Weierstraß-Institut
für Angewandte Analysis und Stochastik

Leibniz-Institut im Forschungsverbund Berlin e. V.

Preprint ISSN 2198-5855

Reconstruction of flow domain boundaries from velocity data via multi-step optimization of distributed resistance

Ondrej Partl[1], Ulrich Wilbrandt[1], Joaquín Mura[2], Alfonso Caiazzo[1]

submitted: April 8, 2022

[1] Weierstrass Institute
Mohrenstr. 39
10117 Berlin, Germany
E-Mail: ondrej.partl@wias-berlin.de
ulrich.wilbrandt@wias-berlin.de
alfonso.caiazzo@wias-berlin.de

[2] Universidad Técnica Federico Santa María
Avda. Vicuña Mackenna
3939 Santiago, Chile
E-Mail: joaquin.mura@usm.cl

No. 2929
Berlin 2022

2020 *Mathematics Subject Classification.* 49M41, 76D55, 76D07.

Key words and phrases. Brinkmann equation, gradient-based optimization, stabilized finite elements, boundary reconstruction.

Edited by
Weierstraß-Institut für Angewandte Analysis und Stochastik (WIAS)
Leibniz-Institut im Forschungsverbund Berlin e. V.
Mohrenstraße 39
10117 Berlin
Germany

Fax: +49 30 20372-303
E-Mail: preprint@wias-berlin.de
World Wide Web: http://www.wias-berlin.de/

Reconstruction of flow domain boundaries from velocity data via multi-step optimization of distributed resistance

Ondrej Partl, Ulrich Wilbrandt, Joaquín Mura, Alfonso Caiazzo

1 Abstract

We reconstruct the unknown shape of a flow domain using partially available internal velocity measurements. This inverse problem is motivated by applications in cardiovascular imaging where motion-sensitive protocols, such as phase-contrast MRI, can be used to recover three-dimensional velocity fields inside blood vessels. In this context, the information about the domain shape serves to quantify the severity of pathological conditions, such as vessel obstructions. We consider a flow modeled by a linear Brinkman problem with a fictitious resistance accounting for the presence of additional boundaries. To reconstruct these boundaries, we employ a multi-step gradient-based variational method to compute a resistance that minimizes the difference between the computed flow velocity and the available data. Afterward, we apply different post-processing steps to reconstruct the shape of the internal boundaries. To limit the overall computational cost, we use a stabilized equal-order finite element method. We prove the stability and the well-posedness of the considered optimization problem. We validate our method on three-dimensional examples based on synthetic velocity data and using realistic geometries obtained from cardiovascular imaging.

DOI 10.20347/WIAS.PREPRINT.2929 Berlin 2022

2 Introduction

The purpose of this paper is to propose, analyze, and validate a framework for the reconstruction of the shape of a fluid domain using internal velocity measurements, assuming that these data are available only in a subset of the computational domain.

Let $\Omega \subset \mathbb{R}^3$ be a given bounded, connected domain whose boundary $\partial\Omega$ can be decomposed as

$$\partial\Omega = \Gamma_{\text{in}} \cup \Gamma_{\text{wall}} \cup \Gamma_{\text{out}},$$

into an inlet Γ_{in}, outlet Γ_{out}, and a wall Γ_{wall}. We assume that each boundary part has a positive two-dimensional Hausdorff measure, and that Γ_{in} and Γ_{out} are not adjacent, i.e., $\overline{\Gamma_{\text{in}}} \cap \overline{\Gamma_{\text{out}}} = \emptyset$.

Our aim is to reconstruct a fluid domain $\Omega_{\text{fluid}} \subset \Omega$, whose boundary is partially unknown. However, we assume that the inlet and outlet boundaries are known, and they correspond to Γ_{in} and Γ_{out}, respectively, i.e., $\Gamma_{\text{in}} \cup \Gamma_{\text{out}} \subset \partial\Omega_{\text{fluid}}$.

The considered problem is inspired by applications in cardiovascular imaging, where motion-sensitive techniques such as phase-contrast MRI can be used to obtain three-dimensional time dependent images of the internal blood velocity field. In this case, Ω_{fluid} represents the lumen of the blood vessel, while $\Omega \setminus \Omega_{\text{fluid}}$ is an obstruction inside the blood vessel.

An important aspect of this work is that we assume that the velocity measurements are not available — or not usable, e.g., due to low resolution — on the whole computational domain: We assume that the velocity data are given on a subset $\Omega_{\text{meas}} \subset \Omega$. Notice that, in general, it holds $\Omega_{\text{meas}} \not\subset \Omega_{\text{fluid}}$ (see Figure 1), i.e., the measurements are available outside the physical fluid domain as well. In the context of medical imaging, this might be the case when cardiovascular devices implanted in the imaged vessels excessively disturb the MRI signal. Also notice that we do not consider any further assumptions on Ω_{meas}. In particular, we do not exclude the possibility that the sought boundary (or the region of interest) intersects the region where the data are missing.

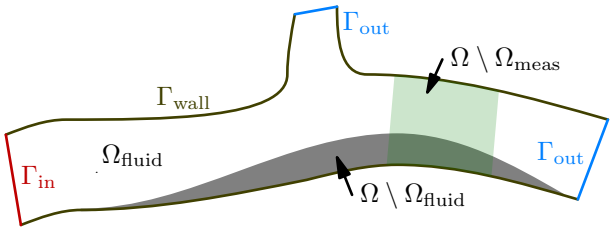


Figure 1: Sketch of the computational domain Ω: Ω_{fluid} (the white region) is the partially unknown fluid domain, which should be reconstructed. Γ_{in} and Γ_{out} denote the known inlet and outlet boundaries. Γ_{wall} stands for the impermeable boundaries of Ω, which are known as well. The region $\Omega \setminus \Omega_{\text{meas}}$ (the green region) models a subdomain where the measurements are not accessible, or their quality is too poor.

To identify the two subsets Ω_{fluid} and $\Omega \setminus \Omega_{\text{fluid}}$, we consider the following optimization framework. Firstly, we describe the flow in Ω via the Brinkman problem

$$\begin{cases} -\mu_{\text{eff}}\Delta\boldsymbol{u} + \sigma\boldsymbol{u} + \nabla p = \boldsymbol{f} & \text{in } \Omega, \\ \nabla \cdot \boldsymbol{u} = g & \text{in } \Omega, \\ \boldsymbol{u} = \boldsymbol{u}_{\text{D}} & \text{on } \Gamma_{\text{in}}, \\ \boldsymbol{u} = \boldsymbol{0} & \text{on } \Gamma_{\text{wall}}, \\ -\mu_{\text{eff}}\partial_{\boldsymbol{n}}\boldsymbol{u} + p\boldsymbol{n} = \mathbf{g}_{\boldsymbol{n}} & \text{on } \Gamma_{\text{out}}, \end{cases} \tag{2.1}$$

where $\boldsymbol{u} : \Omega \to \mathbb{R}^3$ [m/s] represents the fluid velocity, $p : \Omega \to \mathbb{R}$ [Pa] is the fluid pressure, $\boldsymbol{f} : \Omega \to \mathbb{R}^3$ [Pa/m] and $g : \Omega \to \mathbb{R}$ [1/s] are source terms, $\boldsymbol{u}_{\mathrm{D}} : \Gamma_{\mathrm{in}} \to \mathbb{R}^3$ [m/s] and $\mathbf{g}_{\boldsymbol{n}} : \Gamma_{\mathrm{out}} \to \mathbb{R}^3$ [Pa] stand for given data on the boundary, and $\mu_{\mathrm{eff}} > 0$ [Pa · s] denotes the effective viscosity.

The parameter $\sigma : \Omega \to [0, +\infty)$ [Pa · s/m^2] introduces a fictitious resistance to the flow: It is negligible in the fluid domain Ω_{fluid} and very high in the solid domain $\Omega \setminus \Omega_{\mathrm{fluid}}$. We use it to indirectly describe these domains as follows:

$$\sigma = \begin{cases} \gg 0 & \text{in } \Omega \setminus \Omega_{\mathrm{fluid}}, \\ \approx 0 & \text{in } \Omega_{\mathrm{fluid}}. \end{cases} \tag{2.2}$$

For more details on the fictitious domain approach in the context of the Brinkman problem, we refer, e.g., to Takhirov [18], Angot [4, 3], or to the recent results of Aguayo & Carrillo [1] (for a detailed convergence analysis in the case of mixed boundary conditions).

We define the resistance σ as a piecewise binary function, i.e., as a piecewise constant function with only two possible values (one for Ω_{fluid} and one for $\Omega \setminus \Omega_{\mathrm{fluid}}$), and seek a suitable σ using the following two-step approach:

(i) We compute the first approximation of σ by minimizing a quadratic tracking-type cost functional over piecewise constants with positive values. We solve this minimization problem using a gradient-based method. The cost functional is defined as the difference between the simulated and the observed velocity fields combined with an additional L^2-regularization term.

(ii) We apply different post-processing steps to map the first approximation onto a binary function. We formalize these steps from the point of view of the dimension reduction of the control space.

To validate our method, we consider several scenarios based on synthetic velocity data and on a domain modeling the ascending aorta, whose three-dimensional geometry was obtained from medical images, into which we artificially included an internal boundary. We check the accuracy of the shape estimation by comparing the outcome of the algorithm against the generated reference solution.

Our main contribution lies in

(i) the formalization and application of a gradient-based optimization method that accounts for the stabilization of an equal order linear finite element formulation,

(ii) the classification of the cells that enables us to adapt the control space to the available data,

(iii) the introduction of robust post-processing steps that enhance the overall stability of the method and yield satisfactory results in complex three-dimensional settings,

(iv) the validation of the approach also in the case where the data are only partially available, including the situation where the boundary intersects the region of the missing data.

An optimization framework based on an immersed resistance approach has recently been used by Aguayo et al. [2] to reconstruct aortic valve shapes (in 2D) from velocity MRI data. The authors considered the more general case of H^s-regularization together with the Navier–Stokes equations. Numerical validation was presented in 2D using the (inf-sup stable) Taylor–Hood finite elements. In contrast, we utilize equal-order stabilized finite elements (see, e.g., Blank et al. [5]), both for the forward and adjoint problems, which allows us to considerably reduce the complexity of the simulations, especially in 3D.

Our optimization framework can also be viewed as a phase-field approach in which the goal is to determine the optimal distribution of two different phases, parametrized using a space-dependent function. Phase-field approaches for shape optimization in the context of the Navier–Stokes equations have been proposed and analyzed by Garcke and his co-workers, considering integral state constraints [11] and surface functionals [10].

From a more general perspective, optimal control problems using fictitious domains have been described, e.g., in Kogut and Leugering [16, 15] and Haslinger et al. [13]. Moreover, one can also combine our approach with sparsity-inducing approaches involving L^1- or TV-regularization terms, which may improve the results further.

The paper is organized as follows: Section 3 introduces the flow problem in the continuous setting. The continuous optimization problem is presented and analyzed in Section 4. In Section 5, we discuss the discrete forward problem, the stabilized finite element method used for its numerical solution, and the corresponding discrete optimization problem. The complete optimization algorithm, including pre- and post-processing, is described in detail in Section 6, while the numerical results are shown in Sections 7–9. Finally, Section 10 draws conclusions from this study.

Notation: Throughout the paper, we will use standard notations for the Lebesgue and Sobolev spaces. A bold-faced letter will indicate the d-th Cartesian power, e.g., $\boldsymbol{H}^1(\Omega) = [H^1(\Omega)]^d$ or $\boldsymbol{v} = (v_1, \dots, v_d)$.

We also denote with $(\cdot,\cdot)_{\mathcal{O}}$ the L^2-scalar product on an open, bounded set $\mathcal{O} \subseteq \Omega$, while $\langle \cdot,\cdot \rangle_E$ will be used for integrals evaluated on the boundary or on boundary elements $E \subseteq \partial\Omega$. Moreover, $\|\cdot\|_{0,\mathcal{O}}$, $\|\cdot\|_{1,\mathcal{O}}$, and $\|\cdot\|_{\infty,\mathcal{O}}$ will denote the $L^2(\mathcal{O})$-, $H^1(\mathcal{O})$-, and $L^\infty(\mathcal{O})$-norm, respectively. To simplify the notation, we omit the subscript $\mathcal{O}$ in the above notations if $\mathcal{O} = \Omega$ and write $(\cdot,\cdot)$, $\|\cdot\|_0$, $\|\cdot\|_1$, and $\|\cdot\|_\infty$. Finally, $\boldsymbol{n}$ stands for the outer unit normal vector on $\partial\Omega$.

3 The Brinkman system

In this section, we discuss the existence and uniqueness of weak solutions for the Brinkman problem (2.1). Let $H^1_{\Gamma_{\text{wall}}}(\Omega)$ be the subspace of $H^1(\Omega)$ whose elements have a vanishing trace on Γ_{wall}. The traces of this space onto Γ_{in} lie in the space $H^{1/2}_{00}(\Gamma_{\text{in}})$. In other words, a function in $H^{1/2}_{00}(\Gamma_{\text{in}})$ is characterized by the fact that its extension by zero onto all of $\partial\Omega$ is in $H^{1/2}(\partial\Omega)$. Moreover, we will use the same symbol for the Dirichlet data on the inlet Γ_{in}, $\boldsymbol{u}_{\text{D}} \in \boldsymbol{H}^{1/2}_{00}(\Gamma_{\text{in}})$, and for its extension[1] into the domain Ω, $\boldsymbol{u}_{\text{D}} \in \boldsymbol{H}^1_{\Gamma_{\text{wall}}}(\Omega)$. The velocity and pressure test spaces are denoted by

$$\boldsymbol{V} = \boldsymbol{H}^1_{\Gamma_{\text{in}}\cup\Gamma_{\text{wall}}}(\Omega) \quad \text{and} \quad Q = L^2(\Omega),$$

respectively. Finally, the space of admissible functions σ (resistance) is denoted by

$$\Sigma := \{\sigma \in L^\infty(\Omega) \,|\, 0 \le \sigma(x) \le C_\Sigma \text{ a.e.}\} \tag{3.1}$$

for a constant $C_\Sigma \gg 0$.

For a given $\sigma \in \Sigma$, the weak formulation of the Brinkman problem (2.1) reads: Find $(\boldsymbol{u}, p) \in \boldsymbol{H}^1(\Omega) \times Q$, where $\boldsymbol{u} - \boldsymbol{u}_{\text{D}} \in \boldsymbol{V}$, such that for all $(\boldsymbol{v}, q) \in \boldsymbol{V} \times Q$, we have

$$\begin{cases} a(\sigma; \boldsymbol{u}, \boldsymbol{v}) - b(\boldsymbol{v}, p) = \ell_{\boldsymbol{f}}(\boldsymbol{v}), \\ b(\boldsymbol{u}, q) = \ell_g(q), \end{cases} \tag{3.2}$$

[1] The $\boldsymbol{H}^1(\Omega)$-norm of this extension is bounded by its $\boldsymbol{H}^{1/2}(\Gamma_{\text{in}})$-norm, see, for example, [19, Theorem 4.3.4].

where the linear forms $a : \Sigma \times \boldsymbol{H}^1(\Omega) \times \boldsymbol{V} \to \mathbb{R}$, $b : \boldsymbol{H}^1(\Omega) \times Q \to \mathbb{R}$, $\ell_{\boldsymbol{f}} : \boldsymbol{V} \to \mathbb{R}$, and $\ell_g : Q \to \mathbb{R}$ are defined as

$$\begin{aligned} a(\sigma; \boldsymbol{u}, \boldsymbol{v}) &= \mu_{\text{eff}}(\nabla \boldsymbol{u}, \nabla \boldsymbol{v}) + (\sigma \boldsymbol{u}, \boldsymbol{v}), \\ b(\boldsymbol{v}, p) &= (\nabla \cdot \boldsymbol{v}, p), \\ \ell_{\boldsymbol{f}}(\boldsymbol{v}) &= (\boldsymbol{f}, \boldsymbol{v}) + \langle \boldsymbol{g}_{\boldsymbol{n}}, \boldsymbol{v} \rangle_{\Gamma_{\text{out}}}, \\ \ell_g(q) &= (g, q). \end{aligned}$$

For these forms to be well defined, we consider the following regularity assumptions:

$$\boldsymbol{f} \in \boldsymbol{L}^2(\Omega), \quad g \in L^2(\Omega), \quad \text{and} \quad \boldsymbol{g}_{\boldsymbol{n}} \in \boldsymbol{L}^2(\Gamma_{\text{out}}). \tag{3.3}$$

The existence and uniqueness of the solution of (3.2) can be proven by applying standard saddle point theory (see, e.g., [19, Chapter 5.5] or the classical textbooks [7, 12] for this theory): The inf-sup condition

$$\inf_{q \in Q} \sup_{\boldsymbol{v} \in \boldsymbol{V}} \frac{b(\boldsymbol{v}, q)}{\|\boldsymbol{v}\|_1 \|q\|_0} \geq \beta > 0 \tag{3.4}$$

can be shown analogously as in the Stokes case, see [17, Proposition 5.3.2]. We will show continuity and coercivity of $a(\sigma; \cdot, \cdot)$: Let $\boldsymbol{u} \in \boldsymbol{H}^1(\Omega)$ and $\boldsymbol{v} \in \boldsymbol{V}$ be given. Then the Hölder inequality yields

$$\begin{aligned} a(\sigma; \boldsymbol{u}, \boldsymbol{v}) &\leq \mu_{\text{eff}} \|\nabla \boldsymbol{u}\|_0 \|\nabla \boldsymbol{v}\|_0 + \|\sigma\|_\infty \|\boldsymbol{u}\|_0 \|\boldsymbol{v}\|_0 \\ &\leq 2 \max\{\mu_{\text{eff}}, \|\sigma\|_\infty\} \|\boldsymbol{u}\|_1 \|\boldsymbol{v}\|_1 \\ &\leq c_u \|\boldsymbol{u}\|_1 \|\boldsymbol{v}\|_1, \end{aligned} \tag{3.5}$$

where $c_u = 2 \max\{\mu_{\text{eff}}, C_\Sigma\}$. Furthermore, since σ is allowed to vanish on a set of nonzero measure, we have

$$a(\sigma; \boldsymbol{v}, \boldsymbol{v}) \geq \mu_{\text{eff}} \|\nabla \boldsymbol{v}\|_0^2 \geq \frac{\mu_{\text{eff}}}{1 + c_P^2} \|\boldsymbol{v}\|_1^2 = c_l \|\boldsymbol{v}\|_1^2, \tag{3.6}$$

where c_P is the Poincaré constant of Ω, $\|\boldsymbol{v}\|_0 \leq c_P \|\nabla \boldsymbol{v}\|_0$, and $c_l = \mu_{\text{eff}}/(1 + c_P^2)$.

The inf-sup condition (3.4), the continuity, and the coercivity of $a(\sigma; \cdot, \cdot)$ allow us to prove the following result.

