Body-fitted topology optimization of 2D and 3D fluid-to-fluid heat exchangers

Florian Feppon

PhD advisors : Grégoire Allaire, Charles Dapogny

Safran Tech advisors : Julien Cortial, Felipe Bordeu.

10th TOP Webinar
2021, February 23rd

Problem at hand

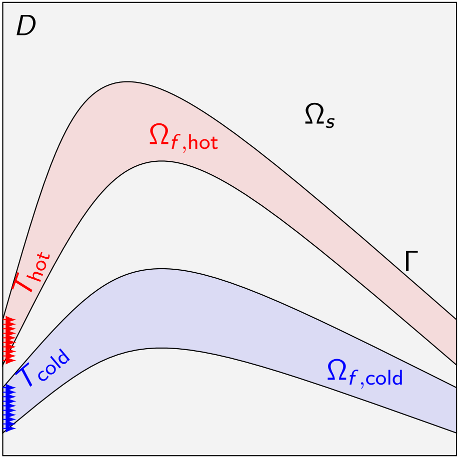


Figure: Settings of the heat exchanger topology optimization problem.

Problem at hand

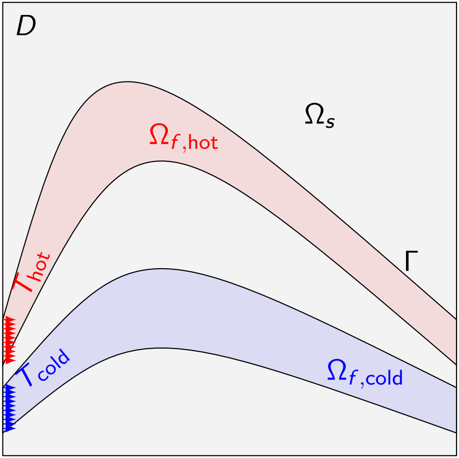


Figure: Settings of the heat exchanger topology optimization problem.

- Navier-Stokes flows in the hot and cold phases $\Omega_{f,hot}$ and $\Omega_{f,cold}$.

Problem at hand

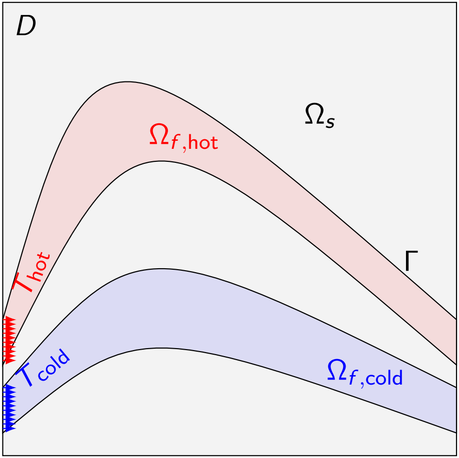


Figure: Settings of the heat exchanger topology optimization problem.

- Navier-Stokes flows in the hot and cold phases $\Omega_{f,hot}$ and $\Omega_{f,cold}$.
- Thermal convection in the fluid phase $\Omega_f = \Omega_{f,hot} \cup \Omega_{f,cold}$.
- Thermal diffusion in Ω_s and Ω_f with conductivities $k_s >> k_f$.

Problem at hand

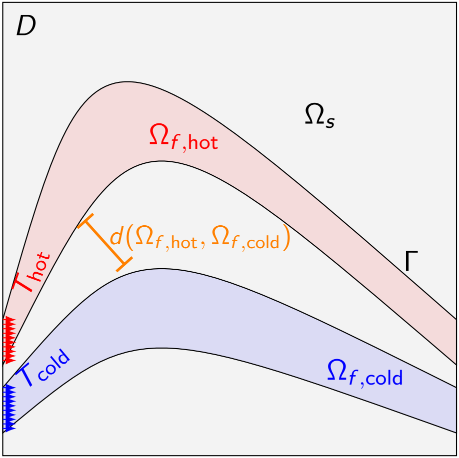


Figure: Settings of the heat exchanger topology optimization problem.

- Navier-Stokes flows in the hot and cold phases $\Omega_{f,hot}$ and $\Omega_{f,cold}$.
- Thermal convection in the fluid phase $\Omega_f = \Omega_{f,hot} \cup \Omega_{f,cold}$.
- Thermal diffusion in Ω_s and Ω_f with conductivities $k_s >> k_f$.
- Non-penetration constraint:

$$d(\Omega_{f,\text{hot}}, \Omega_{f,\text{cold}}) \geqslant d_{\min}.$$

Problem at hand

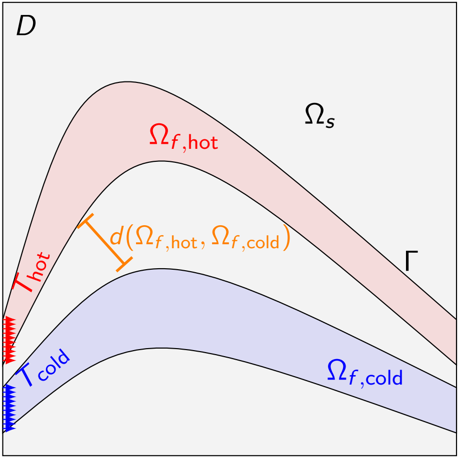


Figure: Settings of the heat exchanger topology optimization problem.

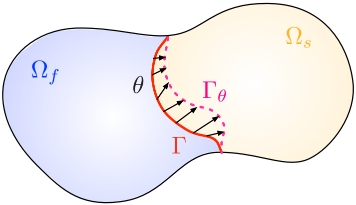
- Navier-Stokes flows in the hot and cold phases $\Omega_{f,hot}$ and $\Omega_{f,cold}$.
- Thermal convection in the fluid phase $\Omega_f = \Omega_{f,hot} \cup \Omega_{f,cold}$.
- Thermal diffusion in Ω_s and Ω_f with conductivities $k_s >> k_f$.
- Non-penetration constraint:

$$d(\Omega_{f,\text{hot}}, \Omega_{f,\text{cold}}) \geqslant d_{\text{min}}.$$

- In 3D!

