RAIRO-Oper. Res. 57 (2023) 2819–2832
https://doi.org/10.1051/ro/2023157

RAIRO Operations Research
www.rairo-ro.org

A FUZZY PARTICLE SWARM OPTIMIZATION METHOD WITH APPLICATION TO SHAPE DESIGN PROBLEM

Youness El Yazidi[1,*] and Abdellatif Ellabib[2]

Abstract. In this study, we focus on a specific class of bilateral free boundaries problems. We approach this problem by formulating it as a shape optimization problem using a defined cost functional. The existence of an optimal solution for the optimization problem is proved. To tackle this problem, we propose an iterative approach that combines the particle swarm optimization and fuzzy logic methods. Additionally, we employ the finite element method as a discretization technique for the state equation. To validate our approaches, we investigate various types of domains. Furthermore, we compare the performance of these approaches in two scenarios: one with exact measurements used for identification and another with noisy measurements.

Mathematics Subject Classification. 49Q10, 65L60, 90C59, 94D05.

Received December 1, 2022. Accepted September 26, 2023.

1. Introduction

Bilateral free boundaries are a recurring phenomenon in engineering scenarios, involving the modeling and analysis of situations where two distinct boundaries can autonomously evolve within a defined domain. This concept finds applications in diverse engineering fields, including heat transfer and material processing [5, 8, 9, 16, 18, 21].

To address such inverse problems, the use of shape optimization transforms the challenge from an ill-posed context into a constrained optimization setting. Following this, the development of robust numerical strategies becomes crucial for approximating solutions. Diverse approaches exist, including gradient, heuristic, and hybrid based methods. Among gradient based approaches, various adaptations have been proposed. For instance, Huang *et al.* [16] integrated the boundary element method with the conjugate gradient technique to identify two interfaces in a heat transfer model. The application of the Newton algorithm was introduced in [19]. In the work by [8], the quasi-Newton method was employed to address bilateral free boundaries emerging in semiconductor contexts.

A range of heuristic techniques have been proposed to address various instances of shape optimization problems. For instance, Wang *et al.* [27] introduced a multi-objective particle swarm optimization algorithm for

Keywords. Finite element method, fuzzy logic, inverse problem, particle swarm optimization, shape optimization.

[1] LaR2a, Department of Mathematics, Faculty of Science, Abdelmalek Essaâdi University, B.P. 2121 M'Hannech II, 93030 Tetouan, Morocco.

[2] Faculty of Science and Technology, Cady Ayyad University, B.P. 549 Avenue Abdelkrim El Khattabi, 40000 Marrakesh, Morocco.

[*]Corresponding author: younesselyazidi@hotmail.com

© The authors. Published by EDP Sciences, ROADEF, SMAI 2023

This is an Open Access article distributed under the terms of the Creative Commons Attribution License (https://creativecommons.org/licenses/by/4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

bi-directional functionally graded plates. In [22], a modified Particle Swarm Optimization (PSO) method was presented to tackle a shape optimization issue within an aerodynamics context. Ge *et al.* [11] proposed a multi-objective genetic algorithm to address shape design problem in tube bank modeling. More recently, attention has turned to neural network-based algorithms. These methods are advantageous due to their ability to handle intricate, nonlinear relationships among optimization variables. Neural networks excel at learning from data and adjusting their internal representations to optimize specific objectives. For example, Zhang *et al.* [29] introduced a robust deep neural network approach to address an aerodynamic shape optimization scenario. Similarly, in [28], a convolutional neural network-based algorithm was employed to approximate solutions for missile design problems.

Other researchers have proposed hybrid approaches that integrate a gradient method with a selected heuristic technique to tackle the challenge of local optimality. For instance, Dashti *et al.* [7] utilized a genetic algorithm to provide an effective initial estimate for the conjugate gradient method. This highlights the fact that a gradient method might encounter divergence issues when initialized poorly [24]. In [9], a hybrid genetic algorithm and conjugate gradient method, in conjunction with the finite element method, were employed to identify bilateral free boundaries. Mozaffari *et al.* [21] employed the imperialist competitive algorithm along with the conjugate gradient method and boundary element method to identify two interfaces. In [10], a hybrid differential evolution and quasi-Newton method were employed to approximate the depletion region in a pn junction semiconductor device.

One might question the preference for heuristic methods (HMs) over gradient-based approaches. The answer is straightforward. The main advantage of HMs lies in their ability to bypass the computation of the cost functional gradient, which can be numerically expensive, especially in the case of shape optimization. Shape optimization involves solving two additional problems: the adjoint problem and the problem of finding the shape gradient. Gradient methods also suffer from the issue of local optimality, where the choice of the initial guess must be close to the exact solution.

The main contribution of this paper is the introduction of a novel hybrid approach that combines particle swarm optimization (PSO) guided by fuzzy logic (FL). The proposed method is applied to a free boundary problem, which is formulated as a constrained minimization problem following a well-known shape optimization technique [14]. To ensure the uniqueness of the optimal solution, a suitable regularization term is incorporated, as suggested by previous works [12, 26]. In order to solve the problem numerically, we employ the finite element method for discretization and utilize the hybrid PSO guided by fuzzy logic for achieving the optimal solution. To demonstrate the efficacy of this hybridization, we compare it with the gradient-based quasi-Newton method, emphasizing the differences and showcasing the superiority of our proposed approach.

The remaining sections of this paper are structured as follows: In Section 2, we present the formulation of the optimization problem associated with the considered ill-posed problem and establish the existence of its solution. Section 3 complements the preceding section; we outline a numerical approach aimed at approximating the solution of our optimization problem. This approach combines particle swarm optimization with fuzzy logic and employs the finite element method. In Section 4, we present numerical examples to demonstrate the validity and performance of the new hybrid algorithm.

