a specific design problem, and a “black box” approach to FEM-based optimization remains elusive. To this end, we propose here an integrated finite element-soft computing method, ie, the soft FEM (SoftFEM), which integrates a finite element solver within a metaheuristic search wrapper. To illustrate this general method, we focus here on solid mechanics problems. For these problems, SoftFEM is able to optimize geometry changes and mechanistic measures based on geometry constraints and material properties inputs. From the optimization perspective, the use of a fitness function based on finite element calculation imposes a series of challenges. To bypass the limitations in search capabilities of the usual optimization techniques (local search and gradient-based methods), we propose, instead a hybrid self adaptive search technique, the multiple offspring sampling (MOS), combining two metaheuristics methods: one population-based differential evolution method and a local search optimizer. The formulation coupling FEM to the optimization wrapper is presented in detail and its flexibility is illustrated with three representative solid mechanics problems. More particularly, we propose here the MOS as the most versatile search algorithm for SoftFEM. A new method for the identification of nonfully determined parameters is also proposed.

KEYWORDS

finite element method, heuristic optimization, soft computing, SoftFEM, solid mechanics

1 | INTRODUCTION

Building on the early work of mathematicians, physicists, and engineers, the finite element method (FEM) was first proposed in the mid-50s, with the first mention of “finite element” being proposed by Clough in 1960.[1] The FEM has since established itself as the method of predilection for the study of computational mechanics of materials and structures. While it has benefited from many complex evolutions such as the extended FEM (XFEM)[2] or the discontinuous Galerkin method,[3] it has generally suffered from a lack of computational agility when used within optimization pipelines.

Optimization problems in computational solid mechanics problems generally aim at identifying the material properties and/or the topological design of a given solid for the optimization of mechanistic or geometrical measures under a set of loading conditions. One approach consists in covering the entire space of input parameter combinations, running a simulation for each one, and choosing the set of parameters that best optimize the problem. The alternative is to run the simulations only for a sampled representative subset of the space of parameter combinations. While the former is computationally intensive and lacks scalability, the latter is prone to produce suboptimal solutions. In practice, an optimization problem needs a trade-off between both approaches, in turn, requiring an expert knowledge in the design of the simulations and in the choice of an adequate stochastic optimization algorithm.

The stochastic nature of material properties has been extensively studied through the stochastic finite element method (SFEM)[4]; coming in three variants: (i) perturbation approach, (ii) spectral decomposition, or (iii) Monte Carlo simulations. The SFEM and subsequent alternatives, eg, the Bayesian FEM,[5] have been used in multiple applications, eg, polycrystalline microstructures,[6] elastodynamics,[7] tribology problems in MEMS.[8] While these methods aim at identifying the effect of material properties variations on the resulting structural behavior, they are unable to identify material properties for a target structural output. A paradigm thus acting as a "black box" with no assumptions on the characteristics of the problem being optimized to the user still remains elusive.

Geometrical optimization and, in the case of the FEM, adaptive remeshing have been the subject of numerous research programs. One of the main challenges is the need for a usable scheme to deal with multiple constraints (both geometric and physical) and complex boundary conditions.[9] Unfortunately, most of the proposed methods rely on a very careful definition of the conditions upon which a solution might be reached. In some cases, a coupling with material parameters optimization was proposed[10,11] but focused on very particular constraint spaces. Again, to the best of the knowledge of the authors, a flexible "black box" method allowing for multiple complex constraints has not been proposed. In particular, such method would need to be flexible enough to capture optimization problems with continuous behaviors (where local search algorithms are a priori more efficient) and optimization problems with noncontinuous problems (where population-based, eg, evolutionary, algorithms are a priori more efficient) and be able to distinguish when to use one or the other independently.

The use of optimization techniques is a common topic in many industrial and scientific problems.[12] For cases in which an analytical description of the problem exists, analytical optimization methods are generally directly available. Otherwise, metaheuristic optimization techniques can be used to solve the problem as a "black box". These methods use a stochastic trial-and-error framework with an explicit or implicit probabilistic mechanism guiding the search process by progressive generation of new candidate solutions. Along with learning methods, they are the two soft computing[13] frameworks increasingly used to work with imprecision, uncertainty, partial truth, and approximation.

In this paper, we propose a new formal approach applying soft computing to computational mechanics finite element material properties and topological optimization. Section 2 surveys the related work on soft computing metaheuristics for optimization applied to computational mechanics. In Section 3, we introduce the formalism to link both methods (in particular, Section 3.3 presents how both methods are coupled). We then propose a taxonomy of the different practical applications of this new method to other additional real-world problems in Section 4. Finally, Section 5 presents the conclusions of the application of this method and highlights its potential.

2 | RELATED WORK

Different metaheuristic methods are commonly proposed in the general soft computing literature. Some of them use a population of candidate solutions such as genetic algorithms (GAs),[14] estimation of distribution algorithms (EDAs),[15] differential evolution (DE),[16] particle swarm optimization (PSO),[17] or ant colony optimization[18]; other methods improve a single solution by a local search mechanism, such as multiple trajectory search (MTS),[19] tabu search (TS),[20] or variable neighborhood search (VNS).[21] Finally, in the last few years, hybrid and memetic algorithms (MAs)[22] that combine multiple metaheuristic methods have become the best performing alternatives in most of the generalized benchmarks.[23]

Although not having been formalized as an integrated method, the use of heuristic optimization has already been applied to different computational mechanics problems.[24] These approaches have mainly considered GA as the preferred optimization techniques, mainly based on their simplicity and their mainstream use as a heuristic optimization method.[25,26] In particular, Corriveau et al[27] applied GA to optimize FEM, for benchmark use cases and to a real-case scenario of fillets at the crown-blade junctions of a hydroelectric turbine. Similar application papers have also applied GA to computational fluid dynamics.[28,29]

Alternative methods to GAs have also been proposed, to a lesser extent, such as sequential approximation optimization,[30] evolutionary strategies,[31,32] DE,[33] PSO,[34] harmony search,[35] and EDAs.[36]

The use of local optimization methods has also been referred in the literature, being simulated annealing (SA)[37,38] one of the most common approaches, but other approaches such as VNS[39] and TS[40] have also been considered. The local optimization approaches based on heuristic methods have been one of the first approaches to overcome the limitations of the traditional analytical or gradient-based optimization approaches.[41]

The approaches based on hybrid optimization methods and MAs are the ones less explored in the field of computational mechanics, with some relevant references such as GA and SA,[42] GA and gradient-based methods,[43] two local optimization methods (SA and TS),[44] or multiobjective DE and incremental learning,[45,46] which compares multiple combinations of MA, GA, SA, TS, and VNS.

Additionally, multiobjective problems draw an additional axis in the application of heuristic approaches to computational mechanics. These problems are defined when multiple and (potentially) competing goals[47-49] are optimized. In many cases, the proposed solutions are alternatives with different trade-off levels among the different objectives. These approaches have been traditionally tackled by Pareto-based multiobjective optimization methods, such as multiobjective GA,[50] nondominated sorting GA II,[51-53] or strength Pareto evolutionary algorithm 2.[54]

Finally, approaches based on metamodelling or surrogate-based methods have been proposed to approximate FEM calculations by a function that would provide analytical solution or a better performance evaluation compared to the actual finite element solving. These approaches have been solved by methods ranging from random search[55] to GA[56] or SA.[57]

Although there have been several approaches to the application of soft computing methods and heuristic optimization to computational mechanics and, in particular, FEM, we have not found a methodological framework that couples concepts from both fields in a formal model. Moreover, some of the most advanced methods in continuous optimization, resulting from international competitions and reference benchmarks, have not been tested in this particular domain, which, in most cases, still uses rather classical heuristic optimization methods, such as GA. This paper aims to explore the formalization and application of advanced hybrid methods such as multiple offspring sampling (MOS).

3 | GENERAL FRAMEWORK

The finite element and stochastic heuristic optimization formulations are first presented. Both formulations are then combined into the proposed SoftFEM.

3.1 | Finite element formulation

In the following, we briefly review the fundamentals of the FEM applied to computational mechanics of materials for large deformation.

3.1.1 | Mechanical framework

Let $\boldsymbol{\varphi} : \Omega_0 \rightarrow \Omega$ be a function that maps a material point $\boldsymbol{X} \in \Omega_0 \subset \mathbb{R}^3$ in the reference configuration to its corresponding point $\boldsymbol{x} = \boldsymbol{\varphi}(\boldsymbol{X}) \in \Omega \subset \mathbb{R}^3$ in the current configuration (see Figure 1). The deformation gradient tensor $\boldsymbol{F}$ is defined as

$$\boldsymbol{F} = \text{Grad}\ \boldsymbol{x}, \tag{1}$$

where Grad is the gradient operator with respect to the reference configuration.

Balance of momentum. The equation of the balance of linear momentum with respect to the reference (or undeformed) configuration can be written as

$$\text{Div}\ \boldsymbol{P} + \rho_0 \boldsymbol{b} = \rho_0 \ddot{\boldsymbol{x}}, \quad \forall \boldsymbol{X} \in \Omega_0, \tag{2}$$

where ρ_0 is the mass density (per unit reference volume) and Div is the divergence operator with respect to the reference configuration. In this expression, the second-order tensor $\boldsymbol{P}$ and vector $\boldsymbol{b}$ stand for the first Piola-Kirchhoff stress tensor and the body force vector per unit mass, respectively.

The balance of angular momentum is generally not imposed directly through the weak form. Instead, the constitutive model is chosen such that it is at all time verified, ie, $\boldsymbol{P} \cdot \boldsymbol{F}^T$ is symmetric.

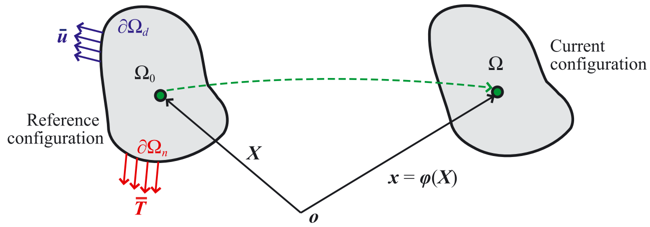

FIGURE 1 Deformable body and the applied boundary conditions in the reference (undeformed) and current (deformed) configurations [Colour figure can be viewed at wileyonlinelibrary.com]

Boundary conditions. The Neumann and Dirichlet boundary conditions, $\bar{\boldsymbol{T}}$ on $\partial\Omega_n$ and $\bar{\boldsymbol{u}}$ on $\partial\Omega_d$, respectively, are imposed in the reference configuration by the following:

$$\begin{cases} \boldsymbol{P}\cdot\boldsymbol{N} = \bar{\boldsymbol{T}},\ \forall \boldsymbol{X} \in \partial\Omega_n \\ \boldsymbol{u} = \bar{\boldsymbol{u}},\ \forall \boldsymbol{X} \in \partial\Omega_d, \end{cases} \tag{3}$$

where $\boldsymbol{N}$ is the normal to the boundary in the reference configuration (see Figure 1).

3.1.2 | Finite element discretization

Definition 1. The **weak form of the balance of linear momentum** (see Equation (2)) is formulated as follows. For all arbitrary admissible virtual displacement $\boldsymbol{\eta}$, with $\boldsymbol{\eta}(\boldsymbol{X}) = \boldsymbol{0}$ for all $\boldsymbol{X} \in \partial\Omega_d$,

$$\int_{\partial\Omega_n} \bar{\boldsymbol{T}}\cdot\boldsymbol{\eta}\,\mathrm{d}S + \int_{\Omega_0} \rho_0\boldsymbol{b}\cdot\boldsymbol{\eta}\,\mathrm{d}V = \int_{\Omega_0} \boldsymbol{P} : \mathrm{Grad}\,\boldsymbol{\eta}\,\mathrm{d}V + \int_{\Omega_0} \rho_0\ddot{\boldsymbol{x}}\cdot\boldsymbol{\eta}\,\mathrm{d}V. \tag{4}$$

Definition 2. The relation between $\boldsymbol{P}$ and $\boldsymbol{F}$ is defined by **the material constitutive law $\boldsymbol{P}(\boldsymbol{F}, \boldsymbol{\Xi}, \boldsymbol{\Lambda})$**, where $\boldsymbol{\Xi}$ is a set of internal variables and $\boldsymbol{\Lambda}$ is the set of material properties (note that the density ρ_0 is considered here as a material property).

Let $\Omega_{0h} = \bigcup_e \Omega^e_{0h}$ be the finite element approximation to the actual undeformed body Ω_0 (see Figure 2), and $\boldsymbol{\varphi}_h$ be a piecewise polynomial approximation of degree k to the actual deformation $\boldsymbol{x} = \boldsymbol{\varphi}(\boldsymbol{X})$, such that

$$\boldsymbol{\varphi}_h \in \left\{ \boldsymbol{\varphi}_h \in C^0(\Omega_{0h}) \middle| \boldsymbol{\varphi}^e_h = \boldsymbol{\varphi}_h\Big|_{\Omega^e_{0h}} \in \mathbb{P}^k\left(\Omega^e_{0h}\right), \forall \Omega^e_{0h} \in \Omega_{0h} \right\}. \tag{5}$$

Following the Galerkin method, $\boldsymbol{\eta}_h$, the polynomial approximation to the virtual displacement $\boldsymbol{\eta}$, is chosen in the same ensemble with the additional condition that $\boldsymbol{\eta}^e_h = \boldsymbol{\eta}_h|_{\Omega^e_{0h}} = \boldsymbol{0}$ on the discretized boundary $\partial\Omega_{dh}$ of $\partial\Omega_d$ for all elements e.