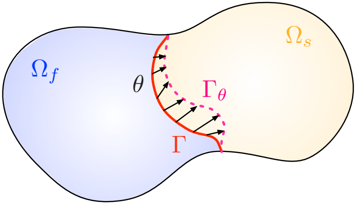
The boundary variation method of Hadamard

$$\min_{\Gamma} \; J(\Gamma)$$



The boundary variation method of Hadamard

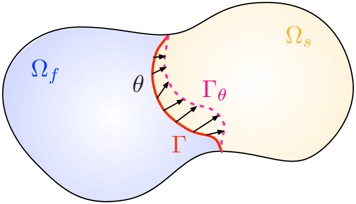
$$\min_{\Gamma} \; J(\Gamma)$$



$$\Gamma_{\boldsymbol{\theta}} = (I + \boldsymbol{\theta})\Gamma, \text{ with } \boldsymbol{\theta} \in W_0^{1,\infty}(D, \mathbb{R}^d), \;\; ||\boldsymbol{\theta}||_{W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)} < 1.$$

The boundary variation method of Hadamard

$$\min_{\Gamma} \; J(\Gamma)$$

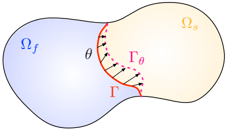


$$\Gamma_{\boldsymbol{\theta}} = (I + \boldsymbol{\theta})\Gamma, \text{ with } \boldsymbol{\theta} \in W_0^{1,\infty}(D, \mathbb{R}^d), \; ||\boldsymbol{\theta}||_{W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)} < 1.$$

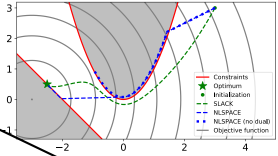
$$J(\Gamma_{\boldsymbol{\theta}}) = J(\Gamma) + \frac{\mathrm{d}J}{\mathrm{d}\boldsymbol{\theta}}(\boldsymbol{\theta}) + o(\boldsymbol{\theta}), \quad \text{with } \frac{|o(\boldsymbol{\theta})|}{||\boldsymbol{\theta}||_{W^{1,\infty}(D,\mathbb{R}^d)}} \xrightarrow{\boldsymbol{\theta}\to 0} 0.$$

Ingredients

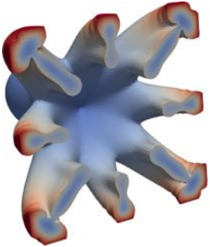
I. Shape derivatives and mesh evolution method



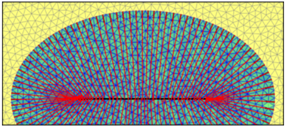
II. Null space optimization algorithm



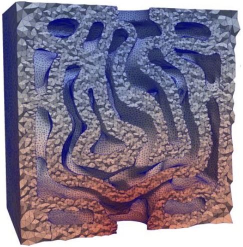
III. Parallel computing



IV. Treatment of geometric constraints

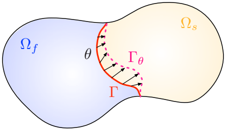


V. Heat exchangers

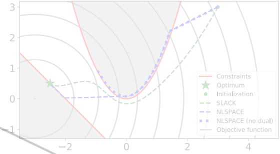


Ingredients

I. Shape derivatives and mesh evolution method



II. Null space optimization algorithm



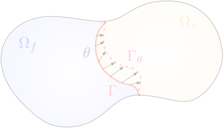
III. Parallel computing

IV. Treatment of geometric constraints

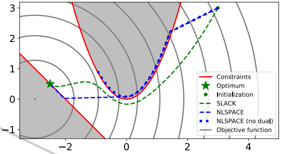
V. Heat exchangers

Ingredients

I. Shape derivatives and mesh evolution method



II. Null space optimization algorithm



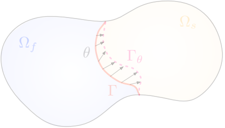
III. Parallel computing

IV. Treatment of geometric constraints

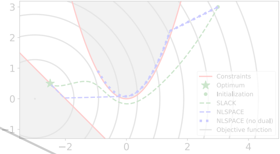
V. Heat exchangers

Ingredients

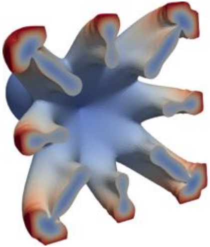
I. Shape derivatives and mesh evolution method



II. Null space optimization algorithm



III. Parallel computing

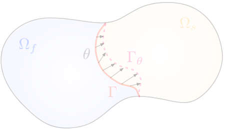


IV. Treatment of geometric constraints

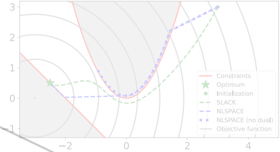
V. Heat exchangers

Ingredients

I. Shape derivatives and mesh evolution method

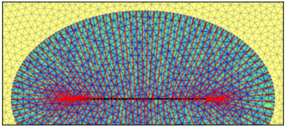


II. Null space optimization algorithm



III. Parallel computing

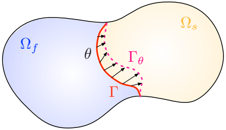
IV. Treatment of geometric constraints



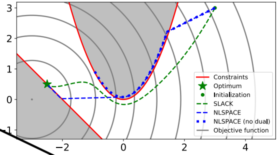
V. Heat exchangers

Ingredients

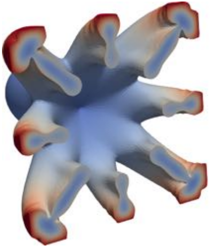
I. Shape derivatives and mesh evolution method



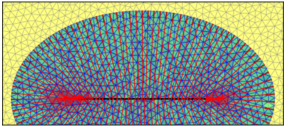
II. Null space optimization algorithm



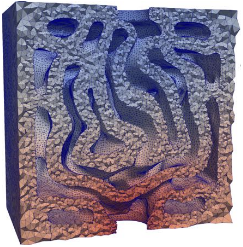
III. Parallel computing



IV. Treatment of geometric constraints

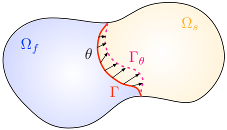


V. Heat exchangers

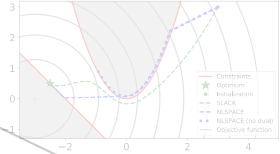


Ingredients

I. Shape derivatives and mesh evolution method



II. Null space optimization algorithm

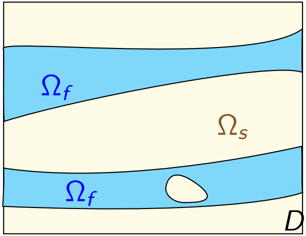


III. Parallel computing

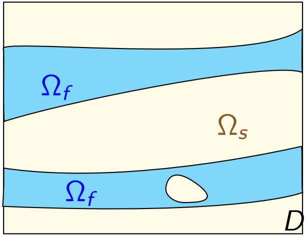
IV. Treatment of geometric constraints

V. Heat exchangers

Coupled physics system



Coupled physics system



- Incompressible Navier-Stokes system for the velocity and pressure $(\boldsymbol{v}, p)$ in Ω_f

$$-\mathrm{div}(\sigma_f(\boldsymbol{v}, p)) + \rho \nabla \boldsymbol{v}\, \boldsymbol{v} = \boldsymbol{f}_f \text{ in } \Omega_f$$

- Convection-diffusion for the temperature T in Ω_f and Ω_s:

$$\begin{aligned} -\mathrm{div}(k_f \nabla T_f) + \rho \boldsymbol{v} \cdot \nabla T_f = Q_f \quad &\text{in } \Omega_f \\ -\mathrm{div}(k_s \nabla T_s) = Q_s \quad &\text{in } \Omega_s \end{aligned}$$

- Boundary conditions on $\Gamma = \partial\Omega_f$:

$$\begin{cases} T_f = T_s \text{ on } \Gamma \\ k_f \nabla T_f \cdot \boldsymbol{n} = k_s \nabla T_s \cdot \boldsymbol{n} \text{ on } \Gamma \\ \boldsymbol{v} = 0 \text{ on } \Gamma. \end{cases}$$

