Received: 8 October 2022 | Accepted: 1 December 2022

DOI: 10.1002/pamm.202200279

Shape Optimization Algorithms for Fluid Dynamics Applications

Jose Alfonso Pinzon Escobar[1,*] and **Martin Siebenborn**[1]

[1] Universität Hamburg, Fakultät für Mathematik, Informatik und Naturwissenschaften, Fachbereich Mathematik, AM – Angewandte Mathematik, Bundesstraße 55, 20146 Hamburg

In this work we present a comparison between shape optimization algorithms in different vector spaces. The main goal is to optimize the surface of an object with respect to a physical quantity. The main focus is on applications that require large element deformations as part of the optimization process, as for instance the removal and creation of geometric singularities such as edges and corners. The algorithms take into account the prevention of element degeneracy and overlapping, for instance by enforcing inequality constraints. For this purpose, an approach in the Hilbert space is compared to another in Banach spaces. The former is based on a nonlinear extension equation, whereas the p-Laplace operator is used in the latter. Computational results are presented in the context of fluid dynamics applications, where the contour of an object is optimized with respect to the energy dissipation.

1 Introduction

The area of shape optimization [1,2] is used to obtain the contour of an object, potentially by improving on a preexisting design or by starting with a random shape. This optimization process is performed with respect to a physical quantity described by a functional $j(\Omega, v)$, whose value depends on the state variable y and on the domain Ω. A partial differential equation (PDE) can be added as a constraint, and it has to be solved for the state equation. In a summarized manner, this defines the field of PDE-constrained shape optimization. This contribution presents a numerical study of PDE-constrained shape optimization algorithms used in fluid dynamics applications.

Throughout this contribution $D(\cdot)$ denotes the Jacobian, the spatial Euclidean gradient operator is defined as $\nabla(\cdot)$, and the directional derivatives with respect to a specific variable are indicated via, for instance $\frac{\partial}{\partial u}(\cdot)\delta_u$ in direction δ_u. For the shape derivative of the functional $J(\Omega)$ in direction u, we use $J'(\Omega)u$.

2 Model Description

As shown in Fig.1a), the domain Ω represents a flow tunnel with an obstacle Ω_{obs} at the center, the initial shape of the obstacle can be randomly given. The contour of the latter is described by Γ_{obs} and will be optimized with respect to the energy dissipation. The main emphasis of the featured optimization algorithms is on applications that require large deformations of the elements around the surface Γ_{obs}, therefore in our numerical experiments we use an initial configuration that includes geometric singularities, e.g. edges and corners. These have to be smoothed out as part of the optimization process, ideally in a computationally efficient manner. In addition, new edges and corners have to be created to reach an optimum, as is shown in Fig. 1b). Both of these algorithmic aspects have to be implemented without a significant loss of mesh quality, in order to prevent two effects. First, the degradation of mesh quality, which can have a direct effect on the convergence of numerical solvers. Second, overlapping between elements which can lead to non-physical configurations and the failure of numerical methods.

