Topology optimization of stokes flow on dynamic meshes using simple optimizers

Jensen, Kristian Ejlebjerg

Published in:
Computers & Fluids

Link to article, DOI:

Publication date:
2018

Document Version
Peer reviewed version

Citation (APA):

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.
Topology Optimization of Stokes Flow on Dynamic Meshes using Simple Optimizers

Kristian Ejlebjerg Jensen

PII: S0045-7930(18)30398-0
DOI: 10.1016/j.compfluid.2018.07.011
Reference: CAF 3953

To appear in: Computers and Fluids

Received date: 9 February 2018
Revised date: 4 July 2018
Accepted date: 13 July 2018


This is a PDF file of an unedited manuscript that has been accepted for publication. As a service to our customers we are providing this early version of the manuscript. The manuscript will undergo copyediting, typesetting, and review of the resulting proof before it is published in its final form. Please note that during the production process errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.
Highlights

- Topology optimization and anisotropic mesh adaptation are combined for Stokes flow
- Introduction of a simple optimization method for unconstrained problems
- 1% viscous dissipation in the solid domain requires Darcy numbers of 1e-9
- A source term in the continuity equation causes arbitrarily complex designs
Topology Optimization of Stokes Flow on Dynamic Meshes using Simple Optimizers

Kristian Ejlebjerg Jensen
Technical University of Denmark, Department of Micro- and Nanotechnology, rsteds Plads, Building 345C, Room 108, DK-2800 Kgs. Lyngby

Abstract

We demonstrate that anisotropic mesh adaptation allows for a better description of solid domains than what is often seen in topology optimization of flow problems. This is due to the fact that the physical length scale related to the Brinkman damping term can be efficiently resolved by the non-uniform meshes. We show that the methodology can be applied to a draining problem, which can give arbitrarily complex designs. We use the optimality criteria method as optimizer for this problem, and we also use it to solve a classical drag minimization problem. Finally, we consider the unconstrained reverse flow problem and we use a new optimizer for this, which uses steepest descent with a step size tuned such that the number of design variables with active box constraints increases exponentially throughout the optimization – after a while only the sign of the sensitivity plays a role. All 3 problems are solved in 2D with minimum Darcy numbers as low as $10^{-9}$, which can be necessary for reducing the relative damping in the solid material to under 1%. The 3 problems are also solved in 3D with Darcy numbers equal to $10^{-5}$, and all the results can be reproduced with the MATLAB script, available at https://github.com/KristianE86/trullekrul.

Keywords: Stokes flow, mesh adaptation, topology optimization, MATLAB

1. Introduction

The field of topology optimization has expanded from structural mechanics [1] to fluid dynamics [2] and general multiphysics [3, 4, 5]. The design is often represented implicitly by means of a design field in what is called a density method [1, 2]. Alternatively, the design can be represented using a level-set function that cuts a background mesh to allow for an explicit representation [6, 7]. Fixed structured meshes consisting of quadrilaterals or hexahedra is the norm, but lately there have been some progress in the use of dynamic unstructured meshes for designing structures with low compliance and weight [8, 9]. Methods are often accelerated using parallel implementations [10, 11], but the use of dynamic meshes has the potential to achieve speed-ups by reducing the cost of modelling the void and thus increasing the efficiency. Parallel implementations
have been used to accelerate topology optimization within Stokes flow [12] and natural convection [13], but we have only found one publication involving topology optimization of flow problems on dynamic meshes [14].

The community of structural engineering has a tradition for releasing small MATLAB scripts [15, 16, 17] for the purpose of reproducibility and code reuse, but recently larger codes capable of parallel computation have been released [10]. On the other hand, it is rare to see the adoption of more general purpose simulation libraries, which might be caused by the difficulty of adapting them for topology optimization. However, it is advantageous, when one can exploit the automatic features for non-linear problems and sensitivity analysis [18, 19]. In this regard the FEniCS package and the related dolfin adjoint-module have to be acknowledged [20, 21]. The optimality criteria method is used as optimizer in [16], but we have been unable to find any work similar to our approach to unconstrained optimization.

Problems within fluid dynamics often involve features with strong anisotropy due to convection dominated phenomena near solid boundaries or free surfaces. Elongated elements are thus popular for ensuring a good compromise between accuracy and computational cost [22]. Ongoing work relates to scaling on hardware with distributed memory [23] and performance improvements for iterative solvers by orientation of elements to decrease element angles [24]. We find that minimization of the viscous dissipation in the presence of a source term in the continuity equation leads to branching structures, and we have not been able to find any previous work demonstrating this.

In the following we will introduce the governing equations together with the considered optimization problems. The problems are described in terms of their geometries, objectives, boundary conditions and sensitivity analysis. Then we describe the computational methods related to the details of topology optimization as well as anisotropic mesh adaptation, optimizers and the attached MATLAB script. Finally, we show results and discuss them, before giving a brief conclusion and outlook.

2. Governing equations

We use the density method, which introduces the design variable, $\gamma$. Solid corresponds to $\gamma = 0$, while fluid corresponds to $\gamma = 1$ and intermediate values correspond to a sponge material, but ultimately the design variables should not take on intermediate values. The design variable is defined in the entire computational domain, $\Omega$, and it is free to vary in the design domain, $\Omega_\gamma$, while it is forced to 1 otherwise. The no-slip boundary condition should be enforced on the fluid-solid interface, but it is common to relax this constraint in the sense that solid regions are approximated as sponge material with low permeability, which can be modelled with a Brinkman damping term [2]. This introduces a length scale in the system over which the velocity decreases into the solid material, but if the damping is high, the length scale is small and the no-slip boundary condition becomes well approximated. The governing
equations thus become

\begin{align}
0 &= \nabla \cdot \sigma - Da^{-1} \mathbf{v} \\
\sigma &= \left[ -p + \eta \left( \nabla \mathbf{v} + [\nabla \mathbf{v}]^T \right) \right] \\
0 &= \nabla \cdot \mathbf{v} + \dot{V} + Kp, \quad \text{where} \\
Da^{-1} &= Da^{-1}_{\min} \frac{1 - \gamma}{q + \gamma} \quad \text{and} \\
\xi &= \frac{1}{2} \left( \nabla \mathbf{v} + [\nabla \mathbf{v}]^T \right).
\end{align}

\( \mathbf{v}, p, \eta, \xi \) and Da are the velocity, pressure, viscosity, strain rate tensor and Darcy number. In the attached numerical code we always work with the non-dimensional version of the governing equations, which corresponds to \( \eta = 1 \).

When the design variable is equal to 1, the damping term disappears, while it reaches its maximum value, \( Da^{-1}_{\min} \), for \( \gamma = 0 \). The \( q \) parameter controls the damping for intermediate design variables, which influences the non-discreteness in the presence of a volume constraint\(^2\), i.e. intermediate design variables give rise to little damping for small values of \( q \), and thus their cost is low. Increasing \( q \) throughout the optimization can thus speed up convergence, when there is an upper bound for the average design variable value, \( \gamma_{\text{mean}} \). \( \dot{V} \) is a source term that introduces a volumetric generation of fluid, which could be due to an effect from the 3rd dimension in 2D or a chemical reaction in 3D. The source term causes branching structures, when we minimize the viscous dissipation subject to a volume constraint. Finally, \( K \), is a compressibility constant, which is used to make the problem positive definite\(^1\).

