ESAIM: COCV 18 (2012) 1122–1149
DOI: 10.1051/cocv/2011192

ESAIM: Control, Optimisation and Calculus of Variations
www.esaim-cocv.org

ADAPTIVE FINITE ELEMENT METHOD FOR SHAPE OPTIMIZATION [*]

Pedro Morin[1], Ricardo H. Nochetto[2], Miguel S. Pauletti[3] and Marco Verani[4]

Abstract. We examine shape optimization problems in the context of inexact sequential quadratic programming. Inexactness is a consequence of using adaptive finite element methods (AFEM) to approximate the state and adjoint equations (*via* the dual weighted residual method), update the boundary, and compute the geometric functional. We present a novel algorithm that equidistributes the errors due to shape optimization and discretization, thereby leading to coarse resolution in the early stages and fine resolution upon convergence, and thus optimizing the computational effort. We discuss the ability of the algorithm to detect whether or not geometric singularities such as corners are genuine to the problem or simply due to lack of resolution – a new paradigm in adaptivity.

Mathematics Subject Classification. 49M25, 65M60.

Received July 1st, 2011. Revised September 19, 2011
Published online 16 January 2012.

1. Shape optimization as adaptive sequential quadratic programming

We consider shape optimization problems for partial differential equations (PDE) that can be formulated as follows: we denote with $u = u(\Omega)$ the solution of a PDE in the domain Ω of $\mathbb{R}^d$ $(d \geq 2)$,

$$\mathcal{B}u(\Omega) = f, \tag{1.1}$$

Keywords and phrases. Shape optimization, adaptivity, mesh refinement/coarsening, smoothing.

[*] *Partially supported by UNL through GRANT CAI+D 062-312, by CONICET through Grant PIP 112-200801-02182, by MinCyT of Argentina through Grant PICT 2008-0622 and by Argentina-Italy bilateral project "Innovative numerical methods for industrial problems with complex and mobile geometries".*
Partially supported by NSF grants DMS-0505454 and DMS-0807811.
Partially supported by NSF grants DMS-0505454 and DMS-0807811, and by Award No. KUS-C1-016-04, made by King Abdullah University of Science and Technology (KAUST).
Partially supported by Italian MIUR PRIN 2008 "Analisi e sviluppo di metodi numerici avanzati per EDP" and by Argentina-Italy bilateral project "Innovative numerical methods for industrial problems with complex and mobile geometries".

[1] Departamento de Matemática, Facultad de Ingeniería Química and Instituto de Matemática Aplicada del Litoral, Universidad Nacional del Litoral, CONICET, Santa Fe, Argentina. pmorin@santafe-conicet.gov.ar; www.imal.santafe-conicet.gov.ar/pmorin

[2] Department of Mathematics and Institute for Physical Science and Technology, University of Maryland, College Park, USA. rhn@math.umd.edu; www.math.umd.edu/~rhn

[3] Department of Mathematics and Institute for Applied Mathematics and Computational Science, Texas A&M University, College Station, 77843 TX, USA. pauletti@math.tamu.edu; www.math.tamu.edu/~pauletti

[4] MOX – Modelling and Scientific Computing – Dipartimento di Matematica "F. Brioschi", Politecnico di Milano, Milano, Italy. marco.verani@polimi.it; mox.polimi.it/~verani

which we call the *state equation*. Given a cost functional $J[\Omega] = J[\Omega, u(\Omega)]$, which depends on Ω itself and the solution $u(\Omega)$ of the state equation, we consider the minimization problem

$$\Omega^* \in \mathcal{U}_{\rm ad}: \qquad J[\Omega^*, u(\Omega^*)] = \inf_{\Omega\in\mathcal{U}_{\rm ad}} J[\Omega, u(\Omega)], \tag{1.2}$$

where $\mathcal{U}_{\rm ad}$ is a set of admissible domains in $\mathbb{R}^d$. We view this as a constrained minimization problem, (1.1) being the constraint. The goal of this paper is, assuming the existence of a local minimizer Ω of (1.1)–(1.2), to formulate and test a practical and efficient computational algorithm that adaptively builds a sequence of domains $\{\Omega_k\}_{k\geq 0}$ converging to Ω. Coupling adaptivity with shape optimization seems to be important but rather novel.

To achieve this goal we will define an adaptive sequential quadratic Programming algorithm (`ASQP`). To motivate and briefly describe the ideas underlying `ASQP`, we need the concept of shape derivative $\delta_\Omega J[\Omega; \mathbf{v}]$ of $J[\Omega]$ in the direction of a velocity $\mathbf{v}$, which usually satisfies

$$\delta_\Omega J[\Omega; \mathbf{v}] = \int_\Gamma g(\Omega)\mathrm{v}\,\mathrm{d}S = \langle g(\Omega), v\rangle_\Gamma, \tag{1.3}$$

where $\mathrm{v} = \mathbf{v}\cdot\boldsymbol{\nu}$ is the normal component of $\mathbf{v}$ to $\Gamma = \partial\Omega$, the boundary of Ω, and $g(\Omega)$ is the *Riesz representation* of the shape derivative. We postpone the precise definition of (1.3) until Section 2.2. We will see later that $g(\Omega)$ depends on $u(\Omega)$ and on the solution $z(\Omega)$ of an adjoint equation. We present `ASQP`in two steps: we first introduce an infinite dimensional sequential quadratic programming (∞-`SQP`) algorithm, which is an ideal but impractical algorithm, and next we discuss its adaptive finite dimensional version, which is responsible for the inexact nature of `ASQP`that renders it practical.

Exact SQP algorithm. To describe the ∞-`SQP`algorithm, we let Ω_k be the current iterate and Ω_{k+1} be the new one. We let $\Gamma_k := \partial\Omega_k$ and $\mathbb{V}(\Gamma_k)$ be a Hilbert space defined on Γ_k. We further let $A_{\Gamma_k}[\cdot,\cdot] : \mathbb{V}(\Gamma_k)\times\mathbb{V}(\Gamma_k) \to \mathbb{R}$ be a symmetric continuous and coercive bilinear form, which induces the norm $\|\cdot\|_{\mathbb{V}(\Gamma_k)}$ and gives rise to the elliptic selfadjoint operator $\mathcal{A}_k$ on Γ_k defined by $\langle\mathcal{A}_k\mathrm{v}, \mathrm{w}\rangle_{\Gamma_k} = A_{\Gamma_k}[\mathrm{v}, \mathrm{w}]$. We then consider the following *quadratic model* $Q_k : \mathbb{V}(\Gamma_k) \to \mathbb{R}$ of $J[\Omega]$ at Ω_k

$$Q_k(\mathrm{w}) := J[\Omega_k] + \delta_\Omega J[\Omega_k; \mathbf{w}] + \frac{1}{2}A_{\Gamma_k}[\mathrm{w}, \mathrm{w}]. \tag{1.4}$$

We denote by v_k the minimizer of $Q_k(\mathrm{w})$, which satisfies

$$\mathrm{v}_k \in \mathbb{V}(\Gamma_k): \qquad A_{\Gamma_k}[\mathrm{v}_k, \mathrm{w}] = -\langle g_k, \mathrm{w}\rangle_{\Gamma_k} \qquad \forall \mathrm{w} \in \mathbb{V}(\Gamma_k), \tag{1.5}$$

with $g_k := g(\Omega_k)$. It is easy to check that v_k given by (1.5) is the unique minimizer of $Q_k(\mathrm{w})$ and the coercivity of the form $A_{\Gamma_k}(\cdot,\cdot)$ implies that v_k is an admissible descent direction, *i.e.* $\delta_\Omega J[\Omega_k; \mathbf{v}_k] < 0$, unless $\mathrm{v}_k = 0$, in which case we are at a stationary point of $J[\Omega]$. We remark that using (1.5) is classical in the literature of shape optimization; see *e.g.* [9, 11, 15, 22].

Once v_k has been found on Γ_k, we need to determine a vector field $\mathbf{v}_k$ in Ω_k so that $\mathbf{v}_k\cdot\boldsymbol{\nu}_k = \mathrm{v}_k$ on Γ_k, along with a suitable stepsize μ so that the updated domain $\Omega_{k+1} = \Omega_k + \mu\mathbf{v}_k := \{\mathbf{y}\in\mathbb{R}^d : \ \mathbf{y} = \mathbf{x} + \mu\mathbf{v}_k(\mathbf{x}), \ \mathbf{x}\in\Omega_k\}$ gives a significant decrease of the functional value $J[\Omega_k]$. We are now ready to introduce the exact (infinite dimensional) sequential quadratic programming algorithm (∞-`SQP`) for solving the constrained optimization problem (1.1)–(1.2):

```
$\infty$-SQP Algorithm
Given the initial domain $\Omega_0$, set $k = 0$ and repeat the following steps:

(1) Compute $u_k = u(\Omega_k)$ by solving (1.1)
(2) Compute the Riesz representation $g_k = g(\Omega_k)$ of (1.3)
(3) Compute the search direction $\mathrm{v}_k$ by solving (1.5) and extend it to $\mathbf{v}_k$
(4) Determine the stepsize $\mu_k$ by line search
(5) Update: $\Omega_{k+1} = \Omega_k + \mu_k\mathbf{v}_k$;   $k := k + 1$
```

The ∞-SQP algorithm is not feasible as it stands, because it requires the exact computation of the following quantities at each iteration:

- the solution u_k to the state equation (1.1);
- the solution z_k to the adjoint equation which in turn defines g_k;
- the solution v_k to problem (1.5);
- the values of the functional J in the line search routine, which in turn depend on u_k.

Adaptive SQP algorithm (ASQP). In order to obtain a practical algorithm, we replace all of the above non-computable operations by finite approximations. This leads to the adaptive sequential quadratic Programming algorithm (ASQP), which adjusts and balances the accuracies of the various approximations along the iteration. It is worth noticing that the adaptive algorithm has to deal with two distinct main sources of error: the approximation of the PDE (*PDE error*) and the approximation of the domain geometry (*geometric error*). We observe that the approximation of (1.1) and the values of the functional J and of its derivative relate to the PDE Error, whereas the approximation of (1.5) and domain update lead to the geometric error. Since it is wasteful to impose a PDE error finer than the expected geometric error, we devise a natural mechanism to balance the computational effort.

The ASQP algorithm is an iteration of the form

$$\mathcal{E}_k \rightarrow \texttt{APPROXJ} \rightarrow \texttt{SOLVE} \rightarrow \texttt{RIESZ} \rightarrow \texttt{DIRECTION} \rightarrow \texttt{LINESEARCH} \rightarrow \texttt{UPDATE} \rightarrow \mathcal{E}_{k+1},$$

where $\mathcal{E}_k = \mathcal{E}_k(\Omega_k, \mathbb{S}_k, \mathbb{V}_k)$ is the total error incurred in at step k, $\mathbb{S}_k = \mathbb{S}_k(\Omega_k)$ is the finite element space defined on Ω_k and $\mathbb{V}_k = \mathbb{V}_k(\Gamma_k)$ is the finite element space defined on the boundary Γ_k. We now describe briefly each module along with the philosophy behind ASQP. Let G_k be an approximation to the shape derivative $g(\Omega_k)$, let $\mathrm{v}_k \in \mathbb{V}(\Gamma_k)$ be the exact solution of (1.5) on Γ_k and let $V_k \in \mathbb{V}_k(\Gamma_k)$ be its finite element approximation.

The discrepancy between v_k and V_k leads to the geometric error. Upon using a first order Taylor expansion around Ω_k, together with (1.5) for the exact velocity v_k, we obtain

$$\left|J[\Omega_k + \mu_k \mathbf{V}_k] - J[\Omega_k + \mu_k \mathbf{v}_k]\right| \approx \mu_k \left|\delta_\Omega J[\Omega_k; V_k - \mathrm{v}_k]\right| = \mu_k \left|A_{\Gamma_k}[\mathrm{v}_k, V_k - \mathrm{v}_k]\right| \leq \mu_k \|\mathrm{v}_k\|_{\Gamma_k} \|\mathrm{v}_k - V_k\|_{\Gamma_k}.$$

Motivated by this expression, we now define two modules, APPROXJ and DIRECTION, in which adaptivity is carried out. These modules are driven by different adaptive strategies and corresponding different tolerances, and tolerance parameters γ (PDE) and θ (geometry). Their relative values allow for different distributions of the computational effort in dealing with the PDE and the geometry.

The routine DIRECTION enriches/coarsens the space $\mathbb{V}_k$ to control the quality of the descent direction

$$\|V_k - \mathrm{v}_k\|_{\Gamma_k} \leq \theta \|V_k\|_{\Gamma_k}, \tag{1.6}$$

which implies that

$$\frac{\langle V_k, \mathrm{v}_k\rangle_{\Gamma_k}}{\|V_k\|_{\Gamma_k}\|\mathrm{v}_k\|_{\Gamma_k}} \geq \sqrt{1-\theta^2},$$

and thus the cosine of the angle between V_k and v_k (as elements of $\mathbb{V}(\Gamma_k)$) is bounded below by $\cos \pi/6 = \sqrt{3}/2$ provided $\theta \leq 1/2$. This guarantees that the angle between the directions V_k and v_k is $\leq \pi/6$. Besides $(1-\theta)\|V_k\|_{\Gamma_k} \leq \|\mathrm{v}_k\|_{\Gamma_k} \leq (1+\theta)\|V_k\|_{\Gamma_k}$, which implies a geometric error proportional to $\mu_k \|V_k\|^2_{\Gamma_k}$, namely

$$\left|J[\Omega_k + \mu_k \mathbf{V}_k] - J[\Omega_k + \mu_k \mathbf{v}_k]\right| \leq \delta \mu_k \|V_k\|^2_{\Gamma_k}, \tag{1.7}$$

with $\delta := \theta(1+\theta) \leq \frac{3}{2}\theta$. Adaptivity in the module DIRECTION is guided by *a posteriori* estimators for the energy error given by the bilinear form $A_{\Gamma_k}[\cdot,\cdot]$. In the applications of Sections 5 and 6, $\mathcal{A}_k$ is related to the Laplace-Beltrami operator over Γ_k.

On the other hand, the module APPROXJ enriches/coarsens the space $\mathbb{S}_k$ to control the error in the approximate functional value $J_k[\Omega_k + \mu_k \mathbf{V}_k]$ to the prescribed tolerance $\gamma\mu_k\|V_k\|^2_{\Gamma_k}$,

$$\left|J[\Omega_k + \mu_k\mathbf{V}_k] - J_k[\Omega_k + \mu_k\mathbf{V}_k]\right| \leq \gamma\mu_k\|V_k\|^2_{\Gamma_k}, \tag{1.8}$$

where $\gamma = \frac{1}{2} - \delta \geq \delta$ prevents excessive numerical resolution relative to the geometric one; this is feasible if $\theta \leq 1/5$. Adaptivity in APPROXJ is guided by the *dual weighted residual* method (DWR) [3,6], taylored to the approximation of the functional value J, instead of the usual energy estimators.

The remaining modules perform the following tasks. The module SOLVE finds finite element solutions $U_k \in \mathbb{S}_k$ of (1.1) and $Z_k \in \mathbb{S}_k$ of an adjoint equation (necessary for the computation of the shape derivative $g_k = g(\Omega_k)$), while RIESZ builds on $\mathbb{S}_k$ an approximation G_k to g_k. Finally, the module LINESEARCH finds an optimal stepsize μ while using, if necessary, Lagrange multipliers to enforce domain constraints present in the definition of $\mathcal{U}_{\text{ad}}$.

Energy decrease. The triangle inequality, in conjunction with conditions (1.7) and (1.8), yields

$$\left|J[\Omega_k + \mu_k\mathbf{V}_k] - J_k[\Omega_k + \mu_k\mathbf{v}_k]\right| \leq \frac{1}{2}\mu_k\|V_k\|^2_{\Gamma_k}, \tag{1.9}$$

which is a bound on the local error incurred in at step k. However, the exact energy decrease reads

$$J[\Omega_k] - J[\Omega_k + \mu_k\mathbf{v}_k] \approx -\mu_k\delta_\Omega J[\Omega_k;\mathbf{v}_k] = \mu_k A_{\Gamma_k}[\mathrm{v}_k,\mathrm{v}_k] = \mu_k\|\mathrm{v}_k\|^2_{\Gamma_k} \geq (1-\theta)^2\mu_k\|V_k\|^2_{\Gamma_k}, \tag{1.10}$$

and leads to the further constraint $(1-\theta)^2 > \frac{1}{2}$ to guarantee the energy decrease

$$J_k[\Omega_k + \mu_k\mathbf{V}_k] < J[\Omega_k].$$

Consistency. If ASQP converges to a stationary point, *i.e.* $\mu_k\|V_k\|^2_{\Gamma_k} \to 0$ as $k \to \infty$, then the modules DIRECTION and APPROXJ approximate the descent direction V_k and functional $J[\Omega_k]$ increasingly better as $k \to \infty$, as dictated by (1.6) and (1.8). In other words, this imposes *dynamic* error tolerance and progressive improvement in approximating U_k, Z_k and G_k as $k \to \infty$.

We observe that (1.8) is not a very demanding test for DWR. So we expect coarse meshes at the beginning, and a combination of refinement and coarsening later as DWR detects geometric singularities, such as corners, and sorts out whether they are genuine to the problem or just due to lack of numerical resolution. This aspect of our approach is a novel paradigm in adaptivity, resorts to ideas developed in [8], and is documented in Sections 5 and 6.