2. Construction of the constrained optimization problem

In this paper we consider the following class of inverse problem, we are looking forward to find (Ω, u) satisfying the state equation

$$\begin{cases} -\operatorname{div}(A\nabla u) = f & \text{in} \quad \Omega \\ A\nabla_\nu u = g_\delta & \text{on} \quad \partial\Omega, \\ u = j_\delta & \text{on} \quad \partial\Omega \end{cases} \tag{1}$$

with Ω is an open set in $\mathbb{R}^2$, $\partial\Omega = I_1 \cup \Gamma_1 \cup I_2 \cup \Gamma_2$, the configuration of Γ_1 and Γ_2 is unknown, while the rest of the boundary in known, ν is the unit normal vector, A and f are given functions, g_δ and j_δ are noisy Cauchy

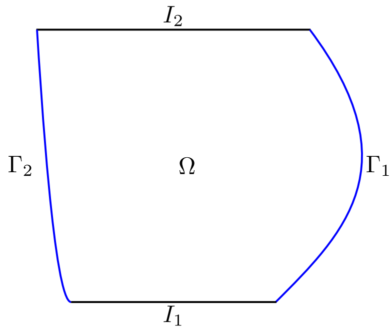

FIGURE 1. Geometry of Ω.

data of the exact data g and j satisfying

$$\|j_\delta - j\|_{H^{1/2}(\partial\Omega)} + \|g_\delta - g\|_{H^{-1/2}(\partial\Omega)} \leq \delta.$$

Let D be a fixed open set in $\mathbb{R}^2$, we assume the following parametrization of Ω, Γ_1 and Γ_2:

$$\Omega(\varphi,\psi) = \{(x,y) : \varphi(y) < x < \psi(y) \text{ and } y \in]a,b[\} \subseteq D$$
$$\Gamma_1(\varphi) = \{(\varphi(y),y) : y \in [a,b]\} \quad \text{and} \quad \Gamma_2(\psi) = \{(\psi(y),y) : y \in [a,b]\},$$

we define now the admissible shapes Θ_{ad} by:

$$\Theta_{ad} = \Big\{\Omega(\varphi,\psi)/(\varphi,\psi) \in (\mathcal{C}^{1,1}([a,b]))^2, \varphi \text{ and } \psi \text{ are Lipschitz functions}\Big\}.$$

Remark 2.1. Note that the free boundaries Γ_1 and Γ_2 must not intersect, we assume then

$$\max_{y\in[a,b]} \varphi(y) < \min_{y\in[a,b]} \psi(y).$$

Now, we split to help the inverse problem (1) into two state equations, the first one correspond to the Dirichlet problem

$$-\nabla(A\nabla u) = f \text{ in } \Omega \qquad \text{and} \qquad u = j_\delta \text{ on } \partial\Omega \tag{2}$$

associated to the variational form

$$\langle \mathcal{E}_D(\Omega); v\rangle = \int_\Omega A\nabla u \nabla v \mathrm{d}x - \int_\Omega f v \mathrm{d}x \text{ for all } v \in H_0^1(\Omega). \tag{3}$$

The Lax-Milgram the existence of a solution of (2) in $H^1(\Omega)$ is guaranteed. This solution will be denoted by $\mathcal{D}(\Omega)$. Thereafter we define the second state equation as next

$$-\nabla(A\nabla u) = f \text{ in } \Omega \qquad \text{and} \qquad A\partial_\nu u = g_\delta \text{ on } \partial\Omega \tag{4}$$

which is associated to the variational form

$$\langle \mathcal{E}_N(\Omega); v\rangle = \int_\Omega A\nabla u \nabla v \mathrm{d}x - \int_\Omega f v \mathrm{d}x - \int_{\partial\Omega} g_\delta v \mathrm{d}s \text{ for all } v \in H^1(\Omega). \tag{5}$$

The solution of weak formulation (5) will be denoted by $\mathcal{N}(\Omega)$, it does exist under the following compatibility condition

$$\int_{\Omega} f \mathrm{d}x + \int_{\partial\Omega} g_{\delta} \mathrm{d}s = 0 \tag{6}$$

with this condition, the strong form of the constrained pure Neumann problem reads

$$-\nabla(A\nabla u) = f \text{ in } \Omega, \qquad u = j_{\delta} \text{ on } \partial\Omega \qquad \text{and} \qquad \mathcal{B}(u) = 0 \tag{7}$$

where the constraint is defined as $\mathcal{B}(u) = \displaystyle\int_{\partial\Omega} u \mathrm{d}s$. To handle this constraint, we define the energy functional

$$K(u) = \frac{1}{2}\int_{\Omega} A|\nabla u|^2 \mathrm{d}x - \int_{\Omega} f u \mathrm{d}x - \int_{\partial\Omega} g_{\delta} u \mathrm{d}s$$

then, we write (7) as an optimization problem, we have

$$\min_{u \in H^1(\Omega)} K(u) \quad \text{ such that } \quad \mathcal{B}(u) = 0. \tag{8}$$

The constraint aims to force the solution of the unconstrained problem (4) to move away from the vicinity of the global minimum of the functional K [4].

Another representation for the optimization problem (8) can be obtained by introducing the next Lagrangian functional:

$$L(u, \rho) = K(u) + \rho(\mathcal{B}(u) - c)$$

ρ is a Lagrange multiplier. Then (8) can be formulated as the saddle point problem

$$\inf_{u \in H^1(\Omega)} \sup_{\rho \in \mathbb{R}} L(u, \rho) \tag{9}$$

we have the following existence result, which can be proven in a similar way as for the pure Neumann problem [4]

Theorem 2.2. *The saddle point problem (9) admits a unique solution. Moreover if (u, ρ) is a saddle point of (9) then u solves the optimization problem (8) and the second state problem (4) as well.*

Let us now recall the following classical result [6, 15]

Lemma 2.3. *For any $\Omega \in \Theta_{ad}$, the variational problem (3) and the constrained optimization problem (9) admit unique solutions in $H^1(\Omega)$ and $H^1_{\diamond}(\Omega)$ respectively. Moreover those solutions are bounded independently of the control variable Ω.*