Topology optimization of Stokes flow is well-posed by itself, so there is no need to introduce a minimum length scale by means of a filter as typically seen in structural topology optimization\(^2\).

### 2.1. Objectives and Boundaries

We consider the classical NFL ball problem\(^2, 12\) and a draining problem, where \( \dot{V} \neq 0 \). The setups are illustrated in figure 1 and 2. Both problems use the viscous dissipation as objective,

\[ \mathcal{O} = \int_{\Omega} \phi d\Omega \quad \text{where} \quad \phi = 2 \eta \xi : \xi + Da^{-1} \mathbf{v}^2. \]

Here \( \phi \) is the dissipation density. Both the draining and NFL ball problems are subject to a volume constraint,

\[ \gamma_{\text{mean}} = \int_{\Omega} \gamma d\Omega / \int_{\Omega} d\Omega. \]

![Diagram of NFL ball problem](image1.png)

Figure 1: The classical NFL ball problem with a central design domain and a prescribed velocity on all boundaries. The overall pressure level is fixed by a Dirichlet boundary condition in a corner. Note that there is a volume constraint on the amount of allowed fluid. The 3D case (right) exploits symmetry and extends the design domain to the extreme values of the \( x \) coordinate.

The problems differ in that the damping should be as high as possible for the NFL ball problem, because the damped region is an approximation of a solid region. On the other hand the damped region describes a material

\(^1\)We use \( K = 2.22 \cdot 10^{-13} \).
with finite permeability for the draining problem and the damping magnitude is expected to alter the optimal design (contrary to the NFL ball problem). This is due to the fact that the source term in the continuity equation always causes significant dissipation in the solid domain.

Flow in a shallow channel is a possible interpretation of the finite permeability [25], so the draining problem might be relevant for thread design on rain tires.

Finally, we consider the reverse flow problem [12, 18], where the objective is to reverse the flow velocity in the central point, \( x_0 \), of a pressure driven channel as illustrated in figure 3. This objective can be expressed as

\[
O_S = \int_{\Omega} \mathbf{v} \cdot \mathbf{\hat{x}} \delta(\mathbf{x} - \mathbf{x}_0) d\Omega,
\]

where \( \delta() \) is the delta function\(^2\) and \( \mathbf{\hat{x}} \) is the \( x \) unit vector. At the in- and outlet, \( \partial \Omega_{\sigma} = -\Gamma_{\text{bnd}} \), we supplement the pressure boundary conditions with Neumann boundary conditions for the normal stress and Dirichlet boundary conditions for the tangential velocity, i.e.

\[
\mathbf{g} \cdot \mathbf{\hat{n}} = -\mathbf{I}_{\text{bnd}} \cdot \mathbf{\hat{n}} \quad \text{and} \quad \mathbf{v} = (\mathbf{v} \cdot \mathbf{\hat{n}}) \mathbf{\hat{n}} \quad \text{on} \quad \partial \Omega_{\sigma} = -\Gamma_{\text{bnd}}
\]

The optimal wall thickness in the reverse flow problem decreases as the maximum damping is increased [18], and this tendency for infinitely thin walls is also seen in other convective problems, where the design only serves to shape the fluid flow [5]. Therefore the reverse flow problem is a good proxy for such problems.

It is trivial to extend the problems to three dimensions, but for the draining problem we have chosen to use the geometry shown in figure 4, because it allows for comparison to prior work involving optimization in heat conduction [26]. Finally, we use element IDs to describe the design domain, but the mesh adaptation library only supports 3D element IDs, if they do not give rise to internal geometrical edges, so we make two adjustments to avoid these for the 3D NFL ball problem, see figure 1: first, symmetries are exploited such that only a quarter of the computational domain has to be considered. Second, the design domain is extended to the extreme values of the \( x \) coordinate.

---

\(^2\)In practice we lock the node at \( x_0 \).
Figure 4: The three-dimensional draining problem takes place in a geometry resembling the slice of a birthday cake. No-slip boundary conditions are prescribed all over, except at a ring in the bottom where the tangential velocity components and the pressure are zero. Note that there is still a constraint on the volume of allowed fluid.

2.2. Sensitivity analysis

We apply the continuous adjoint method to find the variation of the objective function, $\delta O$ with respect to a variation in the design variable, $\delta \gamma$,

$$\delta O = \int_{\Omega} \frac{\partial O}{\partial \gamma} \delta \gamma \, d\Omega,$$

where $\frac{\partial G(\gamma, u(\gamma))}{\partial \gamma} = 0$.

Here $G$ are the governing equations, $u$ are the dependent variables and $\frac{\partial O}{\partial \gamma}$ is the sensitivity.

The continuous adjoint analysis of the three problems are detailed in the appendices. The draining problem is self-adjoint,

$$\frac{\partial O}{\partial \gamma} = - \frac{\partial D a}{\partial \gamma} \vec{v}^2.$$  

The NFL ball problem has the same sensitivity, except the sign is flipped, because the problem is subjected to forced flow rather than a pressure gradient.

For the reverse flow problem, we have to solve an adjoint equation system, see Appendix A.3:

$$\frac{\partial O_{R}}{\partial \gamma} = - \frac{\partial D a}{\partial \gamma} \vec{v} \cdot \vec{v} \quad \text{where} \quad 0 = \nabla \cdot ( - \mathcal{L} \vec{P} + \eta ( \nabla \vec{v} + [\nabla \vec{v}]^T ) ) + \vec{F},$$

where $\vec{v}$, $\vec{P}$ and $\vec{F}$ are the adjoint velocity, pressure and volume force, respectively. $\partial \Omega_\n$ and $\partial \Omega_\nu$ are the boundaries associated with Dirichlet and Neumann boundary conditions for the primal problem\textsuperscript{3}. In other words the adjoint flow is driven solely by a point force at $x_0$. It is worth noting that the sensitivity will always be zero on boundaries where no-slip is prescribed, so if the sensitivity is interpolated directly onto the nodes of the mesh, the design variables on the wall will be fixed, but we want to allow the design to slip along the boundary. To facilitate this, we calculate the average value in the element centers and let each element contribute to the sensitivity of its nodes in proportion to its area/volume, see the proj.tri2xy function.

Finally, the only difference between the primal and adjoint equations is the linear form, which corresponds to the right hand side in the discretized system. Because we use a direct solver, the adjoint problem can thus be solved together with the primal one at almost no extra cost.