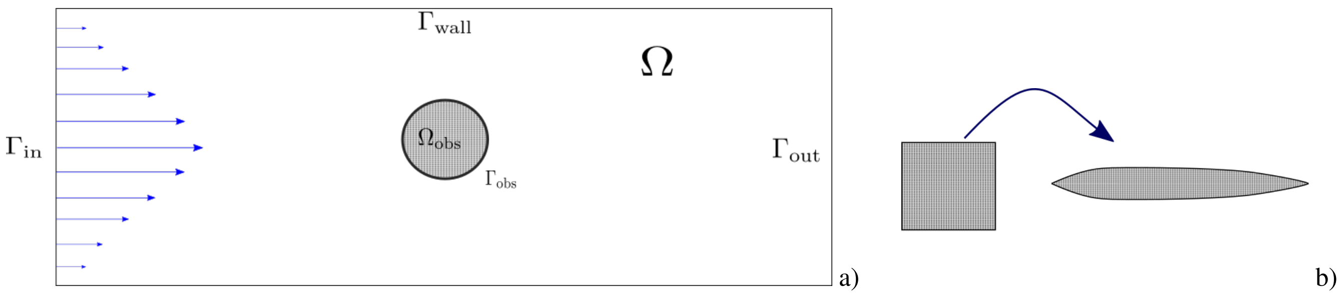

Fig. 1: **a** 2d domain with an obstacle representing a flow tunnel. The initial shape of the obstacle, Γ_{obs}, is optimized with respect to the energy dissipation. There is a given incoming flow, v_∞ at Γ_{in}. **b** For our experiments, an reference configuration with geometric singularities is used. This has to be optimized by removing and creating edges and corners.

* Corresponding author: e-mail jose.pinzon@uni-hamburg.de, phone +49 40 42838 5110

The previously described optimization problem can be formulated as

$$\min_{\Omega\in\mathcal{S}} \quad j(\Omega, v) = \frac{\nu}{2}\int_\Omega D\vec{v} : D\vec{v}\, dx, \tag{1}$$

$$\text{s.t.} \quad e(\Omega, v) = 0 \tag{2}$$

$$g(\Omega) = 0 \tag{3}$$

with the state equation equation (2) corresponds to the incompressible Navier-Stokes equations in strong form

$$\begin{aligned} -\nu\Delta v + (v\cdot\nabla)v + \nabla\mathfrak{p} &= 0 \text{ in } \Omega \\ \operatorname{div} v &= 0 \text{ in } \Omega \\ v &= 0 \text{ on } \Gamma_{\text{obs}} \cup \Gamma_{\text{wall}} \\ v &= v_\infty \text{ on } \Gamma_{\text{in}} \\ Dv\ \cdot n &= \mathfrak{p}n \text{ on } \Gamma_{\text{out}} \end{aligned} \tag{4}$$

where v denotes velocity, $\mathfrak{p}$ the density-specific pressure, ν the viscosity, and v_∞ an inflow velocity profile. Additionally, trivial solutions have to be avoided by preserving the barycenter and volume of the obstacle. This is done by the mapping g, which describes finite dimensional geometric constraints

$$\int_\Omega (\vec{x} + \vec{u}) \det(DF)\, dx = 0, \tag{5}$$

$$\int_\Omega \det(DF) - 1\, dx = 0, \tag{6}$$

imposed on the barycenter and volume, respectively. Where $F(\vec{u})$ is the mapping between the reference and current configurations. These prevent the obstacle from either shrinking to a point -i.e. having no volume thus causing not dissipation, or moving to a part of the domain where the flow velocity is zero, such as Γ_{wall} or outside of Ω.

3 Shape optimization schemes

Shape optimization algorithms based on descent direction methods were proposed in [3–5], where a deformation field generates a series of shape iterates that approach an optimum. This is performed either by iterating through a series of admissible shapes G_{adm} or a series of admissible transformations F_{adm}. These methods have been described in detail in the aforementioned publications. Here we present an overview with the algorithmic aspects that define the descent direction, i.e., the deformation field $\vec{u}$.