Consider now the following energy functional:

$$J(\Omega, \mathcal{D}(\Omega), \mathcal{N}(\Omega)) = \frac{1}{2}\int_{\Omega} (\mathcal{D}(\Omega) - \mathcal{N}(\Omega))^2 \mathrm{d}x. \tag{10}$$

At this point, we summarize the shape optimization problem as follows:

$$\begin{aligned} &\min_{\Omega \in \Theta_{ad}} J(\Omega, \mathcal{D}(\Omega), \mathcal{N}(\Omega)) \\ &\text{s.t. } \langle \mathcal{D}(\Omega); v \rangle = 0 \quad \forall v \in H^1_0(\Omega) \\ &\text{and } \mathcal{N}(\Omega) = \inf_{u \in H^1(\Omega)} \sup_{\rho \in \mathbb{R}} L(u, \rho) \end{aligned} \tag{11}$$

Theorem 2.4. *The constrained optimization problem* (11) *admit at least one solution.*

To establish the existence of an optimal solution satisfying the constraints, we introduce the space of feasible solutions, which comprises all the potential solutions that meet the optimization problem's constraints. This space reads

$$\mathcal{F} = \Big\{(\Omega, \mathcal{D}(\Omega), \mathcal{N}(\Omega));\ \Omega \in \Theta_{ad},\ \mathcal{D}(\Omega) \text{ solves (3) and } \mathcal{N}(\Omega) \text{ solves (9)}\Big\}.$$

Now, we endow this space with the following topology

Definition 2.5. Let us consider a sequence $(\Omega_n, \mathcal{D}(\Omega_n), \mathcal{N}(\Omega_n))_n$ in $\mathcal{F}$ and an element $(\Omega, \mathcal{D}(\Omega), \mathcal{N}(\Omega))$ of $\mathcal{F}$. The space $\mathcal{F}$ is equipped by the following topology

$$(\Omega_n, \mathcal{D}(\Omega_n), \mathcal{N}(\Omega_n)) \underset{n\longrightarrow\infty}{\longrightarrow} (\Omega, \mathcal{D}(\Omega), \mathcal{N}(\Omega)) \text{ in } \mathcal{F} \Leftrightarrow \begin{cases} \Omega_n \underset{n\longrightarrow\infty}{\longrightarrow} \Omega \\ \mathcal{D}(\Omega_n) \underset{n\longrightarrow\infty}{\rightharpoonup} \mathcal{D}(\Omega) \\ \mathcal{N}(\Omega_n) \underset{n\longrightarrow\infty}{\rightharpoonup} \mathcal{N}(\Omega) \end{cases} \tag{12}$$

Ω_n converges to Ω is in the sense of their characteristic functions, $\mathcal{D}(\Omega_n)$ and $\mathcal{N}(\Omega_n)$ converge weakly to $\mathcal{D}(\Omega)$ and $\mathcal{N}(\Omega)$ in $H^1(\Omega)$, which is equivalent to the convergence of their uniform extensions in $H^1(D)$.

With the above topology, we can prove the next compactness result

Lemma 2.6. *The space of feasible solutions $\mathcal{F}$ is compact.*

Proof. Let $\{(\Omega_n, \mathcal{D}(\Omega_n), \mathcal{N}(\Omega_n))\}_n$ be a sequence of $\mathcal{F}$ such that $\mathcal{D}(\Omega_n)$ solves (3) and $\mathcal{N}(\Omega_n)$ solves (9). We shall prove the existence of a converging subsequence of $\{(\Omega_n, \mathcal{D}(\Omega_n), \mathcal{N}(\Omega_n))\}$ in $\mathcal{F}$.

First we have $\Omega_n = \Omega(\varphi_n, \psi_n)$, using Ascoli-Arzéla Theorem we ensure that (φ_n, ψ_n) that converges uniformly as a subsequence to an element (φ, ψ). It is easy to deduce that $\Omega = \Omega(\varphi, \psi)$ lives in Θ_{ad}.

Since $\mathcal{D}(\Omega_n)$ is bounded according to Lemma 2.3, we can use the Rellich Theorem to show the existence of an element u_D such that $\mathcal{D}(\Omega_n)$ converges weakly to u_D in $H^1(D)$. Similarly to [6, 8] we can prove that u_D satisfies the variational form (3) on Ω.

For the second state solution $\mathcal{N}(\Omega_n)$, again with Rellich Theorem we have the existence of an element u_N such that $\mathcal{N}(\Omega_n)$ converges weakly to u_N in $H^1(D)$, we must show that u_N satisfies the constrained optimization problem (8), which will gives us that u_N solves (9). For that we only need to prove that

$$\lim_{n\to\infty} \int_{\partial\Omega_n} \mathcal{N}(\Omega_n)\mathrm{d}s = \int_{\partial\Omega} u_N \mathrm{d}s. \tag{13}$$

Since the moving parts of $\partial\Omega_n$ are $\Gamma_{1,n}$ and $\Gamma_{2,n}$ we will only show that

$$\lim_{n\to\infty} \int_{\Gamma_{1,n}} \mathcal{N}(\Omega_n)\mathrm{d}s = \int_{\Gamma_1} u_N \mathrm{d}s. \tag{14}$$

For that we follow the same steps as in [6], we write then

$$\int_{\Gamma_{1,n}} \mathcal{N}(\Omega_n)\mathrm{d}s - \int_{\Gamma_1} u_N \mathrm{d}s = \int_{\Gamma_1} (\mathcal{N}(\Omega_n) - u_N)\mathrm{d}s + \int_{\Gamma_{1,n}} \mathcal{N}(\Omega_n)\mathrm{d}s - \int_{\Gamma_1} \mathcal{N}(\Omega_n)\mathrm{d}s \tag{15}$$

the first integral, we manipulate Holder inequality and the fact that the trace mapping of $H^1(D)$ into $L^2(\Gamma_1)$ is continuous and compact, therefore we get

$$\lim_{n\to\infty} \int_{\Gamma_1} (\mathcal{N}(\Omega_n) - u_N)\mathrm{d}s = 0. \tag{16}$$