\textsuperscript{3}For numerical reasons we add a small compressibility term, $K \vec{P}$, to the adjoint continuity equation – similar to equation (2).
3. Computational methods

We solve the governing equations (1-2) using the finite element method with quadratic velocities and linear pressures, i.e., we use the classical Taylor-Hood element [27], see the `fem_stokes` function. The discretized equation system is solved using MATLAB’s default direct solver. The design variables are piecewise linear and thus continuous throughout space, but this property makes the total design variable volume mesh dependent, if the design domain is different from the computational domain. A constant volume fraction constraint thus leads to a (slightly) varying absolute volume constraint in the case of the NFL ball problem. To avoid this we allow for a discontinuous design by treating elements outside the design domain as 100% fluid, which corresponds to removing the support for the design variable basis functions outside the design domain. This does not have any side effects and the design does not become discontinuous in practice, because the optimal solid region does not touch the boundary of the design domain in any of the examples.

The mesh is changed in every optimization iteration and this is done after the sensitivity calculation, so both sensitivities and design variables are interpolated onto the new mesh, before the design variables are updated. The overall pseudo-code thus becomes

# Initialize mesh, design variables and parameters
# Solve forward problem and compute sensitivity
# Compute metric field and adapt mesh
# Interpolate design variables and sensitivity
# Update design variables
# Export post-processing data
# If the maximum number of iterations has been reached, compute STL, otherwise go to #2

This approach has previously been applied to optimization in heat conduction problems and structural compliance minimization [26, 28].

In order to find a good local minimum for the volume constrained problems, we vary the damping for intermediate design variables, which is determined by the $q$ parameter in equation (3). We choose to apply exponential continuation starting from

$$q_{\text{init}} = \frac{100 \gamma_{\text{mean}}}{D_a^{-1}(1 - \gamma_{\text{mean}}) - 1}$$


4.1 % of this value of $q$ corresponds to $D_a^{-1} = 1$ for $\gamma = 0.5$, but very little happens in the beginning, so to reduce the iteration count we multiply with 100.

3.1. Anisotropic mesh adaptation

The continuous mesh framework is popular for anisotropic mesh adaptation [29, 30]. It defines a metric, $M$, which

$$\phi_s = 1 - \frac{\int_{\Omega} \gamma \phi d\Omega}{\int_{\partial \Omega} \phi d\Omega}. \quad (4)$$

We compute the relative amount of dissipation inside the solid domain in order to quantify the influence of stronger damping,
describes the optimal mesh using a continuous and positive symmetric tensor field. An edge \( e \) has ideal length if
\[
1 = e^T M e.
\]
Alternatively one may think of the metric as a field of ellipses/ellipsoids as given by the eigendecomposition, but it is worth noting that the eigenvalues correspond to the square of the optimal inverse edge length along the eigenvector. It is possible to compute a metric that minimizes the error of some functional using adjoint techniques in what is often referred to as goal-oriented mesh adaptation [31]. In this work, we exploit that the metric that minimizes the interpolation error, \( \mathcal{L}^g \), of a piece wise linear function, \( f \), can be expressed explicitly using the Hessian, \( \mathbf{H} \) [32].

\[
\mathbf{M}_{cs} = \frac{1}{\beta} \left[ \det(\mathbf{H}) \right]^{-\frac{1}{2\dim}} \mathbf{H}
\]

where

\[
\mathbf{H} = \text{abs}(\mathbf{H}(f)) \quad \text{and}
\]

\[
\mathcal{L}^g = \left( \int_{\Omega} |f - f_{\text{mesh}}|^g d\Omega \right)^{1/g}
\]

Here \( g \) is the norm in which the interpolation error is minimized and \( \beta \) is a scaling factor, \( f_{\text{mesh}} \) is the numerical approximation of \( f \) and \( \dim \) is the spatial dimension. The computation of the absolute value of the Hessian, \( |\mathbf{H}| \), involves an eigendecomposition such that the absolute value can be taken in the principal frame. The code is available in the \texttt{metric.pnorm} function. It is worth noting that \( f \) is nodal, but we need a nodal Hessian and thus apply two derivative recovery operations for which we use Galerkin projection and MATLAB’s default direct solver. The mesh complexity, \( C \) can be calculated as

\[
C = \int_{\Omega} \sqrt{\det(M)} d\Omega.
\]

This is a good estimate for the node count [30] and it can be used to eliminate \( \beta \) in equation (5) such that another formula involving a target node number, \( C_t \) appears. The \texttt{metric.scale} function does exactly this.

We minimize different 2-norm interpolation errors for the 3 problems

\[
\mathcal{L}^{NFL} = \phi, \quad \mathcal{L}^{drain} = |v| \quad \text{and} \quad \mathcal{L}^{S} = |v| + p.
\]

For the NFL ball problem, we first calculate the average dissipation in each element and then apply Galerkin projection to get a nodal representation, see the \texttt{fem.tri2xy} function. This choice of scalar quantities have been found by trial and error, and they allow us to avoid the use of a minimum/maximum edge length as well as gradation control, but it is likely that better choices exists. At least it is expected that adaptation aimed to minimize the error of the objective function and the constraint will be superior, although one would have to make a choice concerning the priority of an accurate objective versus an accurate constraint.

We make use of local coarsening, refinement, swapping and smoothing to arrive at a discrete mesh conforming to the metric [33]. The operations are sketched in figure 5.
and they are called from the adapt_mesh function, which is taken directly from [34]. We quantify the deviation of the elements from their ideal shape using the Vassilevski element quality functional [35], see the elem_qual function. A pseudo-code of the algorithm is provided below

#1 Initialize adaptation, i=1
#2 Coarsen mesh
#3 Swap edges
#4 Refine mesh
#5 Smooth mesh
#6 If i<10, increment i and go to #2
#7 perform 5 smoothing steps

![Local mesh operations](image)

Figure 5: Four local mesh operations can be used to generate meshes that conform to anisotropic metrics: Coarsening and refinement guarantee a good node density, while swapping and smoothing improve element orientation by maximizing the worst local element quality. Only edge swapping is shown in 3D and it works by triangulating the 3D polygon the same way coarsening is performed in 2D. We never use 2-to-3 swapping.

3.2. Optimizers

Topology optimization in Stokes flow (and the density method in general) often involves the use of optimizers based on moving asymptotes [12, 18, 36], but recently it has been shown that it is possible to use a method based on simultaneous analysis and design [37]. We use the optimality criteria method for minimizing the viscous dissipation subject to a volume constraint. The code is in the optC function, and it is taken directly from [16]. It involves finding a numerical parameter, \( \mu \), such that

\[
\gamma_{i+1} = \max \left[ 0, \min \left( 1, \gamma_i - \sqrt{\frac{\partial O}{\partial \gamma} \mu} \right) \right]
\]

and

\[
\gamma_{\text{mean}} = \int_\Omega \gamma_{i+1} d\Omega
\]

That is, it works by multiplying the design variables with the square root of a scaled version of the continuous sensitivity. Then the box constraints are imposed and the scaling factor is tuned, so the new design complies with the volume constraint. The algorithm is specific to single signed sensitivities and the square root makes it more conservative than simply multiplying with the sensitivity.