Theorem 3.1 (Existence and uniqueness of weak solution)**.** *Let $\mu_{\text{eff}} > 0$, $\sigma \in \Sigma$, $\boldsymbol{f} \in \boldsymbol{L}^2(\Omega)$, $g \in L^2(\Omega)$, $\boldsymbol{g}_{\boldsymbol{n}} \in \boldsymbol{L}^2(\Gamma_{\text{out}})$, and $\boldsymbol{u}_{\text{D}} \in \boldsymbol{H}^1_{\Gamma_{\text{wall}}}(\Omega)$ be the given data in* (3.2)*. Then there exists a unique weak solution $(\boldsymbol{u}, p) \in \boldsymbol{H}^1(\Omega) \times Q$ of* (3.2)*. Moreover, the following a priori bound holds:*

$$\|\boldsymbol{u}\|_1 + \|p\|_0 \leq c \left[\|\boldsymbol{f}\|_0 + \|\boldsymbol{g}_{\boldsymbol{n}}\|_{0,\Gamma_{\text{out}}} + \|g\|_0 + \|\boldsymbol{u}_{\text{D}}\|_{\boldsymbol{H}^{1/2}_{00}(\Gamma_{\text{in}})} \right], \tag{3.7}$$

where the constant c depends only on the shape of the domain Ω and on the constants β, c_u, and c_l defined in equations (3.4)–(3.6)*.*

Proof. Using $\boldsymbol{u}_0 = \boldsymbol{u} - \boldsymbol{u}_{\text{D}} \in \boldsymbol{V}$, the weak form (3.2) can be reformulated as

$$\begin{cases} a(\sigma; \boldsymbol{u}_0, \boldsymbol{v}) - b(\boldsymbol{v}, p) = \ell_{\boldsymbol{f}}(\boldsymbol{v}) - a(\sigma; \boldsymbol{u}_{\text{D}}, \boldsymbol{v}), \\ b(\boldsymbol{u}_0, q) = \ell_g(q) - b(\boldsymbol{u}_{\text{D}}, q). \end{cases} \tag{3.8}$$

The inf-sup condition (3.4), the coercivity of $a(\sigma;\cdot,\cdot)$, (3.6), together with standard results of saddle-point problems theory (e. g., [14, Theorem 3.18], [12, Theorem 4.1], or [19, Theorem 5.2.2]), imply the existence and uniqueness of the solution $(\boldsymbol{u}_0, p)$ to system (3.8), and hence of $(\boldsymbol{u}, p)$.

Using the Hölder inequality, we can bound the above right-hand sides by

$$|\ell_{\boldsymbol{f}}(\boldsymbol{v}) - a(\sigma;\boldsymbol{u}_{\mathrm{D}},\boldsymbol{v})| \le c\Big(\|\boldsymbol{f}\|_0 + \|\boldsymbol{g_n}\|_{0,\Gamma_{\mathrm{out}}} + c_u\|\boldsymbol{u}_{\mathrm{D}}\|_{\boldsymbol{H}^{1/2}_{00}(\Gamma_{\mathrm{in}})}\Big)\|\boldsymbol{v}\|_1,$$
$$|\ell_g(q) - b(\boldsymbol{u}_{\mathrm{D}},q)| \le c\Big(\|g\|_0 + \|\boldsymbol{u}_{\mathrm{D}}\|_{\boldsymbol{H}^{1/2}_{00}(\Gamma_{\mathrm{in}})}\Big)\|q\|_0,$$

where $c > 0$ stands for a generic domain-dependent constant. These inequalities, together with [19, equations (5.23) and (5.24)], yield the estimates

$$\|\boldsymbol{u}_0\|_1 \le c\left[\frac{1}{c_l}\Big(\|\boldsymbol{f}\|_0 + \|\boldsymbol{g_n}\|_{0,\Gamma_{\mathrm{out}}} + c_u\|\boldsymbol{u}_{\mathrm{D}}\|_{\boldsymbol{H}^{1/2}_{00}(\Gamma_{\mathrm{in}})}\Big) + \left(\frac{c_u}{c_l}+1\right)\frac{1}{\beta}\Big(\|g\|_0 + \|\boldsymbol{u}_{\mathrm{D}}\|_{\boldsymbol{H}^{1/2}_{00}(\Gamma_{\mathrm{in}})}\Big)\right],$$
$$\|p\|_0 \le c\left[\frac{1}{\beta}\left(\frac{c_u}{c_l}+1\right)\Big(\|\boldsymbol{f}\|_0 + \|\boldsymbol{g_n}\|_{0,\Gamma_{\mathrm{out}}} + c_u\|\boldsymbol{u}_{\mathrm{D}}\|_{\boldsymbol{H}^{1/2}_{00}(\Gamma_{\mathrm{in}})}\Big)\right.$$
$$\left. + \frac{c_u}{\beta^2}\left(\frac{c_u}{c_l}+1\right)\Big(\|g\|_0 + \|\boldsymbol{u}_{\mathrm{D}}\|_{\boldsymbol{H}^{1/2}_{00}(\Gamma_{\mathrm{in}})}\Big)\right].$$

Finally, the estimate (3.7) follows from $\|\boldsymbol{u}\|_1 \le \|\boldsymbol{u}_0\|_1 + \|\boldsymbol{u}_{\mathrm{D}}\|_1$. □

Next, we define an operator that maps a given resistance $\sigma \in \Sigma$ to a weak solution $(\boldsymbol{u}, p)$, which will be used in the definition of the optimization problem and its analysis.

Definition 3.2 (Control-to-state-operator)**.** *The control-to-state operator* $\tau\colon \Sigma \to \boldsymbol{H}^1(\Omega)\times Q$ *is defined as the map* $\sigma \mapsto \tau(\sigma) = (\boldsymbol{u}, p) \in \boldsymbol{H}^1(\Omega)\times Q$ *such that the state* $(\boldsymbol{u}, p)$ *solves the weak Brinkman problem* (3.2) *for the control* σ*.*

According to Theorem 3.1, τ is well defined. Due to (3.7), it is also uniformly bounded, i.e.,

$$\|\tau(\sigma)\|_{\boldsymbol{H}^1(\Omega)\times L^2(\Omega)} \le C, \tag{3.9}$$

where the constant C does not depend on σ.

4 Optimization problem

We formalize and analyze the problem of searching for a resistance $\sigma \in \Sigma$ that corresponds to a solution of the Brinkman problem (3.2) close to available velocity measurements.

4.1 Formulation

For given measurements $\hat{\boldsymbol{u}} \in L^2(\Omega_{\mathrm{meas}})$ [m/s], a positive constant α [$\mathrm{m}^6 \cdot \mathrm{s}^{-4} \cdot \mathrm{Pa}^{-2}$], and $\sigma \in \Sigma$, we define the cost function $J\colon \boldsymbol{V}\times Q\times\Sigma \to \mathbb{R}$ as

$$J((\boldsymbol{u},p),\sigma) \coloneqq \frac{1}{2}\|\boldsymbol{u}-\hat{\boldsymbol{u}}\|^2_{0,\Omega_{\mathrm{meas}}} + \frac{\alpha}{2}\|\sigma\|_0^2. \tag{4.1}$$

Using the control-to-state operator τ, we define the cost functional $F(\sigma) \coloneqq J(\tau(\sigma), \sigma)$ and consider the problem

$$\min_{\sigma \in \Sigma} F(\sigma). \tag{4.2}$$

Due to the coupling of control and state in the underlying equation, problem (4.2) is a bilinear optimization problem.

Theorem 4.1 (Existence of optimal solutions)**.** *The bilinear optimization problem* (4.2) *has a solution* $\sigma \in \Sigma$*.*

Proof. Since F is bounded from below, and Σ is non-empty, there exists an infimum $\hat{F} \coloneqq \inf_{\sigma \in \Sigma} F(\sigma)$ and an infimizing sequence of controls $\sigma_n \in \Sigma$ such that $F(\sigma_n) \to \hat{F}$.

By (3.1), (σ_n) can be regarded as a sequence in $[L^1(\Omega)]^* = L^\infty(\Omega)$ enclosed in a ball of radius C_Σ, and by the theorem of Banach–Alaoglu, this ball is compact, which guarantees that there is a subsequence of (σ_n) converging in the weak-$*$ topology to some $\sigma \in \Sigma$, i.e.,

$$\sigma_n \overset{*}{\rightharpoonup} \sigma \quad \text{in } \Sigma \subset L^\infty(\Omega). \tag{4.3}$$

This subsequence defines a sequence of solutions $(\tau(\sigma_n))$, $\tau(\sigma_n) = (\boldsymbol{u}_n, p_n)$, that is bounded (by (3.9)), and thus contains a subsequence converging weakly to some $(\boldsymbol{u}, p)$, i.e.,

$$\tau(\sigma_n) = (\boldsymbol{u}_n, p_n) \rightharpoonup (\boldsymbol{u}, p). \tag{4.4}$$

We will show that $(\boldsymbol{u}, p) = \tau(\sigma)$: We will do it by using the weak formulation (3.2) and proving that the sequences of zeros $(a(\sigma_n; \boldsymbol{u}_n, \boldsymbol{v}) - b(\boldsymbol{v}, p_n) - \ell_{\boldsymbol{f}}(\boldsymbol{v}))$ and $(b(\boldsymbol{u}_n, q) - \ell_g(q), q)$ converge to $(a(\sigma; \boldsymbol{u}, \boldsymbol{v}) - b(\boldsymbol{v}, p) - \ell_{\boldsymbol{f}}(\boldsymbol{v}))$ and $(b(\boldsymbol{u}, q) - \ell_g(q), q)$, respectively.

For suitable subsequences, it holds:

- $b(\boldsymbol{u}_n, q) \to b(\boldsymbol{u}, q)$ because $\nabla \cdot \boldsymbol{u}_n \rightharpoonup \nabla \cdot \boldsymbol{u}$ in $L^2(\Omega)$ and $q \in L^2(\Omega)$.
- $b(\boldsymbol{v}, p_n) \to b(\boldsymbol{v}, p)$ because $p_n \rightharpoonup p$ in $L^2(\Omega)$ and $\nabla \cdot \boldsymbol{v} \in L^2(\Omega)$.
- $(\mu_{\text{eff}} \nabla \boldsymbol{u}_n, \nabla \boldsymbol{v}) \to (\mu_{\text{eff}} \nabla \boldsymbol{u}, \nabla \boldsymbol{v})$ because $\nabla \boldsymbol{u}_n \rightharpoonup \nabla \boldsymbol{u}$ in $\boldsymbol{L}^2(\Omega)$ and $\nabla \boldsymbol{v} \in \boldsymbol{L}^2(\Omega)$.

Finally, the difference of the remaining resistance terms can be bounded as follows:

$$\begin{aligned} |(\sigma_n \boldsymbol{u}_n, \boldsymbol{v}) - (\sigma \boldsymbol{u}, \boldsymbol{v})| &\leq |((\sigma_n - \sigma)\boldsymbol{u}, \boldsymbol{v})| + |(\sigma_n(\boldsymbol{u}_n - \boldsymbol{u}), \boldsymbol{v})| \\ &\leq |(\sigma_n - \sigma, \boldsymbol{u} \cdot \boldsymbol{v})| + C_\Sigma \|\boldsymbol{u}_n - \boldsymbol{u}\|_0 \|\boldsymbol{v}\|_0. \end{aligned}$$

The first term converges to zero because $\sigma_n \overset{*}{\rightharpoonup} \sigma$ in $L^\infty(\Omega) = [L^1(\Omega)]^*$ and $\boldsymbol{u} \cdot \boldsymbol{v} \in L^1(\Omega)$; the second because $(\boldsymbol{u}_n)$ converges strongly in $\boldsymbol{L}^2(\Omega)$ due to its weak convergence in $\boldsymbol{H}^1(\Omega)$ and the compact embedding of $\boldsymbol{H}^1(\Omega)$ into $\boldsymbol{L}^2(\Omega)$. Therefore, $(\boldsymbol{u}, p) = \tau(\sigma)$.

To conclude the proof, note that the objective J is convex and continuous, hence weakly lower semicontinuous. This means that

$$J((\boldsymbol{u}, p), \sigma) \leq \liminf_{n \to \infty} J((\boldsymbol{u}_n, p_n), \sigma_n),$$

and therefore $J((\boldsymbol{u}, p), \sigma) = F(\sigma) = \hat{F}$. □

4.2 Derivative

Since we want to solve the optimization problem (4.2) via a gradient-based scheme, we need the derivative of F. We compute it using the adjoint problem and the subsequent lemma.

For a given $\sigma \in \Sigma$, the adjoint problem reads as follows: Find $(\boldsymbol{w}, t) \in \boldsymbol{V} \times Q$ such that

$$\begin{cases} a(\sigma; \boldsymbol{w}, \boldsymbol{v}) + b(\boldsymbol{v}, t) = -(\boldsymbol{u}_\sigma - \hat{\boldsymbol{u}}, \boldsymbol{v})_{\boldsymbol{L}^2(\Omega_{\mathrm{meas}})}, \\ -b(\boldsymbol{w}, q) = 0 \end{cases} \tag{4.5}$$

for all $(\boldsymbol{v}, q) \in \boldsymbol{V} \times Q$. In (4.5), $\hat{\boldsymbol{u}}$ denotes the given measurements, and $\boldsymbol{u}_\sigma$ stands for the first component of $\tau(\sigma)$.

Lemma 4.2. *The control-to-state operator τ is continuously differentiable with respect to the L^∞–topology, and we have, for a given $\sigma \in \Sigma$,*

$$\tau'(\sigma)\delta = (\boldsymbol{z}, r) \ \ \forall \delta \in \Sigma, \tag{4.6}$$

where the pair $(\boldsymbol{z}, r) \in \boldsymbol{V} \times Q$ satisfies

$$\begin{cases} a(\sigma; \boldsymbol{z}, \boldsymbol{v}) - b(\boldsymbol{v}, r) = -(\delta \boldsymbol{u}_\sigma, \boldsymbol{v}), \\ b(\boldsymbol{z}, q) = 0 \end{cases} \tag{4.7}$$

for all $(\boldsymbol{v}, q) \in \boldsymbol{V} \times Q$. Here $\boldsymbol{u}_\sigma$ denotes the first component of $\tau(\sigma)$.

Remark 4.1. *The pair $(\boldsymbol{z}, r)$ in Lemma 4.2 is the weak solution to*

$$\begin{cases} -\mu_{\mathrm{eff}} \Delta \boldsymbol{z} + \sigma \boldsymbol{z} + \nabla r = -\delta \boldsymbol{u}_\sigma & \text{in } \Omega, \\ \nabla \cdot \boldsymbol{z} = 0 & \text{in } \Omega, \\ \boldsymbol{z} = \boldsymbol{0} & \text{on } \Gamma_{\mathrm{in}} \cup \Gamma_{\mathrm{wall}}, \\ -\mu_{\mathrm{eff}} \partial_{\boldsymbol{n}} \boldsymbol{z} + r\boldsymbol{n} = \boldsymbol{0} & \text{on } \Gamma_{\mathrm{out}}. \end{cases} \tag{4.8}$$

Proof of Lemma 4.2. We start by deriving two auxiliary estimates: For given $\sigma, \delta \in \Sigma$ such that $\sigma + \delta \in \Sigma$, denote $(\boldsymbol{u}_\sigma, p_\sigma) = \tau(\sigma)$ and $(\boldsymbol{u}_{\sigma+\delta}, p_{\sigma+\delta}) = \tau(\sigma + \delta)$. Then, their difference $(\bar{\boldsymbol{u}}, \bar{p}) = \tau(\sigma + \delta) - \tau(\sigma) \in \boldsymbol{V} \times Q$ solves the following weak equations for all $(\boldsymbol{v}, q) \in \boldsymbol{V} \times Q$:

$$\begin{cases} a(\sigma; \bar{\boldsymbol{u}}, \boldsymbol{v}) - b(\boldsymbol{v}, \bar{p}) = -(\delta \boldsymbol{u}_{\sigma+\delta}, \boldsymbol{v}), \\ b(\bar{\boldsymbol{u}}, q) = 0, \end{cases} \tag{4.9}$$

where the right-hand side can be bounded by

$$\|\delta \boldsymbol{u}_{\sigma+\delta}\|_0 \leq \|\delta\|_\infty \|\boldsymbol{u}_{\sigma+\delta}\|_0 \leq \|\delta\|_\infty \|\boldsymbol{u}_{\sigma+\delta}\|_1 \overset{(3.9)}{\leq} C\|\delta\|_\infty.$$

Applying Theorem 3.1 to (4.9) leads to the estimate

$$\|\bar{\boldsymbol{u}}\|_1 \leq c\|\delta \boldsymbol{u}_{\sigma+\delta}\|_0 \leq cC\|\delta\|_{L^\infty(\Omega)}. \tag{4.10}$$

Now, we can prove the assertion of the lemma by considering the pair

$$(\boldsymbol{U}, P) := \tau(\sigma + \delta) - \tau(\sigma) - \tau'(\sigma)\delta = (\bar{\boldsymbol{u}}, \bar{p}) - (\boldsymbol{z}, r) \ \in \boldsymbol{V} \times Q,$$

which solves, for all $(\boldsymbol{v}, q) \in \boldsymbol{V} \times Q$, the system

$$\begin{cases} a(\sigma; \boldsymbol{U}, \boldsymbol{v}) - b(\boldsymbol{v}, P) = -(\delta\bar{\boldsymbol{u}}, \boldsymbol{v}), \\ b(\boldsymbol{U}, q) = 0. \end{cases} \tag{4.11}$$

Applying Theorem 3.1 to (4.11) yields the estimates

$$\|\boldsymbol{U}\|_1 + \|P\|_0 \le c\|\delta\bar{\boldsymbol{u}}\|_0 \le c\|\delta\|_\infty \|\bar{\boldsymbol{u}}\|_1 \overset{(4.10)}{\le} Cc^2\|\delta\|_\infty^2.$$

This means that $\|(\boldsymbol{U}, P)\|_{\boldsymbol{V}\times Q} = \mathcal{O}(\|\delta\|_\infty^2)$, which proves (4.6). The continuity of the derivative τ' can be shown in a similar way. □

Using the chain rule, we can now compute the derivative $F'(\sigma)$ in the direction of a given increment $\delta \in \Sigma$. Denoting the partial derivatives of J by $J_{\boldsymbol{u}}$, J_p, and J_σ and using equations (4.5)–(4.7), we obtain:

$$\begin{aligned} F'(\sigma)\delta &= (J_{\boldsymbol{u}}(\tau(\sigma),\sigma), J_p(\tau(\sigma),\sigma))\tau'(\sigma)\delta + J_\sigma(\tau(\sigma),\sigma)\delta \\ &\overset{(4.6)}{=} (\boldsymbol{u}_\sigma - \hat{\boldsymbol{u}}, \boldsymbol{z})_{\Omega_{\mathrm{meas}}} + \alpha(\sigma,\delta) \\ &\overset{(4.5)}{=} -a(\sigma; \boldsymbol{w}, \boldsymbol{z}) - b(\boldsymbol{z}, t) + b(\boldsymbol{w}, r) + \alpha(\sigma,\delta) \\ &\overset{(4.7)}{=} (\delta\boldsymbol{u}_\sigma, \boldsymbol{w}) + \alpha(\sigma,\delta) \\ &= (\boldsymbol{u}_\sigma \cdot \boldsymbol{w} + \alpha\sigma, \delta). \end{aligned} \tag{4.12}$$

5 Discrete system

5.1 Discrete stabilized formulation of forward problem

We carry out the spatial discretization via the so-called non-symmetric Galerkin least-squares (GLS) stabilized finite element method (see, e.g., [9] or, for the analysis in the case of a generalized Brinkman system, [5]).