Polynomial shape functions $N^e_a(\boldsymbol{\xi})$ are defined for each element e (a is the node number and $\boldsymbol{\xi}$ is the coordinate vector in the element basis or isoparametric coordinate system), such that, in each element,

$$\boldsymbol{x}^e_h(\boldsymbol{X}) = \sum_{a\in\Omega^e_{0h}} N^e_a(\boldsymbol{\xi})\boldsymbol{x}^a \quad \text{and} \quad \boldsymbol{\eta}^e_h(\boldsymbol{X}) = \sum_{a\in\Omega^e_{0h}} N^e_a(\boldsymbol{\xi})\boldsymbol{\eta}^a, \tag{6}$$

where $\boldsymbol{x}^a$ and $\boldsymbol{\eta}^a$ are the nodal position (or the solution) and virtual displacement of node a, respectively. Note that the mapping from the element basis $\boldsymbol{\xi}$ to the reference coordinate system $\boldsymbol{X} = \boldsymbol{\Phi}(\boldsymbol{\xi})$ is prescribed for all elements.

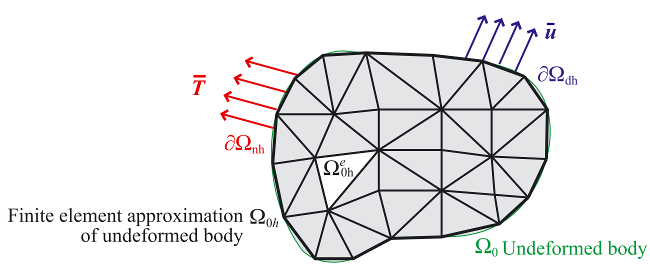

FIGURE 2 Finite element discretization of the undeformed body Ω_0 into elements Ω^e_{0h}, such that $\Omega_{0h} = \bigcup_e \Omega^e_{0h}$ [Colour figure can be viewed at wileyonlinelibrary.com]

Definition 3. By making use of Equations (4) and (6), the **finite element problem** can then be defined as follows. Find all $\boldsymbol{x}^a \in \mathbb{R}^3$ such that, for all admissible virtual displacement $\boldsymbol{\eta}^b \in \mathbb{R}^3$,

$$\sum_e M^e_{ba} \ddot{\boldsymbol{x}}^a + \sum_e \boldsymbol{f}_b^{e\,\text{int}}(\boldsymbol{x}^a) = \sum_e \boldsymbol{f}_b^{e\,\text{ext}}, \tag{7}$$

where the mass matrix $\boldsymbol{M}^e$, the element internal force vector $\boldsymbol{f}_b^{e\,\text{int}}(\boldsymbol{x}^a)$, and external force vector $\boldsymbol{f}_b^{e\,\text{ext}}$ are

$$M^e_{ba} = \int_{\Omega^e_{0h}} \rho_0 N^e_a N^e_b \,\mathrm{d}V \tag{8}$$

$$\boldsymbol{f}_b^{e\,\text{int}}(\boldsymbol{x}^a) = \int_{\Omega^e_{0h}} \boldsymbol{P}\left(\sum_{a\in\Omega^e_{0h}} N^e_a(\xi)\boldsymbol{x}^a\right) \cdot \boldsymbol{\nabla}_0 N^e_b \,\mathrm{d}V \tag{9}$$

$$\boldsymbol{f}_b^{e\,\text{ext}} = \int_{\partial\Omega^e_{nh}} N^e_b \bar{\boldsymbol{T}} \,\mathrm{d}S + \int_{\Omega^e_{0h}} \rho_0 N^e_b \boldsymbol{b} \,\mathrm{d}V. \tag{10}$$

3.2 | Stochastic heuristic optimization formulation

3.2.1 | Optimization problem

The general case for an optimization problem may be defined as finding the solutions (potentially multiple) that, while satisfying a set of criteria to be considered as feasible, maximize or minimize a particular metric (typically defined in the form of a function or set of functions).

Definition 4. An **optimization problem A** is defined as $\mathbf{A} = (f, \Gamma, \succcurlyeq_\Psi)$, with Γ being the nonempty domain in which the problem is defined. This domain may be either discrete or continuous and n-dimensional. The function f is defined as follows:

$$f : \Gamma \xrightarrow{f} \Psi, \tag{11}$$

where Ψ indicates the codomain of f that can be also discrete or continuous and n-dimensional, in the general case. Finally, $\succcurlyeq_\Psi$ is an order relationship defined over Ψ.

Definition 5. The set $X^* \subseteq \Gamma$ is referred as the **optimal set** of the problem **A** if

$$X^* = \left\{ x \in \Gamma \mid \nexists x' \in \Gamma \mid f(x') \succcurlyeq_\Psi f(x) \right\}. \tag{12}$$

Definition 6. The set of values Z^* is the **optimal outcome** of the problem **A** and is defined as follows:

$$Z^* = \{ z \in \Psi \mid z = f(x) \wedge x \in X^* \}. \tag{13}$$

The **mono-objective minimization** is a particular case of the general optimization case in which $\Psi \equiv \mathbb{R}$ and $\succcurlyeq_\Psi$ is the regular less-than-or-equal-to ($\leq$) operator.

Identically, **mono-objective maximization** is defined as a particular case of the general optimization case, but $\succcurlyeq_\Psi$ is the regular greater-than-or-equal-to ($\geq$) operator.

A **constrained optimization problem** is a particular case of the general optimization problem in which Γ is not only defined by an extensive domain (such as an interval in $\mathbb{R}^n$) but also implicitly by a set of constraints.

Definition 7. A is a **constrained optimization problem** $(f, \Gamma, \succcurlyeq_\Psi, G, H)$, where the two sets G and H are defined as follows:

$$G = \left\{ g_i \mid g_i : \Gamma \xrightarrow{g_i} \mathbb{R} \right\}, \quad \forall i : g_i(x) \leq 0 \tag{14}$$

$$H = \left\{ h_i \mid h_i : \Gamma \xrightarrow{h_i} \mathbb{R} \right\}, \quad \forall i : h_i(x) = 0. \tag{15}$$

The sets G and H are named inequality constraint set and equality constraint set, respectively.

Lemma 1. *For a given constrained optimization problem* $\mathbf{A} = (f, \Gamma, \succcurlyeq_\Psi, G, H)$*, there is an* ***equivalent unconstrained optimization problem*** $\mathbf{A}' = (f, \Gamma', \succcurlyeq_\Psi)$*, where* $\Gamma' \subseteq \Gamma$

$$\Gamma' = \left\{ x \in \Gamma \mid, \forall i : g_i(x) \leq 0 \wedge \forall j : h_j(x) = 0 \right\}. \tag{16}$$

Definition 8. $\mathbf{A}$ is a **continuous optimization problem** if $\Gamma \subseteq \mathbb{R}^n$. The value n is called the dimensionality of the problem. Continuous optimization problems are particular cases of the general optimization case.

The **inequality constraint of an equivalent constrained continuous optimization problem** $\mathbf{A} = (f, \mathbb{R}^n, \succcurlyeq_\Psi, G, H)$ is defined as follows:

$$\exists g_i \in G : g_i(x) := x_j - \overline{\omega_i} \tag{17}$$

$$\exists g_i \in G : g_i(x) := \underline{\omega_i} - x_j, \tag{18}$$

where $[\underline{\omega_i}, \overline{\omega_i}]$ are the nondegenerate intervals in $\mathbb{R}$, where the dimension i of the solution space is defined as $x_i \in [\underline{\omega_i}, \overline{\omega_i}]$.

3.2.2 | Heuristic optimization

Heuristic optimization methods are techniques designed to solve an optimization problem $\mathbf{A} = (f, \Gamma, \succcurlyeq_\Psi)$ under two major assumptions.

- There is no needed a priori knowledge about the properties of the function f to be optimized. It neither needs to be continuous nor derivable, but it must be defined over the whole solution space Γ. This assumption is referred as being "black box optimization".
- The exploration of the solution space is constrained to a finite set of tentative solutions $\Sigma \subset \Gamma$. In the case of Γ being a finite set, then $|\Sigma| \ll |\Gamma|$; otherwise, in the case of Γ being an infinite set, Σ must always be finite.

Definition 9. A **general heuristic optimization problem** is defined as $\mathbf{K} = (\mathbf{A}, s)$, where $\mathbf{A}$ is a general optimization problem and s is a search strategy function defined as follows:

$$s : \mathcal{P}(\Gamma) \xrightarrow{s} \mathcal{P}(\Gamma), \tag{19}$$

where $\mathcal{P}(\Gamma)$ is the power set of Γ, ie, the set of all the possible subsets of Γ. This set is defined for finite sets (cardinality N), countable infinite sets (cardinality $\aleph_0$), or, theoretically, noncountable infinite sets (cardinality $\mathfrak{c} = 2^{\aleph_0}$, cardinality of the continuum).

Definition 10. Σ_0 is called the **starting solution set** for a given heuristic optimization problem $\mathbf{K} = (\mathbf{A}, s)$ and is defined as the result of the search strategy s when it is applied to the empty set of candidate solutions

$$\Sigma_0 = s(\emptyset). \tag{20}$$

Definition 11. The **infinite sequence** $\langle \sigma_i \rangle_{i=0}^{\infty}$ is defined as the heuristic search sequence for a search strategy s, ie,

$$\sigma_i = \begin{cases} \Sigma_0 & \text{if } i = 0 \\ s(\sigma_{i-1}) & \text{otherwise.} \end{cases} \tag{21}$$

Definition 12. The ***n*th-order candidate solution** set for a general heuristic optimization problem $\mathbf{K} = (\mathbf{A}, s)$ is defined as Σ_n and corresponds to the following:

$$\Sigma_n = \bigcup_{i=0}^{n} \sigma_i, \tag{22}$$

where $\langle \sigma_i \rangle_{i=0}^{n}$ is the finite heuristic search sequence for the search strategy s.

Definition 13. The set $X^*_{\mathbf{K}}$ is the **heuristic optimal set** obtained from the general heuristic optimization problem $\mathbf{K} = (\mathbf{A}, s)$, defined as follows:

$$X^*_{\mathbf{K}} = \left\{ x \in \Sigma_\infty \mid \nexists x' \in \Sigma_\infty : f(x') \succcurlyeq_\Psi f(x) \right\}, \tag{23}$$

where Σ_∞ is the limit at infinity of the nth-order solution set, ie,

$$\Sigma_\infty = \lim_{n\to\infty} \Sigma_n. \tag{24}$$

Definition 14. The set $X^*_{\mathbf{K}|n}$ is the ***n*th-order heuristic optimal set** obtained from the $\mathbf{K} = (\mathbf{A}, s)$ general heuristic optimization problem, defined as follows:

$$X^*_{\mathbf{K}|n} = \left\{x \in \Sigma_n | \nexists x' \in \Sigma_n : f(x') \succcurlyeq_\Psi f(x)\right\}. \tag{25}$$

Definition 15. A general heuristic optimization problem $\mathbf{K} = (\mathbf{A}, s)$ completely solves a general optimization problem $\mathbf{A} = (f, \Gamma, \succcurlyeq_\Psi)$ by order n if

$$\forall x \in X^* : x \in X^*_{\mathbf{K}|n}. \tag{26}$$

Definition 16. A general heuristic optimization problem $\mathbf{K} = (\mathbf{A}, s)$ partially solves a general optimization problem $\mathbf{A} = (f, \Gamma, \succcurlyeq_\Psi)$ by order n if

$$\exists x \in X^* : x \in X^*_{\mathbf{K}|n}. \tag{27}$$

3.2.3 | Stochastic heuristic optimization

Stochastic heuristic optimization relies on a stochastic search mechanism to solve complex heuristic optimization problems.

Definition 17. A **stochastic heuristic optimization problem** $\mathbf{K} = (\mathbf{A}, S)$ is an extension of a heuristic optimization problem $\mathbf{K} = (\mathbf{A}, s)$ in which the search strategy s is defined by a random field S defined in a probability space $(\mathcal{P}(\Gamma), S, \mathbb{P})$ and indexed by the topological lattice of $\mathcal{P}(\Gamma)$ with probability $\mathbb{P}$, ie,

$$S = \{S(x) : x \in \mathcal{P}(\Gamma)\}, \tag{28}$$

where each $S(x)$ is a random variable also in the power set $\mathcal{P}(\Gamma)$, ie,

$$\mathbb{P}(S(x) = y) \geq 0, \quad \forall y \in \mathcal{P}(\Gamma). \tag{29}$$

Definition 18. The **starting solution set**, in the case of a stochastic heuristic optimization problem $\mathbf{K} = (\mathbf{A}, S)$, is defined as a random variable (we use the same notation Σ_0 defined as equivalent to $S(\emptyset)$), ie,

$$\mathbb{P}(\Sigma_0 = X) = \mathbb{P}(S(\emptyset) = X). \tag{30}$$

Definition 19. Equivalently, the **infinite sequence** $\langle\sigma_i\rangle_{i=0}^\infty$ is also redefined as the sequence of random variables (stochastic processes) of a stochastic heuristic optimization problem $\mathbf{K} = (\mathbf{A}, S)$.

In the case of $i = 0$, the sequence begins with the starting solution set random variable*:

$$\sigma_0 = \Sigma_0. \tag{31}$$

For the rest of the cases, $i > 0$, the next elements of the sequence are defined as the random variable

$$\mathbb{P}\left(\sigma_i = X \,\middle|\, \mathcal{S}(\sigma)_0^{i-1}\right) = \mathbb{P}\left(S(\sigma_{i-1}) = X \,\middle|\, \mathcal{S}(\sigma)_0^{i-1}\right) \geq 0, \tag{32}$$

being $\mathcal{S}(\sigma)_0^{i-1}$ the joint event defined as follows:

$$\mathcal{S}(\sigma)_0^{i-1} \equiv S(\sigma_{i-2}) = \sigma_{i-1}, S(\sigma_{i-3}) = \sigma_{i-2}, \ldots, S(\sigma_1) = \sigma_0. \tag{33}$$

*σ and Σ notations are overloaded to be treated as random variables in the domain of $\mathcal{P}(\Gamma)$ for the case of stochastic heuristic optimization, rather than subsets of Γ.