We have $\mathcal{N}(\Omega_n)|_{\Gamma_{1,n}} = \mathcal{N}(\Omega_n) \circ \varphi_n$ and $\mathcal{N}(\Omega_n)_{|\Gamma_1} = \mathcal{N}(\Omega_n) \circ \varphi$, then we write

$$\begin{aligned}\int_{\Gamma_{1,n}} \mathcal{N}(\Omega_n)\mathrm{d}s - \int_{\Gamma_1} \mathcal{N}(\Omega_n)\mathrm{d}s = \int_a^b \mathcal{N}(\Omega_n) \circ \varphi_n \sqrt{1+(\varphi_n')^2}\mathrm{d}s \\ - \int_a^b \mathcal{N}(\Omega_n) \circ \varphi \sqrt{1+(\varphi')^2}\mathrm{d}s\end{aligned} \tag{17}$$

it implies that

$$\begin{aligned}\int_{\Gamma_{1,n}} \mathcal{N}(\Omega_n)\mathrm{d}s - \int_{\Gamma_1} \mathcal{N}(\Omega_n)\mathrm{d}s = \int_a^b \mathcal{N}(\Omega_n) \circ (\varphi_n - \varphi) \sqrt{1+(\varphi_n')^2}\mathrm{d}s \\ + \int_a^b \mathcal{N}(\Omega_n) \circ \varphi \left(\sqrt{1+(\varphi_n')^2} - \sqrt{1+(\varphi')^2}\right)\mathrm{d}s\end{aligned} \tag{18}$$

we deduce

$$\lim_{n\to\infty} \int_{\Gamma_{1,n}} \mathcal{N}(\Omega_n)\mathrm{d}s - \int_{\Gamma_1} \mathcal{N}(\Omega_n)\mathrm{d}s = 0 \tag{19}$$

similarly we can obtain

$$\lim_{n\to\infty} \int_{\Gamma_{2,n}} \mathcal{N}(\Omega_n)\mathrm{d}s - \int_{\Gamma_2} \mathcal{N}(\Omega_n)\mathrm{d}s = 0. \tag{20}$$

Finally we obtain the convergence (13).

To this end, we can easily deduce that u_N satisfies the variational form (5), with the convergence (13) it follows that $\mathcal{B}(u_N) = 0$, hence u_N solves the optimization problem (8). We deduce that from the sequence $\{\Omega_n, \mathcal{D}(\Omega_n), \mathcal{N}(\Omega_n)\}$ we have shown the existence of a sub-sequence that converge weakly to an element (Ω, u_D, u_N). Thereafter the space $\mathcal{F}$ is compact. □

Now, similarly to the results in [9] we can prove that

Lemma 2.7. *The cost functional J is semi-lower continuous on the space $\mathcal{F}$.*

To conclude the theoretical part, we have covered all the necessary concepts. In the upcoming section, we will introduce our proposed numerical scheme for solving the optimization problem (11).

3. Fuzzy particle swarm optimization method

Particle Swarm Optimization (PSO) is a well-known heuristic method that has gained significant popularity in the last few decades [20, 24]. The core principle of PSO involves working with a population, referred to as a swarm, comprised of candidate solutions. These solutions are iteratively updated based on specific formulas. Each individual within the swarm is known as a particle and represents a potential solution to the optimization problem.

In the PSO algorithm, each particle p_i within the swarm S of size N moves according to its velocity v_i. The best visited position of particle p_i is stored in a variable $\widehat{p}_i$, while the best position among all particles in the swarm is denoted by $\widehat{p}g$. The PSO process continues until it reaches either a maximum iteration number $IterM$ or when the cost function achieves a predefined tolerance level (tol).

In the classic version of PSO algorithm, a particle p_i is updated iteratively with respect to its velocity v_i using the next formula:

$$p_i^{n+1} = p_i^n + v_i^{n+1}. \tag{21}$$

The velocity is updated by a linear combination of the velocity at the previous iteration, the differences of the best position $\widehat{p}_i^n$, the global best position p_g^n with the current particle p_i, this iteration is written as

$$v_i^{n+1} = \omega v_i^n + c_1 r_1(\widehat{p}_i^n - p_i^n) + c_2 r_2(\widehat{p}_g^n - p_i^n) \tag{22}$$

where n is the iteration number, ω is called inertia weight taken in $[0,1]$. The coefficients c_1 and c_2 are the cognitive and the social parameter respectively, usually set $c_1 = c_2 = 2$. r_1 and r_2 are two random vectors numbers distributed in the range $[0,1]$, of the same size as the particle p_i.

It is hard to guide the optimization while the particle p_i is not feasible. Different modification of PSO have been proposed [17, 25]. One can improve the process of the PSO is to consider a generalized update of the velocity v_i, by adding a positive step size α in $[0,1]$, we write the iteration as:

$$v_i^{n+1} = \omega v_i^n + c_1 r_1(\widehat{p}_i^n - p_i^n) + \alpha c_2 r_2(\widehat{p}_g^n - p_i^n) + (1-\alpha) c_2 r_2(\widehat{p}_g^n - p_i^n). \tag{23}$$

The parameter α plays a crucial role in balancing the influence of the global best and the best local particle in the PSO algorithm. When the diversity of the PSO decreases, leading to the possibility of early convergence or being trapped in a local optimum, a larger value of α is required to enhance the exploration capability of the algorithm. Conversely, when the diversity is high and exploration is already sufficient, a smaller value of α can be used to focus more on exploiting the local best solutions.

The inertia weight plays a crucial role in the PSO process as it impacts the balance between exploration and exploitation. A higher inertia weight facilitates global exploration, allowing particles to explore a broader search space. Conversely, a lower inertia weight promotes local exploitation, focusing particles on exploiting promising regions.