To do this, we first introduce a boundary conforming (fitted) and shape-regular triangulation $\mathcal{T}_h$ of Ω (see, e.g., Ciarlet [8, p. 124]), where the characteristic size h is defined as $h = \max_{T\in\mathcal{T}_h} h_T$, h_T being the diameter of the cell $T \in \mathcal{T}_h$.

With the help of the mesh $\mathcal{T}_h$, we define the piecewise linear and piecewise constant finite element spaces for the velocity and pressure as well as the control space:

$$\begin{aligned} \boldsymbol{W}_h &= \big\{\boldsymbol{v}_h \in \boldsymbol{H}^1(\Omega) \cap \boldsymbol{C}^0(\overline{\Omega}) \,\big|\, \boldsymbol{v}_h|_T \in \mathbb{P}_1(T) \ \forall T \in \mathcal{T}_h\big\}, \\ \boldsymbol{V}_h &= \boldsymbol{W}_h \cap \boldsymbol{V} = \boldsymbol{W}_h \cap \boldsymbol{H}^1_{\Gamma_{\mathrm{in}}\cup\Gamma_{\mathrm{wall}}}(\Omega), \\ Q_h &= \big\{q_h \in L^2(\Omega) \cap C^0(\overline{\Omega}) \,\big|\, q_h|_T \in \mathbb{P}_1(T) \ \forall T \in \mathcal{T}_h\big\}, \\ \Sigma_h &= \big\{\sigma_h \in \Sigma \,\big|\, \sigma_h|_T \in \mathbb{P}_0(T) \ \forall T \in \mathcal{T}_h\big\}. \end{aligned}$$

We also assume that the Dirichlet data belong to the velocity finite element space, i.e., $\boldsymbol{u}_{\mathrm{D}} \in \boldsymbol{W}_h$.

Remark 5.1. *In practice, the control space Σ_h can also be based on a mesh different from the one used for the numerical solution of the flow problem. This decouples the resolution of the flow field from that of σ_h, which is useful if the former has strict numerical stability requirements, but a crude approximation suffices for σ_h. Analogously, the data $\hat{\boldsymbol{u}}$ can be defined on a different mesh, e.g., a voxel-based one, as in the case of imaging data.*

Given a piecewise constant resistance $\sigma_h \in \Sigma_h$, the discretized version of the flow problem (3.2) reads: Find $(\boldsymbol{u}_h, p_h) \in \boldsymbol{W}_h \times Q_h$ such that for all $(\boldsymbol{v}_h, q_h) \in \boldsymbol{V}_h \times Q_h$, it holds

$$\begin{cases} a_h(\sigma_h; \boldsymbol{u}_h, \boldsymbol{v}_h) - b(\boldsymbol{v}_h, p_h) + d_h(\sigma_h; \boldsymbol{v}_h, p_h) & = \ell_{\boldsymbol{f},h}(\sigma_h; \boldsymbol{v}_h), \\ b(\boldsymbol{u}_h, q_h) + d_h(\sigma_h; \boldsymbol{u}_h, q_h) + c_h(p_h, q_h) & = \ell_{g,h}(q_h) \end{cases} \tag{5.1}$$

with

$$a_h(\sigma_h; \boldsymbol{u}_h, \boldsymbol{v}_h) = a(\sigma_h; \boldsymbol{u}_h, \boldsymbol{v}_h) + \alpha_{\mathrm{GLS}} \sum_{T \in \mathcal{T}_h} \frac{h_T^2}{\mu_{\mathrm{eff}}} (\sigma_h \boldsymbol{u}_h, \sigma_h \boldsymbol{v}_h)_T, \tag{5.2a}$$

$$d_h(\sigma_h; \boldsymbol{u}_h, q_h) = \alpha_{\mathrm{GLS}} \sum_{T \in \mathcal{T}_h} \frac{h_T^2}{\mu_{\mathrm{eff}}} (\sigma_h \boldsymbol{u}_h, \nabla q_h)_T, \tag{5.2b}$$

$$c_h(p_h, q_h) = \alpha_{\mathrm{GLS}} \sum_{T \in \mathcal{T}_h} \frac{h_T^2}{\mu_{\mathrm{eff}}} (\nabla p_h, \nabla q_h)_T, \tag{5.2c}$$

$$\ell_{\boldsymbol{f},h}(\sigma_h; \boldsymbol{v}_h) = \ell_{\boldsymbol{f}}(\boldsymbol{v}_h) + \alpha_{\mathrm{GLS}} \sum_{T \in \mathcal{T}_h} \frac{h_T^2}{\mu_{\mathrm{eff}}} (\boldsymbol{f}, \sigma_h \boldsymbol{v}_h)_T, \tag{5.2d}$$

$$\ell_{g,h}(q_h) = \ell_g(q_h) + \alpha_{\mathrm{GLS}} \sum_{T \in \mathcal{T}_h} \frac{h_T^2}{\mu_{\mathrm{eff}}} (\boldsymbol{f}, \nabla q_h)_T, \tag{5.2e}$$

where $\alpha_{\mathrm{GLS}} > 0$ is a dimensionless stabilization parameter. We assume that α_{GLS} is independent of the mesh size h and constant in space.

In what follows, we also write (5.1) as

$$A_h(\sigma_h; \boldsymbol{u}_h, p_h, \boldsymbol{v}_h, q_h) = \ell_{\boldsymbol{f},h}(\sigma_h; \boldsymbol{v}_h) + \ell_{g,h}(q_h),$$

where $A_h : \Sigma_h \times \boldsymbol{W}_h \times Q_h \times \boldsymbol{V}_h \times Q_h$ is defined by

$$\begin{aligned} A_h(\sigma_h; \boldsymbol{u}_h, p_h, \boldsymbol{v}_h, q_h) &= a(\sigma_h; \boldsymbol{u}_h, \boldsymbol{v}_h) - b(\boldsymbol{v}_h, p_h) + b(\boldsymbol{u}_h, q_h) \\ &\quad + \alpha_{\mathrm{GLS}} \sum_{T \in \mathcal{T}_h} \frac{h_T^2}{\mu_{\mathrm{eff}}} (\sigma_h \boldsymbol{u}_h + \nabla p_h, \sigma_h \boldsymbol{v}_h + \nabla q_h)_T. \end{aligned} \tag{5.3}$$

Definition 5.1. *The discrete control-to-state operator $\tau_h : \Sigma_h \to \boldsymbol{W}_h \times Q_h$ is defined as the map $\sigma_h \mapsto \tau_h(\sigma_h) = (\boldsymbol{u}_h, p_h)$ such that the state $(\boldsymbol{u}_h, p_h)$ solves the discrete weak Brinkman problem* (5.1) *for the control σ_h.*

The following theorem guarantees the well-posedness and uniform boundedness of τ_h in the case of pure Dirichlet boundary conditions.

Theorem 5.2 (Existence and uniqueness of discrete weak solution)**.** *Let $\mu_{\mathrm{eff}} > 0$, $\sigma_h \in \Sigma_h$, $\boldsymbol{f} \in \boldsymbol{L}^2(\Omega)$, $g \in L^2(\Omega)$. Let $\Gamma_{\mathrm{out}} = \emptyset$ (pure Dirichlet data), and $\boldsymbol{u}_{\mathrm{D}} \in \boldsymbol{W}_h \cap \boldsymbol{H}^1_{\Gamma_{\mathrm{wall}}}(\Omega)$ be the given data in* (5.1)*. Then there exists a unique weak solution $(\boldsymbol{u}_h, p_h) \in \boldsymbol{W}_h \times Q_h$ of* (5.1)*. Moreover, the following a priori bound holds:*

$$\|\boldsymbol{u}_h\|_1 + \|p_h\|_0 \le c(\|\boldsymbol{f}\|_0 + \|g\|_0 + \|\boldsymbol{u}_{\mathrm{D}}\|_1) =: C, \tag{5.4}$$

where the constants c and C are independent of σ_h.

For the proof, we refer the reader, e.g., to [5] or, for the particular case $\sigma = 0$, to [6, Section 6.4.4]. An extension of this result to the case of general boundary conditions is beyond the scope of this paper.

5.2 Discrete optimization problem

The discrete counterpart of the cost functional F is $F_h : \Sigma_h \to \mathbb{R}$, $F_h(\sigma_h) \coloneqq J(\tau_h(\sigma_h), \sigma_h)$, and the optimization problem to be solved is

$$\min_{\sigma_h \in \Sigma_h} F_h(\sigma_h). \tag{5.5}$$

This problem has a solution because Σ_h is compact, and F_h is continuous on Σ_h.

5.3 Derivative

In order to define a gradient-based optimization algorithm, we need to compute the gradient $F_h'(\sigma_h) \in \Sigma_h$ (where we identified Σ_h with its dual). We proceed analogously as in the continuous case, showing the differentiability of the discrete control-to-state operator τ_h, the adjoint equation, and the required expression for the gradient $F_h'(\sigma_h)$.

Lemma 5.3. *The discrete control-to-state operator τ_h is continuously differentiable with respect to the L^∞-topology, and we have for $\sigma_h \in \Sigma_h$*

$$\tau_h'(\sigma_h)\delta_h = (\boldsymbol{z}_h, r_h) \quad \forall \delta_h \in \Sigma_h, \tag{5.6}$$

where $(\boldsymbol{z}_h, r_h) \in \boldsymbol{V}_h \times Q_h$ satisfies

$$\begin{cases} a_h(\sigma_h; \boldsymbol{z}_h, \boldsymbol{v}_h) - b(\boldsymbol{v}_h, r_h) + d_h(\sigma_h; \boldsymbol{v}_h, r_h) = -(\delta_h \boldsymbol{u}_h, \boldsymbol{v}_h) \\ \qquad\qquad + \alpha_{\mathrm{GLS}} \sum_{T \in \mathcal{T}_h} \frac{h_T^2}{\mu_{\mathrm{eff}}} (\boldsymbol{f} - 2\sigma_h \boldsymbol{u}_h - \nabla p_h, \delta_h \boldsymbol{v}_h)_T, \\ b(\boldsymbol{z}_h, q_h) + d_h(\sigma_h; \boldsymbol{z}_h, q_h) + c_h(r_h, q_h) = -d_h(\delta_h; \boldsymbol{u}_h, q_h) \end{cases} \tag{5.7}$$

for all $(\boldsymbol{v}_h, q_h) \in \boldsymbol{V}_h \times Q_h$. In (5.7)*, we have $(\boldsymbol{u}_h, p_h) = \tau_h(\sigma_h)$.*

Remark 5.2. *The terms proportional to α_{GLS} on the right-hand side of the system* (5.7) *are due to the stabilization used in the finite element method. If $\alpha_{\mathrm{GLS}} = 0$, equation* (5.7) *reduces to the discretized version of its continuous counterpart* (4.7)*.*

Proof. For given $\sigma_h, \delta_h \in \Sigma_h$ such that $\sigma_h + \delta_h \in \Sigma_h$, denote $(\boldsymbol{u}_\sigma, p_\sigma) = \tau_h(\sigma_h)$ and $(\boldsymbol{u}_{\sigma+\delta}, p_{\sigma+\delta}) = \tau_h(\sigma_h + \delta_h)$. Then, their difference $(\bar{\boldsymbol{u}}, \bar{p}) = \tau_h(\sigma_h + \delta_h) - \tau_h(\sigma_h) \in \boldsymbol{V}_h \times Q_h$ solves the following weak equations for all $(\boldsymbol{v}_h, q_h) \in \boldsymbol{V}_h \times Q_h$:

$$\begin{cases} a_h(\sigma_h; \bar{\boldsymbol{u}}, \boldsymbol{v}_h) - b_h(\boldsymbol{v}_h, \bar{p}) + d_h(\sigma_h; \boldsymbol{v}_h, \bar{p}) = -(\delta_h \boldsymbol{u}_{\sigma+\delta}, \boldsymbol{v}_h) + e_h(\delta_h; \boldsymbol{v}_h, \boldsymbol{u}_{\sigma+\delta}, p_{\sigma+\delta}), \\ b_h(\bar{\boldsymbol{u}}, q_h) + d_h(\sigma_h; \bar{\boldsymbol{u}}, q_h) + c_h(\bar{p}, q_h) = -d_h(\delta_h, \boldsymbol{u}_{\sigma+\delta}, q_h), \end{cases} \tag{5.8}$$

with

$$e_h(\delta_h; \boldsymbol{v}_h, \boldsymbol{u}_{\sigma+\delta}, p_{\sigma+\delta}) \coloneqq \alpha_{\mathrm{GLS}} \sum_{T \in \mathcal{T}_h} \frac{h_T^2}{\mu_{\mathrm{eff}}} (\boldsymbol{f} - (2\sigma_h + \delta_h)\boldsymbol{u}_{\sigma+\delta} - \nabla p_{\sigma+\delta}, \delta_h \boldsymbol{v}_h)_T.$$

The terms in the right-hand side can be bounded by

$$\begin{aligned} |-(\delta_h \boldsymbol{u}_{\sigma+\delta}, \boldsymbol{v}_h) + e_h(\delta_h; \boldsymbol{v}_h, \boldsymbol{u}_{\sigma+\delta}, p_{\sigma+\delta})| &\le C \|\delta_h\|_\infty \|\boldsymbol{v}_h\|_0, \\ |d_h(\delta_h, \boldsymbol{u}_{\sigma+\delta}, q_h)| &\le \tilde{C} \|\delta_h\|_\infty \sum_{T \in \mathcal{T}_h} \|\nabla q_h\|_T \le C \|\delta_h\|_\infty \|q_h\|_0, \end{aligned} \tag{5.9}$$

where the generic constants C and $\tilde{C}$ depend on the involved functions but not on σ_h. Moreover, both C and $\tilde{C}$ depend on a constant from an inverse estimate of the form $\|\nabla q_h\|_T \leq c_{\text{inv}} h_T^{-1} \|q_h\|_0$ (see, for example, [14, Theorem C.30]).

Applying Theorem 5.2 to the system (5.8) leads to the estimate

$$\|\bar{\boldsymbol{u}}\|_1 + \|\bar{p}\|_0 \leq C \|\delta_h\|_\infty. \tag{5.10}$$

To show that the derivative of τ_h is as claimed, consider $(\boldsymbol{U}, P) = \tau_h(\sigma_h+\delta_h) - \tau_h(\sigma_h) - \tau_h'(\sigma_h)\delta_h = (\bar{\boldsymbol{u}}, \bar{p}) - (\boldsymbol{z}_h, r_h)$ that solves for all $(\boldsymbol{v}_h, q_h) \in \boldsymbol{V}_h \times Q_h$

$$\begin{cases} a_h(\sigma_h; U, \boldsymbol{v}_h) - b_h(\boldsymbol{v}_h, P) + d_h(\sigma_h; \boldsymbol{v}_h, P) = -(\delta_h \bar{\boldsymbol{u}}, \boldsymbol{v}_h) \\ \qquad\qquad - \alpha_{\text{GLS}} \sum_{T \in \mathcal{T}_h} \dfrac{h_T^2}{\mu_{\text{eff}}} (2\sigma_h \bar{\boldsymbol{u}} + \nabla \bar{p} + \delta_h \boldsymbol{u}_{\sigma+\delta}, \delta_h \boldsymbol{v}_h)_T, \\ b_h(U, q_h) + d_h(\sigma_h; U, q_h) + c_h(P, q_h) = -d_h(\delta_h; \bar{\boldsymbol{u}}, q_h). \end{cases}$$

Estimating the right-hand side of this system analogously as in (5.9), applying Theorem 5.2, and using the estimate (5.10) gives (for a generic positive constant C)

$$\|\boldsymbol{U}\|_1 + \|P\|_0 \leq C(\|\bar{\boldsymbol{u}}\|_1 + \|\bar{p}\|_0 + \|\delta_h\|_\infty \|\boldsymbol{u}_{\sigma+\delta}\|_0) \|\delta_h\|_\infty \leq C \|\delta_h\|_\infty^2.$$

This shows that $\|(\boldsymbol{U}, P)\|_{\boldsymbol{V}_h \times Q_h} = \mathcal{O}(\|\delta_h\|_\infty^2)$ and therefore the Lemma. □

The adjoint problem, for a given $\sigma_h \in \Sigma_h$, reads: Find $(\boldsymbol{w}_h, t_h) \in \boldsymbol{V}_h \times Q_h$ such that for all $(\boldsymbol{v}_h, q_h) \in \boldsymbol{V}_h \times Q_h$, it holds

$$\begin{cases} a_h(\sigma_h; \boldsymbol{w}_h, \boldsymbol{v}_h) + b(\boldsymbol{v}_h, t_h) + d_h(\sigma_h; \boldsymbol{v}_h, t_h) = -(\boldsymbol{u}_h - \hat{\boldsymbol{u}}, \boldsymbol{v}_h)_{\boldsymbol{L}^2(\Omega_{\text{meas}})}, \\ -b(\boldsymbol{w}_h, q_h) + d_h(\sigma_h; \boldsymbol{w}_h, q_h) + c_h(t_h, q_h) = 0. \end{cases} \tag{5.11}$$

The system (5.11) can be rewritten as

$$A_h^*(\sigma_h; \boldsymbol{w}_h, t_h, \boldsymbol{v}_h, q_h) = -(\boldsymbol{u}_h - \hat{\boldsymbol{u}}, \boldsymbol{v}_h)_{\boldsymbol{L}^2(\Omega_{\text{meas}})}, \tag{5.12}$$

where $A_h^*(\sigma_h; \boldsymbol{w}_h, t_h, \boldsymbol{v}_h, q_h)$ denotes the adjoint of A_h defined in (5.3).