3.1 Nonlinear extension equation

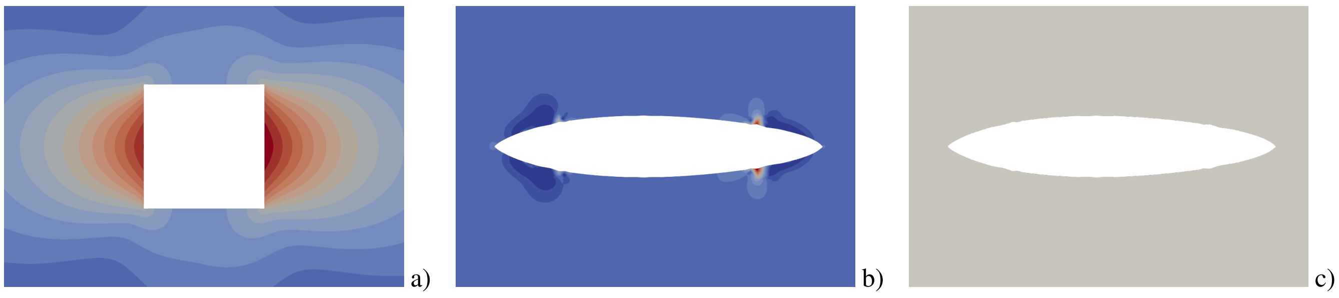

Fig. 2: 2D simulation after 165 optimization steps using a self-adapting nonlinear extension operator $S(\eta, c, \Omega)$. **a** The reference grid is shown with the transformation, **b** η is shown over the areas of the transformed domain where geometrical singularities where created or removed, and **c** the transformed domain is generated only in post-processing.

The variational problem is pulled back to the reference domain via the method of mappings [6]. Thus, it is no longer considered as a problem within a set of admissible shapes, but a set of admissible transformations and the shape optimization problem can be reformulated as an optimal control problem.

At the core of this approach, lies a nonlinear operator $S(\eta, c, \Omega)$ that relates a vector field $\vec{u}$ across the holdall domain Ω to a variable scalar field c that is defined only over the surface of the obstacle, Γ_{obs}, see [3, 4]. It is formulated as finding a deformation field by solving a nonlinear equation, given in its variational form as

$$\int_\Omega (D\vec{u} + D\vec{u}^{\mathrm{T}}) : D\delta_{\vec{u}} + \eta(D\vec{u}\,\vec{u}) \cdot \delta_{\vec{u}}\, dx = \int_{\Gamma_{\text{obs}}} u\vec{n} \cdot \delta_{\vec{u}}\, ds, \tag{7}$$

where $\vec{u} \in W$ such that for all $\delta_{\vec{u}} \in W$ by solving, with $W := \left\{\vec{u} \in H^1(\Omega, \mathbb{R}^d) : \vec{u}|_{\partial\Omega\setminus\Gamma_{\text{obs}}} = 0 \text{ a.e.}\right\}$. In equation (7) it is seen, that the operator S is enriched with an advective nonlinear term to enforce node displacements across the major directions of deformation. Since the aim is to remove and create geometric singularities across the obstacle surface, the variable scalar field η behaves as a nonlinearity control across the domain. It increases wherever singularities are encountered, acting as a self-adapting mechanism for S.

In order to prevent element degeneracy and overlappings between highly deformed elements, an additional constraint has to be imposed. With this in mind, an inequality constraint bounds the determinant of the deformation gradient

$$\det(DF) \geq b \quad \text{in } \Omega. \tag{8}$$

This condition limits the amount of local deformation in each element of the computational mesh. Taking into account that the convergence of the iterative solvers is strongly linked to the mesh quality, the importance of condition equation (8) is particularly evident on the elements that will form the singularities, see Fig. 1, in the direction of the flow, since they will experience the largest deformations.

To formulate an appropriate algorithm, the augmented Lagrangian is given by

$$\begin{aligned}
\mathcal{L}(\vec{u}, \vec{v}, p, c, \eta, \lambda_{\vec{u}}, \lambda_v, \lambda_p, \lambda_{\text{bc}}\lambda_{\text{vol}}) &= \frac{\nu}{2}\int_\Omega (Dv(DF)^{-1}) : (Dv(DF)^{-1}) \det(DF)\, dx \\
&+ \frac{\alpha}{2}\int_{\Gamma_{\text{obs}}} c^2\, ds + \frac{\beta}{2}\int_\Omega ((\eta_{\text{det}} - \det(DF))^+)^2\, dx + \frac{\theta}{2}\int_\Omega \left(\eta - \tfrac{1}{2}(\eta_{\text{ub}} + \eta_{\text{lb}})\right)^2 dx + \tau\|g(\vec{u})\|_2^2 \\
&+ \int_\Omega [\nu(Dv(DF)^{-1}) : (D\lambda_v(DF)^{-1}) + (Dv(DF)^{-1}) \cdot \lambda_v - p\,\mathrm{Tr}(D\lambda_v(DF)^{-1})] \det(DF)\, dx \\
&- \int_\Omega \lambda_p\, \mathrm{Tr}(Dv(DF)^{-1}) \det(DF)\, dx + \int_\Omega [(D\vec{u} + D\vec{u}^\top) : D\lambda_{\vec{u}} + \eta(D\vec{u} \cdot \vec{u})]\, dx \\
&- \int_{\Gamma_{\text{obs}}} c\vec{n} \cdot \lambda_{\vec{u}}\, ds + \lambda_{\text{bc}} \cdot \int_\Omega F(x) \det(DF)\, dx + \lambda_{\text{vol}} \int_\Omega (\det(DF) - 1)\, dx,
\end{aligned} \tag{9}$$

