Numerical Algorithms (2024) 96:621–663
https://doi.org/10.1007/s11075-023-01660-4

ORIGINAL PAPER

An improved numerical approach for solving shape optimization problems on convex domains

Abdelkrim Chakib[1] · Ibrahim Khalil[1] · Azeddine Sadik[1]

Received: 31 January 2023 / Accepted: 4 September 2023 / Published online: 14 October 2023

Abstract
This work is devoted to show the efficiency of a new numerical approach in solving geometrical shape optimization problems constrained to partial differential equations, on a family of convex domains. More precisely, we are interested to an improved numerical optimization process based on the new shape derivative formula, using the Minkowski deformation of convex domains, recently established in Boulkhemair and Chakib (J. Convex Anal. **21**(n°1), 67–87 2014), Boulkhemair (SIAM J. Control Optim. **55**(n°1), 156–171 2017). This last formula allows to express the shape derivative by means of the support function, in contrast to the classical one expressed in term of vector fields Henrot and Pierre 2005, Delfour and Zolésio 2011, Sokolowski and Zolesio 1992. This avoids some of the disadvantages related to the classical shape derivative approach, when one use the finite elements discretization for approximating the auxiliary boundary value problems in shape optimization processes Allaire 2007. So, we investigate here the performance of the proposed shape optimization approach through the numerical resolution of some shape optimization problems constrained to boundary value problems governed by Laplace or Stokes operator. Notably, we carry out a comparative numerical study between its resulting numerical optimization process and the classical one. Finally, we give some numerical results showing the efficiency of the proposed approach and its ability in producing good quality solutions and in providing better accuracy for the optimal solution in less CPU time compared to the classical approach.

Keywords Numerical comparative study · Shape optimization · Geometric optimization · Shape derivative · Minkowski deformation · Support functions · Convex domain

AMS Subject Classification 65K10 · 46N10 · 49Q10 · 49Q12

✉ Abdelkrim Chakib
chakib.fstbm@gmail.com

1 Applied Mathematics Team (AMT), Faculty of Sciences and Techniques, Sultan Moulay Slimane University, Beni Mellal, Morocco

1 Introduction

The shape optimization is a classical and ubiquitous field of research whose study becomes popular in academic researches and industry due to its increasingly wide applications in the field of computer science and engineering [14, 20, 30], structural mechanics [1, 20], optimal control [1, 24, 38], acoustics [42, 43], electromagnetics [19, 20] and chemical reactions [44]. The difficulty to dealing with this type of problems is to obtain analytically the optimal shapes which is usually impossible, notably in the usual case where the optimal shape design problems are constrained to partial differential equations (PDEs). So, a variety of numerical approaches and techniques find their powers to localize (approximate) optimal shapes with the help of the efficient PDEs discretization methods and modern optimization processes [5, 17, 25, 27].

In this respect, several questions arise when one deals with the study of a shape optimization problem. Notably, the existence of an optimal shape, its regularity as well as its geometrical property. On the other hand, the numerical investigation of shape optimization problems using the gradient optimization process is based on the study of the first variation of the cost functional, and in particular on the computation of its gradient or what one call the shape derivative. This notion of derivation with respect to domains was intensively studied in the 1980s by several authors. The first result concerning the differentiability with respect to perturbations of a geometrical domain was obtained by Hadamard in 1907 for solving an eigenvalue shape optimization problem [18]. Then, the shape derivative was introduced by Céa [8–11] and developed later by Murat and Simon [31, 36, 37] and extensively investigated in more recent works [15, 16, 40, 41], as well as in the books [1, 7, 14, 24, 38]. Also, different approaches, were applied to solve many problems in shape optimization, see, for example, the books [1, 14, 24, 34, 38] and the references therein. Finally, let us also mention that recent discussions on the so-called pre-shape derivative methods, based on the volume representation of the shape derivative, which enables to perform mesh and shape optimization approaches at the same time, are done in [28, 29].

This work is devoted to show the efficiency of a new numerical shape optimization approach, based on the shape derivative formula, involving the Minkowski deformation on a family of convex domains, recently established in [2, 3], in solving geometrical shape optimization problems. More precisely, we are interested to an improved numerical shape optimization process, based on this shape derivative formula allowing us to establish its expression by means of the support function, in contrast to the classical shape derivative expressed in term of vector fields [14, 24, 38]. In this context, we precise that the classical approach presents some difficulties from both theoretical and numerical point of view. For example, when one wants to connect the set of admissible domains with vector fields, one has to suppose high smoothness conditions on the initial data in order to differentiate functions depending on the domain. Note also that to solve a conditional shape optimization problem by this method is yet more complicated and usually requires to reduce it to a non conditional problem (for example, by Lagrange's multipliers method). Moreover, we believe that the numerical optimization processes involving the classical shape derivative presents some disadvantages. We refer here to [1], for example, for explanations about the issues that arise when implementing numerically the gradient optimization algorithm using the classical shape derivative. Briefly, when one uses the deformation by vector fields, we have to extend the vector field (obtained only on the boundary) to the whole domain or to re-mesh the domain, at each iteration, and both tasks are expensive. In order to avoid a part of the above issues, we define and use another way of variation of domains, a way that is linked to the convexity context and based on the Minkowski sum. Recall that for any convex bounded domain the support function of this domain is a continuous convex and positive homogeneous function. Conversely, it is known that each continuous convex and positive homogeneous function is the support function of a convex bounded set (sub-differential of this function at the origin). Using this fact, the variation of domains is clearly characterized by the variation of the corresponding support function. Then, when solving optimal shape design problems numerically using an optimization process based the new shape derivative formula, one gets a support function at each step of the implementation, the domain being recovered as the sub-differential of this support function. So in order to show the efficiency of this approach, which is successfully applied for solving concrete problems in [4, 5] using boundary element method, we investigate here its numerical comparative study with the classical one, in solving some shape optimization problems of minimizing appropriate cost functionals constrained to elliptic boundary value problems. In this respect, let us mention that a first numerical comparative result for solving a simple shape optimization problem of minimizing only a least-square cost functional of the velocity, solution of the state Stokes problem, was stated in a short communication [6]. We deal here with the numerical study of more complicated situations for different shape optimization problems of minimizing various cost functional constrained to Laplace or Stokes Operator. For this, we provide a detailed description of the new approach and investigate the numerical shape optimization algorithms using different shape derivative approaches performed by the finite element method for approximating the auxiliary boundary value problems in the shape optimization processes. Finally, we give some numerical experiments including comparison results which confirm our expectation. Mainly the proposed approach using the new shape derivative formula converges to solutions of good quality and offer better accuracy for the optimal shape as well as the associated reconstructed solution in less CPU time compared to the one using the classical approach.

The remainder of this paper is organized as follows: in Sect. 2, we present a brief survey on the classical shape sensitivity analysis based the Hadamard approaches and summarized its resulting numerical shape optimization process. We propose the new numerical shape optimization approach based on the shape derivative formula involving the Minkowski deformation and present its numerical algorithm. Then, we propose the discretization of this shape optimization approach using the finite element method and suggest the discrete numerical optimization process. The Sect. 3 is devoted to the application of both approaches for solving some shape optimization problems of minimizing some appropriate cost functionals subjected to some elliptic boundary value problems governed by Laplace or Stokes equation. Notably, we give some numerical experiments including comparison results showing the efficiency of the new shape optimization approach compared to the classical one.

2 Shape sensitivity approaches and resulting algorithms

The numerical investigation of shape optimization problems is based on the study of the first variation of a shape functional, and in particular, on the computation of its gradient with respect to domains. So in this section, we present a short survey on the notion of differentiability with respect to domains or the so-called the shape derivative. The first part is devoted to introduce the Hadamard's shape derivative approaches extensively studied in the literature (see, for example, [1, 14, 24, 30, 38]) and summarized the resulting numerical algorithm for solving shape optimization problems. Then, we present the new shape derivative approach introduced in [2, 3] for convex domains and suggest its resulting numerical shape optimization process.

In all what follows, let us consider a typical shape optimization problem:

$$\min_{\Omega\in\mathcal{U}} \mathcal{F}(\Omega, u_\Omega, \nabla u_\Omega) \tag{2.1}$$

where $\mathcal{F}$ is a shape cost functional and $\mathcal{U}$ is a family of admissible open smooth enough subsets of D, a large and a smooth set of $\mathbb{R}^d$ ($d \geq 2$), and u_Ω is the solution of boundary value problem on Ω called the state problem.

2.1 Hadamard's shape sensitivity based approaches

In order to carry out the sensitivity analysis of shape cost functionals $\Omega \to J(\Omega)$, one needs to introduce a family of perturbations $\{\Omega_\epsilon\}_\epsilon$ of a given domain $\Omega_0 \subset \mathbb{R}^d$ for $0 \leq \epsilon < 1$. Thereby one can construct a family of transformations $T_\epsilon : \mathbb{R}^d \to \mathbb{R}^d$ for some $\epsilon \in [0, 1[$ and T_ϵ maps Ω onto Ω_ϵ. The family of domains $\{\Omega_\epsilon\}_\epsilon$ is then defined by $\Omega_\epsilon = T_\epsilon(\Omega)$, such that T_ϵ satisfies appropriate regularity assumptions. More precisely, in the case of enough smooth domains, one can use the transformation of domains introduced by Hadamard in 1907 in his famous memory on elastic plates [18] to compute the derivative of a shape functional $\Omega \to J(\Omega)$ by considering the normal perturbations of the boundary. This is the basic idea in the notion of shape derivative in a topological context developed later by several authors. This approach can be briefly described as follows, if Ω_0 is domain of class $\mathcal{C}^\infty$, its outward unit normal vector ν_0 on $\Gamma_0 := \partial\Omega_0$ is in $\mathcal{C}(\Gamma_0, \mathbb{R}^d)$. Let $g \in \mathcal{C}^\infty(\Gamma_0)$ be a given function, since Γ_0 is assumed to be compact, then there exists $\delta > 0$ such that for any $\epsilon \in]-\delta, \delta[$, we have that

$$\Gamma_\epsilon = \Gamma + \epsilon g\nu = \{y \mid y = x + \epsilon g(x)\nu_0(x) \text{ for } x \in \Gamma_0\}$$

is the boundary of the domain Ω_ϵ which is of class $\mathcal{C}^\infty$. Consider an extension $\mathcal{N}_0$ to $\mathbb{R}^d$ of the normal vector field ν_0 defined on Γ_0, $\mathcal{N}_0 \in \mathcal{C}^\infty(\mathbb{R}^d, \mathbb{R}^d)$, we can define the transformation $T_\epsilon = \mathrm{Id}_{\mathbb{R}^d} + \epsilon g_0\mathcal{N}_0$, where g_0 denotes an extension in $\mathcal{C}^\infty(\mathbb{R}^d)$ of $g \in \mathcal{C}^\infty(\Gamma)$.

Then, this approach was widely developed by many authors, see, for example, [14, 24, 30, 31, 38] and the references therein. Notably, there are two variants of the Hadamard's method of shape differentiation. We will present here the more extensively used:

The first one is introduced by Murat and Simon [31, 36, 37] and based on the variation of domains by the transformation $T_\theta := \mathrm{Id}_{\mathbb{R}^d} + \theta$, such that

$$\Omega_\theta := T_\theta(\Omega_0) = \{x + \theta(x) \mid x \in \Omega_0\}, \tag{2.2}$$

where Ω_0 is an element of a family of domains $\mathcal{U}$ and $\theta \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$.

So assume that there exists $\epsilon \in]0, 1[$ such that $\Omega_\theta \in \mathcal{U}\ \ \forall\theta \in B_{W^{1,\infty}}(0, \epsilon)$, where $B_{W^{1,\infty}}(0, \epsilon)$ is the ball of center 0 and radius ϵ in $W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$. Then, the shape derivative of a cost functional $\Omega \to J(\Omega)$ at Ω_0 is defined as the Fréchet derivative at 0 in $W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$ of $\theta \in B_{W^{1,\infty}}(0, \epsilon) \mapsto J((\mathrm{Id}_{\mathbb{R}^d} + \theta)(\Omega_0)) \in \mathbb{R}$.

Note that when $\theta \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$, such that $\|\theta\|_{W^{1,\infty}} \le \epsilon$ (for $\epsilon \in [0, 1[$), the function $\mathrm{Id}_{\mathbb{R}^d} + \theta$ called the perturbation of the identity [13] is a diffeomorphism, which is the main fact used for the existence of the shape derivative with respect to domains.

The second approach is the so-called speed method, introduced by Zolésio et al. [15, 16, 40, 41], and summarized as follows: for a given vector field $V \in \mathcal{C}^1(\mathbb{R} \times \mathbb{R}^d; \mathbb{R}^d)$, consider the solution of the following ordinary differential equation

$$\Phi_V(0, x) = x \quad \text{and} \quad \frac{d\Phi_V(\epsilon, x)}{d\epsilon} = V(\epsilon, \Phi_V(\epsilon, x)), \quad x \in \Omega_0. \tag{2.3}$$

and define the domain

$$\Omega_\epsilon = T_\epsilon(V)(\Omega_0) = \{T_\epsilon(V)(x) \mid x \in \Omega_0\}, \quad T_\epsilon(V)(\Omega_0) := \Phi_V(\epsilon, \Omega_0).$$

Then,

(i) the Eulerian derivative of $J(\Omega)$ at Ω_0:

$$dJ(\Omega_0)(V) := \lim_{\epsilon \to 0} \frac{J(T_\epsilon(V)(\Omega_0)) - J(\Omega_0)}{\epsilon} \tag{2.4}$$

exists for all $V \in \mathcal{C}(]0, \epsilon[, \mathcal{C}^k(D, \mathbb{R}^d))$ $(k \in \mathbb{N})$ and for $|\epsilon| < 1$.

(ii) the mapping $V \to dJ(\Omega_0, V)$ is linear and continuous from $\mathcal{C}(]0, \epsilon[, \mathcal{C}^k(D, \mathbb{R}^d))$ into $\mathbb{R}$.

Note that, using the Taylor expansions for $T_\epsilon(V)$ with respect to ϵ, we can write:

$$\begin{aligned} T_\epsilon(V) = \Phi_V(\epsilon, x) &= \Phi_V(0, x) + \epsilon \left.\frac{d\Phi_V(\epsilon, x)}{d\epsilon}\right|_{\epsilon=0} + \epsilon^2 \left.\frac{d^2\Phi_V(\epsilon, x)}{d\epsilon^2}\right|_{\epsilon=0} \\ &= x + \epsilon V(0, x) + \frac{\epsilon^2}{2} \nabla V(0, x) V(0, x) + o(\epsilon^2). \end{aligned}$$

So $T_\epsilon(V)$ can be approximated by the diffeomorphism $\mathrm{Id}_{\mathbb{R}^d} + \epsilon V(0, .)$ on a neighborhood of x, which is the frequently used deformation in the literature for the variation of domains in the shape derivative approach [1, 7, 14, 24, 38].

2.1.1 Identification shape optimization process

In order to introduce a gradient method for minimizing the shape optimization problem (3.1), let us consider here the frequent variations of a given domain $\Omega_0 \in \mathcal{U}$ defined by the transformation $\epsilon \to \mathrm{Id}_{\mathbb{R}^d} + \epsilon V$ where $V : \mathbb{R}^d \to \mathbb{R}^d$ is a smooth vector field and $\mathrm{Id}_{\mathbb{R}^d}$ is the identity mapping in $\mathbb{R}^d$, such that $\Omega_\epsilon = (\mathrm{Id}_{\mathbb{R}^d} + \epsilon V)(\Omega_0)$, $t \in [0, 1[$. So, let us assume that the functional $\epsilon \to \mathcal{F}(\Omega_\epsilon, u_{\Omega_\epsilon}, \nabla u_{\Omega_\epsilon})$ is differentiable at $\epsilon = 0$. Then, according to the fundamental result called "structure theorem" established by Hadamard and rigorously considered later by Zolésio [14], which plays a crucial role in shape optimization tools both from numerical and theoretical point of view, one can write the shape derivative of $\mathcal{F}$ in the direction of V, denoted by $\delta\mathcal{F}(\Omega_0)[V]$, as follows:

$$\delta\mathcal{F}(\Omega_0)[V] := \int_{\partial\Omega_0} g_0(u_{\Omega_0}, \psi_{\Omega_0})\langle V, \nu_0\rangle d\sigma \tag{2.5}$$

where $g_0(\cdot, \cdot)$ is an integrable function on $\partial\Omega_0$, ψ_{Ω_0} is the solution of an appropriate adjoint state problem, ν_0 denotes the outward unit normal vector to $\partial\Omega_0$ and $\langle ., .\rangle$ designates the inner product in $\mathbb{R}^d$.

This formula known as the canonical form in the shape optimization theory provides a descent direction for the gradient method which can be given by the vector field $V = -g_0\nu_0$. Then, we can update the shape Ω_0 by $\Omega_\epsilon = (\mathrm{Id}_{\mathbb{R}^d} + \epsilon V)(\Omega)$. So we get

$$\mathcal{F}(\Omega_\epsilon, u_{\Omega_\epsilon}, \nabla u_{\Omega_\epsilon}) = \mathcal{F}(\Omega_0, u_{\Omega_0}, \nabla u_{\Omega_0}) - \epsilon \int_{\partial\Omega_0} g_0^2 \langle \nu_0, \nu_0\rangle \, d\sigma + o(\epsilon^2). \tag{2.6}$$

This ensures the decrease of the cost functional. However, the vector field V is defined only on the boundary $\partial\Omega_0$ and must be extended to the whole domain Ω_0. So we can extend it accordingly to the structure theorem [27], as the unique solution in $[H^1(\Omega_0)]^d$ of the following variational problem:

$$\int_{\Omega_0} \nabla V : \nabla\Psi dx + \int_{\Omega_0} V \cdot \Psi dx = -\delta\mathcal{F}(\Omega_0)[\Psi], \ \forall\Psi = (\Psi^{(1)}, ..., \Psi^{(d)}) \in [H^1(\Omega_0)]^d \tag{2.7}$$

where $\nabla V : \nabla\Psi = \sum_{k,j=1}^{d} \frac{\partial V^{(k)}}{\partial x_j}\frac{\partial \Psi_i^{(k)}}{\partial x_j}$ with $V = (V^{(1)}, ..., V^{(d)})$.