Let $(\boldsymbol{w}_h, t_h) \in \boldsymbol{V}_h \times Q_h$ be the solution of (5.11). The derivative $F_h'(\sigma_h)$ in the direction of a given increment $\delta_h \in \Sigma_h$ is then determined by the chain rule:

$$\begin{aligned} F_h'(\sigma_h)\delta_h &= (J_{\boldsymbol{u}}(\tau_h(\sigma_h), \sigma_h), J_p(\tau_h(\sigma_h), \sigma_h)) \tau_h'(\sigma_h)\delta_h + J_\sigma(\tau_h(\sigma_h), \sigma_h)\delta_h \\ &\overset{(5.6)}{=} (\boldsymbol{u}_h - \hat{\boldsymbol{u}}, \boldsymbol{z}_h)_{\Omega_{\text{meas}}} + \alpha(\sigma_h, \delta_h) \\ &\overset{(5.12)}{=} -A_h^*(\sigma_h; \boldsymbol{w}_h, t_h, \boldsymbol{z}_h, r_h) + \alpha(\sigma_h, \delta_h) \\ &= -A_h(\sigma_h; \boldsymbol{z}_h, r_h, \boldsymbol{w}_h, t_h) + \alpha(\sigma_h, \delta_h). \end{aligned} \tag{5.13}$$

Using (5.7), we obtain

$$\begin{aligned} F_h'(\sigma_h)\delta_h &= (\delta_h \boldsymbol{u}_h, \boldsymbol{w}_h) - \alpha_{\text{GLS}} \sum_{T \in \mathcal{T}_h} \frac{h_T^2}{\mu_{\text{eff}}} (\boldsymbol{f} - 2\sigma_h \boldsymbol{u}_h - \nabla p_h, \delta_h \boldsymbol{w}_h)_T \\ &\quad + d_h(\delta_h; \boldsymbol{u}_h, t_h) + \alpha(\sigma_h, \delta_h) \\ &= (\boldsymbol{u}_h \cdot \boldsymbol{w}_h + \alpha\sigma_h + \alpha_{\text{GLS}}\, \eta_h, \delta_h)\,, \end{aligned} \tag{5.14}$$

where for each cell $T \in \mathcal{T}_h$, the term

$$\eta_h\big|_T = \eta_h(\sigma_h; \boldsymbol{u}_h, p_h, \boldsymbol{w}_h, t_h)\big|_T \coloneqq \frac{h_T^2}{\mu_{\text{eff}}} \left[(-\boldsymbol{f} + 2\sigma_h \boldsymbol{u}_h + \nabla p_h) \cdot \boldsymbol{w}_h + \boldsymbol{u}_h \cdot \nabla t_h\right] \tag{5.15}$$

represents the contribution due to the stabilization.

Remark 5.3 (Discrete optimality)**.** *Notice that when deriving a numerical scheme for the optimization problem* (4.2)*, the order of the steps matters: The order optimize-then-discretize (and stabilize) yields the derivative $F_h'(\sigma_h)\delta_h$ corresponding to the discretized version of* (4.12)*, whereas discretize (and stabilize)-then-optimize gives us* (5.14)*. The latter option includes the additional terms defined by* (5.15)*, which scale with the square of the characteristic mesh size used for the finite element method.*

5.4 Pre-processing

We introduce a *pre-processing* the goal of which is to exploit the information available in the measurements $\hat{\boldsymbol{u}}$ to reduce the dimension of the control space.

We define the set of all cells that are surely fluid, ${\mathcal{T}_h}^f$, and we set $\sigma_h \coloneqq 0$ in these cells and keep this during the optimization. These cells are those in which we have the measurements available, and the measured velocity is high enough:

$${\mathcal{T}_h}^f \coloneqq \left\{T \in \mathcal{T}_h \mid T \in \Omega_{\text{meas}} \text{ and } \left\|\tilde{\boldsymbol{u}}\left(\boldsymbol{x}_{B(T)}\right)\right\| \geq \epsilon_f\right\}, \tag{5.16}$$

where $\boldsymbol{x}_{B(T)}$ is the barycenter of T, and $\|\cdot\|$ denotes the Euclidean norm. As for the parameter $\epsilon_f > 0$ [m/s], we want it to be as small as the expected amount of noise in the data allows: If ϵ_f is too small, solid cells with highly noisy data may be wrongly classified as fluid.

The corresponding reduced control space is

$$\widetilde{\Sigma}_h \coloneqq \left\{\sigma_h \in \mathbb{P}_0(\mathcal{T}_h) \mid 0 \leq \sigma_h \leq \sigma_{\max}, \text{ and } \sigma_h|_T = 0 \text{ if } T \in {\mathcal{T}_h}^f\right\}. \tag{5.17}$$

Remark 5.4. *As previously observed (Remark 5.1), it is possible to have two different meshes: one for the finite element solution and one for the shape optimization (for σ_h). In this case, the pre-processing uses the mesh for σ_h.*

Remark 5.5. *In [2], the authors reduced the control space by considering only the part of their (two-dimensional) domain where they expected the boundary to be. Moreover, their data were available in the whole of this part.*

On the contrary, our approach is more general: We reduce the control space solely based on the data and do not assume any a priory information on the sought boundary or the domain geometry. In particular, we do not need to assume that the data are available in the parts of the domain where the boundary is located (see, e.g., Figure 5, page 21). As shown in the numerical tests in Sections 7–9, our algorithm produces a satisfactory solution also in this situation.

Remark 5.6. *The concept of pre-processing can be easily extended by replacing the pointwise evaluation $\left\|\tilde{\boldsymbol{u}}(\boldsymbol{x}_{B(T)})\right\|$ in the definition* (5.16) *with, e.g., the L^p norm on the cell or by taking into account the values in neighboring cells.*

Remark 5.7. *The pre-processing could also be formally included in the derivation of the adjoint problem and the optimality condition. One would then use arguments analogous to those described in Section 5.3 and multiply the resistance in the Brinkman problem by the characteristic function of the mesh $\mathcal{T}_h \backslash {\mathcal{T}_h}^f$.*

6 Optimization algorithm

6.1 Gradient-based optimization

For given measurements $\hat{\boldsymbol{u}} \in L^2(\Omega_{\mathrm{meas}})$, the goal is to minimize the cost functional

$$F_h(\sigma_h) = J(\boldsymbol{u}_h, \sigma_h) = \frac{1}{2}\|\boldsymbol{u}_h - \hat{\boldsymbol{u}}\|^2_{0,\Omega_{\mathrm{meas}}} + \frac{\alpha}{2}\|\sigma_h\|^2_0, \tag{6.1}$$

where $(\boldsymbol{u}_h, p_h) = \tau_h(\sigma_h)$ is the finite element solution of the system (5.1) for a given σ_h.

We solve this problem iteratively using a gradient descent scheme. The update of the control σ_h has the form

$$\sigma_h^{n+1} = P_{\widetilde{\Sigma}_h}\left(\sigma_h^n - \gamma\delta(\sigma_h^n)\right), \quad n = 1, 2, \ldots, \tag{6.2}$$

where $\gamma > 0$ [Pa$^2 \cdot$ s^4/m^6] is the step size, $-\delta(\sigma_h^n)$ is a descent direction, and $P_{\widetilde{\Sigma}}$ denotes the projection onto the reduced set of admissible controls $\widetilde{\Sigma}_h$. This $P_{\widetilde{\Sigma}}$ can be realized as the pointwise projection onto the interval $[0, \sigma_{\max}]$.

The descent direction follows from (5.14):

$$-\delta(\sigma_h^n) \coloneqq \boldsymbol{u}_h^n \cdot \boldsymbol{w}_h^n + \alpha\sigma_h^n + \alpha_{\mathrm{GLS}}\,\eta_h^n, \tag{6.3}$$

where η_h^n is defined in (5.15). It depends on the forward solution $(\boldsymbol{u}_h^n, p_h^n)$ of (5.1) and on $(\boldsymbol{w}_h^n, t_h^n)$, the solution to the adjoint problem (5.12). Using equation (5.14), the definition (6.3) guarantees that

$$-F_h'(\sigma_h^n)\delta(\sigma_h^n) \leq 0\,,$$

i.e., $-\delta(\sigma_h^n)$ is really a descent direction.

Since $\sigma_h^n \in \widetilde{\Sigma}_h \subset \mathbb{P}_0(\mathcal{T}_h)$, the update (6.2) requires the projection of (6.3) onto $\mathbb{P}_0(\mathcal{T}_h)$. We carry out this projection as follows:

Let $\{\phi_j\}_{j=1}^{N_0}$ be an orthogonal basis of $\mathbb{P}_0(\mathcal{T}_h)$ such that $\phi_j = \frac{1}{|T_j|}$ for each $T_j \in \mathcal{T}_h$. We then define $\sigma_{h,j}^n \coloneqq (\sigma_h^n, \phi_j)$ and compute the finite element projection of the update as

$$-\delta(\sigma_h^n)_j \coloneqq -(\delta(\sigma_h^n), \phi_j) = (\boldsymbol{u}_h^n \cdot \boldsymbol{w}_h^n, \phi_j) + \alpha\sigma_{h,j}^n + \alpha_{\mathrm{GLS}}(\eta_h^n, \phi_j), \tag{6.4}$$

where the additional contribution due to the stabilization reads

$$(\eta_h^n, \phi_j) = \frac{h_{T_j}^2}{\mu_{\mathrm{eff}}|T_j|}\int_{T_j}\left((-\boldsymbol{f} + 2\sigma_h^n\boldsymbol{u}_h^n + \nabla p_h^n)\cdot\boldsymbol{w}_h^n + \boldsymbol{u}_h^n \cdot \nabla t_h^n\right).$$

The iterative optimization algorithm is summarized in Algorithm 1.

Algorithm 1: Optimization algorithm

Data: Set $n = 0$ and choose initial control $\sigma_h^0 \in \Sigma_h$, α in (4.1), the step size γ, the parameter ε_f in (5.16), and a stopping tolerance $\varepsilon_{\text{stop}} > 0$.

begin

0 Determine ${\mathcal{T}_h}^f$ and $\widetilde{\Sigma}_h$ defined by (5.16) and (5.17) and modify σ_h^0 accordingly.

repeat

1 Increase n by one and solve equation (5.1) to get $(\boldsymbol{u}_h^n, p_h^n)$.

2 Calculate $F_h^n := F_h(\sigma_h^n) = J((\boldsymbol{u}_h^n, p_h^n), \sigma_h^n) = \frac{1}{2}\|\boldsymbol{u}_h^n - \hat{\boldsymbol{u}}\|_0^2 + \frac{\alpha}{2}\|\sigma_h^n\|_0^2$.

3 Solve the adjoint problem (5.11) to compute $(\boldsymbol{w}_h^n, t_h^n)$.

4 Calculate the descent direction $-\delta(\sigma_h^n)$ using formula (6.4).

5 Calculate the new control σ_h^{n+1} via (6.2).

until $\left| \dfrac{\|\boldsymbol{u}_h^{n+1} - \hat{\boldsymbol{u}}\|_0^2 - \|\boldsymbol{u}_h^n - \hat{\boldsymbol{u}}\|_0^2}{\|\boldsymbol{u}_h^{n+1} - \hat{\boldsymbol{u}}\|_0^2} \right| < \varepsilon_{\text{stop}}$;

end

6.2 Post-processing

As a next step, we further restrict the space of admissible controls by allowing σ_h to have only two possible values and classifying the cells as *fluid* ($\sigma_h = 0$) or *solid* ($\sigma_h = \sigma_{\max}$). This is then followed by two more steps. Hence, we carry out these three post-processing steps:

(i) thresholding, i.e., setting

$$\sigma_h(T) := \begin{cases} 0, & \sigma_h(T) < \sigma_{\text{th}}, \\ \sigma_{\max}, & \text{otherwise} \end{cases} \tag{6.5}$$

for a given threshold $\sigma_{\text{th}} > 0$ [Pa · s/m^2];

(ii) gap filling (Section 6.2.1);

(iii) cluster filtering (Section 6.2.2).

6.2.1 Gap filling

The purpose of this step is to correct the classification of a) isolated fluid cells that are enclosed in solid regions and b) thin layers of fluid cells between the wall Γ_{wall} and solid regions. We call these cells *gaps*. Since the obstruction $\Omega \setminus \Omega_{\text{fluid}}$ stands for accumulated material attached to Γ_{wall}, ideally, none of the above cells should exist.

However, it can happen in the computations that the magnitude of the velocity data in these locations is very small, e.g., due to noise or numerical artifacts. Consequently, these cells remain *invisible* to Algorithm 1 and end up wrongly marked as fluid.

We consider two types of gaps:

- Type 1: A cell $T \in \mathcal{T}_h$ is called a gap of *type 1* if $\sigma_h(T) = 0$ and $\sigma_h(\tilde{T}) = \sigma_{\text{max}}$ for all cells $\tilde{T}$ sharing an edge (in 2D) / a face (in 3D) with T, see Figure 2a, page 17.
- Type 2: The following description is illustrated in Figure 2b, page 17. Let T be an internal cell with the diameter h_T such that $\sigma_h(T) = 0$, and let L_g be a given positive constant.

  We consider (in 2D) a square with the same centroid as T and its edges aligned with the coordinate axes. Further, we consider the lines going through the opposite vertices or the midpoints of the opposite edges of this square. Finally, we consider the line segments that lie in these lines and are of the length $L_g h_T$, and have the above centroid as its midpoint.

  Then we say that T is a gap of *type 2* if at least one of these line segments has both end points in solid cells.

  For a boundary cell T, we choose a positive constant $L_{g,\partial\Omega}$ and call T a gap of type 2 if at least one line segment starting from the centroid of T and directed as the inward normal of a boundary face of T ends in a solid cell.

  The extension of this concept to 3D is straightforward: The above auxiliary square is replaced by a cube, and the above auxiliary lines go through the midpoints of the opposite faces as well.

We use several values of L_g and $L_{g,\partial\Omega}$ when checking whether a given cell is a gap. These values are the same for each cell. The gap filling can be written as Algorithm 2.

The result of this algorithm is very sensitive to the choice of L_g and $L_{g,\partial\Omega}$: Too large values may yield the (erroneous) filling of large fluid areas, especially in the thinnest parts of the fluid domain. We choose these values experimentally as $L_g = \frac{1}{8}, \frac{1}{4}$ and $L_{g,\partial\Omega} = \frac{1}{4}, \frac{1}{2}, \frac{3}{4}$. These choices should assure that when checking whether a cell is a fluid gap, we consider only the two closest layers of cells.

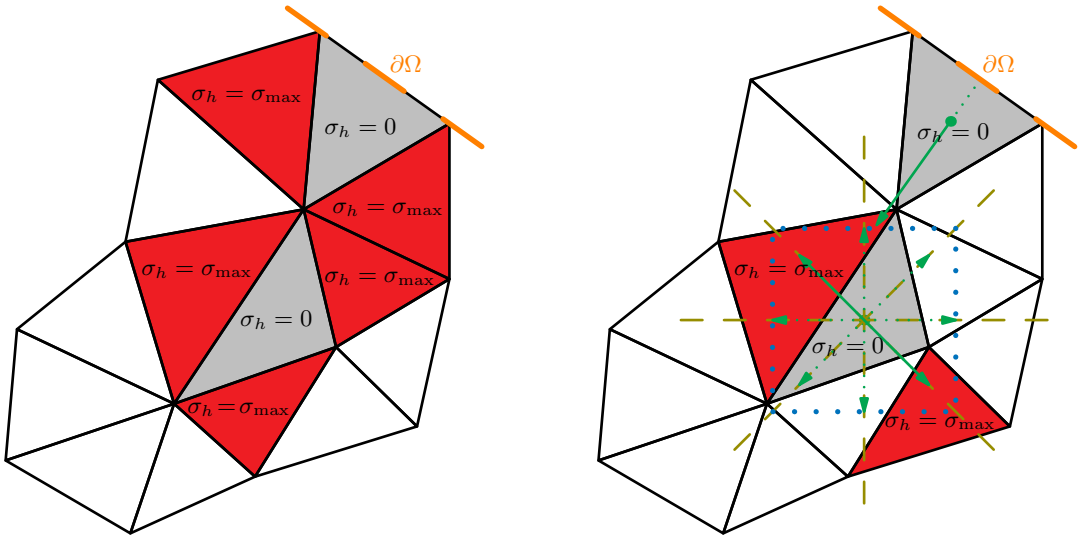
Algorithm 2: Gap filling algorithm

Data: σ_h that takes only two values (0 or $\sigma_{\max}$), and sets of values for the parameters L_g and $L_{g,\partial\Omega}$.

```
repeat
    for all T ∈ T_h such that σ_h(T) = 0 do
        for all values of L_g and L_{g,∂Ω} do
            if T is a gap then
                σ_h(T) := σ_max
            end
        end
    end
until no changes in σ_h have been made;
```

6.2.2 Cluster filtering

Algorithm 1 gives a function σ_h that gradually varies from 0 to a positive value that is typically far below the value of an almost impermeable region. Therefore, we modify this σ_h by the thresholding (6.5), i.e.,



(a) Gaps of type 1 indicated by gray color.

(b) Gaps of type 2 indicated by gray color.

Figure 2: Two types of gaps. In 2b, we also indicate the auxiliary square (dotted blue line), auxiliary lines (dashed olive lines) and auxiliary line segments (green) corresponding to the lower gray cell.

we remove the intermediate values of σ_h by classifying the cells into two categories. Afterward, we employ the gap filling to correct the marks of isolated fluid cells or thin clusters of fluid cells enclosed in solid regions.

However, there may still remain some isolated cells or tiny isolated clusters of cells wrongly identified as solid or fluid. To remove these clusters, we introduce the cluster filtering:

(i) We group all solid and fluid cells into disjoint *clusters* of the maximal size possible, where two cells belong to the same cluster if they are both fluid or both solid and have a common face (in 3D) or edge (in 2D).

(ii) We remove all clusters with a volume smaller than a given threshold by marking them as fluid (in the case of a solid cluster) or solid (in the case of a fluid cluster).

For the numerical examples shown in this paper, we considered a threshold equal to 1% of the volume of the largest solid cluster.

One may think that in practice, the gap filling affects mostly the cells that the cluster filtering affects as well, thus rendering the gap filling unnecessary. However, as explained at the end of Section 8.1, this is not the case.

6.3 Complete algorithm for shape estimation

The algorithm for the estimation of Ω_{fluid} can be summarized as follows:

1 Compute the resistance σ_h that minimizes the functional (6.1) using the gradient-based iterative optimization algorithm, Algorithm 1.