where λ_* are the corresponding Lagrange multipliers, which have to be updated iteratively as part of the algorithm. The penalty method is used to enforce the several constraints, a full description is given in [4] and omitted here. This is used to obtain the necessary optimality conditions to find a minimum, see for instance [7]. The descent direction can be found by solving the different systems of equations obtained by the derivatives of equation (11), and calculating updates for the control variable c, i.e. the variable scalar field defined over the surface of the obstacle.

3.2 p-Laplace operator

This approach is based on finding an approximation to the descent direction u in

$$V_0^\infty := \left\{u \in W^{1,\infty}(\Omega \cup \Omega_{\text{obs}}, \mathbb{R}^d) : \|u\|_{W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)} < 1, u = 0 \text{ a.e. } \Gamma_{\text{in}} \cup \Gamma_{\text{out}} \cup \Gamma_{\text{wall}}\right\},$$

via a relaxation of the p-Laplace operator [8,9] and the corresponding minimization problem

$$\begin{aligned}
\min_{u \in V_0^p} \quad & \frac{1}{p}\int_\Omega (Du : Du)^{p/2}\, dx + J'(\Omega)\, u \\
\text{s.t.} \quad & g(F(\Omega)) = 0, \\
& F = \mathrm{id} + u
\end{aligned} \tag{10}$$

where $p > 2$, with

$$V_0^p = \left\{u \in W^{1,p}(\Omega, \mathbb{R}^d) : \|Du\|_{L^p(\Omega,\mathbb{R}^d)} \leq 1, u = 0 \text{ a.e. on } \Gamma_{\text{in}} \cup \Gamma_{\text{out}} \cup \Gamma_{\text{wall}}\right\}.$$

As before, the mapping g represents the geometrice equality constraints on barycenter and volume. The shape derivative J' is obtained using the method of Céa [10]. An advantage of this approach is that F preserves mesh quality across the whole domain, not only over the surface Γ_{obs}, given that it is a Lipschitz transformation. This means that an additional constraint is not necessary to prevent element degeneracy nor to preserve mesh quality.

As shown in [5], a descent method is proposed using the Lagrangian

$$\begin{aligned} L(u,\lambda) =& \frac{1}{p}\int_\Omega (Du:Du)^{p/2}\,dx + J'(\Omega)\,u \\ &+ \sum_{i=1}^{d}\lambda_i\int_\Omega (x_i+u_i)\det(DF)\,dx + \lambda_{d+1}\int_\Omega \det(DF)-1\,dx \end{aligned} \tag{11}$$

with $\lambda = (\lambda_1,\ldots,\lambda_d,\lambda_{d+1})^T$, the finite dimensional vector of Lagrangian multipliers associated to the geometric constraints.