This is where the utility of Fuzzy Logic becomes apparent. The selection of the inertia weight and cognitive parameter α can be determined based on information from certain input variables. Through Fuzzy Logic, we can derive the appropriate values of w and α considering the specific conditions and characteristics of the optimization problem. Fuzzy Logic enables us to capture complex relationships between input variables and determine corresponding outputs in a flexible and adaptive manner.

Therefore, by employing Fuzzy Logic, we can dynamically adjust the inertia weight and cognitive parameter throughout the PSO process, achieving an effective balance between exploration and exploitation. This approach enhances the overall performance of the algorithm and improves its ability to find optimal solutions.

Fuzzy Logic [2, 13] is a soft computing method that effectively handles uncertainties in system parameters. The Fuzzy Inference System (FIS) is a control strategy that generates outputs based on given inputs, utilizing key concepts of fuzzy logic, such as fuzzy sets, membership functions, and linguistic variables.

A fuzzy inference system comprises three components: fuzzification, inference engine (IF/THEN rules), and defuzzification. In the fuzzification unit, input data is converted into fuzzy data and represented using defined linguistic variables. This fuzzy data is then processed by the inference engine, which makes final decisions based on the specified rules. Finally, the fuzzy outputs from the inference engine are converted into crisp outputs through the defuzzification unit.

In this application our inputs data are the increment of the best global optimum cost C_{igo} and the deviation of the fitness particles C_{dev}, they are given by:

$$C_{igo} = \widehat{p}_g^{n-1} - \widehat{p}_g^n \quad \text{and } C_{dev} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(J_i - \overline{J}\right)^2}. \tag{24}$$

J_i is the fitness of the particle x_i, and $\overline{J}$ is the average fitness in the current swarm.

Basing on nine rules, the FIS will generate two outputs, the inertia weight w and the step size α. First we define the linguistic variable used in our rules, L is low, M is medium and H is high. We write in the next table the used rules in our FIS.

Each column in Table 1 represents a rule that is read as follows: if C_{igo} is low and C_{dev} is low, then the output values of w and α are high. The rationale behind selecting these rules is to introduce diversity into the population when the increment of the best global optimum cost and the deviation of the fitness particles decrease (*i.e.*, C_{dev} is low and C_{dev} is low). This is achieved by increasing the values of w and α to promote exploration and prevent premature convergence.

The process of the FIS is summarized in algorithm 1.

TABLE 1. The used parameters in PSOFL.

C_{igo}	L	L	L	M	M	M	H	H	H
C_{dev}	L	M	H	L	M	H	L	M	H
w	H	M	L	M	H	L	H	M	L
α	H	M	L	L	H	M	H	L	M

Algorithm 1 Fuzzy Inference System

Inputs: C_{igo} and C_{dev}
(1) Fuzzification : Generate fuzzy data using the membership entrance function.
(2) Inference engine : Generate results by applying the rules.
(3) Defuzzification : Apply the membership function for the output variables.
Outputs: w and α

Again we call the finite element method to solve the problems (3) when Γ_1 and Γ_2 are supposed known. We summarize the steps of the PSOFL is algorithm 2 and its flowchart in 2.

Algorithm 2 PSOFL

Choose a precision ε, the coefficients c_1, c_2, the cycle size N_{cycle}, and the bonds $L_1^0, L_1^1, L_2^0, L_2^1$. Set initial values $w = \alpha = 1$, and set $m = 0$.
(1) Generate a random swarm $S(0)$ in the range $[L_1^0, L_1^1] \times [L_2^0, L_2^1]$
(2) For each individual $\Omega_{i,m}$ in the swarm $S(m)$ compute the fitness:
– compute the state solutions $\mathcal{D}(\Omega_{i,m})$ and $\mathcal{N}(\Omega_{i,m})$
– evaluate $J(\Omega_{i,m})$.
(3) Extract the current best particle in the swarm $S(m)$,
(4) **if** $| J(\Omega_{best}) | \leq \varepsilon$
Stop,
else
set $t = t + 1$ and continue to step (5).
(5) Update the swarm $S(m)$ using equations (21) and (23),
(6) **if** $mod(m, N_{cycle}) = 0$
- compute C_{igo} and C_{dev} using (24),
- update w and α by algorithm 1,
else
Continue.
(7) Set $m = m + 1$ and back to step (2).

Next, we will proceed with testing the aforementioned algorithm to solve our shape optimization problem (11).

4. Numerical results and discussion

In this section, we will conduct a series of experiments to validate the effectiveness of the proposed hybridization approach using several benchmark optimization problems. The latter part of this section will focus on applying the PSOFL algorithm to a specific shape design problem.

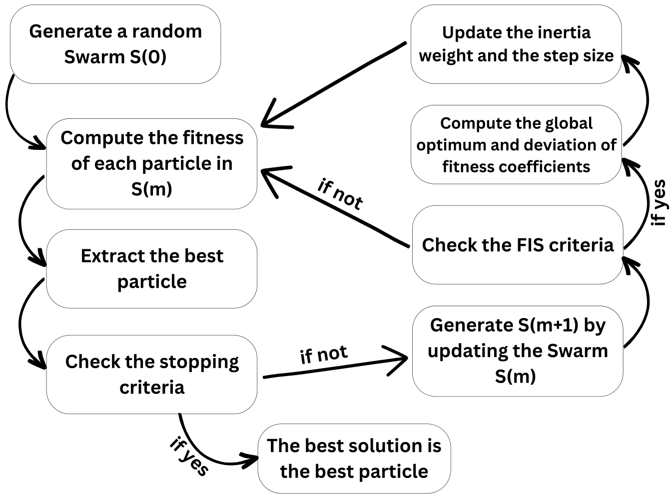

FIGURE 2. Flowchart of algorithm 2.

TABLE 2. The cost for some examples.