Shape derivatives

Proposition

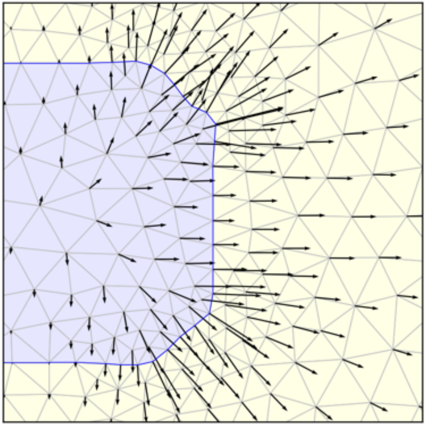
Let $J(\Gamma, T, \boldsymbol{v}, p)$ an arbitrary cost function. If J has continuous partial derivatives, then $\Gamma \mapsto J(\Gamma, \boldsymbol{u}(\Gamma), T(\Gamma), \boldsymbol{v}(\Gamma), p(\Gamma))$ is shape differentiable and the shape derivative reads[1]*:*

$$
\begin{aligned}
&\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\theta}}\Big[J(\Gamma_{\boldsymbol{\theta}}, \boldsymbol{v}(\Gamma_{\boldsymbol{\theta}}), p(\Gamma_{\boldsymbol{\theta}}), T(\Gamma_{\boldsymbol{\theta}}), \boldsymbol{u}(\Gamma_{\boldsymbol{\theta}}))\Big](\boldsymbol{\theta}) \\
&\quad = \overline{\frac{\partial \mathfrak{J}}{\partial \boldsymbol{\theta}}}(\boldsymbol{\theta}) + \int_\Gamma (\boldsymbol{f}_f \cdot \boldsymbol{w} - \sigma_f(\boldsymbol{v}, p) : \nabla \boldsymbol{w} + \boldsymbol{n} \cdot \sigma_f(\boldsymbol{w}, q) \nabla \boldsymbol{v} \cdot \boldsymbol{n} + \boldsymbol{n} \cdot \sigma_f(\boldsymbol{v}, p) \nabla \boldsymbol{w} \cdot \boldsymbol{n})(\boldsymbol{\theta} \cdot \boldsymbol{n}) \mathrm{d}s \\
&\quad + \int_\Gamma \left(k_s \nabla T_s \cdot \nabla S_s - k_f \nabla T_f \cdot \nabla S_f + Q_f S_f - Q_s S_s - 2k_s \frac{\partial T_s}{\partial \boldsymbol{n}} \frac{\partial S_s}{\partial \boldsymbol{n}} + 2k_f \frac{\partial T_f}{\partial \boldsymbol{n}} \frac{\partial S_f}{\partial \boldsymbol{n}} \right) (\boldsymbol{\theta} \cdot \boldsymbol{n}) \mathrm{d}s
\end{aligned}
$$

[1] Feppon et al., *Shape optimization of a coupled thermal fluid–structure problem in a level set mesh evolution framework* (2019)

Body-fitted meshes

We rely on body fitted meshes[2,3].



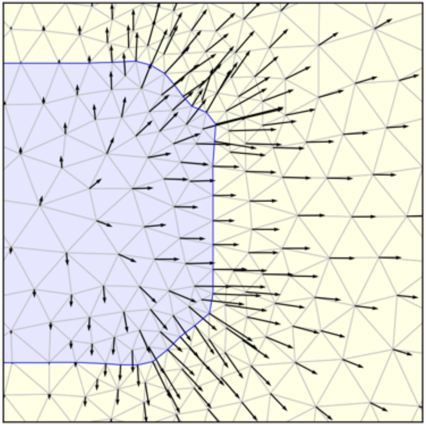
[2] Allaire, Dapogny, and Frey, *Shape optimization with a level set based mesh evolution method* (2014)

[3] Feppon et al., *Shape optimization of a coupled thermal fluid–structure problem in a level set mesh evolution framework* (2019)

Body-fitted meshes

We rely on body fitted meshes[2,3].

- Fluid-Solid interface Γ exactly captured, no need of physics interpolation because no porous regions.



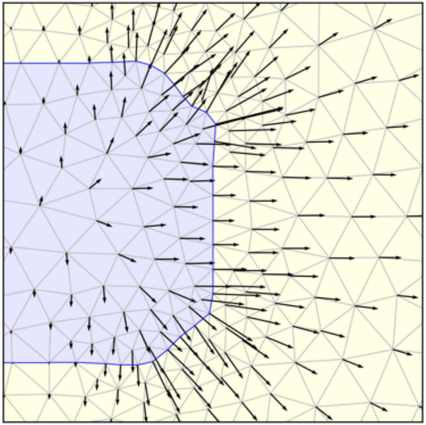
[2] Allaire, Dapogny, and Frey, *Shape optimization with a level set based mesh evolution method* (2014)

[3] Feppon et al., *Shape optimization of a coupled thermal fluid–structure problem in a level set mesh evolution framework* (2019)

Body-fitted meshes

We rely on body fitted meshes[2,3].

- Fluid-Solid interface Γ exactly captured, no need of physics interpolation because no porous regions.
- Remeshing with `Mmg` enabling mesh size control.

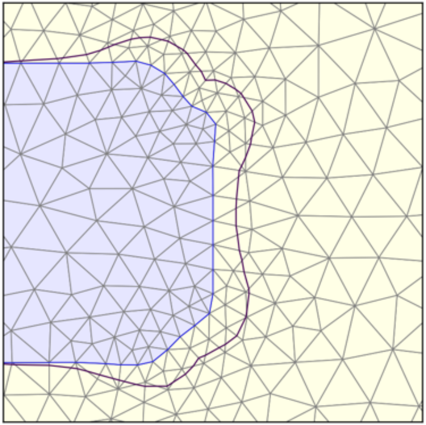


[2] Allaire, Dapogny, and Frey, *Shape optimization with a level set based mesh evolution method* (2014)

[3] Feppon et al., *Shape optimization of a coupled thermal fluid–structure problem in a level set mesh evolution framework* (2019)

Body-fitted meshes

We rely on body fitted meshes[2,3].



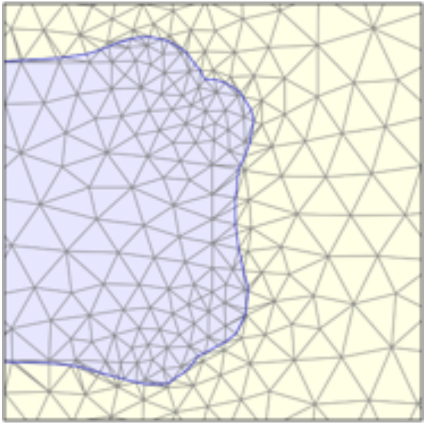
- Fluid-Solid interface Γ exactly captured, no need of physics interpolation because no porous regions.
- Remeshing with `Mmg` enabling mesh size control.

[2] Allaire, Dapogny, and Frey, *Shape optimization with a level set based mesh evolution method* (2014)

[3] Feppon et al., *Shape optimization of a coupled thermal fluid–structure problem in a level set mesh evolution framework* (2019)

Body-fitted meshes

We rely on body fitted meshes[2,3].