The reverse flow problem is unconstrained so we cannot use the optimality criteria method as an update strategy. We use an equally simple technique in the sense that we take a step in the direction of the continuous sensitivity and tune the step size, \( \mu \), such that a predefined proportion of the box constraints become active, i.e.

\[
\gamma_{i+1} = \max \left[ 0, \min \left( 1, \gamma_i - \frac{\partial O}{\partial \gamma} \mu \right) \right]
\]

and

\[
q = \frac{\text{is_zero_or_one(} \gamma_{i+1} \text{)}}{\text{number_of_design_variables}}
\]

Here we have chosen the proportion to be identical to the \( q \) parameter\(^5\). We stick with the exponential continuation,

---

\(^5\)Preliminary tests indicate that it is also feasible to weigh the box constraints with the design variable volume so that the proportion relates to the volume in which the box constraints are active.
but only use the first quarter of the iterations for the continuation. The code is available in the `optD` function, but once all the constraints are active, the new design variable field only depends on the sensitivity and not the current design variable field,
\[
\gamma_{i+1} = \frac{1}{2} - \text{sign} \left( \frac{\partial O}{\partial \gamma} \right) \frac{1}{2}.
\]

### 3.3. Attached MATLAB script

The attached MATLAB script has 8 input parameters:

1. **beta** is the scaling factor in equation (5), but if it is negative, the absolute value will be used as a target node number, see equation (6).

2. **Tvol** is the volume fraction, \( \gamma_{\text{mean}} \), and initial design variable value.

3. **DaMin** is Da_{\text{min}}, i.e. the maximum damping term.

4. **Ls** is a vector with geometrical length scales, i.e. \( L_1 \) and \( L_2 \) for the reverse flow problem and \( L_{\text{char}} \) for the NFL ball problem.

5. **meshszI** is the initial mesh size.

6. **itertotal** is the total number of iterations.

7. **outnm** is a string with the name of an output directory.

8. **prob** is an integer specifying the problem type.

The script outputs the design variable, the pressure, the flow velocity, the dissipation and (in 2D) the stream function to binary VTK format using the `export_vtk` function. The 3D results are also exported to binary STL format using the `export_stl` function. Furthermore, a 2D array is saved in the ascii file `sclr.mat`, and it contains information about the node count, objective function, timings etc.

The code has been tested with the input listed in table 1, all of which are available as comments in the top of the provided script. The code contains some 6,000+ MATLAB statements, so the use of mesh adaptation does complicate the implementation. The optimization loop itself and the assembly of the discretized Stokes problem are however quite compact and well documented, so it should be feasible to use the code for other projects as long as the mesh adaptation is left untouched.

<table>
<thead>
<tr>
<th>No</th>
<th>problem</th>
<th>( C_t )</th>
<th>( u_{\text{max}} )</th>
<th>( \gamma_{\text{mean}} )</th>
<th>( D_{\text{max}}^{-1} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NFL</td>
<td>NA</td>
<td>200</td>
<td>0.8</td>
<td>( 10^3 )</td>
</tr>
<tr>
<td>2</td>
<td>NFL</td>
<td>1 \cdot 10^3</td>
<td>141</td>
<td>0.8</td>
<td>( 10^3 )</td>
</tr>
<tr>
<td>3</td>
<td>NFL</td>
<td>2 \cdot 10^3</td>
<td>200</td>
<td>0.8</td>
<td>( 10^3 )</td>
</tr>
<tr>
<td>4</td>
<td>NFL</td>
<td>4 \cdot 10^3</td>
<td>283</td>
<td>0.8</td>
<td>( 10^3 )</td>
</tr>
<tr>
<td>5</td>
<td>NFL</td>
<td>4 \cdot 10^3</td>
<td>283</td>
<td>0.8</td>
<td>( 10^3 )</td>
</tr>
<tr>
<td>6</td>
<td>NFL</td>
<td>4 \cdot 10^3</td>
<td>283</td>
<td>0.8</td>
<td>( 10^7 )</td>
</tr>
<tr>
<td>7</td>
<td>NFL</td>
<td>4 \cdot 10^3</td>
<td>283</td>
<td>0.999</td>
<td>( 10^3 )</td>
</tr>
<tr>
<td>8</td>
<td>Reverse</td>
<td>NA</td>
<td>10,000</td>
<td>NA</td>
<td>( 10^9 )</td>
</tr>
<tr>
<td>9</td>
<td>Reverse</td>
<td>5 \cdot 10^3</td>
<td>10,000</td>
<td>NA</td>
<td>( 10^9 )</td>
</tr>
<tr>
<td>10</td>
<td>Reverse</td>
<td>1 \cdot 10^4</td>
<td>10,000</td>
<td>NA</td>
<td>( 10^9 )</td>
</tr>
<tr>
<td>11</td>
<td>Reverse</td>
<td>2 \cdot 10^4</td>
<td>10,000</td>
<td>NA</td>
<td>( 10^9 )</td>
</tr>
<tr>
<td>12</td>
<td>Reverse</td>
<td>5 \cdot 10^3</td>
<td>1,000</td>
<td>NA</td>
<td>( 10^5 )</td>
</tr>
<tr>
<td>13</td>
<td>Reverse</td>
<td>5 \cdot 10^3</td>
<td>3,000</td>
<td>NA</td>
<td>( 10^7 )</td>
</tr>
<tr>
<td>14</td>
<td>Drain</td>
<td>1 \cdot 10^4</td>
<td>1,000</td>
<td>0.1</td>
<td>( 10^6 )</td>
</tr>
<tr>
<td>15</td>
<td>Drain</td>
<td>1 \cdot 10^4</td>
<td>1,000</td>
<td>0.1</td>
<td>( 10^7 )</td>
</tr>
<tr>
<td>16</td>
<td>Drain</td>
<td>1 \cdot 10^4</td>
<td>1,000</td>
<td>0.1</td>
<td>( 10^8 )</td>
</tr>
<tr>
<td>17</td>
<td>NFL 3D</td>
<td>1 \cdot 10^4</td>
<td>200</td>
<td>0.95</td>
<td>( 10^5 )</td>
</tr>
<tr>
<td>18</td>
<td>Reverse 3D</td>
<td>1 \cdot 10^5</td>
<td>200</td>
<td>NA</td>
<td>( 10^7 )</td>
</tr>
<tr>
<td>19</td>
<td>Drain 3D</td>
<td>1 \cdot 10^4</td>
<td>200</td>
<td>0.1</td>
<td>( 10^8 )</td>
</tr>
</tbody>
</table>

Table 1: The tested inputs to `top5002.m`. \( C_t \) is the target node count, see equation (6). We use an initial mesh size of \( 0.01L_{\text{char}} \) and \( 0.1L_{\text{char}} \) in 2D and 3D, respectively.

The problems are non-dimensionalized in space using the \( L_{\text{char}} \) length scale (\( R \) for the 3D draining problem).

The other geometrical parameters are \( \delta_{\text{NFL}} = 0.05L_{\text{char}} \) for the NFL balls, while we use \( L_1 = 2L_{\text{char}} \) and \( L_2 = 3L_{\text{char}} \) for the reverse flow problem. Finally, we set \( r_1 = 0.5R \),...
\[ r_2 = 1.5R, \ z_1 = 0.75R, \ rw = 0.7R \text{ and } \theta = \pi/4 \] for the 3D draining problem.