2 Carry out the post-processing steps i–iii from Section 6.2 (in this order).

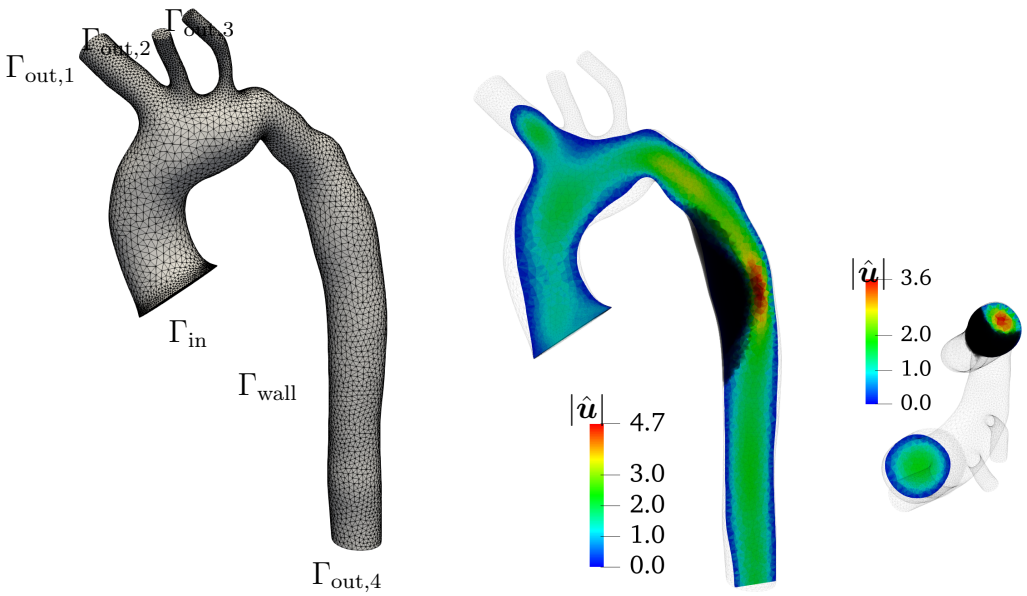


Figure 3: Left: View of the surface mesh of the considered aortic geometry with the corresponding decomposition of the boundary. The diameter of the inlet boundary is around 2.5 cm. The height of the domain is around 19 cm. Right: Front and bottom views of the artificial obstruction $\Omega \backslash \Omega_{\text{flow}}$ (black region), together with the corresponding fluid velocity field used as $\hat{\boldsymbol{u}}$ [m/s] on two slices orthogonal to the view.

Note that one iteration of Algorithm 1 is several times more expensive than the whole post-processing. But since Algorithm 1 needs to perform at least hundreds of iterations (see Figure 6, page 22), the computational costs for step 2 are negligible in comparison with step 1.

7 Numerical simulations

7.1 Model setup

Geometry We consider various scenarios of a flow in an aortic geometry (Figure 3, left) in which a portion of the original fluid domain was artificially marked as a solid boundary. Hence, the original geometry stands for Ω, the new domain is Ω_{flow}, and the internal solid domain is $\Omega \backslash \Omega_{\text{flow}}$ (see Figure 3, center and right).

The computational mesh was segmented from medical imaging, and it was provided by the research group of Prof. Leonid Goubergrits (Institute of Computer-assisted Cardiovascular Medicine, Charité University Hospital, Berlin).

Parameters for the forward problem In (2.1), we set $\mu_{\text{eff}} = 3.5 \cdot 10^{-6}$ Pa · s (equal to the viscosity of blood), $\boldsymbol{f} = \boldsymbol{0}$, and $g = 0$. The inlet velocity profile $\boldsymbol{u}_{\text{in}}$ is defined as the interpolation of MRI measurements onto the piecewise linear finite element space. The Neumann boundary conditions on the outlets $\Gamma_{\text{out},1}, \ldots, \Gamma_{\text{out},4}$ are defined as $g_{\boldsymbol{n},1} = 0.1\boldsymbol{n}$ Pa and $g_{\boldsymbol{n},i} = \boldsymbol{0}$ for $i = 2, 3, 4$. Finally, we always use the value $\alpha_{\text{GLS}} = 10^{-2}$ in (5.2).

The definitions of Ω_{fluid} and Ω_{meas} are scenario-dependent: As for the scenarios in Sections 8.1–8.5 and 9.1, we use Ω_{fluid} defined as in Figure 5, page 21. The region Ω_{meas} is either equal to Ω (in Sections 8.1–8.2), or defined as in Figure 5 (in Sections 8.3–8.5 and 9.1). In the scenario in Section 9.2, Ω_{fluid} and Ω_{meas} are defined as in Figure 17, page 35.

The domain Ω is discretized using a mesh $\mathcal{T}_h$ of 106,097 tetrahedra (21,324 DOF for p_h and for each component of $\boldsymbol{u}_h$) in Figure 3 whose cell diameters vary in the range $[0.7,\ 5.0]$ mm. We use this mesh for all parts of the complete algorithm in Section 6.3.

Parameters for Algorithm 1 We always set $\sigma_h^0 = 0$, $\epsilon_f = 1$ m/s in (5.16), $\alpha = 10^{-5}\ \mathrm{m}^6 \cdot \mathrm{s}^{-4} \cdot \mathrm{Pa}^{-2}$ in (4.1), $\gamma = 10^8\ \mathrm{Pa}^2 \cdot \mathrm{s}^4/\mathrm{m}^6$ in (6.2), and $\varepsilon_{\mathsf{stop}} = 10^{-4}$.

Parameters for the post-processing We always set $\sigma_{\mathrm{th}} = 2\ \mathrm{Pa} \cdot \mathrm{s}/\mathrm{m}^2$ (this choice is explained in Section 7.6), $\sigma_{\max} = 10^4\ \mathrm{Pa} \cdot \mathrm{s}/\mathrm{m}^2$ in (6.5), $L_g = \frac{1}{8},\ \frac{1}{4}$, and $L_{g,\partial\Omega} = \frac{1}{4},\ \frac{1}{2},\ \frac{3}{4}$ in Algorithm 2.

Hardware and software parameters We carried out the simulations by employing our in-house code ParMooN [**?**] on the computer HPE Synergy 660 Gen10 with 2 Xeon eighteen-core processors, 3000 MHz, and 768 GB RAM. We used 32 MPI processes, and one step of Algorithm 1 took approximately 6 s.

7.2 Synthetic velocity measurements

We applied our optimization framework to synthetic velocity data $\hat{\boldsymbol{u}}$, i.e., to data based on results of numerical simulations. We generated these data as follows:

1 (Reference solution) Define a binary function σ_{true} equal to 0 in the fluid region Ω_{flow} and equal to $\sigma_{\max}$ in the solid region $\Omega \backslash \Omega_{\mathrm{flow}}$, see Figure 3, right.

2 (Perfect measurements) Solve the flow problem (5.1) with $\sigma_h = \sigma_{\mathsf{true}}$.

And only for the noisy measurements:

3 (Noisy measurements) Interpolate the numerical solution onto $\mathbb{P}_0(\mathcal{T}_h)$ and then to each component, add a Gaussian noise field with the mean 0 and the standard deviation $1/3$. This deviation is about 8% of the maximum magnitude of the initial velocity, which is approximately 4 m/s.

4 (Interpolation) Interpolate the resulting (piecewise-constant) velocity field onto $\boldsymbol{W}_h$ (the velocity finite element space). Figure 4 shows this field.

Remark 7.1. *The reference solution used for the numerical study was generated with $\sigma_{\max} = 10^4$. We also performed a set of experiments using $\sigma_{\max} = 10^8$, which yielded negligible differences.*

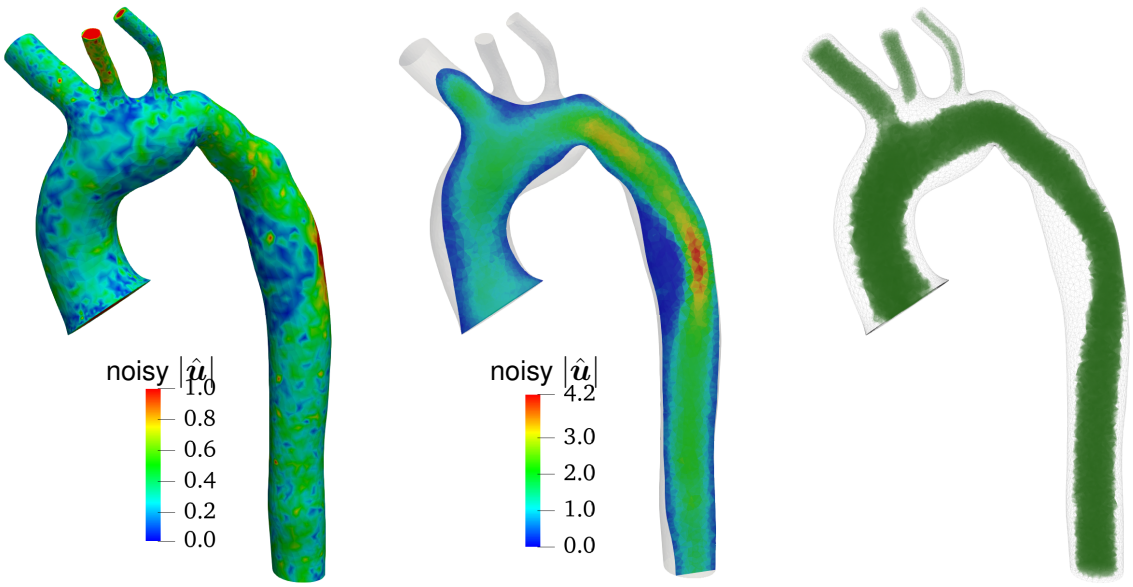


Figure 4: Left: Magnitude of the synthetic velocity data affected by noise [m/s] on the domain boundary. Notice that the color range is rescaled between 0 and 1 m/s, while the actual maximum magnitude of the noisy data over the whole domain is about 4.4 m/s. Center: Cross-sectional view of the noisy velocity data. Right: Cells identified as fluid and excluded from the optimization (in green) starting from noisy synthetic data.

7.3 Estimation of boundary shape

Notice that the goal of the post-processing is no longer to reduce the cost functional F_h. Its purpose is to classify the cells into two subdomains — solid and fluid — and, at the same time, to reduce the effect of numerical artifacts. Therefore, to evaluate the performance of the whole algorithm in terms of the shape approximation, we consider two regions:

$$\Omega_{\mathrm{F}} := \{T \in \mathcal{T}_h \,|\, \sigma_h(T) = 0\} \quad \text{and} \quad \Omega_{\mathrm{S}} := \{T \in \mathcal{T}_h \,|\, \sigma_h(T) > 0\}.$$

If the exact shape of Ω_{flow} is known (as in our examples with synthetic data), we can define a shape approximation error as

$$\mathcal{E} := \underbrace{|\Omega_{\mathrm{flow}} \cap \Omega_{\mathrm{S}}|}_{=:\mathcal{E}_S} + \underbrace{|(\Omega \setminus \Omega_{\mathrm{flow}}) \cap \Omega_{\mathrm{F}}|}_{=:\mathcal{E}_F}, \tag{7.1}$$

i.e, $\mathcal{E}_S$ denotes the volume of the cells that are wrongly identified as solid (but should be fluid), and $\mathcal{E}_F$ stands for the volume of the cells that belong to the solid domain but were marked as fluid.

Observe that the error (7.1) can be evaluated only in a controlled setting where the reference solution is known, as in the cases considered in this paper. A possible alternative is to use, as a posteriori criteria, the error with respect to the available velocity data (see also Section 7.6).

7.4 Overview of scenarios

We test the complete algorithm for the shape estimation described in Section 6 on several scenarios that differ in the availability and quality of the measurements $\hat{\boldsymbol{u}}$. We consider the following cases that are based on the geometry in Figure 3, page 18:

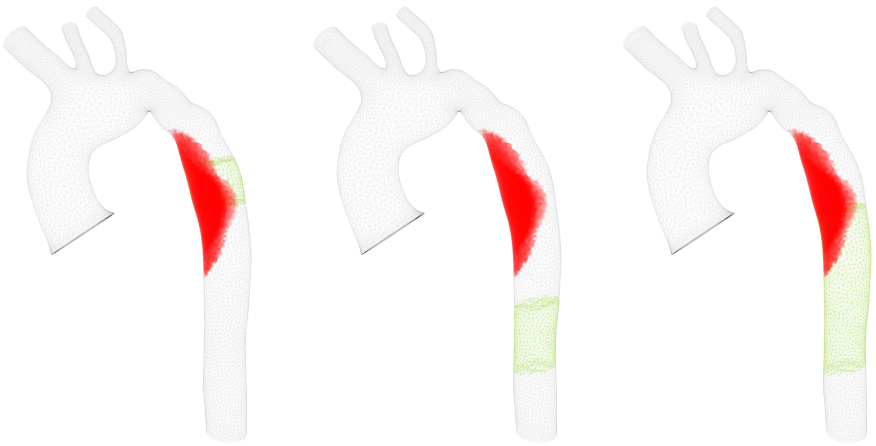


Figure 5: Our scenarios with partial measurements. Left: Case (partial-inside). Center: Case (partial-outside). Right: Case (partial-large). The red area stands for the obstruction $\Omega \backslash \Omega_{\mathrm{flow}}$ (which is the same in all scenarios). The velocity data $\hat{\boldsymbol{u}}$ are unavailable in the green regions. The regions in the left and center cases have similar volumes.

- *full + noise-free* (Section 8.1): Noise-free measurements available on the whole domain.
- *full + noise* (Section 8.2): Noisy measurements available on the whole domain.
- *partial-inside + noise* (Sections 8.3 and 9.1): Noisy measurements that are not available in a region containing a part of the internal boundary. In Section 9.1 a finer mesh is used for the velocity and the pressure.
- *partial-outside + noise* (Section 8.4): Noisy measurements that are not available in a region that does not intersect the internal boundary.
- *partial-large + noise* (Section 8.5): Noisy measurements that are not available in a large downstream region covering a part of the internal solid boundary.

Figure 5 shows the above regions where data are not available, together with the location of the internal boundary that we estimate.

7.5 Summary of main results

In this section, we summarize the overall qualitative and quantitative results of Algorithm 1 and the post-processing in the above cases. Further aspects of the respective scenarios are discussed in the subsequent sections.

7.5.1 Algorithm 1

First, Figure 4 (right), page 20, shows the results of the pre-processing in the scenario *full + noise*: Almost half of the cells were identified as fluid and excluded from the optimization. In the other scenarios, the results were similar, except that no cells were identified as fluid in the regions of missing data.

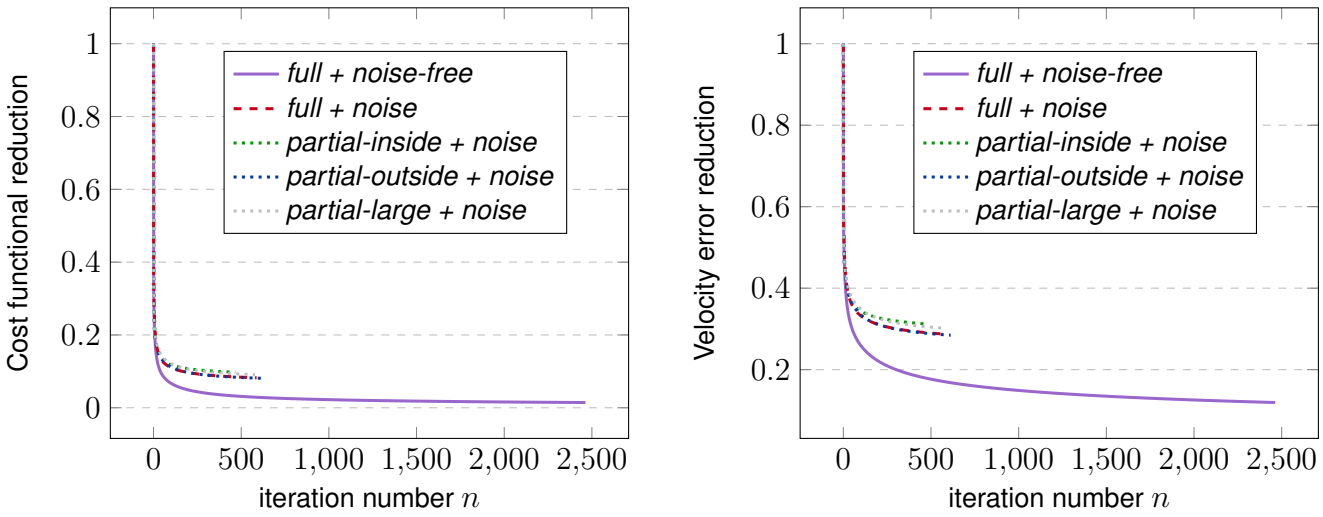


Figure 6: Left. Reduction of the cost functional value (computed as F_h^n/F_h^0) as a function of the iteration number n. Right: Reduction of the $\boldsymbol{L}^2$-error with respect to the data, computed as $\frac{\|\boldsymbol{u}_h^n - \hat{\boldsymbol{u}}\|^2_{0,\Omega_{\mathrm{meas}}}}{\|\boldsymbol{u}_h^0 - \hat{\boldsymbol{u}}\|^2_{0,\Omega_{\mathrm{meas}}}}$, as a function of the iteration number n.

The main results are depicted in Figure 6 (the reduction of the cost functional over the optimization loop):

Figure 6 (left) shows that the algorithm considerably reduced the cost functional in the noise-free setup when the measurements were available over the whole domain: The final value of the cost functional was about 1% of the initial value. Adding noise to the measurements lowered the performance, the final value being about 8% of the initial value. The scenarios with partially available measurements yielded similar overall performances.

Figure 6 (right) shows the difference between the measurements and the simulated velocity at each optimization iteration. In the noise-free scenario, with the data available on the whole domain, the error was reduced by about 80% in less than 500 iterations. Afterward, the decrease continued, and the algorithm stopped after about 2,500 iterations. Although this scenario is unrealistic in practice, we used it to validate the algorithm in the most favorable situation and to obtain reference results for more realistic cases. Considering noisy data available over the whole domain resulted in an error reduction of around 70%. When the data were also only partially available, the results worsened slightly if $\Omega \setminus \Omega_{\mathrm{meas}}$ contained the internal boundary (the cases *partial-inside + noise* and *partial-large + noise*).

Although this work focuses on boundary reconstruction based on velocity measurements, the pressure field is also a quantity of interest in several applications (e.g., medical diagnosis). Therefore, Figure 7 provides an overview of the pressure fields corresponding to the optimal resistance σ_h, together with the true pressure field $\hat{p}$ corresponding to the noise-free data $\hat{\boldsymbol{u}}$. In particular, it shows the average pressures on selected planes orthogonal to the centerline of the domain Ω.