- Fluid-Solid interface Γ exactly captured, no need of physics interpolation because no porous regions.
- Remeshing with `Mmg` enabling mesh size control.

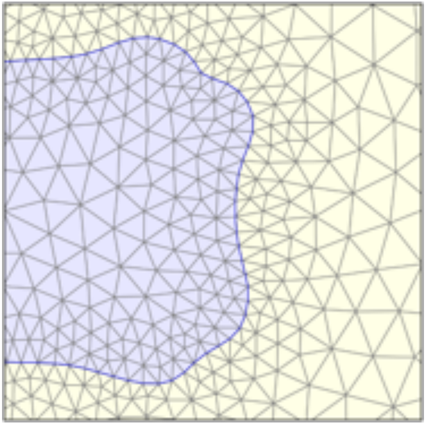
---

[2] Allaire, Dapogny, and Frey, *Shape optimization with a level set based mesh evolution method* (2014)

[3] Feppon et al., *Shape optimization of a coupled thermal fluid–structure problem in a level set mesh evolution framework* (2019)

Body-fitted meshes

We rely on body fitted meshes[2,3].



- Fluid-Solid interface Γ exactly captured, no need of physics interpolation because no porous regions.
- Remeshing with `Mmg` enabling mesh size control.

---

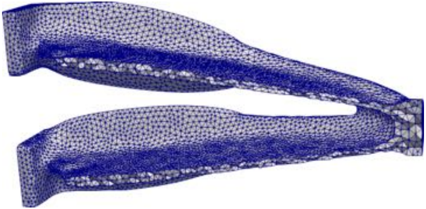
[2] Allaire, Dapogny, and Frey, *Shape optimization with a level set based mesh evolution method* (2014)

[3] Feppon et al., *Shape optimization of a coupled thermal fluid–structure problem in a level set mesh evolution framework* (2019)

Body-fitted meshes

We rely on body fitted meshes[2,3].

- Fluid-Solid interface Γ exactly captured, no need of physics interpolation because no porous regions.
- Remeshing with `Mmg` enabling mesh size control.

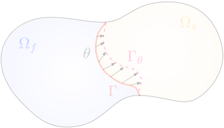


[2] Allaire, Dapogny, and Frey, *Shape optimization with a level set based mesh evolution method* (2014)

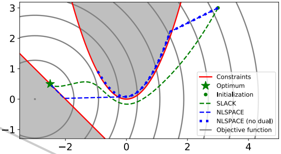
[3] Feppon et al., *Shape optimization of a coupled thermal fluid–structure problem in a level set mesh evolution framework* (2019)

Ingredients

I. Shape derivatives and mesh evolution method



II. Null space optimization algorithm



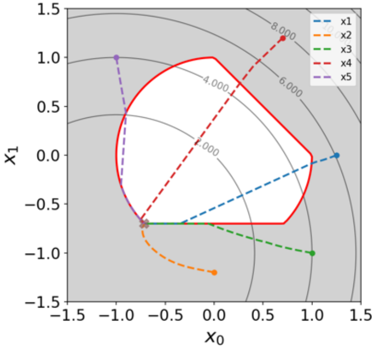
III. Parallel computing

IV. Treatment of geometric constraints

V. Heat exchangers

Null space optimization algorithm

- Nonlinear constrained optimization on manifolds with a moderate number of constraints



Open source implementation[4]:

`https://gitlab.com/florian.feppon/null-space-optimizer`

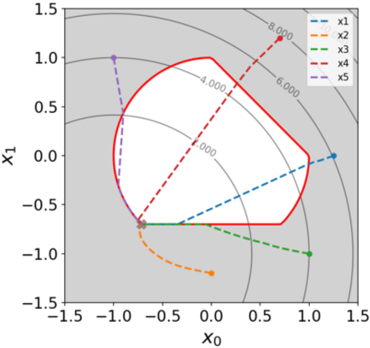
`pip install nullspace_optimizer`

---

[4] Feppon, Allaire, and Dapogny, *Null space gradient flows for constrained optimization with applications to shape optimization* (2019)

Null space optimization algorithm

- Nonlinear constrained optimization on manifolds with a moderate number of constraints
- Generalization of the unconstrained gradient flow: no hard tuning of parameters

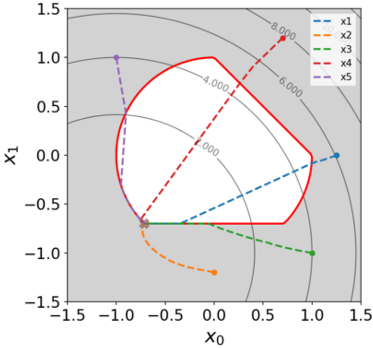


Open source implementation[4]:
`https://gitlab.com/florian.feppon/null-space-optimizer`
`pip install nullspace_optimizer`

[4] Feppon, Allaire, and Dapogny, *Null space gradient flows for constrained optimization with applications to shape optimization* (2019)

Null space optimization algorithm

- Nonlinear constrained optimization on manifolds with a moderate number of constraints
- Generalization of the unconstrained gradient flow: no hard tuning of parameters
- Adapted to the infinite dimensional setting of the method of Hadamard

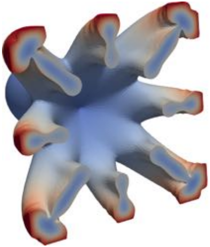


Open source implementation[4]:
`https://gitlab.com/florian.feppon/null-space-optimizer`
`pip install nullspace_optimizer`

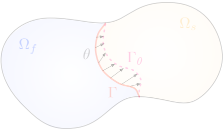
[4] Feppon, Allaire, and Dapogny, *Null space gradient flows for constrained optimization with applications to shape optimization* (2019)

Ingredients

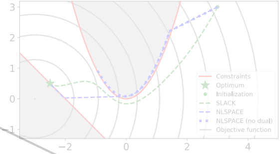
III. Parallel computing



I. Shape derivatives and mesh evolution method



II. Null space optimization algorithm

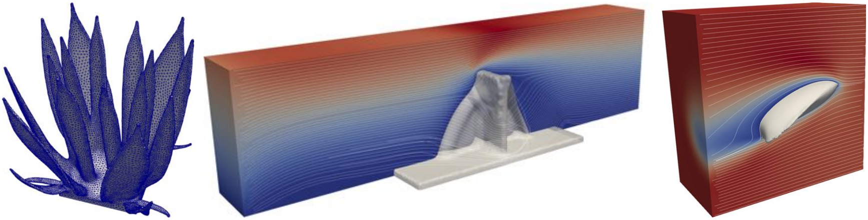


V. Heat exchangers

IV. Treatment of geometric constraints

Parallel computing

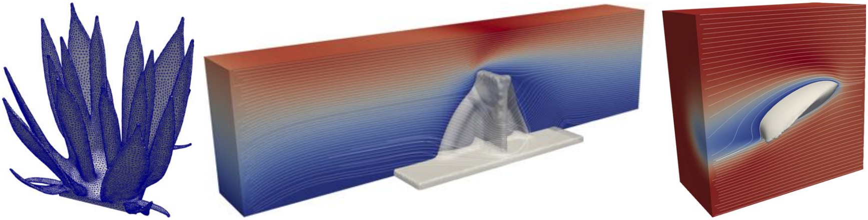
- Use of Domain Decomposition and adapted preconditioners for solving finite element problems : all FEM related operations are achieved in parallel[5].