In fact, we can find more details about the choice of the descent direction for example in [1]. We note also that for the choice of the step size ρ, for the gradient method, one can opt for the approach inspired from the Armijo-Goldstein strategy in [26] as follows:

$$\rho = \alpha \frac{\mathcal{F}(\Omega, u_\Omega, \nabla u_\Omega)}{\|V\|^2_{H^1(\Omega)}}, \ \text{for some } \alpha \in]0, 1[. \tag{2.8}$$

Thereby, we have

$$\begin{aligned}\mathcal{F}(\Omega_\rho, u_{\Omega_\rho}, \nabla u_{\Omega_\rho}) &\simeq \mathcal{F}(\Omega_0, u_{\Omega_0}, \nabla u_{\Omega_0}) + \rho\delta\mathcal{F}(\Omega_0)[V] + O(\rho^2)\\ &= \mathcal{F}(\Omega_0, u_{\Omega_0}, \nabla u_{\Omega_0}) + \alpha\frac{\mathcal{F}(\Omega_0, u_{\Omega_0}, \nabla u_{\Omega_0})}{\|V\|^2_{H^1(\Omega)}}\delta\mathcal{F}(\Omega_0)[V] + O(\rho^2)\\ &= (1-\alpha)\mathcal{F}(\Omega_0, u_{\Omega_0}, \nabla u_{\Omega_0}) + O(\rho^2).\end{aligned}$$

Then, the vector fields ρV defines a descent direction for $\alpha \in]0, 1[$. Moreover, we note that when the step size parameter is such that $\alpha > 1$, the vector fields ρV still define a descent direction. This last choice of the parameters α is illustrated in some numerical tests in the numerical results section.

Hence, the numerical gradient algorithm for solving the shape optimization problem (2.1) is summarized as follows:

Algorithm 1 Numerical optimization algorithm based on the classical formula.

```
1. Choose initial domain $\Omega_0 \in \mathcal{U}$ and a precision Eps.
2. At step $k \geq 0$, calculate $u_k$ and $\psi_k$ the solution of state and adjoint problem in $\Omega_k$ respectively.
4. Compute $g_k = g_k(u_k, \psi_k)$ on $\Gamma_k := \partial\Omega_k$.
3. Compute the direction $V_k$ solution of (2.7) in $\Omega_k$.
4. Update the domain $\Omega_{k+1} = (\mathrm{Id}_{\mathbb{R}^d} + \rho V_k)(\Omega_k)$, with optimal step size $\rho_k$ given in (2.8).
5.Adapt-mesh.
6. If $|J(\Omega_k)| \leq$ Eps, **Return** $\Omega_k$; Else, **Back to previous step 2.**
```

2.2 Shape derivative via minkowski deformation

The shape derivative approach presented in the previous section presents some difficulties from both theoretical and numerical point of view. For example, when one wants to connect the set of admissible domains with vector fields, one has to suppose high smoothness conditions on the initial data in order to differentiate functions depending on the domain and we have to extend the vector fields (obtained only on the boundary) to the whole domain or to re-mesh the domain, at each iteration, and both tasks are expensive. In order to avoid a part of the above issues, we define and use another way of variation of domains, a way that is linked to the convexity context and based on the Minkowski sum. This allows to define a novel concept of computing the derivative with respect to domains using the so called support functions in convex analysis.

In order to be more precise, let D be a fixed smooth convex and bounded open subset of $\mathbb{R}^d$, let us consider the set of admissible domains $\mathcal{U}$ given by

$$\mathcal{U} = \{\Omega \subset D \ / \ \Omega \text{ is open, convex and of class } C^2\}$$

and let us recall that a support function P_Ω of a bounded convex domain Ω is given by

$$P_\Omega(x) = \sup_{y\in\Omega}\langle x, y\rangle = \sup_{y\in\overline{\Omega}}\langle x, y\rangle\,,$$

where $\langle x, y\rangle$ denotes the standard scalar product of x and y in $\mathbb{R}^d$.

So, let us define the shape derivative approach based on the variation of domains using the Minkowski sum. Let $\Omega_0 \in \mathcal{U}$ and Ω be a convex domain. The deformed domain denoted by Ω_ϵ is given by the Minkowski sum as follows:

$$\Omega_\epsilon = \Omega_0 + \epsilon\Omega := \{x + \epsilon y \mid x \in \Omega_0,\ \ y \in \Omega\},\ \ \epsilon \in [0, 1].$$

Consider a real-valued shape function $J : \Lambda \in \mathcal{U} \longmapsto J(\Lambda) \in \mathbb{R}$ defined on a family $\mathcal{U}$ of subsets of $\mathbb{R}^d$. The shape functional J is called shape differentiable at Ω_0 in the direction of Ω, if the eulerian derivative

$$\delta J(\Omega_0)[\Omega] := \lim_{\epsilon\to 0^+} \frac{J(\Omega_\epsilon) - J(\Omega_0)}{\epsilon}.$$

exists for all convex domain Ω. Then, the expression $\delta J(\Omega_0)[\Omega]$ is called the shape derivative of J at Ω_0 in the direction of Ω.

In this context, let us precise that the shape derivative of a volume functional J defined on $\mathcal{U}$ by

$$\Lambda \to J(\Lambda) = \int_\Lambda f dx, \quad \text{where} \quad f \quad \text{is a fixed function defined in } \mathbb{R}^d,$$

using a convex deformation of kind:

$$(1-\epsilon)\Omega_0 + \epsilon\Omega, \qquad \text{for } \Omega_0, \Omega \in \mathcal{U} \text{ and } \epsilon \in [0, 1]$$

was first established by A. A. Niftiyev and Y. Gasimov [32], for the function f is of class C^1. More precisely they show that

$$\delta J(\Omega_0)[\Omega] := \lim_{\epsilon\to 0^+} \frac{J((1-\epsilon)\Omega_0 + \epsilon\Omega) - J(\Omega_0)}{\epsilon} = \int_{\partial\Omega_0} f(x) \left(P_\Omega(\nu_0(x)) - P_{\Omega_0}(\nu_0(x))\right) d\sigma(x), \tag{2.9}$$

where $\nu_0(x)$ denotes the outward unit normal vector to $\partial\Omega_0$ at x, and P_{Ω_0}, P_Ω are the support functions of the domains Ω_0, Ω, respectively.

Recently, A. Boulkhemair and A. Chakib [3] extended this formula to the case where f is in the space $W^{1,1}_{loc}(\mathbb{R}^n)$. Then, based on the Brunn-Minkowski theory (see, for example, R. Schneider, [35]), they also proposed a similar shape derivative formula to (2.9) by considering the Minkowski deformation

$$\Omega_0 + \epsilon\Omega, \qquad \text{for } \Omega_0, \Omega \in \mathcal{U}, \text{ and } \epsilon \in [0, 1],$$

that is,

$$\lim_{\epsilon\to 0^+} \frac{J(\Omega_0 + \epsilon\Omega) - J(\Omega_0)}{\epsilon} = \int_{\partial\Omega_0} f(x)\, P_\Omega(\nu_0(x))\, d\sigma(x), \tag{2.10}$$

where $\nu_0(x)$ denotes the outward unit normal vector to $\partial\Omega_0$ at x, and P_Ω is the support function of the domain Ω.

In fact, this formula holds true for bounded convex domains, see [2]. Moreover, according to this work, there is an equivalence between the formulas (2.9) and (2.10) of shape derivative with respect to respectively convex and Minkowski deformations. So depending on what it is needed, we use one of the two deformations and consequently investigate the associated shape derivative formula. In this context, we mention that numerical optimization processes based on the gradient method involving these shape derivative formulas are successfully applied for solving concrete problems in [4, 5], using boundary element method for approximating the auxiliary boundary value problems.

We note also that the above formulas allow us to express the shape derivative of the cost functional by means of support functions, and when solving numerically problems one gets a support function at each step of the implementation, then the domain can be recovered as the sub-differential of the support function. Thereby, during the numerical process, we get at each step support functions instead of domains. From this fact, we believe that, in the context of convexity and the numerical implementation, the use of these formulas involving the support functions is more advantageous than the classical shape derivative formula expressed in term of vectors fields. So the aim of this paper is to prove numerically these expectations by a comparative numerical study between the two approaches through the resolution of some shape optimization problems constrained to boundary value problems governed by elliptic operators.

2.2.1 New numerical optimization process involving support function

In this section, we will describe the numerical process adopted to solve numerically the problem (3.1), using the shape derivative formulas (2.10) and (2.9). Let us first recall that when one assumes that Ω is strongly convex, according to [3], the domains $\Lambda_\epsilon = \Omega_0 + \epsilon\Omega$ for small enough ϵ can be considered as deformations of the domain Ω_0 by the following explicit vector field V defined by $V(0) = 0$ and

$$V(x) = J_{\Omega_0}(x)\, \nabla P_\Omega\left(\nu_0\left(\frac{x}{J_{\Omega_0}(x)}\right)\right), \quad x \in \mathbb{R}^d, x \neq 0 \tag{2.11}$$

where P_Ω is the support function of Ω, ν_0 is the outward unit normal vector to $\partial\Omega_0$ and J_{Ω_0} is the gauge function associated to Ω which is defined by $J_{\Omega_0}(x) = \inf\{\lambda; \lambda > 0,\ x \in \lambda\Omega_0\}$. Thereby $\Lambda_\epsilon = (\mathrm{Id}_{\mathbb{R}^d} + \epsilon V)(\Omega_0)$ for small enough ϵ. Furthermore, since Ω_0 and Ω are of class $\mathcal{C}^2$, then $V \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d) \cap \mathcal{C}^1$. Therefore, if $\epsilon \to F(\Omega_\epsilon, u_{\Lambda_\epsilon}, \nabla u_{\Lambda_\epsilon})$ is differentiable at 0^+, using the formula (2.5), we get

$$\delta\mathcal{F}(\Omega_0)[\Omega] := \langle g_0(u_{\Omega_0}, \psi_{\Omega_0}), \langle V, \nu_0\rangle\rangle_{L^2(\partial\Omega_0)},$$

where $g_0(u_{\Omega_0}, \psi_{\Omega_0})$ is an integrable function on $\partial\Omega_0$, ψ_{Ω_0} is the solution of an appropriate adjoint state problem. It follows from Lemma 4.1 of [3] that $\langle V, \nu_0\rangle = P_\Omega(\nu_0)$ on $\partial\Omega_0$, so that

$$\delta\mathcal{F}(\Omega_0)[\Omega] = \langle g_0(u_{\Omega_0}, \psi_{\Omega_0}), P_\Omega(\nu_0)\rangle_{L^2(\partial\Omega_0)}. \tag{2.12}$$

So the numerical optimization algorithm for solving the shape optimal design problem (3.1), based on the shape derivative formula (2.12) is summarized in the following algorithm.

Algorithm 2 Numerical optimization algorithm involving support function.

```
1. Choose initial domain $\Omega_0 \in \mathcal{U}$, fix step size $\rho \in ]0,1[$ and a precision Eps.
2. At step $k \geq 0$, calculate $u_k$ and $\psi_k$ the solution of the state and adjoint state problem in $\Omega_k$ respectively.
4. Compute $g_k = g_k(u_k, \psi_k)$ on $\Gamma_k := \partial\Omega_k$.
5. Compute $\widehat{P_k}$ the solution of
       $$\arg\min_{\varphi\in\mathcal{P}} \ \delta\mathcal{F}_k(\varphi) \qquad (2.13)$$
   where
       $$\delta\mathcal{F}_k(\varphi) := \int_{\Gamma_k} g_k(x)\varphi(\nu_k(x))\,ds$$
   and $\mathcal{P} = \{\Phi \in \mathcal{C}(\overline{D}) \ / \ \Phi$ is convex and homogeneous of degree 1 and $0 \leq \Phi \leq P_D\}$
5. Update the domain $\Omega_{k+1} = \Omega_k + \rho\widetilde{\Omega}_k$, where $\widetilde{\Omega}_k$ is associated to $\widehat{P_k}$ by its sub-differential:
       $$\widetilde{\Omega}_k = \partial\widehat{P_k}\,(0) = \left\{l \in \mathbb{R}^2 \ / \ \widehat{P_k}\,(x) \geq \langle l, x\rangle, \ \forall x \in \mathbb{R}^2\right\}.$$
6. Adapt-mesh.
7. If $\|\mathcal{F}(\Omega_k, u_k, \nabla u_k)\| \leq$ Eps, **Return** $\Omega_k$; Else, **Back to previous step 2.**
```

We note that the problem (2.13) admits a solution $\widehat{p} \in \mathcal{P}$. Since the functional $j : p \in \mathcal{P} \to j(p) = \langle g_{\partial\Omega_0}, p \circ \nu_0\rangle_{L^2(\partial\Omega)}$ is continuous on a compact set $\mathcal{P}$ of $\mathcal{C}(\overline{D})$. Indeed, let $p \in \mathcal{P}$, so it is the support function of a unique convex bounded open set which is its sub-differential at 0, that is, $p = P_{\partial p(0)}$ (see, for example, [33, 39]). So, for all $x, y \in \overline{D}$, using the fact that a support function is sub-linear and homogenous of degree 1 and $p \leq P_D$, we get

$$\begin{aligned}
\forall x, y \in \overline{D}, \ \forall p \in \mathcal{P} \ |p(x) - p(y)| &= |P_{\partial p(0)}(x) - P_{\partial p(0)}(y)| \\
&\leq \sup_{w\in\mathbb{S}^{n-1}} P_{\partial p(0)}(w)\|x - y\| \\
&\leq \sup_{w\in\mathbb{S}^{n-1}} P_D(w)\|x - y\|.
\end{aligned}$$

which implies that the family $\mathcal{P}$ is equicontinuous. On the other hand, it is clear that j is linear, and moreover, it is continuous since:

$$\forall p, q \in \mathcal{P} \ \ |j(p)| \leq \|g_{\partial\Omega_0}\|_{L^1(\partial\Omega_0)}\|p\|_{\infty,D}.$$

So it follows from the Ascoli-Arzela theorem's that $\mathcal{P}$ is relatively compact in $\mathcal{C}(\overline{D})$. Then, it is easy to check that it is closed in $(\mathcal{C}(\overline{D}), \|\cdot\|_{\infty,D})$ and then it is compact in $(\mathcal{C}(\overline{D}), \|\cdot\|_{\infty,D})$.

We note also that, the shape derivative of a general class of shape functionals $J(\Omega)$ in direction of a vector field V has the generic form:

$$\delta J(\Omega)[V] = \int_{\partial\Omega} g\langle V(x), \nu(x)\rangle d\sigma(x) =: \langle g_{|\Gamma}, \langle V(x), \nu(x)\rangle\rangle_{L^2(\partial\Omega)}. \tag{2.14}$$

where the scalar function $g : \partial\Omega \to \mathbb{R}$ is the shape gradient of J with respect to the $L^2(\partial\Omega)$ inner product. In the case of convex domains, using the shape derivative formulas (2.10) and (2.9), this formula becomes

$$\delta g(\Omega)[\Theta] := \int_{\partial\Omega} g_{\partial\Omega}(x) P_\Theta(\nu(x)) d\sigma(x) = \langle g_{\partial\Omega}, P_\Theta(\nu)\rangle_{L^2(\partial\Omega)}. \tag{2.15}$$

Here, $\delta J(\Omega)[\Theta]$ depends only on the normal component of P_Θ on the boundary $\partial\Omega$. This expression allows to easily to deduce the direction of descent, as it was summarized in the above algorithm. Indeed, the sequence of domains $(\Omega_k)_{k\in\mathbb{N}}$ must be constructed in such a way that $(J(\Omega_k))_{k\in\mathbb{N}}$ is decreasing. So let $k \in \mathbb{N}^*$, then for a small $\rho \in]0, 1[$, we have

$$J(\Omega_{k+1}) - J(\Omega_k) = J(\Omega_k + \rho\Theta_k) - J(\Omega_k) = \rho\left(\int_{\partial\Omega_k} g_k P_{\Theta_k} \circ \nu_k d\sigma\right) + O(\rho^2).$$

Thus, if we take $P_{\Theta_k} = \widehat{P}_k$ the solution of $\arg\min\limits_{p\in\mathcal{E}} \Lambda_k(p)$, we get

$$\Lambda_k(\widehat{P}_k) = \int_{\partial\Omega_k} g_k P_{\Theta_k} \circ \nu_k d\sigma \leq \Lambda_k(0) = 0.$$

This ensures the decrease of the objective functional J. Consequently $\widehat{\Omega}_k = \partial\widehat{P}_k(0)$ defines a descent direction for J.

2.3 Discretization of the proposed shape optimization process

In what follows, we suppose that $d = 2$. Let $\Omega_0 \in \mathcal{U}_{ad}$ and let $X = (x_k)_{k=1}^N$ be a partition of $\Gamma_0 := \partial\Omega_0$, such that N is the number of the nodes located at the boundary Γ_0 and $x_k = (x_k^{(1)}, x_k^{(2)})$. Let us also consider the family $\zeta = (\xi_k)_{k=1}^N$, with $\xi_k = (\xi_k^{(1)}, \xi_k^{(2)})$ is the centroid of the boundary elements $C_i = [x_i, x_{i+1}]$ which is also a partition of Γ_0. Let us also denote by $\mathcal{T}_h$ a regular mesh of Ω_0 (see [12]) and by N_{mn} the number of the freedom degrees generated from the mesh of the domain limited by the N boundary elements.