Prior work. The idea of coupling FEM, *a posteriori* error estimators and optimal design error estimators to efficiently solve shape optimization problems is not new. The pioneering work [4] presents an iterative scheme, where the Zienkiewicz-Zhu error indicator and the L^2 norm of the shape gradient are both used at each iteration to improve the PDE error and the geometric error, respectively. However, the algorithm in [4] does not resort to any dynamically changing tolerance, that would allow, as it happens for ASQP, to produce coarse meshes at the beginning of the iteration and a combination of geometric and PDE refinement/coarsening later on. Moreover, [4] does not distinguish between fake and genuine geometric singularities that may arise on the domain boundary during the iteration process, and does not allow the former to disappear. More recently, the use of adaptive modules for the numerical approximation of PDEs has been employed by several authors [2,29,30] to improve the accuracy of the solution of shape optimization problems. However, in these papers the critical issue of linking the adaptive PDE approximation with an adaptive procedure for the numerical treatment of the domain geometry is absent. We address this linkage below.

Outline. The rest of this paper is organized as follows. In Section 2 we introduce the Lagrangian $\mathcal{L}$ for the constrained minimization problem (1.1)–(1.2) and derive the adjoint equation and shape derivative of $\mathcal{L}$. In Section 3 we introduce the finite element discretization along with a brief summary of DWR and a novel error estimate. In Section 4 we present in detail the ASQP algorithm, and discuss its several building blocks. We next apply ASQP to two benchmark problems for viscous incompressible fluids governed by the Stokes equations. We examine drag minimization in Section 5 and aortic-coronary by-pass optimization in Section 6. In both sections we derive the shape derivative as well as the full expression of the dual weighted residual estimate. We also document the performance of ASQPwith several interesting numerical simulations, which were implemented within ALBERTA [31] and postprocessed with PARAVIEW [19]. We end this paper in Section 7 with some conclusions.

2. Lagrangian formalism

2.1. State and adjoint equations

We consider a (nonlinear) functional $J[\Omega, u(\Omega)]$ depending on a domain Ω and the solution $u = u(\Omega)$ of a *state* equation, which is a (linear) PDE defined in Ω. In strong form it reads $\mathcal{B}u = f$ and in weak form can be written as follows:

$$u \in \mathbb{S}: \quad B[u, w] = \langle f, w\rangle \qquad \forall w \in \mathbb{S}. \tag{2.1}$$

Here $\mathbb{S}$ is a Hilbert space, $\mathbb{S}^*$ is its dual, $\mathcal{B} : \mathbb{S} \to \mathbb{S}^*$ is a linear isomorphism, and B is the corresponding bilinear form. Therefore, B is continuous and satisfies the inf-sup condition

$$\inf_{w\in\mathbb{S}} \sup_{v\in\mathbb{S}} \frac{B[v, w]}{\|v\|\|w\|} = \inf_{v\in\mathbb{S}} \sup_{w\in\mathbb{S}} \frac{B[v, w]}{\|v\|\|w\|} > 0.$$

If $f \in \mathbb{S}^*$, then (2.1) has a unique solution $u = u(\Omega)$. Our goal is to minimize $J[\Omega, u(\Omega)]$ always maintaining the state constraint (2.1) in the process. To this end, we introduce the *Lagrangian*

$$\mathcal{L}[\Omega, u, z] := J[\Omega, u] - B[u, z] + \langle f, z\rangle, \tag{2.2}$$

for $u, z \in \mathbb{S}$. The *adjoint* variable z is a Lagrange multiplier for (2.1).

The first order stationarity conditions, namely the state and adjoint equations for (u, z), read

$$\langle \delta_z \mathcal{L}[\Omega, u, z], w\rangle = 0, \tag{2.3}$$
$$\langle \delta_u \mathcal{L}[\Omega, u, z], w\rangle = 0, \tag{2.4}$$

for all test functions $w \in \mathbb{S}$. Equations (2.3) and (2.4) imply respectively for all $w \in \mathbb{S}$

$$u = u(\Omega) \in \mathbb{S}: \quad B[u, w] = \langle f, w\rangle, \tag{2.5}$$
$$z = z(\Omega) \in \mathbb{S}: \quad B[w, z] = \langle \delta_u J[\Omega, u], w\rangle, \tag{2.6}$$

which are the weak forms of state equation $\mathcal{B}u = f$ and adjoint equation $\mathcal{B}^* z = \delta_u J[\Omega, u]$. Therefore, if we enforce (2.5), then the Lagrangian reduces to the cost functional

$$\mathcal{L}[\Omega, u, z] = J[\Omega, u] \tag{2.7}$$

no matter whether Ω is a minimizer or not. This is useful to construct a descent direction for $J[\Omega, u]$ *via* $\mathcal{L}[\Omega, u, z]$, perhaps using a discrete gradient flow.

2.2. Shape derivatives

To construct a descent direction we need $\delta_\Omega J[\Omega, u]$, which may not necessarily vanish unless we are already at a stationary point. We now recall a basic rule for shape differentiation. If $\phi = \phi(x)$ does not depend on Ω and

$$J[\Omega] = \int_\Omega \phi \, \mathrm{d}x$$

then the shape derivative of $I[\Omega]$ in the direction $\mathbf{V}$ is given by [32], Propositions 2.45, 2.50 and (2.145),

$$\langle \delta_\Omega J[\Omega], \mathbf{V} \rangle = \int_\Gamma \phi V \, \mathrm{d}S \tag{2.8}$$

where $V = \mathbf{V} \cdot \boldsymbol{\nu}$ is the normal velocity to Γ. This is unfortunately not enough: $\mathcal{L}$ also involves integrals of functions which solve PDE in Ω, such as $u(\Omega)$ and $z(\Omega)$. If $\phi(\Omega, x)$ also depends on Ω, then

$$\langle \delta_\Omega J[\Omega], \mathbf{V} \rangle = \int_\Omega \phi'(\Omega; \mathbf{V}) + \int_\Gamma \phi V \, \mathrm{d}S \tag{2.9}$$

where $\phi'(\Omega; \mathbf{V})$ stands for the shape derivative of $\phi(\Omega, x)$ in the direction $\mathbf{V}$ [32], Sections 2.31–2.33. This requires computing the shape derivatives of the state and adjoint variables in the direction $\mathbf{V}$, namely $u'(\Omega; \mathbf{V})$ and $z'(\Omega; \mathbf{V})$, which will be solutions of elliptic boundary value problems.

To render the discussion concrete, let $u(\Omega)$ solve the *Dirichlet* problem

$$\mathcal{B}u(\Omega) = f \quad \text{in } \Omega, \qquad u(\Omega) = \ell \quad \text{on } \partial\Omega, \tag{2.10}$$

with $\mathcal{B}$ a linear second order selfadjoint operator in $\mathbb{S}$ and $f \in L^2(\mathbb{R}^d)$, $\ell \in H^1(\mathbb{R}^d)$ independent of Ω. The shape derivative $u'(\mathbf{V}) := u'(\Omega; \mathbf{V})$ is the solution to the following Dirichlet problem [32]

$$\mathcal{B}u'(\mathbf{V}) = 0 \quad \text{in } \Omega, \qquad u'(\mathbf{V}) = -\nabla(u - \ell) \cdot \mathbf{V} \quad \text{on } \partial\Omega. \tag{2.11}$$

To obtain this expression it is first necessary to extend $u(\Omega)$ to a larger domain, for example by setting it equal to ℓ outside Ω and thus $u(\Omega) \in H^1(\mathbb{R}^d)$; see details in [12, 32].

Finally, the shape derivative of $J[\Omega, u(\Omega)]$ can be computed by means of the usual chain rule. For example, if $\phi(\Omega, x) = \frac{1}{2} u(\Omega, x)^2$ with $u(\Omega, x)$ solution to (2.10), then (2.9) yields

$$\delta_\Omega J[\Omega; \mathbf{V}] = \int_\Omega u(\Omega)\, u'(\mathbf{V}) \, \mathrm{d}x + \int_{\partial\Omega} \frac{1}{2} u(\Omega)^2 V \, \mathrm{d}S. \tag{2.12}$$

The computation of $\delta_\Omega \mathcal{L}[\Omega, u(\Omega), z(\Omega)]$ resorts once more to the chain rule

$$\langle \delta_\Omega \mathcal{L}[\Omega, u(\Omega), z(\Omega)], \mathbf{V} \rangle = \langle \delta_\Omega \mathcal{L}[\Omega, u, z], \mathbf{V} \rangle + \langle \delta_u \mathcal{L}[\Omega, u, z], u'(\mathbf{V}) \rangle + \langle \delta_z \mathcal{L}[\Omega, u, z], z'(\mathbf{V}) \rangle, \tag{2.13}$$

where on the right-hand side we regarded the variables Ω, u, z as independent. If either $u'(\mathbf{V})$ or $z'(\mathbf{V})$ were admissible test functions, then either the second or third term would vanish in light of (2.3) and (2.4). However, this is not the case when u, z satisfy a Dirichlet problem such as (2.10) and their shape derivatives $u'(\mathbf{V})$, $z'(\mathbf{V})$ have a non-vanishing trace dictated by (2.11).

There is however an approach to circumvent computing $u'(\mathbf{V})$, $z'(\mathbf{V})$ provided the ultimate goal is to obtain $\delta_\Omega J[\Omega; \mathbf{V}]$ [1, 10]. Since such an approach hinges on suitably modifying the Lagrangian $\mathcal{L}$ of (2.2), which plays also a vital role in deriving the estimates for DWR of Section 3, we do not adopt it here but briefly discuss it now. If we append to $J[\Omega, u]$ the PDE and boundary conditions in (2.10) *via* Lagrange multipliers z, ξ, we end up with the modified Lagrangian

$$\mathcal{L}[\Omega, u, z, \xi] := J[\Omega, u] + \int_\Omega (\mathcal{B}u - f) z \, \mathrm{d}x + \int_{\partial\Omega} (u - \ell) \xi \, \mathrm{d}s,$$

with functions u, z, ξ defined in $\mathbb{R}^d$ but independent of Ω. If $u = u(\Omega)$ is the solution of (2.10), then $J[\Omega, u(\Omega)] = \mathcal{L}[\Omega, u(\Omega), z, \xi]$ for all z, ξ. Hence, using the chain rule yields

$$\delta_\Omega J[\Omega; \mathbf{V}] = \langle \delta_\Omega \mathcal{L}[\Omega, u, z, \xi] \,|_{u=u(\Omega)}, \mathbf{V} \rangle + \langle \delta_u \mathcal{L}[\Omega, u, z, \xi] \,|_{u=u(\Omega)}, u'(\mathbf{V}) \rangle$$

for all z, ξ. Since $z(\Omega), \xi(\Omega)$ are solutions of $\langle \delta_u \mathcal{L}[\Omega, u(\Omega), z(\Omega), \xi(\Omega)], v\rangle = 0$ for all v, we see that

$$\delta_\Omega J[\Omega; \mathbf{V}] = \langle \delta_\Omega \mathcal{L}[\Omega, u, z, \xi]\,|_{u=u(\Omega), z=z(\Omega), \xi=\xi(\Omega)}, \mathbf{V}\rangle.$$

Using the PDE satisfied by $u'(\mathbf{V})$ and $z'(\mathbf{V})$, and suitable regularity assumptions, (2.13) can be rewritten as a duality pairing on the deformable part Γ of $\partial\Omega$ [32], Section 2.11 and Theorem 2.27,

$$\langle \delta_\Omega \mathcal{L}[\Omega, u(\Omega), z(\Omega)], \mathbf{V}\rangle = \langle g, V\rangle_\Gamma. \tag{2.14}$$

We view g, which concentrates on Γ and pairs with the normal component V of $\mathbf{V}$, as a Riesz representative of the shape derivative of $\mathcal{L}$. In Sections 5 and 6, we examine two examples for the Stokes flow, carry out these calculations in detail, and give explicit expressions for g.

3. Dual weighted residual method

We now want to evaluate the PDE error using finite element methods (FEM). Therefore, we assume that the domain Ω is fixed and omit it as argument in both J and $\mathcal{L}$; thus $\mathcal{L}[\Omega, u, z] = \mathcal{L}[u, z]$. We recall that if we enforce the state equation (2.5), then (2.7) holds as well.

Given a conforming and shape-regular triangulation $\mathcal{T}$ of Ω, for any $T \in \mathcal{T}$ we denote by $h_T := |T|^{\frac{1}{d}}$ its *size*. Let $\mathbb{S}_\mathcal{T} \subset \mathbb{S}$ be a finite element subspace that satisfies the discrete inf-sup condition

$$\inf_{W\in\mathbb{S}_\mathcal{T}} \sup_{V\in\mathbb{S}_\mathcal{T}} \frac{B[V,W]}{\|V\|\|W\|} = \inf_{V\in\mathbb{S}_\mathcal{T}} \sup_{W\in\mathbb{S}_\mathcal{T}} \frac{B[V,W]}{\|V\|\|W\|} \geq \beta > 0,$$

with β independent of $\mathcal{T}$. This yields existence and uniqueness of the Galerkin solutions to the following finite element approximations of (2.5)–(2.6)

$$U \in \mathbb{S}_\mathcal{T}: \qquad B[U, W] = \langle f, W\rangle \qquad \forall W \in \mathbb{S}_\mathcal{T}, \tag{3.1}$$

$$Z \in \mathbb{S}_\mathcal{T}: \qquad B[W, Z] = \langle \delta_u J[U], W\rangle \qquad \forall W \in \mathbb{S}_\mathcal{T}, \tag{3.2}$$

which are stationary points of $\mathcal{L}$ in $\mathbb{S}_\mathcal{T}$. It remains to introduce the primal and dual residuals for $w \in \mathbb{S}$

$$R(U, Z; w) := \langle \delta_z \mathcal{L}[U, Z], w\rangle = \langle f, w\rangle - B[U, w], \tag{3.3}$$

$$R^*(U, Z; w) := \langle \delta_u \mathcal{L}[U, Z], w\rangle = \langle \delta_u J[U], w\rangle - B[w, Z]. \tag{3.4}$$

In view of (3.1)–(3.2), these residuals satisfy Galerkin orthogonality

$$R(U, Z; W) = R^*(U, Z; W) = 0 \qquad \forall\, W \in \mathbb{S}_\mathcal{T}. \tag{3.5}$$

The error $J[u] - J[U] = \mathcal{L}[u, z] - \mathcal{L}[U, Z]$ can be estimated in terms of the residuals R and R^*. This leads to the following error representation formula, whose proof can be found in [3,6,16]. We present it here for completeness.

Proposition 3.1 (error representation)**.** *The following a posteriori expression for $J[u] - J[U]$ is valid*

$$J[u] - J[U] = \frac{1}{2} R(U, Z; z - W_z) + \frac{1}{2} R^*(U, Z; u - W_u) + E \qquad \forall\, W_z, W_u \in \mathbb{S}_\mathcal{T} \tag{3.6}$$

where the remainder term $E = E(u, z, U, Z)$ is given by

$$E = \frac{1}{2}\int_0^1 \langle \delta_u^3 J[su + (1-s)U], e, e, e\rangle\ s(s-1)\,\mathrm{d}s \tag{3.7}$$

with $e = u - U$ the primal error. In addition, if J is a linear functional, then the two residuals R, R^* are equal, namely $R(U, Z; z) = R^*(U, Z; u)$, and

$$J[u] - J[U] = R(U, Z; z - W_z) \quad \forall\, W_z \in \mathbb{S}_\mathcal{T}. \tag{3.8}$$

Proof. By the fundamental theorem of Calculus

$$\mathcal{L}[u,z]-\mathcal{L}[U,Z]=\int_0^1\Big(\langle\delta_u\mathcal{L}[s(u,z)+(1-s)(U,Z)],u-U\rangle+\langle\delta_z\mathcal{L}[s(u,z)+(1-s)(U,Z)],z-Z\rangle\Big)\,\mathrm{d}s.$$

The trapezoidal rule, in conjunction with the fact that $\delta_u\mathcal{L}[u,z]=\delta_z\mathcal{L}[u,z]=0$, yields

$$\mathcal{L}[u,z]-\mathcal{L}[U,Z]=\frac{1}{2}\langle\delta_u\mathcal{L}[(U,Z],u-U\rangle+\frac{1}{2}\langle\delta_z\mathcal{L}[U,Z],z-Z\rangle+E,$$

where E satisfies (3.7) by direct computation. The equality (3.6) follows from (2.7), (3.3), (3.4), and (3.5). To prove (3.8), we observe that $\langle\delta_u J[u],w\rangle=J[w]$ if J is linear, whence (2.6) and (3.5) yield

$$J[u]-J[U]=\langle\delta_u J[u],u-U\rangle=B[u-U,z]=R(U,Z;z-W)=R^*(U,Z;u-W)\quad\forall\,W\in\mathbb{S}_{\mathcal{T}}.$$

This completes the proof. □

The *Dual Weighted Residual* method (DWR) consists of splitting R,R^* into element contributions

$$R(U,Z;w)=\sum_{T\in\mathcal{T}}\langle r(U,Z),w\rangle_T+\langle j(U,Z),w\rangle_{\partial T},\quad R^*(U,Z;w)=\sum_{T\in\mathcal{T}}\langle r^*(U,Z),w\rangle_T+\langle j^*(U,Z),w\rangle_{\partial T}$$

where $r(U,Z)=f-\mathcal{B}U$, $r^*(U,Z)=\delta_u J[U]-\mathcal{B}^*Z$ are the interior residuals, or strong form of the PDE, and $j(U,Z)$, $j^*(U,Z)$ are the jump residuals. They are both computable since they depend only on the computed discrete solutions U and Z. In most applications, the duality pairings $\langle\cdot,\cdot\rangle_T$, $\langle\cdot,\cdot\rangle_{\partial T}$ appearing in the last two expressions are just the $L^2(T)$, $L^2(\partial T)$ inner products, respectively. Consequently, the first two terms in (3.6) yield the (constant-free) bounds

$$\begin{aligned}|R(U,Z;z-W_z)|&\le\sum_{T\in\mathcal{T}}\|r(U,Z)\|_{L^2(T)}\|z-W_z\|_{L^2(T)}+\|j(U,Z)\|_{L^2(\partial T)}\|z-W_z\|_{L^2(\partial T)},\\|R^*(U,Z;u-W_u)|&\le\sum_{T\in\mathcal{T}}\|r^*(U,Z)\|_{L^2(T)}\|u-W_u\|_{L^2(T)}+\|j^*(U,Z)\|_{L^2(\partial T)}\|u-W_u\|_{L^2(\partial T)},\end{aligned}\tag{3.9}$$

and the quantities $\|z-W_z\|_{L^2(T)}$, $\|u-W_u\|_{L^2(T)}$ as well as those on ∂T are regarded as local weights. Estimating these weights requires knowing the state and adjoint variables u and z, and finding suitable quasi-interpolants W_u and W_z. We present now a novel local interpolation estimate for a given function v $(=u,z)$ expressed in terms of jumps of the discontinuous Lagrange interpolant $\Pi_{\mathcal{T}}v$ of v plus a higher order remainder. Similar estimates without justification are proposed in [6] for polynomial degree 1.