4. Results and discussion

Above each optimization result, we plot the iteration number, objective function, node count, total computational time\(^6\) and relative amount of dissipation inside the solid domain, \(\phi_s\), see equation (4). The latter is substituted with the value of \(Da_{\text{min}}\) for the draining problems. The 2D results are plotted as colored wireframe meshes with blue corresponding to fluid and red solid, i.e. \(\gamma = 1\) and \(\gamma = 0\), respectively. The 3D results are plotted in terms of the \(\gamma = 0.5\) contour. Throughout the following we will refer to the optimizations by their No in table 1. For color figures, the readers are referred to the online version.

Figure 6 shows that the dynamic meshes give better objective functions than fixed meshes for the NFL ball problem with \(Da_{\text{min}}^{-1} = 10^9\) and that the solid-fluid transition region can be well resolved with 4,000 nodes. The transition region depends on \(Da_{\text{min}}\) as demonstrated in figure 7, which also shows that high volume fractions are trivial to handle with this implementation, although the ball occupying just 0.1% of the design domain is not perfectly centered.

Figure 8 shows optimization No 18 for the 3D NFL ball in terms of the iso surface, but also a slice illustrating that

\[ i = 200, \ \phi=42.861, \ \text{nodes}=3584, \ 0.16 \text{ h}, \ \phi_s=0.03 \% \]

\[ i = 283, \ \phi=42.782, \ \text{nodes}=7055, \ 0.46 \text{ h}, \ \phi_s=0.03 \% \]

\[ i = 141, \ \phi=43.347, \ \text{nodes}=1752, \ 0.06 \text{ h}, \ \phi_s=0.02 \% \]

Figure 6: The NFL ball problem is solved on a fixed mesh and three dynamic meshes for \(Da_{\text{min}}^{-1} = 10^9\) (No 1-4). The dynamic meshes are increasingly finer, but even the coarsest give a lower objective function than the fixed mesh.

the ball can be fatter in 3D, because the flow is not restricted to only flowing over and under the ball. In theory the angles at the leading and trailing edges should be 90° and 120° for 2D and 3D, respectively [38].
Figure 7: The NFL ball problem is solved on three dynamic meshes for $Da_{-1}^{-1}$ equal to $10^5$, $10^7$ and $10^9$ (No 4-6). Note that as $Da_{-1}^{-1}$ increases, the solid-fluid transition becomes increasingly sharper, the objective increases, while the dissipation in the solid decreases. Finally, an optimization with a 99.9 % volume fraction at $Da_{-1}^{-1} = 10^9$ is demonstrated (No 7).

The 2D reverse flow problem is solved on different meshes in figure 9, which shows that the fixed mesh is unable to resolve the wall for $Da_{-1}^{-1} = 10^9$ and therefore produces a higher objective function than the dynamic meshes, but making the dynamic meshes finer has almost no effect on the objective function and $\phi_s$, while increasing $Da_{-1}^{-1}$ from $10^5$ to $10^7$ has a strong influence as shown in figure 10. This also shows that $Da_{-1}^{-1} = 10^9$ is necessary to achieve under 1 % damping in the solid material. Finally, the corresponding 3D result is plotted in figure 11 with a breakdown of the computational time in figure 12. A previous study of the 3D reverse flow problem was set up such that diagonal flow was unlikely [12], but despite having plenty

\[
\begin{align*}
i = 283, \phi &= 41.153, \text{nodes}=6011, 0.34 \text{ h}, \phi_s = 3.51 \% \\
i = 283, \phi &= 42.625, \text{nodes}=6592, 0.37 \text{ h}, \phi_s = 0.34 \% \\
i = 283, \phi &= 42.782, \text{nodes}=7055, 0.46 \text{ h}, \phi_s = 0.03 \% \\
i = 283, \phi &= 3.499, \text{nodes}=6236, 0.4 \text{ h}, \phi_s = 0.07 \%
\end{align*}
\]
of room for diagonal flow\textsuperscript{7}, we too find a symmetric design. The fact that only 6 % of the computational time is used for the mesh adaptation, even though a pure MATLAB implementation is used, can be attributed to the 2nd order discretization of the velocity and the use of a linear solver.

Lastly, the 3D and 2D draining results are shown in figures 13 and 14, respectively. Branching structures are commonly observed in heat conduction even though hairy structures are expected \cite{39}, but in the case of fluid flow, branching structures are expected, because the flow rate in a pressure driven pipe scales with the radius to the fourth power.

\textsuperscript{7}Diagonal flow in the sense that the optimal flow configuration has a symmetry plane that divides the domain along the diagonal.

5. Conclusion

Our method introduces substantial complexity, which is obvious from the 6,000+ MATLAB statements required to reproduce the results.

Nevertheless, we have demonstrated that it is feasible
The 3D reverse flow problem gives a design similar to the 2D case. Below two diagonal slices provide an alternative illustration of the working mechanism and to illustrate the spatial element concentrations one of the slices shows the edges of the element cuts. Note that the slicing of large tetrahedrons may produce small triangles, so one should be careful about drawing any conclusions with regards to mesh quality from this representation.

A breakdown of the computational time used to do optimization No 18. The node count is an average value for the whole optimization. The solution of the forward and adjoint problems dominate the computational cost, while the design variable update and metric calculation take up less than 0.6% of the time.

Figure 13: The 3D draining result is shown for $Da_{\text{min}} = 10^{-5}$ and a volume fraction of 10%.

The most obvious improvement relates to the use of continuation in $Da_{\text{min}}$ as this should reduce the number of iterations required for the 2D reverse flow problem with $Da_{\text{min}} = 10^{-9}$. Furthermore, it is preferable to drive the mesh adaptation using a goal oriented approach rather than the heuristic techniques used here.
Alternatively, the method could be extended to account for inertia, which is relevant in terms of both drag minimization and draining problems. This would increase the cost of the forward problem even more, which would call for the use of an iterative solver. A faster solver might also allow the method to be used for designing 3D mixers [5] or other devices with thin walls for shaping fluid flow. Such problems would be interesting in terms of testing the optimizer used for the unconstrained problems.

7. Acknowledgement

I am grateful that Fridolin Okkels got the idea for the draining problem and shared it with me. This work is supported by the Villum Foundation (Grant No. 9301) and the Danish Council for Independent Research (DNRF122).