[5] Feppon et al., *Topology optimization of thermal fluid–structure systems using body-fitted meshes and parallel computing* (2020)

Parallel computing

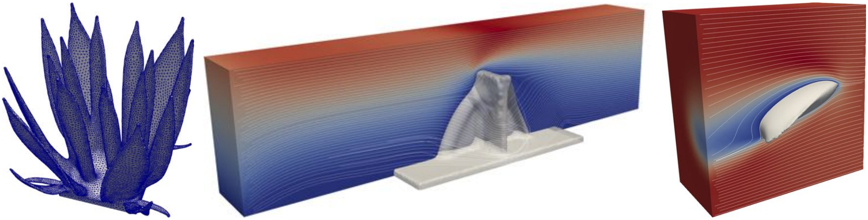
- Use of Domain Decomposition and adapted preconditioners for solving finite element problems : all FEM related operations are achieved in parallel[5].
- We solve fluid FEM problems on meshes up to 4.8 millions of Tetrahedra with 30 CPUs.



[5] Feppon et al., *Topology optimization of thermal fluid–structure systems using body-fitted meshes and parallel computing* (2020)

Parallel computing

- Use of Domain Decomposition and adapted preconditioners for solving finite element problems : all FEM related operations are achieved in parallel[5].
- We solve fluid FEM problems on meshes up to 4.8 millions of Tetrahedra with 30 CPUs.
- Mesh adaptation and Isosurface discretization is still sequential. A future release of (Par)Mmg will allow to do it in parallel.

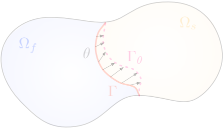


[5] Feppon et al., *Topology optimization of thermal fluid–structure systems using body-fitted meshes and parallel computing* (2020)

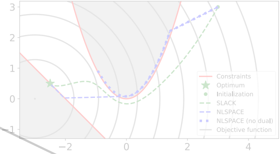
Ingredients

III. Parallel computing

I. Shape derivatives and mesh evolution method

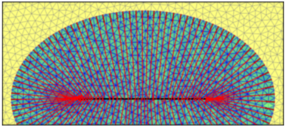


II. Null space optimization algorithm



V. Heat exchangers

IV. Treatment of geometric constraints



Non mixing constraint

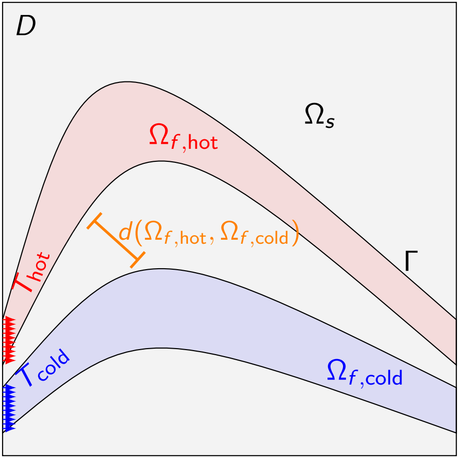


Figure: Settings of the heat exchanger topology optimization problem .

Non-penetration constraint:

$$d(\Omega_{f,\text{hot}}, \Omega_{f,\text{cold}}) \geqslant d_{\text{min}}.$$

Non mixing constraint

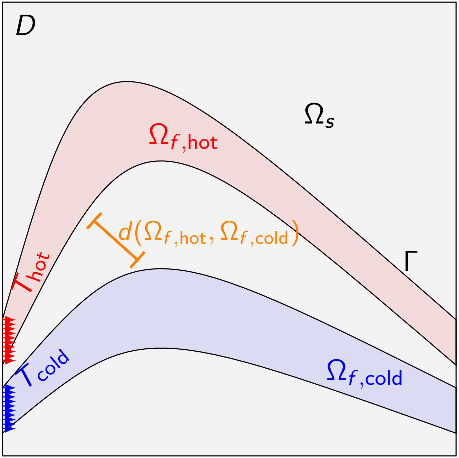


Figure: Settings of the heat exchanger topology optimization problem .

Non-penetration constraint:

$$d(\Omega_{f,\text{hot}}, \Omega_{f,\text{cold}}) \geqslant d_{\min}.$$

We enforce it by imposing

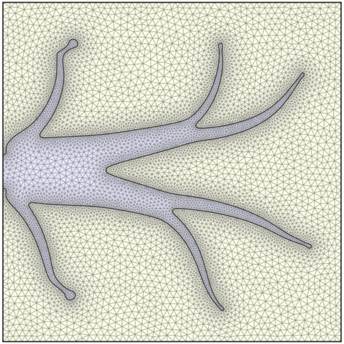
$$\forall x \in \Omega_{f,\text{cold}},\ d_{\Omega_{f,\text{hot}}}(x) \geqslant d_{\min},$$

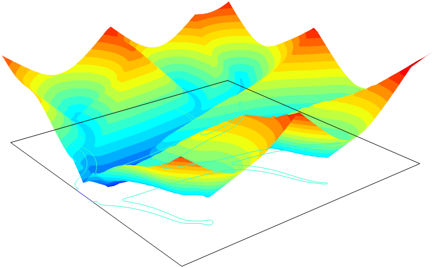
where $d_{\Omega_{f,\text{hot}}}$ is the signed distance function to the domain $\Omega_{f,\text{hot}}$.

The signed distance function

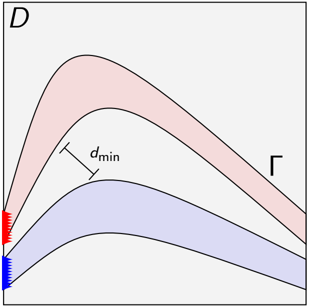
The signed distance function d_Ω to the domain $\Omega \subset D$ is defined by:

$$\forall x \in D, \;\; d_\Omega(x) = \begin{cases} -\min\limits_{y \in \partial\Omega} ||y - x|| & \text{if } x \in \Omega, \\ \min\limits_{y \in \partial\Omega} ||y - x|| & \text{if } x \in D \backslash \Omega. \end{cases}$$





2D heat exchangers

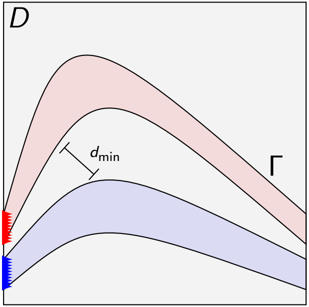


Heat exchanger problem with limited pressure loss and non-mixing constraint:

$$\min_{\Gamma} \quad J(\Omega_f) = -\left(\int_{\Omega_{f,cold}} \rho c_p \boldsymbol{v} \cdot \nabla T \mathrm{d}x - \int_{\Omega_{f,hot}} \rho c_p \boldsymbol{v} \cdot \nabla T \mathrm{d}x \right)$$

$$s.c. \begin{cases} \mathrm{DP}(\Omega_f) = \int_{\partial\Omega_f^D} p \mathrm{d}s - \int_{\partial\Omega_f^N} p \mathrm{d}s \leq \mathrm{DP}_0 \\ \qquad\qquad Q_{hot\leftrightarrow cold}(\Omega_f) \geqslant d_{\min}. \end{cases}$$