Definition 20. The ***n*th-order candidate solution set** for a stochastic heuristic optimization problem $\mathbf{K} = (\mathbf{A}, S)$ is defined as Σ_n and corresponds to the random variable

$$\mathbb{P}(\Sigma_n = X) = \mathbb{P}\left(\bigcup_{i=0}^{n} \sigma_i = X \,\middle|\, S(\sigma)_0^n\right). \tag{34}$$

Definition 21. The **stochastic optimal set** is defined as a random variable $\mathcal{X}^*_{\mathbf{K}}$ obtained from the stochastic heuristic optimization problem $\mathbf{K} = (\mathbf{A}, S)$, ie,

$$\mathbb{P}\left(\mathcal{X}^*_{\mathbf{K}} = X\right) = \mathbb{P}\left(\{x \in \Sigma_\infty \mid \nexists x' \in \Sigma_\infty : f(x') \succcurlyeq_\Psi f(x)\} = X \,\middle|\, S(\sigma)_0^n\right) \tag{35}$$

where Σ_∞ is the limit at infinity of the nth-order solution set random variable

$$\Sigma_\infty = \lim_{n\to\infty} \Sigma_n. \tag{36}$$

Definition 22. The random variable $\mathcal{X}^*_{\mathbf{K}|n}$ is the ***n*th-order stochastic optimal set** obtained from the $\mathbf{K} = (\mathbf{A}, S)$ general stochastic heuristic optimization problem, defined as follows:

$$\mathbb{P}\left(\mathcal{X}^*_{\mathbf{K}|n} = X\right) = \mathbb{P}\left(\left\{x \in \Sigma_n \mid \nexists x' \in \Sigma_n : f(x') \succcurlyeq_\Psi f(x)\right\} \,\middle|\, S(\sigma)_0^n\right). \tag{37}$$

Definition 23. A general stochastic heuristic optimization problem $\mathbf{K} = (\mathbf{A}, S)$ completely solves a general optimization problem $\mathbf{A} = (f, \Gamma, \succcurlyeq_\Psi)$ by order n with a probability $\rho_{\mathbf{K}|n}$ if

$$\mathbb{P}\left(\mathcal{X}^*_{\mathbf{K}|n} = X^*\right) = \rho_{\mathbf{K}|n}. \tag{38}$$

Definition 24. A general stochastic heuristic optimization problem $\mathbf{K} = (\mathbf{A}, S)$ partially solves a general optimization problem $\mathbf{A} = (f, \Gamma, \succcurlyeq_\Psi)$ by order n with a probability $\mu_{\mathbf{K}|n}$ if

$$\mathbb{P}\left(\mathcal{X}^*_{\mathbf{K}|n} \cap X^* \neq \varnothing\right) = \mu_{\mathbf{K}|n}. \tag{39}$$

3.2.4 | Markovian analysis of the stochastic heuristic optimization

Let $\mathbf{K} = (\mathbf{A}, S)$ be a stochastic heuristic optimization problem. Although, in theory, S may be any random field, in practice, S behaves as a Markov random field exhibiting the local Markov property

$$\mathbb{P}\left(S(x) = y \,\middle|\, S(x') = y', x \neq x'\right) = \mathbb{P}\left(S(x) = y \,|\, \Delta_x\right), \tag{40}$$

where Δ_x is the set of random variables conditions corresponding to the immediate lower neighbors of x in the lattice graph

$$\Delta_x = \left\{S(x') = y' : x' = x/\{e\} \mid e \in x\right\}. \tag{41}$$

In other words, the probability that the random variable $S(x)$ assumes a value, for a subset x of Γ, depends on the values of the random variables of all the subsets with the same elements as in x except one. This means that search strategies produce similar subsets when the input subsets are also similar.

This property is referred as the biased search strategy that is based on the rationale that any effective search strategy is never uniformly random and tries to take advantage of the information about the already visited elements of the domain Γ. According to this information, the search strategies may perform two different kinds of actions.

1. **Exploration**: Try to generate elements in the areas of the search space where the strategy has not generated elements yet.
2. **Exploitation**: Try to generate elements in the proximity of the already obtained (good) elements.

In a general optimization problem $\mathbf{A} = (f, \Gamma, \succcurlyeq_\Psi)$, metric d is defined on the partition set of the problem domain Γ as follows:

$$d : \mathcal{P}(\Gamma) \times \mathcal{P}(\Gamma) \xrightarrow{d} \mathbb{R}^+, \tag{42}$$

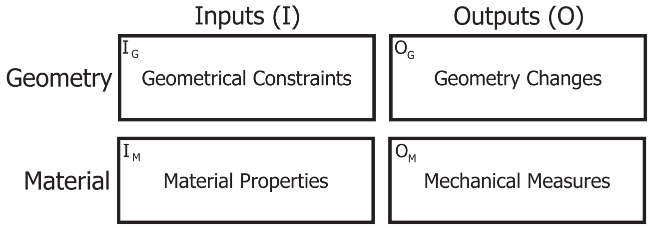

FIGURE 3 The four categories defining the taxonomy of the different problem configurations; the vertical axis distinguishes geometrical information from material and mechanical information, whereas the horizontal one distinguishes input information from expected outputs

where d satisfies all the regular metric properties (nonnegativity, identity of indiscernible, symmetry, and triangular inequality).

Definition 25. Let $\mathbf{K} = (\mathbf{A}, S)$ be a stochastic heuristic optimization problem for a general optimization problem $\mathbf{A} = (f, \Gamma, \succcurlyeq_\Psi)$, and d a metric defined on Γ. We define a $[\underline{\rho_0}, \overline{\rho_0}]$**-intensity explorative strategy** S as a random field such as

$$\underline{\rho_0} d(X, Y) \leq \mathbb{P}(S(X) = Y) \leq \overline{\rho_0} d(X, Y). \tag{43}$$

In practice, when condition Equation (43) is met, stochastic search strategies are assumed to behave according to a Markov Chain model, simplifying the calculation of the probability as follows:

$$\mathbb{P}\left(S(\sigma_{i-1}) = X \,\middle|\, S(\sigma)_0^{i-1} \right) = \mathbb{P}\left(S(\sigma_{i-1}) = X \,\middle|\, S(\sigma_{i-2}) = \sigma_{i-1} \right). \tag{44}$$

3.3 | SoftFEM

SoftFEM aims at wrapping the FEM formulation with a metaheuristic optimization search. In particular, the method identifies material properties and/or geometrical constraints that maximize/minimize either geometrical changes or mechanical measures†:

- **Inputs**: explicit (fixed values of the simulation model) or implicit (free variables defined as either range of possible values or as a results of a series of constraints);
- **Outputs**: derived data resulting from the finite element simulation for a given set of inputs, these data are the objective measure to optimize (maximize or minimize).

An orthogonal representation of this approach is given in Figure 3.

This input and output information is projected into a general optimization problem $\mathbf{A_{fem}} = (f, \Gamma, \succcurlyeq_\Psi)$, where

$$\Gamma \subseteq I_G \times I_M \tag{45}$$

$$\Psi \subseteq O_G \times O_M, \tag{46}$$

where I_M corresponds to the set of material properties, I_G to the original geometry, O_M to the state of deformation and stress of the deformed body, and O_G to the geometry of the deformed body

$$I_M = \mathbf{\Lambda} \tag{47}$$

$$I_G = \Omega_o \tag{48}$$

$$O_M = \{ \mathfrak{F} = \boldsymbol{F}(\boldsymbol{X}) \mid \boldsymbol{X} \in \Omega_0 \} \times \{ \mathfrak{p} = \boldsymbol{P}(\boldsymbol{X}) \mid \boldsymbol{X} \in \Omega_0 \} \tag{49}$$

$$O_G = \Omega_h, \tag{50}$$

where Ω_h is the finite element deformed geometry.

This general optimization problem $\mathbf{A_{fem}}$ can be treated as the equivalent stochastic heuristic optimization problem $\mathbf{K_{fem}} = (\mathbf{A_{fem}}, S)$, with adequate metrics.

†Note that mass (input or output) is implicitly included under both geometry (volume) and material/mechanical (density).

Example 1. As an example of this integrated framework, we may analyze a case in which we want to find the material properties $I = I_M = \mathbf{\Lambda}$ that optimizes the geometry of the finite element body after deformation $O = O_G = \Omega_h$, for instance to approximate a desired final geometry T as a target. In this case, we define the fitness function $f(x) = g(h(x))$, ie,

$$f : \mathbf{\Lambda} \xrightarrow{f} \mathbb{R} \tag{51}$$

$$h : \mathbf{\Lambda} \xrightarrow{h} \Omega_h \tag{52}$$

$$g : \Omega_h \xrightarrow{g} \mathbb{R}, \tag{53}$$

where h is the result of the finite element calculation considering given material properties as inputs and returning the deformed geometry of the final object; and g is the function that quantifies the difference in the geometries between the deformed object and its undeformed counterpart, as measured by any metric between Ω_h and T.

In order to solve this problem using a stochastic heuristic search, a strategy needs to be defined such that the distance metric d in Equation (42) may be expressed as follows:

$$d : \mathcal{P}(\mathbf{\Lambda}) \times \mathcal{P}(\mathbf{\Lambda}) \xrightarrow{d} \mathbb{R}^+. \tag{54}$$

This function may be defined as follows:

$$d(X, Y) = \sum_{x \in X} \left(\min_{y \in Y} (d'(x, y)) \right) + \sum_{y \in Y} \left(\min_{x \in X} (d'(x, y)) \right), \tag{55}$$

where $d'(x, y)$ is a distance metric between two sets of material properties $x, y \in \mathbf{\Lambda}$. This metric can be easily defined as the absolute value (or norm if more than one material property is considered) of the difference between the two sets of material properties.

A potentially successful search strategy S must be chosen such that:

- it satisfies Equation (43) (for d);
- there is an expected improvement in the application strategy to the different solutions to minimize the fitness function[‡]:

$$\mathbb{E} \left(\min_{x \in \sigma_i} f(x) - \min_{y \in \sigma_{i-1}} f(y) \right) \leq 0. \tag{56}$$

To summarize, making use of SoftFEM to solve an optimization problem in computational mechanics requires to (1) set the finite element simulation parameters that are fixed, (2) identify the free variables that describe the search space Γ (either as intervals $[a, b]^n \in \mathbb{R}^n$ or by means of constraints H and G), (3) formulate the output information to optimize in Ψ, and (4) select an appropriate search strategy S.

For the selection of the most appropriate strategy, there are two pieces of information to be considered.

1. The quality of the tentative solution produced at each iteration of the search strategy, defined as the set of pairs

$$(x, f(x)) | \{x \in \sigma_i\}. \tag{57}$$

2. The improvement, in terms of quality, of solutions between iterations of the strategy, defined as follows:

$$(\sigma_i, Q_i - Q_{i-1}), \tag{58}$$

where Q_i is an arbitrary quality measure computed from all the solutions produced in the iteration i by the strategy S that defines the aforementioned sequence of $\langle \sigma_i \rangle_{i=0}^{\infty}$.

Most of the stochastic heuristic optimization techniques define their strategies addressing item 1 (eg, GAs, EDAs, local search methods). Only self-/auto-adaptive methods define their strategies considering also item 2. The approach we finally select for this problem, MOS, belongs to the latter (see Section 4.2). Applying this approach to Equation (44), we tailor a strategy directly on the right-hand side of the equation. The objective is to model the difference in the quality of the whole solution set generated by a strategy between two iterations. This model drives the search process more efficiently between the desired $[\underline{\rho_0}, \overline{\rho_0}]$ boundaries, compared to the indirect search driven by just the individual quality of the solutions in a given iteration.

[‡]Although this condition is neither sufficient nor necessary to ensure optimality, it ensures an incremental improvement in the quality of the solution.

3.4 | Implementation details

Although SoftFEM is a methodological framework to integrate both FEM and heuristic optimization and, in this sense, implementation-agnostic, some technical details concerning the actual way the software is developed, may be highlighted. SoftFEM's main platform relies on GAEDALib,[58] which implements the MOS algorithm that optimizes a given problem. In the case of SoftFEM, this problem is encapsulated into a Python wrapper that allows to integrate different components of a traditional FEM pipeline, such as Abaqus,[59] Gmsh,[60] or OxFEMM,[61] that have been used to conduct the experiments in this paper. This approach is flexible enough to allow any similar pipeline to be constructed within this wrapper, in which Input (I) and Output (O) are exchanged between the pipeline and the optimization framework.

4 | EXPERIMENTAL EVALUATION

The objective of this section is to evaluate the effectiveness of the chosen methods in finding an optimal solution. We focus here on representative computational solid mechanics problems with different continuity properties of the fitness function f (see Equation (11)). Doing so, we analyze the search strategies in terms of how the search space is covered and how fast and how reliably the best solution is reached.

4.1 | Heuristic methods

For the experiments shown in this paper, we have selected three different heuristic algorithms: one population-based method, DE; one local search algorithm, MTS; and one hybrid method, MOS, combining both.