So the discrete approximation of the gradient of $\mathcal{F}$, $\delta\mathcal{F}(\Omega_0)[\Omega]$ reads:

$$\delta\mathcal{F}_0(P) = \sum_{k=1}^{N-1} \int_{x_k}^{x_{k+1}} g\, P(\nu) = \sum_{k=1}^{N} \ell_k\, g_k\, P_k \tag{2.16}$$

where $\ell_k = \|x_{k+1} - x_k\|$; $P = (P_k = P(\nu_k)_{k=1}^N)$; $\nu_k = \nu(\xi_k)$ is the outward unit normal vector to the boundary elements $[x_k, x_{k+1}]$ that can be computed explicitly by the relation $\nu(\xi_k) = ((x_{k+1}^{(2)} - x_k^{(2)})/\ell_k, (-x_{k+1}^{(1)} + x_k^{(1)})/\ell_k)$, $g = g(u_0, \psi_0)$ is the shape gradient which depends on u_0 the solution of the state problem and ψ_0 the solution of an appropriate adjoint state problem and g_k is the approximate value of the shape gradient function g at the node x_k, using finite element discretization of the state and adjoint state problems.

Then, the next step is to deal with the discretization of the problem (2.13). For this purpose based on the polygonal shape of the discretized boundary $\Gamma = \cup_{k=1}^{N-1}[x_k, x_{k+1}]$, we derive some equations using the fact that the support function is sub-linear and homogeneous of degree 1. So, let $B(0, R)$ be a large ball which contains D, the space of the admissible support function is discretized as follows [5]:

$$\mathcal{P} = \left\{ P = (P_1, \cdots, P_N) \in \mathbb{R}^N / P \quad \text{satisfies equations (2.17) and (2.18)} \right\}$$

where

$$r = r\|\nu(\xi_k)\| = P_{B(0,r)}(\nu(\xi_k)) \leq P_k \leq P_D(\nu(\xi_k)) \leq P_{B(0,R)}(\nu(\xi_k)) = R\|\nu(\xi_k)\| = R \quad \text{for} \quad k = 1, \ldots, N. \tag{2.17}$$

and

$$\begin{cases} P_1 \leq (1 - \lambda_1)\, P_N + \lambda_1\, P_2, \\ P_k \leq (1 - \lambda_k)\, P_{k-1} + \lambda_k\, P_{k+1}, \text{ for } k = 2, \ldots, N-1 \\ P_N \leq (1 - \lambda_N)\, P_{N-1} + \lambda_N\, P_1 \end{cases} \tag{2.18}$$

with

$$\lambda_1 = \frac{\|\nu(\xi_1) - \nu(\xi_N)\|}{\|\nu(\xi_2) - \nu(\xi_N)\|}, \quad \lambda_i = \frac{\|\nu(\xi_i) - \nu(\xi_{i-1})\|}{\|\nu(\xi_{i+1}) - \nu(\xi_{i-1})\|} \quad \text{for } i = 2, \ldots, N-1;$$
$$\lambda_N = \frac{\|\nu(\xi_N) - \nu(\xi_{N-1})\|}{\|\nu(\xi_1) - \nu(\xi_{N-1})\|}.$$

The final step is to identify the boundary of the shape Ω_{k+1}, at each iteration $k \geq 0$, defined by the following: $\Omega_{k+1} = \Omega_k + \epsilon\, \partial \widehat{P_k}\,(0)$. For this, let $\widehat{P_k}$ be the solution of the problem (2.13) and let us consider its sub-differential at 0, $\partial \widehat{P_k}\,(0)$, which is a bounded convex domain [33]. Based on Lemma 4.7. of [3], for all $\delta > 0$, the domain $\partial \widehat{P_k}\,(0)$ can be approximated by strongly convex sub-domains denoted by $\Omega^{(k)}$, such that

$$d^H(\partial \widehat{P_k}\,(0)\,, \overline{\Omega^{(k)}}) \leq \delta. \tag{2.19}$$

On the other hand, according to [3], we can connect the two shapes Ω_k and $\Omega^{(k)}$, by an explicit vector fields as follows:

$$\Omega_k + \epsilon\, \Omega^{(k)} = (\mathrm{Id}_{\mathbb{R}^d} + \epsilon\, \Xi_k)(\Omega_k) \text{ and } \partial(\Omega_k + \epsilon\, \Omega^{(k)}) = (\mathrm{Id}_{\mathbb{R}^d} + \epsilon\, \Xi_k)(\partial \Omega_k)$$

where Ξ_k is the vector field defined by

$$\Xi_k(0) = 0 \text{ and } \Xi_k(x) = J_{\Omega_k}(x) \nabla P_{\Omega^{(k)}}\left(\nu_{\partial\Omega_k}\left(\frac{x}{J_{\Omega_k}(x)}\right)\right), \; x \in \mathbb{R}^2, x \neq 0$$

such that Ξ_k satisfies the following properties:

$$P_{\Omega^{(k)}}(\nu_{\partial\Omega_k}) = \langle \Xi_k, \nu_{\partial\Omega_k} \rangle \text{ and } \nu_{\partial\Omega^{(k)}}(\Xi_k) = \nu_{\partial\Omega_k}. \tag{2.20}$$

where $\nu_{\partial\Omega^{(k)}}$ and $\nu_{\partial\Omega_k}$ denotes the outward vectors normal to $\partial\Omega^{(k)}$ and $\partial\Omega_k$ respectively. Moreover, from equation (2.19), by using the properties of Hausdorff distance on convex domains (see, for example, [35, 39]), we get

$$d^H(\Omega_{k+1}, \Omega_k + \epsilon\, \Omega^{(k)}) = d^H(\Omega_k, \Omega_k) + \epsilon\, d^H(\partial \widehat{P}_k\,(0)\,, \overline{\Omega^{(k)}}) \leq \epsilon\delta.$$

So, we can approach Ω_{k+1} by the set $(\mathrm{Id}_{\mathbb{R}^d} + \epsilon\, \Xi_k)(\Omega_k)$.

Now, let us determine Γ_{k+1}, using the partition $\Gamma_k = \bigcup_{i=1}^{N-1} C_i$. This is equivalent to compute the set $\Xi_k(\Gamma_k)$. For this, we can use the equation (2.20) and the family $(P_i)_i$ (solution of the problem (2.13)) to get an approximation of the family $(\Xi_k(\xi_i) = \nabla P_{\Omega^{(k)}}(\nu_{\partial\Omega_k}(\xi_i)))_i$, or we can use the properties in (2.20) to get $(\Xi_k(\xi_i))_i$ as the solution of the following systems (see [5, 32]). So denote by $\Xi_k(\xi_i) = l_i = (l_i^{(1)}, l_i^{(2)})$, we have that $l_i - \xi_i$ is collinear to $\nu(\xi_i) = \nu_{\partial\Omega_k}(\xi_i) = (\nu^{(1)}(\xi_i), \nu^{(2)}(\xi_i))$, i.e.,

$$\nu^{(1)}(\xi_i)\, l_i^{(2)} - \nu^{(2)}(\xi_i)\, l_i^{(1)} = \nu^{(1)}(\xi_i)\, \xi_i^{(2)} - \nu^{(2)}(\xi_i)\, \xi_i^{(1)} \quad \text{for } i = 1, \ldots, N. \tag{2.21}$$

On the other hand, we have

$$\nu^{(1)}(\xi_i)\, l_i^{(1)} + \nu^{(2)}(\xi_i)\, l_i^{(2)} = \widehat{P}_i \quad \text{for } i = 1, \ldots, N. \tag{2.22}$$

We are now ready to provide a precise sketch of the discrete shape optimization algorithm accordingly to the previous considerations.

We note that the problem (2.23) can be read as follows:

$$\arg \min_{P=(P_i)_i} j_k(P) := \sum_{i=1}^{N} \ell_i\, \mathcal{G}_i\, P_i \tag{2.24}$$

$$\text{with } \ell_i = \|x_{i+1} - x_i\|, \quad i = 1, ..., N$$

$$\text{subject to} \tag{2.25}$$

$$\mathcal{A}P \leq \mathcal{B} := \mathcal{B} = (b_i)_{i=1}^{3N} \text{ where}$$

Algorithm 3 Discrete algorithmic description of the optimization process.

```
∘ For iteration k=0,...
1. Choose the initial data $\Gamma_k = \partial\Omega_k = \bigcup_{i=1}^{N-1} C_i$, $C_i = [x_i, x_{i+1}]$, $x_i = (x_i^{(1)}, x_i^{(2)})$ and Compute $(\xi_i)_{i=1}^N$, such that $\xi_i = (\xi_i^{(1)}, \xi_i^{(2)})$ is the centroid of the boundary elements $C_i$. Consider a mesh $\mathcal{T}_h^{(k)}$ of $\Omega_k$. Fix step size $\rho \in ]0,1[$ and a precision Eps.
2. Calculate $u_k$ and $\psi_k$ the approximate solutions of the state and adjoint state problems, which are of size $N_{mn}$.
3. Compute $g_i = g(u_k^i, \psi_k^i)$ the approximate values of the shape gradient $g(u_k, \psi_k)$ at $x_i$, for $i = 1, ..., N$.
4. Compute $\widehat{P}^{(k)} = (\widehat{P^{(k)}}_i)_{i=1}^N$ the solution of the following problem
```

$$\arg \min_{P=(P_i)_i} \delta\mathcal{F}_k(P) \tag{2.23}$$

$$\begin{aligned}
&\text{where } P \text{ subject to}\\
&P_1 - \lambda_1 P_2 + (\lambda_1 - 1) P_N \le 0\\
&P_i - \lambda_i P_{i+1} + (\lambda_i - 1) P_{i-1} \le 0, \quad \text{for } i = 2, \dots, N-1\\
&P_N - \lambda_N P_1 + (\lambda_N - 1) P_{N-1} \le 0\\
&-P_i \le -r, \ P_i \le R, \ i = 1, ..., N
\end{aligned}$$

```
    where
```

$$\delta\mathcal{F}_k(P) := \sum_{i=1}^N \ell_i \, g_i \, P_i$$

```
    with $\ell_i = \|x_{i+1} - x_i\|$, $i = 1, ..., N$ and
```

$$\lambda_1 = \frac{\|\nu(\xi_1) - \nu(\xi_N)\|}{\|\nu(\xi_2) - \nu(\xi_N)\|}, \quad \lambda_i = \frac{\|\nu(\xi_i) - \nu(\xi_{i-1})\|}{\|\nu(\xi_{i+1}) - \nu(\xi_{i-1})\|} \quad \text{for } i = 2, \dots, N-1; \quad \lambda_N = \frac{\|\nu(\xi_N) - \nu(\xi_{N-1})\|}{\|\nu(\xi_1) - \nu(\xi_{N-1})\|}.$$

```
5. if $\|\mathcal{F}(\Omega_k, u_k, \nabla u_k)\| \le$ Eps, Return $\Omega_k$.
6. Compute the boundary $\Gamma_{k+1} = \bigcup_{i=1}^{N-1}[(c_i^{(1)}, c_i^{(2)}), (c_{i+1}^{(1)}, c_{i+1}^{(2)})]$ of $\overline{\Omega}_{k+1}$ with
```

$$(c_i^{(1)}, c_i^{(2)}) = (1-\rho)\,(x_i^{(1)}, x_i^{(2)}) + \rho\,(l_i^{(1)}, l_i^{(2)}), \ \text{for } i = 1, \dots, N,$$

```
    where $(l_i^{(1)}, l_i^{(2)})_i$ are the solutions of the systems
```

$$\begin{cases} l_i^{(1)} = \nu^{(2)}(\xi_i)(x_i^{(1)}\nu^{(2)}(\xi_i) - x_i^{(2)}\nu^{(1)}(\xi_i)) + \nu^{(1)}(\xi_i)\widehat{P}_i^{(k)}, \\ l_i^{(2)} = \nu^{(1)}(\xi_i)(x_i^{(2)}\nu^{(1)}(\xi_i) - x_i^{(1)}\nu^{(2)}(\xi_i)) + \nu^{(2)}(\xi_i)\widehat{P}_i^{(k)}, \end{cases} \quad \text{for } i = 1, \dots, N.$$

$$\mathcal{A} = \begin{bmatrix} A \\ -\mathrm{I}_N \\ \mathrm{I}_N \end{bmatrix} \text{ where } A = \begin{bmatrix} 1 & -\lambda_1 & 0 & \dots & 0 & (\lambda_1 - 1) \\ (\lambda_2 - 1) & 1 & -\lambda_2 & \dots & 0 & 0 \\ 0 & (\lambda_3 - 1) & 1 & -\lambda_3 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ -\lambda_N & 0 & 0 & \dots & (\lambda_N - 1) & 1 \end{bmatrix};$$

$$b_i = \begin{cases} 0, & \text{for } i = 1, ..., N \\ -r, & \text{for } i = N+1, ..., 2N \\ R, & \text{for } i = 2N, ..., 3N \end{cases}$$

with

$$\lambda_1 = \frac{\|\nu(\xi_1) - \nu(\xi_N)\|}{\|\nu(\xi_2) - \nu(\xi_N)\|}, \quad \lambda_i = \frac{\|\nu(\xi_i) - \nu(\xi_{i-1})\|}{\|\nu(\xi_{i+1}) - \nu(\xi_{i-1})\|} \quad \text{for } i = 2, \ldots, N-1;$$
$$\lambda_N = \frac{\|\nu(\xi_N) - \nu(\xi_{N-1})\|}{\|\nu(\xi_1) - \nu(\xi_{N-1})\|}$$

and I_N denotes the matrix identity in $\mathbb{R}^{N\times N}$.

We note that this linear optimization problem can be solved using for example the simplex method.

3 Application for solving some classical shape optimization problems

In this section, we test the efficiency of the proposed approach through the resolution of some shape optimization problems. Especially, we propose a comparative numerical study between this proposed shape gradient approach and the classical one.

3.1 Statement of the shape optimization problems

The first considered shape optimization problems consist in minimizing a generic volume cost functional constrained to a Laplace-Dirichlet or Laplace-Neumann boundary value problems and the second one is a fluid mechanics shape optimal design problem which aims to minimize appropriate cost functional constrained to Stokes boundary value problem. So, let D be a large smooth domain in $\mathbb{R}^2$ and let us denote the set of admissible shapes by

$$\mathcal{U} = \{\Theta \subset D \ / \ \Theta \text{ is open}, \quad \Theta \in \mathcal{C}^2 \cap \mathcal{K}\},$$

where $\mathcal{K}$ denotes the set of all convex domains and $\mathcal{C}^2$ denotes the space of domains with boundaries of class C^2. We will solve numerically the following models:

Model 1: Shape optimization problems for classical elliptic operator

Let us consider the following shape optimization problem:

$$\min_{\Omega\in\mathcal{U}} \mathcal{F}(\Omega, u_\Omega, \nabla u_\Omega) \text{where} \mathcal{F}(\Omega, u_\Omega, \nabla u_\Omega) := \varpi\|u_\Omega - \varphi_1\|^2_{L^2(\Omega)} + \beta\|\nabla u_\Omega - \varphi_2\|^2_{L^2(\Omega)} \tag{3.1}$$

and u_Ω satisfies the following state problem:

$$-\mu\Delta u + \lambda u = f \quad \text{in} \quad \Omega, \tag{3.2}$$
$$\eta\partial_\nu u + \gamma u = g \quad \text{on} \quad \Gamma := \partial\Omega, \tag{3.3}$$

where φ_1, φ_2, f and g are given functions, ϖ, β, μ, η and γ are given constants that satisfy appropriate assumptions allowing the existence and the uniqueness of the solution of the boundary value problem (3.2)-(3.3) and $\partial_\nu u = \langle \nabla u|_\Gamma, \nu\rangle$, with ν is the outward unit normal vector to Γ and $\langle\cdot,\cdot\rangle$ denotes the scalar product in $\mathbb{R}^2$.

Model 2: Shape optimization problems in fluid dynamics

Let us consider a fluid mechanics shape optimization problem constrained to Stokes equation:

$$\min_{\Omega\in\mathcal{U}} \mathcal{S}(\Omega, w_\Omega, Dw_\Omega) \text{ where } \mathcal{S}(\Omega, w_\Omega, \nabla w_\Omega) := \rho \int_\Omega \|w_\Omega - \phi_1\|^2 dx + \sigma \int_\Omega \|\nabla w_\Omega - \phi_2\|^2 dx \quad (3.4)$$

and (w_Ω, p_Ω) is solution of the Stokes system:

$$\begin{cases} -\kappa\Delta w + \nabla p = h & \text{in} \quad \Omega \\ \operatorname{div} w = 0 & \text{in} \quad \Omega \\ w = 0 & \text{on} \quad \Gamma := \partial\Omega \end{cases} \quad (3.5)$$

where Ω is the domain occupied by a fluid, $\kappa > 0$ is the viscosity coefficient, $w = (w_\ell)_{\ell=1}^2$ represents the fluid velocity, p is the associated pressure, ϕ_1 is the target velocity, ϕ_2 is a given function and h is a source term. The norm $\|\nabla w\|$ is associated to the usual Frobenius inner product given by the following: $\nabla w : \nabla w = \displaystyle\sum_{k,\ell=1}^{2} \frac{\partial w_\ell}{\partial x_k}\frac{\partial w_\ell}{\partial x_k}$.