2D heat exchangers



Heat exchanger problem with limited pressure loss and non-mixing constraint:

$$\min_{\Gamma} \qquad J(\Omega_f) = -\left(\int_{\Omega_{f,cold}} \rho c_p \boldsymbol{v} \cdot \nabla T \mathrm{d}x - \int_{\Omega_{f,hot}} \rho c_p \boldsymbol{v} \cdot \nabla T \mathrm{d}x\right)$$

$$s.c. \begin{cases} \mathtt{DP}(\Omega_f) = \int_{\partial\Omega_f^D} p \mathrm{d}s - \int_{\partial\Omega_f^N} p \mathrm{d}s \leq \mathtt{DP}_0 \\ Q_{hot\leftrightarrow cold}(\Omega_f) = \int_D j(d_{\Omega_{f,hot}}) \mathrm{d}x \geqslant d_{\min}. \end{cases}$$

Shape derivatives of geometric constraints

For instance

$$Q_{hot\leftrightarrow cold}(\Omega_f) := \left(\int_{\Omega_{f,cold}} \frac{1}{|d_{\Omega_{f,hot}}|^p} \mathrm{d}x \right)^{-\frac{1}{p}} \simeq \left\| \frac{1}{d_{\Omega_{f,hot}}} \right\|^{-1}_{L^\infty(\Omega_{f,cold})}.$$

Shape derivatives of geometric constraints

For instance

$$Q_{hot\leftrightarrow cold}(\Omega_f) := \left(\int_{\Omega_{f,cold}} \frac{1}{|d_{\Omega_{f,hot}}|^p} \mathrm{d}x \right)^{-\frac{1}{p}} \simeq \left\| \frac{1}{d_{\Omega_{f,hot}}} \right\|_{L^\infty(\Omega_{f,cold})}^{-1}.$$

This reduces to the setting of computing the shape derivative of some penalty functional $Q_{hot\leftrightarrow cold}(\Omega_f)$ with:

$$Q_{hot\leftrightarrow cold}(\Omega_f) := \int_D j(d_{\Omega_{f,hot}}) \mathrm{d}x.$$

Shape derivatives of geometric constraints

The shape derivative of $Q_{hot\leftrightarrow cold}(\Omega_f)$ is given by[6]:

$$Q'_{hot\leftrightarrow cold}(\Omega)(\boldsymbol{\theta}) = \int_{\partial\Omega_{f,hot}} u(y)\ \boldsymbol{\theta}\cdot\boldsymbol{n}\,\mathrm{d}y$$

$$\text{with } u(y) = -\int_{z\in\mathrm{ray}(y)} j'(d_{\Omega_{f,hot}}(z)) \prod_{1\leq i\leq n-1}(1+\kappa_i(y)d_{\Omega_{f,hot}}(z))\mathrm{d}z, \qquad \forall y\in\partial\Omega.$$

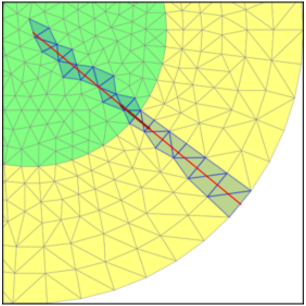
[6] Allaire, Jouve, and Michailidis, *Thickness control in structural optimization via a level set method* (2016)

Shape derivatives of geometric constraints

The shape derivative of $Q_{hot\leftrightarrow cold}(\Omega_f)$ is given by[6]:

$$Q'_{hot\leftrightarrow cold}(\Omega)(\boldsymbol{\theta}) = \int_{\partial\Omega_{f,hot}} u(y)\, \boldsymbol{\theta}\cdot\boldsymbol{n}\,\mathrm{d}y$$

$$\text{with } u(y) = -\int_{z\in\mathrm{ray}(y)} j'(d_{\Omega_{f,hot}}(z)) \prod_{1\leq i\leq n-1} (1+\kappa_i(y)d_{\Omega_{f,hot}}(z))\mathrm{d}z, \qquad \forall y\in\partial\Omega.$$



The computation of $u(y)$ requires a priori integration along the normal rays and the computation of curvatures $\kappa_i(y)$.

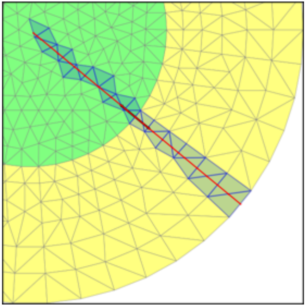
[6] Allaire, Jouve, and Michailidis, *Thickness control in structural optimization via a level set method* (2016)

Shape derivatives of geometric constraints

The shape derivative of $Q_{hot \leftrightarrow cold}(\Omega_f)$ is given by[6]:

$$Q'_{hot \leftrightarrow cold}(\Omega)(\boldsymbol{\theta}) = \int_{\partial \Omega_{f,hot}} u(y)\, \boldsymbol{\theta} \cdot \boldsymbol{n}\, \mathrm{d}y$$

$$\text{with } u(y) = -\int_{z \in \mathrm{ray}(y)} j'(d_{\Omega_{f,hot}}(z)) \prod_{1 \leq i \leq n-1} (1 + \kappa_i(y) d_{\Omega_{f,hot}}(z)) \mathrm{d}z, \qquad \forall y \in \partial \Omega.$$



The computation of $u(y)$ requires a priori integration along the normal rays and the computation of curvatures $\kappa_i(y)$.

[6] Allaire, Jouve, and Michailidis, *Thickness control in structural optimization via a level set method* (2016)

Shape derivatives of geometric constraints

It turns out that it is possible to compute u without integrating along the rays[7]:

[7] Feppon, Allaire, and Dapogny, *A variational formulation for computing shape derivatives of geometric constraints along rays* (2019)

Shape derivatives of geometric constraints

It turns out that it is possible to compute u without integrating along the rays[7]:

Proposition

Let $\hat{u} \in V_\omega$ be the solution to the variational problem

$$\forall v \in V_\omega,\ \int_{\partial\Omega_{f,hot}} \hat{u} v \mathrm{d}s + \int_D \omega(\nabla d_{\Omega_{f,hot}} \cdot \nabla \hat{u})(\nabla d_{\Omega_{f,hot}} \cdot \nabla v)\mathrm{d}x = -\int_D j'(d_{\Omega_{f,hot}}) v \mathrm{d}x.$$

Then $u(y) = \hat{u}(y)$ for any $y \in \partial\Omega_{f,hot}$.