Figure 14: The 2D draining problem is shown for three different values of $D_{\text{amin}}$ with decreasing values resulting in increasingly complex designs and objective functions (No 14-16).
Appendix A. Sensitivity Analysis

Appendix A.1. Draining problem

Governing equations and objective function

\[ \mathcal{O} = \int_{\partial \Omega} \mathbf{v} \cdot \sigma \hat{\mathbf{n}} ds = \int_{\Omega} (\xi : \sigma - \mathbf{v} \cdot \mathbf{F}) d\Omega \]  
(A.1)

\[ 0 = \nabla \cdot \sigma + \mathbf{F} \]  
(A.2)

\[ 0 = \nabla \cdot \mathbf{v} + \dot{V} \]  
where

\[ \sigma = -p + 2\eta \xi, \quad \xi = \frac{1}{2} \left[ \nabla \mathbf{v} + [\nabla \mathbf{v}]^T \right] \]

\[ \mathbf{F} = -D^{-1} \mathbf{v}, \]

where \( \mathbf{v}, p, \eta, \mathbf{F}, \sigma, \xi \) and \( D^{-1} \) are the velocity, pressure, viscosity, volumetric force, stress, rate of deformation and damping term, respectively. It is easy to see that the objective function can be simplified due to equation (A.3),

\[ \mathcal{O} = \int_{\Omega} \left( 2\eta \xi : \epsilon + \dot{\mathbf{V}} - \mathbf{v} \cdot \mathbf{F} \right) d\Omega, \]  
(A.4)

because \( \xi : I = \nabla \cdot \mathbf{v} = -\dot{\mathbf{V}}. \)

A variation in the design variable field, \( \delta \gamma \) will result in a variation of the objective function,

\[ \delta \mathcal{O} = \int_{\Omega} \left( 2\eta \frac{\partial \xi}{\partial \gamma} : \epsilon + \frac{\partial \dot{\mathbf{V}}}{\partial \gamma} - \frac{\partial \mathbf{v}}{\partial \gamma} \cdot \frac{\partial \mathbf{F}}{\partial \gamma} \right) \delta \gamma d\Omega \]

\[ + \int_{\Omega} \frac{\partial D^{-1}}{\partial \gamma} \mathbf{v}^2 \delta \gamma d\Omega \]  
(A.5)

This can be expanded further, but the point is that the derivatives of \( \mathbf{v} \) and \( p \) with respect to \( \gamma \) are unknown and therefore the derivative of \( \xi \) is also unknown, so it somehow has to be eliminated. This can be achieved by constructing other partial differential equations with the same terms and adding/subtraction the equations. The starting point of this procedure is the introduction of the adjoint velocity \( \tilde{\mathbf{v}} \), which by definition is invariant with respect to the \( \delta \gamma \) variation. Multiplying \( \tilde{\mathbf{v}} \) with equation (A.2) and integrating over the domain, \( \Omega \), yields

\[ 0 = \int_{\Omega} \tilde{\mathbf{v}} \cdot \left( \nabla \cdot \sigma + \mathbf{F} \right) d\Omega \]

\[ = \int_{\partial \Omega} \tilde{\mathbf{v}} \cdot \sigma \hat{\mathbf{n}} ds - \int_{\Omega} (\xi : \sigma - \tilde{\mathbf{v}} \cdot \mathbf{F}) d\Omega \]

\[ = \int_{\partial \Omega} \tilde{\mathbf{v}} \cdot \sigma \hat{\mathbf{n}} ds - \int_{\Omega} (\xi : \sigma - \tilde{\mathbf{v}} \cdot \mathbf{F}) d\Omega \]  
(A.6)

where \( \xi = \frac{1}{2} \left( \nabla \tilde{\mathbf{v}} + [\nabla \tilde{\mathbf{v}]^T} \right). \)

Here we have used the divergence theorem and the fact that the stress tensor is symmetric. Taking the variation with respect to \( \delta \gamma \) gives

\[ 0 = \int_{\partial \Omega} \tilde{\mathbf{v}} \cdot \frac{\partial \sigma}{\partial \gamma} : \tilde{\mathbf{n}} \delta \gamma ds \]

\[ - \int_{\Omega} \left( \xi : \frac{\partial \mathbf{v}}{\partial \gamma} - \tilde{\mathbf{v}} \cdot \frac{\partial \mathbf{F}}{\partial \gamma} \right) \delta \gamma d\Omega \]

\[ = \int_{\partial \Omega} \tilde{\mathbf{v}} \cdot \frac{\partial \sigma}{\partial \gamma} : \tilde{\mathbf{n}} \delta \gamma ds \]

\[ - \int_{\Omega} \left( \xi : \frac{\partial \mathbf{v}}{\partial \gamma} - \tilde{\mathbf{v}} \cdot \frac{\partial \mathbf{F}}{\partial \gamma} \right) \delta \gamma d\Omega \]

\[ - \int_{\Omega} \tilde{\mathbf{v}} \cdot \left[ D^{-1} \frac{\partial \mathbf{v}}{\partial \gamma} + \frac{\partial D^{-1}}{\partial \gamma} \mathbf{v} \right] \delta \gamma d\Omega, \]  
(A.7)

adding equations (A.5) and (A.7) yields

\[ \delta \mathcal{O} = \int_{\partial \Omega} \tilde{\mathbf{v}} \cdot \frac{\partial \sigma}{\partial \gamma} : \tilde{\mathbf{n}} \delta \gamma ds \]

\[ + \int_{\Omega} \left( 2\eta \xi - \eta \tilde{\xi} \right) : \frac{\partial \mathbf{v}}{\partial \gamma} + \frac{1}{2} \tilde{\xi} : \frac{\partial p}{\partial \gamma} \right) \delta \gamma d\Omega \]

\[ + \int_{\Omega} \left[ 2D^{-1} \mathbf{v} - D^{-1} \tilde{\mathbf{v}} \right] \frac{\partial \mathbf{v}}{\partial \gamma} \delta \gamma d\Omega \]

\[ + \int_{\Omega} \left( \frac{\partial D^{-1}}{\partial \gamma} \mathbf{v} \cdot (\mathbf{v} - \tilde{\mathbf{v}}) \right) \delta \gamma d\Omega. \]


It is easy to see that we can eliminate all derivatives with respect to $\gamma$ by assuming $\bar{v} = 2v$. The boundary term remains, but for a pressure driven setup this also drops out, if the boundary conditions are restricted to zero velocity and fixed boundary stress. Finally $\bar{\varepsilon} : \bar{\varepsilon} = \nabla \cdot \bar{v} = \nabla \cdot v = 0$, so we get

$$
\delta O = -\int_{\Omega} \frac{\partial Da^{-1}}{\partial \gamma} \nabla^2 \delta \gamma d\Omega \quad (A.8)
$$

Appendix A.2. NFL ball problem

$$
\mathcal{O} = \int_{\partial \Omega} v \cdot \sigma \hat{n} ds = \int_{\Omega} \left( \frac{1}{2} : \sigma - v \cdot F \right) d\Omega \quad (A.9)
$$

$$
0 = \nabla \cdot \sigma + F \quad (A.10)
$$

$$
0 = \nabla \cdot v \quad (A.11)
$$

where $v$, $p$, $\eta$, $F$, $\sigma$, $\varepsilon$ and $Da^{-1}$ are the velocity, pressure, viscosity, volumetric force, stress, rate of deformation and damping term, respectively. A variation in the design variable field, $\delta \gamma$ will result in a variation of the objective function,

$$
\delta \mathcal{O} = \int_{\partial \Omega} \left( \frac{\partial v}{\partial \gamma} \cdot \sigma + \frac{\partial \sigma}{\partial \gamma} \cdot \hat{n} \delta \gamma ds \right) \quad (A.12)
$$