Lemma 3.2 (local interpolation estimate)**.** *Let $m\ge1$ be the polynomial degree, $d=2$ be the dimension, and $v\in H^{m+2}(\mathcal{N}(T))$ where $\mathcal{N}(T)$ is a discrete neighborhood of $T\in\mathcal{T}$. There exist constants $C_1,C_2>0$, solely dependent on mesh regularity, so that*

$$\|v-\Pi_{\mathcal{T}}v\|_{L^2(\mathcal{N}(T))}+h_T^{1/2}\|v-\Pi_{\mathcal{T}}v\|_{L^2(\partial T)}\le C_1\sum_{j=0}^{m}h_T^{j+1/2}\|[\![D^j\Pi_{\mathcal{T}}v]\!]\|_{L^2(\partial T)}+C_2h_T^{m+2}|v|_{H^{m+2}(\mathcal{N}(T))},\tag{3.10}$$

where $[\![\cdot]\!]$ denotes jump accross interelement sides.

Proof. We scale to the reference element $\widehat{T}$, where the desired estimate contains no powers of meshsize. We then proceed by contradiction: assume there is a sequence $\hat{v}_n\in H^{m+2}(\widehat{\mathcal{N}}(\widehat{T}))$ so that

$$\|\hat{v}_n-\widehat{\Pi}\hat{v}_n\|_{L^2(\widehat{\mathcal{N}}(\widehat{T}))}=1,\qquad\sum_{j=0}^{m}\|[\![D^j\widehat{\Pi}\hat{v}_n]\!]\|_{L^2(\partial\widehat{T})}+|\hat{v}_n|_{H^{m+2}(\widehat{\mathcal{N}}(\widehat{T}))}\to0$$

as $n \to \infty$. For a subsequence, still labeled $\hat{v}_n$, we have that $\hat{v}_n \to \hat{v} \in H^{m+2}(\widehat{\mathcal{N}}(\widehat{T}))$ weakly and thus strongly in $H^{m+1}(\widehat{\mathcal{N}}(\widehat{T}))$ and pointwise. The latter yields convergence $\widehat{\Pi}\hat{v}_n \to \widehat{\Pi}\hat{v}$ in $L^2(\widehat{\mathcal{N}}(\widehat{T}))$ as well as convergence of $[\![D^j\widehat{\Pi}\hat{v}_n]\!] \to [\![D^j\widehat{\Pi}\hat{v}]\!]$ in $L^2(\partial\widehat{T})$ for $0 \le j \le m$ because $\widehat{\Pi}\hat{v}_n$ is a piecewise polynomial of degree $\le m$. Hence, $[\![D^j\widehat{\Pi}\hat{v}]\!] = 0$ on $\partial\widehat{T}$ and $\widehat{\Pi}\hat{v}$ is a global polynomial of degree $\le m$ in $\widehat{\mathcal{N}}(\widehat{T})$.

On the other hand, the fact that $|\hat{v}|_{H^{m+2}(\widehat{\mathcal{N}}(\widehat{T}))} = 0$ implies that $\hat{v}$ is a polynomial of degree $\le m+1$ in $\widehat{\mathcal{N}}(\widehat{T})$. Therefore, $\hat{v} - \widehat{\Pi}\hat{v}$ vanishes at the $\frac{1}{2}(m+1)(m+2)$ canonical nodes of $\widehat{T}$. Moreover, $\hat{v} - \widehat{\Pi}\hat{v}$ vanishes at the additional $\frac{3}{2}m(m+1)$ canonical nodes outside $\widehat{T}$ but in $\widehat{\mathcal{N}}(\widehat{T})$. Since $\frac{3}{2}m(m+1) \ge m+2$ for all $m \ge 1$, we infer that $\hat{v} - \widehat{\Pi}\hat{v} = 0$ in $\widehat{\mathcal{N}}(\widehat{T})$, whence $\hat{v}$ is a global polynomial of degree $\le m$. This contradicts the property $\|\hat{v} - \widehat{\Pi}\hat{v}\|_{L^2(\widehat{\mathcal{N}}(\widehat{T}))} = 1$ and proves the asserted estimate for $\|v - \Pi_{\mathcal{T}}v\|_{L^2(\mathcal{N}(T))}$.

The same reasoning applies to $\|\hat{v} - \widehat{\Pi}\hat{v}\|_{L^2(\partial\widehat{T}))}$, and thus concludes the proof. □

Except in degenerate situations, the remainder in (3.10) is asymptotically of higher order, whence

$$\|v - \Pi_{\mathcal{T}}v\|_{L^2(\mathcal{N}(T))} + h_T^{1/2}\|v - \Pi_{\mathcal{T}}v\|_{L^2(\partial T)} \lesssim \sum_{j=0}^{m} h_T^{j+1/2}\|[\![D^j\Pi_{\mathcal{T}}v]\!]\|_{L^2(\partial T)}, \tag{3.11}$$

as $h_T \to 0$. This may be viewed as a discrete version of the celebrated Bramble-Hilbert estimate. Unfortunately, however, the remainder in (3.10) cannot in general be removed. The estimate (3.11) is not really computable because it requires knowing v. If $V \in \mathbb{S}_{\mathcal{T}}$ is a Galerkin approximation of $v \in \mathbb{S}$, then we expect its behavior to be similar to that of $\Pi_{\mathcal{T}}v$, which leads to the heuristic bound

$$\|v - V\|_{L^2(\mathcal{N}(T))} + h_T^{1/2}\|v - V\|_{L^2(\partial T)} \lesssim \sum_{j=0}^{m} h_T^{j+1/2}\|[\![D^jV]\!]\|_{L^2(\partial T)}. \tag{3.12}$$

Combining (3.6) and (3.9) with (3.12) we end up with the *a posteriori* upper bound

$$|J[u] - J[U]| \lesssim \sum_{T\in\mathcal{T}} \eta(T) \tag{3.13}$$

with element indicator

$$\begin{aligned}\eta(T) = &\Big(h_T^{1/2}\|r(U,Z)\|_{L^2(T)} + \|j(U,Z)\|_{L^2(\partial T)}\Big)\sum_{j=0}^{m} h_T^j\|[\![D^jZ]\!]\|_{L^2(\partial T)}\\ &+ \Big(h_T^{1/2}\|r^*(U,Z)\|_{L^2(T)} + \|j^*(U,Z)\|_{L^2(\partial T)}\Big)\sum_{j=0}^{m} h_T^j\|[\![D^jU]\!]\|_{L^2(\partial T)},\end{aligned} \tag{3.14}$$

which is the bound proposed in [6] for $m = 1$. One important drawback of this bound, discussed in [6], is the fact that there are unknown interpolation constants in it. This is less severe in the present context because we are mostly concerned with the correct distribution of spatial degrees of freedom rather than accurate bounds. Hence, for our purposes, the heuristic bound (3.13) is justified.

4. The adaptive sequential quadratic programming algorithm

In this section we describe the modules pertaining to the `ASQP` algorithm. Recall that $k \ge 1$ stands for the adaptive counter and Ω_k is the current domain produced by `ASQP` with deformable boundary Γ_k. Let $\mathbb{S}_k = \mathbb{S}_{\mathcal{T}_k}(\Omega_k)$ and $\mathbb{V}_k = \mathbb{V}_{\mathcal{T}_k}(\Gamma_k)$ be the finite element spaces on the bulk and boundary, which are compatible and fully determined by one underlying mesh $\mathcal{T}_k$ of Ω_k.

We define ASQP as follows:

```
ADAPTIVE SEQUENTIAL QUADRATIC PROGRAMMING ALGORITHM (ASQP)

Given the initial domain $\Omega_0$, a triangulation $\mathcal{T}_0$ of $\Omega_0$, and the parameter $0 < \theta \le \frac{1}{5}$,
set $\gamma = \frac{1}{2} - \theta(1+\theta)$, $k = 0$, $\varepsilon_0 = +\infty$, $\mu_0 = 1$ and repeat the following steps:
(1) $[\mathcal{T}_k, U_k, Z_k, J_k, G_k] =$ APPROXJ$(\Omega_k, \mathcal{T}_k, \varepsilon_k)$
(2) $[\mathbf{V}_k, \mathcal{T}_k] =$ DIRECTION$(\Omega_k, \mathcal{T}_k, G_k, \theta)$
(3) $[\Omega_{k+1}, \mathcal{T}_{k+1}, \mu_{k+1}] =$ LINESEARCH$(\Omega_k, \mathcal{T}_k, \mathbf{V}_k, J_k, \mu_k)$
(4) $\varepsilon_{k+1} := \gamma\mu_{k+1}\|\mathbf{V}_k\|^2_{\Gamma_k}$; $k \leftarrow k+1$.
```

In theory this algorithm is an infinite loop giving a more acurate approximation as the iterations progress, but in practice we implement a stopping criteria in LINESEARCH. In the next few subsections, we describe in detail each module of ASQP.

4.1. The module APPROXJ

This is a typical adaptive loop based on *a posteriori* error estimators, in which the domain Ω remains fixed. In this context we use the goal oriented estimators alluded to in Section 3 and refine and coarsen separately. The module APPROXJ is defined as follows:

```
$[\mathcal{T}_*, U_*, Z_*, J_*, G_*] =$ APPROXJ$(\Omega, \mathcal{T}, \varepsilon)$
do
    $[U, Z] =$ SOLVE$(\Omega, \mathcal{T})$
    $\{\eta(T)\}_{T\in\mathcal{T}} =$ ESTIMATE$(U, Z, \mathbb{S}_\mathcal{T})$
    $[\mathcal{R}, \mathcal{C}] =$ MARK$(\mathcal{T}, \{\eta(T)\}_{T\in\mathcal{T}})$
    if $(\eta(\mathcal{T}) > \varepsilon)$
        $[\mathcal{T}, \mathcal{C}] =$ REFINE$(\mathcal{T}, \mathcal{R})$
    elseif $(\eta(\mathcal{C}) < \delta\varepsilon)$
        $\mathcal{T} =$ COARSEN$(\mathcal{T}, \mathcal{C})$
    endif
while $(\eta(\mathcal{T}) > \varepsilon)$
$\mathcal{T}_* = \mathcal{T}$; $U_* = U$; $Z_* = Z$
$J_* =$ EVALJ$(\Omega, \mathcal{T}_*, U_*)$
$G_* =$ RIESZ$(\Omega, \mathcal{T}_*, U_*, Z_*)$
```

The module SOLVE computes the solution to the primal and dual discrete problems (3.1)–(3.2). The module ESTIMATE determines the local indicators $\eta(T)$, $T \in \mathcal{T}$ of the DWR method given by (3.14).

The module MARK selects some elements of $\mathcal{T}$ and assigns them to the set $\mathcal{R}$ of elements marked for refinement or to the set $\mathcal{C}$ of elements marked for coarsening. In both cases, MARK uses the *maximum strategy* which turns out to be more local and thus effective than others in this application. In fact, given parameters $0 < \delta_- \ll \delta_+ < 1$, we let $\eta_* = \max_{T\in\mathcal{T}} \eta(T)$ and apply the rules:

$$\eta(T) > \delta_+\eta_* \quad \Rightarrow \quad T \in \mathcal{R}; \qquad \eta(T) < \delta_-\eta_* \quad \Rightarrow \quad T \in \mathcal{C}.$$

The module REFINE subdivides the elements in the set $\mathcal{R}$ *via* bisection, and perhaps a few more elements to keep conformity of $\mathcal{T}$; REFINE also updates the set $\mathcal{C}$ which may have been affected by refinements. In contrast, the module COARSEN deals with the set of elements $\mathcal{C}$ selected for coarsening. Alternation of REFINE and COARSEN is crucial in this context, in which geometric singularities detected early on may disappear as the algorithm progresses towards the optimal shape. We illustrate this new paradigm with simulations in Sections 5 and 6.

Finally, the module EVALJ evaluates the functional $J[\Omega, U_*]$ on the updated mesh $\mathcal{T}_*$, whereas RIESZ computes a finite element approximation G_* to the shape derivative $g(\Omega)$.

4.2. The module DIRECTION

Given a tolerance $\theta \leq 1/5$, an approximate shape derivative G, and a domain Ω described through a triangulation $\mathcal{T}$, the call

$$[\mathbf{V}_*, \mathcal{T}_*] = \texttt{DIRECTION}(\Omega, \mathcal{T}, G, \theta)$$

finds an approximate descent direction $\mathbf{V}_*$ and an updated mesh $\mathcal{T}_*$ as follows: we let $\mathbb{V}(\Gamma)$ be a Hilbert space over Γ, $A_\Gamma : \mathbb{V}(\Gamma) \times \mathbb{V}(\Gamma) \to \mathbb{R}$ a continuous and coercive bilinear form, and define the *exact descent direction* v as

$$\mathrm{v} \in \mathbb{V}(\widetilde{\Gamma}) : \qquad A_{\widetilde{\Gamma}}(\mathrm{v}, \mathrm{w}) = -\langle G, \mathrm{w}\rangle_{\widetilde{\Gamma}} \quad \forall\, \mathrm{w} \in \mathbb{V}(\widetilde{\Gamma}),$$

i.e., v is the weak solution of $\mathcal{A}\mathrm{v} = -G$ on a smooth surface $\widetilde{\Gamma}$ being approximated by Γ. Let $\mathbb{V}_\mathcal{T}$ be the finite element space over the restriction of the mesh $\mathcal{T}$ to the boundary Γ of Ω, and let V satisfy

$$V \in \mathbb{V}_\mathcal{T} : \qquad A_\Gamma(V, W) = -\langle G, W\rangle_\Gamma \quad \forall\, W \in \mathbb{V}_\mathcal{T}.$$

The module DIRECTION then performs (stationary) adaptivity through an alternation of refinement and coarsening so that on the output mesh $\mathcal{T}_*$, the finite element solution $V_* \in \mathbb{V}_* := \mathbb{V}_{\mathcal{T}_*}$ satisfies

$$\|V_* - \mathrm{v}\|_{\mathbb{V}(\Gamma)} \leq \theta \|V_*\|_{\mathbb{V}(\Gamma)} \tag{4.1}$$

and is also a descent direction because (4.1) controls the angle between v and V_*; see (1.6) in Section 1. The choice of A_Γ is critical to obtain a sequence of relatively smooth domains and avoid instabilities [15]. We have successfully implemented the weighted Laplace-Beltrami bilinear form A_Γ defined by

$$A_\Gamma(\mathrm{v}, \mathrm{w}) = \int_\Gamma \rho \nabla_\Gamma \mathrm{v} \nabla_\Gamma \mathrm{w} + \mathrm{v}\mathrm{w}\, \mathrm{d}S \qquad \forall \mathrm{v}, \mathrm{w} \in \mathbb{V}(\Gamma) = H^1(\Gamma);$$

see [9,11,22] for alternative choices of the bilinear form A_Γ. The weight ρ depends on the optimization problem under study, as well as on its relative scales (see Sects. 5 and 6). The error control (4.1) is achieved by resorting to residual *a posteriori* error estimates for the $H^1(\Gamma)$-norm [13, 14, 21]. More precisely, if $V \in \mathbb{V}_\mathcal{T}$ denotes the Galerkin approximation to v on a mesh $\mathcal{T}$ we define the Laplace-Beltrami (LB) error indicator by [21]

$$\eta_\Gamma^2(T; V) := h_T^2 \| -\rho \Delta_\Gamma V + V - G\|_{L^2(T)}^2 + h_T \|[\![\rho \nabla_\Gamma V]\!]\|_{L^2(\partial T)}^2 + \|\boldsymbol{\nu} - \boldsymbol{\nu}_\mathcal{T}\|_{L^\infty(T)}^2 \|\sqrt{\rho}\, \nabla_\Gamma V + V\|_{L^2(T)}^2,$$

for T a *surface* element of $\mathcal{T}$ contained in Γ. These indicators satisfy $\|\mathrm{v} - V\|_{\mathbb{V}(\Gamma)}^2 \leq C \sum_{T \subset \Gamma} \eta_\Gamma^2(T; V)$. The first two terms are the usual indicators for a reaction-diffusion equation, whereas the last one is a *geometric indicator*, that takes into account the error in approximating the domain through $\boldsymbol{\nu} - \boldsymbol{\nu}_\mathcal{T}$. Here $\boldsymbol{\nu}$ denotes the exact normal to the smooth surface $\widetilde{\Gamma}$ being approximated by Γ and $\boldsymbol{\nu}_\mathcal{T}$ denotes the normal of the discrete surface. Since we do not have access to the exact smooth surface $\widetilde{\Gamma}$ we estimate $\|\boldsymbol{\nu} - \boldsymbol{\nu}_\mathcal{T}\|_{L^\infty(T)}$ by the jump term $\|[\![\boldsymbol{\nu}_\mathcal{T}]\!]\|_{L^\infty(\partial T)}$, and the computable estimator reads

$$\eta_\Gamma^2(T; V) := h_T^2 \| -\rho \Delta_\Gamma V + V - G\|_{L^2(T)}^2 + h_T \|[\![\rho \nabla_\Gamma V]\!]\|_{L^2(\partial T)}^2 + \|[\![\boldsymbol{\nu}_\mathcal{T}]\!]\|_{L^\infty(\partial T)}^2 \|\sqrt{\rho}\, \nabla_\Gamma V + V\|_{L^2(T)}^2.$$