[7] Feppon, Allaire, and Dapogny, *A variational formulation for computing shape derivatives of geometric constraints along rays* (2019)

Shape derivatives of geometric constraints

It turns out that it is possible to compute u without integrating along the rays[7]:

Proposition

Let $\hat{u} \in V_\omega$ be the solution to the variational problem

$$\forall v \in V_\omega,\ \int_{\partial\Omega_{f,hot}} \hat{u} v \mathrm{d}s + \int_D \omega(\nabla d_{\Omega_{f,hot}} \cdot \nabla \hat{u})(\nabla d_{\Omega_{f,hot}} \cdot \nabla v)\mathrm{d}x = -\int_D j'(d_{\Omega_{f,hot}}) v \mathrm{d}x.$$

Then $u(y) = \hat{u}(y)$ for any $y \in \partial\Omega_{f,hot}$.

- This variational problem can easily be solved with FEM in 2D and 3D

[7]Feppon, Allaire, and Dapogny, *A variational formulation for computing shape derivatives of geometric constraints along rays* (2019)

Shape derivatives of geometric constraints

It turns out that it is possible to compute u without integrating along the rays[7]:

Proposition

Let $\hat{u} \in V_\omega$ be the solution to the variational problem

$$\forall v \in V_\omega,\ \int_{\partial\Omega_{f,hot}} \hat{u} v \mathrm{d}s + \int_D \omega(\nabla d_{\Omega_{f,hot}} \cdot \nabla \hat{u})(\nabla d_{\Omega_{f,hot}} \cdot \nabla v)\mathrm{d}x = -\int_D j'(d_{\Omega_{f,hot}}) v \mathrm{d}x.$$

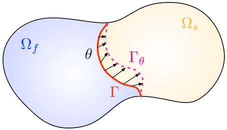
Then $u(y) = \hat{u}(y)$ for any $y \in \partial\Omega_{f,hot}$.

- This variational problem can easily be solved with FEM in 2D and 3D
- This allows to handle conveniently geometric constraints (e.g. maximum thickness, minum distance, etc...) in 2D and 3D level set based topology optimization.

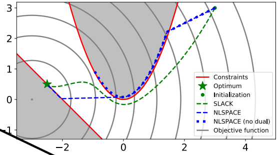
[7] Feppon, Allaire, and Dapogny, *A variational formulation for computing shape derivatives of geometric constraints along rays* (2019)

Ingredients

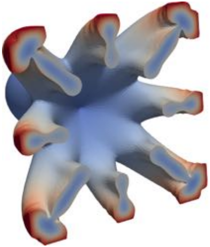
I. Shape derivatives and mesh evolution method



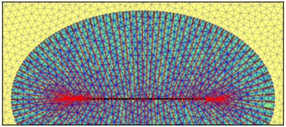
II. Null space optimization algorithm



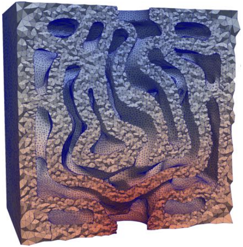
III. Parallel computing



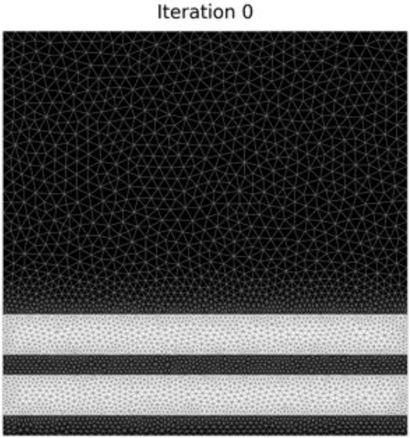
IV. Treatment of geometric constraints



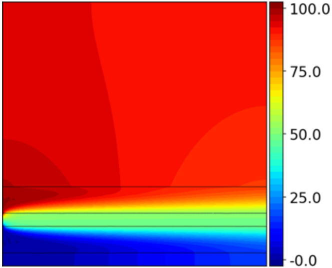
V. Heat exchangers



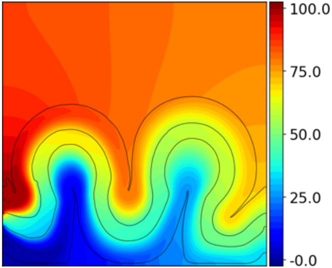
Ingredients



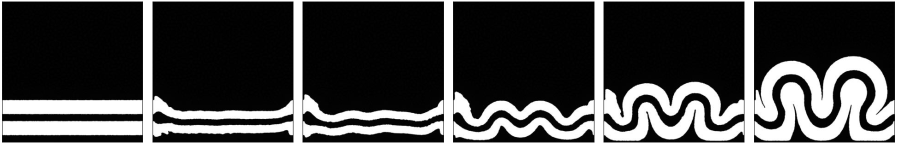
2D Heat Exchangers with non-mixing constraint



(a) Initial temperature



(b) Final temperature.



(c) Intermediate iterations 0, 8, 20, 50, 88 et 200.

3D fluid-to-fluid heat exchanger

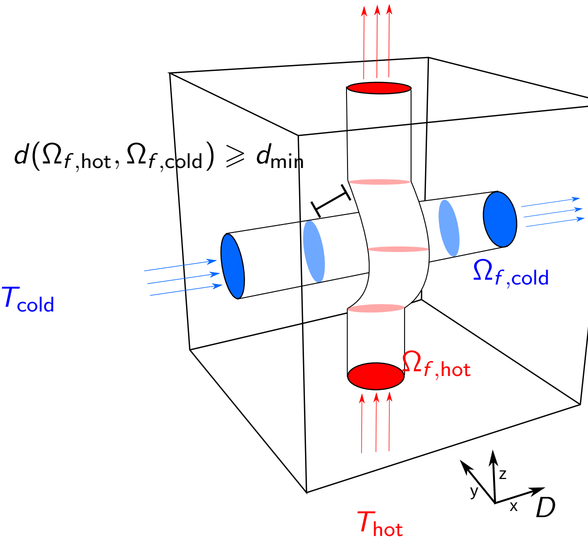


Figure: Schematic of the 3D setting.