- DE is a population-based heuristic method that iteratively improves a set of solutions (population) by means of a series of recombination operations.[16] This method belongs to the family of evolutionary algorithms and is similar to a GA (both are population-based methods and use several variation operators). The DE-based algorithms have recently shown a much better performance than traditional GAs for the particular case of continuous optimization.[23] In essence, DE makes use of two variation operators: mutation and recombination. Different DE strategies can be proposed depending on the number of mutant vectors, the reference solution, and the recombination operator considered.[16] For this study, we have selected the de/rand/1/exp strategy, which is one of the simplest, yet effective, DE strategies. For each iteration and candidate solution $\boldsymbol{x}^i$, the algorithm randomly selects three other solutions ($\boldsymbol{x}^a$, $\boldsymbol{x}^b$, and $\boldsymbol{x}^c$) and performs the following mutation operation:
$$\boldsymbol{n} = \boldsymbol{x}^c + F(\boldsymbol{x}^a - \boldsymbol{x}^b), \qquad \text{where } \; F \in [0, 2]. \tag{59}$$
The resulting vector $\boldsymbol{n}$ is recombined with the original vector $\boldsymbol{x}^i$ using the exponential crossover operation (randomly selecting, for each vector component j, either the corresponding component from mutated vector n_j or from the original vector x^i_j). Then, the algorithm compares the fitness value of the newly generated solution with the fitness from the original solution, keeping the best one. This process is repeated for each solution in the population and for the whole population several times.
  DE is a well-performing paradigmatic example of a population-based algorithm. These algorithms search characteristics (exploration and exploitation) are regulated by a series of configuration parameters, as shown in Table 1. The values selected for this experimentation have been proved to be successful in previous studies[62] and are used here as a first approximation.
- MTS is a local search technique that iteratively tries to improve a single solution.[19] It is actually made up of three different local search algorithms but, in this work, only the first one was used (referred to as MTS-LS1) as it was shown that it provides most of the search capabilities of the algorithm.[63] This local search iterates through all the components of the solution and, for each component, modifies its value by subtracting/adding a given amount that is a function of a parameter SR. If this new value improves the quality of the solution, then it is kept and the algorithm moves on to the next component. If not, then the original value is restored. The SR parameter is initialized according to the ranges of values of each component and is dynamically adjusted (to half of its value) when all the components of the solution have been explored with no improvement in the fitness value of the solution.
  MTS is a very effective local search algorithm, particularly successful in large scale problems but also very competitive in mid to low dimensionality cases. The performance of the local search depends on the magnitude of the search step SR and how its value is adjusted. In most of the cases and, in particular, for problems with just one or a few local optima,

TABLE 1 Parameters of the different heuristic optimization algorithms (D is the dimension of the problem)

Parameter	Value
Number of fitness evaluations	100 D
DE / MOS population	15
DE F	0.5
DE CR	0.1
DE crossover	Exponential
MOS Min participation	5%
MOS quality function	Fitness increment

Abbreviations: DE, differential evolution; MOS, multiple offspring sampling.

this algorithm exhibits a great exploitative behavior, being able to quickly converge to an optimum. For this reason, it is also a good technique to be combined with other, more explorative, algorithms.

- MOS is a hybrid adaptive search framework that dynamically manages two or more search techniques (subalgorithms).[58,62] The MOS framework distributes the number of fitness evaluations among a group of configured techniques according to their comparative performance in an adaptive dynamic way. There are different configuration parameters for MOS itself (such as the measure to compare performances, the number of fitness evaluations for each comparison round, or the minimum number of fitness evaluations to grant to each techniques) plus the other parameters of the compounding techniques.

4.2 | MOS algorithm

While other hybrid approaches making use of DE and MTS may be available, MOS has been the best performing algorithm in several IEEE CEC LSGO competitions and has shown great capacity to combine the explorative and exploitative characteristics of the managed techniques.[23]

Algorithm 1 presents the pseudo-code of the MOS framework. As can be seen, the framework divides the overall search process into a number of predefined steps (line 2). The participation of each compounding algorithm is then initialized uniformly (line 3) and the overall evolutionary process starts (line 5). At the beginning of each step, the quality of the solutions created by each of the algorithms is evaluated (line 7). The rationale behind measuring the quality of new solutions is to favor those algorithms creating more useful ones (in terms of their fitness, diversity of the population, or whichever characteristic we might consider beneficial for the search).

Algorithm 1 MOS Framework (FE: fitness evaluation; QF: quality function; PF: participation function)

1: Create initial overall population of candidate solutions P_0
2: Initialise step size: $FEs_{step} = FEs_{max}/N_{steps}$
3: Uniformly distribute participation among the N used algorithms (A_j):
   $\forall j\ \Pi_0^{(j)} = \frac{1}{N}$
   $FEs_0^{(j)} = \Pi_0^{(j)} \cdot FEs_{step}$
4: Evaluate initial population P_0
5: **while** FEs_{max} not exceeded **do**
6:     Start new step i
7:     Update Quality: $\forall j\ Q_i^{(j)} = QF(P_{i-1}^{(j)})$
8:     Update Participation: $\forall j\ \Pi_i^{(j)} = PF(Q_i^{(j)})$
9:     Update FEs allocated to each algorithm: $\forall j\ FEs_i^{(j)} = \Pi_i^{(j)} \cdot FEs_{step}$
10:     **for** each algorithm A_j **do**
11:         **while** $FEs_i^{(j)}$ not exceeded **do**
12:             Evolve P_i with A_j
13:         **end while**
14:     **end for**
15: **end while**

Let $\sigma_i^{(j)}$ be the set of solutions generated by the technique j on iteration i, and N the number of techniques. Then,

$$\sigma_i = \bigcup_{j=1}^{N} \sigma_i^{(j)}. \tag{60}$$

An example of an acceptable quality function is given by the following:

$$Q_i^{(j)} = \sum_{x \in \sigma_i^{(j)}} \frac{f(x)}{\left|\sigma_i^{(j)}\right|}. \tag{61}$$

This quality function (line 7) computes, for each algorithm A_j, the average fitness of all the solutions created by the algorithm during the iteration i.

With these updated quality values, the participation (ie, the number of fitness evaluations) that each algorithm will create during the next step is now readjusted (lines 8-9). As in the case of the quality function, multiple participation functions can be defined. In our case, we use a dynamic participation function that increases, proportionally, the participation of the best performing algorithm while decreasing the participation of the remaining algorithms. Equations (62)-(66) show how this participation adjustment takes place. In Equation (66), ξ acts as a modulating factor to control the size of the participation adjustment, ie,

$$\Pi_i^{(j)} = \begin{cases} \Pi_{i-1}^{(j)} + \zeta & \text{if } j \in \Upsilon_i, \\ \Pi_{i-1}^{(j)} - D_i^{(j)} & \text{otherwise,} \end{cases} \tag{62}$$

where Υ_i is the set of techniques with the highest quality function score ($Q_i^{\max}$) and ζ the increment in participation granted to these techniques, ie,

$$\zeta = \frac{\sum_{k \notin \Upsilon_i} D_i^{(k)}}{|\Upsilon_i|} \tag{63}$$

$$\Upsilon_i = \underset{j \in [1,N]}{\arg\max}\, Q_i^{(j)} \tag{64}$$

$$Q_i^{\max} = \max \left\{ Q_i^{(j)} \; \forall j \in [1, N] \right\} \tag{65}$$

$$D_i^{(j)} = \xi \cdot \frac{Q_i^{\max} - Q_i^{(j)}}{Q_i^{\max}} \cdot \Pi_{i-1}^{(j)}, \quad \forall j \in [1,N] \,/\, j \notin \Upsilon_i. \tag{66}$$

Finally, once all the updated participation ratios have been computed, the framework runs, in sequence, all the compounding algorithms, each of them reusing the output population of the previous one (lines 10 to 14).

This process is repeated until the maximum number of fitness evaluations is exhausted.

4.3 | Test cases

The test cases are chosen to represent:

1. a bioengineering problem (chosen here to be a simplified axonal growth[61]) with a strong continuity of the outputs with respect to the inputs, ie, small variations of input values lead to small variations of output values;
2. a plant sciences problem (chosen here to be a simplified plant cell wall shape study[64]) with a "noise" behavior of the outputs;
3. a materials engineering problem (chosen here to be a crystal plasticity problem[65]) with a computing-intensive complex interactions between inputs.

The analysis of the results for each of the test cases is conducted along the following two lines:

- the analysis of the search strategy and how the different algorithms explore the search space and exploit the most promising areas;
- the identification of the input parameters that require additional information to be fully determined.

The problems cover a representative combination of the types of input and output information (see Figure 4).

The maximum amount of fitness evaluations (computational mechanics simulations) granted to each problem to be solved is fixed to 100 times the number of free parameters (dimensions D). The first two problems were solved 25 times for each of the heuristic techniques to have statistically sound data. The third problem was solved once using the most

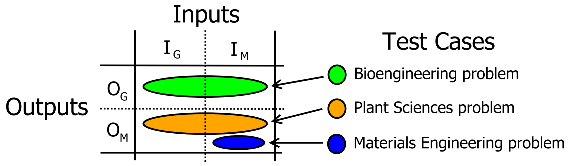

FIGURE 4 The three test cases under study are depicted in the corresponding taxonomy category combination

promising of the heuristic techniques (according to the first two problems). This problem is by far the most computationally expensive of the three test cases and includes the largest number of input parameters (thus, requiring more fitness simulations than the rest of the problems).

4.3.1 | Case 1

Problem description: We consider here a 3D rod of arbitrary dimensions 0.02 × 0.04 × 0.6 (unit is meter by default). The constitutive model is linear elastic with arbitrary material properties. This model is coupled to a growth model where the growth deformation gradient tensor is defined by the following:

$$\mathbf{F}_g(t) = \mathbf{I} + (G_c t - 1)\mathbf{n}_0 \otimes \mathbf{n}_0, \tag{67}$$

where t is the time, G_c is the growth rate parameter, and $\mathbf{n}_0$ is the growth direction vector (see the work of Garcıa-Grajales et al[61] for more details). The rod is clamped at its base and pulled dynamically in the z-direction on its top face at 0.2 m/s for 1 s. The top face is not allowed to move laterally in the y-direction but is free to move in the x-direction. If the rod is pulled faster than it grows, the top face surface area is thus expected to decrease ($\Delta A_{\text{top}} < 0$). If the rod is pulled slower than it grows, the top face area is also expected to change: either increasing if the rod "fattens" ($\Delta A_{\text{top}} > 0$), decreasing because of intense buckling ($\Delta A_{\text{top}} < 0$), or an alternance between both (this behavior is a priori dependent on the aspect ratio of the rod). There should thus be an optimum G_c allowing the top surface area to remain the same (or as close to as possible) after 1 s. The problem is run on OxFEMM[61] (see Figure 5).

Degrees of freedom: 612 nodes / 2324 elements / 1 internal variable.

Input parameters: $\Gamma = \{G_c\}$

- G_c: growth rate parameter; $0 < G_c < 2$

Objective function (minimize): $f = |\Delta A_{\text{top}}(t = 1)|$. The average execution time per simulation is 154 s.

Best overall solution: G_c=0.5846 s^{-1}.

As depicted in Figure 6, both the MTS and the hybrid MOS-based MTS-DE methods systematically solve the problem within the 25 repetitions. However, DE shows a wide dispersion in the quality of the best solution and does not reach the same value for the free parameter. Nonetheless, all the algorithms are able to find the best solution at least once (see Table 2). While DE is not always able to find the same value, its standard deviation is three orders of magnitude smaller than the actual optimal value. The statistical tests show that there is a difference among the three sets of results with a p-value of 5.96×10^{-16} (Kruskal-Wallis multiple test). The pairwise comparisons estimate the difference between the DE results and both MTS and MTS-DE with a p-value of $3.68 \times 10^{-5} < 0.05$ (Wilcoxon test) and 2.97×10^{-10} (Mann-Whitney test) using both Holm correction for the family-wise error rate. The MTS and MTS-DE show no statistical difference.

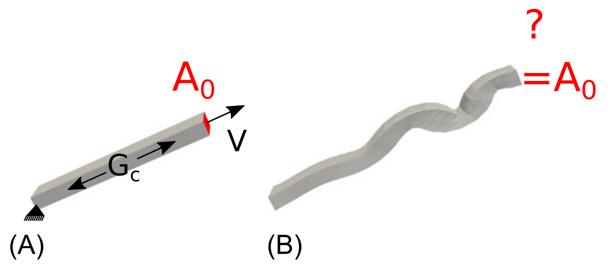

FIGURE 5 Case 1: A, Anchored rod submitted to a pull at a velocity V while growing with a growth rate parameter G_c; B, The optimization problem consists in finding the value of G_c such that the end area remains the same after 1 s [Colour figure can be viewed at wileyonlinelibrary.com]

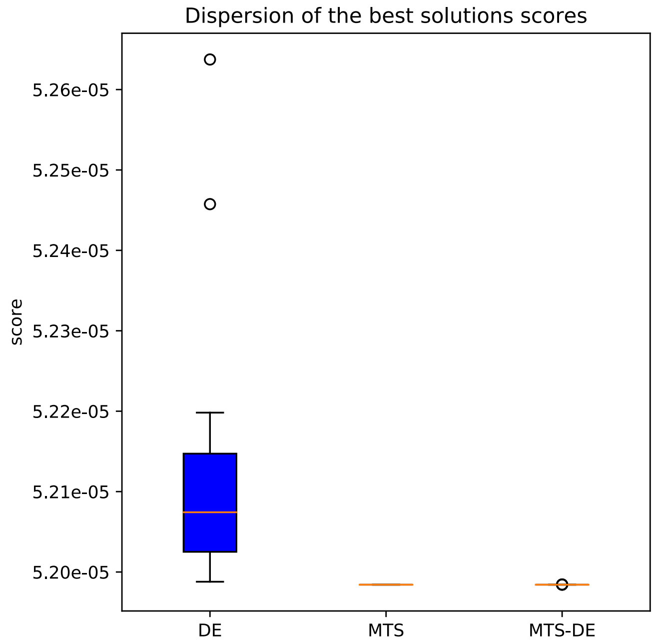

FIGURE 6 Comparison of the final results for Case 1 of each of the 25 executions for the three different search strategies (DE, MTS, and MTS-DE): box plot with the dispersion (mean value, interquartile range (IQR), 1.5 IQR, and outliers) of the best solutions found. DE, differential evolution; MTS, multiple trajectory search (out of the 25 executions) in terms of their fitness value [Colour figure can be viewed at wileyonlinelibrary.com]

TABLE 2 Average and standard deviation of the results for Case 1 obtained by each search strategy (out of the 25 executions), including the best overall $|\Delta A_{\text{top}}(t = 1)|$ found by each of them

Technique	Average	StdDev	Best
DE	5.2115E-05	1.4506E-07	5.1988E-05
MTS	5.1984E-05	0.0000E+00	5.1984E-05
MTS-DE	5.1984E-05	6.3246E-11	5.1984E-05

Abbreviations: DE, differential evolution; MTS, multiple trajectory search.