For two-dimensional domains with polygonal boundaries, a simpler upper bound is obtained as follows. Since both the Clément or Scott-Zhang interpolation operators are H^1-stable, they are used in deriving *a posteriori* error estimates to approximate test functions in H^1 and thus get the appropriate powers of h_T in $\eta_\Gamma^2(T; V)$; see [14, 21]. When the underlying (boundary) mesh is one-dimensional one can resort, instead, to Lagrange interpolation because H^1 is embedded in the space of continuous functions. It turns out that the jump $[\![\rho \nabla_\Gamma V]\!]$ is multiplied by the test function minus its interpolant and evaluated at vertices. Since the Lagrange interpolant coincides with the function at vertices, for one dimensional boundary meshes the term $[\![\rho \nabla_\Gamma V]\!]$ drops out and we obtain the following simpler estimator:

$$\eta_\Gamma^2(T; V) := h_T^2 \| -\rho \Delta_\Gamma V + V - G\|_{L^2(T)}^2 + \|[\![\boldsymbol{\nu}_\mathcal{T}]\!]\|_{L^\infty(\partial T)}^2 \|\sqrt{\rho}\, \nabla_\Gamma V + V\|_{L^2(T)}^2.$$

With this definition of *a posteriori* error indicators we execute a loop of the form

$$\texttt{SOLVE} \quad \rightarrow \quad \texttt{ESTIMATE} \quad \rightarrow \quad \texttt{MARK} \quad \rightarrow \quad \texttt{REFINE/COARSEN}$$

with MARK based on the maximum strategy, as described in Section 4.1, until the last discrete solution V_* satisfies $C\sum_{T\subset\Gamma}\eta_\Gamma^2(T;V_*) \le \theta^2\|V_*\|_{\mathbb{V}(\Gamma)}$, thereby giving (4.1).

To advance the domain we need a vector velocity $\mathbf{V}_*$ such that V_* is its normal component. Since Γ is piecewise polynomial, the normal $\boldsymbol{\nu}$ is discontinuous. We thus define an average vector velocity $\mathbf{V}_*$, and output of DIRECTION, as follows

$$\mathbf{V}_* \in \mathbb{V}_*^d: \qquad \langle \mathbf{V}_*, \boldsymbol{\varphi}\rangle_\Gamma = \langle V_*, \boldsymbol{\nu}\cdot\boldsymbol{\varphi}\rangle_\Gamma \quad \forall \boldsymbol{\varphi}\in\mathbb{V}_*^d, \tag{4.2}$$

as in [5,15].

4.3. The module LINESEARCH

Given a domain Ω described by a triangulation $\mathcal{T}$, a vector velocity $\mathbf{V}$, the functional value $J[\Omega]$, and the previous timestep μ, the LINESEARCHmodule computes a new timestep μ_* and updates both the domain Ω to Ω_* and the mesh $\mathcal{T}$ to $\mathcal{T}_*$ as follows:

```
$[\Omega_*,\mathcal{T}_*,\mu_*]$ = LINESEARCH($\Omega,\mathcal{T},\mathbf{V},J,\mu$)
    $m$ = GEOSTEP($\mathbf{V}$) %find max possible geometric step
    $\mu = \min(\mu, m)$
    $J_{\mathrm{old}} = J$, $J$ = TRYSTEP($\mu,\Omega,\mathcal{T},\mathbf{V}$)
    if ($J_{\mathrm{old}} < J$) %energy is not decreasing reduce time step
      [success, $\mu$] = DECREASESTEP($J,\mu,\Omega,\mathcal{T},\mathbf{V}$)
      if (success == false) %we reached the stopping criteria
        break   end if
    else %energy is decreasing, can we get better?
      [success, $\mu$] = TRYDECREASE($J,\mu,\Omega,\mathcal{T},\mathbf{V}$)
      if ( success == false )
        [success, $\mu$] = TRYINCREASE($J,\mu,\Omega,\mathcal{T},\mathbf{V}$)
      end if
    end if
    $[\Omega_*,\mathcal{T}_*]$ = UPDATE($\Omega,\mathcal{T},\mathbf{V},\mu$), $\mu_* = \mu$
```

The use of the module GEOSTEP is a mechanism to avoid mesh distortion due to tangential motion of nodes, *i.e.* we need to control the effect of the tangential component $(\mathbf{I}-\boldsymbol{\nu}\otimes\boldsymbol{\nu})\mathbf{V}$ of $\mathbf{V}$. Such a control boils down to a geometric restriction of steplength: the output m of GEOSTEP is the largest admissible steplength that avoids node crossing and is computed as follows. If ρ_T is the diameter of the largest inscribed ball in $T\in\mathcal{T}$, and $\mathbf{z}$ is a generic boundary node, then we let $d(\mathbf{z})$ be the nodal function that takes the minimum of ρ_T over all $T\in\mathcal{T}$ that share $\mathbf{z}$. The quantity $\vartheta\frac{\mathrm{d}(\mathbf{z})}{|(\mathbf{I}-\boldsymbol{\nu}\otimes\boldsymbol{\nu})\mathbf{V}(\mathbf{z})|}$ gives the largest steplength allowed for node $\mathbf{z}$ to move without entangling the mesh, provided $\vartheta\le 1/2$, and represents a worst case scenario; m is thus the smallest of those values for all boundary nodes $\mathbf{z}$. Practice suggests that $\vartheta = 1/3$ is a good choice for linear meshes whereas $\vartheta = 1/6$ is a safe choice for quadratic meshes controlled by the hybrid method of [23].

The module TRYSTEP finds the energy of a deformation of Ω by $\mu\mathbf{V}$ corresponding to a given timestep μ as follows:

```
$J_*$ = TRYSTEP($\mu,\Omega,\mathcal{T},\mathbf{V}$)
    $[\Omega_*,\mathcal{T}_*]$ = UPDATE($\mu,\Omega,\mathcal{T},\mathbf{V}$)
    $[U_*,Z_*]$ = SOLVE($\Omega_*,\mathcal{T}_*$)
    $J_*$ = EVALJ($\Omega_*,\mathcal{T}_*,U_*$)
```

Here the module UPDATE advances the domain to the new configuration Ω_* and updates the mesh $\mathcal{T}$ to $\mathcal{T}_*$. This is done as follows:

```
[$\Omega_*, \mathcal{T}_*$] = UPDATE($\Omega, \mathcal{T}, \mathbf{V}, \mu$)

    $\Omega_* = \Omega$
    $\mathbf{x} = \mathbf{x} + \mu\mathbf{V}(\mathbf{x}) \quad \forall \mathbf{x} \in \partial\Omega_*$
    MESHOPTIMIZE($\Omega_*$)
```

We first move the boundary using $\mathbf{V}$ and then we move the interior nodes using the mesh smoothing routine MESHOPTIMIZE that optimizes the location of the star center nodes trying to improve their quality; see [23] for details.

The module SOLVE finds primal and dual solutions of (3.1)–(3.2) on the new finite element space $\mathbb{S}_*$. Finally, EVALJ evaluates the new functional $J_* = J[\Omega_*, U_*]$.

The modules TRYDECREASE and TRYINCREASE decrement or increment the timestep as long as the energy keeps decreasing, and use the parameters $0 < a < 1 < b$ provided by the user. The module DECREASESTEP has a built-in stopping mechanism: when the energy cannot be reduced anymore while keeping the timestep above a threshold timestep μ_0 the algorithm stops.

```
[success, $\mu^*$] = TRYDECREASE($J, \mu, \Omega, \mathcal{T}, \mathbf{V}$)

    %Reduce $\mu$ while energy keeps decreasing
    $\mu^* = \mu$, success = false
    do
        $J_{\text{old}} = J$, $\mu^* = a * \mu^*$
        $J$ = TRYSTEP($\mu^*, \Omega, \mathcal{T}, \mathbf{V}$)
    while ($J < J_{\text{old}}$)
    $\mu^* = \mu^*/a$
    if ($\mu^* < \mu$) success = true;
```

```
[success, $\mu^*$] = TRYINCREASE($J, \mu, \Omega, \mathcal{T}, \mathbf{V}$)

    %increment $\mu$ while energy keeps decreasing
    $\mu^* = \mu$, success = false
    do
        $J_{\text{old}} = J$, $\mu^* = b * \mu^*$
        $J$ = TRYSTEP($\mu^*, \Omega, \mathcal{T}, \mathbf{V}$)
    while ($J < J_{\text{old}}$)
    $\mu^* = \mu^*/b$
    if ($\mu^* > \mu$) success = true
```

```
[success, $\mu^*$] = DECREASESTEP($J, \mu, \Omega, \mathcal{T}, \mathbf{V}$)

    %decrease $\mu$ until we reduce energy or stop criteria
    $\mu^* = \mu$, success = false
    do
        $J_{\text{old}} = J$, $\mu^* = a * \mu^*$
        $J$ = TRYSTEP($\mu^*, \Omega, \mathcal{T}, \mathbf{V}$)
    while ($J > J_{\text{old}}$) and ($\mu^* > \mu_0$)
    if ($\mu^* > \mu_0$) success = true;
```

Remark 4.1 (Wolfe-Armijo conditions)**.** Even though it looks very simple minded, our linesearch and backtracking algorithm is very robust. We have also tried to use the celebrated Armijo-Wolfe conditions, but their behavior was not so robust because it depends on having a reliable computation of the functional derivative in addition to the functional itself.

4.4. Geometrically consistent mesh modification

After the final UPDATE in LINESEARCH some post-processing takes place to ensure a healthy and geometrically sound mesh for the next iteration. It involves a *geometrically consistent* relocation of the newly created nodes (if any) and a mesh quality check with the possibility of remeshing if certain threshold is not satisfied. Below and in the next subsection we explain each process.

The presence of corners (or kinks) on the deformable boundary Γ_k is usually problematic. First, the scalar product $A_{\Gamma_k}(\cdot,\cdot)$ of (1.5) includes a Laplace-Beltrami regularization term ($\rho > 0$) which stabilizes the boundary update but cannot remove kinks because V_k is smooth. Secondly, DWR regards kinks as true singularities and tries to refine around them accordingly. The combination of these two effects leads to numerical artifacts (ear formation) and halt of computations. The geometrically consistent mesh modification (GCMM) method of [8] circumvents this issue. Assuming that a piecewise polynomial approximation $\mathbf{H}_k$ to the vector curvature of Γ_k is available, [8] provides a method to place the nodes after a mesh modification such as refinement, coarsening or smoothing takes place. The method requires transfering $\mathbf{H}_k$ to an intermediate modified mesh and yields the new position $\mathbf{X}_k$ of the free boundary from the fundamental geometric identity $-\Delta_{\Gamma_k}\mathbf{X}_k = \mathbf{H}_k$. This preserves geometric consistency – which is violated by simply interpolating Γ_k – as well as accuracy [8], and rounds kinks. For "fake" kinks (*e.g.* initial corners) the effect of GCMM permits the optimization flow to get rid of the kinks, a highly desirable outcome. For "genuine" kinks, *i.e.* those that exist in the optimal shape, the optimization flow may in principle conflict with the smoothing effects of GCMM. Since GCMM is only applied when adaptivity takes place, the optimization flow dominates overall. Therefore, genuine corners of the optimal shape being stable are increasingly better resolved despite the fact that descent directions are smooth – and so are all the intermediate shapes. The numerical curvature of genuine corners thus depends on the local meshsize and the regularization parameter ρ.

Since in the present context a vector curvature approximation $\mathbf{H}_k$ is not directly available, as required in [8], we resort to the techniques described in [5, 24] and we use a star average to construct a continuous normal $\boldsymbol{\nu}_k$ from the discontinuous element normals. In fact, for each boundary node $\mathbf{x}$ we define $\boldsymbol{\nu}_k(\mathbf{x})$ as the normalization of $\frac{1}{|\omega|}\sum_{T\subset\omega}|T|\mathbf{n}_T$, where ω is the set of elements T belonging to the boundary triangulation and sharing $\mathbf{x}$ as a vertex and $\mathbf{n}_T$ is the element normal. Having the nodal values, this defines a unique piecewise linear and globally continuous function. We next consider the scalar approximation to mean curvature $H_k = \mathrm{div}_{\Gamma_k}\boldsymbol{\nu}_k$, let $\mathbf{H}_k = H_k\boldsymbol{\nu}_k$, and proceed as in [8].

4.5. Remeshing

A rule of thumb for dealing with complicated domain deformations is that remeshing is indispensable and unavoidable. Our approach is to use remeshing only when necessary for the continuation of the simulation. At the end of each iteration we check the mesh quality, and if it falls below a given threshold remeshing takes place. In the drag simulation of Section 5.6, for example, 4 remeshings were necesary in 180 iterations, whereas the less complex deformations for the bypass simulations of Section 6 did not require remeshing.

A disadvantage of remeshing in the context of adaptivity is that most mesh generators create unstructured meshes. Since the hierarchical mesh structure is then lost, coarsening cannot be performed beyond the structure of the new (macro) mesh. This problem could be easily solved by using a hierarchical mesh generator, but its discussion is beyond the scope of this work.

4.6. Volume constraint

If the definition of $\mathcal{U}_{\mathrm{ad}}$ involves a fixed-volume constraint V_0, like in Section 5, such a constraint is enforced as follows in the module UPDATE of LINESEARCH. Given a descent direction $\widetilde{\mathbf{V}}$ (from the module DIRECTION) for the unconstrained energy $J[\Omega]$ with $|\Omega| = V_0$, then $[\Omega_*, \mathcal{T}_*] =$UPDATE$(\Omega, \mathcal{T}, \widetilde{\mathbf{V}}, \mu)$ returns a new domain Ω_* with the same prescribed volume $|\Omega_*| = V_0$ and intuitively a smaller associated energy $J[\Omega_*]$; the latter is in turn ultimately checked in LINESEARCH. The module UPDATE proceeds by moving the boundary of Ω by $\mu\widetilde{\mathbf{V}}$ and

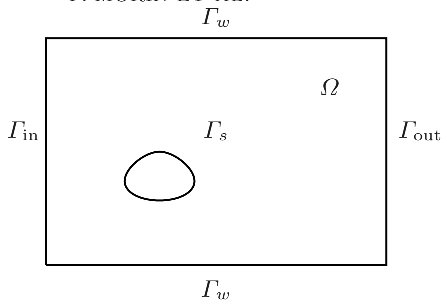

FIGURE 1. Domain Ω for drag minimization for Stokes flow: $\Omega \subset \mathbb{R}^d$, $d \geq 2$, is a bounded domain with its boundary subdivided into an *inflow* part Γ_{in}, an *outflow* part Γ_{out}, a part considered as *walls* Γ_w, and an obstacle Γ_s which is the deformable part to be optimized.

then projects it onto the manifold $\mathcal{U}_{\text{ad}}$ of shapes with the fixed volume V_0. This projection is not arbitrary but based on the augmented energy functional

$$\mathcal{J}[\Omega] = J[\Omega] - \lambda\big(|\Omega| - V_0\big),$$

where λ is a Lagrange multiplier to ensure that $|\Omega| = V_0$. The shape derivative of $\mathcal{J}$ in the direction $\mathbf{V}$ is given by

$$\langle \delta_\Omega \mathcal{J}[\Omega], \mathbf{V}\rangle = \langle \delta_\Omega J[\Omega], \mathbf{V}\rangle - \lambda\langle \boldsymbol{\nu}, \mathbf{V}\rangle,$$

and a gradient flow would choose to move along the descent direction $\widetilde{\mathbf{V}} = \mathcal{A}_\Gamma^{-1}(-\delta_\Omega J[\Omega])$ to decrease the unconstrained energy $J[\Omega]$. Once the boundary of Ω has been deformed using $\mu\widetilde{\mathbf{V}}$ and a new configuration Ω_* reached, it seems natural to evolve the domain with the normal flow

$$\frac{\mathrm{d}}{\mathrm{d}\lambda}\mathbf{X} = \boldsymbol{\nu} \quad \text{in } \Omega(t); \qquad \mathbf{X}(0) = \mathrm{Id} \quad \text{in } \Omega(0) = \Omega_*,$$

until we find a zero of the scalar function $f(\lambda) = |\Omega(\lambda)| - V_0$. Such a zero can be found *via* a Newton method with step $\delta\lambda = -\frac{f(\lambda)}{f'(\lambda)}$. Since $f'(\lambda) = |\partial\Omega(\lambda)|$, we now have the two ingredients of the following algorithm, namely a Newton correction of λ (starting from $\lambda = 0$) and a normal update of $\Omega(\lambda)$:

```
$\Omega = \Omega_*$
while $\left(\frac{|\Omega| - V_0}{V_0} > \epsilon\right)$
  $\delta\lambda = -\frac{|\Omega| - V_0}{|\partial\Omega|}$    %compute newton step
  $\boldsymbol{\nu}$ = AVERAGENORMAL($\partial\Omega$);  $\Omega = \Omega + \delta\lambda\boldsymbol{\nu}$    %update the domain
end while
```

Here ϵ is a given tolerance for the Newton method whereas the function `AVERAGENORMAL`$(\partial\Omega)$ computes a continuous normal $\boldsymbol{\nu}$ over the piecewise polynomial boundary $\partial\Omega$, with nodal unit length. To this end, it uses the same averaging procedure (4.2) on stars described in Section 4.4 (see also [5]). Notice that in view of the use of `AVERAGENORMAL` which changes the normal at each iteration, it seems to be more appropriate to refer to the above algorithm as to a quasi-Newton scheme.