		PSO			PSOFL		
Function	n	Cost	Iter	cpu	Cost	Iter	cpu
Booth	2	9.037e-08	39	0.041	8.939e-09	37	0.091
Himmeblau's	2	2.143e-09	40	0.121	7.227e-09	35	0.083
McCormick	2	-1.9132	43	0.027	-1.9132	41	0.031
Lvi	2	7.022e-08	51	0.043	1.582e-08	44	0.039
Sphere	2	7.653e-08	35	0.021	3.688e-08	33	0.071
	10	4.218e-04	100	0.231	8.379e-06	100	0.240
Rastrigin	2	3.158e-08	51	0.049	7.015e-08	58	0.072
	10	5.977	100	0.078	4.975	100	0.103
Rosenbrook	2	2.297e-05	100	0.076	9.581e-08	85	0.068
	10	7.747	100	0.067	5.1326	100	0.078

4.1. Algorithm validation

In this part, we present the obtained numerical results by testing PSOFL for finding the minimum of the some benchmark functions selected from [1], where the optimization problem can be written as

$$\min_{x \in I} F(x), \quad I \subset \mathbb{R}^n. \tag{25}$$

The size of swarm is set to 20 and the maximum number of iterations is 100 and the function tolerance is set to be 10^{-7}. In Table 2 we summarize the obtained cost, number of iterations and the elapsed time for each example.

It is evident that the new hybrid PSO demonstrates superior efficiency compared to the standard version. The number of iterations is reduced, and the cost estimation is more accurate with the new algorithm. It should be noted that the proposed hybrid algorithm may take slightly longer to execute compared to the old version, primarily due to the additional time spent on the fuzzy inference system stage. However, despite the slight increase in execution time, the hybridization approach proves to be efficient and robust when compared to the standard PSO version.

TABLE 3. The used parameters in PSOFL.

Swarm size	Max iterations	Cognitive coefficient	Social coefficient
30	200	2	0.5

4.2. Application to shape design problem

In this part, we present the numerical results obtained using the proposed algorithm. All the scripts were implemented in MATLAB R2018b and the tests were conducted on a laptop with an Intel Core i5 1.70 GHz processor and 8 GB of RAM.

We assume that the state problems have the following analytical solution:

$$u(x,y) = exp(x+y), \qquad x,y \in \Omega(\varphi,\psi)$$

. To simplify the representation of the free boundaries, we assume that $A(x) = 1$ and compute the remaining functions f, j, and g using the given analytical solution. Additionally, we parameterize the free boundaries using Bzier curves [23]. In other words, each particle in the swarm is represented by a vector containing the control points of Γ_0 and Γ_1. The use of Bzier curves facilitates the numerical computation, reduces the number of variables, and provides a more convenient solution.

As mentioned earlier, we use a maximum number of iterations as the stopping criterion for the PSOFL algorithm in this application. For the Fuzzy Inference System, we have chosen Gaussian membership functions for the inputs and triangular membership functions for the outputs. Table 3 presents the parameters used to tune Algorithm 2.

To ensure the credibility of our proposed method, we will compare it to a gradient method. In this case, we consider the quasi-Newton method described in the work [8]. We keep the same settings for the quasi-Newton algorithm as in the referenced work. For the sake of notation, we will denote the algorithm in [8] as "Algo-[8]".

The first example deals with the approximation of the exact boundaries Γ_1 and Γ_2 given by:

$$\Gamma_1 = \Big\{\Big(\frac{y-3}{2}, y\Big) /\ y \in [-1,1]\Big\} \text{ and } \Gamma_2 = \Big\{\Big(\frac{4+(y+1)^2}{4}, y\Big) / \ y \in [-1,1]\Big\},$$

the second example, we assume the free boundaries are given by:

$$\Gamma_1 = \Big\{\Big(\frac{(y-1)^2-8}{4}, y\Big) / \ y \in [-1,1]\Big\},$$
$$\Gamma_2 = \Big\{\Big(2+sin(\frac{2\pi y}{3}), y\Big) / \ y \in [-1,1]\Big\}.$$

In Figure 3 we remark that the optimal and the exact boundaries for the first example are coincided. The optimized parameters w and α in the first example are 0.65 and 0.75 respectively.

In the second example, the approximated free boundaries closely resemble the exact ones, with the only difference being the bottom limit of Γ_2 where they are not coincided. However, overall, the numerical boundaries obtained are of good quality. The Fuzzy Inference System (FIS) has generated the optimal parameters again, with $w = 0.65$ and $\alpha = 0.75$. The results obtained using the proposed approaches are comparable to an order of magnitude of 10^{-4} in Example 1 and 10^{-2} in Example 2. It is worth noting that the PSOFL algorithm achieves the optimal solution with a fewer number of iterations compared to the quasi-Newton method.

In addition, it is important to highlight the difference in optimization using PSO and PSOFL to justify the robustness of this hybrid approach. In Table 4, we provide the optimal costs obtained for each example using PSO, PSOFL, and the quasi-Newton method. Furthermore, we compare the elapsed time (CPU) for each method.

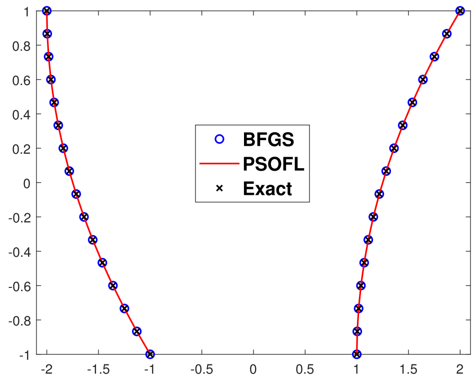

FIGURE 3. Comparison of the optimal boundaries of BFGS and PSOFL for example 1.