The point is that the derivatives of $v$ and $p$ with respect to $\gamma$ are unknown and therefore the derivative of $\sigma$ is also unknown, so it has to be eliminated. We introduce adjoint velocity $\bar{v}$, multiply it with equation (A.10) and integrate over the domain:

$$
0 = \int_{\Omega} \bar{v} \cdot (\nabla \cdot \sigma + F) d\Omega
= \int_{\partial \Omega} \bar{v} \cdot \sigma \cdot \hat{n} ds + \int_{\Omega} (\bar{\varepsilon} \cdot \frac{\partial \sigma}{\partial \gamma} + \bar{\varepsilon} \cdot F) d\Omega,
$$

where $\bar{\varepsilon} = \frac{1}{2} \left( \nabla \bar{v} + [\nabla \bar{v}]^T \right)$

$$
= \int_{\partial \Omega} (\bar{v} \cdot \sigma - v \cdot 2\eta \bar{\varepsilon} \cdot \hat{n} ds + \int_{\Omega} (v \cdot \nabla \cdot 2\eta \bar{\varepsilon} + \bar{\varepsilon} \cdot p + \bar{v} \cdot F) d\Omega.
$$

The variation becomes

$$
0 = \int_{\partial \Omega} \left( \frac{\partial \sigma}{\partial \gamma} \cdot \bar{v} + \frac{\partial v}{\partial \gamma} \cdot \bar{\sigma} \right) \cdot \hat{n} \delta \gamma ds + \int_{\Omega} \left( \frac{\partial \sigma}{\partial \gamma} \cdot \bar{v} + \frac{\partial v}{\partial \gamma} \cdot \bar{\sigma} \right) \cdot \hat{n} \delta \gamma ds.
$$

We now repeat the process for equation (A.11) and the adjoint pressure, $\bar{p}$

$$
0 = \int_{\Omega} \bar{p} \nabla \cdot v d\Omega
= \int_{\Omega} \bar{p} \nabla \cdot v d\Omega
= \int_{\partial \Omega} v \cdot \bar{p} \cdot \hat{n} ds - \int_{\Omega} \nabla \cdot \bar{p} \cdot v d\Omega.
$$

The variation becomes

$$
0 = \int_{\partial \Omega} \frac{\partial v}{\partial \gamma} \cdot \bar{p} \cdot \hat{n} \delta \gamma ds
= \int_{\partial \Omega} \nabla \cdot \bar{p} \cdot \frac{\partial v}{\partial \gamma} \delta \gamma d\Omega.
$$

Adding equations (A.12), (A.13) and (A.14) yields

$$
\delta \mathcal{O} = \int_{\partial \Omega} \left( \frac{\partial v}{\partial \gamma} \cdot \left[ \sigma + \frac{1}{2} \bar{p} - 2\eta \bar{\varepsilon} \right] + \frac{\partial \sigma}{\partial \gamma} \cdot [v + \bar{v}] \right) \cdot \hat{n} \delta \gamma ds
$$
\[ + \int_{\Omega} \frac{\partial v}{\partial \gamma} \cdot \left[ \nabla \cdot \left( -\frac{1}{2} \tilde{p} + 2\eta \tilde{\varepsilon} \right) - D_{\alpha}^{-1} \tilde{v} \right] \delta \gamma d\Omega \]
\[ + \int_{\bar{x}} \frac{\partial p}{\partial \gamma} - \frac{\partial D_{\alpha}^{-1}}{\partial \gamma} \cdot v \cdot \tilde{v} \delta \gamma d\Omega \]

The derivative of the velocity, \( v \), with respect to the design variable, \( \gamma \) drops out on the boundary of a flow driven setup. For the the volume terms to drop out, we must require that the adjoint velocity and pressure satisfy the same equations as the primal ones, i.e. \( \bar{\tilde{\varepsilon}} = \nabla \cdot \bar{\tilde{v}} \). This means that the boundary terms for the adjoint variables have to be flipped to satisfy \( \tilde{v} + v = 0 \) such that the last boundary term can be canceled. We are left with

\[ \delta \mathcal{O} = \int_{\Omega} \frac{\partial D_{\alpha}^{-1}}{\partial \gamma} \nabla^2 \delta \gamma d\Omega. \]  
(A.15)

It is worth noting that the sensitivity for a pressure driven setup is identical except for the sign, see equation (A.8).

**Appendix A.3. Reverse flow**

Consider the following optimization problem, where the Stokes flow in some point, \( x_0 \), is minimized along the \( x \)-direction.

\[ \mathcal{O} = \int_{\Omega} \delta(x - x_0) v \cdot \hat{x} d\Omega \]  
(A.16)

\[ 0 = \nabla \cdot \bar{\sigma} + \bar{F} \]
\[ 0 = \nabla \cdot \bar{\varepsilon} \quad \text{where} \]

\[ \bar{\sigma} = -\frac{1}{2} \bar{p} + 2\eta \bar{\varepsilon} \]
\[ \bar{\varepsilon} = \frac{1}{2} \left( \nabla v + [\nabla v]^T \right) \]
\[ \bar{F} = -D_{\alpha}^{-1} v. \]

A variation in the design variable \( \delta \gamma \), which will results in a variation of the objective function,

\[ \delta \mathcal{O} = \int_{\Omega} \delta(x - x_0) \frac{\partial v}{\partial \gamma} \cdot \hat{x} \delta \gamma d\Omega \]  
(A.18)

We eliminate the derivative of \( v \) with respect to \( \gamma \) by multiplying the adjoint velocity with equation (A.2) and integrating over the domain, \( \Omega \):

\[ 0 = \int_{\partial \Omega} \bar{\varepsilon} \cdot \hat{n} d\sigma - \int_{\Omega} \left( \nabla \cdot \bar{\varepsilon} - 2\eta \nabla \bar{v} \right) d\Omega \]
\[ = \int_{\partial \Omega} \bar{\varepsilon} \cdot \hat{n} d\sigma - \int_{\Omega} \left( \bar{\varepsilon} - \bar{v} \cdot \bar{F} \right) d\Omega \]

where \( \frac{1}{2} \left( \bar{\varepsilon} = \nabla \bar{v} + [\nabla \bar{v}]^T \right) \)

\[ = \int_{\partial \Omega} \bar{v} \cdot \hat{n} d\sigma - \int_{\Omega} \left( \bar{\varepsilon} - 2\eta \nabla \bar{v} \right) \cdot \hat{n} d\sigma \]
\[ = \int_{\Omega} \left( \bar{\varepsilon} - \bar{v} \cdot \bar{F} \right) \cdot \hat{n} d\sigma \]
\[ = \int_{\Omega} \left( -p \nabla \cdot \bar{v} + 2\eta \nabla \cdot [\nabla \bar{v} + [\nabla \bar{v}]^T] \right) d\Omega, \]

where we have used the divergence theorem (twice) and the fact that the stress tensor is symmetric. Taking the variation with respect to \( \delta \gamma \) yields