The differences in the performance of the three algorithms are shown in Figure 7. DE provides (with the selected parameters) a more explorative behavior than exploitative. The problem, as seen in the three plots, is clearly unimodal (with a single global optimum) (see Figure 7A). These problems are efficiently solved by local search strategies (such as MTS). Looking at how the different strategies explore the search space, the last iterations of MTS are always very close to the optimum value (see Figure 7B), whereas both of the other strategies show red dots in areas further away of this value. The hybrid approach MTS-DE makes use of the DE component to propose solutions in other areas of the search space (expecting to escape form local optima, which is not needed in this case) but manages to keep the right balance to use MTS intensive exploitation to find always the optimal value (see Figure 7C).

4.3.2 | Case 2

Problem description: We consider here a 2D T-shaped cell wall separating two plant cells. The inside of both cells is submitted to an arbitrary pressure (turgor pressure) taken here to be 500 kPa. The foot of the T is clamped at the bottom, whereas the horizontal bar ends are only free to move vertically. The cell wall is linear elastic with an arbitrary Young's modulus (taken here to be 1 Pa), whereas the region at the intersection between both walls has a different Young's modulus E. This problem reproduces in a simplified manner the plant cell wall shape study of the work of Kirchhelle et al.[64] Before meshing and loading, the T shape is modified into a Y shape, where the angle α made by the top walls is allowed to vary. The problem consists in minimizing the difference ΔP between the maximum and the minimum pressure in the resulting

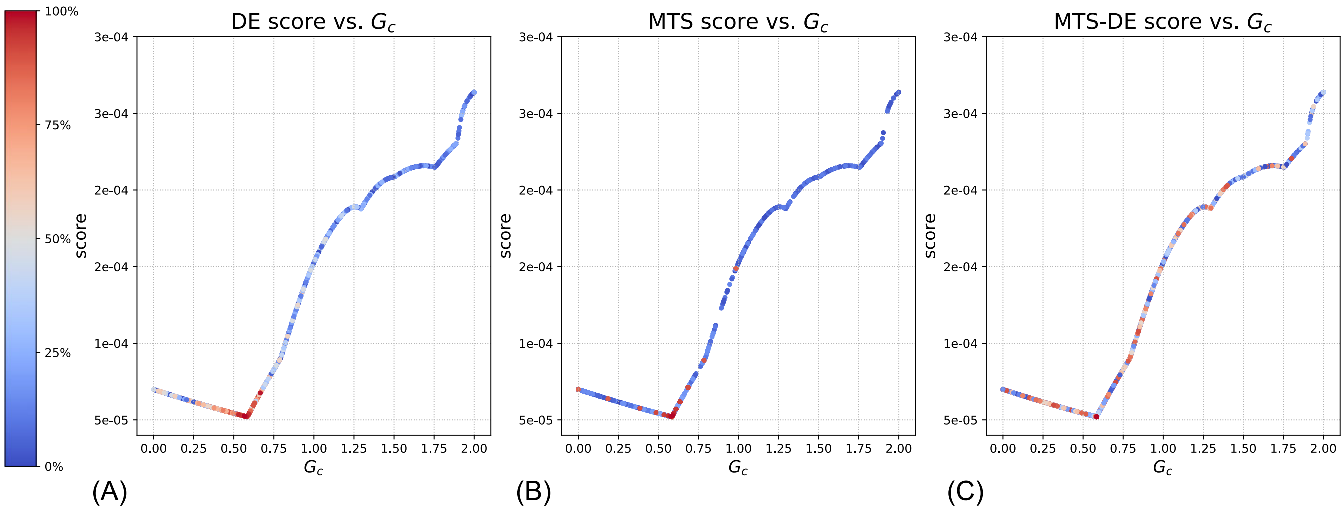

FIGURE 7 Comparison of the three different search strategies (DE, MTS, and MTS-DE) for Case 1: each of the three plots, A, B, and C, corresponds to the results of a given strategy showing the explored (by the 25 runs) values for G_c against the objective function; the color (from blue to red) indicates whether the candidate solution was generated at the beginning or the end of the search process. DE, differential evaluation; MTS, multiple trajectory search

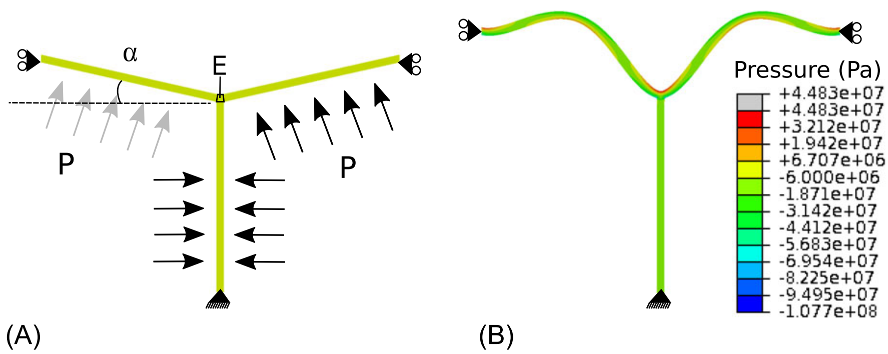

FIGURE 8 Case 2: A, The intersection between two outter plant cells with a top wall inclination of α, and a triple point Young's modulus E, submitted to turgor pressure P; B, The optimization problem consists in finding the values of α and E such that the pressure span within the wall is minimized

loaded structure by finding an optimum set of E and α. The problem is meshed automatically with Gmsh[60] and run on Abaqus[59] through a Python wrapper (see Figure 8).

Degrees of freedom: 40 666 nodes / 78 300 elements / 0 internal variables.

Input parameters: $\Gamma = \{E, \alpha\}$

- E: Young's modulus of the intersection region of the cell walls; $0.5 < E < 2$.
- α: Angle made by the top walls with respect to the horizontal; $0 < \alpha < \pi/6$.

Objective function (minimize): $f = \Delta P$. The average execution time per simulation is 4.1 s.

Best overall solution: $E = 0.7325$, $\alpha = 0.2095$ (found by MTS-DE).

A comparison of the final results for all three algorithms is shown in Figure 9A. The MTS method solutions are more spread than the ones of the other two strategies, with the DE results having the narrowest variation (one order of magnitude smaller out of the 25 repetitions). Figure 9B shows the solutions in the E vs. α space for the three strategies for the 25 runs. Two clusters of solutions are observed with the DE method only converging in one, whereas MTS finds more solutions in the other cluster. In all three cases, very similar solutions are found, although the best solution is obtained by MTS-DE (the smallest value of ΔP is reached) (see Figure 9A and Table 3). The statistical tests show that there is a difference among the three sets of results with a p-value of 0.0163 (Kruskal-Wallis multiple test). The pairwise comparisons estimate the difference between the DE results and MTS with a p-value of $0.0074 < 0.05$ (Wilcoxon test) and $0.052 \nless 0.05$ (Mann-Whitney test), and the statistical difference between DE and MTS-DE with a p-value of $0.029 < 0.05$

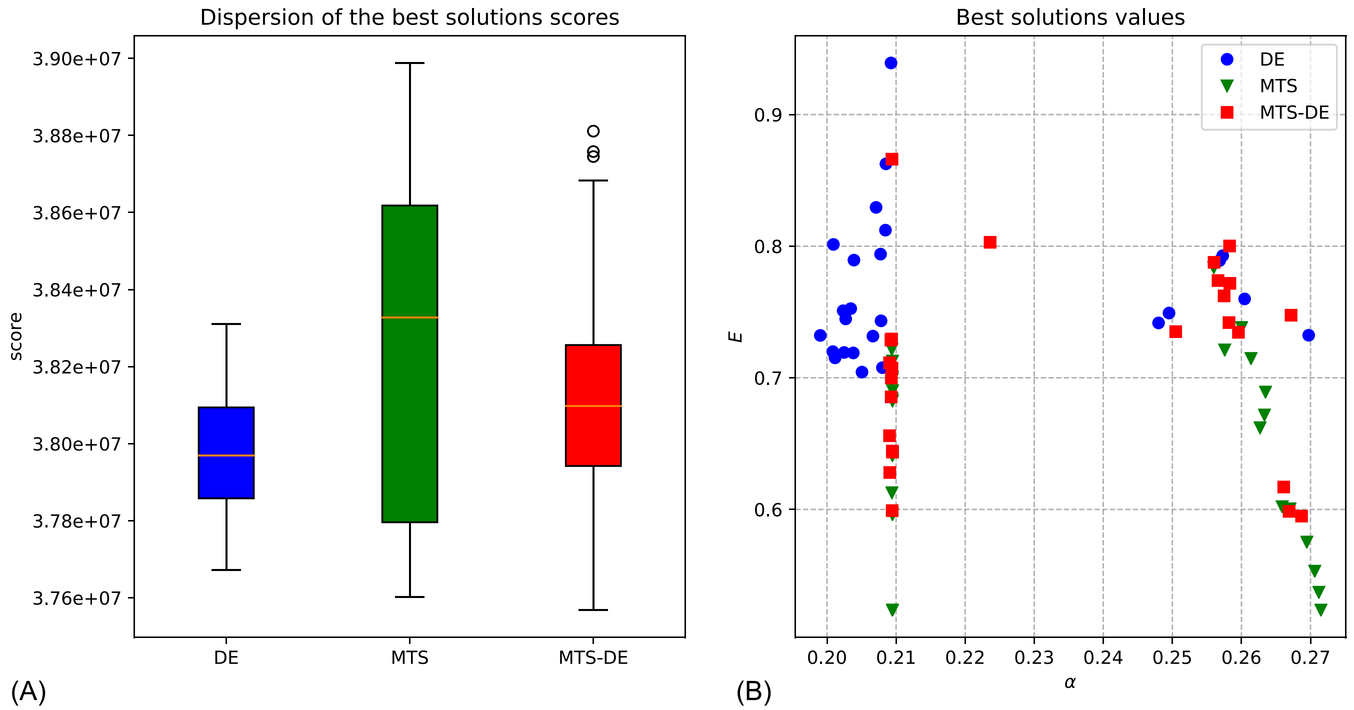

FIGURE 9 Comparison of the final results for Case 2 of each of the 25 executions for the three different search strategies (DE, MTS, and MTS-DE): A, Box plot with the dispersion (mean value, interquartile range (IQR), 1.5 IQR, and outliers) of the best solutions found (out of the 25 executions) in terms of their fitness value; B, Representation of E vs. α space of the best final points of the 25 solutions for each technique. DE, differential evaluation; MTS, multiple trajectory search [Colour figure can be viewed at wileyonlinelibrary.com]

TABLE 3 Average and standard deviation of the results for Case 2 obtained by each search strategy (out of the 25 executions), including the best overall ΔP found by each of them

Technique	Average	StdDev	Best
DE	3.774E+07	9.957E+04	3.762E+07
MTS	3.819E+07	5.034E+05	3.759E+07
MTS-DE	3.784E+07	2.098E+05	3.756E+07

Abbreviations: DE, differential evolution; MTS, multiple trajectory search.

(Wilcoxon test) and 0.102 $\nless$ 0.05 (Mann-Whitney test). Finally, the comparison between MTS and MTS-DE has a p-value of 0.032 $<$ 0.05 (Wilcoxon test) and also 0.102 $\nless$ 0.05 (Mann-Whitney test). Holm correction for the family-wise error rate has been used in all the cases.

The objective evolution for the three different strategies is shown in Figure 10. The fitness landscape for α (see Figures 10A to 10C) shows a large basin of attraction and continuity, independently of the E parameter whose secondary influence allows for fine-tuning in the last stage of convergence (see Figures 10D to 10F). Conversely, E exhibits a significant variation in fitness depending on the α value (see Figures 10G to 10I). It is not believed to be pure noise data, as the three algorithms converge roughly to the same values.

This problem is more difficult than the previous one in terms of optimization complexity because one of the parameters (α) highly biases the search process not only because of its major influence on the fitness function but also because of its large basin of attraction for a relative neutrality.[§] In this case, the local searcher (MTS) struggles to continue the search once it pivots from one variable to the other. On the contrary, the DE strategy is able to identify the different influences of each search variable and easily combine those components with good α values while refining E progressively. Once again, the MOS-based hybrid strategy combines efficiently DE and MTS to be competitive with the best of them in average results. In this case, it is even able to find the best overall solution.

[§]Neutrality: relatively moderate variation of the fitness function for a continuous range of values of one or more variables.