In the sequel, in order to test the validity and the efficiency of the proposed approach summarized in the Algorithm 2 involving the new shape derivative formula, we propose its numerical comparative study with the classical approach based on the Algorithm 1 in terms of computation time (time CPU) and the accuracy of the obtained optimal solutions by dealing with numerical resolution of the problem (3.1) constrained to (3.2)-(3.3) for different values of the constants ϖ, β, μ and γ. Then, we deal with the numerical approximation of the shape optimization problem in fluid dynamics (3.4), by minimizing the cost functional $\mathcal{S}$ for different values of ρ and σ, constrained to the Stokes problem (3.5).

3.2 Numerical results for shape optimization problems

The numerical algorithms are implemented using the programming software FreeFem++ [21], so in order to increase the mesh quality, notably its uniformness and regularity, the "adapt-mesh" tool can be used if necessary. The numerical experiments is done on a workstation with an Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz. For all the numerical test the target shape is taken as the disc of center 0 and radius 1 and the stopping criterion precision is chosen the same for the two algorithms. Also the considered initial domain can be chosen as follows:

- case 1: square $[-2, 2]^2$;
- case 2: disk $\{(x, y) | x = 3\cos(t), \quad y = 3\sin(t), t \in [0, 2\pi]\}$;
- case 3: ellipse $\{(x, y) | x = 2.5\ \cos(t), \quad y = 3\ \sin(t), t \in [0, 2\pi]\}$.

In order to perform a numerical comparative study between the two approaches based on the gradient method, let us first start by analyzing the sensitivity and the influence of the choice of the step size parameter on the convergence performance of these algorithms.

3.2.1 Influence of the step size on the shape optimization processes

In this section, we deal with the numerical analysis of the sensitivity of the two shape optimization processes with respect to the step size parameter through the numerical resolution of the shape optimization problem (3.1) of minimizing the cost functional $\mathcal{F}$ with $\varpi = \frac{1}{2}$ and $\beta = 0$:

$$\mathcal{F}(\Omega, u_\Omega, \nabla u_\Omega) = \frac{1}{2} \|u_\Omega - u_{ex}\|^2_{L^2(\Omega)}, \tag{3.6}$$

where u_Ω is the solution of the Dirichlet boundary value state problem associated to the parameters $\mu = 1$, $\lambda = 1$, $\eta = 0$ and $\gamma = 1$. The shape gradient of this functional at $\Omega_0 \in \mathcal{U}$, for the two shape derivative formulas, is given by the following:

$$\left(|u_{\Omega_0} - u_{ex}|^2 + \frac{\partial u_{\Omega_0}}{\partial \nu_0} \frac{\partial \psi_0}{\partial \nu_0} \right)\Bigg|_{\partial \Omega_0}, \tag{3.7}$$

where ψ_0 is the solution of the adjoint state problem:

$$-\Delta \psi_0 + \psi_0 = (u_{\Omega_0} - u_{ex}) \text{ in } \Omega_0, \quad \psi_0 = 0 \text{ on } \partial\Omega_0 \tag{3.8}$$

The aim of the following numerical tests is to reconstruct the target shape and the exact solution of the state problem, which is taken to be $u_{ex} = x^2 + y^2 - 1$, using both

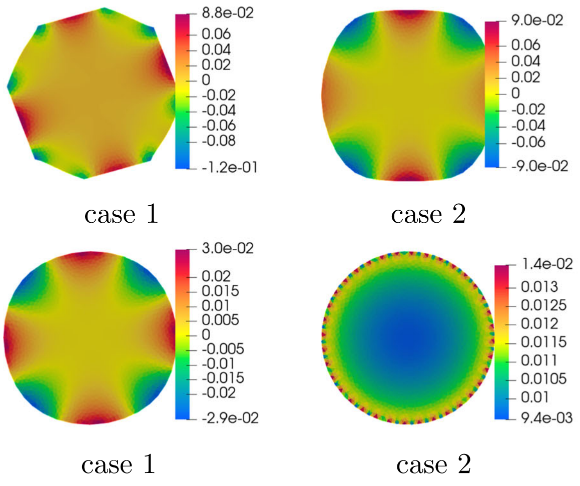
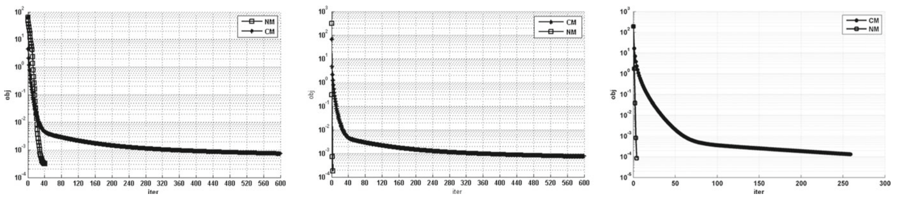


Fig. 1 The computed error of the solution in optimal domain using Algorithm 1 (above) and Algorithm 2 (below) for different cases



case 1 case 2 case 3

Fig. 2 Comparison of the convergence histories of the objective functional using Algorithms 1 and 2 for different cases

algorithms for different initial shapes. Thereby, we propose first to compare the two algorithms using a fixed optimal steps size (the best ones allowing a rapid convergence of the algorithms). Then, we compare the Algorithm 2 (Algorithm of the new method "NM") using a fixed optimal step size and the Algorithm 1 (Algorithm of the classical method "CM") performed with updated step size by Armijo-Goldstein strategy (2.8), for different values of α.

Shape optimization processes with fixed optimal step-size

The implementation of the two algorithms is done using the number of elements $N = 36$ and by choosing fixed step size parameters allowing rapid convergence of the two algorithms. In this respect, we note that these parameters are chosen in an optimal way, such that, if we take greater values of these parameters than the chosen ones, the two algorithms diverges. So the optimal step size parameters used respectively for both algorithms and for different initial shapes are taken such that, in case 1 (ρ=0.1 for Algorithm 2 and ρ=0.052 for Algorithm 1), in case 2 (ρ=0.9 for Algorithm 2 and ρ=0.043 for Algorithm 1) and in case 3 (ρ=0.79 for Algorithm 2 and ρ=0.034 for Algorithm 1). Thereby, in Fig. 1, we present the errors $(u - u_{\text{ex}})$ in the optimal domains obtained by the finite element discretization for different initial domains: case 1, case 2, and case 3. The variation of the objective or cost functionals with respect to the number of iterations are shown respectively in Fig. 2 for different cases. These results shows that the Algorithm 2 converges faster to more accurate optimal solutions than the Algorithm 1, for different initial shapes. This expectation is confirmed in Table 1, where we present the CPU-time, the final objective cost, the iteration numbers and the time reduction for both approaches. We observe that the proposed approach converges in less number of iterations compared to the classical one and the CPU execution time is reduced by at least **93%**. These numerical tests show the efficiency of the

Table 1 Numerical comparison between Algorithms 1 and 2 for different cases

	CM			NM			Time reduction
	CPU-time (s)	Iteration	Objective	CPU-time (s)	Iteration	Objective	
Case 1	**813.213**	600	7.45×10^{-4}	**10.03**	41	1.39×10^{-4}	**98,76%**
Case 2	**234.8**	163	7.68×10^{-4}	**6.37**	3	4.70×10^{-6}	**97.25%**
Case 3	**560.497**	259	1.76×10^{-4}	**8.16**	4	1.99×10^{-4}	**98.54%**

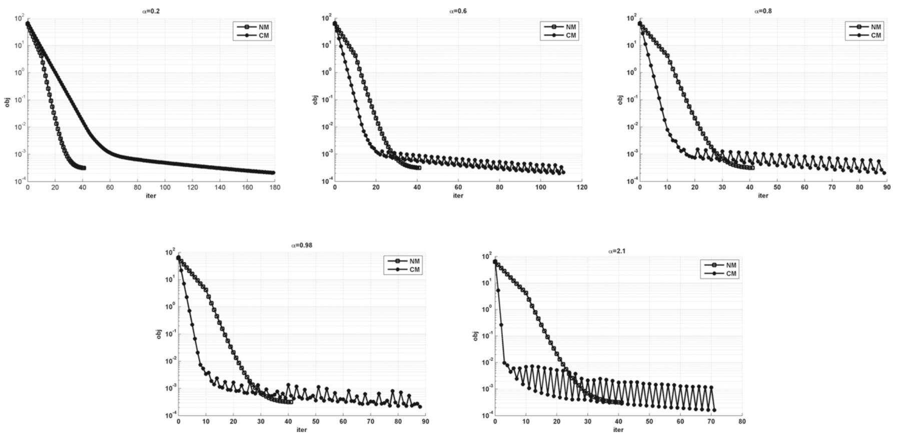


Fig. 3 Comparison of the convergence histories of the objective functional using Algorithms 1 and 2 for different values of α: case 1

Algorithm 2 in terms of CPU time execution and the accuracy of optimal solutions compared to Algorithm 1, when one use fixed optimal step size parameters.

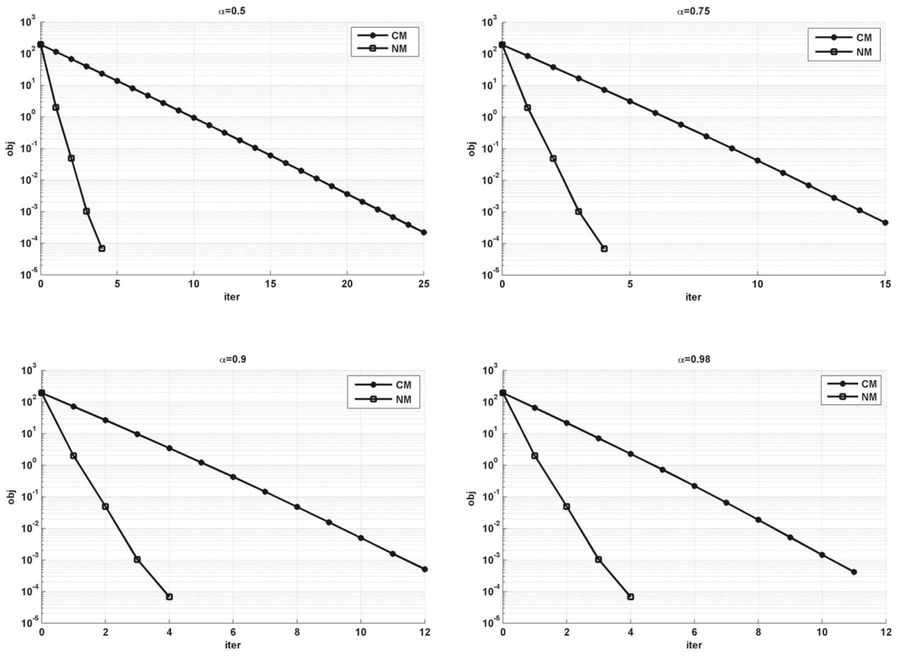


Fig. 4 Comparison of the convergence histories of the objective functional using Algorithms 1 and 2 for different values of α: case 3

Table 2 Numerical comparison between Algorithms 1 and 2 for different values of α: case 1

	CM			Time reduction
	CPU-time (s)	Iteration	Objective	
$\alpha = 0.2$	**229.80**	179	2.11×10^{-4}	**90.84%**
$\alpha = 0.6$	**137.22**	111	2.15×10^{-4}	**84.67%**
$\alpha = 0.8$	**108.97**	89	2.05×10^{-4}	**80.70%**
$\alpha = 0.98$	**106.17**	88	2.13×10^{-4}	**80.19%**
$\alpha_{\text{opt}} = 2.1$	**40.35**	71	1.62×10^{-4}	**47.88%**

In the sequel, in order to improve the performance of the Algorithm 1, we propose to update the step size parameter using the Armijo-Goldstein strategy (2.8), for different values of α.

Algorithm 2 with fixed step size and Algorithm 1 performed with Armijo-Goldstein step size

In this case, the number of elements is taken $N = 36$ and the Algorithm 2 is implemented using only a fixed optimal step size parameter, while the Algorithm 1 is performed using the Armijo-Goldstein strategy (2.8) for different values of α, including its optimal value denotes α_{opt} allowing a rapid convergence of this algorithm and chosen such that if we take greater value of it the algorithm diverges. Thereby, the variation of the objective with respect to the number of iterations for different values of α is presented in Figs. 3 and 4, respectively, for the case 1 and case 3. The CPU-time, the final objective functional, the iteration numbers and the time reduction for both approaches are illustrated in Tables 2 and 3, for the different initial shapes. We observe in this case also that the proposed approach converges in less number of iterations to accurate solutions compared to the classical one and the CPU execution time is reduced by at least (**90.84%** for case 1) and (**90.13%** for case 3), which decrease when the parameters α increase to reaches there optimal values, and the CPU time reduction in this case reaches (**47%** for case 1) and (**46%** for case 3). These numerical tests show the efficiency of the Algorithm 2, using only a fixed optimal step size parameter, in terms of CPU time execution and the accuracy of optimal solutions compared to Algorithm 1 performed with the Armijo-Goldstein strategy, for different values of α.

Table 3 Numerical comparison between Algorithms 1 and 2 for different values of α: case 3

	CM			Time reduction
	CPU-time (s)	Iteration	Objective	
$\alpha = 0.5$	**82.72**	25	2.21×10^{-4}	**90.13%**
$\alpha = 0.75$	**34.7**	15	4.57×10^{-4}	**76.48%**
$\alpha = 0.9$	**15.49**	12	5.10×10^{-4}	**47,32%**
$\alpha_{\text{opt}} = 0.98$	**15.17**	11	4.16×10^{-4}	**46.20%**

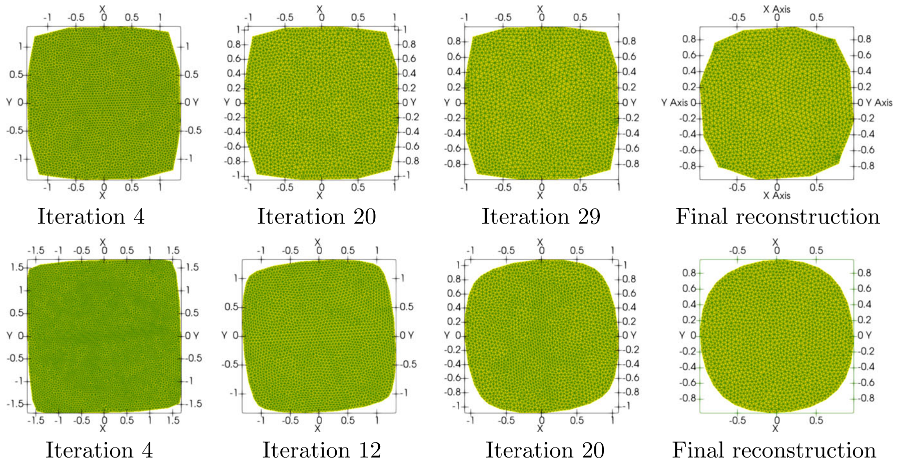


Fig. 5 Comparisons on the reconstructed shape using Algorithm 1 (above) and Algorithm 2 (below): case 1

From these numerical results, we conclude that the Algorithm 2 with only a fixed optimal step size parameter is more efficient than the Algorithm 1 performed with the Armijo-Goldstein strategy for optimal value of α, in terms of CPU time execution and the accuracy of optimal solutions. This is due to the fact that the classical approach requires, at each iteration, the resolution of the state and the adjoint state problems as well as the extension boundary value problem of the vector fields to the whole domain (2.7), which are of size N_{mn} the number of the nodes generated from the mesh of the domain limited by the N boundary elements (which is such that $N_{mn} \gg N$),

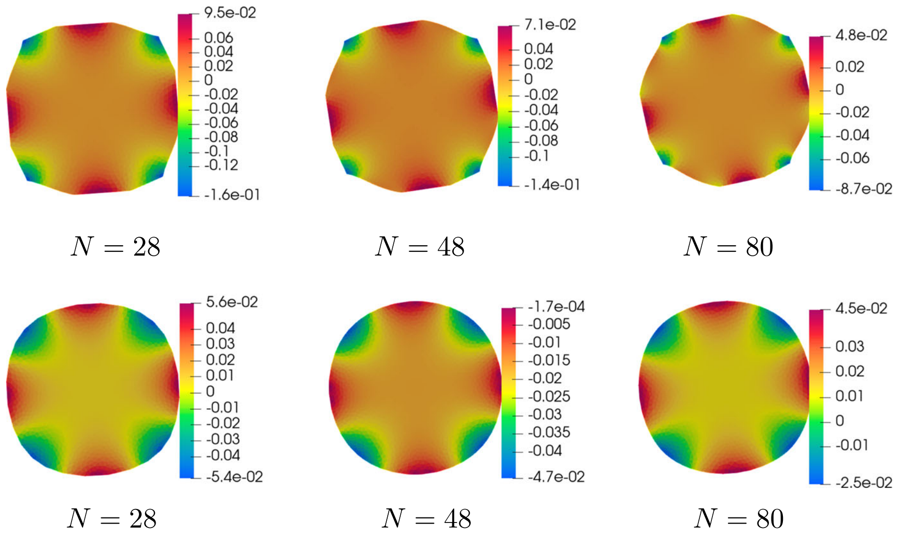


Fig. 6 The computed error of the solution in optimal domain using Algorithm 1 (above) and Algorithm 2 (below): case 1

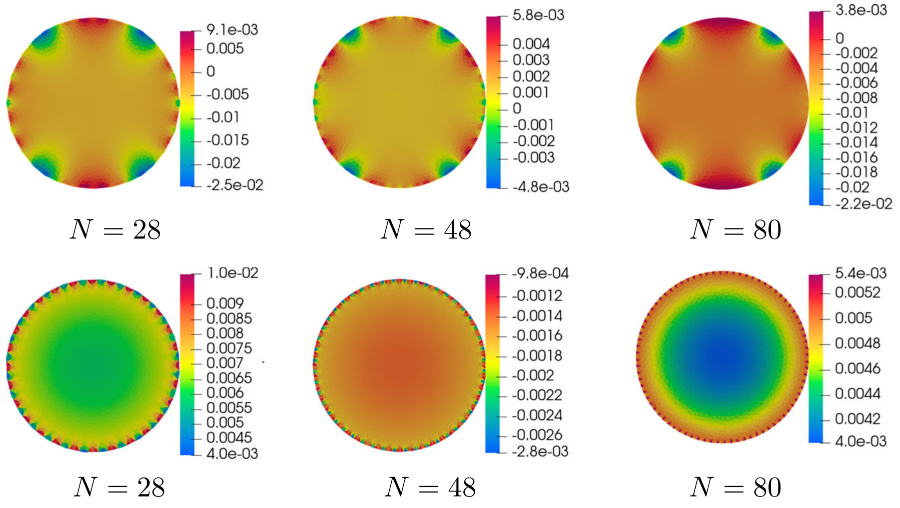


Fig. 7 The computed error of the solution in optimal domain using Algorithm 1 (above) and Algorithm 2 (below): case 2

in contrast to the proposed approach which requires in addition to the resolution of the state and the adjoint state problems, to minimize only a linear optimization problem (2.13) of size N.