Outside the obstructed region, which is located between planes 4 and 6, the pressure drop behaves similarly in all scenarios, and we obtained a good agreement with the reference pressure. This is because we imposed the same Neumann boundary conditions on the outlet boundaries. The main differences can be observed on plane 5 for the scenario *partial-inside + noise* (the average pressure is lower than $\hat{p}$) and on plane 6 for the scenarios *partial-outside + noise* and *partial-large + noise* (the

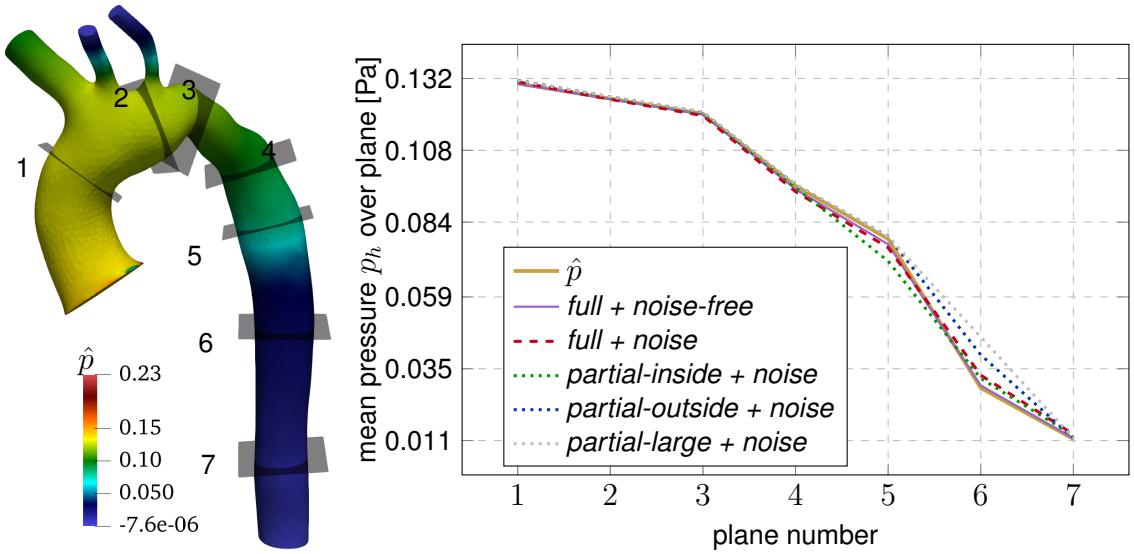


Figure 7: Left. Pressure field $\hat{p}$ corresponding to the noise-free data $\hat{\boldsymbol{u}}$ on the domain boundary together with the positions of the reference planes. Right. Average pressures over the reference planes after the optimization loop and $\hat{p}$. The obstruction is located approximately between planes 4 and 6.

average pressure is higher than $\hat{p}$).

This indicates that the mismatch in the pressure is due to the unavailability of the data $\hat{\boldsymbol{u}}$. In fact, Algorithm 1 tends to incorrectly increase σ_h in the flow regions where the data are not available. This led to an increase in the pressure p_h on plane 6 in the last two scenarios. However, it did not lead to an increase in p_h on plane 5 in the scenario *partial-inside + noise* because the algorithm produced a lower σ_h in the obstructed area lying in the region of the missing data. This yielded a lower average pressure in this part of Ω.

7.5.2 Post-processing

Table 1 overviews the results of the post-processing applied to the results of Algorithm 1: In the noise-free situation, the algorithm correctly recognized 99% of the solid cells (the fourth column of Table 1). At the same time, a volume of about 2.4 cm^3 (circa 35% of the obstruction volume) was erroneously marked as solid.

Considering the noisy measurements, we still observe good accuracy in the marking of solid cells (more than 90%). The volume of the fluid cells wrongly marked as solid also remains comparable with the scenario without noise.

Note that the fluid cells wrongly marked as solid always correspond to up to one layer of cells along $\partial\Omega_{\text{fluid}}$ (see, e.g., Figure 12, page 29). This also explains the fact that this error is similar in all scenarios. In the situation with a large portion of missing measurements, the accuracy in the marking of solid cells decreased (to merely 79%). But at the same time, the volume of the cells erroneously marked as solid reduced.

Also note that both the gap filling and cluster filtering can systematically reduce the error, regardless of the considered scenario. They improve the robustness of the procedure regardless of the presence of noise, and their effect even increases when the quality and availability of data deteriorates. This, together with the pictures of the estimated Ω_{fluid} in the following sections, means that each post-

		$\mathcal{E}_S$	$\mathcal{E}_F$	$\mathcal{E}$
	Initial errors (m^3)	0	6.895e-6 (100%)	6.895e-6 (100%)
full + noise-free	Thresholding	1.740e-6	1.698e-6 (24.8%)	3.438e-6 (49.9%)
	Gap filling	2.393e-6	8.784e-8 (1.3%)	2.481e-6 (36.0%)
	Cluster filtering	2.398e-6	1.133e-8 (0.2%)	2.409e-6 (34.9%)
full + noise	Thresholding	1.957e-6	2.462e-6 (35.7%)	4.419e-6 (64.1%)
	Gap filling	2.732e-6	4.809e-7 (7.0%)	3.212e-6 (46.6%)
	Cluster filtering	2.424e-6	4.177e-7 (6.1%)	2.842e-6 (41.2%)
partial-inside + noise	Thresholding	1.884e-6	3.004e-6 (43.6%)	4.888e-6 (70.9%)
	Gap filling	2.591e-6	7.962e-7 (11.5%)	3.387e-6 (49.1%)
	Cluster filtering	2.463e-6	6.401e-7 (9.3%)	3.104e-6 (45.0%)
partial-outside + noise	Thresholding	1.620e-6	2.528e-6 (36.7%)	4.148e-6 (60.2%)
	Gap filling	2.185e-6	5.413e-7 (7.9%)	2.726e-6 (39.5%)
	Cluster filtering	2.156e-6	4.652e-7 (6.7%)	2.621e-6 (38.0%)
partial-large + noise	Thresholding	4.247e-6	3.211e-6 (46.6%)	4.247e-6 (61.6%)
	Gap filling	1.410e-6	1.522e-6 (22.1%)	2.931e-6 (42.5%)
	Cluster filtering	1.393e-6	1.449e-6 (21.0%)	2.842e-6 (41.2%)

Table 1: Errors $\mathcal{E}_F$, $\mathcal{E}_S$, $\mathcal{E}$ (the total volumes of cells wrongly marked as fluid and solid, respectively, and $\mathcal{E} = \mathcal{E}_F + \mathcal{E}_S$) after each post-processing step. The percentage values indicate how large the errors are relative to the initial error.

processing step plays an important role in improving the quality of the solution: The thresholding alone yields, in general, larger errors in the identification of solid cells, especially in the case of noisy measurements.

7.6 Effect of choice of threshold σ_{th}

One of the outcomes of our simulations is that the final result strongly depends on the value of the threshold σ_{th} in the thresholding. The purpose of this section is to shed some light on the problem of how to choose σ_{th} in practice.

7.6.1 Scenario *partial-inside + noise*

First, we investigate the sensitivity of the algorithm with respect to σ_{th} for the scenario *partial-inside + noise*: Figure 8 (page 25) shows the estimated obstruction $\Omega \setminus \Omega_{\text{fluid}}$ for $\sigma_{\text{th}} = 1, 2, \ldots, 5$, together with the true solution.

Note that the above values of σ_{th} are far below the values of σ_h corresponding to a nearly impermeable region ($\sigma_{\max} = 10^4$) that we used for the generation of the synthetic data $\hat{\boldsymbol{u}}$. This is because Algorithm 1 always produces an optimal σ_h the maximum of which is far below $\sigma_{\max}$; the velocity field is insensitive to any further increase in σ_h (see Figure 11, page 28). This behavior has been also reported in recent related problems (see [2]), and it motivates the use of the post-processing.

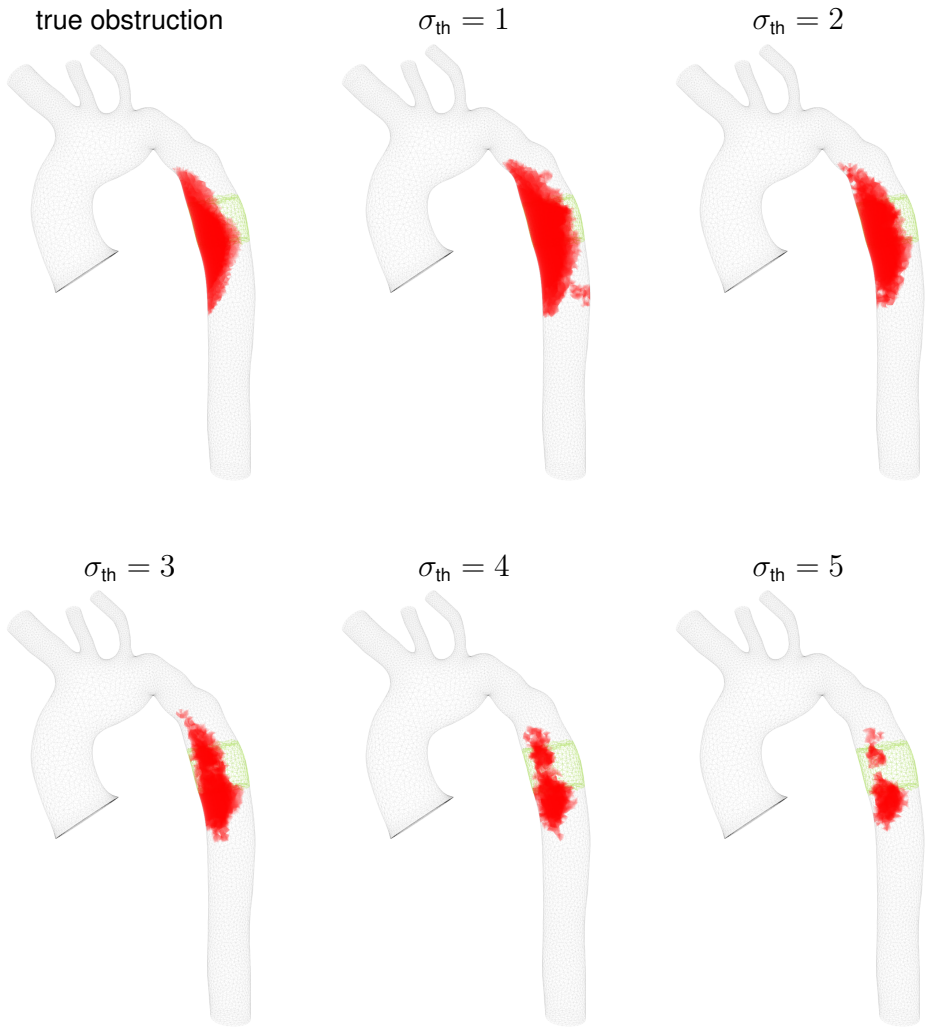


Figure 8: Scenario *partial-inside + noise*: Estimated shape of the obstruction $\Omega \setminus \Omega_{\text{fluid}}$ after the cluster filtering for various thresholds σ_{th}. Small thresholds incorrectly enlarge the obstruction, but this obstruction resembles the true one more.

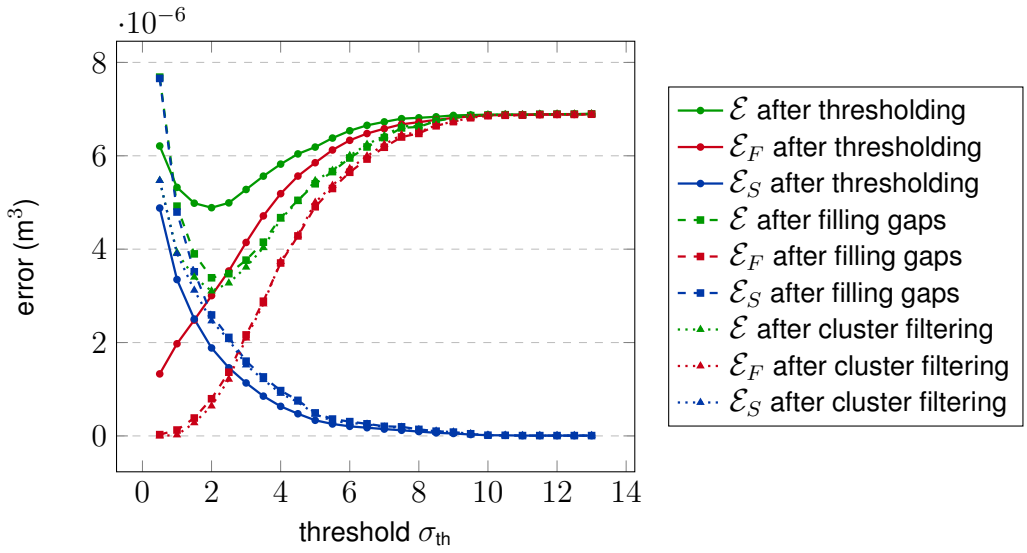


Figure 9: Scenario *partial-inside + noise*: Errors $\mathcal{E}_F$, $\mathcal{E}_S$, $\mathcal{E}$ (the total volumes of cells wrongly marked as fluid and solid, respectively, and $\mathcal{E} = \mathcal{E}_F + \mathcal{E}_S$) after each post-processing step for the thresholds $\sigma_{\text{th}} = 0.5,\ 1.0,\ 1.5,\ \ldots,\ 13$. These thresholds cover the range of the optimal σ_h computed by Algorithm 1. In each curve, the colors refer to the individual errors, while the markers (circles, squares, and triangles) denote the individual post-processing steps.

Further, Figure 9 shows how both $\mathcal{E}_F$ and $\mathcal{E}_S$ (the total volumes of the cells incorrectly marked as fluid and solid, respectively) depend on σ_{th}: As expected, $\mathcal{E}_S$ is a decreasing function of σ_{th} that is zero after a certain σ_{th} (as σ_{th} increases, less cells with positive σ_h values after the optimization are marked as solid). Vice versa, $\mathcal{E}_F$ is increasing in σ_{th}.

Moreover, the shape approximation error $\mathcal{E} = \mathcal{E}_F + \mathcal{E}_S$ has a minimum that is close to $\sigma_{\text{th}} = 2$. For this value, the error $\mathcal{E}$ improved in the last two post-processing steps the most (cf. the green lines with dots and triangles).

7.6.2 Other scenarios

Next, Figure 10 (top) shows the shape approximation error $\mathcal{E}$ as a function of σ_{th} for all considered cases. This error is clearly sensitive to σ_{th}, and in the considered cases, it always attains its minimum value for σ_{th} approximately in the interval $[1.5, 3.0]$.

Unfortunately, the shape approximation error cannot be computed in general. Thus, it cannot be used as a criterion for the choice of the threshold σ_{th}. In order to have a practical guideline for the threshold selection, we also analyze how the error in the velocity depends on σ_{th}, see Figure 10 (bottom). Namely, for each value of the threshold σ_{th}, we compute the flow solution $\boldsymbol{u}_h$ based on the corresponding estimated solid and fluid domains, together with the difference between this solution and the data $\hat{\boldsymbol{u}}$. Since the velocity measurements generally contain noise, and they are not available over the whole domain Ω, one shall not expect the minima of $\mathcal{E}$ and the velocity error to be attained at the same σ_{th}. However, in the considered cases, the minima of the velocity errors are attained for $\sigma_{\text{th}} \in [4,\ 9.5]$, which is slightly higher than the optimal σ_{th} for $\mathcal{E}$ (between 1.5 and 3.0). Hence, we believe that a good practical choice of σ_{th} is a value slightly lower than the optimal σ_{th} for the velocity error.

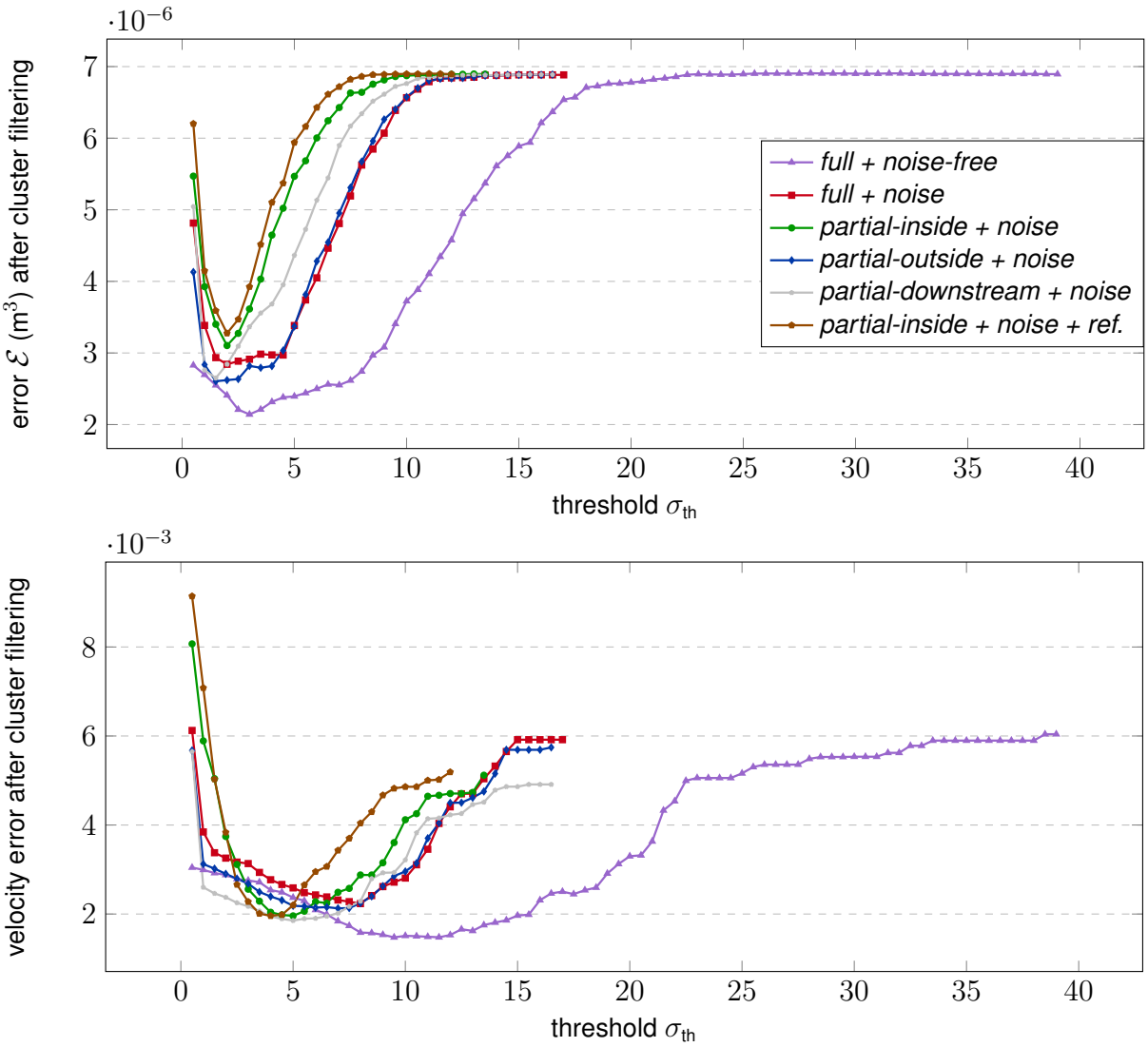


Figure 10: Errors $\mathcal{E}$ (top) and velocity errors (bottom) after the cluster filtering for individual scenarios and the thresholds $\sigma_{\text{th}} = 0.5,\ 1.0,\ 1.5,\ \ldots$ covering the ranges of the optimal σ_h computed by Algorithm 1. The minima of the velocity errors do not correspond to the minima of $\mathcal{E}$. Hence, the problems of minimizing the velocity error and approximating the obstruction are not completely related to each other.