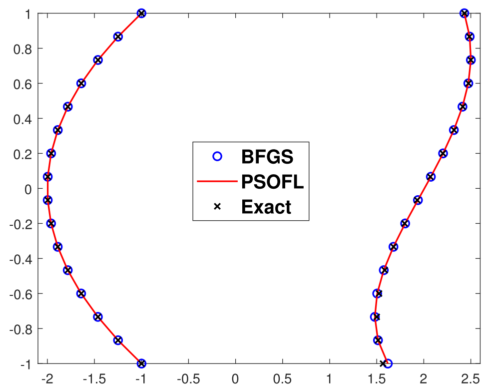

FIGURE 4. Comparison of the optimal boundaries of BFGS and PSOFL for example 2.

TABLE 4. The cost for some examples.

		PSO	PSOFL	BFGS
Example 1	COST	1.518×10^{-2}	1.103×10^{-5}	1.825×10^{-5}
	CPU	87.44	97.92	23.53
Example 2	COST	3.661×10^{-1}	2.712×10^{-2}	2.714×10^{-2}
	CPU	90.52	99.74	25.91

It is evident that there is a significant difference in the optimal costs between PSO and PSOFL. The hybridization of PSO with fuzzy logic has improved the performance of the standard PSO method. It can be concluded that PSOFL generates higher quality solutions compared to PSO alone. The results obtained using algorithm [8] are comparable to those of PSOFL, as both methods yield high quality solutions. In example 1, PSOFL achieves a better cost than the BFGS method, while in example 2, both methods converge to the same cost. It is also noticeable that heuristic methods take longer to find the optimal solution, which is normal due to the large number of evaluations of the state equation. Despite this drawback of heuristic methods, they provide more accurate solutions in fewer iterations.

TABLE 5. The optimal cost for the case of noisy data.

Noise level	Example 1		Example 2	
	PSOFL	BFGS	PSOFL	BFGS
0%	0.00001	0.00002	0.02712	0.02710
1%	0.00421	0.00392	0.02951	0.02833
5%	0.01580	0.01744	0.03535	0.03819
10%	0.03308	0.03908	0.03716	0.03932

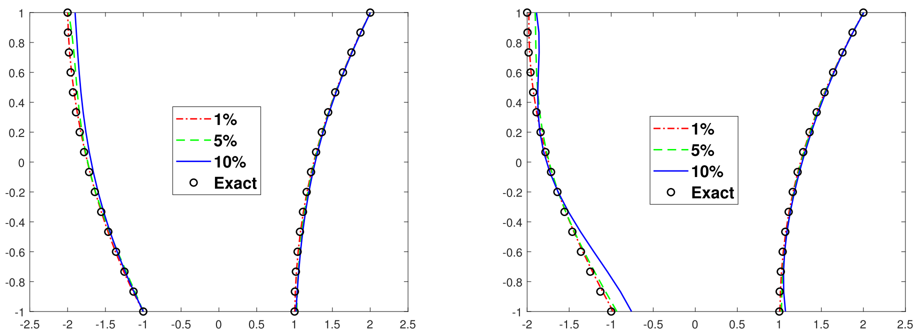

FIGURE 5. Comparison of PSOFL in left and BFGS in right for example 1 with noisy data.

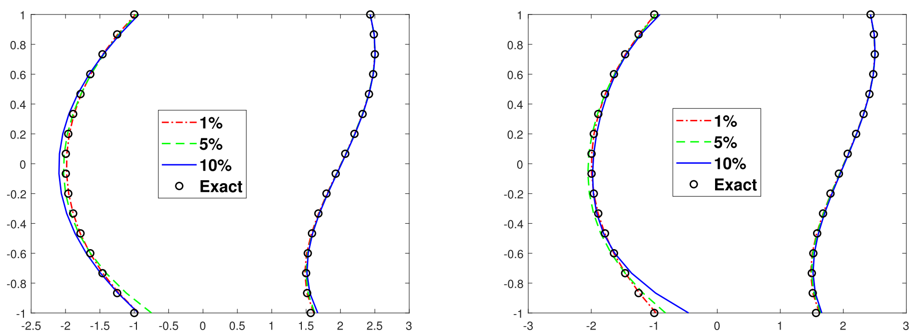

FIGURE 6. Comparison of PSOFL in left and BFGS in right for example 2 with noisy data.

In the case of noisy measurements, we construct the noisy data g_2^ρ and g_3^ρ by adding noise to the exact data g_2 and g_3. The noise is generated by two random vectors R_1 and R_2, uniformly distributed in the interval $[-1, 1]$. The noisy data are then given by $g_2^\rho = g_2 + \rho R_1$ and $g_3^\rho = g_3 + \rho R_2$, where ρ represents the noise level. We rerun our proposed algorithms to approximate the previous two examples using the noisy data. Table 5 presents the results obtained in this case.

The following figures depict the optimal boundaries for the two examples, considering different noise levels. Despite the presence of noise in the measurements, the optimal boundaries obtained using the proposed algorithms still exhibit good quality and provide a satisfactory approximation of the exact boundaries. These results demonstrate the effectiveness of the proposed algorithms in solving the inverse problem under noisy conditions.

5. Conclusion

In this paper, we addressed a bilateral free boundaries problem and successfully applied the shape optimization technique to convert it into a minimization problem. We established the existence of a unique solution for the resulting optimization problem. To solve this inverse problem, we proposed a numerical approach based on the hybrid Particle Swarm Optimization/Fuzzy Logic, in conjunction with the finite element method.

We demonstrated the effectiveness of our proposed approaches by conducting various experiments on different types of boundaries. The results, presented in Figures 3 and 4, showcase the ability of our algorithms to accurately approximate the exact boundaries. Furthermore, we examined the robustness of our algorithms by considering noisy data. As expected, the presence of noise affected the quality of the approximation, as shown in Figures 5 and 6. However, even under these challenging conditions, the optimal bilateral boundaries remained close to the exact boundaries.

In conclusion, we can confirm that the combination of PSOFL with the FEM is a promising strategy for solving bilateral free boundaries problems. The proposed algorithms exhibit advantages in terms of iteration numbers and the quality of the optimal boundaries, particularly when dealing with noisy data.

References

[1] N. Andrei, An unconstrained optimization test functions collection. *Adv. Model. Optim.* **10** (2008) 147–161.