In the sequel, in order to show the performance of the proposed approach, we will present more numerical tests for different values of boundary elements N using the

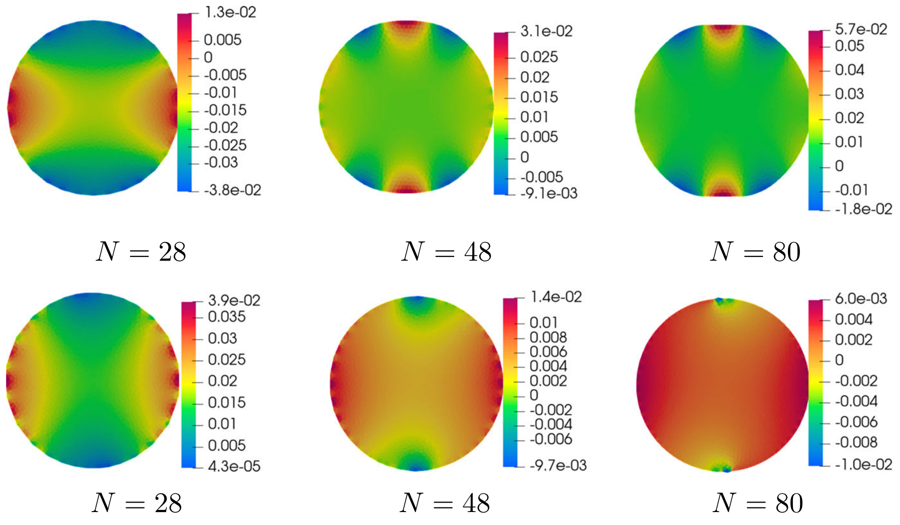


Fig. 8 The computed error of the solution in optimal domain using Algorithm 1 (above) and Algorithm 2 (below): case 3

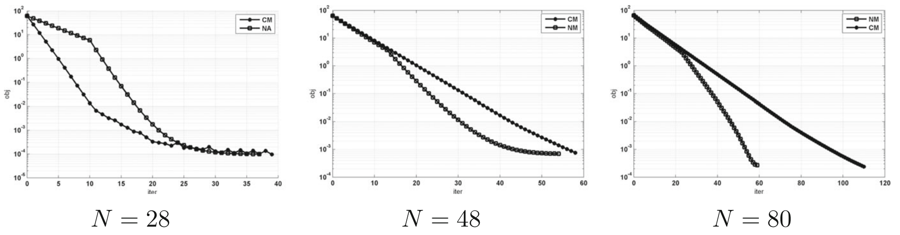


Fig. 9 Comparison of the convergence histories of the objective functional using Algorithms 1 and 2: case 1

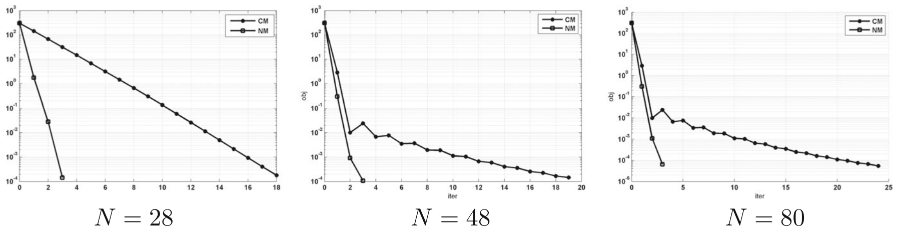


Fig. 10 Comparison of the convergence histories of the objective functional using Algorithms 1 and 2: cases 2

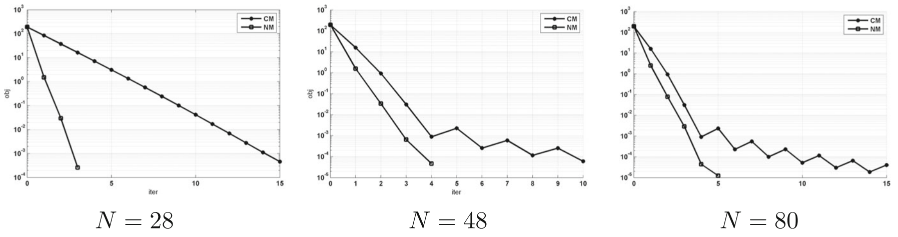


Fig. 11 Comparison of the convergence histories of the objective functional using Algorithms 1 and 2: cases 3

Table 4 Numerical comparison between Algorithms 1 and 2: case 2

	CM			NM			Time reduction
	CPU-time (s)	Iteration	Objective	CPU-time (s)	Iteration	Objective	
$N = 48$	**16.53**	19	1.1×10^{-4}	7.315	3	3.93×10^{-6}	**56%**
$N = 80$	**34.77**	24	6.67×10^{-5}	14.655	3	3.24×10^{-5}	**58%**

Table 5 Numerical comparison between Algorithms 1 and 2 for different cases and for $N = 28$

	CM			NM			Time reduction
	CPU-time (s)	Iteration	Objective	CPU-time (s)	Iteration	Objective	
disc	**12.573**	13	1.1×10^{-4}	**6.675**	3	1.43×10^{-4}	**47%**
ellipse	**20.107**	15	5.2×10^{-4}	**7.588**	3	7.27×10^{-4}	**62,26%**
square	**49.795**	39	2.76×10^{-4}	**14.504**	37	4.2×10^{-4}	**71%**

Algorithm 2 with fixed optimal step size parameter and the Algorithm 1 performed with the Armijo-Goldstein strategy.

Numerical examples tests for different values of boundary elements
For these tests, the initial shape is taken to be the square. Let us first illustrate the convergence of the two algorithms, for the number of elements $N = 28$. So, we present in Fig. 5, the successive and optimal shapes for both approaches. The obtained optimal shapes are reached after 35 and 39 iteration respectively for the Algorithm 2 and the Algorithm 1. We observe that the numerical optimal shape obtained by Algorithm 2 is of good quality compared to the one obtained by Algorithm 1 which presents four singular corner points appearing on its boundary. Then, the errors on the solutions in the optimal domains obtained by the finite element method for different initial domains case 1, case 2, and case 3 and for different N ($N \in \{28, 48, 80\}$) are presented respectively in Figs. 6, 7, and 8. Also the variation of the objective or cost functionals with respect to the number of iterations are presented respectively in Figs. 9, 10, and 11. These results shows that the new Algorithm 2 converges faster to more accurate optimal solutions than the Algorithm 1 for different initial shapes and different values of N. This expectation is confirmed in Table 4 where we present the CPU-time, the final objective cost, the iteration numbers and the time reduction for both approaches, when we use the initial shape of case 2. We observe that the proposed approach converges in less number of iterations compared to the classical one and the CPU execution time is reduced by at least **50%** which increases with the number of elements N to reaches **58%**. In the Table 5, we illustrate the CPU-time, the final objective cost and the time reduction, for $N = 28$ and different initial shapes, for both approaches. We see that the time reduction obtained by using the square as an initial shape is very important (**71%**) compared to the one obtained by using the disk as an initial shape (**47%**), which is obvious due to singular points of the square (see Fig. 5).

3.2.2 Shape optimization problems of minimizing L^2−gradient and H^1 norms constrained to Dirichlet boundary value problem

In this section, we consider the numerical approximation of shape optimization problems, constrained to Laplace-Dirichlet boundary value state problem, of minimizing cost functionals involving the L^2−norm of the gradient and the H^1-norm of the solution of the state problem. For all the numerical tests the exact solution is taken $u_{ex} = x^2 + y^2 - 1$.

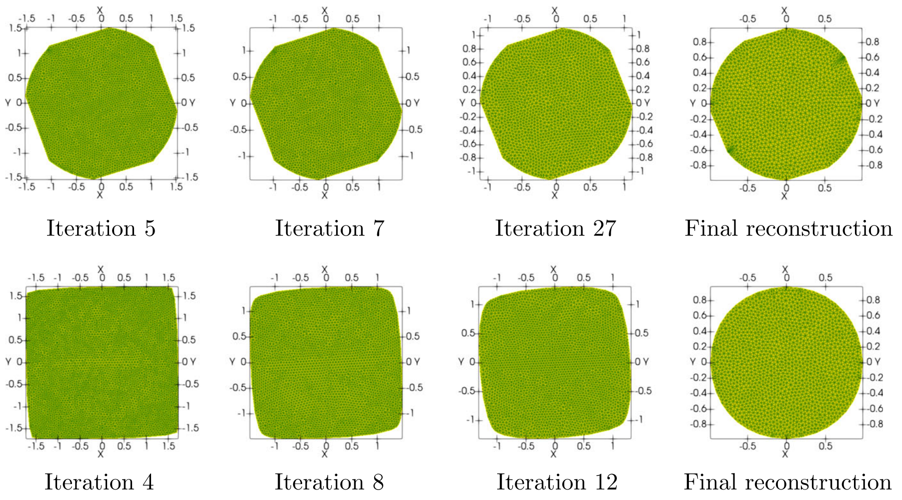


Fig. 12 Comparisons on the reconstructed shape using Algorithm 1 (above) and Algorithm 2 (below)

Our main objective is to propose a comparative numerical study between the Algorithm 2 and the Algorithm 1 in terms of computation time (time CPU) and the accuracy of the obtained optimal solutions for different initial domains associated to the three considered cases.

Cost functional involving the L^2—norm of the gradient of the state problems solution's

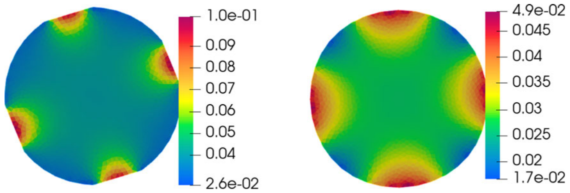
We perform now a comparison numerical result between the Algorithm 2 and the Algorithm 1 for solving the shape optimization problem (3.1) of minimizing the cost functional $\mathcal{F}$, for $\varpi = 0$ and $\beta = \dfrac{1}{2}$:

$$\mathcal{F}(\Omega, u_\Omega, \nabla u_\Omega) = \frac{1}{2}\|\nabla u_\Omega - \nabla u_{ex}\|^2_{L^2(\Omega)}. \tag{3.9}$$

where u_Ω is the solution of the following Dirichlet state problem:

$$(PE_2)\begin{cases} -\Delta u + u = f_{ex} & \text{in } \mathcal{D} \\ u = g_{ex} & \text{on } \partial\mathcal{D}. \end{cases}$$

Fig. 13 The computed error of the solution in optimal shape using Algorithm 1 (left) and Algorithm 2 (right)



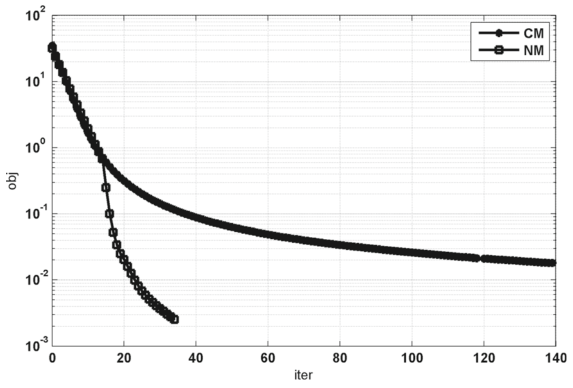


Fig. 14 Comparison of the convergence histories of the objective functional using Algorithms 1 and 2

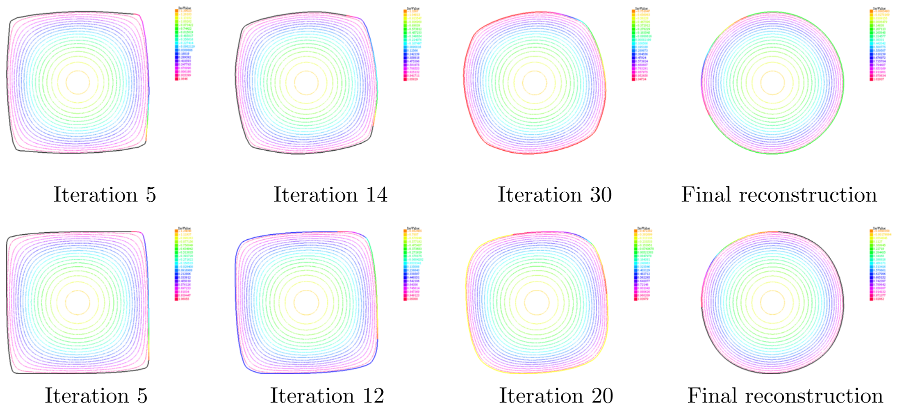


Fig. 15 Comparison of the reconstructed solution using Algorithm 1 (above) and Algorithm 2 (below): case 1

Table 6 Numerical comparison between Algorithms 1 and 2

	CM	NM
CPU (s)	**125.302**	**51.98**
Iteration	139	34
Objective	1.14×10^{-2}	2.5×10^{-3}
Time reduction	**58,51%**	

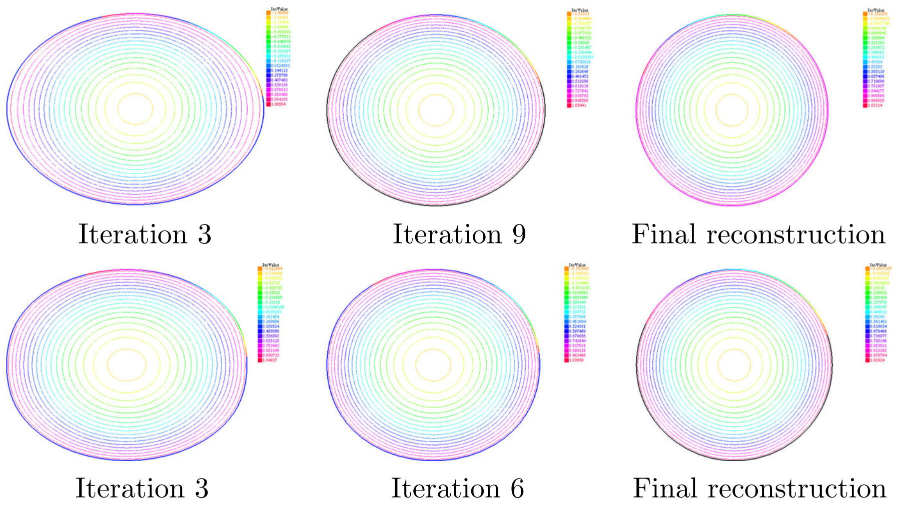


Fig. 16 Comparison of the reconstructed solution using Algorithm 1 (above) and Algorithm 2 (below): case 3

So, the expression of the gradient at $\Omega_0 \in \mathcal{U}$, for the two shape derivative formulas, is given by the following:

$$\left(\frac{\partial u_{\Omega_0}}{\partial \nu_0}\frac{\partial p_0}{\partial \nu_0} + \frac{1}{2}\left(|\nabla u_{ex}|^2 - [\frac{\partial u_{\Omega_0}}{\partial \nu_0}]^2\right)\right)\Big|_{\partial\Omega_0},$$

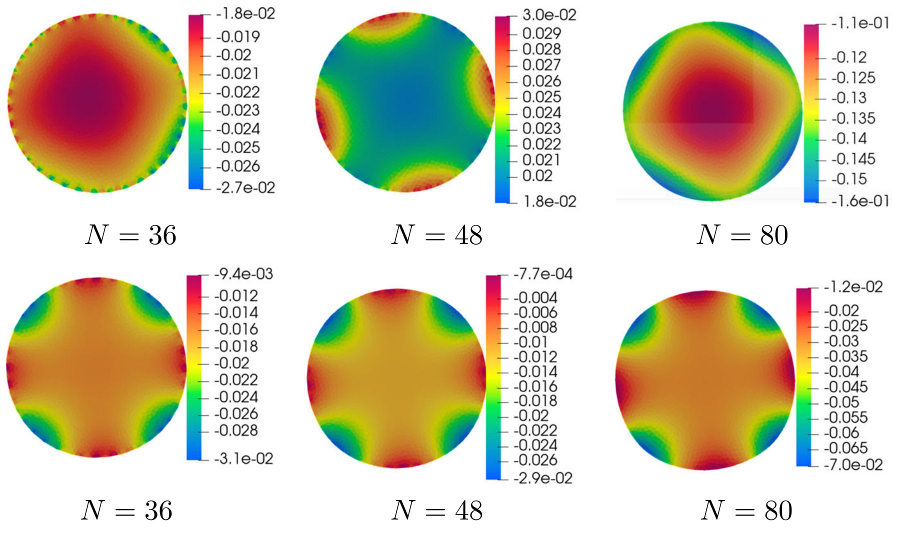
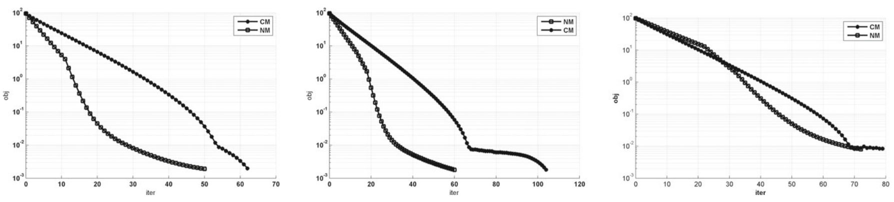


Fig. 17 The computed errors of the solutions in the optimal domain using Algorithm 1 (above) and Algorithm 2 (below): case 1



$N = 36$ $N = 48$ $N = 80$

Fig. 18 Comparison of the convergence histories of the objective functional using Algorithm 1 and Algorithm 2: case 1

where p_0 is the solution of the adjoint state problem (PE_2) with $f = \Delta u_{ex} - \Delta u$.