\[ 0 = \int_{\partial \Omega} \left( \bar{v} \cdot \frac{\partial \sigma}{\partial \gamma} - 2\eta \frac{\partial v}{\partial \gamma} \cdot \frac{\bar{\varepsilon}}{2} \right) \cdot \hat{n} \delta \gamma d\sigma \]
\[ = \int_{\partial \Omega} \left( \bar{v} \cdot \frac{\partial \sigma}{\partial \gamma} - 2\eta \frac{\partial v}{\partial \gamma} \cdot \frac{\bar{\varepsilon}}{2} \right) \cdot \hat{n} \delta \gamma d\sigma \]
\[ = \int_{\partial \Omega} \left( \bar{v} \cdot \frac{\partial \sigma}{\partial \gamma} - \frac{\partial v}{\partial \gamma} \cdot \frac{\bar{\varepsilon}}{2} \right) \cdot \hat{n} \delta \gamma d\sigma] (A.19) \]
Repeating the process for equation (A.17) gives
\[
0 = \int \tilde{\rho} \nabla \cdot \mathbf{v} d\Omega
= \int_{\partial \Omega} \tilde{\rho} \cdot \mathbf{n} ds - \int_{\Omega} \mathbf{v} \cdot \nabla \tilde{\rho} d\Omega
0 = \int_{\partial \Omega} \frac{\partial}{\partial \gamma} \tilde{\rho} \cdot \mathbf{n} \delta \gamma ds - \int_{\Omega} \frac{\partial}{\partial \gamma} \mathbf{v} \cdot \nabla \tilde{\rho} \delta \gamma d\Omega \tag{A.20}
\]
Adding equations (A.18), (A.19) and (A.20) yields
\[
\delta O = \int_{\Omega} \frac{\partial}{\partial \gamma} \mathbf{v} \cdot \left[ - \nabla \tilde{\rho} + 2\eta \tilde{\gamma} + \tilde{\mathbf{F}} \right] \delta \gamma d\Omega
+ \int_{\Omega} \left( -\tilde{\mathbf{v}} \cdot \nabla \frac{\partial \mathbf{Da}^{-1}}{\partial \gamma} + \frac{\partial \mathbf{v}}{\partial \gamma} \cdot \tilde{\mathbf{F}} \right) \delta \gamma d\Omega
+ \int_{\partial \Omega} \left( \tilde{\mathbf{v}} \cdot \frac{\partial \tilde{\gamma}}{\partial \gamma} + \frac{\partial \mathbf{v}}{\partial \gamma} \cdot \left( \frac{-\tilde{\gamma}}{2} \right) \right) \cdot \mathbf{n} \delta \gamma ds,
\]
where \( \tilde{\mathbf{F}} = \delta(\mathbf{x} - x_0) \hat{\mathbf{x}} - \mathbf{Da}^{-1} \tilde{\mathbf{v}} \),
so the sensitivity becomes
\[
\frac{\partial O}{\partial \gamma} = -\tilde{\mathbf{v}} \cdot \frac{\partial \mathbf{Da}^{-1}}{\partial \gamma} \text{ where } 0 = \nabla \cdot \tilde{\mathbf{v}}.
\]
Finally, the same analysis might be carried out with \( \dot{\mathbf{V}} \neq 0 \) and
\[
\mathcal{O} = \int_{\Omega} \dot{\mathbf{V}} \mathbf{p} d\Omega,
\]
so that the point force drops out and the problem becomes self-adjoint.

References
puter Methods in Applied Mechanics and Engineering, 282:22–

[8] Asger Nyman Christiansen, J Andreas Bærentzen, Morten
Nobel-Jørgensen, Niels Aage, and Ole Sigmund. Combined
shape and topology optimization of 3d structures. Computers

[9] Kristian Ejlebjerg Jensen. Anisotropic mesh adaptation and
topology optimization in three dimensions. Journal of Mechani-

Topology optimization using petsc: An easy-to-use, fully par-
ellel, open source topology optimization framework. Structural

optimization in 3d using parallel computing. Computer methods

[12] Niels Aage, Thomas H Poulsen, Allan Gersborg-Hansen, and
Ole Sigmund. Topology optimization of large scale stokes
flow problems. Structural and Multidisciplinary Optimization,

[13] Joe Alexandersen, Ole Sigmund, and Niels Aage. Large scale
three-dimensional topology optimisation of heat sinks cooled
by natural convection. International Journal of Heat and Mass

[14] Florian Feppon, Grégoire Allaire, Felipe Bordeu, Julien Cortial,
and Charles Dapogny. Shape optimization of a coupled thermal
fluid-structure problem in a level set mesh evolution framework.
2018.

Matlab-based finite element method solver for large problems.
Geochemistry, Geophysics, Geosystems, 9(4), 2008.

[16] Erik Andreassen, Anders Clausen, Mattias Schevenels, Boyan S
Lazarov, and Ole Sigmund. Efficient topology optimization in
matlab using 88 lines of code. Structural and Multidisciplinary

[17] Anderson Pereira, Cameron Talischi, Glauco H Paulino,
Ivan FM Menezes, and Marcio S Carvalho. Fluid flow topol-
ogy optimization in polytop: stability and computational im-
plementation. Structural and Multidisciplinary Optimization,

[18] Laurits Højgaard Olesen, Fridolin Okkels, and Henrik Bruus.
A high-level programming-language implementation of topo-
logy optimization applied to steady-state navier–stokes flow.
International Journal for Numerical Methods in Engineering,

[19] Kristian Ejlebjerg Jensen. Solving stress and compliance con-
strained volume minimization using anisotropic mesh adapta-
tion, the method of moving asymptotes and a global p-norm.
Structural and Multidisciplinary Optimization, 54(4):831–841,
2016.

[20] Martin Alnæs, Jan Blechta, Johan Hake, August Johansson,
Benjamin Kehlet, Anders Logg, Chris Richardson, Johannes
Ring, Marie E Rognes, and Garth N Wells. The fenics project

[21] Patrick E Farrell, David A Ham, Simon W Funke, and Marie E
Rognes. Automated derivation of the adjoint of high-level tran-
sient finite element programs. SIAM Journal on Scientific Com-

[22] Frédéric Alauzet and Adrien Loseille. A decade of progress on
anisotropic mesh adaptation for computational fluid dynamics.

[23] Adrien Loseille, Frédéric Alauzet, and Victorien Menier. Unique
cavity-based operator and hierarchical domain partitioning for
fast parallel generation of anisotropic meshes. Computer-Aided

[24] D Marcum and F Alauzet. 3d metric-aligned and orthogonal so-
lution adaptive mesh generation. Procedia Engineering, 203:78–
90, 2017.

[25] Kristian Ejlebjerg Jensen, Peter Szabo, Fridolin Okkels, and
MA Alves. Experimental characterisation of a novel viscoelastic

[26] Kristian Ejlebjerg Jensen. Solving 2d/3d heat conduction prob-
lems by combining topology optimization and anisotropic mesh adaptation. In 12th World Congress on Structural and Multi-disciplinary Optimization 5th-9th June, 2017.


[37] Anton Evgrafov. On chebyshevs method for topology optimiza-