[2] B. Barnabas, Mathematics of Fuzzy Sets and Fuzzy Logic. Springer, Berlin, Heidelberg (2013).

[3] S. Bartles, Numerical Approximation of Partial Differential Equations. Springer (2016).

[4] P. Bochev and R.B. Lehoucq, On the finite element solution of the pure neumann problem. *SIAM Rev.* **47** (2005) 50–66.

[5] M. Bouchlaghem and E. Mermri, A Uzawa algorithm with multigrid solver for a bilateral obstacle problem. *Appl. Math. Comput.* **389** (2021) 1125553.

[6] A. Boulkhemair, A. Chakib and A. Nachaoui, A shape optimization approach for a class of free boundary problems of Bernoulli type. *Appl. Math.* **58** (2013) 205–221.

[7] M. Dashti Ardakani and M. Khodadad, Shape estimation of a cavity by inverse application of the 2D elastostatics problem. *Int. J. Comput. Methods* **10** (2013) 1350042.

[8] Y. El Yazidi and A. Ellabib, Augmented Lagrangian approach for a bilateral free boundaries problem. *J. Appl. Math. Comput.* **67** (2021) 69–88.

[9] Y. El Yazidi and A. Ellabib, An iterative method for optimal control of bilateral free boundaries problem. *Math. Methods Appl. Sci.* **44** (2021) 11664–11683.

[10] Y. El Yazidi and A. Ellabib, A new hybrid method for shape optimization with application to semiconductor equations. *Numer. Algebra Control. Optim.* (2021).

[11] Y. Ge, Y. Lin, S. Tao, Q. He, B. Chen and S.-M. Huang, Shape optimization for a tube bank based on the numerical simulation and multi-objective genetic algorithm. *Int. J. Therm. Sci.* **161** (2021) 106787.

[12] M. Hanke, H.W. Engl and A. Neubauer, Regularization of inverse problems. In Vol. 375 of *Mathematics and Its Applications*, Springer (2000).

[13] J. Harris, An Introduction to Fuzzy Logic Applications. Springer, Netherlands (2000).

[14] J. Haslinger and R.A.E. Mäkinen, Introduction to Shape Optimization: Theory, Approximation, and Computation. SIAM (2003).

[15] M. Hinze, B. Kaltenbacher and T.N.T. Quyen, Identifying conductivity in electrical impedance tomography with total variation regularization. *Numer. Math.* **138** (2018) 723–765.

[16] C.H. Huang and C.C. Shih, A shape identification problem in estimating simultaneously two interfacial configurations in a multiple region domain. *Appl. Therm. Eng.* **26** (2006) 77–88.

[17] S. Khan, M. Kamran, O.U. Rehman, L. Liu and S. Yang, A modified PSO algorithm with dynamic parameters for solving complex engineering design problem. *Int. J. Comput. Math.* **95** (2018) 2308–2329.

[18] S. Koike, Regularity of solutions of obstacle problems -old & new-. In Vol. 346 of *Nonlinear Partial Differential Equations for Future Applications*, edited by S. Koike, H. Kozono, T. Ogawa, S. Sakaguchi, PDEFA 2017. Springer Proceedings in Mathematics & Statistics, Springer, Singapore (2017).

[19] J.C. Liu, Shape Reconstruction of Conductivity Interface Problems. *Int. J. Comput. Methods* **16** (2018) 1850092.

[20] C. Maurice, Particle Swarm Optimization. ISTE (2006).

[21] M.H. Mozaffari, M. Khodadad and M. Dashti Ardakani, Simultaneous identification of multi-irregular interfacial boundary configurations in non-homogeneous body using surface displacement measurements. *J. Mech. Eng. Sci.* **231** (2016) 1–12.

[22] A. Nejat, P. Mirzabeygi and P.M. Shariat, Airfoil shape optimization using improved Multiobjective Territorial Particle Swarm algorithm with the objective of improving stall characteristics. *Struct. Multidiscipl. Optim.* **49** (2014) 953–967.

[23] H. Prautzsch, W. Boehm and M. Paluszny, Bézier and B-Spline Techniques. Springer, Berlin, Heidelberg (2002).

[24] S. Salhi, Heuristic Search the Emerging Science of Problem Solving. Palgrave Macmillan (2017).

[25] F. Valdez, J.C. Vazquez, P. Melin and O. Castillo, Comparative study of the use of fuzzy logic in improving particle swarm optimization variants for mathematical functions using co-evolution. *Appl. Soft Comput.* **52** (2017) 1070–1083.

[26] C.R. Vogel, Computational methods for inverse problems. In: *Society for Industrial and Applied Mathematics* (2002) 195.

[27] C. Wang, J.M. Koh, T. Yu, N.G. Xie and K.H. Cheong, Material and shape optimization of bi-directional functionally graded plates by GIGA and an improved multi-objective particle swarm optimization algorithm. *Comput. Methods Appl. Mech. Eng.* **366** (2020) 113017.

[28] P. Wu, W. Yuan, L. Ji, L. Zhou, Z. Zhou, W. Feng and Y. Guo, Missile aerodynamic shape optimization design using deep neural networks. *Aerosp. Sci. Technol.* **126** (2022) 107640.

[29] X. Zhang, F. Xie, T. Ji, Z. Zhu and Y. Zheng, Multi-fidelity deep neural network surrogate model for aerodynamic shape optimization. *Comput. Methods Appl. Mech. Eng.* **373** (2021) 113485.

Please help to maintain this journal in open access!

This journal is currently published in open access under the Subscribe to Open model (S2O). We are thankful to our subscribers and supporters for making it possible to publish this journal in open access in the current year, free of charge for authors and readers.

Check with your library that it subscribes to the journal, or consider making a personal donation to the S2O programme by contacting subscribers@edpsciences.org.

More information, including a list of supporters and financial transparency reports, is available at https://edpsciences.org/en/subscribe-to-open-s2o.