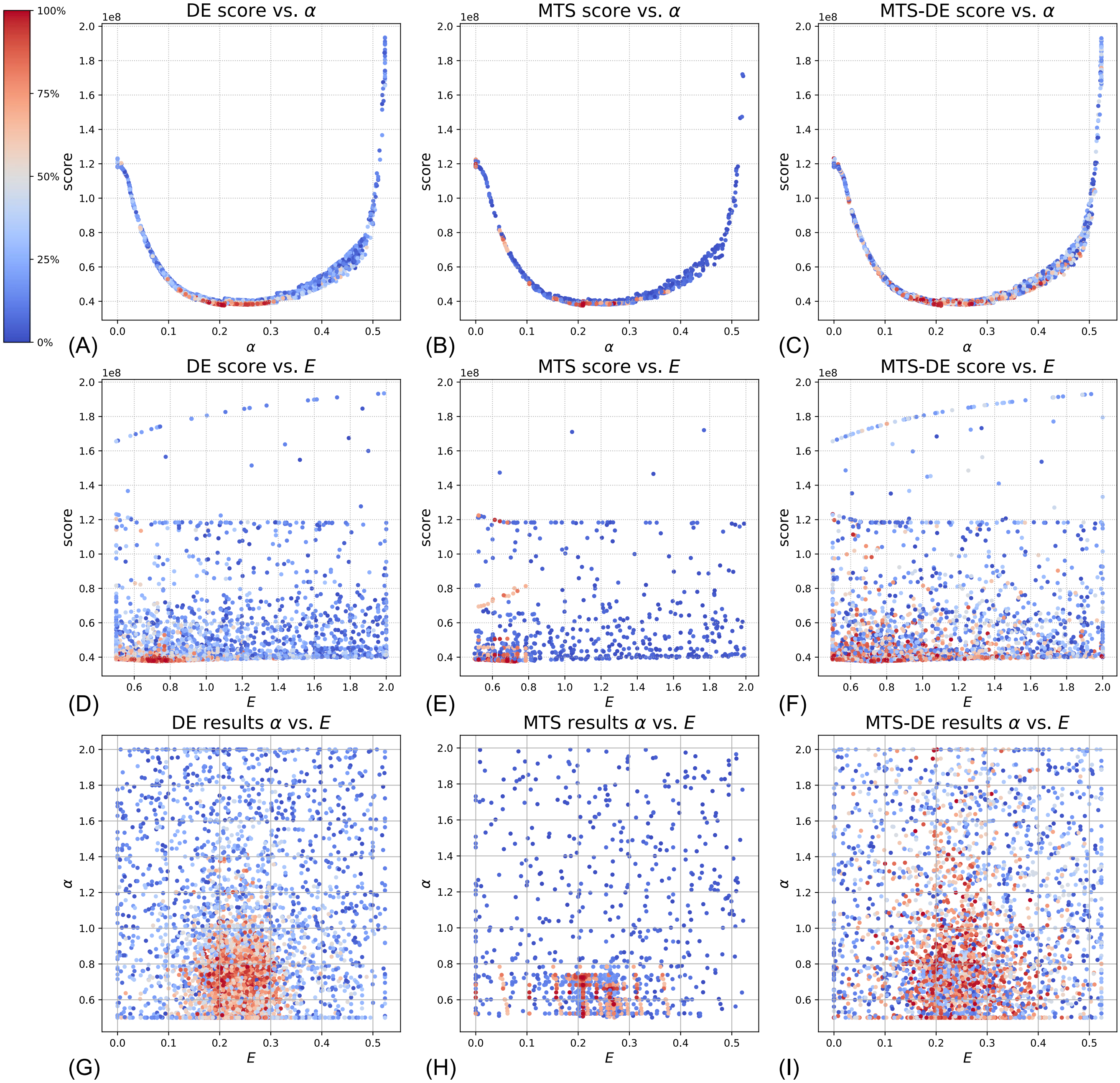

FIGURE 10 Comparison of the three different search strategies (DE, MTS, and MTS-DE) for Case 2: each of the three columns corresponds to the results of a given strategy showing the explored values for E and α against the objective function, as well as the final α vs. E plot. Each of the candidate solutions explored by the 25 runs is depicted. The color (from blue to red) indicates whether the candidate solution was generated at the beginning or the end of the search process. DE, differential evaluation; MTS, multiple trajectory search

4.3.3 | Case 3

Problem description: We consider here a polycrystalline prism meshed with 2592 cubic elements, each one representing one grain with its own crystalline orientation. A hexagonal close packed crystal plasticity model for Mg AZ31B with twinning is considered for two compressions in the rolling direction (RD) and normal direction (ND) (see the work of Fernández et al[65] for more details). In this reference, the large number of material properties (231 to be provided in the input file for each grain) is composed of known parameters (eg, crystalline orientation imposed by pole figure) but also unknown parameters that need to be calibrated. While Fernandez et al narrowed down the calibration to values not available in the literature, a significant effort was still needed to manually analyze their sensibility with respect to the final calibration targets (the stress-strain curves of the RD and ND compressions). We focus here on a subset of these characterized by a lack of experimental validation and/or a high known influence and aim at recalibrating the curve automatically by fitting those parameters. The problem is run on Abaqus.[59]

Degrees of freedom: 3211 nodes / 2592 elements / 121 internal variables.

Input parameters: $\Gamma = \{s_{0;\text{basal}}, s_{0;pyr<c+a>}, s_{0;pyr<a>}, s_{0;\text{prism}}, h_{0;sl}, h_{0;tw}, h_{0;tw-sl}, q_{sl}\}$

- $s_{0;\text{basal}}$: Critical resolved shear stress of the basal slip systems; $0 < s_{0;\text{basal}} < 200$ MPa;
- $s_{0;pyr<c+a>}$: Critical resolved shear stress of the pyramidal $<c+a>$ slip systems; $0 < s_{0;pyr<c+a>} < 200$ MPa;
- $s_{0;pyr<a>}$: Critical resolved shear stress of the pyramidal$< a >$ slip systems; $0 < s_{0;pyr<a>} < 200$ MPa;
- $s_{0;\text{prism}}$: Critical resolved shear stress of the prismatic systems; $0 < s_{0;pyr<a>} < 400$ MPa;
- $h_{0;sl}$: Reference self-hardening parameter of the slip systems; $0 < h_{0;sl} < 1500$ MPa;
- $h_{0;tw}$: Reference self-hardening parameter of the twin systems; $0 < h_{0;tw} < 1500$ MPa;
- $h_{0;tw-sl}$: Reference hardening parameter of the twin systems on the slip systems; $0 < h_{0;tw-sl} < 1500$ MPa;
- q_{sl}: Ratio of reference cross-hardening to reference self-hardening parameter; $0 < q_{sl} < 10$.

Objective function (minimize): $f = \text{DTW}_{RD} + \text{DTW}_{ND}$, where DTW_{RD} and DTW_{ND} are functions summing the error in the RD and ND cases, respectively, between the experimental and numerical stress-strain curves following a dynamic time warping algorithm.[66] The average execution time per simulation is 2 h (× 12 CPUs).

Best overall solution (manual calibration provided in parentheses) (see Figure 11): $s_{0;\text{basal}} = 177.4520$ MPa (9 MPa), $s_{0;pyr<c+a>} = 107.5783$ MPa (115 MPa), $s_{0;pyr<a>} = 153.4912$ MPa (115 MPa), $s_{0;\text{prism}} = 258.6643$ MPa (80 MPa), $h_{0;sl} = 623.3875$ MPa (600 MPa), $h_{0;tw} = 157.6780$ MPa (80 MPa), $h_{0;tw-sl} = 1475.0977$ MPa (1200 MPa), and $q_{sl} = 10$ (2).

Because this problem is purposefully computationally expensive, we performed a single run of 505 fitness evaluations (1 month wall-clock time in total) with the most promising of the three techniques: the MOS-based hybrid algorithm using both DE and MTS. Note that the fitness is expected to exhibit both a continuous behavior when one parameter change does not trigger a jump to another deformation mechanism (eg, from one slip system to another) and a noncontinuous one when a parameter triggers a change of deformation mechanism.

Figure 12 shows the process of improvement for the best found solution vs. the number of fitness evaluations of the search. The exploration cycles tend to cover a narrower area as the algorithm progresses (DE also provides exploitation but at a much slower pace than MTS). In the enclosed graph of this figure, the alternance between the different exploration/exploitation phases (DE and MTS) can be seen. The fitness value vs. the number of fitness evaluations (and

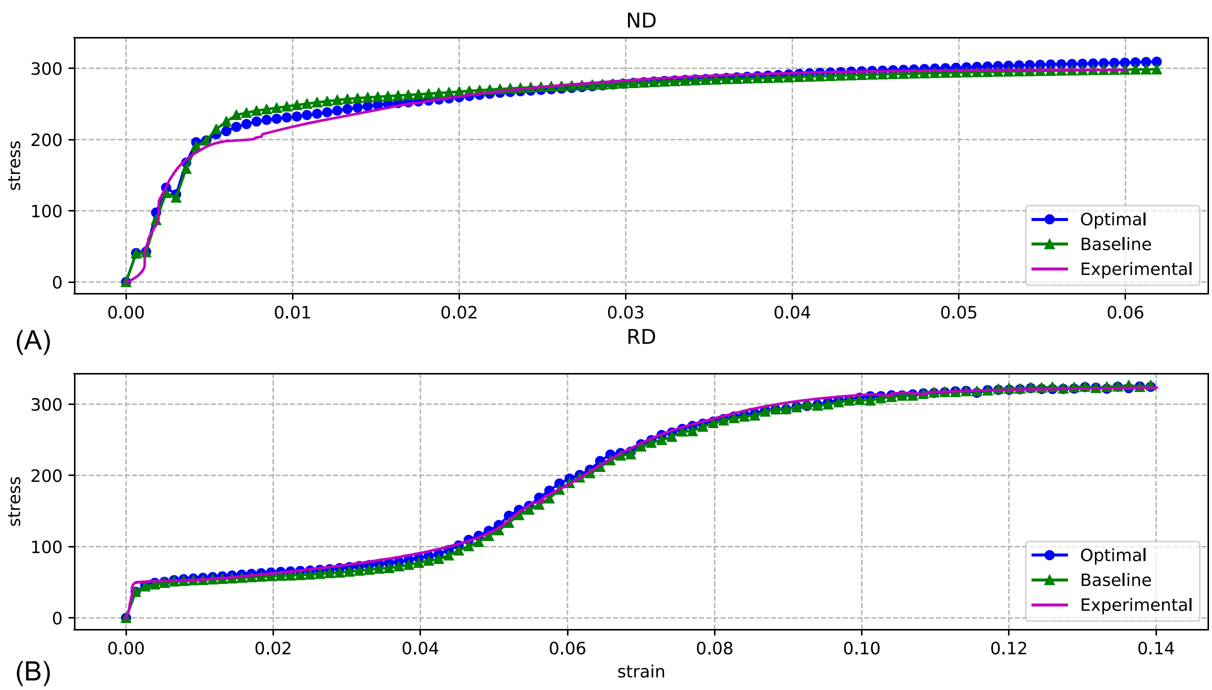

FIGURE 11 Case 3: comparative analysis of the reference (baseline) manually calibrated parameter combination and the best solution found by SoftFEM related to the original experimental data for both the normal direction (ND) and rolling direction (RD) cases [Colour figure can be viewed at wileyonlinelibrary.com]

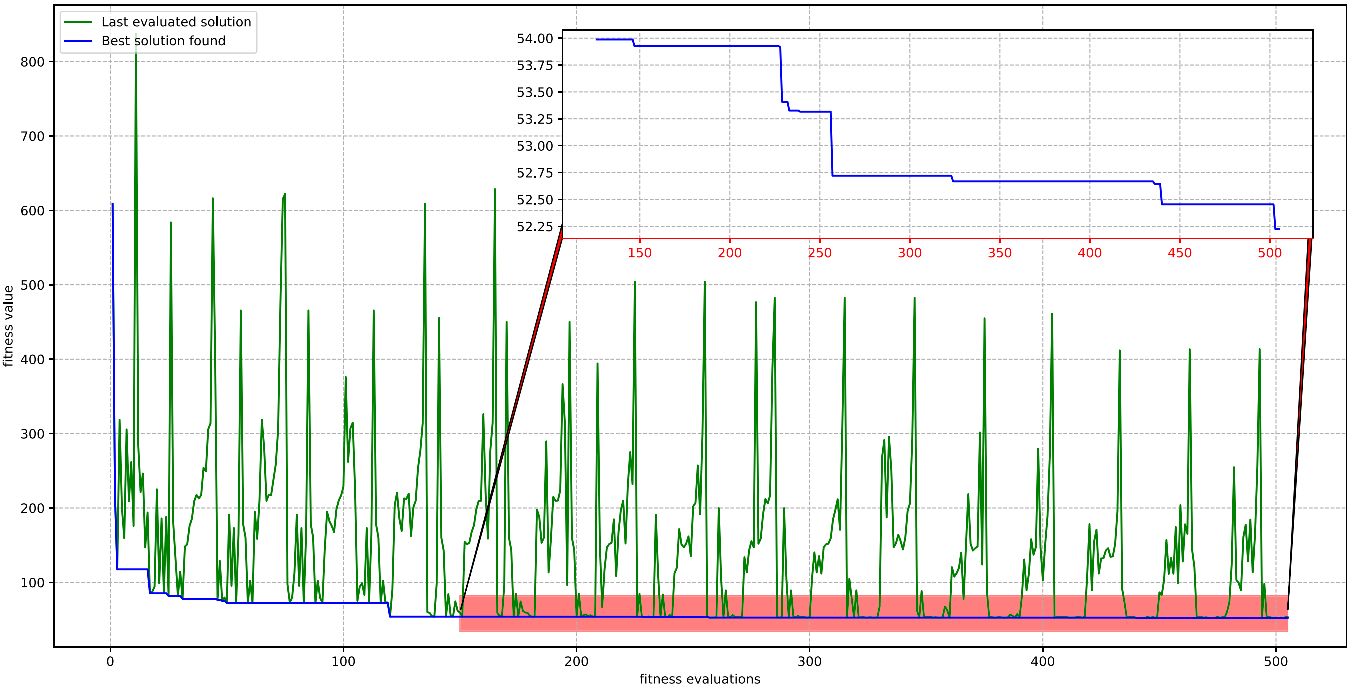

FIGURE 12 Evolution of the search process for Case 3: the main graph shows the evolution of the best overall solution (blue) found as candidate solutions are explored (number of fitness evaluations); it also includes the value of the last solution evaluated by the fitness function (green). The enclosed graph shows the best solution after the first 125 fitness evaluations

not the number of generations or iterations) was plotted here to show this evolution process independently from the population size. This representation is commonly used in the heuristic optimization literature to compare algorithms with different or dynamic population sizes (such as here, where the DE component uses a population and MTS uses a single solution).