In this example, we choose the initial shape as the square given in case 1 and we set the number of elements $N = 36$ for both approaches. The successive and optimal shape is plotted in Fig. 12. The computed errors of the solutions in the optimal shape are presented in Fig. 13. Then, the convergence history for the objective functional is plotted in Fig. 14, which shows that the proposed approach converges faster for more accurate optimal solution than the classical one with the same precision of the stopping criterion. This is confirmed in Table 6 where we present the CPU-time, the number of iteration, the final cost functional and the CPU-time reduction. In this case the CPU time for the proposed approach is reduced by **58%** compared to the classical one.

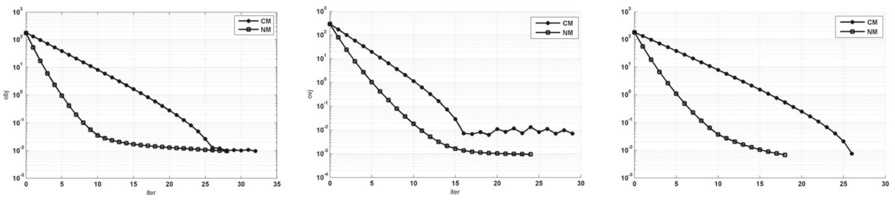
Cost functional involving the H^1-norm of the state problems solution's

We deal here with the numerical approximation of the shape optimization problem (3.1) of minimizing the cost functional $\mathcal{F}$, for $\varpi = \frac{1}{2}$ and $\beta = \frac{1}{2}$:

$$\mathcal{F}(\Omega, u_\Omega, \nabla u_\Omega) = \frac{1}{2}\|u_\Omega - v_1\|^2_{L^2(\Omega)} + \frac{1}{2}\|\nabla u_\Omega - \nabla v_2\|^2_{L^2(\Omega)} \tag{3.10}$$

where $v_1 = v_2 = u_{ex}$ and u_Ω is the solution of the Dirichlet state problem. The shape gradient of this functional at $\Omega_0 \in \mathcal{U}$, for the two shape derivative formulas, is given by the following:

$$\left(\frac{\partial u_{\Omega_0}}{\partial \nu_0}\frac{\partial p_0}{\partial \nu_0} + \frac{1}{2}\left(|\nabla v_2| - \left[\frac{\partial u_{\Omega_0}}{\partial \nu_0}\right]^2 \right) + \frac{1}{2}\left|u_{\Omega_0} - v_1\right|^2 \right)\Bigg|_{\partial\Omega_0} . \tag{3.11}$$



$N = 36$ $N = 48$ $N = 80$

Fig. 19 Comparison of the convergence histories of the objective functionals using Algorithm 1 and Algorithm 2: case 3

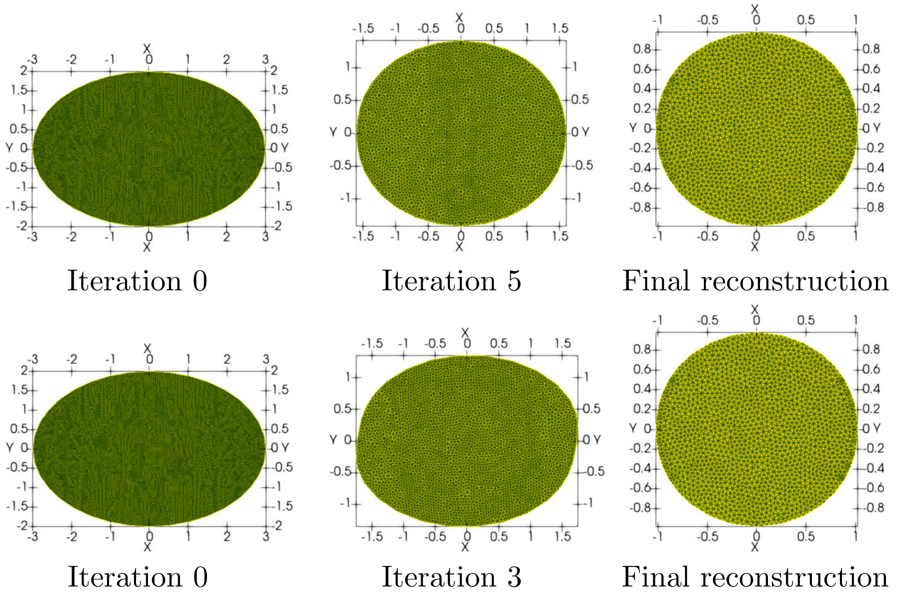


Fig. 20 Comparison of the reconstructed shapes using Algorithm 1 (above) and Algorithm 2 (below): case 3

where p_0 is the solution of the adjoint state problem:

$$-\Delta p_0 + p_0 = (\Delta v_2 - \Delta u) + (u - v1) \text{ in } \Omega_0, \quad p_0 = 0 \text{ on } \partial\Omega_0 \tag{3.12}$$

For this example we take $N = 80$, so in Figs. 15 and 16, we plot the successive and the final reconstructed solutions respectively for initial shapes the square (case 1) and the ellipse (case 3). The computed errors of the solutions in the optimal shapes are

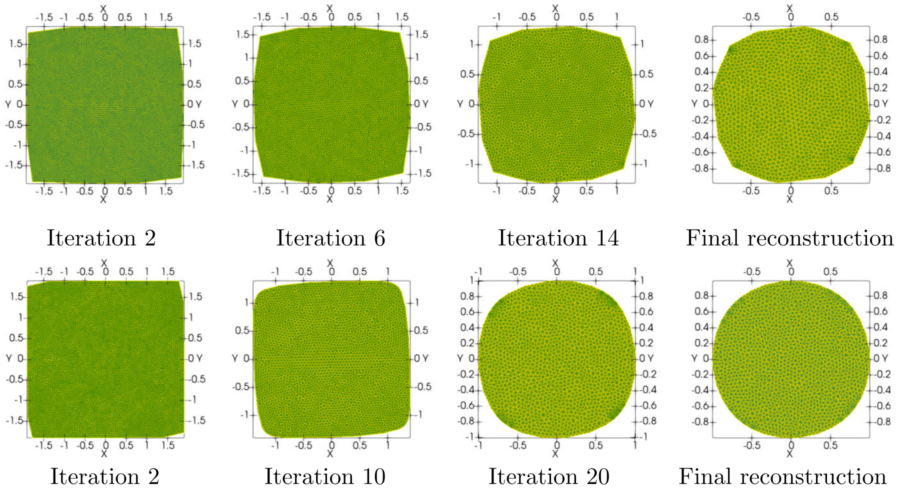


Fig. 21 Comparison of the reconstructed solutions using Algorithm 1 (above) and Algorithm 2 (below): case 1

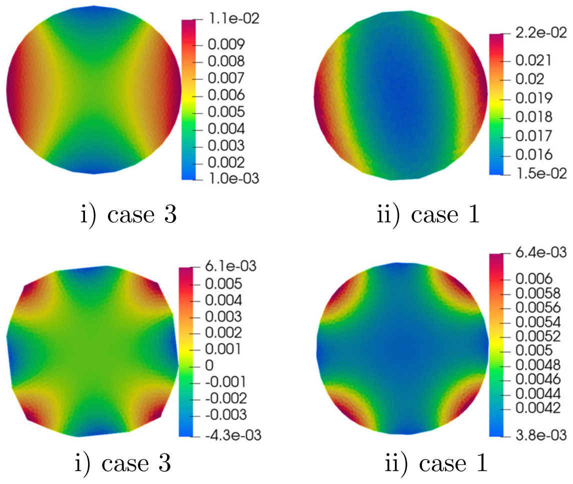


Fig. 22 The computed error of the solution in the optimal shape for ellipse as initial shape (first row) and square as initial shape (second row) using Algorithm 1 (left) and Algorithm 2 (right)

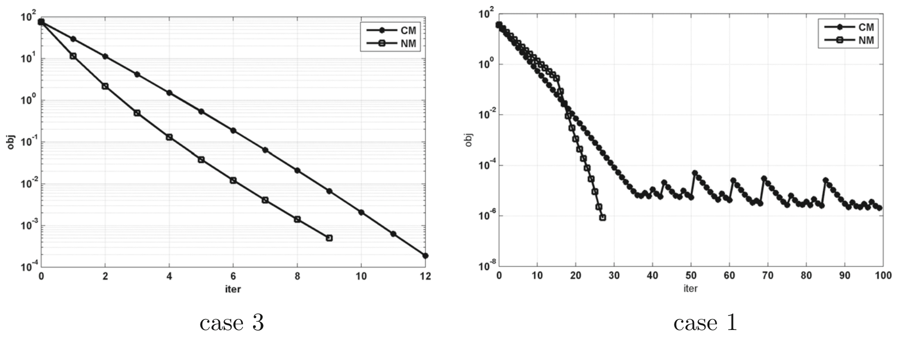


Fig. 23 Comparison of the convergence histories of the objective functional using Algorithms 1 and 2 for both Cases

Table 7 Numerical comparison between Algorithms 1 and 2: case 1

	CM			NM			Time reduction
	CPU-time (s)	Iteration	Objective	CPU-time (s)	Iteration	Objective	
$N = 36$	**132.37**	63	1.98×10^{-3}	**50.41**	24	1.91×10^{-3}	**62%**
$N = 48$	**197.498**	104	1.81×10^{-3}	**67.121**	60	1.80×10^{-3}	**66%**
$N = 80$	**300.14**	79	8.39×10^{-3}	**90.85**	72	8.04×10^{-3}	**69.73%**

Table 8 Numerical comparison between Algorithms 1 and 2: case 3

	CM			NM			Time reduction
	CPU-time (s)	Iteration	Objective	CPU-time (s)	Iteration	Objective	
$N = 36$	**37.015**	33	9.67×10^{-3}	**17.029**	28	7.20×10^{-3}	**54%**
$N = 48$	**47.701**	29	7.83×10^{-4}	**20.087**	20	9.46×10^{-4}	**58%**
$N = 80$	**69.58**	26	7.53×10^{-3}	**25.41**	18	6.79×10^{-3}	**63.48%**

Table 9 Numerical comparisons between Algorithms 1 and 2

	CM			NM			Time reduction
	CPU-time (s)	Iteration	Objective	CPU-time (s)	Iteration	Objective	
case 3	**125.55**	12	1.88×10^{-4}	**13.93**	9	1.80×10^{-4}	**88,90%**
case 1	**460.677**	100	2.10×10^{-6}	**250.861**	27	8.50×10^{-7}	**45,54%**

shown in Fig. 17 for the initial shape is square. The comparison of the convergence history of the objective functional for both approaches and for different initial shapes are presented in Figs. 18 and 19. In Tables 7 and 8, we give the final cost functional, the CPU time, the iterations number and the CPU time reduction for $N = 36$, $N = 48$ and $N = 80$. In this example, we deduce also that the proposed approach converges faster to more accurate optimal solutions in less number of iterations compared to the classical one for both initial shapes and for different values of N.

3.2.3 Shape optimization constrained to elliptic Neumann boundary value problem

We consider the shape optimization problem 2.1 of minimizing the cost functional $\mathcal{F}$ with $\varpi = 1$ and $\beta = 0$:

$$\mathcal{F}(\Omega, u_\Omega, \nabla u_\Omega) = \|u_\Omega - u_{ex}\|^2_{L^2(\Omega)}, \tag{3.13}$$

where $u_{ex} = x^2 + y^2 - 1$ and u_Ω is the solution of the Neumann boundary value state problem associated to the parameters $\mu = 1$, $\lambda = 1$, $\eta = 1$ and $\gamma = 0$. The shape gradient of this functional at $\Omega_0 \in \mathcal{U}$, for the two shape derivative formulas, is given by the following:

$$\Big(|u - u_{ex}|^2 + \nabla u.\nabla p + p(u - u_{ex})\Big)\Big|_{\partial\Omega_0}, \tag{3.14}$$

where p is the solution of the adjoint state problem:

$$-\Delta p + p = -2(u_0 - u_{ex}) \text{ in } \Omega_0, \quad \frac{\partial p}{\partial \nu} = 0 \text{ on } \partial\Omega_0. \tag{3.15}$$

In this example, we take $N = 36$, so in Figs. 20 and 21, we plot the successive and the final reconstructed shapes respectively for initial shapes the ellipse (case 3) and the square (case 1). The computed errors of the solutions in the optimal shapes are shown in Fig. 22 for the initial shape is square and ellipse. The comparison of the convergence history of the objective cost functional for both approaches and for different initial shapes are presented in Fig. 23. Finally, in Table 9, we give the final cost functional, the CPU time, the iterations number and the CPU time reduction for different cases. In this example, we deduce also that the proposed approach converges faster to more accurate optimal solutions in less number of iterations compared to the classical one for both initial shapes.

3.2.4 Shape optimization constrained to elliptic Stokes flows equation

In this section, we deal with the numerical approximation of shape optimization problem (3.4) of minimizing different cost functionals constrained to Stokes equation, especially the functional of minimizing the L^2-norm of the solution of the state problem or the functional energy. So, the main objective is to achieve a comparative numerical study between the Algorithm 2 and the Algorithm 1 in terms of the CPU (s) time and the accuracy of the obtained optimal solutions, through the resolution of this class of problems for different initial domains associated the three above cases. For this, the Taylor-Hood finite element discretization $\mathbb{P}_2/\mathbb{P}_1$ is used with different number of boundary elements N for approximating respectively the fluid velocity and the associated pressure for the Stokes problem for the following data: the viscosity coefficient is given by $\kappa = 0.01$, the target velocity and pressure are respectively given by $u_d(x, y) = (-y\,(x^2 + y^2 - 4), x\,(x^2 + y^2 - 4))$, $p_{ex} = x + y - 1$ and $f_{ex} = (8\,\kappa\, y + 1, 1 - 8\,\kappa\, x)$.