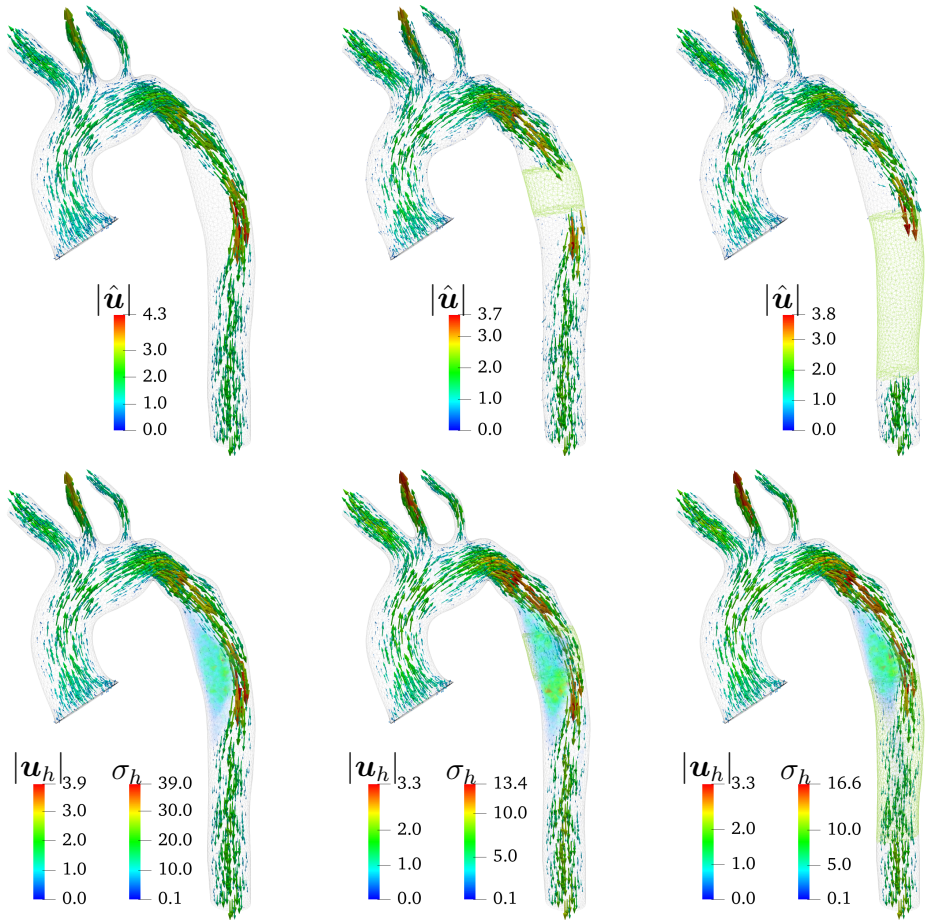


Figure 11: Scenarios *full + noise-free* (left column), *partial-inside + noise* (center column) and *partial-large + noise* (right column). Top row: Velocity data $\hat{\boldsymbol{u}}$ [m/s] in the above scenarios. The green region depicts the area where the data $\hat{\boldsymbol{u}}$ are not available. Bottom row: Velocity field $\boldsymbol{u}_h$ [m/s] obtained by solving the forward problem with the optimal σ_h and this σ_h [Pa · s/m^2].

8 Detailed results in scenarios

8.1 Scenario *full + noise-free*

This is the ideal situation: The data $\hat{\boldsymbol{u}}$ are noise-free and available in the whole domain. Figure 11 (top-left) shows a selected view of the data $\hat{\boldsymbol{u}}$, and Figure 11 (bottom-left) shows the optimal resistance σ_h — prior to any post-processing — and the velocity field $\boldsymbol{u}_h$ obtained by solving the forward problem with the optimal resistance.

The maximum of σ_h in Figure 11 (bottom-left) lies far below the high value ($\sigma_{\text{max}} = 10^4$) used for the generation of the synthetic data; the velocity field is insensitive to any further increase in σ_h. This behavior was observed in all cases considered in this work.

The effect of the post-processing steps is summarized in Figure 12: Classifying the cells as solid or fluid based on a threshold (Figure 12, top-left) located the obstructed area and captured the shape of the internal boundaries very well. The shape approximation error $\mathcal{E}$ reduced by approximately 50% — see Table 1, page 24.

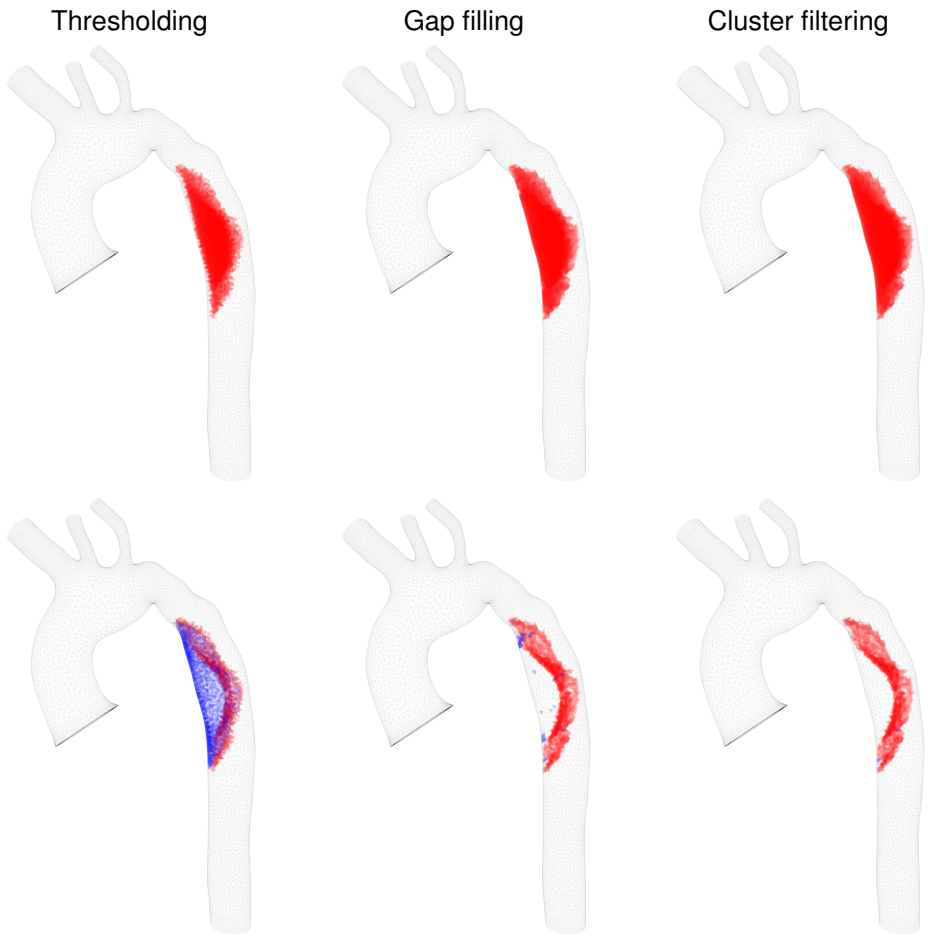


Figure 12: Scenario *full + noise-free*: Results of the post-processing steps. Top row: $\Omega \setminus \Omega_{\text{fluid}}$ (in red) after each post-processing step. Bottom row: Cells incorrectly marked as fluid (in blue) and solid (in red) after each post-processing step.

Figure 12 (bottom-left) shows the cells that the thresholding marked incorrectly. These cells are basically on two types of spots: (i) near $\partial\Omega$, where the cells remain fluid ($\sigma_h = 0$); and (ii) at the solid-fluid interface, where the algorithm tends to incorrectly extend the solid part. The first type of cells is corrected using the gap filling; the results are shown in Figure 12 (center column). This step further decreased $\mathcal{E}$ to 36% of the initial error. Note that the cells in question cannot be corrected via the cluster filtering because they belong to the largest cluster of fluid cells.

Finally, Figure 12 (right column) shows the effect of the cluster filtering that improved the solution again: Some small isolated fluid clusters disappeared. The final $\mathcal{E}$ is about 35% of the initial error, and there are only several cells incorrectly marked as fluid.

8.2 Scenario *full + noise*

With the Gaussian noise included in the data $\hat{\boldsymbol{u}}$, Algorithm 1 meets the convergence criterion after about 500 iterations, yielding a higher cost functional and a worse velocity error than in the previous case without the noise, see Figure 6, page 22. Moreover, the resistance σ_h is generally lower than that obtained in the noise-free scenario, while the peak velocity is slightly lower than in the previous case (the corresponding figure is omitted).

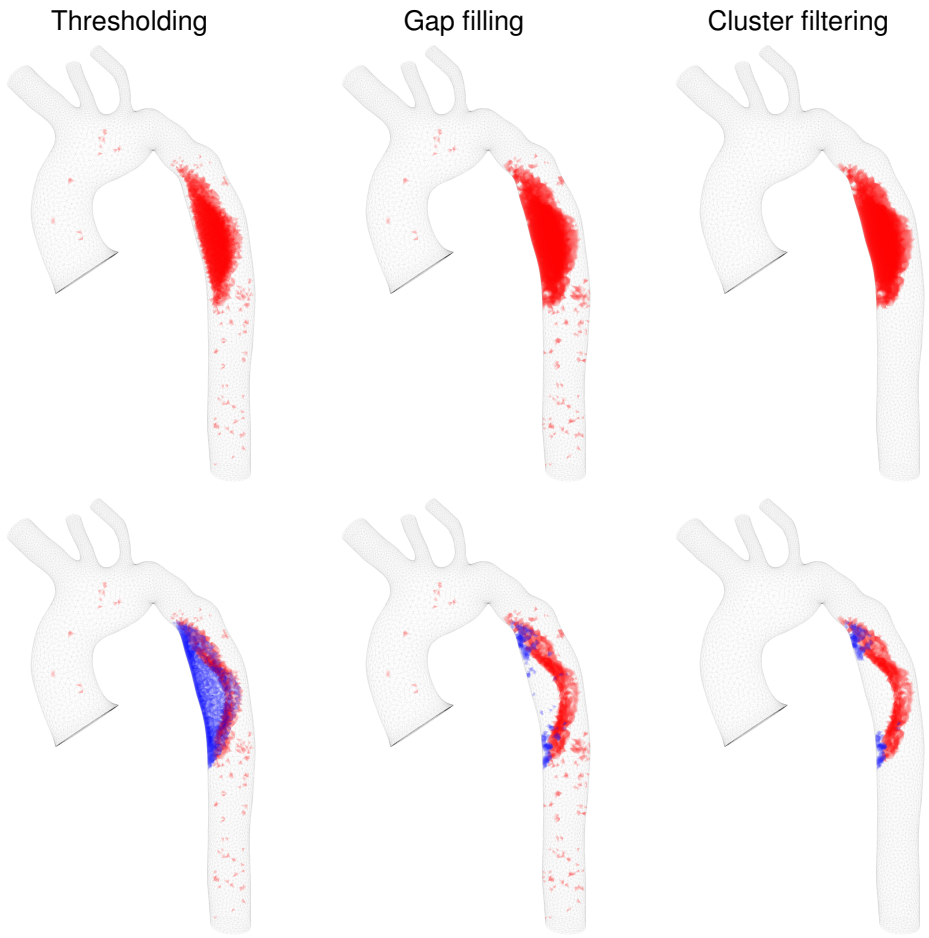


Figure 13: Scenario *full + noise*: Results of the post-processing steps. Top row: $\Omega \setminus \Omega_{\text{fluid}}$ (in red) after each post-processing step. Bottom row: Incorrectly marked fluid (in blue) and solid (in red) cells after each post-processing step.

As for the approximation of the internal boundary, the thresholding reduced the shape approximation error $\mathcal{E}$ by almost 36% (Table 1, page 24, and Figure 13, left column), producing an approximation of the obstruction similar to that in the noise-free case. However, the resulting boundary layer of cells wrongly marked as fluid is thicker in some parts, and there are many small isolated clusters of incorrectly identified cells scattered throughout the whole domain Ω.

The gap filling corrected the wrong identification of fluid cells near the boundary along the middle part of the obstruction (Figure 13, center column); however, it was significantly less successful in the front and rear parts.

Contrary to the noise-free case, the effect of the cluster filtering is clearly visible now (Figure 13, right column): Many small isolated fluid and solid clusters have been correctly removed. The complete algorithm reduced $\mathcal{E}$ by about 59%.

8.3 Scenario *partial-inside + noise*

This scenario tests how our algorithm performs when the noisy data $\hat{\boldsymbol{u}}$ are unavailable in a small region covering a part of the internal solid boundary as well (Figures 11, center (page 28), and 5, left,

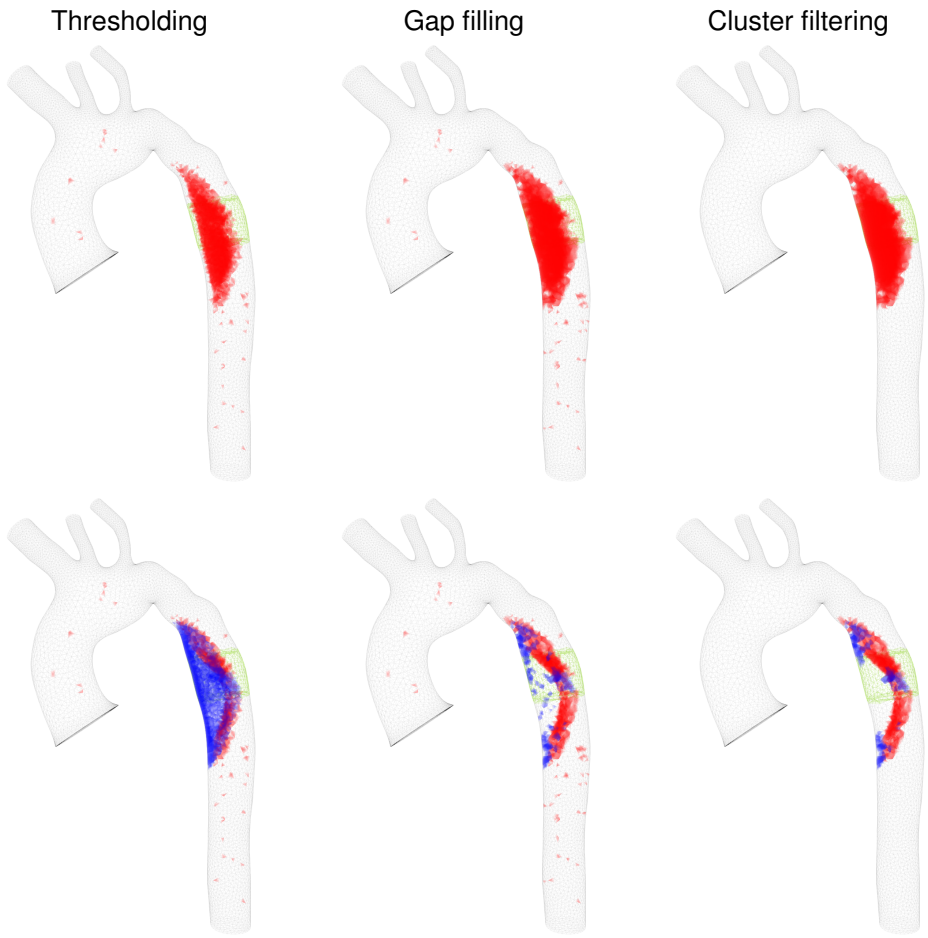


Figure 14: Scenario *partial-inside + noise*: Results of the post-processing steps. Top row: $\Omega \setminus \Omega_{\mathrm{fluid}}$ (in red) after each post-processing step. Bottom row: Cells incorrectly marked as fluid (in blue) and solid (in red) after each post-processing step.

page 21).

The results of Algorithm 1 depicted in Figure 6 (page 22) are slightly worse than in the previous case. Figure 11 (center), page 28, indicates that this could be a consequence of the missing data: The values of σ_h in $(\Omega \setminus \Omega_{\mathrm{meas}}) \cap (\Omega \setminus \Omega_{\mathrm{fluid}})$ — i.e., the solid domain without available measurements — are generally lower then in $\Omega_{\mathrm{meas}} \cap (\Omega \setminus \Omega_{\mathrm{fluid}})$ — i.e., the solid domain where measurements are available.

The approximation of the internal boundary (Figure 14) is accordingly worse than in the previous case: The thresholding marked many cells in the interior of $(\Omega \setminus \Omega_{\mathrm{meas}}) \cap (\Omega \setminus \Omega_{\mathrm{fluid}})$ incorrectly as fluid (cf. the concentration of the blue spots in Figures 14 and 13, left columns), decreasing the shape approximation error $\mathcal{E}$ only by about 29% (Table 1, page 24).

The gap filling corrected many of these marks (Figure 14, center column), resulting in the highest reduction of $\mathcal{E}$ between two consecutive post-processing steps in Table 1 (by almost 22%).

Finally, the cluster filtering removed many tiny isolated clusters of incorrectly marked cells (Figure 14, right column), yielding the final reduction of $\mathcal{E}$ by 55%.

Note that apart from the boundary cells wrongly marked as fluid in $\Omega \setminus \Omega_{\mathrm{meas}}$, the final approximation of the internal boundary is very similar to the previous case (cf. Figures 14 and 13, right columns).

8.4 Scenario *partial-outside + noise*

This scenario is similar to the previous one, but the region of the missing data $\hat{\boldsymbol{u}}$ does not intersect the obstruction (Figure 5, center, page 21).

The results of Algorithm 1 and the approximation of the internal boundary (Figure 6, page 22, and Table 1, page 24) are similar to the case where data are available in the whole domain (Section 8.2). This indicates that it is not crucial for the performance of the algorithm to have access to the data $\hat{\boldsymbol{u}}$ outside the parts of Ω in which the obstruction lies.

Note that the error $\mathcal{E}_S$ is even lower than the scenario *full + noise*. This shows that it is not necessarily advantageous to have more data when these data are corrupted by noise.

8.5 Scenario *partial-large + noise*

In this scenario, the region of the missing data $\hat{\boldsymbol{u}}$ is considerably larger than in Sections 8.3 and 8.4 (Figure 5, right, page 21), but the results of Algorithm 1 (Figure 6, page 22) are comparable to those obtained in the other scenarios.

The volume of the fluid cells wrongly classified as solid, $\mathcal{E}_S$, (see Table 1, page 24) is even significantly better than in all other cases. This is because the algorithm failed to detect a large portion of the obstruction inside $\Omega \setminus \Omega_{\mathrm{meas}}$, i.e., it did not mark any cells as the solid ones in this region at all (see Figure 15). Conversely, the volume of the solid cells wrongly classified as fluid, $\mathcal{E}_F$, is the highest one of all the scenarios.