5. DRAG MINIMIZATION FOR STOKES FLOW

5.1. The stokes problem

We consider the flow around an obstacle described by the following Stokes equations. Let $\Omega \subset \mathbb{R}^d$, $d \geq 2$ be a bounded domain as depicted in Figure 1.

Let the velocity $\mathbf{u} := \mathbf{u}(\Omega)$ and the pressure $p := p(\Omega)$ solve the following problem:

$$\begin{aligned} -\operatorname{div}(\mathbf{T}(\mathbf{u},p)) &= 0 && \text{in } \Omega \\ \operatorname{div}\mathbf{u} &= 0 && \text{in } \Omega \\ \mathbf{u} &= \mathbf{u}_d && \text{on } \Gamma_{\text{in}} \cup \Gamma_s \cup \Gamma_w \\ \mathbf{T}(\mathbf{u},p)\boldsymbol{\nu} &= 0 && \text{on } \Gamma_{\text{out}} \end{aligned} \tag{5.1}$$

where $\mathbf{T}(\mathbf{u},p) := 2\mu\epsilon(\mathbf{u}) - p\mathbf{I}$ is the Cauchy tensor with $\epsilon(\mathbf{u}) = \frac{\nabla\mathbf{u}+\nabla\mathbf{u}^T}{2}$, $\mu > 0$ is the viscosity, and

$$\mathbf{u_d} = \begin{cases} \mathbf{v}_\infty & \text{on } \Gamma_{\text{in}} \\ \mathbf{0} & \text{on } \Gamma_w \cup \Gamma_s, \end{cases}$$

with $\mathbf{v}_\infty = V_\infty \hat{\mathbf{v}}_\infty$, $\hat{\mathbf{v}}_\infty$ being the unit vector pointing in the direction of the incoming flow and V_∞ a scalar function.

In order to state a weak formulation of this problem, we introduce the following bilinear forms:

$$\begin{aligned} a[\cdot,\cdot] : [H^1(\Omega)]^d \times [H^1(\Omega)]^d \to \mathbb{R}, \qquad & a[\mathbf{u},\mathbf{v}] := 2\mu \int_\Omega \epsilon(\mathbf{u}) : \epsilon(\mathbf{v})\, \mathrm{d}x, \\ b[\cdot,\cdot] : L^2(\Omega) \times [H^1(\Omega)]^d \to \mathbb{R}, \qquad & b[p,\mathbf{v}] := -\int_\Omega p \operatorname{div}\mathbf{v}\, \mathrm{d}x. \end{aligned} \tag{5.2}$$

We let $\Gamma_d := \Gamma_{\text{in}} \cup \Gamma_s \cup \Gamma_w$ be the Dirichlet boundary, introduce the affine manifolds

$$\begin{aligned} [H^1_{\Gamma_d}(\Omega)]^d &= \{\mathbf{u} \in [H^1(\Omega)]^d : \mathbf{u} = \mathbf{0} \text{ on } \Gamma_d\}, \\ \mathbf{u_d} \oplus [H^1_{\Gamma_d}(\Omega)]^d &= \{\mathbf{u} \in [H^1(\Omega)]^d : \mathbf{u} = \mathbf{u_d} \text{ on } \Gamma_d\}, \end{aligned}$$

and set $\mathbb{S}(\mathbf{v}) := (\mathbf{v} \oplus [H^1_{\Gamma_d}(\Omega)]^d) \times L^2(\Omega)$. The weak formulation of the Stokes problem (5.1) reads

$$(\mathbf{u},p) \in \mathbb{S}(\mathbf{u_d}) : \quad B[(\mathbf{u},p),(\mathbf{v},q)] = 0 \quad \forall (\mathbf{v},q) \in \mathbb{S}(\mathbf{0}), \tag{5.3}$$

where

$$B[(\mathbf{u},p),(\mathbf{v},q)] := a[\mathbf{u},\mathbf{v}] + b[p,\mathbf{v}] + b[q,\mathbf{u}]. \tag{5.4}$$

5.2. Cost functional and Lagrangian

We let the cost functional measuring the obstacle *drag* be

$$I[\Omega,(\mathbf{u},p)] := -\int_{\Gamma_s} (\mathbf{T}(\mathbf{u},p)\boldsymbol{\nu}) \cdot \hat{\mathbf{v}}_\infty \, \mathrm{d}S, \tag{5.5}$$

where $(\mathbf{u},p)$ solves (5.3). We would like to minimize the linear boundary functional I subject to the state constraint (5.3) among all admissible configurations with *fixed volume* that can be obtained by piecewise smooth perturbations of the obstacle boundary Γ_s [25, 26]. We thus introduce the functional with Lagrange multiplier $\lambda \in \mathbb{R}$ and $|\Omega| = \int_\Omega \mathrm{d}x$

$$J[\Omega,(\mathbf{u},p),\lambda] := I[\Omega,(\mathbf{u},p)] + \lambda\big(|\Omega| - |\Omega_0|\big). \tag{5.6}$$

It is useful to rewrite I as a volume integral. This helps derive, as well as compute, the adjoint equation and shape derivative of I. We introduce a function $\boldsymbol{\Phi}_\infty \in [H^1(\Omega)]^d$ such that

$$\boldsymbol{\Phi}_\infty = \begin{cases} -\hat{\mathbf{v}}_\infty & \text{in } \mathcal{N}(\Gamma_s) \\ \mathbf{0} & \text{on } \Gamma_w \cup \Gamma_{\text{in}}, \end{cases}$$

where $\mathcal{N}(\mathit{\Gamma}_s)$ is a neighborhood of $\mathit{\Gamma}_s$. The traction-free boundary condition $\mathbf{T}(\mathbf{u},p)\boldsymbol{\nu} = 0$ on $\mathit{\Gamma}_{\text{out}}$ and Gauss theorem yield

$$\begin{aligned} I[\mathit{\Omega},(\mathbf{u},p)] &= \int_{\partial\mathit{\Omega}} (\mathbf{T}(\mathbf{u},p)\boldsymbol{\nu})\cdot\mathbf{\Phi}_\infty \,\mathrm{d}S = \int_{\mathit{\Omega}} \operatorname{div}(\mathbf{T}(\mathbf{u},p)\mathbf{\Phi}_\infty)\,\mathrm{d}x \\ &= \int_{\mathit{\Omega}} \operatorname{div}(\mathbf{T}(\mathbf{u},p))\cdot\mathbf{\Phi}_\infty + \mathbf{T}(\mathbf{u},p):\nabla\mathbf{\Phi}_\infty\,\mathrm{d}x = a[\mathbf{u},\mathbf{\Phi}_\infty] + b[p,\mathbf{\Phi}_\infty]. \end{aligned} \tag{5.7}$$

According to (2.2), the Lagrangian is defined as follows for all $(\mathbf{v},r)\in\mathbb{S}(\mathbf{0})$:

$$\begin{aligned} \mathcal{L}[\mathit{\Omega},(\mathbf{u},p),(\mathbf{v},r),\lambda] &:= J[\mathit{\Omega},(\mathbf{u},p),\lambda] - B[(\mathbf{u},p),(\mathbf{v},r)] \\ &= a[\mathbf{u},\mathbf{\Phi}_\infty - \mathbf{v}] + b[p,\mathbf{\Phi}_\infty - \mathbf{v}] - b[r,\mathbf{u}] + \lambda\big(|\mathit{\Omega}| - |\mathit{\Omega}_0|\big) \\ &= B[(\mathbf{u},p),(\mathbf{\Phi}_\infty - \mathbf{v}, -r)] + \lambda\big(|\mathit{\Omega}| - |\mathit{\Omega}_0|\big) \\ &= B[(\mathbf{u},p),(\mathbf{z},q)] + \lambda\big(|\mathit{\Omega}| - |\mathit{\Omega}_0|\big), \end{aligned} \tag{5.8}$$

where $\mathbf{z} = \mathbf{\Phi}_\infty - \mathbf{v}$ and $q = -r$ are the adjoint variables.

5.3. Adjoint equation

We now derive the adjoint equation from the Lagrangian (5.8).

Lemma 5.1 (adjoint equation for (5.8))**.** *The adjoint pair* $(\mathbf{z},q)$ *satisfies the weak equation*

$$(\mathbf{z},q)\in\mathbb{S}(\mathbf{\Phi}_\infty): \quad B[(\mathbf{w},s),(\mathbf{z},q)] = 0 \quad \forall(\mathbf{w},s)\in\mathbb{S}(\mathbf{0}), \tag{5.9}$$

as well as the strong form

$$\begin{aligned} -\operatorname{div}(\mathbf{T}(\mathbf{z},q)) &= 0 && \textit{in } \mathit{\Omega} \\ \operatorname{div}\mathbf{z} &= 0 && \textit{in } \mathit{\Omega} \\ \mathbf{z} &= \mathbf{\Phi}_\infty && \textit{on } \mathit{\Gamma}_{\text{in}}\cup\mathit{\Gamma}_s\cup\mathit{\Gamma}_w \\ \mathbf{T}(\mathbf{z},q)\boldsymbol{\nu} &= 0 && \textit{on } \mathit{\Gamma}_{\text{out}}. \end{aligned} \tag{5.10}$$

Proof. Differentiate (5.8) with respect to $(\mathbf{u},p)$ to arrive at

$$\langle\delta_{(\mathbf{u},p)}\mathcal{L}[\mathit{\Omega},(\mathbf{u},p),(\mathbf{z},q),\lambda],(\mathbf{w},s)\rangle = B[(\mathbf{w},s),(\mathbf{z},q)] = 0 \quad \forall(\mathbf{w},s)\in\mathbb{S}(\mathbf{0}),$$

which is (5.9). The strong form (5.10) results from (5.9) by integration by parts. □

5.4. Shape derivative

We now compute the shape derivative $\delta_{\mathit{\Omega}}\mathcal{L}[\mathit{\Omega},(\mathbf{u},p),(\mathbf{z},q),\lambda]$, recalling that $\mathbf{u}$, p, $\mathbf{z}$, q depend on $\mathit{\Omega}$, using the rules described in Section 2.2. See also [7,22] for similar results.

Lemma 5.2 (shape derivative of (5.8))**.** *Let* $(\mathbf{u},p)$ *be the solution to* (5.3) *and* $(\mathbf{z},q)$ *be the solution to* (5.9). *The shape derivative of* $\mathcal{L}[\mathit{\Omega},(\mathbf{u},p)(\mathit{\Omega}),(\mathbf{v},q)(\mathit{\Omega}),\lambda(\mathit{\Omega})]$ *in the direction* $\mathbf{V}$ *is given by*

$$\big\langle\delta_{\mathit{\Omega}}\mathcal{L}[\mathit{\Omega},(\mathbf{u},p)(\mathit{\Omega}),(\mathbf{z},q)(\mathit{\Omega}),\lambda(\mathit{\Omega})],\mathbf{V}\big\rangle = -2\mu\int_{\mathit{\Gamma}_s}\epsilon(\mathbf{u}):\epsilon(\mathbf{z})\;V\;\mathrm{d}S + \lambda\int_{\mathit{\Gamma}_s} V\;\mathrm{d}S. \tag{5.11}$$

Proof. In view of (5.8), we have

$$\begin{aligned} \mathcal{L}[\mathit{\Omega},(\mathbf{u},p)(\mathit{\Omega}),(\mathbf{z},q)(\mathit{\Omega}),\lambda(\mathit{\Omega})] &= a[\mathbf{u},\mathbf{z}] + b[p,\mathbf{z}] + b[q,\mathbf{u}] + \lambda\left(|\mathit{\Omega}| - |\mathit{\Omega}_0|\right) \\ &= \int_{\mathit{\Omega}}\left(2\mu\epsilon(\mathbf{u}):\epsilon(\mathbf{z}) - p\;\operatorname{div}\mathbf{z} - q\;\operatorname{div}\mathbf{u}\right)\mathrm{d}x + \lambda\left(\int_{\mathit{\Omega}}\mathrm{d}x - \int_{\mathit{\Omega}_0}\mathrm{d}x\right). \end{aligned}$$

Invoking (2.13), and recalling that $\operatorname{div}\mathbf{u} = \operatorname{div}\mathbf{z} = 0$, we deduce

$$\begin{aligned}\big\langle \delta_\Omega \mathcal{L}[\Omega, (\mathbf{u},p)(\Omega), (\mathbf{v},q)(\Omega), \lambda(\Omega)], \mathbf{V}\big\rangle = &\int_{\Gamma_s} \Big(2\mu\epsilon(\mathbf{u}) : \epsilon(\mathbf{z}) + \lambda\Big) V \,\mathrm{d}S \\ &+ \int_\Omega \Big(2\mu\epsilon(\mathbf{u}') : \epsilon(\mathbf{z}) - p' \operatorname{div}\mathbf{z} - q \operatorname{div}\mathbf{u}'\Big)\,\mathrm{d}x \\ &+ \int_\Omega \Big(2\mu\epsilon(\mathbf{u}) : \epsilon(\mathbf{z}') - q' \operatorname{div}\mathbf{u} - p \operatorname{div}\mathbf{z}'\Big)\,\mathrm{d}x \\ &+ \lambda'\Big(|\Omega| - |\Omega_0|\Big),\end{aligned} \tag{5.12}$$

with $(\mathbf{u}', p') = (\mathbf{u}'(\mathbf{V}), p'(\mathbf{V}))$ and $(\mathbf{z}', q') = (\mathbf{z}'(\mathbf{V}), q'(\mathbf{V}))$ the shape derivatives. Moreover, we have

$$\int_\Omega \Big(2\mu\epsilon(\mathbf{u}') : \epsilon(\mathbf{z}) - p' \operatorname{div}\mathbf{z} - q \operatorname{div}\mathbf{u}'\Big)\,\mathrm{d}x = B[(\mathbf{u}', p'), (\mathbf{z}, q)] \tag{5.13}$$

$$\int_\Omega \Big(2\mu\epsilon(\mathbf{u}) : \epsilon(\mathbf{z}') - q' \operatorname{div}\mathbf{u} - p \operatorname{div}\mathbf{z}'\Big)\,\mathrm{d}x = B[(\mathbf{u}, p), (\mathbf{z}', q')]. \tag{5.14}$$

Recalling (2.10) and (2.11), the shape derivatives $(\mathbf{u}', p')$ and $(\mathbf{z}', q')$ satisfy the boundary value problems

$$\begin{aligned} -\operatorname{div}(\mathbf{T}(\mathbf{u}', p')) &= 0 && \text{in } \Omega \\ \operatorname{div}\mathbf{u}' &= 0 && \text{in } \Omega \\ \mathbf{u}' &= 0 && \text{on } \Gamma_{\text{in}} \cup \Gamma_w \\ \mathbf{u}' &= -\nabla\mathbf{u}\,\boldsymbol{\nu}\, V && \text{on } \Gamma_s \\ \mathbf{T}(\mathbf{u}', p')\boldsymbol{\nu} &= 0 && \text{on } \Gamma_{\text{out}} \end{aligned} \tag{5.15}$$

and

$$\begin{aligned} -\operatorname{div}(\mathbf{T}(\mathbf{z}', q')) &= 0 && \text{in } \Omega \\ \operatorname{div}\mathbf{z}' &= 0 && \text{in } \Omega \\ \mathbf{z}' &= 0 && \text{on } \Gamma_{\text{in}} \cup \Gamma_w \\ \mathbf{z}' &= -\nabla(\mathbf{z} - \boldsymbol{\Phi}_\infty)\,\boldsymbol{\nu}\, V && \text{on } \Gamma_s \\ \mathbf{T}(\mathbf{z}', q')\boldsymbol{\nu} &= 0 && \text{on } \Gamma_{\text{out}}. \end{aligned} \tag{5.16}$$