Using the derivatives with respect to u in the direction $\mu_u \in V_0^p$ and λ into direction $\mu_\lambda \in \mathbb{R}^{d+1}$ we can obtain the nonlinear optimality system

$$\begin{aligned} \frac{\partial}{\partial u}L(u,\lambda)\mu_u &= 0 \qquad \forall\ \mu_u \in V_0^p \\ \frac{\partial}{\partial \lambda}L(u,\lambda)\mu_\lambda &= 0 \qquad \forall\ \mu_\lambda \in \mathbb{R}^{d+1} \end{aligned} \tag{12}$$

where the geometric constraints have been incorporated to the descent directions. The solution of the nonlinear system of equations equation (12) gives us a deformation field in $W^{1,p}$, as well as the necessary Lagrange multipliers to preserve the constraints g on every step. This means that an external mechanism for the update of the Lagrange multipliers is no longer necessary, as in the augmented Lagrangian approach. Compared to equation (9), the Lagrangian in equation (11) represents also a simpler implementation without the need to heuristically determine penalty terms, which are majorly grid dependent.

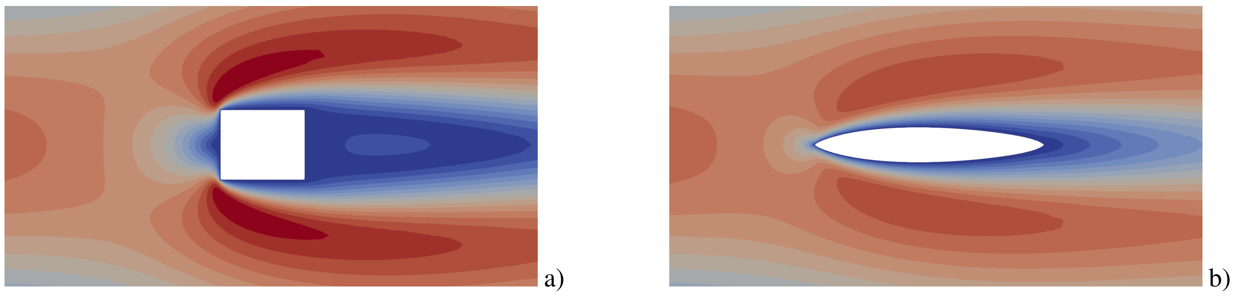

Fig. 3: A simulation using the p-Laplacian approach for 41 optimization steps. The streamlines show the flow around obstacle. A value of $p = 4.8$ was used. **a** The initial configuration generates drag around Γ_{obs}. **b** An optimum is achieved and the geometric singularities were removed and created with a deformation field in a suitable Banach space.

A descent method algorithm can be formulated using the linearization of equation (12), whose solution can be obtained Newton's method. The caveat is that the relaxed p-Laplacian problem has to be solved for incremental values of p up to a given $p_{\max}$, on every optimization step. This is done to have a good initial guess in each increase, because for high values of p, the problem is highly nonlinear. Moreover, the linearization is a saddle point problem, which was solved with the Schur complement matrix in [5] and required the effect of A^{-1} several times.

4 Numerical results and discussion

In this section, we present numerical results for the shape optimization schemes described in section 3. In order to compare the schemes, care was taken that the same grid and refinement levels were used. These simulations were carried out using UG4 [11], in the distributed-memory system `LISE`. This was done until convergence of the simulation or after the end of the assign walltime, whichever occurs first. All linear problems were solved using the BiCGStab method with a multigrid preconditioner.

The energy dissipation for the extension equation and p-Laplace approaches are shown in Fig. 4. Given that the extension equation algorithm does not incorporate the geometric constraints into the descent direction, the obtained shapes will not necessarily be optimal with respect to the cost function. This is only guaranteed locally for converged values of the Lagrange multipliers, so every major update of these will lead to the observed jumps on the plot and can lead to counter intuitive shapes, such as the one shown in Fig. 5. It is seen how the p-Laplace based algorithm has faster convergence. Less steps are needed to achieve an optimum and to remove the singularities of the reference configuration. Also, since every optimization step fulfills the geometric constraints, no bouncing is observed in the shapes.