9 Additional tests

9.1 Effect of mesh refinement

In this section, we investigate how the mesh size affects the performance of our algorithm. We consider the same geometry and data $\hat{\boldsymbol{u}}$ as in the scenario *partial-inside + noise* (Section 8.3), but we solve the flow problem on a finer computational mesh, which we obtained by refining the original mesh uniformly. This also applies to the noisy data, which we interpolate onto the finer grid. However, the control space remains the same. In other words, we use the original coarse mesh

(i) to compute the descent direction in Algorithm 1;

(ii) for σ_h both in Algorithm 1 and in the post-processing.

The finer mesh consists of 848,776 tetrahedra (157,107 DOF for p_h and for each component of $\boldsymbol{u}_h$). The parameter values are the same as in the case *partial-inside + noise* (Section 8.3), with the exception of $\gamma = 5 \cdot 10^8$ $\mathrm{Pa}^2 \cdot \mathrm{s}^4/\mathrm{m}^6$ in (6.2).

The reduction of the cost functional and the $\boldsymbol{L}^2$-error in Figure 16 are slightly better than in the original scenario, and Table 2 shows that the errors $\mathcal{E}_S$ are very similar to those in Section 8.3, but the errors $\mathcal{E}_F$ are slightly larger. However, the overall results are similar to those in Section 8.3 (the detailed figures are omitted).

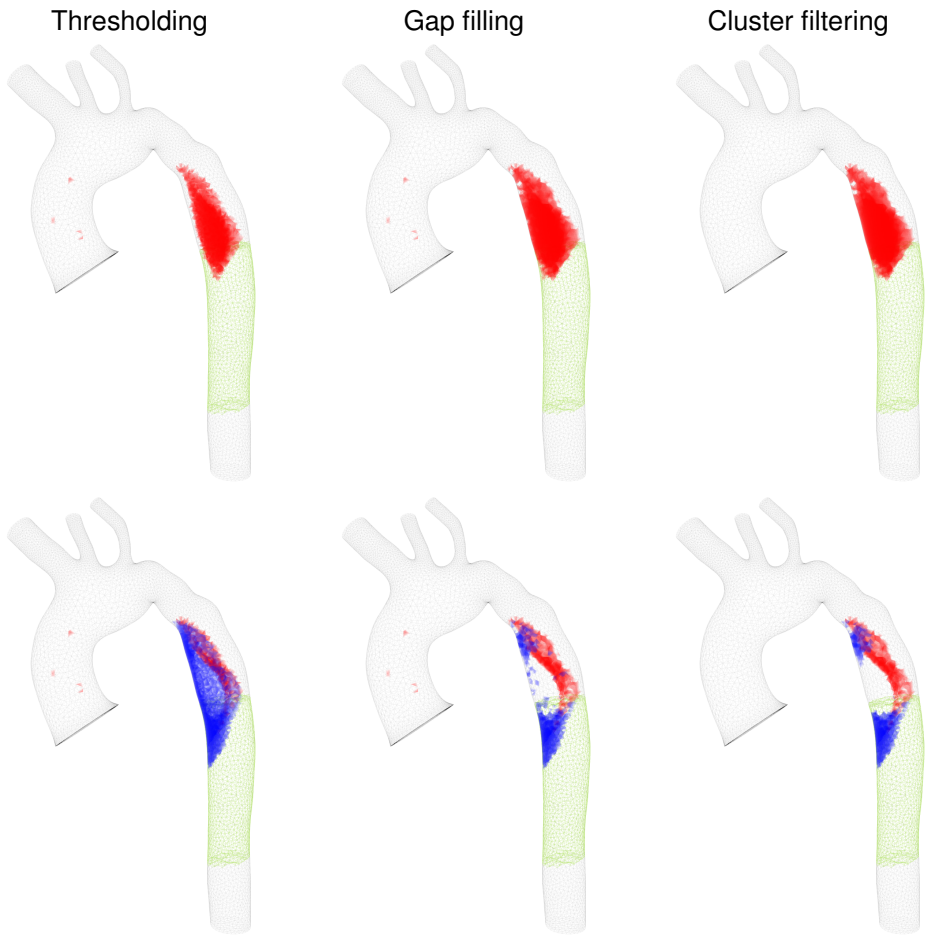


Figure 15: Scenario *partial-large + noise*: Results of the post-processing steps. Top row: $\Omega \setminus \Omega_{\text{fluid}}$ (in red) after each post-processing step. Bottom row: Cells incorrectly marked as fluid (in blue) and solid (in red) after each post-processing step.

		$\mathcal{E}_S$	$\mathcal{E}_F$	$\mathcal{E}$
	Initial errors (m^3)	0	6.895e-6 (100%)	6.895e-6 (100%)
partial-inside + noise	Thresholding	1.884e-6	3.004e-6 (43.6%)	4.888e-6 (70.9%)
	Gap filling	2.591e-6	7.962e-7 (11.5%)	3.387e-6 (49.1%)
	Cluster filtering	2.463e-6	6.401e-7 (9.3%)	3.104e-6 (45.0%)
partial-inside + noise + ref.	Thresholding	1.851e-6	3.020e-6 (43.8%)	4.871e-6 (70.6%)
	Gap filling	2.567e-6	9.377e-7 (13.6%)	3.505e-6 (50.8%)
	Cluster filtering	2.491e-6	7.837e-7 (11.4%)	3.275e-6 (47.5%)

Table 2: Errors $\mathcal{E}_F$, $\mathcal{E}_S$, $\mathcal{E}$ (the total volumes of cells wrongly marked as fluid and solid, respectively, and $\mathcal{E} = \mathcal{E}_F + \mathcal{E}_S$) after each post-processing step for the scenario *partial-inside + noise* considering two different mesh sizes. The percentage values indicate how large the error $\mathcal{E}$ is relative to the initial error.

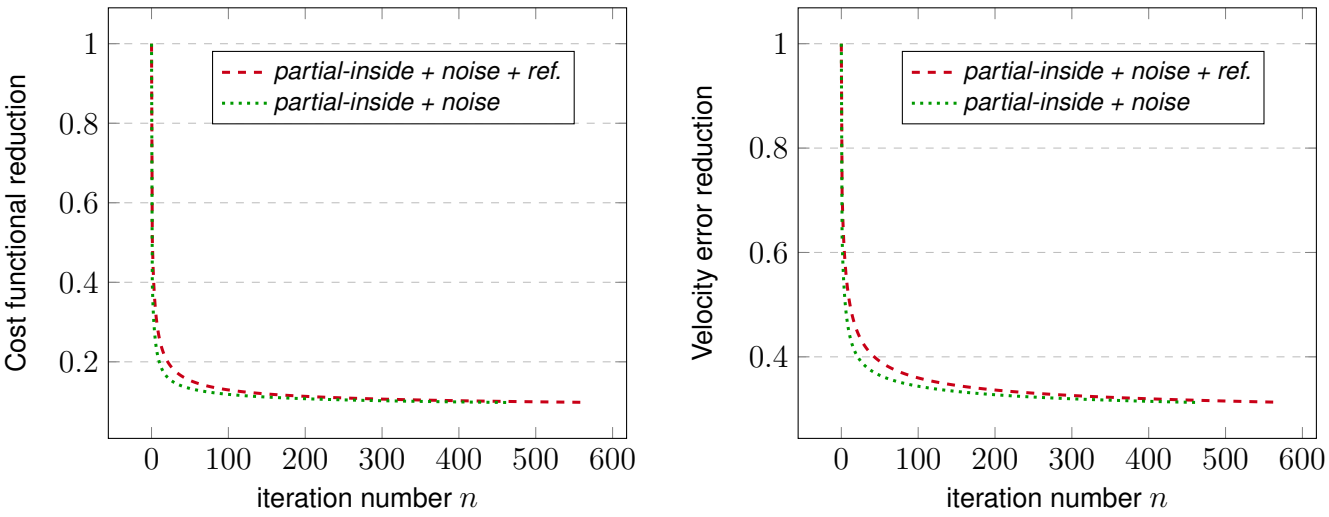


Figure 16: Performance of Algorithm 1 in the test in Section 9.1. Left. Reduction of the cost functional value (computed as F_h^n/F_h^0) as a function of the iteration number n. Right: Reduction of the $\boldsymbol{L}^2$-error with respect to the data, computed as $\dfrac{\|\boldsymbol{u}_h^n - \hat{\boldsymbol{u}}\|^2_{\boldsymbol{L}^2(\Omega_{\text{meas}})}}{\|\boldsymbol{u}_h^0 - \hat{\boldsymbol{u}}\|^2_{\boldsymbol{L}^2(\Omega_{\text{meas}})}}$, as a function of the iteration number n.

		$\mathcal{E}_S$	$\mathcal{E}_F$	$\mathcal{E}$
	Initial errors (m^3)	0	1.343e-5	1.343e-5 (100%)
several + partial-inside + noise	Thresholding	5.166e-6	5.332e-6 (39.7%)	1.050e-5 (78.2%)
	Gap filling	7.770e-6	1.411e-6 (10.5%)	9.181e-6 (68.4%)
	Cluster filtering	7.064e-6	9.412e-7 (7.0%)	8.005e-6 (59.6%)

Table 3: Errors $\mathcal{E}_F$, $\mathcal{E}_S$, $\mathcal{E}$ (the total volumes of cells wrongly marked as fluid and solid, respectively, and $\mathcal{E} = \mathcal{E}_F + \mathcal{E}_S$) after each post-processing step for the scenario with several obstructions. The percentage values indicate how large the error $\mathcal{E}$ is relative to the initial error.

9.2 Effect of several solid subdomains

Additionally, we tested the algorithm in a case with several obstructions inside Ω, using partially unavailable noisy measurements (Figure 17, left). The other parameters were the same as in Section 7.1.

As for the velocity error and the functional reduction, the algorithm performed similarly as in the previous scenarios (the graph is omitted): Both quantities monotonically decreased, fulfilling the stopping criterion in about 1,300 iterations; the error in the velocity decreased by about 96%.

Further, Table 3 shows the errors in the approximation of the solid and fluid domains: The whole algorithm reduced the shape approximation error $\mathcal{E}$ by roughly 40%.

Finally, Figure 17 shows two typical tendencies of the algorithm that we did not see in the previous scenarios:

(i) The algorithm tends to incorrectly locate low permeability regions between the outlet and solid regions close to this outlet. This issue is clearly visible after the thresholding (Figure 17, center-left), and it is corrected by the cluster filtering (Figure 17, right).

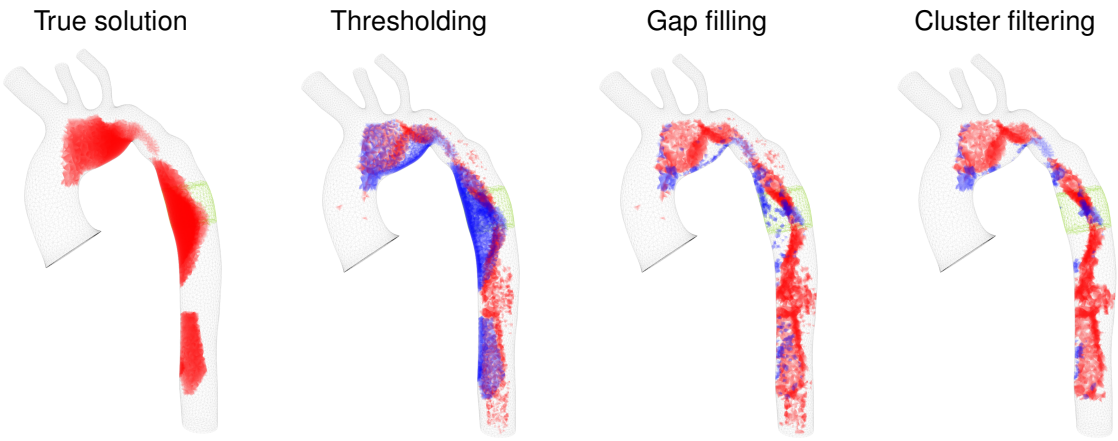


Figure 17: Left: The obstruction $\Omega \setminus \Omega_{\text{flow}}$ (in red) and the region of the unavailable velocity data $\hat{\boldsymbol{u}}$ (in green). The other pictures: The solid cells wrongly marked as fluid (in blue) and the fluid cells wrongly marked as solid (in red) after the thresholding (center-left), after the gap filling (center-right) and after the cluster filtering (right).

(ii) The algorithm tends to incorrectly join near solid regions. In Figure 17, we can see that it wrongly partially joined the two lowermost obstructions but extended the gap between the two uppermost obstructions.

Also note that the middle obstruction is the same as that in the previous scenarios, and accordingly, it is approximated similarly as in Figure 14, page 31.

10 Conclusions

We presented, analyzed, and numerically validated a framework for the reconstruction of internal boundaries of a fluid flow domain using partial internal velocity measurements. This type of problem is motivated by applications in imaging sciences in which information about the internal velocity field can be recovered (by, e.g., phase-contrast MRI), but the data may be highly inaccurate in certain regions (for instance, due to interference with other devices, noise, or limited resolution).

Our framework was based on a Brinkman model with a distributed resistance that modeled the presence of the internal solid boundaries without resolving them explicitly in the computational mesh. We solved the Brinkman problem using a stabilized equal-order finite element method, which allowed us to reduce the computational complexity. The extension of the proposed method to the Navier–Stokes–Brinkman case, including different turbulence models, is the subject of ongoing research.

The proposed estimation algorithm combines (i) a gradient-based optimization of the distributed resistance (a piecewise constant function) done by minimizing a cost functional based on the available velocity data and a L^2-regularization, and (ii) several heuristic post-processing steps. The post-processing depends on several parameters. Our results show that it can considerably improve the robustness and the accuracy of the method, especially when the quality and the availability of the data deteriorate.

In the first part of the paper, we analyzed the existence of the optimal resistance in both the continuous and discrete settings, considering explicitly the case of a stabilized finite element method. As next, we validated the approach on several scenarios with a realistic three-dimensional mesh and synthetic

data of different quality and availability. In each case, the algorithm correctly identified the majority of solid cells (more than 90% in most settings, reduced to 80% when the region of unavailable data covered roughly one-fourth of the whole domain).

We also observed that the algorithm tends to erroneously mark the fluid cells adjacent to the solid boundary (up to one layer of cells). This suggested that the performance could be improved by considering a cost functional based on total variation (TV) regularization, which allows the presence of jumps in the resistance, and hence, may yield sharper interfaces. The usage of different regularizations is beyond the scope of this paper and will be considered in future work.

References

[1] J. Aguayo, C. Bertoglio, and A. Osses. A distributed resistance inverse method for flow obstacle identification from internal velocity measurements. Inverse Problems, 37(2):025010, jan 2021.

[2] J. Aguayo and H. Carrillo. Analysis of obstacles immersed in viscous fluids using Brinkman's law for steady Stokes and Navier–Stokes equations. Technical report, https://arxiv.org/pdf/2012.08635.pdf, 2021.

[3] P. Angot. Analysis of singular perturbations on the Brinkman problem for fictitious domain models of viscous flows. Math. Methods Appl. Sci., 22(16):1395–1412, 1999.

[4] P. Angot. A fictitious domain model for the Stokes/Brinkman problem with jump embedded boundary conditions. C. R. Math. Acad. Sci. Paris, 348(11-12):697–702, 2010.

[5] L. Blank, A. Caiazzo, F. Chouly, A. Lozinski, and J. Mura. Analysis of a stabilized penalty-free Nitsche method for the Brinkman, Stokes, and Darcy problems. ESAIM Math. Model. Numer. Anal., 52(6):2149–2185, 2018.

[6] T. Bodnár, G. P. Galdi, and Š. Nečasová, editors. Fluids under pressure. Advances in Mathematical Fluid Mechanics. Birkhäuser/Springer, Cham, [2020] ©2020.

[7] F. Brezzi and M. Fortin. Mixed and hybrid finite element methods, volume 15 of Springer Series in Computational Mathematics. Springer-Verlag, New York, 1991.

[8] P. G. Ciarlet. The finite element method for elliptic problems, volume 40 of Classics in Applied Mathematics. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2002. Reprint of the 1978 original [North-Holland, Amsterdam; MR0520174 (58 #25001)].

[9] J. Douglas Jr. and J. P. Wang. An absolutely stabilized finite element method for the Stokes problem. Math. Comp., 52(186):495–508, 1989.

[10] H. Garcke, M. Hinze, C. Kahle, and K. F. Lam. A phase field approach to shape optimization in Navier-Stokes flow with integral state constraints. Adv. Comput. Math., 44(5):1345–1383, 2018.

[11] Harald Garcke, Claudia Hecht, Michael Hinze, Christian Kahle, and Kei Fong Lam. Shape optimization for surface functionals in Navier-Stokes flow using a phase field approach. Interfaces Free Bound., 18(2):219–261, 2016.

[12] V. Girault and P.-A. Raviart. Finite element methods for Navier-Stokes equations, volume 5 of Springer Series in Computational Mathematics. Springer-Verlag, Berlin, 1986. Theory and algorithms.

[13] J. Haslinger, K.-H. Hoffmann, and M. Kočvara. Control/fictitious domain method for solving optimal shape design problems. RAIRO Modél. Math. Anal. Numér., 27(2):157–182, 1993.

[14] V. John. Finite Element Methods for Incompressible Flow Problems. Springer Series in Computational Mathematics. Springer International Publishing, 2016.

[15] P. Kogut and G. Leugering. Erratum: "Homogenization of optimal control problems in variable domains. Principle of the fictitious homogenization". Asymptot. Anal., 27(2):183, 2001.

[16] P. Kogut and G. Leugering. Homogenization of optimal control problems in variable domains. Principle of the fictitious homogenization. Asymptot. Anal., 26(1):37–72, 2001.

[17] A. Quarteroni and A. Valli. Domain decomposition methods for partial differential equations. Numerical Mathematics and Scientific Computation. The Clarendon Press, Oxford University Press, New York, 1999. Oxford Science Publications.

[18] A. Takhirov. Stokes-Brinkman Lagrange multiplier/fictitious domain method for flows in pebble bed geometries. SIAM J. Numer. Anal., 51(5):2874–2886, 2013.

[19] U. Wilbrandt. Stokes-Darcy equations. Lecture Notes in Mathematical Fluid Mechanics. Birkhäuser/Springer, Cham, 2019. Analytic and numerical analysis.

[20] U. Wilbrandt, C. Bartsch, N. Ahmed, N. Alia, F. Anker, L. Blank, A. Caiazzo, S. Ganesan, S. Giere, G. Matthies, R. Meesala, A. Shamim, J. Venkatesan, and V. John. Parmoon – a modernized program package based on mapped finite elements. Computers and Mathematics with Applications, 74:74–88, 2016.