Quadratic functional

We consider the shape optimization problem (3.4) of minimizing the cost functional $\mathcal{S}$ for $\rho = \frac{1}{2}$ and $\sigma = 0$:

$$\Omega \in \mathcal{U} \longmapsto \mathcal{S}(\Omega, w_{\Omega_0}, \nabla w_{\Omega_0}) := \frac{1}{2}\int_{\Omega_0} \|w_{\Omega_0} - u_d\|^2 dx, \tag{3.16}$$

where $(\omega_{\Omega_0}, p_{\Omega_0})$ is solution of the problem (3.5) on Ω_0. The shape gradient of this functional at $\Omega_0 \in \mathcal{U}$, for the two shape derivative formulas, is given by the following:

$$\left(\|w_{\Omega_0} - u_d\| + \nabla w_{\Omega_0} : \nabla\psi_0\right)\Big|_{\partial\Omega_0}, \tag{3.17}$$

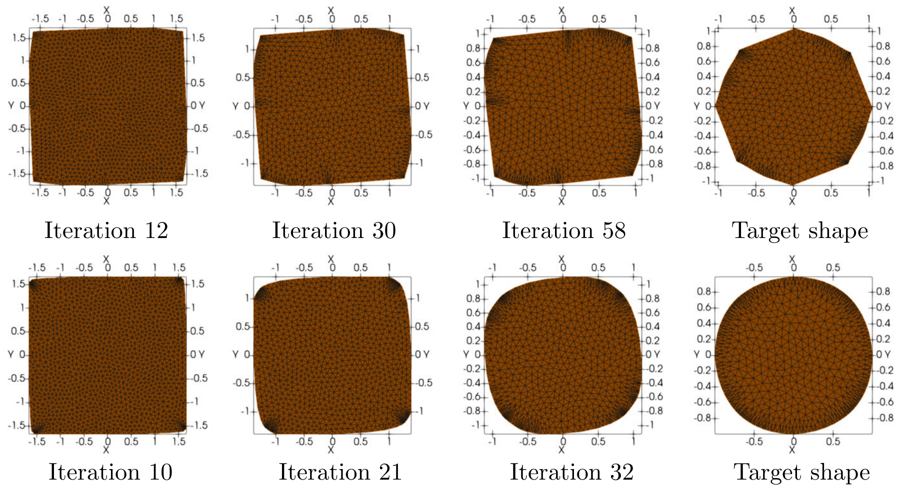


Fig. 24 Comparison of the reconstructed shapes using Algorithm 1 (above) and Algorithm 2 (below): case 1

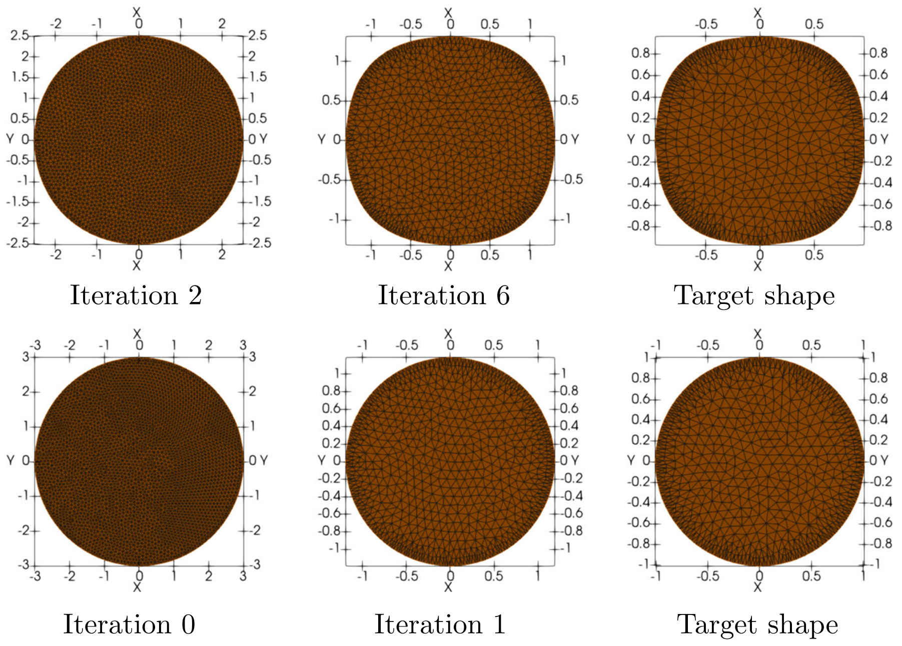


Fig. 25 Comparison of the reconstructed shape using Algorithm 1 (above) and Algorithm 2 (below): case 3

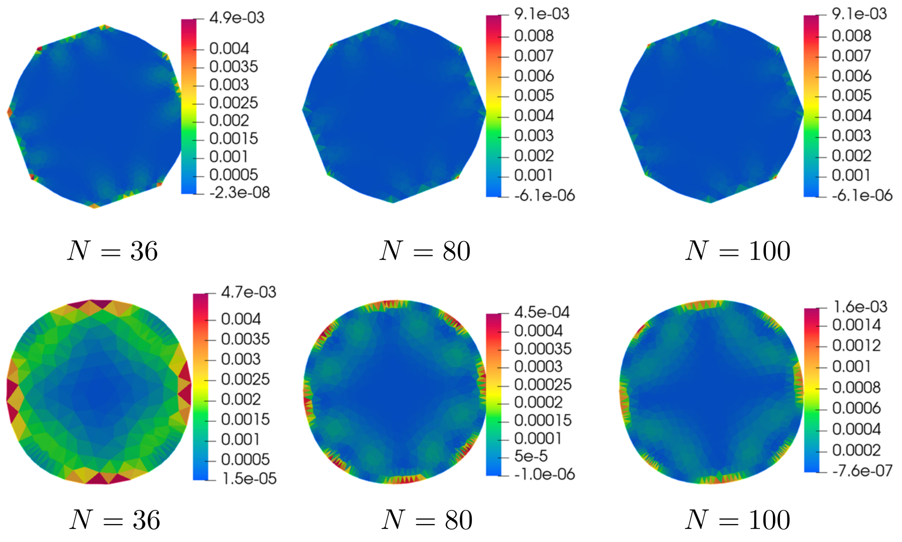


Fig. 26 The computed error of the solution in the optimal domain using Algorithm 1 (above) and Algorithm 2 (below): case 1

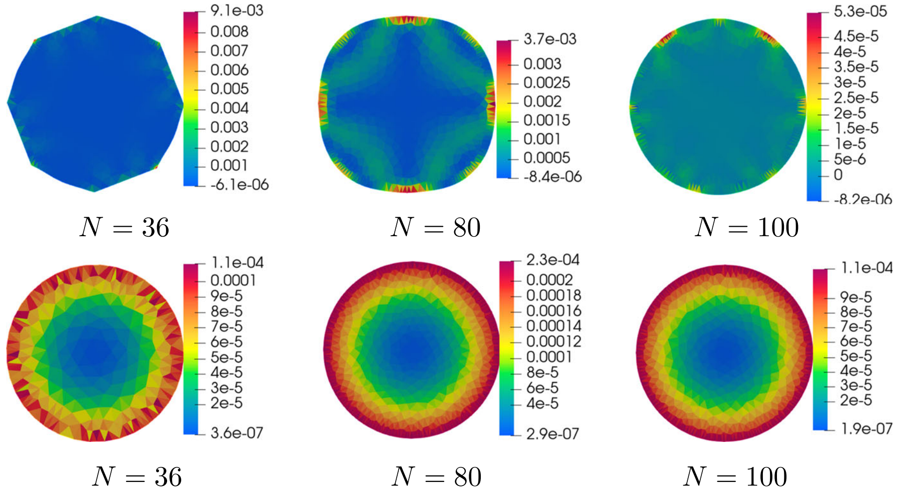


Fig. 27 The computed error of the solution in the optimal domain using Algorithm 1 (above) and Algorithm 2 (below): case 2

such that (ψ_0, p_0) is the solution of the adjoint state stokes problem:

$$-\kappa\Delta\psi_0+\nabla p_0=u_0-u_d \ \text{ in } \ \Omega_0, \quad \text{div}\psi_0=0 \ \text{ in } \ \Omega_0 \quad \psi_0=0 \ \text{ on } \partial\Omega_0. \quad (3.18)$$

For the first test we take $N = 80$, thereby the successive and optimal shape are plotted in Figs. 24 and 25 respectively for the initial shape is a square (case 1) and a disc (case 3). Then, the computed errors of the solutions in the optimal shape with

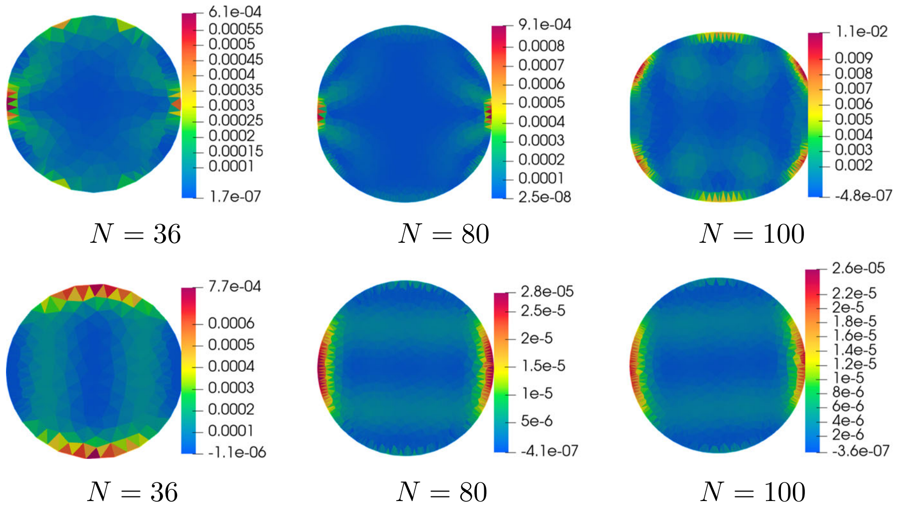


Fig. 28 The computed error of the solution in the optimal domain using Algorithm 1 (above) and Algorithm 2 (below): case 3

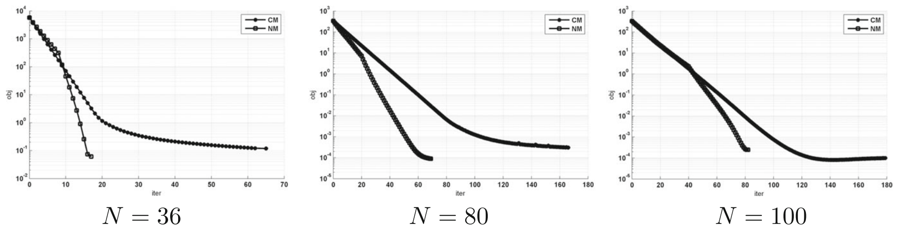


Fig. 29 Comparison of the convergence histories of the objective functional using Algorithms 1 and 2: case 1

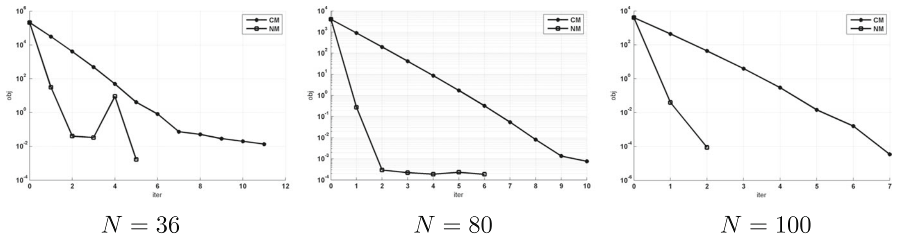


Fig. 30 Comparison of the convergence histories of the objective functional using Algorithms 1 and 2: case 2

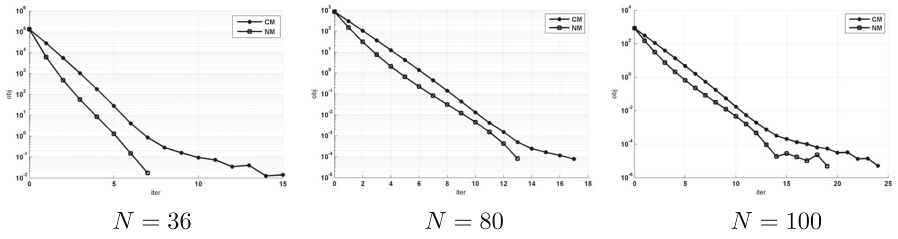


Fig. 31 Comparisons of the convergence histories of the objective functional using Algorithms 1 and 2: case 3

Table 10 Numerical comparison between Algorithms 1 and 2: case 1

	CM			NM			Time reduction
	CPU-time (s)	Iteration	Objective	CPU-time (s)	Iteration	Objective	
$N = 36$	**103.45**	64	9.6×10^{-2}	**60.79**	17	6.1×10^{-2}	**41.23%**
$N = 80$	**740.378**	167	3.03×10^{-4}	**412.06**	69	9.14×10^{-5}	**44,34%**
$N = 100$	**1120.456**	180	9.99×10^{-5}	**580.40**	83	2.46×10^{-4}	**48,19%**

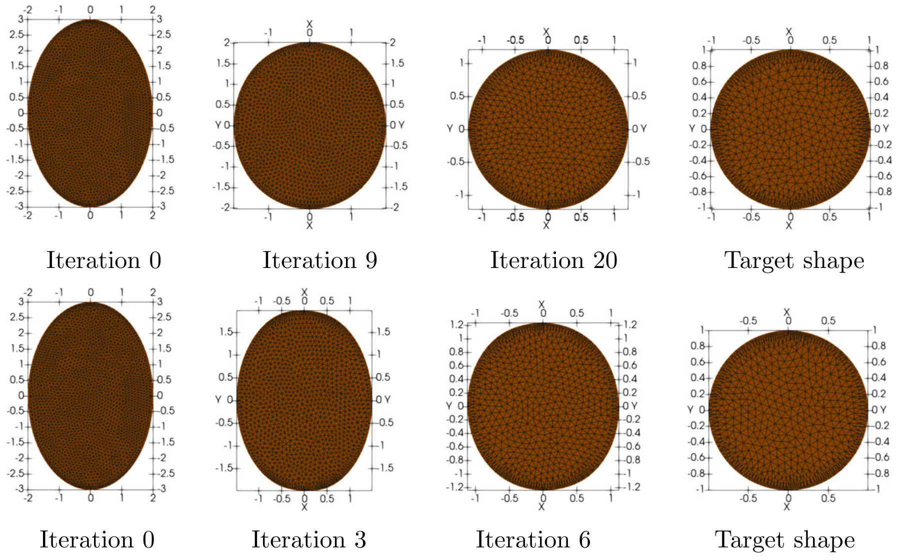


Fig. 32 Comparison of the reconstructed shapes using Algorithm 1 (above) and Algorithm 2 (below)

the iso-value meshes are presented in Figs. 26, 27, and 28 for different number of element N and different initial shapes. The convergence history for the objective functional is plotted in Figs. 29, 30, and 31, which shows that the proposed approach converges faster for more accurate optimal solution than the classical one using the same precision of the stopping criterion. The performance of the proposed approach is confirmed in Table 10 where we present the CPU-time, the number of iteration, the final cost functional and the CPU-time reduction. In this case the CPU time of execution for this approach is reduced by **48%** compared to that of the classical one

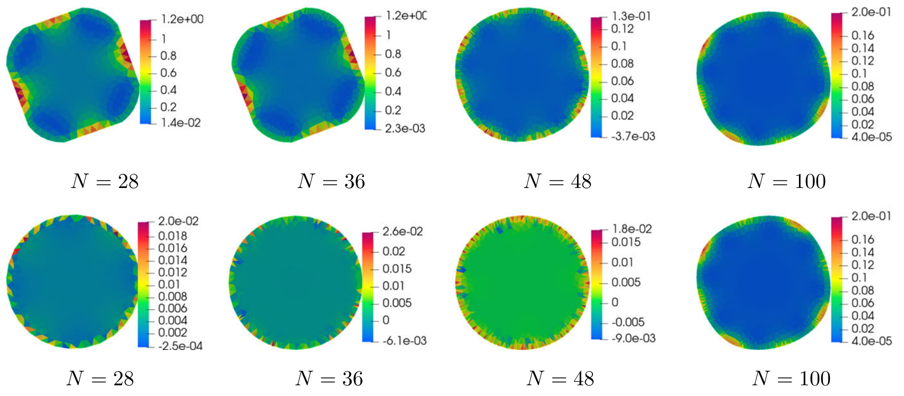


Fig. 33 The computed error of the solution in the optimal domain using Algorithm 1 (above) and Algorithm 2 (below): case 1

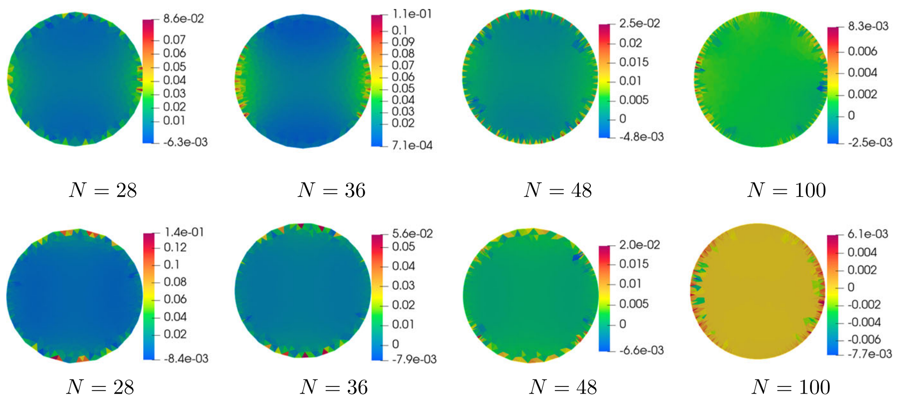


Fig. 34 The computed error of the solution in the optimal domain using Algorithm 1 (above) and Algorithm 2 (below): case 3

Energy functional Now we consider the shape optimization problem (3.1) of minimizing the cost functional $\mathcal{F}$ for $\rho = 0$ and $\sigma = \kappa$:

$$\Omega \in \mathcal{U} \longmapsto \mathcal{S}(\Omega, w_\Omega, \nabla w_\Omega) := \kappa \int_\Omega \|\nabla w_\Omega - \nabla u_d\|^2 dx. \tag{3.19}$$

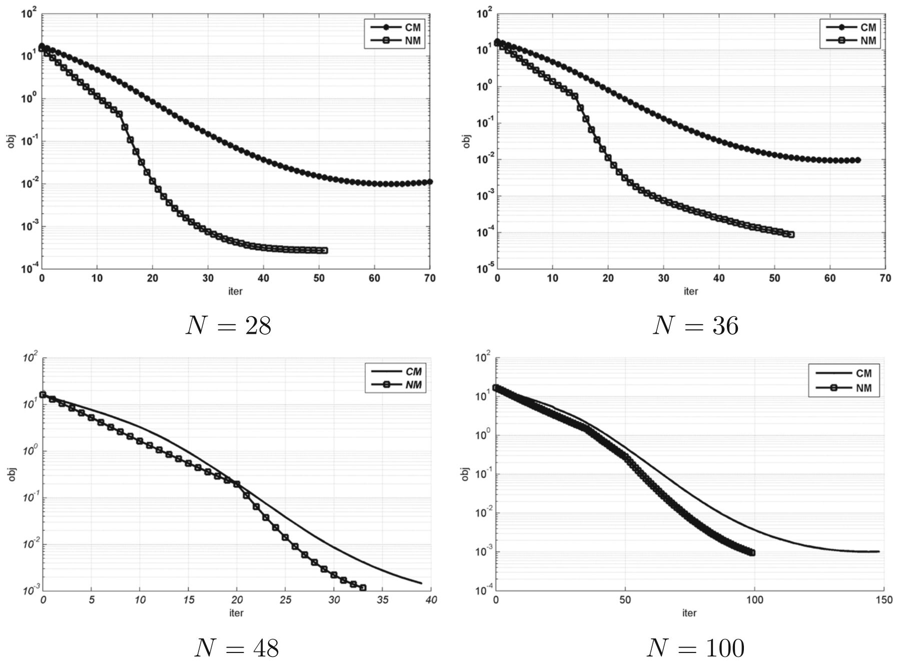


Fig. 35 The computed error of the solution in the optimal domain using Algorithms 1 and 2: case 1