The initial steps of a 2d simulation are used to compare the computational performance of the two schemes. Iteration counts are presented for the routine necessary to calculate the deformation field $\vec{u}$, and the total iterations represent the complete

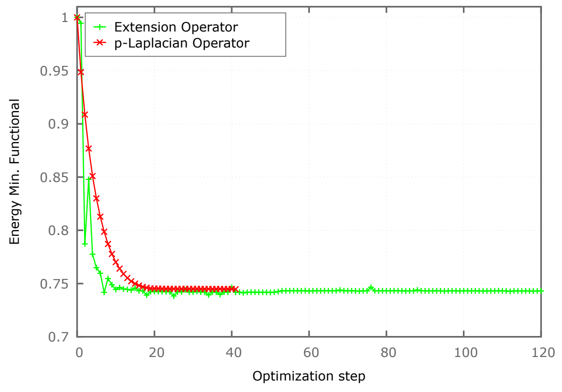

Fig. 4: The energy dissipation functional is plotted for 2d simulations of the algorithms described in Sec. 3. The number of optimization steps needed to achieve an optimum widely varies between the two schemes.

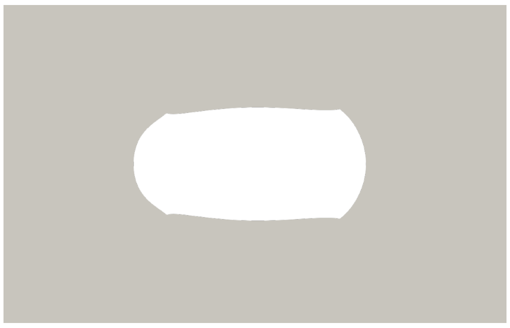

Fig. 5: The set of admissible transformations related the extension equation approach depends on the values of the Lagrange multipliers, this might result in shapes that do not fully preserve the geometric constraints. The achieved optimum is guaranteed to fulfill all the imposed constraints.

computational effort within an optimization step. The wall times are given per step, however the solution of the state equation and its adjoint are not included in this calculation. In Tables 1 and 2, it can be seen how the computational effort per step of the Hilbert space approach is lower than the p-Laplace algorithm. This is mainly related to the need to compute the effect of A^{-1} several times for each value of p to solve equation (12), which results in both a high number of iterations and a larger time per step.

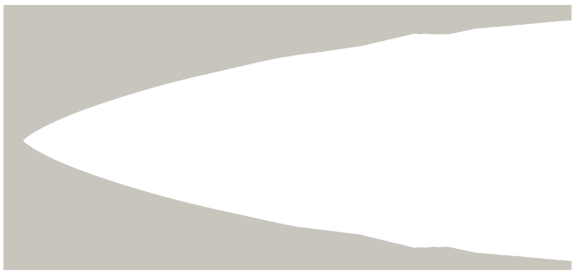

Fig. 6: Extension equation. After 165 optimization steps, the geometric singularities of the reference configuration are still visible. This shows that a lot of computational effort was spent on trying to remove them.

Table 1: Iteration counts are provided for the Newton's method used to solve equation (7), and its underlying linearization. The fourth column shows the linear iterations needed for all the solvers in one optimization step. The time in [s] is shown per step. The last rows show the average and sum for the first 6 optimization steps. Times and iteration counts for the state equation and its adjoint are omitted.

Step	Newton	Lin. Its.	Tot. Its.	Time[s]
0	1	0	30	0.08
1	4	13	53	0.12
2	4	13	53	0.12
3	4	12	51	0.12
4	4	12	51	0.11
5	4	12	51	0.11
AVG	4	11	48	0.11
SUM	21	113	55	0.65

Comparing Figs. 6 and 7, the computational effort needed to remove the reference configuration's geometric singularities is observed. Although the extension equation requires less time per step, it is seen that overall it requires a dramatically larger number of steps to smooth the obstacle's surface. In contrast, the p-Laplace based algorithm achieves this within 41 steps as seen in Fig. 7. Thus, we can say there is a tradeoff. On one hand, the time and computational effort are large in the p-Laplace operator approach, compared to the Hilbert space method that uses the extension equation. On the other, less steps are necessary to reach an optimum by removing the initially present edges and corners.