With heuristic optimization techniques, a global optimum solution cannot be ensured not only due to the stochastic nature of the search (if it is the case) but mainly because heuristic methods are, in essence, nonexhaustive nonanalytical search strategies and only a partial space of the potential solutions is covered. However, different stopping criteria may be imposed. The most usual one is the stagnation of progress when no actual improvement is achieved in a number of consecutive iterations. In this case, this convergence condition was not met at the moment we reached the maximum number of evaluations indicating that further improvement might still be possible. Nonetheless, the solution found by SoftFEM improves the manually calibrated solution known for this problem taken as our baseline for comparison (see Figure 11). The objective function value for the initial (baseline) solution was 65.1711, and the best SoftFEM solution is 52.2251.

This problem is particularly interesting from the optimization perspective because the epistatic[¶] nature of the problem. Figure 13 shows the comparative graphs of the objective function for each one of the eight parameters. There is no parameter that clearly shows that all the configurations with a given value for that parameter are always better than configurations with other values for the same parameter. Epistatic problems cannot be solved by decomposing the problem by parameters and trying to optimize them independently each of them at a time. The best performing search strategies require a global optimization playing with multiple input parameters at once.

The final parameter values are in general in agreement with their manually calibrated counterparts. The final value of q_{sl} lies on the boundary of the domain constraint, but as can be seen in Figure 13H, its influence is not fully identified, and other values could potentially be considered. Similarly, all but one critical resolved shear stresses are relatively close to what the manually calibrated (and experimental) values are, leaving the basal critical resolved shear stress unphysically large. Again, Figure 13A seems to indicate that, while this value leads to the best solution, lower (< 50 MPa) values could

[¶]Epistasis: refers to the property (of the objective function of the problem) that gives more importance to the combination of the values of two or more input parameters than the additive value of these parameters independently.

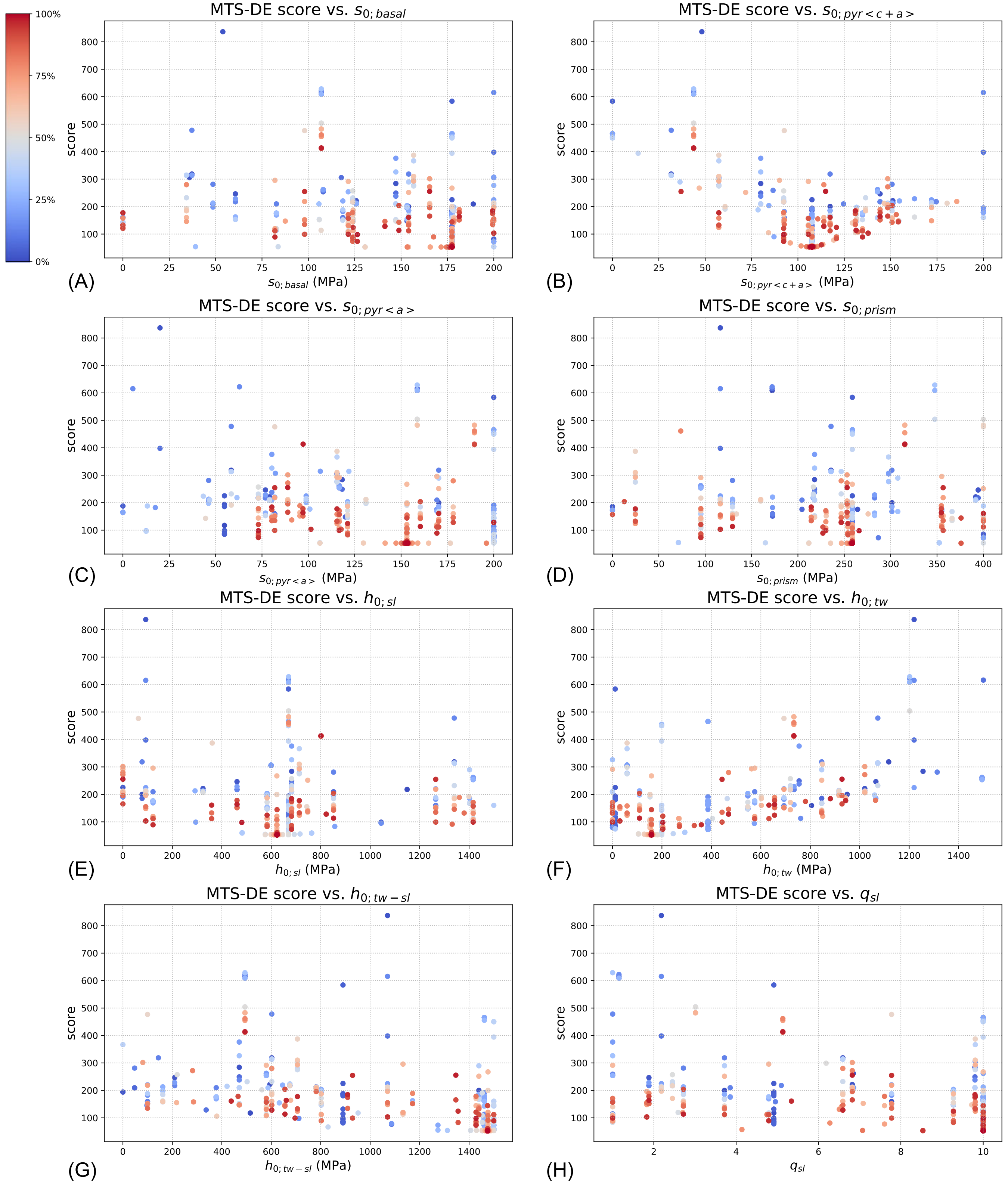

FIGURE 13 Objective function values for Case 3 compared vs. input parameter (1 through 8): each point represents a single solution of the execution; the color (from blue to red) indicates whether the candidate solution was generated at the beginning or the end of the search process. DE, differential evaluation; MTS, multiple trajectory search

also lead to relatively small values of the fitness. For instance, the best solution when imposing $s_{0;\text{basal}} < 50$ MPa was found to be 39.14 MPa (see Figure 14 for the comparison against the overall best solution).

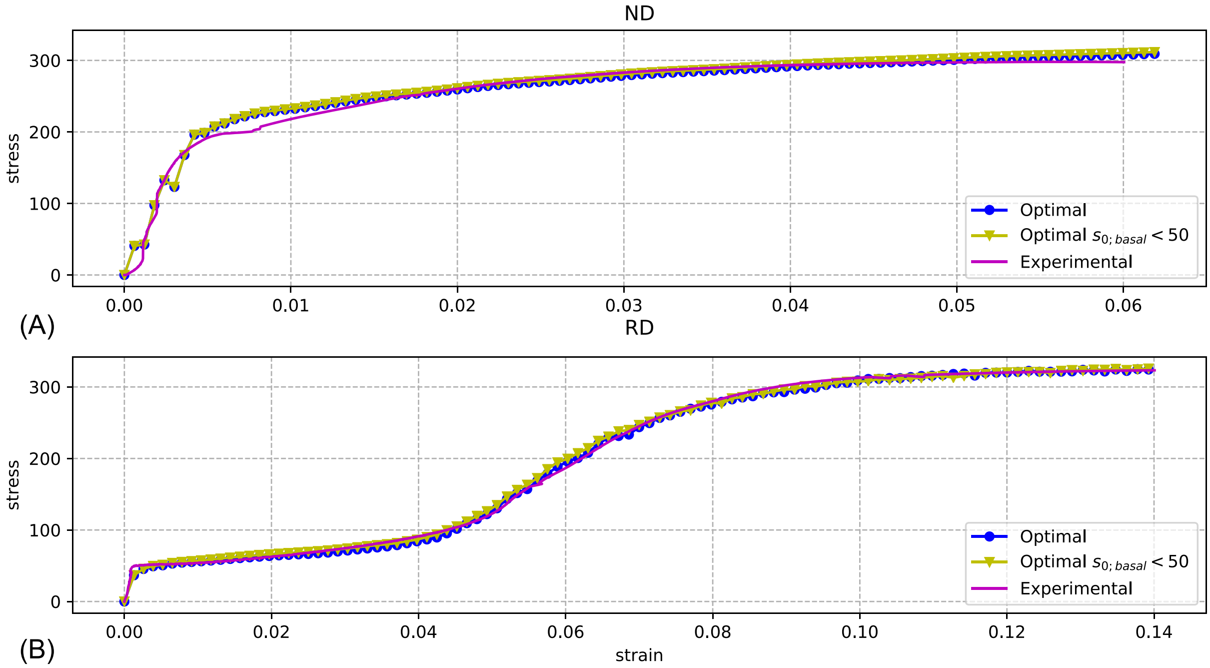

FIGURE 14 Stress-strain curves for the overall best solution of Case 3 both normal direction (ND) and rolling direction (RD), and the best solution under the additional constraint $s_{0;basal} < 50$ MPa [Colour figure can be viewed at wileyonlinelibrary.com]

The epistatic nature of this observation effectively provides a novel tool for the identification of the need of additional experimental data for some of the model parameters. In this case, one can conclude that the RD and ND compression stress-strain curves are not enough to fully determine $s_{0;\text{basal}}$, $s_{0;pyr<a>}$, $s_{0;\text{prism}}$, and q_{sl}, as their influence on the fitness remains globally "flat" (see Figure 13). The other parameters, however, all exhibit a tendency (clearer for some than others) toward one value in particular.

5 | CONCLUSION

In this work, SoftFEM, a new framework wrapping FEM with heuristic optimization, was formalized. This effort addresses the current gap in the literature for an integrated methodology that seamlessly combines two numerical methods, a partial differential equation solver (FEM) and the body of several stochastic heuristic optimization methods from soft computing, to provide a generalized optimization framework for engineering and physical problems. The resulting methodology is based on a common formalization that ensures the use of a shared representation for the solution and evaluation spaces and defines the problem in a common language for both methods.

The MOS framework adopted here is considered one of the state-of-the-art methods in large-scale global optimization and the winner of multiple competitions on this topic held at the IEEE CEC. This framework allows to combine multiple search strategies and run them simultaneously, adjusting the participation of each of them in the overall search process dynamically. This characteristic makes it, in our opinion, a perfect candidate to be the optimization technique of choice for the SoftFEM method, as it leverages the effort of choosing a particular optimization method from the users. In many cases,[23,62,63,67-72] the framework is able to exploit the existing synergies among the different methods under consideration and obtain better results than any of the individual algorithms. In other cases,[63,73,74] those synergies do not exist, but still the framework is able to identify the best performing method and adjust the participation ratios accordingly, with a minimum overhead, thus simplifying the overall process. For these reasons, the MOS framework has been selected as the optimization platform in SoftFEM.

Three cases from different disciplines with different convergence properties were used. The first two allowed for a direct comparison of the performances of DE, MTS, and MOS combining both, whereas the third case leveraged MOS for a complex multidimensional problem. MOS was found to offer the best versatility, similar to or outperforming both DE and MTS in the first two cases. In particular, MOS avoids the need to know a priori the continuity and convergence properties of the problem at hand and is able to automatically optimize a problem while choosing the most appropriate method (local search or population-based). Additionally, the approach provides a novel tool to identify parameters that require additional experimental data for full calibration.

Finally, it must be emphasized that SoftFEM heavily relies on the adequate identification of the fitness metric. Future work should focus on the use of complementary learning methods to model the behavior of the fitness function without the need to run a simulation. This can be done by providing surrogate functions that approximate the results of the actual finite element calculation for a number of cases, allowing the exploration (although in an approximate way) of more areas of the search space and only performing the actual finite element simulations for those promising results. This surrogate-based approach may allow to overcome some of the limitations in terms of the computational cost for SoftFEM.

ACKNOWLEDGEMENTS

J.-M.P. and A.L. acknowledge funding from the Spanish Ministry of Science and Innovation (TIN2017-83132-C2-2-R) and Universidad Politécnica de Madrid (PINV-18-XEOGHQ-19-4QTEBP).