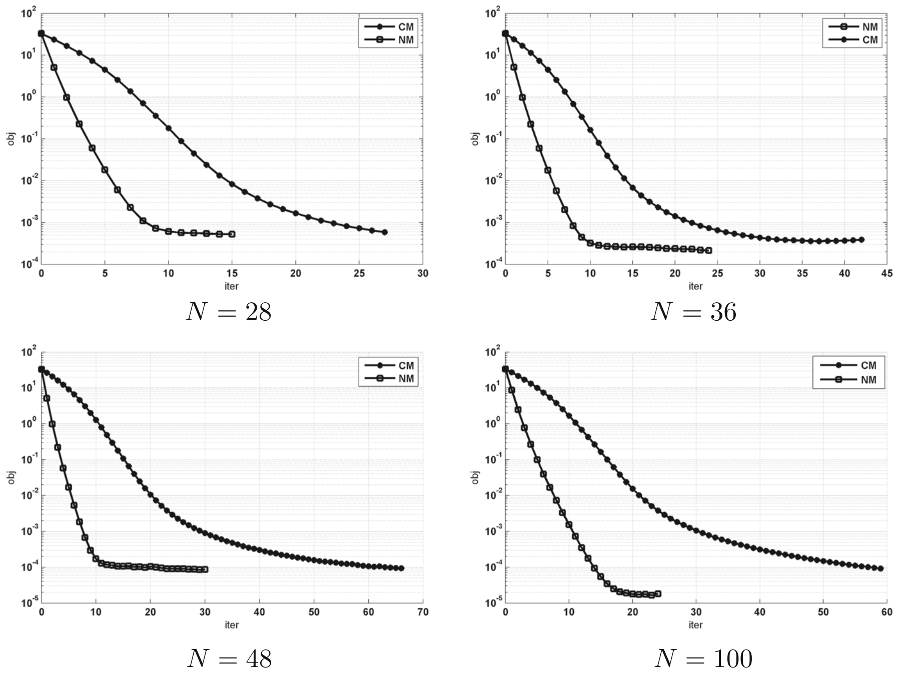


Fig. 36 The computed error of the solution in optimal domain using Algorithms 1 and 2: case 3

The shape gradient of this functional at $\Omega_0 \in \mathcal{U}$, for the two shape derivative formulas, is given by the following:

$$(\kappa \left[(\nabla w_{\Omega_0}.\nu_0).(\nabla p_0.\nu_0) - \|\nabla w_{\Omega_0}.\nu_0\|^2 \right])\Big|_{\partial\Omega_0},$$

such that (ψ_0, p_0) is the solution of the adjoint state stokes problem:

$$-\kappa\Delta\psi_0 + \nabla q_0 = \kappa\Delta(w_{\Omega_0} - u_d) \text{ in } \Omega_0, \quad \text{div}\psi_0 = 0 \text{ in } \Omega_0 \quad \psi_0 = 0 \text{ on } \partial\Omega_0. \tag{3.20}$$

For this example, the successive and optimal shape are plotted in Fig. 32 for the initial shape is ellipse case 3) and for $N = 100$. The computed optimal shapes are reached after 24 and 59 iteration respectively for the Algorithms 2 and 1. The errors on the solutions in the optimal domains obtained by the finite element method for different initial domains considered in case 1 and case 3 and for different values of N ($N \in \{28, 36, 48, 100\}$) are presented respectively in Figs. 33 and 34. Also the variation of the objective functionals with respect to the number of iterations are presented respectively in Figs. 35 and 36. These results shows that the new Algorithm 2 converges faster to more accurate optimal solutions than the Algorithm 1 for different values of N. This expectation is confirmed in Table 11 where we present the CPU-time, the final objective functional, the iteration numbers and the time reduction for

Table 11 Numerical comparison between Algorithms 1 and 2: case 3

	CM			NM			Time reduction
	CPU-time (s)	Iteration	Objective	CPU-time (s)	Iteration	Objective	
$N = 36$	**225.77**	42	3.71×10^{-4}	**112.868**	28	2.00×10^{-4}	**50%**
$N = 48$	**324.584**	66	9.30×10^{-5}	**148.511**	30	8.67×10^{-5}	**54,24%**
$N = 100$	**441.156**	59	9.15×10^{-3}	**153.418**	24	1.82×10^{-5}	**65,22%**

both approaches, when we use the initial shape of case 3. We observe for this test also that the proposed approach converges in less number of iterations compared to the classical one and the CPU execution time is reduced by at least **50%** which increases with the number of elements N to reaches **65%**. This expectation is confirmed in Table 12 where we illustrate the CPU-time, the final objective functional and the time reduction, for $N = 28$ and different initial shapes, for both approaches.

3.2.5 Discussions and concluding remarks on the numerical results

The comparative numerical study between the Algorithm 1 and the Algorithm 2 concerns the quality of optimal solutions and the CPU time reduction. In this regard, we note that for all the obtained numerical results, the Algorithm 2 reaches the optimal solutions on less number of iterations compared to the Algorithm 1. Moreover, we observe from these numerical results, that the difference between the numbers of iterations at convergence for the two algorithms and the CPU time reduction depend strongly on the choice of the step descent parameter in the gradient methods and the number of boundary elements discretization N as well as the type of the boundary condition in the state problems and the initial domains in the shape optimization processes. Indeed:

- Concerning the choice of the step descent parameter in the gradient methods, we have analyzed the sensitivity and the influence of the choice of the step size parameter on the convergence performance of the both algorithms. More precisely, we have compared first the two algorithms using a fixed optimal steps size (the best ones allowing a rapid convergence of the algorithms). Then, we compared the Algorithm 2 using a fixed optimal step size and the Algorithm 1 performed

Table 12 Numerical comparison between Algorithms 1 and 2 for different cases and $N = 28$

	CM			NM			Time reduction
	CPU-time (s)	Iteration	Objective	CPU-time (s)	Iteration	Objective	
Square	**310.20**	71	1.13×10^{-2}	**146.028**	51	2.73×10^{-4}	**53%**
Ellipse	**98.5**	27	5.86×10^{-4}	**55.107**	15	5.25×10^{-4}	**44,05%**

with updated step size by Armijo-Goldstein strategy (2.8). From the obtained numerical results, we conclude that the Algorithm 2 with only a fixed optimal step size parameter is more efficient than the Algorithm 1 performed with the Armijo-Goldstein strategy, in terms of CPU time execution and the accuracy of optimal solutions. This is due to the fact that the classical approach (Algorithm 1) requires, at each iteration, the resolution of the state and the adjoint state problems as well as the extension boundary value problem of the vector fields to the whole domain (2.8), which are of size N_{mn} the number of the freedom degrees generated from the mesh of the domain limited by the N boundary elements (which is such that $N_{mn} \gg N$), in contrast to the proposed approach which requires in addition to the resolution of the state and the adjoint state problems, to minimize only a linear optimization problem (2.13) of size N.

- Regarding the influence of the number of boundary elements N or its resulting number of the freedom degrees N_{mn} on the reduction of the CPU time, we have observed from some example tests, that even if the difference between the iteration numbers of the two algorithms is not large, the CPU time reduction is very remarkable. This is due to the fact that the Algorithm 1 requires to solve more additional linear systems of size N_{mn} (which are of number more greater than the iterations number of the Algorithm 2) compared to the Algorithm 2. This is in fact illustrated for example in Table 7 (last line for the case $N = 80$) where we remark that even if the difference between the iteration numbers of the two algorithms is just 7 iterations, the reduction of the CPU times reaches 69%, and for example in Table 8 (first line for the case $N = 36$) where the obtained CPU time reduction reaches 54%.
- We note also that the CPU time reduction is influenced by the geometrical regularity of the considered initial domains as well as on the type of the boundary condition (Dirichlet or Neumann condition) in the considered state problems. Indeed, when one consider the Dirichlet boundary value state problem on the square as initial shape, the shape optimization process requires a mesh refinement of the successive domains to overcome to the inconveniences due to the singularity of the corners points, thereby the number N_{mn} may be too large ($N_{mn} \ggg N$) compared to the case where the initial shape is more regular, such as the ellipse or the disc. This means that the reduction in CPU time is more important, when the initial domain is a square, even if the difference between the number of iterations of the two algorithms is not large. This is due to the fact that the Algorithm 1 requires to solve some additional linear systems of size N_{mn} (which are of number more greater than the iterations number of the Algorithm 2) compared to the Algorithm 2. This is in fact illustrated in the example tests in Table 5, where even if the difference between the iteration numbers of the two algorithms is not large, the CPU time reduction is remarkable and reaches 71%. This expectation is no longer true, when one consider the Neumann boundary value state problem on the square as initial shape, since the accuracy of the solutions of the state and adjoint state problems for both algorithms is affected by the discontinuity of the normal derivative on the corner points. This is in fact illustrated in Table 9, where the CPU time reduction for the example test where the initial shape is the ellipse is more important compared to the one where the initial shape is the square.

4 Conclusion

We conclude from this numerical study that the use of the shape derivative involving the support functions in the gradient shape optimization process is more advantageous than the one using the classical shape derivative involving vector fields, when the finite element method is used for the discretization of the auxiliary problems. This is illustrated through the comparative numerical results showing the efficiency of the resulting numerical process "Algorithm 2" of the proposed approach and its ability in producing good quality solutions and in providing better accuracy for the optimal solution in less CPU time compared to the classical approach "Algorithm 1," even if this last algorithm is based on the gradient method performed with the optimal step strategy of Armijo-Goldstein [26], while the Algorithm 2 is accomplished with only a fixed step chosen once for all the iterations of the optimization process.

Acknowledgements I. Khalil is supported by the National Center for Scientific and Technical Re-search (CNRST Maroc, The Research Excellence Grant, No. 11 USMS2022).

Author contribution All authors contributed equally to the manuscript and read and approved the final manuscript.

Funding The National Center for Scientific and Technical Research.

Data availability The data that support the findings of this study are available on request from the corresponding author.

Declarations

Ethical approval Not applicable

Conflict of interest The authors declare no competing interests.

References

1. Allaire, G.: Conception optimale de structures, Mathematiques et Applications 58. Springer, Berlin (2007)
2. Boulkhemair, A.: On a shape derivative formula in the Brunn-Minkowski theory. SIAM J. Control Optim. **55**(n°1), 156–171 (2017)
3. Boulkhemair, A., Chakib, A.: On a shape derivative formula with respect to convex domains. J. Convex Anal. **21**(n°1), 67–87 (2014)
4. Boulkhemair, A., Chakib, A., Sadik, A.: On numerical study of constrained coupled shape optimization problems based on a new shape derivative method. Numer. Methods Partial Differ. Eq. **39**, 2018–2059 (2023)
5. Boulkhemair, A., Chakib, A., Nachaoui, A., Niftiyev, A.A., Sadik, A.: On a numerical shape optimal design approach for a class of free boundary problems. Comput Optim Appl, Springer **77**, 509–537 (2020)
6. Chakib, A., Khalil, I., Ouaissa, H., Sadik, A.: On an effective approach in shape optimization problem for Stokes equation. Optimization Letters, pp. 1–8, (2023)
7. Bucur, D., Buttazzo, G.: Variational methods in some shape optimization problems. Scuola normale superiore (2002)
8. Céa, J.: Optimisation, théorie et algorithmes, Dunod, Paris, (1971)

9. Céa, J.: Problems of shape optimal design. In: Haug, C. (ed.) Optimization of distributed parameters structures, Part II, pp. 1005–1048. Sijthoff Noordhoff, Alphen aan den Rijn (1981)
10. Céa, J., Garreau, S., Guillaume, Ph., Masmoudi, M.: The shape and topological optimizations connection. Comput. Methods Appl. Mech. Engrg **188**(4), 713–726 (2000)
11. Céa, J., Gioan, A.J., Michel, J.: Quelques résultats sur l'identification de domaines. Calcolo **10**, 207–232 (1974)
12. Ciarlet, P.: The finite element method for elliptic problems, Society for Industrial and Applied Mathematics, (2002)
13. Ciarlet, P.: Mathematical elasticity I. Elsevier Science Publishers, B.V., Amsterdam, Te Netherlands (1988)
14. Delfour, M. C., Zolésio, J. P.: Shapes and geometries: metrics, analysis, differential calculus, and optimization. Siam, 22, (2011)
15. Delfour, M.C., Zolésio, J.P.: Anatomy of the shape Hessian. Annali di Matematica Pura ed Applicata **CLIX**(IV), 315–339 (1991)
16. Delfour, M.C., Zolésio, J.P.: Velocity method and Lagrangian formulation for the computation of the shape Hessian. SIAM J. Control Optim. **29**(n°6), 1414–1442 (1991)
17. Feppon, F., Allaire, G., Bordeu, F., Cortial, J., and Dapogny, C., Shape optimization of a coupled thermal fluid structure problem in a level set mesh evolution frameworki, SeMA Journal, 1–46 (2019)
18. Hadamard, J.: Mémoire sur le problème d'analyse relatif à l'équilibre des plaques élastiques encastrées, (1907), dans Oeuvres de J. Hadamard, CNRS Paris (1968)
19. Hintermüller, M., Laurain, A., Yousept, I.: Shape sensitivities for an inverse problem in magnetic induction tomography based on the eddy current model. Inverse Probl **31**(6), 065006 (2015)
20. Haslinger, J., Neittaanmäki, J.: Finite element approximation for optimal shape, material, and topology design. John Wiley Sons (1996)
21. Hecht, F.: New development in FreeFem++. J. Numer. Math. **20**(3–4), 251–266 (2012)
22. Henrot, A.: Shape optimization and spectral theory. Shape optimization and spectral theory, De Gruyter Open Poland (2017)
23. Henrot, A.: Extremum problems for eigenvalues of elliptic operators. Springer Science Business Media (2006)
24. Henrot, A., Pierre, M.: Variation et optimisation de formes. Une analyse géométrique, Mathematics and Applications, 48, Springer, Berlin, (2005)
25. Gong, W., Li, J., Zhu, S.: Improved Discrete Boundary Type Shape Gradients for PDE-constrained Shape Optimization. SIAM J Sci Comput, **44**(4), (2022)
26. Ito, K., Kunisch, K., Peichl, G.: Variational approach to shape derivative for a class of Bernoulli problem. J. Math. Anal. Appl. **314**, 126149 (2006)
27. Laurain, A., Sturm, K.: Distributed shape derivative via averaged adjoint method and applications. ESAIM: Math. Model. Numer. Anal **50**(4), 1241–1267 (2016)
28. Luft, D., Schulz, V.: Pre-shape calculus: foundations and application to mesh quality optimization. Control Cybern n°**3**(50), 263–302 (2021)
29. Luft, D., Schulz, V.: Simultaneous shape and mesh quality optimization using pre-shape calculus. Control Cybern n° **4**(50), 473–520 (2021)
30. Mohammadi, B., Pironneau, O.: Applied shape optimization for fluids, 2nd edn. Numerical Mathematics and Scientific Computation. Oxford University Press, Oxford (2010)
31. Murat, F., Simon, J.: Sur le contrôle par un domaine géométrique. Pré-publication du laboratoire d'analyse numérique, n° 76015, Université Paris VI, (1976)
32. Niftiyev, A.A., Gasimov, Y.S.: Control by boundaries and eigenvalue problems with variable domains. Publishing House of Baku State Universit, Baku(in Russian) (2004)
33. Penot, J.-P.: Calculus without derivatives, Graduate Texts in Mathematics. Laboratoire Jacques-Louis Lions. Universite Pierre et Marie Curie, Paris, France Springer (2013)
34. Pironneau, O.: Optimal shape design for elliptic systems. Springer Series in Computational Physics, Springer, New York (1984)
35. Schneider, R.: Convex bodies: the Brunn-Minkowski theory, 151st edn. Cambridge university press (2014)
36. Simon, J.: Differentiation with respect to the domain in boundary-value problems. Numerical Functional Analysis and Optimization n°**2**(1980), 649–687 (1980)

37. Simon, J.: Second variation for domain optimization problems. In: Kappel, F., Kunish, K., Schappacher, W. (eds.) Control and Estimation of Distributed Parameter Systems, International Series of Numerical Mathematics, n° **91**, 361–378 (1989)
38. Sokolowski, J., Zolesio, J. P.: Introduction to shape optimization. Springer, Berlin, Heidelberg, 5–12, (1992)
39. Webster, L.: Convexity. Oxford Science Publications, Oxford University Press, Oxford (1994)
40. Zolésio, J. P.: Sur la localisation d'un domaine, Ph.D. thesis, Universite de Nice, (1973)
41. Zolésio, J. P.: The material derivative (or speed) method for shape optimization. In Optimization of Distributed Parameter Structures, volume 50 of NATO Adv. Study Inst. Ser. E: Appl. Sci., Vol. II, pages 1089-1151. Nijhoff, The Hague, (1981). (Iowa City, Iowa, 1980)
42. Zhu, S., Liu, C., Wu, Q.: Binary level set methods for topology and shape optimization of a two-density inhomogeneous drum. Comput Methods Appl Mech Eng **199**(45–48), 2970–2986 (2010)
43. Zhu, S., Wu, Q., Liu, C.: Variational piecewise constant level set methods for shape optimization of a two-density drum. J. Comput. Phys. **229**(13), 5062–5089 (2010)
44. Zhu, S., Hu, X., Wu, Q.: A level set method for shape optimization in semilinear elliptic problems. J. Comput. Phys. **355**, 104–120 (2018)