5 Conclusion

The computational aspects and performance of two shape optimization algorithmic schemes were compared in this work. It was shown how an approach in suitable Banach vector spaces incorporates the geometric constraints to the descent direction and reduces the necessary steps needed for the removal of geometric singularities present in the reference configuration. Additionally, it allows for an implementation with no domain-dependent parameters. The extension equation approach lead to a better performance, in terms of number of iterations and time to solution. It was studied how the method requires a higher number of steps to smooth the reference configuration. Moreover, the dependence on heuristically determined parameters and on the convergence of the Lagrange multipliers adds complexity to the formulation of the algorithm as well as to its

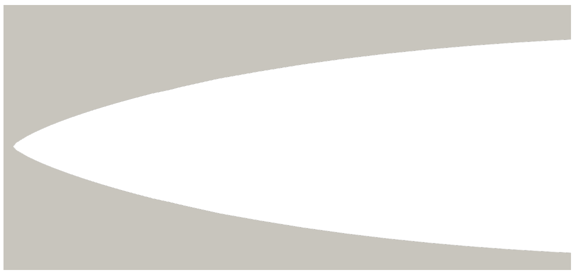

Fig. 7: p**-Laplace operator**. After 41 steps the edges and corners of the reference configuration have been fully removed.

Table 2: Iteration counts are provided for the Newton's method used to solve equation (12), and its underlying linearization. The fourth column shows the linear iterations compute the effect of A^{-1}. The time in [s] is shown per step. The last rows show the average and sum for the first 6 optimization steps. Times and iteration counts for the state equation and its adjoint are omitted.

Step	Newton	Lin. Its.	Tot. Its.	Time[s]
0	80	264	2209	5.75
1	80	249	2205	5.73
2	80	248	2208	5.71
3	80	249	2234	5.81
4	80	251	2251	5.76
5	80	252	2279	5.79
AVG	80	253	2231	5.73
SUM	480	113	13386	34.55

implementation. In contrast to the p-Laplacian approach, where a simplified implementation was obtained at the cost of more computational effort.

Acknowledgements The research took place within the "LFF-GK11 Simulation-Based Design Optimization of Dynamic Systems Under Uncertainties (SENSUS)" funded by the State of Hamburg. The authors gratefully acknowledge this support. Selected computations were performed with resources provided by the North-German Super-computing Alliance (HLRN). Open access funding enabled and organized by Projekt DEAL.

References

[1] J. Sokolowski and J. P. Zolésio, Introduction to Shape Optimization (Springer, Berlin, Heidelberg, 1992).
[2] B. Mohammadi and O. Pironneau, Applied Shape Optimization for Fluids (Oxford University Press, 2009).
[3] S. Onyshkevych and M. Siebenborn, Journal of Optimization Theory and Applications **189**, 291–316 (2020).
[4] J. Pinzon and M. Siebenborn, Fluid dynamic shape optimization using self-adapting nonlinear extension operators with multigrid preconditioners, 2022.
[5] P. M. Müller, J. Pinzon, T. Rung, and M. Siebenborn, A scalable algorithm for shape optimization with geometric constraints in banach spaces, 2021.
[6] J. Haubner, M. Siebenborn, and M. Ulbrich, SIAM **43**(3), A1997–A2018 (2021).
[7] M. Hinze, R. Pinnau, M. Ulbrich, and S. Ulbrich, Optimization with PDE Constraints, Mathematical Modelling: Theory and Applications, Vol. 23 (Springer, 2009).
[8] H. Ishii and P. Loreti, SIAM journal on mathematical analysis **37**(2), 411 – 437 (2005).
[9] K. Deckelnick, P. Herbert, and M. Hinze, ESAIM: COCV **28**(2) (2022).
[10] G. Allaire, C. Dapogny, and F. Jouve, Chapter 1 - shape and topology optimization, in: Geometric Partial Differential Equations - Part II, , Handbook of Numerical Analysis, Vol. 22 (Elsevier, 2021), pp. 1 – 132.
[11] A. Vogel, S. Reiter, M. Rupp, A. Nägel, and G. Wittum, Comp. Vis. Sci. **16**(4), 165–179 (2013).