3D fluid-to-fluid heat exchanger

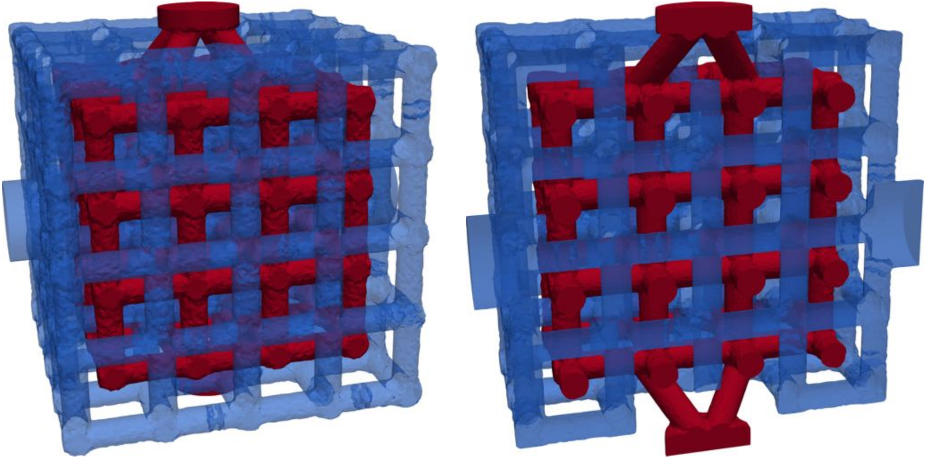
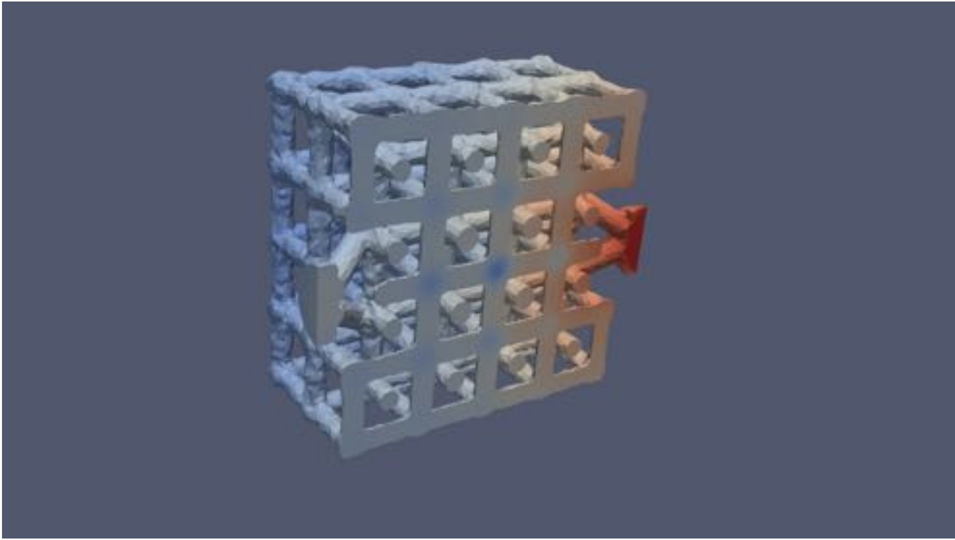
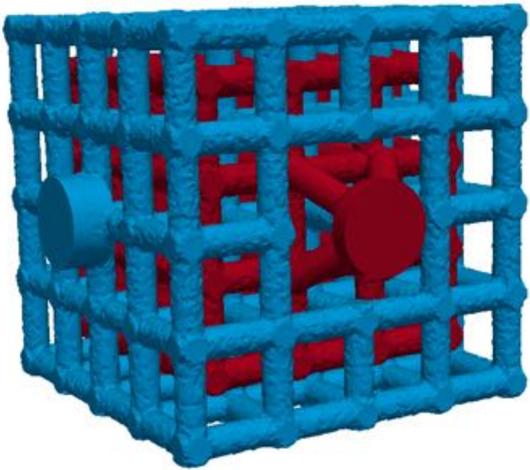


Figure: Initial distribution of fluid considered for the 3D heat exchanger test case.

3D fluid-to-fluid heat exchanger



3D fluid-to-fluid heat exchanger



3D fluid-to-fluid heat exchanger

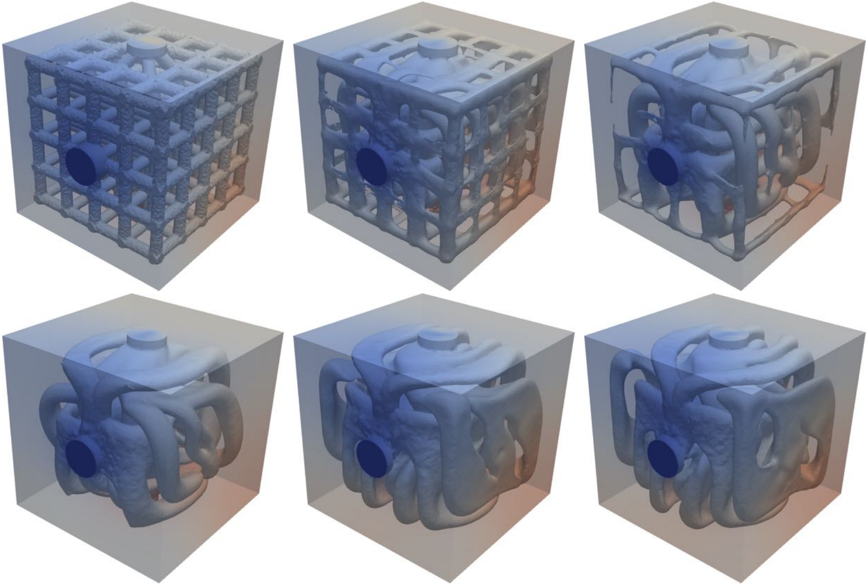
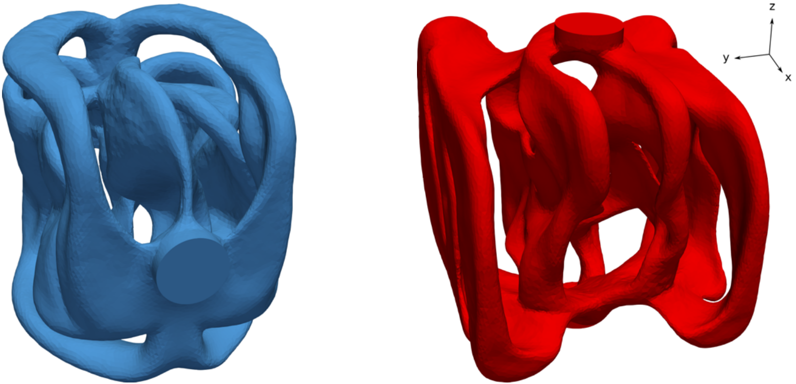


Figure: Intermediate iterations.

3D fluid-to-fluid heat exchanger



(a) Cold phase (b) Hot phase

Figure: Separate plots of the topologically optimized cold and hot fluid phases in the configuration $d_{\min} = 0.04$.

3D fluid-to-fluid heat exchanger

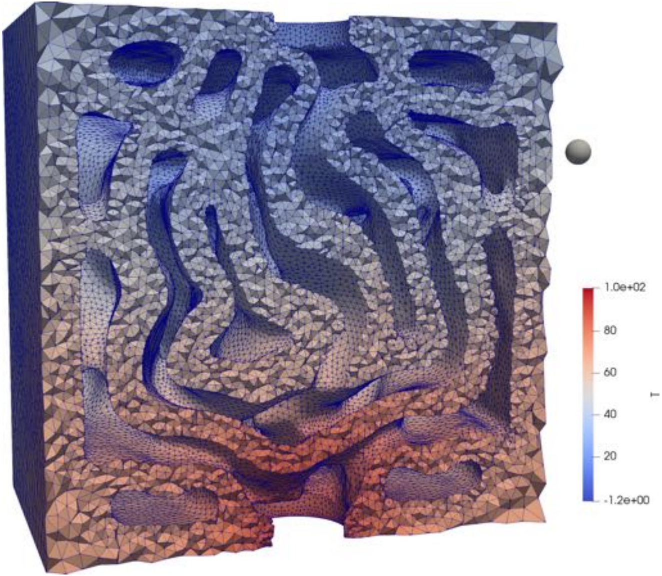


Figure: Cut of the resulting solid domain

Many thanks for your attention!