Since the pair $(\mathbf{u}', p') \notin \mathbb{S}(\mathbf{0})$, combining (5.13) with (5.15) we obtain

$$B[(\mathbf{u}', p'), (\mathbf{z}, q)] = -\int_{\Gamma_s} \big(\mathbf{T}(\mathbf{z}, q)\boldsymbol{\nu}\big) \cdot \big(\nabla\mathbf{u}\,\boldsymbol{\nu}\big) V \,\mathrm{d}S.$$

We now exploit the fact that $\mathbf{u} = \mathbf{0}$ on Γ_s to write

$$\big(\mathbf{T}(\mathbf{z}, q)\boldsymbol{\nu}\big) \cdot \big(\nabla\mathbf{u}\,\boldsymbol{\nu}\big) = \mathbf{T}(\mathbf{z}, q) : \nabla\mathbf{u}. \tag{5.17}$$

By the definition of the Cauchy tensor $\mathbf{T}(\mathbf{z}, q)$,

$$\mathbf{T}(\mathbf{z}, q) : \nabla\mathbf{u} = 2\mu\epsilon(\mathbf{z}) : \epsilon(\mathbf{u}) - q \operatorname{div}\mathbf{u} = 2\mu\epsilon(\mathbf{z}) : \epsilon(\mathbf{u}),$$

whence

$$B[(\mathbf{u}', p'), (\mathbf{z}, q)] = -2\mu \int_{\Gamma_s} \epsilon(\mathbf{z}) : \epsilon(\mathbf{u}) V \,\mathrm{d}S.$$

Similarly, using now (5.16) in conjunction with the fact that $\mathbf{z} = \boldsymbol{\Phi}_\infty$ is constant in $\mathcal{N}(\Gamma_s)$ we infer that

$$B[(\mathbf{u}, p), (\mathbf{z}', q')] = -\int_{\Gamma_s} \big(\mathbf{T}(\mathbf{u}, p)\boldsymbol{\nu}\big) \cdot \big(\nabla\mathbf{z}\,\boldsymbol{\nu}\big) V \,\mathrm{d}S = -2\mu \int_{\Gamma_s} \epsilon(\mathbf{u}) : \epsilon(\mathbf{z}) V \,\mathrm{d}S.$$

Inserting the last two expressions into (5.12), and realizing that $|\Omega| = |\Omega_0|$, we conclude the asserted expression for the shape derivative of the Lagrangian. □

5.5. Dual weighted residual estimator

Let $\mathcal{T}$ be a conforming and shape regular triangulation of Ω. Let $\mathbb{U}_{\mathcal{T}} \times \mathbb{Q}_{\mathcal{T}} \subset [H^1(\Omega)]^d \times L^2(\Omega)$ be a stable pair of finite element spaces [18] for the Stokes equations so that $\mathbb{U}_{\mathcal{T}}$ (resp. $\mathbb{Q}_{\mathcal{T}}$) contains polynomials of degree $\leq m$ (resp. $\leq m-1$) for $m \geq 1$. Let

$$\mathbb{U}_{\mathcal{T}}(\mathbf{v}) = \{\mathbf{U} \in \mathbb{U}_{\mathcal{T}} : \mathbf{U} = \mathbf{v} \text{ on } \Gamma_d\} \tag{5.18}$$

where we assume $\mathbf{v} \in \mathbb{U}_{\mathcal{T}}$ and set $\mathbb{S}_{\mathcal{T}}(\mathbf{v}) := \mathbb{U}_{\mathcal{T}}(\mathbf{v}) \times \mathbb{Q}_{\mathcal{T}}$. The finite element approximation to the Stokes problem (5.3) reads

$$(\mathbf{U}, P) \in \mathbb{S}_{\mathcal{T}}(\mathbf{u_d}) : \quad B[(\mathbf{U}, P), (\mathbf{W}, \Phi)] = 0 \quad \forall (\mathbf{W}, \Phi) \in \mathbb{S}_{\mathcal{T}}(\mathbf{0}), \tag{5.19}$$

where we assume $\mathbf{u}_d \in \mathbb{U}_{\mathcal{T}}$.

We next evaluate the PDE error induced by the finite element method. To this end, we assume that the domain Ω is fixed, whence $J[\Omega, (\mathbf{u}, p), \lambda] = I[(\mathbf{u}, p)]$. In particular, we are interested in deriving an *a posteriori* error estimate for the quantity $\left|I[(\mathbf{u}, p)] - I[(\mathbf{U}, P)]\right|$ where $(\mathbf{u}, p)$ is the solution to (5.3) and $(\mathbf{U}, P)$ that of (5.19). Applying the abstract theory of the Dual Weighted Residual method presented in Section 3 we obtain the following result. Even though a similar estimate has been derived in [17], we present the proof now for completeness.

Lemma 5.3 (DWR estimate for the drag)**.** *The following error estimate holds*

$$\begin{aligned} \left|J[\Omega, (\mathbf{u}, p), \lambda] - J[\Omega, (\mathbf{U}, P), \lambda]\right| \leq \sum_{T \in \mathcal{T}} \Big(& \|r(\mathbf{U}, P)\|_{L^2(T)} \|\mathbf{z} - \mathbf{Z}\|_{L^2(T)} \\ & + \|j(\mathbf{U}, P)\|_{L^2(\partial T)} \|\mathbf{z} - \mathbf{Z}\|_{L^2(\partial T)} + \|\rho(\mathbf{U})\|_{L^2(T)} \|q - Q\|_{L^2(T)} \Big), \end{aligned} \tag{5.20}$$

where $(\mathbf{Z}, Q)$ *is the finite element approximation to the solution* $(\mathbf{z}, q)$ *of the adjoint problem* (5.10) *and*

$$r(\mathbf{U}, P)|_T := -\operatorname{div} \mathbf{T}(\mathbf{U}, P), \qquad \rho(\mathbf{U})|_T := \operatorname{div} \mathbf{U},$$

$$j(\mathbf{U}, P)|_S := \begin{cases} \frac{1}{2}[\mathbf{T}(\mathbf{U}, P)\boldsymbol{\nu}], & S \not\subset \partial\Omega, \\ \mathbf{T}(\mathbf{U}, P)\boldsymbol{\nu}, & S \subset \Gamma_{\text{out}}, \\ 0 & \textit{otherwise}, \end{cases}$$

where $[\cdot]$ *denotes the jump across the interelement side* S.

Proof. Applying (3.8) to (5.6), with linear functional I obeying (5.7), yields

$$J[\Omega, (\mathbf{u}, p), \lambda] - J[\Omega, (\mathbf{U}, P), \lambda] = B[(\mathbf{U}, P), (\mathbf{z} - \mathbf{Z}, q - Q)].$$

Integrating $B[(\mathbf{U}, P), (\mathbf{z} - \mathbf{Z}, q - Q)]$ by parts over the elements $T \in \mathcal{T}$, and collecting the boundary terms from adjacent elements to form jumps, we obtain the expression

$$B[(\mathbf{U}, P), (\mathbf{z} - \mathbf{Z}, q - Q)] = \sum_{T \in \mathcal{T}} \int_T r(\mathbf{U}, P)(\mathbf{z} - \mathbf{Z}) + \rho(\mathbf{U})(q - Q)\, \mathrm{d}x + \int_{\partial T} j(\mathbf{U}, P)(\mathbf{z} - \mathbf{Z})\, \mathrm{d}S.$$

Applying the Cauchy-Schwarz inequality leads to (5.20), which is consistent with (3.9). □

In view of the discussion following Lemma 3.2, especially (3.13)–(3.14), for the simulations below we use the following heuristic bound in the module `ESTIMATE` of the adaptive algorithm `ASQP`:

$$\left|J[\Omega, (\mathbf{u}, p), \lambda] - J[\Omega, (\mathbf{U}, P), \lambda]\right| \lesssim \sum_{T \in \mathcal{T}} \eta_{\mathcal{T}}(T).$$

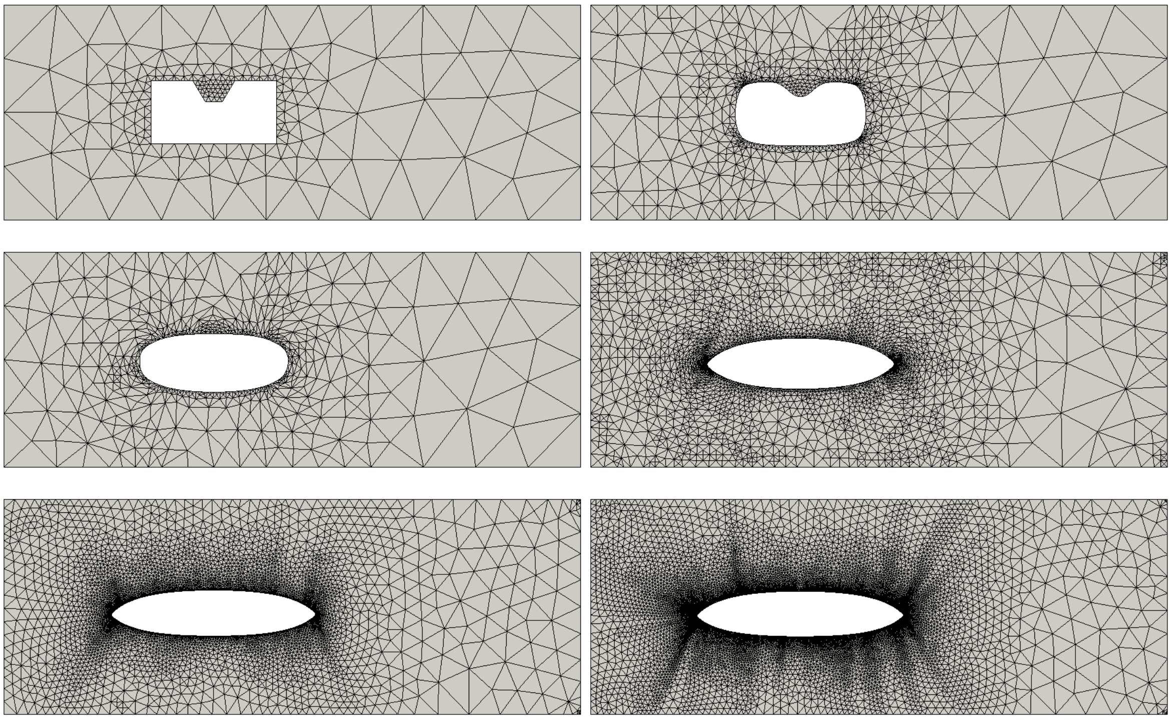

FIGURE 2. Drag optimization: snapshots at iterations $0, 10, 28, 50, 130$ and 174 of the evolution of a non-convex obstacle in a channel flow using the `ASQP` algorithm to find the optimal rugby-ball shape that minimizes its drag. The flow is modeled as a stationary Stokes fluid. The obstacle is constrained to maintain its initial volume. Taylor-Hood finite elements with $m = 2$ (quadratics for velocity and linears for pressure) are employed for approximating both the state and adjoint problems. For the boundary we consider the Laplace-Beltrami operator with $\rho = 0.05$. It is worth noticing that the initial refinement due to the presence of (non-genuine) corners on the initial shape disappears later on, and new refinement appears around the (genuine) corners of the optimal shape. This is the combined effect of DWR (Sect. 3) and GCMM (Sect. 4.4).

where the explicit element indicators are given by

$$\eta_{\mathcal{T}}(T) := \Big(h_T^{1/2} \|r(\mathbf{U}, P)\|_{L^2(T)} + \|j(\mathbf{U})\|_{L^2(\partial T)} \Big) \sum_{j=1}^{m} h_T^j \|[D^j \mathbf{Z}]\|_{L^2(\partial T)} + h_T^{1/2} \|\rho(\mathbf{U})\|_{L^2(T)} \sum_{j=0}^{m-1} h_T^j \|[D^j Q]\|_{L^2(\partial T)}.$$

Since the various terms $h_T^j \|[D^j \mathbf{Z}]\|_{L^2(\partial T)}$ in this heuristic bound are expected to be of the same order, except for pathological situations, we just take $j = 1$ in our numerical implementation.

5.6. Numerical experiments

We perform the numerical simulations in two dimensions. In Figure 2 we show a sequence of meshes, starting from a non-convex initial obstacle, and arriving at the optimal rugby-ball shape, with the same volume [25, 26].

This simulation is rather demanding due to the non-convexity and non-smoothness of the initial obstacle shape. In Figure 2 we observe its evolution under the `ASQP` algorithm. It can be seen that DWR finds large

TABLE 1. Drag optimization: adaptivity statistics of the `ASQP` method for the first 35 iterations. The number of marked elements for refinement/coarsening is denoted with LB for Laplace-Beltrami and DWR for the dual weighted residual method. After iteration 35 the first occurrence of remeshing appears (**R**) due to failed mesh quality check. Subsequent remeshings occured after iterations 62, 100 and 154.

Iteration	2	3	4	5	8	9	10	17	18	23	26	28	30	31	32	33	35	**R**
DWR-ref		220	216	94	79		153					80	225		11	241	94	**R**
DWR-coars		34	51	54	48		90					778		63	43	23	135	**R**
LB-ref	11	16	19	19	13	19	36	7	37	6	10	62						**R**
LB-coars			2				8			2		1						**R**

estimators close to the corners of the initial shape, and forces some refinement in order to control the PDE error. Afterwards, using the direction dictated by the shape derivative $\delta_\Omega J[\Omega]$ and the scalar product $A_\Gamma(\cdot,\cdot)$, as well as GCMM, `ASQP` smooths out those corners and straightens out the non-convex part of Γ. A few iterations later the elements that were initially around the original corners are coarsened due to the current smoothness of Γ, and the elements around the newly formed corners are strongly refined to reduce both the PDE and geometric errors (see Fig. 3).

In Table 1 we show numerical data that document some aspects of the practical behavior of `ASQP`. During the first 10 iterations the refinement process dominates the coarsening procedure; this is mostly due to the detection of the initial corners. At iteration 28 a strong coarsening occurs in response to the flattening of the initially highly refined corners. The method is thus able to detect and coarsen fake corners (see Fig. 3). The first remeshing occurs after iteration 35. Subsequent remeshing occured after iterations 62, 100 and 154.

6. Optimization of an aortic-coronary by-pass

6.1. Cost functional, Lagrangian, and adjoint equation

We consider now a model of blood flow through an aortic-coronary by-pass. Let $\Omega \subset \mathbb{R}^d$, $d \geq 2$ be a bounded domain of $\mathbb{R}^d$ as depicted in Figure 4. Let the velocity-pressure pair $(\mathbf{u}, p) = (\mathbf{u}(\Omega), p(\Omega))$ solve the Stokes problem (5.1) in strong form or (5.3) in weak form. Let the finite element pair $(\mathbf{U}, P)$ solve the Galerkin problem (5.19). We are interested in the *total dissipated energy* in Ω, which is given by

$$I[\Omega, (\mathbf{u}, p)] := 2\mu \int_\Omega |\epsilon(\mathbf{u})|^2 \,\mathrm{d}x. \tag{6.1}$$

It is worth noticing that in two dimensions I measures the vorticity of an incompressible flow. The corresponding minimization problem has been considered in [27] to find the optimal design of an aorto-coronary by-pass (see also [28]). We supplement (6.1) with a penalization of the perimeter of Γ_s and thus consider for $\varepsilon > 0$ fixed

$$J[\Omega, (\mathbf{u}, p)] := I[\Omega, (\mathbf{u}, p)] + \varepsilon |\Gamma_s|. \tag{6.2}$$

According to (2.2), the Lagrangian associated with this shape optimization problem reads

$$\mathcal{L}[\Omega, (\mathbf{u}, p), (\mathbf{z}, q)] := J[\Omega, (\mathbf{u}, p)] - B[(\mathbf{u}, p), (\mathbf{z}, q)]. \tag{6.3}$$

Lemma 6.1 (adjoint equation for (6.3))**.** *The adjoint pair* $(\mathbf{z}, q)$ *satisfies the weak form*

$$(\mathbf{z}, q) \in \mathbb{S}(\mathbf{0}): \quad B[(\mathbf{w}, s), (\mathbf{z}, q)] = 4\mu \langle \epsilon(\mathbf{u}), \epsilon(\mathbf{w}) \rangle \quad \forall (\mathbf{w}, s) \in \mathbb{S}(\mathbf{0}), \tag{6.4}$$

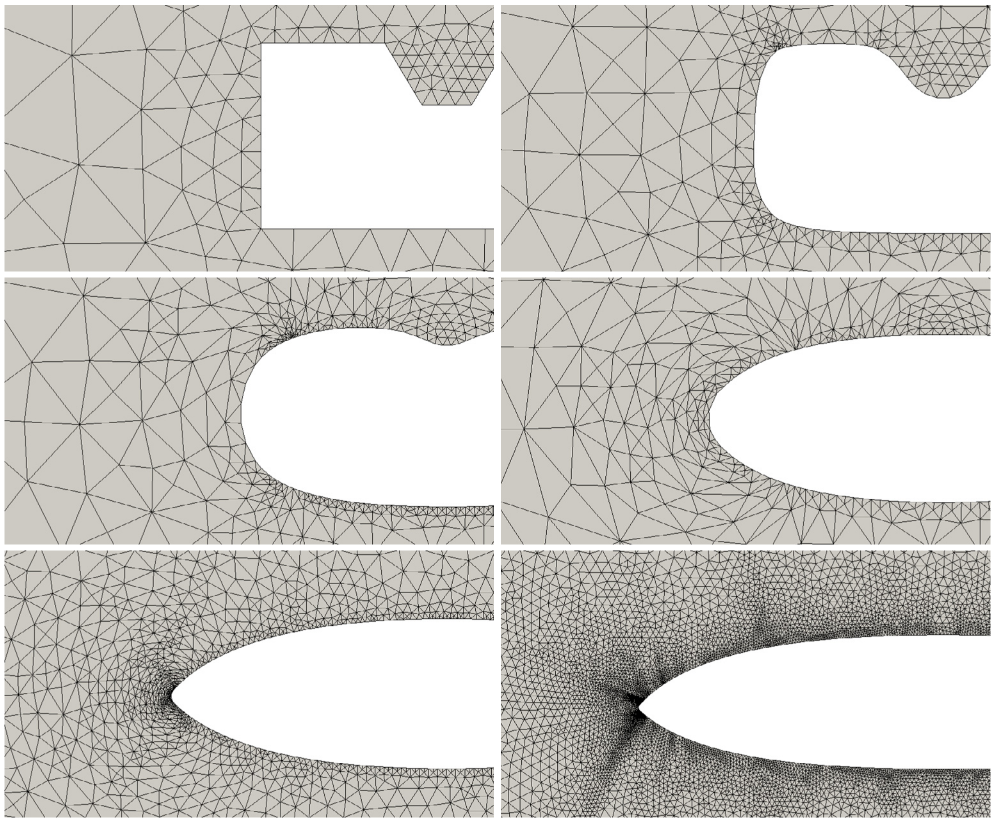

FIGURE 3. Drag optimization: detection of genuine geometric singularities (iterations 0, 5, 20, 32, 50 and 160). Zoom of the evolution of the non-convex obstacle towards the optimal shape that minimizes the drag with a given volume. The geometric singularities given by the artificial corners of the initial shape are quickly detected by the `ASQP` method. They are refined by the combined effect of the LB and DWR error estimates, and smoothed out by the energy minimizing iterations and GCMM (first three frames). A few iterations later the elements that were initially around the original corners are coarsened due to the current smoothness of Γ (forth frame). As the genuine singularity of the problem (the corner of the rugby ball) appears the `ASQP` method is able to recognize it and to refine around it (last two frames) so as to improve both the PDE and geometric approximation.