ORCID

Antonio LaTorre https://orcid.org/0000-0002-8718-5735
Antoine Jérusalem https://orcid.org/0000-0001-5026-8038

REFERENCES

1. Bathe K-J. *Finite Element Procedures*. New Delhi, India: PHI Learning; 1996.
2. Moës N, Dolbow J, Belytschko T. A finite element method for crack growth without remeshing. *Int J Numer Methods Eng*. 1999;46(1):131-150. https://doi.org/10.1002/(SICI)1097-0207(19990910)46:1<131::AID-NME726>3.0.CO;2-J
3. Noels L, Radovitzky R. An explicit discontinuous Galerkin method for non-linear solid dynamics: formulation, parallel implementation and scalability properties. *Int J Numer Methods Eng*. 2008;74(9):1393-1420. arXiv:1010.1724. https://doi.org/10.1002/nme.2213
4. Stefanou G. The stochastic finite element method: past, present and future. *Comput Methods Appl Mech Eng*. 2009;198(9):1031-1051. https://doi.org/10.1016/j.cma.2008.11.007
5. Haukaas T, Gardoni P. Model uncertainty in finite-element analysis: Bayesian finite elements. *J Eng Mech*. 2011;137(8):519-526. https://doi.org/10.1061/(ASCE)EM.1943-7889.0000253
6. Guilleminot J, Noshadravan A, Soize C, Ghanem RG. A probabilistic model for bounded elasticity tensor random fields with application to polycrystalline microstructures. *Comput Methods Appl Mech Eng*. 2011;200(17):1637-1648. https://doi.org/10.1016/j.cma.2011.01.016
7. Soize C. Maximum entropy approach for modeling random uncertainties in transient elastodynamics. *J Acoust Soc Am*. 2001;109(5):1979-1996. https://doi.org/10.1121/1.1360716
8. Hoang T-V, Wu L, Paquay S, Golinval J-C, Arnst M, Noels L. A computational stochastic multiscale methodology for MEMS structures involving adhesive contact. *Tribol Int*. 2017;110:401-425. https://doi.org/10.1016/j.triboint.2016.10.007
9. Sigmund O, Maute K. Topology optimization approaches. *Struct Multidiscip Optim*. 2013;48(6):1031-1055. https://doi.org/10.1007/s00158-013-0978-6
10. Jebeli SM, Panahi MS. An evolutionary approach for simultaneous optimization of material property distribution and topology of FG structures. *Eng Comput*. 2015;32(2):234-257. https://doi.org/10.1108/EC-07-2013-0188
11. Taheri AH, Suresh K. An isogeometric approach to topology optimization of multi-material and functionally graded structures. *Int J Numer Methods Eng*. 2017;109(5):668-696. https://doi.org/10.1002/nme.5303
12. Vanderplaats GN. *Numerical Optimization Techniques for Engineering Design: With Applications*. Vol. 1. New York, NY: McGraw-Hill; 1984.
13. Zadeh LA. Fuzzy logic, neural networks, and soft computing. *Commun ACM*. 1994;37(3):77-85.
14. Mitchell M. *An Introduction to Genetic Algorithms*. Cambridge, MA: MIT Press; 1998.
15. Mühlenbein H. The equation for response to selection and its use for prediction. *Evol Comput*. 1997;5(3):303-346.
16. Storm R, Price KV. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *J Glob Optim*. 1997;11(4):341-359.
17. Kennedy J, Eberhart R. Particle swarm optimization, in neural networks. In: Proceedings of the IEEE International Conference on Neural Networks; 1995; Perth, Australia.
18. Dorigo M, Di Caro G. Ant colony optimization: a new meta-heuristic. In: Proceedings of the 1999 Congress on Evolutionary Computation (CEC99), vol. 2; 1999; Washington, DC.
19. Tseng LY, Chen C. Multiple trajectory search for large scale global optimization. Paper presented at: 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence); 2008; Hong Kong.
20. Glover FW, Laguna M. *Tabu Search*. Vol. 1. Boston, MA: Springer Science & Business Media; 1998.
21. Mladenović N, Hansen P. Variable neighborhood search. *Comput Oper Res*. 1997;24(11):1097-1100.

22. Talbi E-G. A taxonomy of hybrid metaheuristics. *J Heuristics*. 2002;8(5):541-564.
23. LaTorre A, Muelas S, Peña JM. A comprehensive comparison of large scale global optimizers. *Inform Sci*. 2014;316:517-549. https://doi.org/10.1016/j.ins.2014.09.031
24. Filho JLR, Treleaven PC, Alippi C. Genetic-algorithm programming environments. *Computer*. 1994;27(6):28-43.
25. Annicchiarico W, Cerrolaza M. A structural optimization approach and software based on genetic algorithms and finite elements. *Eng Optim*. 1999;32(1):1-31.
26. Shayanfar M, Abbasnia R, Khodam A. Development of a GA-based method for reliability-based optimization of structures with discrete and continuous design variables using OpenSees and TCL. *Finite Elem Anal Des*. 2014;90:61-73.
27. Corriveau G, Guilbault R, Tahan A. Genetic algorithms and finite element coupling for mechanical optimization. *Adv Eng Softw*. 2010;41(3):422-426.
28. Smolka J. Genetic algorithm shape optimisation of a natural air circulation heating oven based on an experimentally validated 3-D CFD model. *Int J Therm Sci*. 2013;71:128-139.
29. López D, Angulo C, Fernández de Bustos I, García V. Framework for the shape optimization of aerodynamic profiles using genetic algorithms. *Math Probl Eng*. 2013;2013.
30. Wang D, Wu Z, Fei Y, Zhang W. Structural design employing a sequential approximation optimization approach. *Comput Struct*. 2014;134:75-87.
31. Schmid F, Hirschen K, Meynen S, Schäfer M. An enhanced approach for shape optimization using an adaptive algorithm. *Finite Elem Anal Des*. 2005;41(5):521-543.
32. Gagné C, Parizeau M. Genericity in evolutionary computation software tools: principles and case-study. *Int J Artif Intell Tools*. 2006;15(2):173-194.
33. Ho-Huu V, Do-Thi TD, Dang-Trung H, Vo-Duy T, Nguyen-Thoi T. Optimization of laminated composite plates for maximizing buckling load using improved differential evolution and smoothed finite element method. *Compos Struct*. 2016;146:132-147.
34. Temür R, Türkan YS, Toklu YC. Geometrically nonlinear analysis of trusses using particle swarm optimization. In: *Recent Advances in Swarm Intelligence and Evolutionary Computation*. Cham, Switzerland: Springer; 2015:283-300.
35. Lee KS, Geem ZW. A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. *Comput Methods Appl Mech Eng*. 2005;194(36-38):3902-3933.
36. Valdez SI, Faurrieta N, Marroquín JL, Botello S. Automated shape optimization using a multigrid method and estimation of distribution algorithms. Paper presented at: COMPLAS - XIII International Conference on Computational Plasticity: Fundamentals and Applications (COMPLAS 2015); 2015; Barcelona, Spain.
37. Sonmez FO. Shape optimization of 2D structures using simulated annealing. *Comput Methods Appl Mech Eng*. 2007;196(35-36):3279-3299.
38. Kim WJ, O'Brien JD. Optimization of a two-dimensional photonic-crystal waveguide branch by simulated annealing and the finite-element method. *JOSA B*. 2004;21(2):289-295.
39. Corz A, Gomez-Ruiz JA, Pelaez JI, Tenorio E, Veintimilla J. Design and optimization of symmetric laminated composites using a variable neighbourhood search-based model. *Eng Optim*. 2012;44(4):505-520.
40. Pai N, Kaw A, Weng M. Optimization of laminate stacking sequence for failure load maximization using tabu search. *Compos B: Eng*. 2003;34(4):405-413.
41. Yang X-S. *Engineering Optimization: An Introduction With Metaheuristic Applications*. Hoboken, NJ: John Wiley & Sons; 2010.
42. Levin RI, Lieven NAJ. Dynamic finite element model updating using simulated annealing and genetic algorithms. *Mech Syst Signal Process*. 1998;12(1):91-120.
43. Vicini A, Quagliarella D. Airfoil and wing design through hybrid optimization strategies. *AIAA J*. 1999;37(5):634-641.
44. Rao ARM, Arvind N. Optimal stacking sequence design of laminate composite structures using tabu embedded simulated annealing. *Struct Eng Mech*. 2007;25(2):239-268.
45. Pholdee N, Bureerat S, Yıldız A. Hybrid real-code population-based incremental learning and differential evolution for many-objective optimisation of an automotive floor-frame. *Int J Veh Des*. 2017;73(1-3):20-53.
46. Pelaez JI, Gomez-Ruiz JA, Veintimilla J, Vaccaro G, Witt Patricia. Memetic computing applied to the design of composite materials and structures. *Math Probl Eng*. 2017;2017.
47. Coello Coello CA. *An Empirical Study of Evolutionary Techniques for Multiobjective Optimization in Engineering Design* [PhD thesis]. New Orleans, LA: Tulane University; 1996.
48. Andersson J. *A Survey of Multiobjective Optimization in Engineering Design*. Technical Report. Linköping, Sweden: Department of Mechanical Engineering, Linköping University; 2000.
49. Fleming PJ, Purshouse RC, Lygoe RJ. Many-objective optimization: an engineering design perspective. In: *Evolutionary Multi-Criterion Optimization*. Berlin, Germany: Springer; 2005:14-32.
50. Courteille E, Mortier F, Leotoing L, Ragneau E. *Multi-Objective Robust Design Optimization of an Engine Mounting System*. Technical Report. Warrendale, PA: SAE International; 2005.
51. Honda S, Igarashi T, Narita Y. Multi-objective optimization of curvilinear fiber shapes for laminated composite plates by using NSGA-II. *Compos B: Eng*. 2013;45(1):1071-1078.
52. Govindarajalu J, Karuppan S, Manoharan T. Tolerance design of mechanical assembly using NSGA II and finite element analysis. *J Mech Sci Technol*. 2012;26(10):3261-3268.

53. Galvan E, Malak RJ, Hartl DJ, Baur JW. Performance assessment of a multi-objective parametric optimization algorithm with application to a multi-physical engineering system. *Struct Multidiscip Optim*. 2018;58(2):489-509.
54. Noilublao N, Bureerat S. Simultaneous topology, shape and sizing optimisation of a three-dimensional slender truss tower using multiobjective evolutionary algorithms. *Comp Struct*. 2011;89(23-24):2531-2538.
55. Brigham JC, Aquino W. Surrogate-model accelerated random search algorithm for global optimization with applications to inverse material identification. *Comput Methods Appl Mech Eng*. 2007;196(45-48):4561-4576.
56. Park HS, Dang XP. Structural optimization based on CAD-CAE integration and metamodeling techniques. *Comput Aided Des*. 2010;42(10):889-902.
57. Duddeck F. Multidisciplinary optimization of car bodies. *Struct Multidiscip Optim*. 2008;35(4):375-389.
58. LaTorre A. *A Framework for Hybrid Dynamic Evolutionary Algorithms: Multiple Offspring Sampling (MOS)* [PhD thesis]. Madrid, Spain: Universidad Politecnica de Madrid; 2009.
59. Dassault Systèmes. Abaqus 6. 2014.
60. Gmsh: a 3-D finite element mesh generator with built-in pre- and post-processing facilities. *Int J Numer Methods Eng*. 2009;79(11):1309-1331.
61. Garcıa-Grajales JA, Jérusalem A, Goriely A. Continuum mechanical modeling of axonal growth. *Comput Methods Appl Mech Eng*. 2017;314:147-163. Part of special issue: Special Issue on Biological Systems Dedicated to William S. Klug.
62. LaTorre A, Muelas S, Peña JM. A MOS-based dynamic memetic differential evolution algorithm for continuous optimization: a scalability test. *Soft Comput Fusion Found Methodol Appl*. 2010;15(11):2187-2199. https://doi.org/10.1007/s00500-010-0646-3
63. LaTorre A, Muelas S, Peña JM. Large scale global optimization: experimental results with mos-based hybrid algorithms. Paper presented at: 2013 IEEE Congress on Evolutionary Computation (CEC 2013); 2013; Cancun, Mexico.
64. Kirchhelle C, Chow C-M, Foucart C, et al. The specification of geometric edges by a plant Rab GTPase is an essential cell-patterning principle during organogenesis in Arabidopsis. *Dev Cell*. 2016;36(4):386-400.
65. Fernández A, Prado MTP, Wei Y, Jérusalem A. Continuum modeling of the response of a MG alloy AZ31 rolled sheet during uniaxial deformation. *Int J Plast*. 2011;27(11):1739-1757.
66. Berndt DJ, Clifford J. Using dynamic time warping to find patterns in time series. In: AAAIWS'94 Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining (KDD Workshop), vol. 10; 1994; Seattle, WA.
67. Mahdavi S, Shiri ME, Rahnamayan S. Metaheuristics in large-scale global continues optimization: a survey. *Inform Sci*. 2015;295:407-428.
68. Sato Y, Oku Y, Fukuda M. Applying GA with local search by taking hamming distances into consideration to credit erasure processing problems. In: Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation; 2014; Vancouver, Canada.
69. LaTorre A, Muelas S, Peña JM. Multiple offspring sampling in large scale global optimization. In: 2012 IEEE Congress on Evolutionary Computation (CEC 2012); 2012; Brisbane, Australia.
70. Xiong N, Molina D, Ortiz ML, Herrera F. A walk into metaheuristics for engineering optimization: principles, methods and recent trends. *Int J Comput Intell Syst*. 2015;8(4):606-636.
71. Liu C, Li B. Memetic algorithm with adaptive local search depth for large scale global optimization. Paper presented at: 2014 IEEE Congress on Evolutionary Computation (CEC); 2014; Beijing, China.
72. LaTorre A, Muelas S, Peña J-M. Benchmarking a hybrid DE-RHC algorithm on real world problems. Paper presented at: 2011 IEEE Congress of Evolutionary Computation (CEC); 2011; New Orleans, LA.
73. dos Santos EB, Ebecken NFF, Hruschka ER. Learning Bayesian network structures using multiple offspring sampling. Paper presented at: 2011 11th International Conference on Intelligent Systems Design and Applications; 2011; Cordoba, Spain.
74. LaTorre A, Muelas S, Peña JM. Benchmarking a MOS-based algorithm on the BBOB-2010 noiseless function testbed. Paper presented at: 12th Genetic and Evolutionary Computation Conference (GECCO 2010); 2010; Portland, OR. https://doi.org/10.1145/1830761.1830785

How to cite this article: Peña JM, LaTorre A, Jérusalem A. SoftFEM: The Soft Finite Element Method. *Int J Numer Methods Eng*. 2019;118:606–630. https://doi.org/10.1002/nme.6029