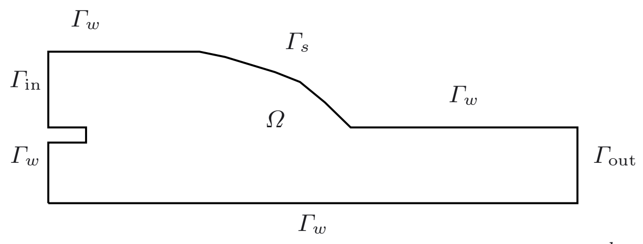

FIGURE 4. Domain Ω for coronary by-pass shape optimization: $\Omega \subset \mathbb{R}^d$, $d \geq 2$, is a bounded domain with boundary split into an inflow part Γ_{in}, an outflow part Γ_{out}, a part considered as a wall Γ_w, and a deformable part Γ_s, which is is the main design variable. The end points of Γ_s connecting to Γ_w are fixed and are not part of the optimization.

as well as the strong form

$$\begin{aligned} -\operatorname{div}(\mathbf{T}(\mathbf{z},q)) &= -4\mu\operatorname{div}(\epsilon(\mathbf{u})) && \text{in } \Omega,\\ \operatorname{div}\mathbf{z} &= 0 && \text{in } \Omega,\\ \mathbf{z} &= \mathbf{0} && \text{on } \Gamma_d,\\ \mathbf{T}(\mathbf{z},q)\boldsymbol{\nu} &= 4\mu\epsilon(\mathbf{u})\boldsymbol{\nu} && \text{on } \Gamma_{\text{out}}, \end{aligned} \tag{6.5}$$

Proof. It suffices to differentiate $\mathcal{L}$ in (6.3) with respect to $(\mathbf{u},p)$ to get (6.4) and to integrate (6.4) by parts to obtain (6.5). □

6.2. Shape derivative

We now compute $\delta_\Omega\mathcal{L}[\Omega,(\mathbf{u},p)(\Omega),(\mathbf{z},q)(\Omega)]$ for $\mathcal{L}$ in (6.3).

Lemma 6.2 (shape derivative of (6.3))**.** *Let* $(\mathbf{u},p)$ *be the solution of* (5.3) *and* $(\mathbf{z},q)$ *be that of* (6.4)*. The shape derivative of* $\mathcal{L}[\Omega,(\mathbf{u},p)(\Omega),(\mathbf{z},q)(\Omega)]$ *in the direction* $\mathbf{V}$ *is given by*

$$\langle\delta_\Omega\mathcal{L}[\Omega,(\mathbf{u},p)(\Omega),(\mathbf{z},q)(\Omega)],\mathbf{V}\rangle = 2\mu\int_{\Gamma_s}\epsilon(\mathbf{u}):\big(\epsilon(\mathbf{z})-\epsilon(\mathbf{u})\big)\;V\,\mathrm{d}S+\varepsilon\int_{\Gamma_s}\kappa\;V\,\mathrm{d}S, \tag{6.6}$$

where κ *is the mean curvature of* Γ_s*.*

Proof. In view of (6.3), we can write

$$\mathcal{L}[\Omega,(\mathbf{u},p)(\Omega),(\mathbf{z},q)(\Omega)] = \int_\Omega\Big(2\mu|\epsilon(\mathbf{u})|^2+2\mu\epsilon(\mathbf{u}):\epsilon(\mathbf{z})-p\;\operatorname{div}\mathbf{z}-q\;\operatorname{div}\mathbf{u}\Big)\,\mathrm{d}x+\varepsilon\int_{\Gamma_s}\mathrm{d}S.$$

Applying the chain rule (2.13) we end up with

$$\begin{aligned}\langle\delta_\Omega\mathcal{L}[\Omega,(\mathbf{u},p)(\Omega),(\mathbf{z},q)(\Omega)],\mathbf{V}\rangle = &\int_{\Gamma_s}\Big(2\mu|\epsilon(\mathbf{u})|^2+2\mu\epsilon(\mathbf{u}):\epsilon(\mathbf{z})-p\;\operatorname{div}\mathbf{z}-q\;\operatorname{div}\mathbf{u}\Big)\;V\,\mathrm{d}S\\ &+\int_\Omega 4\mu\epsilon(\mathbf{u}):\epsilon(\mathbf{u}')\,\mathrm{d}x\\ &+\int_\Omega\Big(2\mu\epsilon(\mathbf{u}'):\epsilon(\mathbf{z})-p'\operatorname{div}\mathbf{z}-q\operatorname{div}\mathbf{u}'\Big)\,\mathrm{d}x\\ &+\int_\Omega\Big(2\mu\epsilon(\mathbf{u}):\epsilon(\mathbf{z}')-p\operatorname{div}\mathbf{z}'-q'\operatorname{div}\mathbf{u}\Big)\,\mathrm{d}x\\ &+\varepsilon\int_{\Gamma_s}\kappa V\,\mathrm{d}S,\end{aligned}$$

where the shape derivatives $(\mathbf{u}',p')=(\mathbf{u}'(\mathbf{V}),p'(\mathbf{V}))$ and $(\mathbf{z}',q')=(\mathbf{z}'(\mathbf{V}),q'(\mathbf{V}))$ satisfy (5.15) and (5.16) respectively, except that

$$-\operatorname{div}(\mathbf{T}(\mathbf{z}',q')) = -4\mu\operatorname{div}\big(\epsilon(\mathbf{u}')\big)\qquad\text{in }\Omega. \tag{6.7}$$

We next examine each term separately. We first observe that integration by parts yields

$$\int_\Omega 4\mu\epsilon(\mathbf{u}):\epsilon(\mathbf{u}')\,\mathrm{d}x = -4\mu\int_\Omega\operatorname{div}(\epsilon(\mathbf{u}))\mathbf{u}'\,\mathrm{d}x+4\mu\int_{\partial\Omega}(\epsilon(\mathbf{u})\boldsymbol{\nu})\cdot\mathbf{u}'\,\mathrm{d}S. \tag{6.8}$$

Employing (6.5), integrating by parts, and using the weak form of (5.15), we infer that

$$\begin{aligned}-4\mu\int_\Omega\operatorname{div}\big(\epsilon(\mathbf{u})\big)\mathbf{u}'\,\mathrm{d}x &= \int_\Omega\mathbf{T}(\mathbf{z},q):\nabla\mathbf{u}'\,\mathrm{d}x-\int_{\partial\Omega}\big(\mathbf{T}(\mathbf{z},q)\boldsymbol{\nu}\big)\cdot\mathbf{u}'\,\mathrm{d}S\\ &= B[(\mathbf{u}',p'),(\mathbf{z},q)]+\int_{\Gamma_s}\big(\nabla\mathbf{u}\boldsymbol{\nu}\big)\cdot\big(\mathbf{T}(\mathbf{z},q)\boldsymbol{\nu}\big)V\,\mathrm{d}S-\int_{\Gamma_{\text{out}}}\big(\mathbf{T}(\mathbf{z},q)\boldsymbol{\nu}\big)\cdot\mathbf{u}'\,\mathrm{d}S.\end{aligned}$$

Since $B[(\mathbf{u}', p'), (\mathbf{z}, q)] = 0$ for $(\mathbf{z}, q) \in \mathbb{S}(\mathbf{0})$ we deduce

$$-4\mu \int_{\Omega} \operatorname{div}\big(\epsilon(\mathbf{u})\big)\mathbf{u}' \, \mathrm{d}x = 2\mu \int_{\Gamma_s} \epsilon(\mathbf{u}) : \epsilon(\mathbf{z}) V \, \mathrm{d}S - \int_{\Gamma_{\text{out}}} \big(\mathbf{T}(\mathbf{z}, q)\boldsymbol{\nu}\big) \cdot \mathbf{u}' \, \mathrm{d}S,$$

where we have used, as in (5.17), that $(\nabla \mathbf{u}\, \boldsymbol{\nu}) \cdot (\mathbf{T}(\mathbf{z}, q)\boldsymbol{\nu}) = 2\mu\, \epsilon(\mathbf{u}) : \epsilon(\mathbf{z})$. We observe now that the last term cancels with a corresponding term in (6.8) because $\mathbf{T}(\mathbf{z}, q)\boldsymbol{\nu} = 4\mu\epsilon(\mathbf{u})\boldsymbol{\nu}$ according to (6.5). On the other hand, in light of (5.15), the remaining integral over $\partial\Omega \backslash \Gamma_{\text{out}}$ in (6.8) becomes

$$4\mu \int_{\Omega \backslash \Gamma_{\text{out}}} \big(\epsilon(\mathbf{u})\boldsymbol{\nu}\big) \cdot \mathbf{u}' \, \mathrm{d}S = -4\mu \int_{\Gamma_s} \big(\epsilon(\mathbf{u})\boldsymbol{\nu}\big) \cdot \nabla \mathbf{u}\boldsymbol{\nu} V \, \mathrm{d}S = -4\mu \int_{\Gamma_s} \epsilon(\mathbf{u}) : \epsilon(\mathbf{u}) V \, \mathrm{d}S,$$

where we have used again the argument in (5.17) to write $\big(\epsilon(\mathbf{u})\boldsymbol{\nu}\big) \cdot \nabla \mathbf{u}\boldsymbol{\nu} = \epsilon(\mathbf{u}) : \epsilon(\mathbf{u})$. This implies

$$\int_{\Omega} 4\mu\epsilon(\mathbf{u}) : \epsilon(\mathbf{u}') \, \mathrm{d}x = 2\mu \int_{\Gamma_s} \epsilon(\mathbf{u}) : \epsilon(\mathbf{z}) V \, \mathrm{d}S - 4\mu \int_{\Gamma_s} \epsilon(\mathbf{u}) : \epsilon(\mathbf{u}) V \, \mathrm{d}S.$$

Similarly, since $(\mathbf{z}, q) \in \mathbb{S}(0)$, the weak form of (5.15) yields

$$\int_{\Omega} \Big(2\mu\epsilon(\mathbf{u}') : \epsilon(\mathbf{z}) - p' \operatorname{div} \mathbf{z} - q \operatorname{div} \mathbf{u}'\Big) \, \mathrm{d}x = B[(\mathbf{u}', p'), (\mathbf{z}, q)] = 0,$$

whereas, proceeding as before

$$\begin{aligned}
\int_{\Omega} \Big(2\mu\epsilon(\mathbf{u}) : \epsilon(\mathbf{z}') - p \operatorname{div} \mathbf{z}' - q' \operatorname{div} \mathbf{u}\Big) \, \mathrm{d}x &= B[(\mathbf{u}, p), (\mathbf{z}', q')] \\
&= \int_{\Gamma_s} (\mathbf{T}(\mathbf{u}, p)\boldsymbol{\nu}) \cdot \mathbf{z}' \, \mathrm{d}S \\
&= -\int_{\Gamma_s} (\mathbf{T}(\mathbf{u}, p)\boldsymbol{\nu}) \cdot (\nabla \mathbf{z}\, \boldsymbol{\nu}) V \, \mathrm{d}S \\
&= -2\mu \int_{\Gamma_s} \epsilon(\mathbf{u}) : \epsilon(\mathbf{z}) V \, \mathrm{d}S
\end{aligned}$$

because of (5.17); note that we do not resort to (6.7) because $(\mathbf{z}', q')$ is viewed as a test function. Collecting the expressions above we arrive easily at the desired formula (6.6). □

6.3. Dual weighted residual estimator

We now estimate the PDE error induced by the finite element approximation and assume that the domain Ω is fixed; thus $J[\Omega, (\mathbf{u}, p)] = I[(\mathbf{u}, p)]$.

Lemma 6.3 (DWR estimate for the by-pass)**.** *The following error estimate holds*

$$\begin{aligned}
|J[\Omega, (\mathbf{u}, p)] - J[\Omega, (\mathbf{U}, P)]| \leq \frac{1}{2} \sum_{T \in \mathcal{T}} \Big(&\|r^*(\mathbf{U}, \mathbf{Z}, Q)\|_{L^2(T)} \|\mathbf{u} - \mathbf{U}\|_{L^2(T)} \\
&+ \|j^*(\mathbf{U}, \mathbf{Z}, Q)\|_{L^2(\partial T)} \|\mathbf{u} - \mathbf{U}\|_{L^2(\partial T)} \\
&+ \|\rho^*(\mathbf{Z})\|_{L^2(T)} \|p - P\|_{L^2(T)} \\
&+ \|r(\mathbf{U}, P)\|_{L^2(T)} \|\mathbf{z} - \mathbf{Z}\|_{L^2(T)} \\
&+ \|j(\mathbf{U}, P)\|_{L^2(\partial T)} \|\mathbf{z} - \mathbf{Z}\|_{L^2(\partial T)} \\
&+ \|\rho(\mathbf{U})\|_{L^2(T)} \|q - Q\|_{L^2(T)} \Big),
\end{aligned} \tag{6.9}$$

where $(\mathbf{U}, P)$ *and* $(\mathbf{Z}, Q)$ *are the finite element solutions of the state problem* (5.1) *and the adjoint problem* (6.5)*, respectively, and the residuals are given by*

$$\begin{aligned} r^*(\mathbf{U},\mathbf{Z},Q)|_T &:= -4\mu\,\mathrm{div}(\epsilon(\mathbf{U})) + \mathrm{div}\,\mathbf{T}(\mathbf{Z},Q), \qquad \rho^*(\mathbf{Z})|_T := \mathrm{div}\,\mathbf{Z}, \\ r(\mathbf{U},P)|_T &:= \mathrm{div}\,\mathbf{T}(\mathbf{U},P), \qquad \rho(\mathbf{U})|_T := \mathrm{div}\,\mathbf{U}, \end{aligned}$$

$$j^*(\mathbf{U},\mathbf{Z},Q)|_S = \begin{cases} \frac{1}{2}[\![4\mu\epsilon(\mathbf{U})\boldsymbol{\nu} - \mathbf{T}(\mathbf{Z},Q)\boldsymbol{\nu}]\!], & S \not\subset \partial\Omega, \\ 4\mu\epsilon(\mathbf{U})\boldsymbol{\nu} - \mathbf{T}(\mathbf{Z},Q)\boldsymbol{\nu}, & S \subset \Gamma_{\mathrm{out}}, \\ 0, & \textit{otherwise}, \end{cases}$$

$$j(\mathbf{U},P)|_S = \begin{cases} -\frac{1}{2}[\![\mathbf{T}(\mathbf{U},P)\boldsymbol{\nu}]\!], & S \not\subset \partial\Omega, \\ -\mathbf{T}(\mathbf{U},P)\boldsymbol{\nu}, & S \subset \Gamma_{\mathrm{out}}, \\ 0, & \textit{otherwise}, \end{cases}$$

with $[\![\cdot]\!]$ *denoting the jump across the interelement side* S*.*

Proof. In view of (3.3) and (3.4) for J obeying (6.2), we can write for all $(\mathbf{w}, s) \in \mathbb{S}$

$$\begin{aligned} R((\mathbf{U},P),(\mathbf{Z},Q);(\mathbf{w},s)) &= -B[(\mathbf{U},P),(\mathbf{w},s)) \\ R^*((\mathbf{U},P),(\mathbf{Z},Q);(\mathbf{w},s)) &= 4\mu\int_\Omega \epsilon(\mathbf{U}) : \epsilon(\mathbf{w})\,\mathrm{d}x - B[(\mathbf{w},s),(\mathbf{Z},Q)]. \end{aligned}$$

Therefore, applying (3.6) and realizing that the remainder $E = 0$ because J is quadratic, we obtain

$$\begin{aligned} J[\Omega,(\mathbf{u},p)] - J[\Omega,(\mathbf{U},P)] = \frac{1}{2}\Bigg\{ & 4\mu\int_\Omega \epsilon(\mathbf{U}) : \epsilon(\mathbf{u}-\mathbf{U})\,\mathrm{d}x \\ & -B[(\mathbf{u}-\mathbf{U},p-P),(\mathbf{Z},Q)] - B[(\mathbf{U},P),(\mathbf{z}-\mathbf{Z},q-Q)]\Bigg\}. \end{aligned}$$

Splitting the integrals over elements $T \in \mathcal{T}$, integrating by parts, and collecting the boundary terms from adjacent elements to form jumps, yields

$$\begin{aligned} J[\Omega,(\mathbf{u},p)] - J[\Omega,(\mathbf{U},P)] = \frac{1}{2}\Bigg\{ \sum_{T\in\mathcal{T}} \int_T r^*(\mathbf{U},\mathbf{Z},Q)(\mathbf{u}-\mathbf{U}) + \int_{\partial T} j^*(\mathbf{U},\mathbf{Z},Q)(\mathbf{u}-\mathbf{U}) \\ + \int_T \rho^*(\mathbf{Z})(p-P) + \int_T r(\mathbf{U},P)(\mathbf{z}-\mathbf{Z}) + \int_{\partial T} j(\mathbf{U},P)(\mathbf{z}-\mathbf{Z}) + \int_T \rho(\mathbf{U})(q-Q) \Bigg\}. \end{aligned}$$

On using the Cauchy-Schwarz inequality we obtain the asserted estimate (6.9). □

In view of the discussion following Lemma 3.2, especially (3.13), for the simulations below we use the following heuristic bound in the module `ESTIMATE` of the adaptive algorithm `ASQP`:

$$\big|J[\Omega,(\mathbf{u},p)] - J[\Omega,(\mathbf{U},P)]\big| \lesssim \sum_{T\in\mathcal{T}} \eta_{\mathcal{T}}(T) + \eta^*_{\mathcal{T}}(T)$$

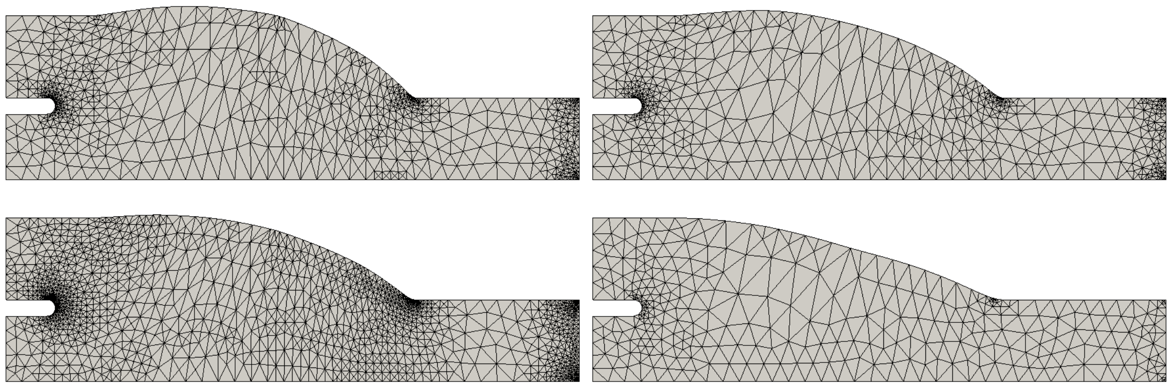

FIGURE 5. Coronary by-pass: optimal configurations obtained with different penalization parameters: $\varepsilon = 0.5\times10^{-5}$ (top-left), 0.8×10^{-5} (top-right), 1.0×10^{-5} (bottom-left) and 5.0×10^{-5} (bottom-right). Taylor-Hood finite element with $m = 2$ are employed for approximating state and adjoint problems and for the boundary we consider the pure Laplace-Beltrami opertor. Small values of the penalization parameter yield a Taylor-patch like geometry [33] (top-left), while large values of ϵ lead to by-pass configurations similar to those in [20] (bottom-right). Corresponding energy plots are shown in Figure 6.

where

$$\begin{aligned}\eta_{\mathcal{T}}(T) &:= \Big(h_T^{1/2}\|r(\mathbf{U},P)\|_{L^2(T)} + \|j(\mathbf{U},P)\|_{L^2(\partial T)}\Big)\sum_{j=1}^{m} h_T^j\|[D^j\mathbf{Z}]\|_{L^2(\partial T)}\\ &\quad + h_T^{1/2}\|\rho(\mathbf{U})\|_{L^2(T)}\sum_{j=0}^{m-1} h_T^j\|[D^jQ]\|_{L^2(\partial T)},\\ \eta_{\mathcal{T}}^*(T) &:= \Big(h_T^{1/2}\|r^*(\mathbf{U},\mathbf{Z},Q)\|_{L^2(T)} + \|j^*(\mathbf{U},\mathbf{Z},Q)\|_{L^2(\partial T)}\Big)\sum_{j=1}^{m} h_T^j\|[D^j\mathbf{U}]\|_{L^2(\partial T)}\\ &\quad + h_T^{1/2}\|\rho^*(\mathbf{Z})\|_{L^2(T)}\sum_{j=0}^{m-1} h_T^j\|[D^jP]\|_{L^2(\partial T)}.\end{aligned}$$

6.4. Numerical experiments

We perform the numerical simulations in two dimensions. We now elaborate on the iterations that led to the meshes shown in Figure 5 and explore the different behavior of the algorithm when changing the perimeter penalization parameter ε in the cost functional. In Figure 5 we depict the optimal bypass configurations obtained with the following penalization parameters: $\varepsilon = 0.5\times10^{-5}$, 0.8×10^{-5}, 1.0×10^{-5} and 5.0×10^{-5}. Small values of ε lead to a *Taylor-patch* like geometry [33], while large values of ε lead to by-pass configurations similar to those in [20]. Remeshing was unnecessary for any of these by-pass simulations.

We observe the effect of DWR that refines at the junction between the deformable curve Γ_s and the wall Γ_w, which is a reentrant corner of Ω. DWR is also sensitive to changes of boundary conditions and thus refines at the corners of the outflow boundary Γ_{out} where the traction-free condition changes to no-slip. Since the heuristic element indicators $\eta_{\mathcal{T}}(T)$ and $\eta_{\mathcal{T}}^*(T)$ of DWR contain terms of the form $h_T^j\|[D^j\mathbf{Z}]\|_{L^2(\partial T)}$ and $h_T^j\|[D^j\mathbf{Z}]\|_{L^2(\partial T)}$, which are expected to be of the same order except for pathological cases, we take $j = 1$ in our numerical experiments.

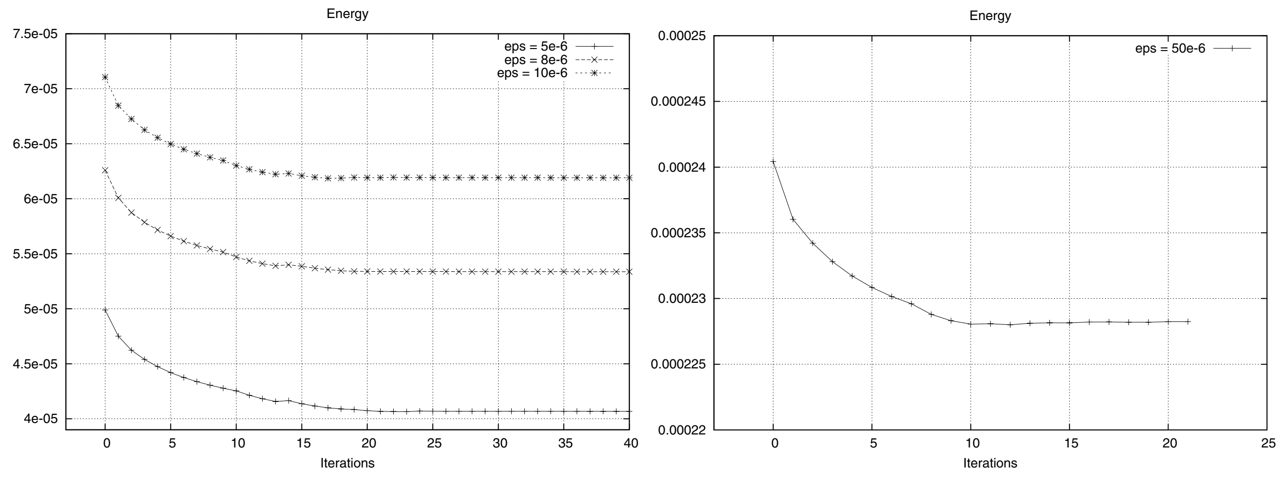

FIGURE 6. Coronary by-pass optimization: histories of convergence of the energy functional $J[\Omega_k, (\mathbf{U}_k, P_k)(\Omega_k)]$ in terms of the iteration counter k for different values of the perimeter penalization parameter (left: $\varepsilon = 0.5\times 10^{-5}$, 0.8×10^{-5}, 1.0×10^{-5}; right: 5.0×10^{-5}). Figure 5 displays the corresponding optimal configurations.

In Figure 6 we show the expected monotone behavior of $J[\Omega_k, (\mathbf{U}_k, P_k)(\Omega_k)]$ vs the number of iterations k for a discrete gradient flow, for the four values of ε listed above.

7. Conclusions and further comments

- We develop an adaptive sequential quadratic programming (ASQP) algorithm for shape optimization, which dynamically changes the tolerance and equidistributes the computational effort between the PDE and geometry approximation. This leads to rather coarse meshes in the early stages of the optimization when the shape is still far from optimal and full numerical accuracy is a waste.
- We give a formula that relates the PDE and geometric errors dynamically and depends on the best domain deformation to perturb the energy functional $J[\Omega]$. Such a deformation is characterized *via* shape differential calculus. The dual weighted residual (DWR) method controls the PDE error and the Laplace-Beltrami (LB) error indicator deals with the geometric error associated to domain deformations. This appears to be the first work with all these critical ingredients.
- We exploit the geometrically consistent mesh modification (GCMM) algorithms of [8] within ASQP. This allows for detection and removal of kinks that are fake (numerical artifacts) but not of those that are genuine to the problem. This is a new paradigm in adaptivity and its resolution is a crucial contribution. It is important to notice that both DWR and LB would insist on refining kinks regardless of their nature in the absence of a robust detection and correction mechanish.
- We apply the ASQP algorithm to two relevant problems governed by the stationary Stokes equation. We first examine a drag minimization problem, which has been the subject of intense research in the literature but without including adaptivity. We next study an aorto-coronary by-pass model. In both cases we discuss the ingredients of ASQP in detail and document its performance with interesting simulations. It is worth realizing that the applicability of ASQP goes far beyond viscous fluids.
- We present a novel interpolation error estimate which measures regularity in terms of jumps of the interpolant plus a higher order correction term (see Lem 3.2). This may be viewed as a *discrete Bramble-Hilbert* lemma and justifies asymptotic expressions used in the literature of DWR.

References

[1] G. Allaire, *Conception optimale de structures.* Springer-Verlag, Berlin (2007).

[2] P. Alotto, P. Girdinio, P. Molfino and M. Nervi, Mesh adaption and optimization techniques in magnet design. *IEEE Trans. Magn.* **32** (1996) 2954–2957.

[3] W. Bangerth and R. Rannacher, *Adaptive Finite Element Methods for Differential Equations*, Birkhäuser (2003)

[4] N.V. Banichuk, A. Falk and E. Stein, Mesh refinement for shape optimization, *Struct. Optim.* **9** (1995) 46–51.

[5] E. Bänsch, P. Morin and R.H. Nochetto, Surface diffusion of graphs: variational formulation, error analysis and simulation. *SIAM J. Numer. Anal.* **42** (2004) 773–799.

[6] R. Becker and R. Rannacher, An optimal control approach to *a posteriori* error estimation in finite element methods. *Acta Numer.* **10** (2001) 1–102.

[7] J.A. Bello, E. Fernandez-Cara, J. Lemoine and J. Simon, The differentiability of the drag with respect to the variations of a Lipschitz domain in a Navier-Stokes flow. *SIAM J. Control Optim.* **35** (1997) 626–640.

[8] A. Bonito and R.H. Nochetto and M.S. Pauletti, Geometrically consistent mesh modification. *SIAM J. Numer. Anal.* **48** (2010) 1877–1899.

[9] M. Burger, A framework for the construction of level set methods for shape optimization and reconstruction. *Interfaces Free Bound.* **5** (2003) 301–329.

[10] J. Céa, Conception optimale ou identification de formes: calcul rapide de la dérivée directionnelle de la fonction coût. *RAIRO Modél. Math. Anal. Numér.* **20** (1986) 371–402.

[11] F. de Gournay, Velocity extension for the level-set method and multiple eigenvalues in shape optimization. *SIAM J. Control Optim.* **45** (2006) 343–367.

[12] M.C. Delfour and J.-P. Zolésio, Shapes and Geometries. *SIAM Advances in Design and Control* **22** (2011).

[13] A. Demlow, Higher-order finite element methods and pointwise error estimates for elliptic problems on surfaces. *SIAM J. Numer. Anal.* **47** (2009) 805–827.

[14] A. Demlow and G. Dziuk, An adptive finite element method for the Laplace-Beltrami operator on implicitly defined surfaces. *SIAM J. Numer. Anal.* **45** (2007) 421–442.

[15] G. Dogan, P. Morin, R.H. Nochetto and M. Verani. Discrete gradient flows for shape optimization and applications. *Comput. Methods Appl. Mech. Engrg.* **196** (2007) 3898–3914.

[16] M. Giles and E. Süli, Adjoint methods for PDEs: *a posteriori* error analysis and postprocessing by duality. *Acta Numer.* **11** (2002) 145–236.

[17] M. Giles, M. Larson, J.M. Levenstam and E. Süli, *Adaptive error control for finite element approximation of the lift and drag coefficients in viscous flow.* Technical Report 1317 (1997) http://eprints.maths.ox.ac.uk/1317/.

[18] V. Girault and P.A. Raviart, *Finite Element Methods for Navier-Stokes Equations: Theory and Algorithms, Springer Series in Computational Mathematics* **5**. Springer-Verlag, Berlin (1986)

[19] A. Henderson, *ParaView Guide, A Parallel Visualization Application.* Kitware Inc. (2007).

[20] M. Lei, J.P. Archie and C. Kleinstreuer, Computational design of a bypass graft that minimizes wall shear stress gradients in the region of the distal anastomosis. *J. Vasc. Surg.* **25** (1997) 637–646.

[21] K. Mekchay, P. Morin, and R.H. Nochetto, AFEM for Laplace Beltrami operator on graphs: design and conditional contraction property. *Math. Comp.* **80** (2011) 625–648.

[22] B. Mohammadi, O. Pironneau, *Applied shape optimization for fluids.* Oxford University Press, Oxford (2001).

[23] M.S. Pauletti, *Parametric AFEM for geometric evolution equations and coupled fluid-membrane interaction.* Ph.D. thesis, University of Maryland, College Park, ProQuest LLC, Ann Arbor, MI (2008)

[24] M.S. Pauletti, Second order method for surface restoration. Submitted.

[25] O. Pironneau, On optimum profiles in Stokes flow. *J. Fluid Mech.* **59** (1973) 117–128.

[26] O. Pironneau, On optimum design in fluid mechanics. *J. Fluid Mech.* **64** (1974) 97–110.

[27] A. Quarteroni and G. Rozza, Optimal control and shape optimization of aorto-coronaric bypass anastomoses. *Math. Models Methods Appl. Sci.* **13** (2003) 1801–1823.

[28] G. Rozza, *Shape design by optimal flow control and reduced basis techniques: applications to bypass configurations in haemodynamics.* Ph.D.thesis, École Polytechnique Fédèrale de Lausanne (2005).

[29] J.R. Roche, Adaptive method for shape optimization, *6th World Congresses of Structural and Multidisciplinary Optimization.* Rio de Janeiro (2005).

[30] A. Schleupen, K. Maute and E. Ramm, Adaptive FE-procedures in shape optimization. *Struct. Multidisc. Optim.* **19** (2000) 282–302.

[31] A. Schmidt and K.G. Siebert, *Design of Adaptive Finite Element Software, The Finite Element Toolbox ALBERTA, Lecture Notes in Computational Science and Engineering* **42**. Springer, Berlin (2005).

[32] J. Sokołowski and J.-P. Zolésio, *Introduction to Shape Optimization.* Springer-Verlag, Berlin (1992).

[33] R.S. Taylor, A. Loh, R.J. McFarland, M. Cox and J.F. Chester, Improved technique for polytetrafluoroethylene bypass grafting: long-term results using anastomotic vein patches. *Br. J. Surg.* **79** (1992) 348–354.